

Ein Plotter-Skript für PsTricks

Xenia Rendtel

12. Januar 2012

Inhaltsverzeichnis

1. Vorwort	17
1.1. Version 1	17
1.2. Version 1.5	17
2. Aufbau des Skriptes	19
3. Allgemeine Einstellungen	21
3.1. Skalierung und Größe des Bildes	21
3.2. Grenzen des Bildes	21
3.3. Schriftgröße eines Bildes	22
3.4. Allgemeine Einstellungen eines Bildes	22
4. Einfache Zeichenroutinen	23
4.1. Linien, Vektoren und Polygone	23
4.1.1. Einfache Linien zeichnen	23
4.1.2. Vektoren zeichnen	25
4.1.3. Polygonzüge zeichnen	25
4.1.4. Regelmäßiges n -Eck	26
4.2. Dreiecke, Rechtecke und Vierecke zeichnen	27
4.2.1. Dreiecke	27
4.2.2. Rechteck	28
4.2.3. Viereck	28
4.2.4. Parallelogramme zeichnen	29
4.3. Kreise, Ellipsen und Kurven	30
4.3.1. Kreise und Kreisbögen	30
4.3.2. Weitere Kreisbefehle	31
4.3.3. Ringausschnitte	31
4.3.4. Ellipsen zeichnen	32
4.3.5. Kurven zeichnen	33
4.4. Beschriftungen, Punkte und Kreuze zeichnen	34
4.4.1. Beschriftungen	34
4.4.2. Punkte zeichnen	35
4.4.3. Kreuze zeichnen	36
4.5. Grafiken einfügen	38
4.6. Weitere Zeichenelemente	38
4.6.1. Winkel und Winkelbezeichnungen	38
4.6.2. Knoten	39
4.6.3. Neue Seite	40
5. Koordinatensysteme und Gitter	41
5.1. Einfache Koordinatensysteme	41
5.1.1. Einige Beispiele	42
5.2. Weitere Achsentypen	42
5.3. Erweiterung der Achsen mit Teilstrichen und Pfeilarten	43

5.4.	Trigonometrische Achsenbeschriftungen	43
5.4.1.	Beispiele	44
5.5.	Achsenbeschriftung	44
5.5.1.	Punkte an den Achsen	44
5.5.2.	Pfeile an den Achsen	46
5.5.3.	Pfeile an den Achsen mit Kästen	46
5.6.	Gitter zeichnen	47
5.6.1.	Beispiele	48
5.7.	Matheheft und Millimeterpapier	48
5.8.	Grids	49
6.	Plotten von Funktionen und Daten	51
6.1.	Funktionen zeichnen	51
6.2.	Markierungen an Graphen	51
6.3.	Ableitungen zeichnen	52
6.4.	Tangenten und Tangendendreiecke	53
6.4.1.	Zeichnen einer Tangente	53
6.4.2.	Zeichnen eines Tangendendreiecks	53
6.5.	Sekanten und Sekantendreiecke	54
6.5.1.	Zeichnen von Sekanten	54
6.5.2.	Zeichnen eines Sekantendreiecks	54
6.6.	Stammfunktionen und Integrale	55
6.6.1.	Stammfunktionen	55
6.6.2.	Flächen unter einer Funktion	56
6.6.3.	Einschluß zwischen zwei Funktionen	57
6.7.	Rotationskörper um die x -Achse	58
6.8.	Parametrisierte Kurven	58
6.9.	Zeichnen von Funktionsfolgen	59
6.10.	Plotten von Daten	59
7.	Diagramme	63
7.1.	Säulen- und Balkendiagramme	63
7.1.1.	Säulendiagramme	63
7.1.2.	Balkendiagramme	66
7.1.3.	Gestapelte Säulen- und Balkendiagramme	67
7.1.4.	Weitere Säulen- und Balkendiagrammtypen	68
7.2.	Kreis- und Ringdiagramme	70
7.2.1.	Torten- oder Kreisdiagramm	70
7.2.2.	Ringdiagramm	71
7.3.	Piktogramme und Strichlisten	72
7.3.1.	Piktogramm	72
7.3.2.	Eine Strichliste	73
7.4.	Eine Legende	74
7.5.	Die Binomialverteilung	75
7.5.1.	Die Binomialverteilung als Säulendiagramm	75
7.5.2.	Die Binomialverteilung als Strichdiagramm	76
8.	Dreidimensionale Abbildungen	79
8.1.	Die Parallelprojektion	79
8.2.	Zeichenbefehle für den $\mathbb{R}^3$	82

8.2.1.	Linien, Vektoren und Polygone	82
8.2.2.	Dreiecke und Vierecke im $\mathbb{R}^3$ zeichnen	85
8.2.3.	Ellipsen und Kreise	86
8.2.4.	Kurven zeichnen	87
8.2.5.	Beschriftungen, Punkte und Kreuze im $\mathbb{R}^3$ zeichnen	88
8.2.6.	Grafiken einfügen	90
8.3.	Koordinatensysteme und Gitter im $\mathbb{R}^3$	90
8.3.1.	Einfache Koordinatensysteme	90
8.3.2.	Gitter zeichnen im $\mathbb{R}^3$	91
8.4.	Plotten von Funktionen und Daten im $\mathbb{R}^3$	91
8.4.1.	Funktionen zeichnen im $\mathbb{R}^3$	91
8.4.2.	Parametrisierte Kurven im $\mathbb{R}^3$	93
8.4.3.	Plotten von Daten im $\mathbb{R}^3$	94
8.5.	Objekte für die analytische Geometrie	94
8.5.1.	Geraden im Raum	95
8.5.2.	Ebenengleichungen	95
8.5.3.	Winkel im $\mathbb{R}^3$	98
8.5.4.	Kugeln	99
8.6.	Körper im $\mathbb{R}^3$	99
8.6.1.	Ein Quader	99
8.6.2.	Eine Pyramide und ein Pyramidenstumpf	100
8.6.3.	Ein Kreiszylinder	100
8.6.4.	Ein Kegel und ein Kegelstumpf	101
9.	Spielfeld	103
9.1.	Uhrdarstellung	103
9.2.	Regression	103
A.	Die Quell-Dateien	107
A.1.	Plotter.pl	107
A.2.	Das Modul für Koordinatensysteme	239
A.3.	Modul zur Erzeugung von Dateien	255
A.4.	Das Modul für Diagramme	271
A.5.	Das Modul für dreidimensionale Zeichnungen	281
A.6.	Das Modul für Funktionen	322
A.7.	Das Modul für den Funktionsparser	332
A.8.	Das Modul für Gitter	347
A.9.	Das Modul für Graphen	352
A.10.	Das Modul für Tupel	370
A.11.	Das Modul für Zeichenroutinen	372
A.12.	Das Modul für Grafiken	400
A.13.	Die Spielfeld-Module	402

Abbildungsverzeichnis

3.1. Der Befehl <code>grenzengitter</code>	21
4.1. Der Befehl <code>linie</code>	23
4.2. Der Befehl <code>linie</code> mit Text	23
4.3. Der Befehl <code>linie</code> mit zusätzlichem Abstand	23
4.4. Der Befehl <code>linie</code> mit mehreren Wertepaaren	24
4.5. Der Befehl <code>linie</code> erweitert	24
4.6. Der Befehl <code>linie</code> mit Text	24
4.7. Der Befehl <code>linie</code> mit zusätzlichem Abstand	24
4.8. Eine Linie wird über seinen Startpunkt und die Verschiebung definiert.	25
4.9. Der Befehl <code>vlinie</code>	25
4.10. Der Befehl <code>vektor</code>	25
4.11. Der Befehl <code>polygon</code>	26
4.12. Ein regelmäßiges n -Eck	26
4.13. Der Befehl <code>neck</code>	26
4.14. Der Befehl <code>dreieck</code>	27
4.15. Ein Dreieck wird über zwei Kanten und einen Winkel definiert	27
4.16. Der Befehl <code>dreieck</code>	27
4.17. Ein rechtwinkliges Dreieck wird über zwei Punkte und einen weiteren Winkel definiert	28
4.18. Der Befehl <code>dreieck</code>	28
4.19. Der Befehl <code>rechteck</code>	28
4.20. Der Befehl <code>viereck</code>	29
4.21. Ein Parallelogramm über zwei Kanten und zwei Winkel definiert	29
4.22. Der Befehl <code>parallelogramm</code>	29
4.23. Ein Kreis wird gezeichnet	30
4.24. Ein Kreis mit Text	30
4.25. Ein Kreisbogen	30
4.26. Ein geschlossener Kreisbogen	31
4.27. Ein Kreisbogen mit Pfeil	31
4.28. Die Definition des Befehls <code>kreissskala</code>	31
4.29. Der Befehl <code>kreissskala</code>	31
4.30. Ein Ringausschnitt	32
4.31. Eine Ellipse	32
4.32. Ein Ellipsenbogen	32
4.33. Ein geschlossener Ellipsenbogen	33
4.34. Ein Ellipsenbogen mit Pfeil	33
4.35. Eine Kurve	33
4.36. Eine Bezierkurve	34
4.37. Der Befehl <code>beschriftung</code>	34
4.38. Der Befehl <code>beschriftungsgroesse</code>	35
4.39. Der Befehl <code>beschriftung</code>	35
4.40. Der Befehl <code>punkt</code>	36
4.41. Der Befehl <code>punkt</code> mit einem Funktionsaufruf	36

4.42. Die Definition des Befehls <code>kreuz</code>	36
4.43. Der Befehl <code>kreuz</code>	37
4.44. Der Befehl <code>kreuz</code> an dem Graphen $f(x) = x^2$	37
4.45. Eine Grafik einfügen	38
4.46. Ein Winkel	38
4.47. Ein weiterer Winkel	39
4.48. Der Befehl <code>knoten</code>	39
4.49. Erstes Beispiel für den Befehl <code>knoten</code>	39
4.50. Zweites Beispiel für den Befehl <code>knoten</code>	40
4.51. Seitenwechsel, erste Seite	40
4.52. Seitenwechsel, zweite Seite	40
5.1. Bezugspunkte beim Zeichnen von Koordinatenachsen	41
5.2. Ein erstes Koordinatensystem	42
5.3. Ein zweites Koordinatensystem	42
5.4. Ein Koordinatensystem mit dem Befehl <code>mitte</code>	42
5.5. Eine x -Achse	43
5.6. Eine y -Achse	43
5.7. Eine x -Achse mit weiteren Teilstrichen	43
5.8. Eine x -Achse ohne Pfeile	43
5.9. Ein trigonometrisches Koordinatensystem	44
5.10. Der Befehl <code>deg</code>	44
5.11. Der Befehl <code>xachserad</code>	44
5.12. Der Befehl <code>xachsenpunkt</code>	45
5.13. Der Befehl <code>yachsenpunkt</code>	45
5.14. Der Befehl <code>xachsenpunktohne</code>	45
5.15. Der Befehl <code>yachsenpunktohne</code>	45
5.16. Der Befehl <code>xpfeil</code>	46
5.17. Der Befehl <code>ypfeil</code>	46
5.18. Der Befehl <code>xpfeilkasten</code>	47
5.19. Der Befehl <code>ypfeilkasten</code>	47
5.20. Der Befehl <code>kariert</code>	48
5.21. Ein weiteres Gitter	48
5.22. Der Befehl <code>gitter</code>	48
5.23. Der Befehl <code>matheheft</code>	48
5.24. Der Befehl <code>matheheft</code>	49
5.25. Der Befehl <code>millimeterpapier</code>	49
5.26. Der Befehl <code>grid</code>	49
5.27. Der Befehl <code>punkte</code>	49
6.1. Der Befehl <code>graph</code>	51
6.2. Der Befehl <code>graphmarkierung</code>	52
6.3. Der Befehl <code>ableitung</code>	52
6.4. Der Befehl <code>tangente</code>	53
6.5. Der Befehl <code>tangentendreieck</code>	54
6.6. Der Befehl <code>sekante</code>	54
6.7. Der Befehl <code>sekantendreieck</code>	55
6.8. Der Befehl <code>stammfunktion</code>	55
6.9. Die Obersumme von $f(x) = x^2$	56
6.10. Die Untersumme von $f(x) = x^2$	56

6.11. Der Befehl <code>integral</code>	57
6.12. Der Befehl <code>einschluss</code>	57
6.13. Der Befehl <code>graphrot</code>	58
6.14. Der Befehl <code>graphrot</code> mit feineren Einteilung	58
6.15. Der Befehl <code>parametricplot</code>	59
6.16. Der Befehl <code>folge</code>	59
6.17. Der Befehl <code>listplot</code>	60
6.18. Der Befehl <code>fileplot</code>	60
6.19. Der Befehl <code>dataplot</code>	60
6.20. Der Befehl <code>barplot</code>	61
7.1. Die Definition des Säulendiagramms	64
7.2. Die zweite Definition des Säulendiagramms	64
7.3. Säulendiagramm	65
7.4. Säulendiagramm Nr. 2	65
7.5. Säulendiagramm Nr. 3	65
7.6. Balkendiagramm	66
7.7. Balkendiagramm Nr. 2	67
7.8. Balkendiagramm Nr. 2	67
7.9. Gestapeltes Balkendiagramm	67
7.10. Gestapeltes Säulendiagramm	68
7.11. Balkendiagramm in Dreiecksform Nr. 1	68
7.12. Balkendiagramm in Dreiecksform Nr. 2	69
7.13. Balkendiagramm in Dreiecksform Nr. 3	69
7.14. Säulendiagramm in Dreiecksform Nr. 1	69
7.15. Säulendiagramm in Dreiecksform Nr. 2	70
7.16. Säulendiagramm in Dreiecksform Nr. 3	70
7.17. Ein Kreisdiagramm	71
7.18. Ein Kreisdiagramm mit $r=2,0.25,0$	71
7.19. Veränderte Zeile 3 <code>farbe=blue,solid,blue!20 r=2,0.5 text="U-Bahn"</code> . .	71
7.20. Ein Ringdiagramm	72
7.21. Veränderte Zeile 5 <code>farbe=blue,solid,blue!20 r=1.5,0.75,0.5 text="zu Fuss"</code>	72
7.22. Ein Piktogramm	73
7.23. Eine Strichliste	73
7.24. Eine Legende	74
7.25. Eine zweite Legende	74
7.26. Die Binomialverteilung	75
7.27. Die kumulierte Binomialverteilung	75
7.28. Die Binomialverteilung teilweise dargestellt	76
7.29. Die Binomialverteilung als Strichdiagramm	76
7.30. Die kumulierte Binomialverteilung als Strichdiagramm	76
7.31. Die Binomialverteilung als Strichdiagramm teilweise dargestellt	77
8.1. Koordinaten in einem dreidimensionalen Koordinatensystem (aus : [VOSS, 2008, S. 355])	79
8.2. <code>coortype=0</code>	80
8.3. <code>coortype=1</code>	80
8.4. <code>coortype=2</code>	81
8.5. <code>coortype=3</code>	81
8.6. Der Befehl <code>linie</code> im $\mathbb{R}^3$	82

8.7. Der Befehl <code>linie</code> im $\mathbb{R}^3$ mit Text	82
8.8. Der Befehl <code>linie</code> im $\mathbb{R}^3$ mit mehreren Wertepaaren	83
8.9. Eine Linie wird über seinen Startpunkt und die Verschiebung definiert.	83
8.10. Der Befehl <code>vlinie</code> im $\mathbb{R}^3$	84
8.11. Der Befehl <code>vektor</code> im $\mathbb{R}^3$	84
8.12. Der Befehl <code>polygon</code> im $\mathbb{R}^3$	85
8.13. Der Befehl <code>dreieck</code> im $\mathbb{R}^3$	85
8.14. Der Befehl <code>viereck</code> im $\mathbb{R}^3$	86
8.15. Der Befehl <code>ellipse</code> im $\mathbb{R}^3$	86
8.16. Der Befehl <code>kreis</code> im $\mathbb{R}^3$	87
8.17. Eine Kurve im $\mathbb{R}^3$	87
8.18. Eine Bezierkurve im $\mathbb{R}^3$	88
8.19. Der Befehl <code>beschriftung</code> im $\mathbb{R}^3$	88
8.20. Der Befehl <code>punkt</code> im $\mathbb{R}^3$	89
8.21. Der Befehl <code>kreuz</code> im $\mathbb{R}^3$	89
8.22. Eine Grafik einfügen	90
8.23. Ein erstes Koordinatensystem im $\mathbb{R}^3$	90
8.24. Ein zweites Koordinatensystem im $\mathbb{R}^3$	91
8.25. Der Befehl <code>kariert</code> im $\mathbb{R}^3$	91
8.26. Der Befehl <code>graph</code> im $\mathbb{R}^3$	92
8.27. Der Befehl <code>graph</code> mit <code>pstricks</code> und <code>hiddenLine</code>	93
8.28. Der Befehl <code>parmetricplot</code> im $\mathbb{R}^3$	93
8.29. Der Befehl <code>parmetricplot</code> mit dem Intervall u	94
8.30. Der Befehl <code>listplot</code> im $\mathbb{R}^3$	94
8.31. Der Befehl <code>gerade</code>	95
8.32. Der Befehl <code>ebene</code> in der Parametergleichung	96
8.33. Der Befehl <code>ebene</code> in der Koordinatengleichung	96
8.34. Die Ebene $6x + 4z = 12$	97
8.35. Die Ebene $6x + 4z = 12$	97
8.36. Eine Ebene in Normalenform	98
8.37. Ein Winkel im $\mathbb{R}^3$	98
8.38. Eine Kugel	99
8.39. Ein Quader	99
8.40. Eine Pyramide	100
8.41. Ein Pyramidenstumpf	100
8.42. Ein Kreiszylinder	101
8.43. Ein Kegel	102
8.44. Ein Kegelstumpf	102
9.1. Eine erste Uhr	103
9.2. Eine zweite Uhr	103
9.3. Die lineare Regression	104

Tabellenverzeichnis

6.1. Entladestrom eines Kondensators	59
7.1. Schulweg	64
8.1. Parameter für dreidimensionale Funktionen	92

Listings

3.1. Der Befehl <code>grenzengitter</code>	21
4.1. Der Befehl <code>linie</code>	23
4.2. Der Befehl <code>linie</code> mit Text	23
4.3. Der Befehl <code>linie</code> mit zusätzlichem Abstand	23
4.4. Der Befehl <code>linie</code> mit mehreren Wertepaaren	24
4.5. Der Befehl <code>linie</code> erweitert	24
4.6. Der Befehl <code>linie</code> mit Text	24
4.7. Der Befehl <code>linie</code> mit zusätzlichem Abstand	24
4.8. Der Befehl <code>vlinie</code>	25
4.9. Der Befehl <code>vektor</code>	25
4.10. Der Befehl <code>polygon</code>	26
4.11. Der Befehl <code>neck</code>	26
4.12. Der Befehl <code>dreieck</code>	27
4.13. Der Befehl <code>dreieck</code>	27
4.14. Der Befehl <code>dreieck</code>	28
4.15. Der Befehl <code>rechteck</code>	28
4.16. Der Befehl <code>viereck</code>	29
4.17. Der Befehl <code>parallelogramm</code>	29
4.18. Ein Kreis wird gezeichnet	30
4.19. Ein Kreis mit Text	30
4.20. Ein Kreisbogen	30
4.21. Ein geschlossener Kreisbogen	31
4.22. Ein Kreisbogen mit Pfeil	31
4.23. Der Befehl <code>kreisskala</code>	31
4.24. Ein Ringausschnitt	32
4.25. Eine Ellipse	32
4.26. Ein Ellipsenbogen	32
4.27. Ein geschlossener Ellipsenbogen	33
4.28. Ein Ellipsenbogen mit Pfeil	33
4.29. Eine Kurve	33
4.30. Eine Bezierkurve	34
4.31. Der Befehl <code>beschriftung</code>	34
4.32. Der Befehl <code>beschriftunggroesse</code>	35
4.33. Der Befehl <code>beschriftung</code>	35
4.34. Der Befehl <code>punkt</code>	36
4.35. Der Befehl <code>punkt</code> mit einem Funktionsaufruf	36
4.36. Der Befehl <code>kreuz</code>	37
4.37. Der Befehl <code>kreuz</code> an dem Graphen $f(x) = x^2$	37
4.38. Eine Grafik einfügen	38
4.39. Ein Winkel	38
4.40. Ein weiterer Winkel	39
4.41. Der Befehl <code>knoten</code>	39

4.42. Erstes Beispiel für den Befehl <code>knoten</code>	39
4.43. Zweites Beispiel für den Befehl <code>knoten</code>	40
4.44. Der Befehl <code>neuesbild</code>	40
5.1. Ein erstes Koordinatensystem	42
5.2. Ein zweites Koordinatensystem	42
5.3. Ein Koordinatensystem mit dem Befehl <code>mitte</code>	42
5.4. Eine x -Achse	43
5.5. Eine y -Achse	43
5.6. Eine x -Achse mit weiteren Teilstrichen	43
5.7. Eine x -Achse ohne Pfeile	43
5.8. Ein trigonometrisches Koordinatensystem	44
5.9. Der Befehl <code>deg</code>	44
5.10. Der Befehl <code>xachserad</code>	44
5.11. Der Befehl <code>xachsenpunkt</code>	45
5.12. Der Befehl <code>yachsenpunkt</code>	45
5.13. Der Befehl <code>xachsenpunktohne</code>	45
5.14. Der Befehl <code>yachsenpunktohne</code>	45
5.15. Der Befehl <code>xpfeil</code>	46
5.16. Der Befehl <code>ypfeil</code>	46
5.17. Der Befehl <code>xpfeilkasten</code>	47
5.18. Der Befehl <code>ypfeilkasten</code>	47
5.19. Der Befehl <code>kariert</code>	48
5.20. Ein weiteres Gitter	48
5.21. Der Befehl <code>gitter</code>	48
5.22. Der Befehl <code>matheheft</code>	48
5.23. Der Befehl <code>matheheft</code>	49
5.24. Der Befehl <code>millimeterpapier</code>	49
5.25. Der Befehl <code>grid</code>	49
5.26. Der Befehl <code>punkte</code>	49
6.1. Der Befehl <code>graph</code>	51
6.2. Der Befehl <code>graphmarkierung</code>	52
6.3. Der Befehl <code>ableitung</code>	52
6.4. Der Befehl <code>tangente</code>	53
6.5. Der Befehl <code>tangentendreieck</code>	54
6.6. Der Befehl <code>sekante</code>	54
6.7. Der Befehl <code>sekantendreieck</code>	55
6.8. Der Befehl <code>stammfunktion</code>	55
6.9. Die Obersumme von $f(x) = x^2$	56
6.10. Die Untersumme von $f(x) = x^2$	56
6.11. Der Befehl <code>integral</code>	57
6.12. Der Befehl <code>einschluss</code>	57
6.13. Der Befehl <code>graphrot</code>	58
6.14. Der Befehl <code>graphrot</code> mit feineren Einteilung	58
6.15. Der Befehl <code>parametricplot</code>	59
6.16. Der Befehl <code>folge</code>	59
6.17. Der Befehl <code>listplot</code>	60
6.18. Der Befehl <code>fileplot</code>	60
6.19. Der Befehl <code>dataplot</code>	60

6.20. Der Befehl <code>barplot</code>	61
7.1. Säulendiagramm	65
7.2. Säulendiagramm Nr. 2	65
7.3. Säulendiagramm Nr. 3	65
7.4. Balkendiagramm	66
7.5. Balkendiagramm Nr. 2	67
7.6. Balkendiagramm Nr. 2	67
7.7. Gestapeltes Balkendiagramm	67
7.8. Gestapeltes Säulendiagramm	68
7.9. Balkendiagramm in Dreiecksform Nr. 1	68
7.10. Balkendiagramm in Dreiecksform Nr. 2	69
7.11. Balkendiagramm in Dreiecksform Nr. 3	69
7.12. Säulendiagramm in Dreiecksform Nr. 1	69
7.13. Säulendiagramm in Dreiecksform Nr. 2	70
7.14. Säulendiagramm in Dreiecksform Nr. 3	70
7.15. Ein Kreisdiagramm	71
7.16. Ein Ringdiagramm	72
7.17. Ein Piktogramm	73
7.18. Eine Strichliste	73
7.19. Eine Legende	74
7.20. Eine zweite Legende	74
7.21. Die Binomialverteilung	75
7.22. Die kumulierte Binomialverteilung	75
7.23. Die Binomialverteilung teilweise dargestellt	76
7.24. Die Binomialverteilung als Strichdiagramm	76
7.25. Die kumulierte Binomialverteilung als Strichdiagramm	76
7.26. Die Binomialverteilung als Strichdiagramm teilweise dargestellt	77
8.1. <code>coortype=0</code>	80
8.2. <code>coortype=1</code>	80
8.3. <code>coortype=2</code>	81
8.4. <code>coortype=3</code>	81
8.5. Der Befehl <code>linie</code> im $\mathbb{R}^3$	82
8.6. Der Befehl <code>linie</code> im $\mathbb{R}^3$ mit Text	82
8.7. Der Befehl <code>linie</code> im $\mathbb{R}^3$ mit mehreren Wertepaaren	83
8.8. Der Befehl <code>vlinie</code> im $\mathbb{R}^3$	84
8.9. Der Befehl <code>vektor</code> im $\mathbb{R}^3$	84
8.10. Der Befehl <code>polygon</code> im $\mathbb{R}^3$	85
8.11. Der Befehl <code>dreieck</code> im $\mathbb{R}^3$	85
8.12. Der Befehl <code>viereck</code> im $\mathbb{R}^3$	86
8.13. Der Befehl <code>ellipse</code> im $\mathbb{R}^3$	86
8.14. Der Befehl <code>kreis</code> im $\mathbb{R}^3$	87
8.15. Eine Kurve im $\mathbb{R}^3$	87
8.16. Eine Bezierkurve im $\mathbb{R}^3$	88
8.17. Der Befehl <code>beschriftung</code> im $\mathbb{R}^3$	88
8.18. Der Befehl <code>punkt</code> im $\mathbb{R}^3$	89
8.19. Der Befehl <code>kreuz</code> im $\mathbb{R}^3$	89
8.20. Eine Grafik einfügen	90
8.21. Ein erstes Koordinatensystem im $\mathbb{R}^3$	90

8.22. Ein zweites Koordinatensystem im $\mathbb{R}^3$	91
8.23. Der Befehl <code>kariert</code> im $\mathbb{R}^3$	91
8.24. Der Befehl <code>graph</code> im $\mathbb{R}^3$	92
8.25. Der Befehl <code>graph</code> mit <code>pstricks</code> und <code>hiddenLine</code>	93
8.26. Der Befehl <code>parmetricplot</code> im $\mathbb{R}^3$	93
8.27. Der Befehl <code>parmetricplot</code> mit dem Intervall u	94
8.28. Der Befehl <code>listplot</code> im $\mathbb{R}^3$	94
8.29. Der Befehl <code>gerade</code>	95
8.30. Der Befehl <code>ebene</code> in der Parametergleichung	96
8.31. Der Befehl <code>ebene</code> in der Koordinatengleichung	96
8.32. Die Ebene $6x + 4z = 12$	97
8.33. Die Ebene $6x + 4z = 12$	97
8.34. Eine Ebene in Normalenform	98
8.35. Ein Winkel im $\mathbb{R}^3$	98
8.36. Eine Kugel	99
8.37. Ein Quader	99
8.38. Eine Pyramide	100
8.39. Ein Pyramidenstumpf	100
8.40. Ein Kreiszylinder	101
8.41. Ein Kegel	102
8.42. Ein Kegelstumpf	102
9.1. Eine erste Uhr	103
9.2. Eine zweite Uhr	103
9.3. Die lineare Regression	104
A.1. <code>plotter.pl</code>	107
A.2. <code>achsen.pm</code>	239
A.3. <code>dateierzeugen.pm</code>	255
A.4. <code>diagramme.pm</code>	271
A.5. <code>dreidimensional.pm</code>	281
A.6. <code>funktionen.pm</code>	322
A.7. <code>funktionsparser.pm</code>	332
A.8. <code>gitter.pm</code>	347
A.9. <code>graphen.pm</code>	352
A.10. <code>paar.pm</code>	370
A.11. <code>zeichnen.pm</code>	372
A.12. <code>bildbearbeitung.pm</code>	400
A.13. <code>uhr.pm</code>	402
A.14. <code>knoten.pm</code>	403
A.15. <code>regression.pm</code>	407

1. Vorwort

1.1. Version 1

Im Laufe meiner Arbeit als Mathematik- und Physiklehrerin habe ich bisher einige Zeichnungen für Arbeitsblätter o.ä. erstellt. Da ich für mich den Anspruch hatte, dass diese eine ähnliche Gestalt haben sollten, hatte ich mir für Xfig ein Perl-Skript gebaut, welches mir die verschiedenen Zeichenelemente direkt erzeugt. Als ich dann vor einiger Zeit Pstricks entdeckt hatte, hatte ich das Xfig-Skript nach Pstricks konvertiert. Dabei tauchten im Laufe der Zeit einige Probleme auf, die mit einigem gebastel auch wieder zu beheben waren. Da ich irgendwann keine Lust mehr auf das gebastel hatte, habe ich das alte Skript in die Ecke geworfen und von vorne angefangen ein neues Konzept zu erstellen und habe angefangen neu zu programmieren. Es sind einige neue Funktionalitäten hinzugekommen. Mittlerweile habe ich zwar auch festgestellt, dass einige weitere neue Ideen in das Konzept noch nicht ganz einfach abzubilden sind, aber mein nächster Plan ist nach der Fertigstellung das ganze auf Java zu portieren und eventuell als online-Skript zur Verfügung zu stellen. Aber all das ist noch Zukunftsmusik in die ganz weite Zukunft. Ob dies jemals stattfinden wird ist die andere Frage.

Ich habe mich natürlich auch teilweise gefragt, ob dieses Skript eigentlich Sinn macht und es nicht besser ist einfach direkt in PsTricks zu arbeiten. Schließlich gibt es dazu eine Menge Dokumentation¹, aber andererseits fallen immer wieder gleiche Anforderungen an, sodass ich möchte, dass meine Dateien einheitlich aussehen und wenn ich einmal etwas ändere ich nicht alle Dateien wieder neu anfassen muss. So ist auch ein Abschnitt zu Diagrammen entstanden, damit z.B. in Klasse 5 Balken und Tortendiagramme gezeichnet werden können.

Letztendlich macht mir Programmieren auch eine Menge Spaß und so habe ich an diesem Projekt eine Menge gelernt und würde wahrscheinlich mittlerweile wieder alles von vorne neu anfangen, wenn ich die Zeit hätte. Vielleicht kann aber auch noch jemand anderes hieraus Nutzen ziehen.

Xenia Rendtel

Hamburg, den 12.08.2010

1.2. Version 1.5

Mittlerweile ist wieder einige Zeit ins Land gegangen und es hat sich nebenbei einiges an meinem Plotter-Skript getan. Meistens habe ich dies einfach bei der normalen Arbeit mit eingebaut, wenn ich festgestellt habe, ich benötige noch dies und das für den aktuellen Unterricht. So sind im Bereich der Statistik einige Dinge hinzugekommen.

Es sind ebenfalls einige Fehler behoben worden.

Was ist nun neu hinzugekommen? Es gibt ein paar neue Perl-Dateien und es sind einige Dinge neu aufgeteilt worden. So gibt es nun die folgenden neuen Dateien:

- `uhr.pm`
- `knoten.pm`
- `regression.pm`

¹Siehe z.B. (Voss, 2008) und (tug, 2009)

mit zugehörigen Aufrufen.
Neue Befehle sind:

- `grenzengitter`
- Einen Vektor mit Startpunkt und Verschiebung mit `vlinie` zeichnen
- Es gibt zwei weitere Dreiecksbefehle.
- Der Befehl `beschriftungsgroesse` setzt die Schriftgröße der Beschriftung.
- `kreisskala`
- Ausschnitt einer Ellipse mit `ellipsebogen` zeichnen
- `ticks`
- `pfeile`
- `xachsenpunktohne`
- `yachsenpunktohne`
- Gitter für ein Standardmatheheft mit `matheheft`
- Rotationskörper mit `graphrot` zeichnen
- Zeichnen von Funktionsfolgen mit `folge`
- Der Befehl `barplot` ist neu.
- Der Befehl `farbverlauf` bei den Diagrammen ist neu.
- Die Befehle `binomialstrich` und `binomialteilstrich` sind analog zu `binomial` und `binomialteil` hinzugekommen.

Änderungen gibt es in:

- Der Befehl `linie` wurde erweitert.
- Den Befehlen `beschriftung`, `punkt` und `kreuz` kann man einen Funktionsterm übergeben.
- Der Befehl `knoten` wurde erweitert.
- Die Diagramme `sauelendiagramm` und `balkendiagramm` wurden erweitert.

Xenia Rendtel

Hamburg, den 09.01.2012

2. Aufbau des Skriptes

Zunächst hatte ich am Anfang eine riesige Perl-Datei, aber im Laufe der Zeit wurde dies einfach zu groß. So sind mehrere Pakete entstanden, die jeweils die verschiedenen Funktionalitäten enthalten. Die Hauptdatei ist die Datei `plotter.pl`. Diese Datei dient zum Auslesen der eingegebenen Befehle. In dieser Datei werden mehrere Module eingebunden, diese sind:

- `achsen.pm`
- `dateierzeugen.pm`
- `diagramme.pm`
- `dreidimensional.pm`
- `funktionen.pm`
- `funktionsparser.pm`
- `gitter.pm`
- `graphen.pm`
- `paar.pm`
- `zeichnen.pm`
- `uhr.pm`
- `knoten.pm`
- `regression.pm`

Auf diese Module wird allerdings hier nicht weiter eingegangen. Man findet diese im Anhang A. Dazu müssen in einer Datei mit der Extension `ptxt` verschiedene Objekte angegeben, z.B. Vektoren, Linien, etc. und diese `ptxt`-Datei wird dann durch das Perl-Skript auf der Kommandozeile mit dem Aufruf

```
plotter.pl datei.ptxt > datei.ltx
```

in eine `datei.ltx` verwandelt. Die mit den folgenden $\text{\LaTeX}$ -Befehlen

```
latex datei.ltx
latex datei.ltx
dvips -E -j0 datei.dvi
ps2pdf datei.ps
```

in eine pdf-Datei konvertiert wird.

In den folgenden Abschnitten erläutere ich zunächst die möglichen Befehle und darauf folgt im Anhang A der Quellcode.

3. Allgemeine Einstellungen

3.1. Skalierung und Größe des Bildes

Als erstes sollte in der ptxt-Datei enthalten sein, wie groß man die Datei skaliert haben möchte. Dazu gibt es zwei Möglichkeiten

skalierung $\langle r \in \mathbb{R} \rangle$

oder

skalierung $x=\langle r \in \mathbb{R} \rangle$ $y=\langle r \in \mathbb{R} \rangle$

Wenn man z.B. nur

skalierung 3

setzt, so wird der Skalierungsfaktor für die x - und y -Achse auf den gleichen Wert gesetzt.

Um ein Bild auf eine bestimmte Größe zu setzen gibt es den Befehl **groesse**, der die Ausmaße in x - und y -Richtung festsetzt. Die Syntax lautet:

groesse $x=\langle r \in \mathbb{R}_{>0} \rangle$ $y=\langle r \in \mathbb{R}_{>0} \rangle$

3.2. Grenzen des Bildes

Mit dem Befehl **grenzen** wird definiert, in welchen Grenzen das Bild gezeichnet werden soll.

grenzen $x=\langle x_{min} \in \mathbb{R} \rangle, \langle x_{max} \in \mathbb{R} \rangle$ $y=\langle y_{min} \in \mathbb{R} \rangle, \langle y_{max} \in \mathbb{R} \rangle$

Desweiteren gibt es den Befehl **grenzengitter**, der einem ein Koordinatensystem und Gitter zur Verfügung stellt.

grenzengitter $x=\langle x_{min} \in \mathbb{R} \rangle, \langle x_{max} \in \mathbb{R} \rangle$ $y=\langle y_{min} \in \mathbb{R} \rangle, \langle y_{max} \in \mathbb{R} \rangle$

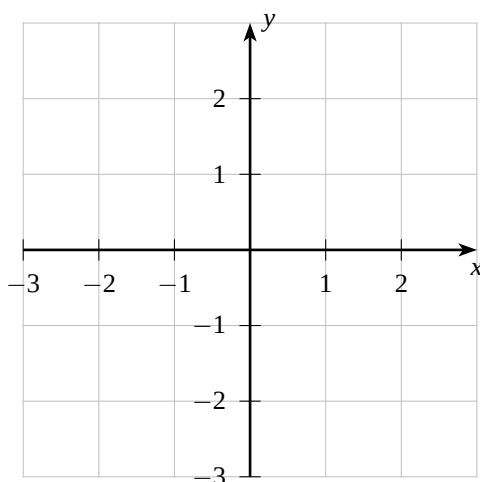


Abb. 3.1.: Der Befehl **grenzengitter**

Listing 3.1: Der Befehl **grenzengitter**

```
1 grenzengitter x=-3,3 y=-3,3
```

3.3. Schriftgröße eines Bildes

Mit dem Befehl `schriftgroesse` setzt man die Größe der Schrift eines Bildes.

schriftgroesse $\langle i \in [10;11;12] \rangle$

3.4. Allgemeine Einstellungen eines Bildes

Mit dem Befehl `einstellung` kann man vorweg definieren, in welcher Farbe standardmäßig das Bild gezeichnet werden soll und in welcher Stärke dies geschieht. Könne

einstellung

`[layer= $\langle n \in [0;100] \rangle$ ]`

`[farbe= $\langle \text{Linienfarbe} \rangle, \langle \text{Füllart} \rangle, \langle \text{Füllfarbe} \rangle$ ]`

linie= $\langle \text{Linienstaerke} \rangle, \langle \text{Linienart} \rangle [, \langle \text{Linienfarbe} \rangle]$

Standardmäßig ist die Farbe auf `black` gesetzt, die Füllart ist `solid`, die Füllfarbe ist `black!20` und die Linienstärke ist auf 1 pt gesetzt.

Zusätzlich gibt es noch den Befehl `ohnekopf`, der bewirkt, dass eine PsTricks-Datei ohne Vor- und Nachspann erzeugt wird.

4. Einfache Zeichenroutinen

Mit meinem Skript gibt es die Möglichkeit verschiedene Zeichenobjekte zu erzeugen. Dies sind Objekte wie Linien, Polygone, Kreise und Ellipsen.

4.1. Linien, Vektoren und Polygone

4.1.1. Einfache Linien zeichnen

Eine Vielzahl von Zeichenobjekten sind aus Linien aufgebaut. So darf natürlich auch den einzelnen Befehl einer Linie nicht fehlen. Eine **Linie** zeichnet man mit dem Befehl `linie` der wie folgt aufgebaut ist:

linie
`[layer=<n ∈ [0;100]>]`
`[farbe=<Linienfarbe>]`
`[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Pfeiltyp>]]`
`[parameter=<Pstricksbefehle>]`
`p=<x1 ∈ ℝ>,<y1 ∈ ℝ> p=<x2 ∈ ℝ>,<y2 ∈ ℝ>`
`[r=<Abstand r ∈ ℝ>]`
`[text=<Text>[,<Drehwinkel des Textes in Grad>,<Textausrichtung>,<x-Richtung>,<y-Richtung>]]`

In `p` stehen die Koordinaten des Anfangs und Endpunktes der Linie. Zusätzlich kann noch ein Text mit `text` an die Linie geschrieben werden mit dem Befehl `text`, der optional ist. Wird nur ein Text für sich gesetzt, so wird der mittig von der Linie platziert. In `r` steht der Abstand von den ursprünglichen Koordinaten.



Abb. 4.1.: Der Befehl `linie`

Listing 4.1: Der Befehl `linie`

```
1 linie farbe=black linie=1,solid p=0,0 p=1,1
```



Abb. 4.2.: Der Befehl `linie` mit Text

Listing 4.2: Der Befehl `linie` mit Text

```
1 linie linie=1,solid,"| ->" p=0,0 p=1,1 text="$r$"  
    ,0,180,0,-0.4
```

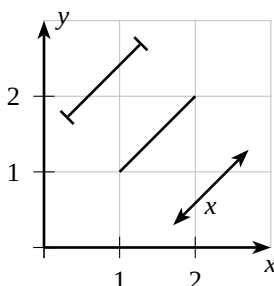


Abb. 4.3.: Der Befehl `linie` mit zusätzlichem Abstand

Listing 4.3: Der Befehl `linie` mit zusätzlichem Abstand

```
1 grenzgitter x=0,3 y=0,3  
2 linie p=1,1 p=2,2  
3 linie linie=1,solid,"<->" p=1,1 p=2,2 r=-1 text="  
    $x$",0,270,0,-0.2  
4 linie linie=1,solid,"| -|" p=1,1 p=2,2 r=1
```

Man kann auch an eine Linie mehrere Werte übergeben, damit diese Koordinaten miteinander verbunden werden. Dazu wird der Befehl `linie` etwas abgewandelt, wie man in 4.4 sieht.

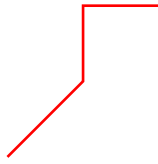


Abb. 4.4.: Der Befehl `linie` mit mehreren Wertepaaren

Listing 4.4: Der Befehl `linie` mit mehreren Wertepaaren

```

1 linie farbe=red linie=1,solid
2 0,0
3 1,1
4 1,2
5 2,2
6 ende

```

Linien mit einer bestimmten Länge und einem Winkel

Zusätzlich zu dem Standardlinienbefehl gibt es noch die Erweiterung des Befehls `linie`, an den man die Startposition, die Länge der Linie und den Winkel übergibt.¹

`linie`

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt ( $l \in \mathbb{R}_{>0}$ )>,<Linienart>[,<Pfeiltyp>]]
[parameter="<Pstricksbefehle>"]
p=< $x_{Start} \in \mathbb{R}$ >,< $y_{Start} \in \mathbb{R}$ > r=<Linienlänge  $l \in \mathbb{R}_{>0}$ >,<Winkel in Grad>
[r=<Abstand  $r \in \mathbb{R}$ >]
[text="<Text>[,<Drehwinkel des Textes in Grad>,<Textausrichtung>,<x-Richtung>,<y-Richtung>]]

```

Das Beispiel dazu:



Abb. 4.5.: Der Befehl `linie` erweitert

Listing 4.5: Der Befehl `linie` erweitert

```

1 linie farbe=blue linie=2,dashed,"|—" p=0,0 r=1,40

```

Hierbei kann ebenfalls Text mit angefügt werden, wie bereits im ursprünglichen `linie`-Befehl.



Abb. 4.6.: Der Befehl `linie` mit Text

Listing 4.6: Der Befehl `linie` mit Text

```

1 linie linie=1,solid,"|—" p=0,0 r=1,45 text="$r$"
,0,180,0,-0.4

```

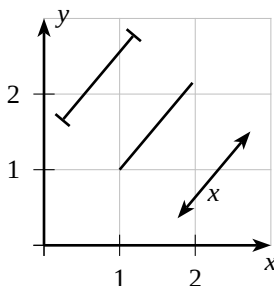


Abb. 4.7.: Der Befehl `linie` mit zusätzlichem Abstand

Listing 4.7: Der Befehl `linie` mit zusätzlichem Abstand

```

1 grenzgitter x=0,3 y=0,3
2 linie p=1,1 r=1.5,50
3 linie linie=1,solid,"<-->" p=1,1 r=1.5,50 r=-1 text
="$x$",0,270,0,-0.2
4 linie linie=1,solid,"|—" p=1,1 r=1.5,50 r=1

```

¹Dies war ursprünglich der Befehl `liniepolar` in der vorherigen Version.

Linien mit Verschiebungen

Ein neuer Befehl ist der Linienbefehl `vlinie`. Bei diesem gibt man von einem Startpunkt die Verschiebung in x - und y -Richtung an.

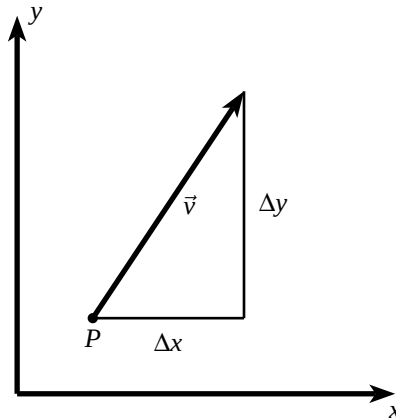


Abb. 4.8.: Eine Linie wird über seinen Startpunkt und die Verschiebung definiert.

`vlinie`

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[, "<Pfeiltyp>"]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ> r=<Δx ∈ ℝ>,<Δy ∈ ℝ>
[text=<Text>[,<Drehwinkel des Textes in Grad>,<Textausrichtung>,<x-Richtung>,<y-Richtung>]]
```

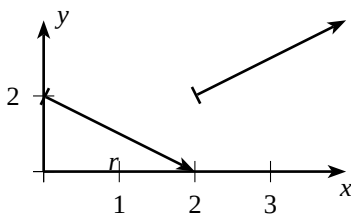


Abb. 4.9.: Der Befehl `vlinie`

Listing 4.8: Der Befehl `vlinie`

```
1 achsen
2 vlinie farbe=black linie=1,solid,"l->" p=0,2 r
   =2,-1 text="$r$",0,180,0,-0.4
3 vlinie farbe=black linie=1,solid,"l->" p=2,2 r=2,1
```

4.1.2. Vektoren zeichnen

Vektoren kann man natürlich zeichnen, indem man den `pfeiltyp` auf `"->"` setzt. Dies erledigt allerdings auch schon der Befehl `vektor` selbst, der analog zum Befehl der Linie ist.



Abb. 4.10.: Der Befehl `vektor`

Listing 4.9: Der Befehl `vektor`

```
1 vektor farbe=black linie=1,solid p=0,0 p=1,1
```

4.1.3. Polygonzüge zeichnen

Möchte man anstelle von aneinandergereihten Linien einen geschlossen Polygonzug zeichnen, so nutzt man den Befehl `polygon`, der fast ähnlich zum Linienbefehl aussieht. Zusätzlich zu der Linienfarbe kann mit `farbe` die Linienfarbe und Füllfarbe definiert werden.

polygon

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
```

[**lese** <Dateiname>] oder [$x \in \mathbb{R}$, $y \in \mathbb{R}$] beliebig oft

ende

Ist die Farbe nicht gesetzt, so werden die Standardwerte hierfür gesetzt. Möchte man trotzdem noch eine andere Linienfarbe, so hat man die Möglichkeit dies unter `linie` mit zu setzen.

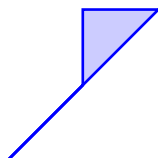


Abb. 4.11.: Der Befehl `polygon`

Listing 4.10: Der Befehl `polygon`

```
1 polygon farbe=blue,solid,blue!20 linie=1,solid,blue
   parameter=""
2 lese Abbildungen/test.dat
3 ende
```

4.1.4. Regelmäßiges n -Eck

Anstelle eines unregelmäßigen Polygonzugs gibt es auch die Möglichkeit ein regelmäßiges n -Eck zu zeichnen. Ein regelmäßiges Vieleck hat gleich lange Seiten und gleiche Innenwinkel φ , die sich mit $\varphi = \frac{360^\circ}{n}$ berechnen lassen. Ein reguläres n -Eck hat stets einen Umkreis mit Radius r . Die Länge jeder Seite wird mit a bezeichnet, die Seitenanzahl mit n . Dabei ist zusätzlich λ der Drehwinkel bezüglich der x -Achse.

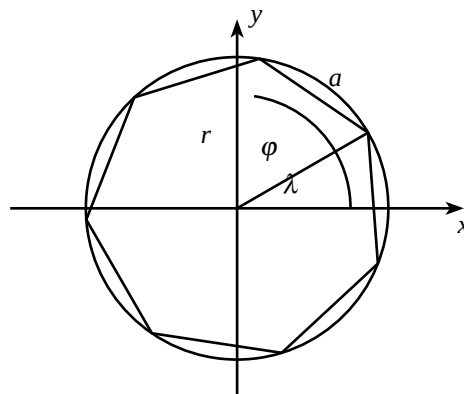


Abb. 4.12.: Ein regelmäßiges n -Eck

Die Syntax für das Skript ist dabei die folgende:

neck

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ>
r=<Eckenzahl e ∈ ℕ>,<Radius r ∈ ℝ>0>,<Drehwinkel in Grad>
```

ende

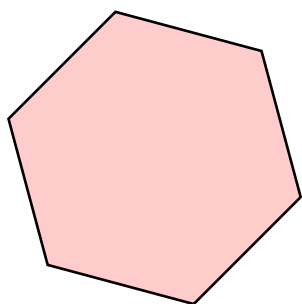


Abb. 4.13.: Der Befehl `neck`

Listing 4.11: Der Befehl `neck`

```
1 neck farbe=black,solid,red!20 linie=1,solid
   parameter="" p=0,0 r=6,2,45
```

4.2. Dreiecke, Rechtecke und Vierecke zeichnen

4.2.1. Dreiecke

Es kommt ja auch häufiger vor, dass man ein **Dreieck** oder **Rechteck** zeichnen möchte. Dies könnte man natürlich über ein Polygon mit drei bzw. vier Punkten zeichnen, aber es gibt auch die Befehle `dreieck` und `rechteck`, die diese Objekte zeichnen. Die Syntax des Dreiecks ist die folgende:

dreieck

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ> p=<x2 ∈ ℝ>,<y2 ∈ ℝ> p=<x3 ∈ ℝ>,<y3 ∈ ℝ>
```

Beim Dreieck werden die Koordinaten der drei Eckpunkte angegeben.

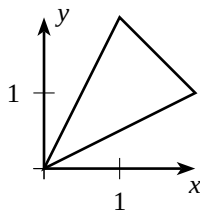


Abb. 4.14.: Der Befehl `dreieck`

Listing 4.12: Der Befehl `dreieck`

```
1 achsen
2 dreieck p=0,0 p=2,1 p=1,2
```

Bei einem weiteren Dreieck-Befehl werden der Startpunkt, zwei Kantenlängen und der eingeschlossene Winkel angegeben. Zusätzlich der Drehwinkel bezüglich der x -Achse.

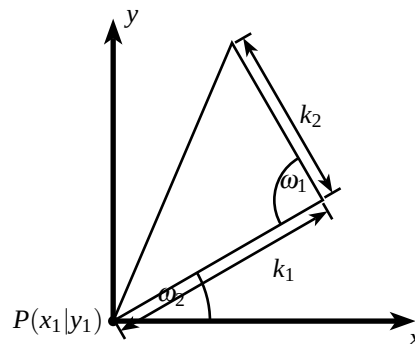


Abb. 4.15.: Ein Dreieck wird über zwei Kanten und einen Winkel definiert

Der Befehl ist im folgenden aufgerufen:

dreieck

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ> r=<k1 ∈ ℝ>,<k2 ∈ ℝ>,<ω1 ∈ ℝ>,<ω2 ∈ ℝ>
```

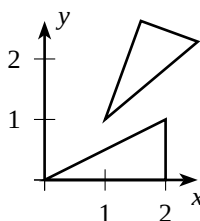


Abb. 4.16.: Der Befehl `dreieck`

Listing 4.13: Der Befehl `dreieck`

```
1 skalierung 0.8
2 achsen
3 dreieck p=0,0 r=2,1,90,0
4 dreieck p=1,1 r=2,1,30,40
```

Bei einem weiteren Dreiecksbefehl wird ein **rechtwinkliges Dreieck** über zwei Punkte und den eingeschlossenen Winkel beim ersten Punkt definiert.

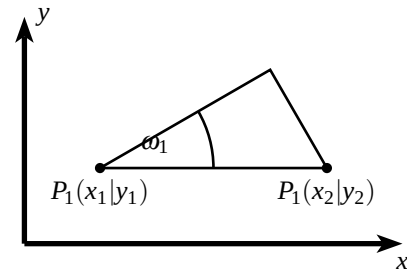


Abb. 4.17.: Ein rechtwinkliges Dreieck wird über zwei Punkte und einen weiteren Winkel definiert

Der Befehl sieht wie folgt aus:

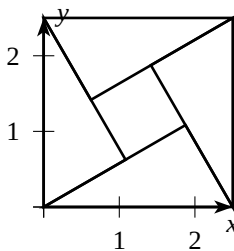


Abb. 4.18.: Der Befehl `dreieck`

Listing 4.14: Der Befehl `dreieck`

```
1 achsen
2 dreieck p=0,0 p=2.5,0 r=30
3 dreieck p=2.5,0 p=2.5,2.5 r=30
4 dreieck p=0,2.5 p=2.5,2.5 r=120
5 dreieck p=0,0 p=0,2.5 r=120
```

4.2.2. Rechteck

Bei einem **Rechteck** übergibt man an den Startwert, der die linke untere Ecke des Rechteckes darstellt, die Breite und Höhe des Rechteckes.

rechteck

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ> r=<Breite b ∈ ℝ>0>,<Höhe h ∈ ℝ>0>[,<Drehwinkel in Grad>]
```

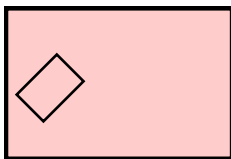


Abb. 4.19.: Der Befehl `rechteck`

Listing 4.15: Der Befehl `rechteck`

```
1 rechteck farbe=black,solid,red!20 linie=2,solid
  parameter="" p=0,0 r=3,2
2 rechteck p=0.5,0.5 r=0.75,0.5,45
```

4.2.3. Viereck

Möchte man ein allgemeines **Viereck** zeichnen, so gibt es noch des Befehl `viereck`, der analog zum Dreieck funktioniert, nur das vier Koordinaten übergeben werden.

viereck

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ> p=<x2 ∈ ℝ>,<y2 ∈ ℝ> p=<x3 ∈ ℝ>,<y3 ∈ ℝ> p=<x4 ∈ ℝ>,<y4 ∈ ℝ>
```

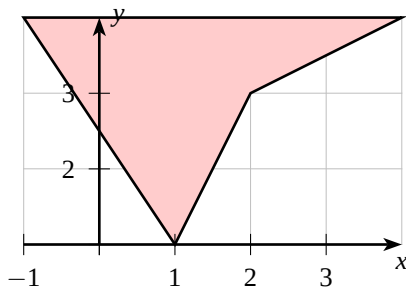


Abb. 4.20.: Der Befehl `viereck`

Listing 4.16: Der Befehl `viereck`

```
1 gitter
2 achsen
3 viereck farbe=black,solid,red!20 linie=1,solid
   parameter="" p=1,1 p=2,3 p=4,4 p=-1,4
```

4.2.4. Parallelogramme zeichnen

Als eine Neuerung kann man seit 2011 mit dem Skript **Parallelogramme** zeichnen. Ein Parallelogramm ist dabei definiert über einen Eckpunkt P , zwei Kanten und einen Winkel definiert. Gleichzeitig kann das Parallelogramm noch gedreht werden.

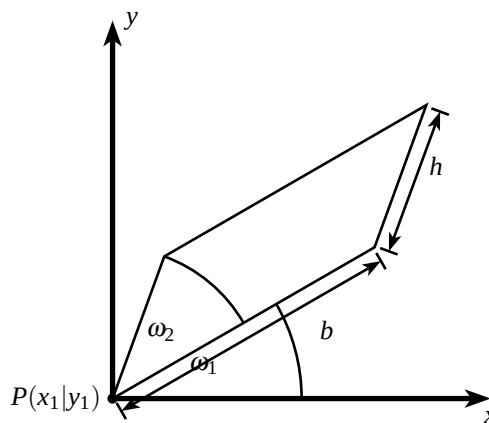


Abb. 4.21.: Ein Parallelogramm über zwei Kanten und zwei Winkel definiert

parallelogramm

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ> r=<Breite b ∈ ℝ>0>,<Höhe h ∈ ℝ>0>,<Drehwinkel ω1 x-Achse in Grad>,<Winkel
ω2 zwischen Kanten in Grad>
```

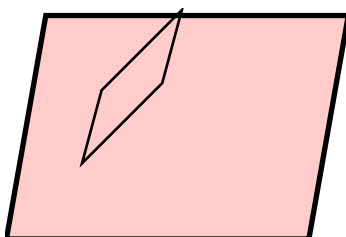


Abb. 4.22.: Der Befehl `parallelogramm`

Listing 4.17: Der Befehl `parallelogramm`

```
1 parallelogramm farbe=black,solid,red!20
   linie=2,solid parameter="" p=0,0 r=4,3,0,80
2 parallelogramm p=1,1 r=1.5,1,45,30
```

4.3. Kreise, Ellipsen und Kurven

Wie kommt das Runde in das Eckige? Bisher wurden hier nur Befehle aufgeführt, die alle durch Polygonzüge zu zeichnen waren. Jetzt wird es rund und kurvig.

4.3.1. Kreise und Kreisbögen

Zunächst wird hier dargestellt, wie ein **Kreis** gezeichnet wird. Dazu gibt es den Befehl `kreis`, der wie folgt aufgerufen wird:

```
kreis
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
  [parameter="<Pstricksbefehle>"]
  p=<x ∈ ℝ>,<y ∈ ℝ>
  r=<Radius r ∈ ℝ>0>
  [text="<Text>"[,<Drehwinkel des Textes in Grad>,<Textausrichtung>]]
```

Dabei steht in `p` der Mittelpunkt des Kreises. Mit dem Zusatz `text` ist es möglich Text in die Mitte des Kreises zu schreiben.



Abb. 4.23.: Ein Kreis wird gezeichnet

Listing 4.18: Ein Kreis wird gezeichnet

```
1 kreis farbe=black,solid,red!20 linie=1,solid
   parameter="" p=0,0 r=0.5
```



Abb. 4.24.: Ein Kreis mit Text

Listing 4.19: Ein Kreis mit Text

```
1 kreis p=0,0 r=0.5 text="$M$",0,c
```

Anstelle eines Kreises kann auch ein **Kreisbogen** gezeichnet werden. Die Syntax ist die folgende

```
kreisbogen
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
  [parameter="<Pstricksbefehle>"]
  p=<x ∈ ℝ>,<y ∈ ℝ>
  r=<Radius r ∈ ℝ>0>,<Startwinkel in Grad>,<Endwinkel in Grad>
  [geschlossen]
  [pfeil]
```

Dabei steht in `p` der Mittelpunkt des Kreisbogens und in `r` steht der Radius, sowie der Start- und Endwinkel. Dabei wird vom Startwinkel aus gegen den Uhrzeigersinn gezeichnet. Ist `geschlossen` gesetzt, so wird der Kreisbogen in der Füllfarbe ausgemalt. Ist `pfeil` gesetzt, so wird ein Pfeil zum Endwinkel hin gesetzt.

Beispiele für Kreisbögen



Abb. 4.25.: Ein Kreisbogen

Listing 4.20: Ein Kreisbogen

```
1 kreisbogen farbe=black,solid,red linie=1,solid p=0,0 r
   =0.5,0,300
```



Abb. 4.26.: Ein geschlossener Kreisbogen

Listing 4.21: Ein geschlossener Kreisbogen

```
1 kreisbogen farbe=black,solid,red!20 linie=1,solid p=0,0 r
   =0.5,0,300 geschlossen
```



Abb. 4.27.: Ein Kreisbogen mit Pfeil

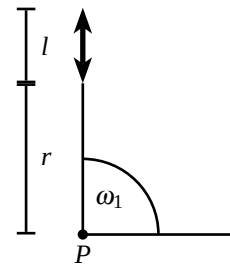
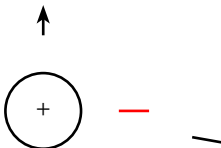
Listing 4.22: Ein Kreisbogen mit Pfeil

```
1 kreisbogen farbe=black,solid,red!20 linie=1,solid p=0,0 r
   =0.5,0,300 pfeil
```

4.3.2. Weitere Kreisbefehle

In einigen Fällen möchte man ja von einem Kreis aus eine Linie zeichnen, dazu gibt es den Befehl `kreisskala`.

```
kreisskala [layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt
  (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"] p=<x ∈ ℝ>,<y ∈ ℝ>
r=<Radius r ∈ ℝ>0>,<Länge l ∈ ℝ>0>,<Winkel in Grad>
```

Abb. 4.28.: Die Definition des Befehls `kreisskala`Abb. 4.29.: Der Befehl `kreisskala`Listing 4.23: Der Befehl `kreisskala`

```
1 kreis p=0,0 r=0.5 text="+"
2 kreisskala farbe=black linie=1,solid,"->" p=0,0 r=1,0.4,90
3 kreisskala farbe=red linie=1,solid p=0,0 r=1,0.4,0
4 kreisskala linie=1,solid p=0,0 r=2,0.4,-10
5 # beschriftungskreis farbe=black p=0,0,3,90 text="$90^{\circ}$
   Nordpol"
```

4.3.3. Ringausschnitte

Anstelle eines Kreisbogens kann auch ein **Ring** gezeichnet werden mit dem Befehl `ring`, der die folgende Syntax hat:

```
ring
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<Außenradius r ∈ ℝ>0>,<Ringbreite b ∈ ℝ>0>,<Startwinkel in Grad>,<Endwinkel in Grad>
```

Dabei steht in `p` der Mittelpunkt des Ringbogens und in `r` steht der Außenradius und die Breite des Ringes, sowie der Start- und Endwinkel. Dabei wird vom Startwinkel aus gegen den Uhrzeigersinn gezeichnet.



Abb. 4.30.: Ein Ringausschnitt

Listing 4.24: Ein Ringausschnitt

```
1 ring farbe=black,solid,red!20 linie=1,solid
   parameter="" p=0,0 r=1,0.5,0,300
```

4.3.4. Ellipsen zeichnen

Bisher wurde Kreise gezeichnet, die ja ein Spezialfall einer Ellipse sind, da die große und kleine Halbachse gleich groß sind. Es gibt in meinem Skript nun auch die Möglichkeit **Ellipsen** zu zeichnen, mit dem Befehl `ellipse` ist dies möglich. Dieser ist fast analog zum Kreisbefehl. Man übergibt allerdings in `r` zunächst die große Halbachse und dann die kleine Halbachse.

`ellipse`

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<große Halbachse a ∈ ℝ>0>0>
```

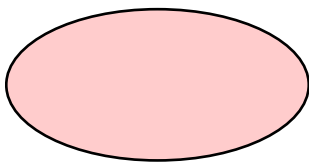


Abb. 4.31.: Eine Ellipse

Listing 4.25: Eine Ellipse

```
1 ellipse farbe=black,solid,red!20 linie=1,solid
   parameter="" p=0,0 r=2,1
```

Anstelle einer Ellipse kann auch ein **Ellipsenbogen** gezeichnet werden. Die Syntax ist die folgende

`ellipsebogen`

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<große Halbachse a ∈ ℝ>0>0

```

Dabei steht in `p` der Mittelpunkt des Ellipsenbogens und in `r` stehen die große und kleine Halbachse, sowie der Start- und Endwinkel. Dabei wird vom Startwinkel aus gegen den Uhrzeigersinn gezeichnet. Ist `geschlossen` gesetzt, so wird der Kreisbogen in der Füllfarbe ausgemalt. Ist `pfeil` gesetzt, so wird ein Pfeil zum Endwinkel hin gesetzt.

Beispiele für Ellipsenbögen



Abb. 4.32.: Ein Ellipsenbogen

Listing 4.26: Ein Ellipsenbogen

```
1 ellipsebogen farbe=black,solid,red!20 linie
   =1,solid p=0,0 r=2,1,0,200
```

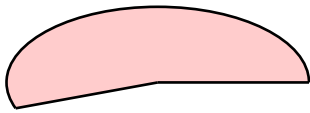


Abb. 4.33.: Ein geschlossener Ellipsenbogen

Listing 4.27: Ein geschlossener Ellipsenbogen

```
1 ellipsebogen farbe=black,solid,red!20 linie
   =1,solid p=0,0 r=2,1,0,200 geschlossen
```



Abb. 4.34.: Ein Ellipsenbogen mit Pfeil

Listing 4.28: Ein Ellipsenbogen mit Pfeil

```
1 ellipsebogen farbe=black,solid,red!20 linie
   =1,solid p=0,0 r=2,1,0,200 pfeil
```

4.3.5. Kurven zeichnen

Ähnlich zum Linienbefehl gibt es noch den Befehl der **Kurve**, der mit `kurve` aufgerufen wird. Dieser zeichnet eine Kurve zwischen den angegebenen Werten.

```
kurve
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,"<Pfeiltyp>"]]
  [parameter="<Pstricksbefehle>"]
<x ∈ ℝ>,<y ∈ ℝ> beliebig oft
ende
```

Dabei ist zu beachten, dass bei diesem Befehl die Kurve durch alle angegebenen Punkte geht.

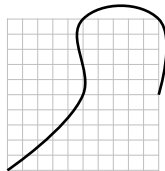


Abb. 4.35.: Eine Kurve

Listing 4.29: Eine Kurve

```
1 gitter
2 kurve farbe=black linie=1,solid
3 0,0
4 1,1
5 1,2
6 2,2
7 2,1
8 ende
```

Im Gegensatz zur Kurve geht die **Bezierkurve** nicht durch die angegebenen Punkte. Eine Bezierkurve n -ten Grades benötigt $n + 1$ Kontrollpunkte P_i , wobei $i \in [0; n]$ ist. Die Bezierkurve ist dann für $t \in [0; 1]$ definiert als

$$C(t) = \sum_{i=0}^n B_{i,n}(t) P_i \quad (4.1)$$

wobei $B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$ das i -te **Bernsteinpolynom** n -ten Grades ist.

Dies ist mit dem Befehl `bezier` realisiert, der ansonsten analog zur Kurve ist.

```
bezier
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,"<Pfeiltyp>"]]
  [parameter="<Pstricksbefehle>"]
<x ∈ ℝ>,<y ∈ ℝ> beliebig oft
ende
```

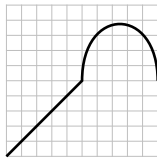


Abb. 4.36.: Eine Bezierkurve

Listing 4.30: Eine Bezierkurve

```

1 gitter
2 bezier farbe=black linie=1,solid parameter=
   ""
3 0,0
4 1,1
5 1,2
6 2,2
7 2,1
8 ende

```

4.4. Beschriftungen, Punkte und Kreuze zeichnen

4.4.1. Beschriftungen

Allgemeine Beschriftungen

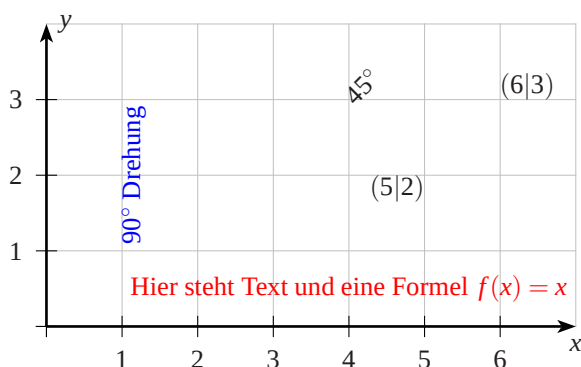
Innerhalb des Skriptes gibt es bereits einige **Beschriftungen**, da man dies aber ja auch unabhängig von anderen Objekten setzen möchte, gibt es den Befehl der `beschriftung`. An diesen wird die Position des Textes der Textwinkel und die Drehung übergeben. Standardmäßig wird der Text horizontal und zentriert dargestellt.

`beschriftung`

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
p=<x ∈ ℝ>,<y ∈ ℝ>
text="<Text>"[,<Textwinkel in Grad>,<Drehung in Grad>]

```

Abb. 4.37.: Der Befehl `beschriftung`Listing 4.31: Der Befehl `beschriftung`

```

1 grenzgitter x=0,7 y=0,4
2 achsen
3 beschriftung farbe=red p=4,0.5 text="
   Hier steht Text und eine Formel  $f(x)=x$ "
4 beschriftung farbe=blue p=1,2 text="
    $90^\circ$  Drehung",90,0
5 beschriftung p=4,3 text=" $45^\circ$ "
   ,45,45
6 beschriftung p=6,3 text=" $(6|3)$ ",0,45
7 beschriftung p=5,2 text=" $(5|2)$ ",0,225

```

Mit dem Befehl `beschriftung` kann die Größe der Beschriftung variiert werden. Dabei kann man die folgenden allgemeinen $\text{\LaTeX}$ -Schriftgrößen benutzen.

```

\tiny \scriptsize \footnotesize \small \normalsize \large \Large
\LARGE \huge \Huge

```

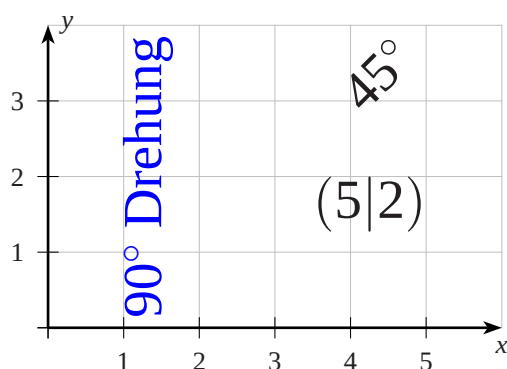


Abb. 4.38.: Der Befehl `beschriftungsgroesse`

Listing 4.32: Der Befehl `beschriftungsgroesse`

```

1 beschriftungsgroesse "huge"
2 grenzgitter x=0,6 y=0,4
3 achsen
4 beschriftung farbe=blue p=1,2 text="$90^{\circ}$ Drehung",90,0
5 beschriftung p=4,3 text="$45^{\circ}$",45,45
6 beschriftung p=5,2 text="$(5|2)$",0,225

```

Beschriftungen von Funktionen

Eine Erweiterung des Befehls `beschriftung` ist, dass man anstelle am Punkt $(x|y)$ auch an der Stelle $(x|f(x))$ einen Text zeichnen lassen kann, indem man die Funktion f übergibt.

`beschriftung`

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
p=<x ∈ ℝ>,<"Funktionsterm f(x)">
text="<Text>"[,<Textwinkel in Grad>,<Drehung in Grad>[,<Δy ∈ ℝ>]]

```

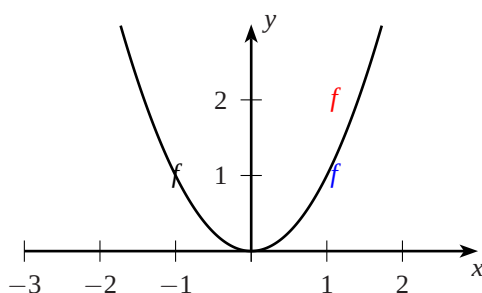


Abb. 4.39.: Der Befehl `beschriftung`

Listing 4.33: Der Befehl `beschriftung`

```

1 achsen
2 graph x=-3,3 y=0,3 x^2
3 beschriftung p=-1,"x^2" text="$f$"
4 beschriftung farbe=blue p=1,"x^2" text="$f$",0,0
5 beschriftung farbe=red p=1,"x^2" text="$f$",0,1

```

4.4.2. Punkte zeichnen

Allgemeine Punkte

Möchte man einen **Koordinatenpunkt** $(x|y)$ mit Beschriftung versehen, so gibt es den Befehl `punkt`, an den man einen Text übergeben kann. Die Syntax sieht wie folgt aus:

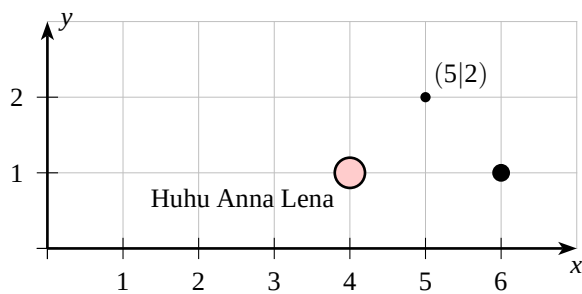
`punkt`

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ_{>0})>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<Radius r ∈ ℝ_{>0} des Punktes>
[text="<Text>"[,<Drehung in Grad>]]

```

Der Radius eines Punktes ist standardmäßig auf 0,05 gesetzt.

Abb. 4.40.: Der Befehl `punkt`Listing 4.34: Der Befehl `punkt`

```

1 grenzgitter x=0,7 y=0,3
2 punkt farbe=black,solid,red!20 linie=1,
   solid parameter="" p=4,1 r=0.2 text="
   Huhu Anna Lena",225
3 punkt p=5,2 text="$(5|2)$"
4 punkt p=6,1 r=0.1

```

Punkte an Funktionen

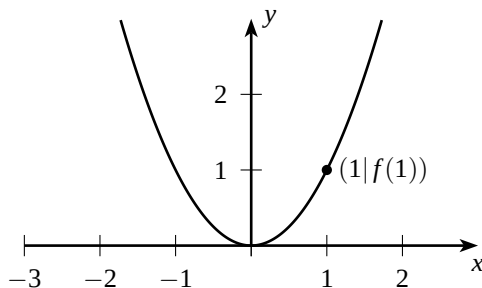
Der Befehl `punkt` ist ebenfalls mit der Möglichkeit einen Funktionsterm zu übergeben erweitert worden.

`punkt`

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt ( $l \in \mathbb{R}_{>0}$ )>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<"Funktionsterm  $f(x)$ ">
r=<Radius  $r \in \mathbb{R}_{>0}$  des Punktes>
[text="<Text>"[,<Drehung in Grad>]]

```

Abb. 4.41.: Der Befehl `punkt` mit einem FunktionsaufrufListing 4.35: Der Befehl `punkt` mit einem Funktionsaufruf

```

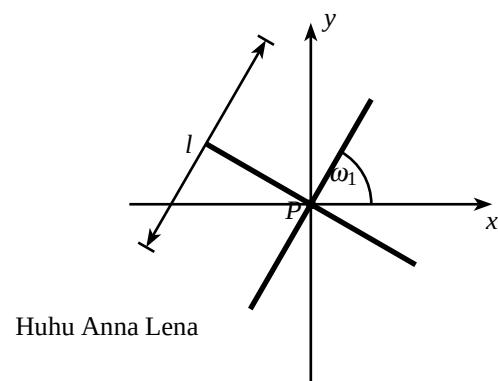
1 achsen
2 graph x=-3,3 y=0,3 x^2
3 punkt p=1,"x^2" text="$(1|f(1))$",0

```

4.4.3. Kreuze zeichnen

Allgemeine Kreuze

Möchte man anstelle eines Koordinatenpunkt $(x|y)$ ein **Kreuz** zeichnen, so gibt es den Befehl `kreuz`, an den man ebenfalls einen Text übergeben kann.

Abb. 4.42.: Die Definition des Befehls `kreuz`

Die Syntax sieht wie folgt aus:

kreuz

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<Länge des Kreuzes l ∈ ℝ>0>,<Drehwinkel des Kreuzes in Grad>]
[text="<Text>"[,<Drehung in Grad>]]
```

Standardmäßig ist die Länge des Kreuzes auf 0,5 LE gesetzt und der Drehwinkel beträgt 45°. Die Drehung des Textes ist als Standard ebenfalls 45°.

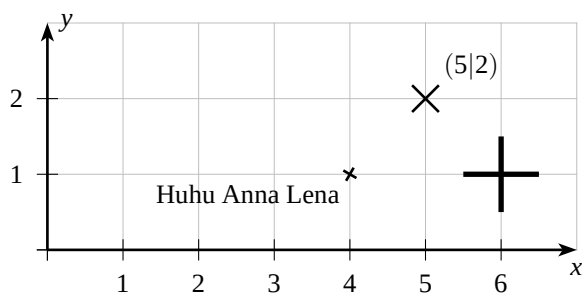


Abb. 4.43.: Der Befehl `kreuz`

Listing 4.36: Der Befehl `kreuz`

```
1 grenzgitter x=0,7 y=0,3
2 achsen
3 kreuz farbe=black linie=1,solid
   parameter="" p=4,1 r=0.2,30 text="
   Huhu Anna Lena",225
4 kreuz p=5,2 text="$(5|2)$"
5 kreuz linie=2,solid p=6,1 r=1,0
```

Kreuze an Funktionen

Der Befehl `kreuz` ist ebenfalls mit der Möglichkeit einen Funktionsterm zu übergeben erweitert worden.

kreuz

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<"Funktionsterm f(x)">
r=<Länge des Kreuzes l ∈ ℝ>0>,<Drehwinkel des Kreuzes in Grad>]
[text="<Text>"[,<Drehung in Grad>]]
```

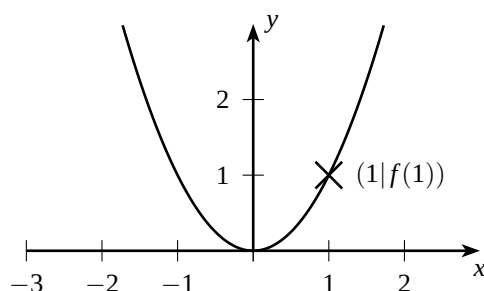


Abb. 4.44.: Der Befehl `kreuz` an dem Graphen $f(x) = x^2$

Listing 4.37: Der Befehl `kreuz` an dem Graphen $f(x) = x^2$

```
1 achsen
2 graph x=-3,3 y=0,3 x^2
3 kreuz p=1,"x^2" text="$(1|f(1))$",0
```

4.5. Grafiken einfügen

Möchte man eine **Grafik** in eine Zeichnung mit einfügen, so gibt es den Befehl `bild`, der einem dies ermöglicht. Die Syntax ist dabei die folgende:

bild

```
[layer=<n ∈ [0;100]>]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<Breite des Bildes l ∈ ℝ>>0>[,<Drehwinkel des Bildes in Grad>]
<"Dateiname">
```

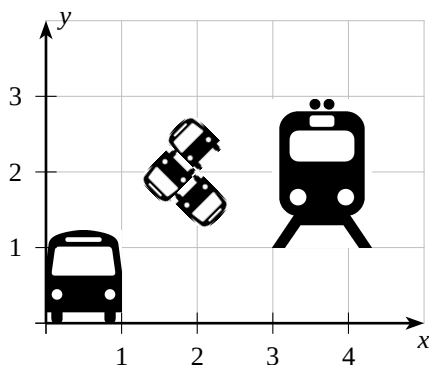


Abb. 4.45.: Eine Grafik einfügen

Listing 4.38: Eine Grafik einfügen

```
1 grenzgitter x=0,5 y=0,4
2 bild p=0,0 r=1 Abbildungen/bus.eps
3 bild p=2,2 r=0,5,45 Abbildungen/bus.eps
4 bild p=2,2 r=0,5,135 Abbildungen/bus.eps
5 bild p=2,2 r=0,5,225 Abbildungen/bus.eps
6 bild p=3,1 r=1,3 Abbildungen/zug.eps
```

4.6. Weitere Zeichenelemente

4.6.1. Winkel und Winkelbezeichnungen

Da man für die Geometrie ja auch Winkelbögen und Beschriftungen benötigt gibt es den Befehl `winkel`, mit dem man dies zeichnen kann.

winkel

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>>0>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ> p=<x2 ∈ ℝ>,<y2 ∈ ℝ> p=<x3 ∈ ℝ>,<y3 ∈ ℝ>
r=<Radius r ∈ ℝ>>0 des Bogens>,<a l i>
text="<Text>"
```

Dabei steht der <Radius und i stehen dafür, ob der Bogen gegen oder mit dem Uhrzeigersinn gezeichnet werden soll.

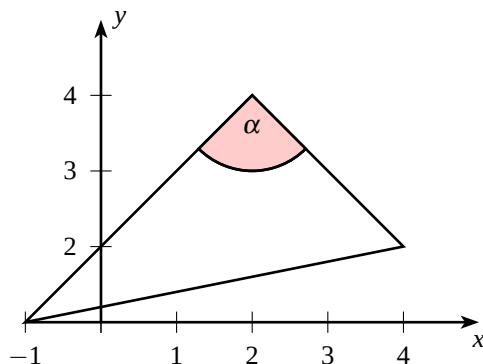


Abb. 4.46.: Ein Winkel

Listing 4.39: Ein Winkel

```
1 achsen
2 dreieck linie=1,solid parameter="" p=-1,1 p=2,4 p=4,2
3 winkel farbe=black,solid,red!20 linie=1,solid p=-1,1 p=2,4 p=4,2 r=1,i text="$\alpha$"
```

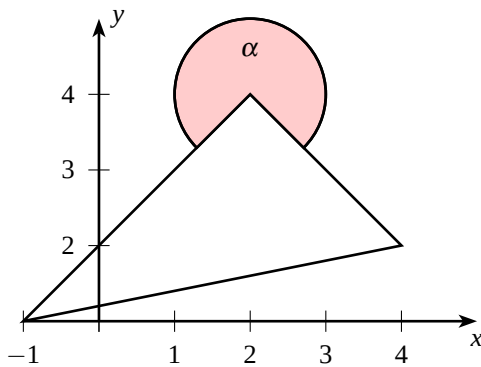


Abb. 4.47.: Ein weiterer Winkel

Listing 4.40: Ein weiterer Winkel

```

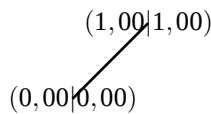
1 achsen
2 dreieck p=-1,1 p=2,4 p=4,2
3 winkel farbe=black,solid,red!20 linie=1,solid p
   =-1,1 p=2,4 p=4,2 r=1,a text="$\alpha$"

```

4.6.2. Knoten

Knoten anzeigen

Ein Hilfskonstrukt ist der Befehl `knoten`, der einem alle Positionen anzeigt. Dies hilft einem, wenn Positionen berechnet werden und man diese mit weiteren Zeichenobjekten verbunden werden sollen.

Abb. 4.48.: Der Befehl `knoten`Listing 4.41: Der Befehl `knoten`

```

1 knoten
2 linie farbe=black linie=1,solid p=0,0 p=1,1

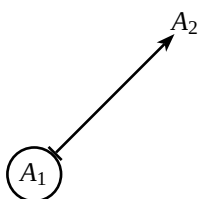
```

Knoten definieren und verbinden

Anstelle von Koordinatenpunkten kann man dem Befehl `linie` auch Knoten definieren und an Linien die Namen dieser Knoten übergeben. Hierbei habe ich mir [ZANDT et al., 2011] zu Hilfe genommen. Dieser Abschnitt ist aber insgesamt noch sehr rudimentär und muss noch fertig gestellt werden. Hier nur die Anfänge meiner Arbeit:

Mit dem Befehl `knoten` kann man sich einen Punkt P definieren. Dieser kann mit dem Befehl `linie` oder `kreis` verbunden werden. Dies soll noch deutlich erweitert werden.

Beispiele Hier einmal ein paar Beispiele:

Abb. 4.49.: Erstes Beispiel für den Befehl `knoten`Listing 4.42: Erstes Beispiel für den Befehl `knoten`

```

1 knoten parameter="fillstyle=solid" p=0,0 r="
   $A_1$","A1","cnodeput"
2 knoten parameter="fillstyle=solid" p=2,2 r="
   $A_2$","A2","rnode"
3 linie farbe=black linie=1,solid,"| ->" p="A1"
   p="A2"

```

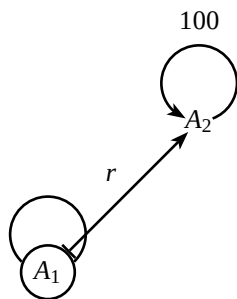


Abb. 4.50.: Zweites Beispiel für den Befehl `knoten`

Listing 4.43: Zweites Beispiel für den Befehl `knoten`

```

1 knoten parameter="fillstyle=solid" p=0,0 r="
  $A_1$","A1","cnodeput"
2 knoten parameter="fillstyle=solid" p=2,2 r="
  $A_2$","A2","rnode"
3 linie farbe=black linie=1,solid,"l ->" p="A1"
  p="A2" text="$r$"
4 kreis farbe=black,solid,red!20 linie=1,solid
  parameter="" p="A1" r=0.5
5 kreis farbe=black,solid,red!20 linie=1,solid,"
  ->" parameter="" p="A2" r=0.5 text="100"
  ,0

```

4.6.3. Neue Seite

Möchte man eine pdf-Datei mit mehreren Seiten erstellen, so gibt es den Befehl `neuesbild`, der eine neue Seite erstellt.

Die Syntax ist

neuesbild

[layer=< $n \in [0;100]$ >]

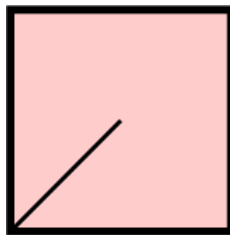


Abb. 4.51.: Seitenwechsel, erste Seite

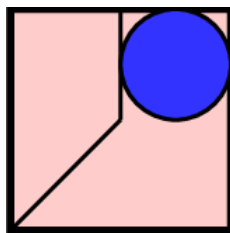


Abb. 4.52.: Seitenwechsel, zweite Seite

Listing 4.44: Der Befehl `neuesbild`

```

1 rechteck farbe=black,solid,red!20 linie=2,
  solid p=0,0 r=2,2
2 linie p=0,0 p=1,1
3 neuesbild
4 rechteck farbe=black,solid,red!20 linie=2,
  solid p=0,0 r=2,2
5 linie p=0,0 p=1,1
6 linie p=1,1 p=1,2
7 kreis farbe=black,solid,blue!80 p=1.5,1.5 r=0.5

```

5. Koordinatensysteme und Gitter

Mein Skript bietet die Möglichkeiten Koordinatensysteme und Gitter zu zeichnen. In diesem Abschnitt werden die einzelnen Möglichkeiten erläutert.

5.1. Einfache Koordinatensysteme

Möchte man ein **Koordinatensystem** zeichnen, so kann man dies natürlich mit einfachen Pfeilen erstellen, aber es gibt auch das Makro `koordinatensystem`, welches wie folgt aufgerufen wird:

koordinatensystem

```
[layer=<n ∈ [0;100]>]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
y=<ymin ∈ ℝ>,<ymax ∈ ℝ>
einteilung=<xSkala ∈ ℝ>0Skala ∈ ℝ>0>0)>,<Linienart>[,<Linienfarbe>]]
skala=<keine|nurstriche|nureins|normal|strichundeins>
text="Text an der x-Achse","Text an der y-Achse"
[mitte]
```

Dabei stehen die x und y Werte für die Ausmaße des Koordinatensystems. Die Einteilung steht für die Einteilung auf den jeweiligen Achsen und `skala` beschreibt, wie diese Einteilung gezeichnet werden soll. Ist `mitte` gesetzt, so wird die Achsenbeschriftung in die Mitte der positiven Achsenabschnitte des Koordinatensystems gezeichnet. Zu guter letzt werden die Beschriftungen in x und y -Richtung gesetzt.

Soll das Koordinatensystem nicht den Ursprung $(0|0)$ als Schnittpunkt haben, so gibt es den Befehl `achsenschnitt`, der mit der folgenden Syntax aufgerufen wird:

achsenschnitt

```
x=<xSchnitt ∈ ℝ>
y=<ySchnitt ∈ ℝ>
```

Als Erklärung siehe die folgende Abbildung 5.1.

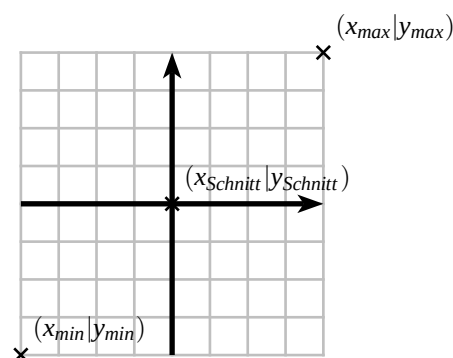


Abb. 5.1.: Bezugspunkte beim Zeichnen von Koordinatenachsen

5.1.1. Einige Beispiele

Hier nun ein paar einfache Beispiele für Koordinatenachsen:

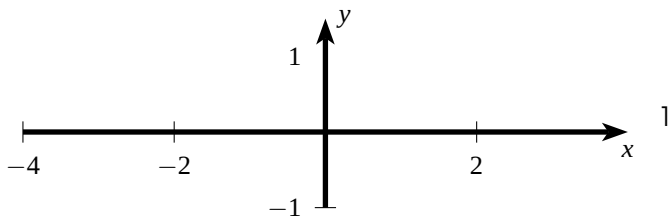


Abb. 5.2.: Ein erstes Koordinatensystem

Listing 5.1: Ein erstes Koordinatensystem

```
1 koordinatensystem x=-4,4 y=-1,1.5
   einteilung=2,1 linie=2,solid,black
   skala=normal text="$x$","$y$"
```



Abb. 5.3.: Ein zweites Koordinatensystem

Listing 5.2: Ein zweites Koordinatensystem

```
1 achsenschnitt x=2 y=1
2 koordinatensystem x=-4,4 y=-1,2
   einteilung=2,1 linie=2,solid,black,"-
   " skala=normal text="$x$","$y$"
```

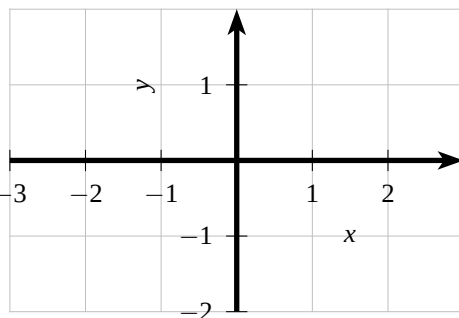


Abb. 5.4.: Ein Koordinatensystem mit dem Befehl `mitte`

Listing 5.3: Ein Koordinatensystem mit dem Befehl `mitte`

```
1 gitter
2 koordinatensystem x=-3,3 y=-2,2
   einteilung=1,1 linie=2,solid,black,"
   ->" skala=normal text="$x$","$y$"
   mitte
```

5.2. Weitere Achsentypen

Es gibt auch noch weitere Achsentypen. Dazu gehören die Befehle `xachse` und `yachse`. Die Syntax ist:

xachse

```
[layer=<n ∈ [0;100]>]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>,<xskala ∈ ℝ>>0
[linie=<Linienstärke in pt (l ∈ ℝ>>0)>,<Linienart>[,<Linienfarbe>]]
[skala=<keine|nurstriche|nureins|normal|strichundeins>]
text="<x-Achsenbeschriftung>"
```

und

yachse

```
[layer=<n ∈ [0;100]>]
y=<ymin ∈ ℝ>,<ymax ∈ ℝ>,<yskala ∈ ℝ>>0
[linie=<Linienstärke in pt (l ∈ ℝ>>0)>,<Linienart>[,<Linienfarbe>]]
[skala=<keine|nurstriche|nureins|normal|strichundeins>]
text="<y-Achsenbeschriftung>"
```

Werden beide Befehle in eine Datei geschrieben, so wird wieder ein reguläres Koordinatensystem gezeichnet.

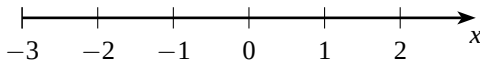


Abb. 5.5.: Eine x-Achse

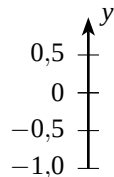


Abb. 5.6.: Eine y-Achse

Listing 5.4: Eine x-Achse

```
1 xachse x=-3,3,1 skala=normal text="$x$"
```

Listing 5.5: Eine y-Achse

```
1 yachse y=-1,1,0.5 skala=normal text="$y$"
```

5.3. Erweiterung der Achsen mit Teilstrichen und Pfeilarten

Als Erweiterung gibt es noch den Befehl `ticks`, der die Anzahl der Teilstriche auf der x - und y -Achse festsetzt. Der Befehl `pfeile` gibt die Pfeilart an den Achsen vor.

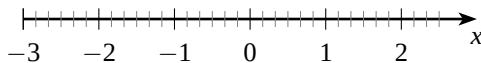


Abb. 5.7.: Eine x-Achse mit weiteren Teilstrichen

Listing 5.6: Eine x-Achse mit weiteren Teilstrichen

```
1 ticks 5
2 xachse x=-3,3,1 skala=normal text="$x$"
```

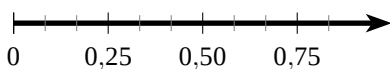


Abb. 5.8.: Eine x-Achse ohne Pfeile

Listing 5.7: Eine x-Achse ohne Pfeile

```
1 groesse x=5
2 ticks 2
3 pfeile -
4 xachse x=0,1,0.25 linie=2,solid,black
   skala=normal text=""
```

Diese Befehle können für jede Achsenart gesetzt werden.

5.4. Trigonometrische Achsenbeschriftungen

Für trigonometrische Funktionen gibt es die Möglichkeit auch entsprechende Achsen zu zeichnen. Hierbei wird mit dem Befehl `xyachserad` an der x -Achse die trigonometrische Beschriftung vorgenommen. Die Syntax ist die folgende:

xyachserad

```
[layer=<n ∈ [0;100]>]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
y=<ymin ∈ ℝ>,<ymax ∈ ℝ>
einteilung=<xSkala ∈ ℝ>0>,<ySkala ∈ ℝ>0>[,<rad | deg | degtorad | radtodeg>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Pfeiltyp>]]
[skala=<keine | nurstriche | nureins | normal | strichundeins>]
text=<"x-Achsenbeschriftung">,<"y-Achsenbeschriftung">
```

Bei einer trigonometrischen Achse kann man selbst entscheiden, ob man die x -Achse in Grad oder Bogenmaß angeben möchte. Standardmäßig ist die Achse auf Grad gesetzt und die Beschriftung auf Bogenmaß.

5.4.1. Beispiele

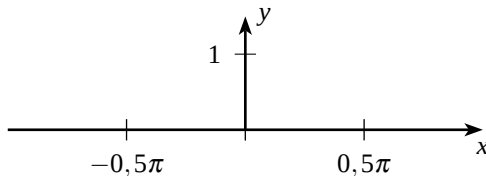


Abb. 5.9.: Ein trigonometrisches Koordinatensystem

Listing 5.8: Ein trigonometrisches Koordinatensystem

```
1 xyachserad x=-180,180 y=0,1.5
   einteilung=90,1 linie=1,solid,black
   skala=normal text="$x$","$y$"
```

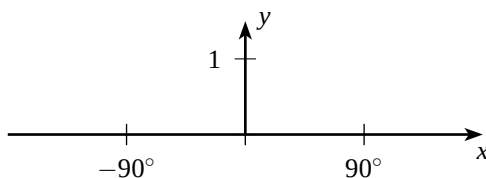


Abb. 5.10.: Der Befehl deg

Listing 5.9: Der Befehl deg

```
1 xyachserad x=-180,180 y=0,1.5
   einteilung=90,1,deg linie=1,solid,
   black skala=normal text="$x$","$y$"
```

Eine weitere Möglichkeit ist der Befehl `xachserad`, der es ermöglicht nur eine x -Achse mit trigonometrischer Einteilung zu zeichnen. Die Syntax ist

xachserad

```
[layer=<n ∈ [0;100]>]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>,<xskala ∈ ℝ>0>
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Pfeiltyp>]]
text="<x-Achsenbeschriftung>"
```

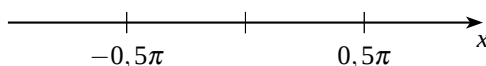


Abb. 5.11.: Der Befehl `xachserad`

Listing 5.10: Der Befehl `xachserad`

```
1 xachserad x=-180,180,90 text="$x$"
```

Den Sourcecode für die Koordinatensysteme findet man im Anhang unter A.2.

5.5. Achsenbeschriftung

5.5.1. Punkte an den Achsen

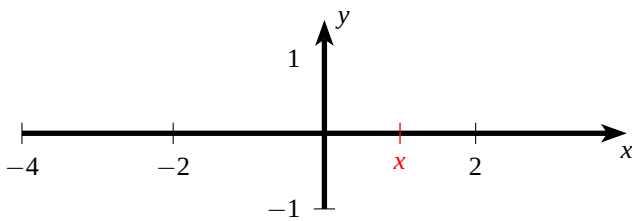
Möchte man einen weiteren Punkt mit an die Achse schreiben, so gibt es die beiden Befehle `xachsenpunkt` und `yachsenpunkt`, die die folgende Syntax haben:

xachsenpunkt

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
x=<x ∈ ℝ>
text="<Text an der Stelle (x|0)>"
```

yachsenpunkt

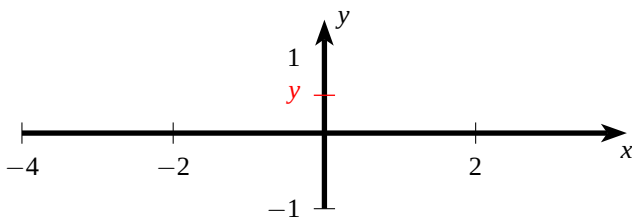
```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
y=<y ∈ ℝ>
text="<Text an der Stelle (0|y)>"
```

Abb. 5.12.: Der Befehl `xachsenpunkt`Listing 5.11: Der Befehl `xachsenpunkt`

```

1 koordinatensystem x=-4,4 y=-1,1.5
  einteilung=2,1 linie=2,solid,black
  skala=normal text="$x$","$y$"
2 xachsenpunkt farbe=red x=1 text="$x$"

```

Abb. 5.13.: Der Befehl `yachsenpunkt`Listing 5.12: Der Befehl `yachsenpunkt`

```

1 koordinatensystem x=-4,4 y=-1,1.5
  einteilung=2,1 linie=2,solid,black
  skala=normal text="$x$","$y$"
2 yachsenpunkt farbe=red y=0.5 text="$y$"

```

Analog zu den eben genannten Befehlen kann man mit `xachsenpunktohne` und `yachsenpunktohne` die Achsen beschriften ohne einen Teilstrich. Die Syntax ist:

xachsenpunktohne

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
x=<x ∈ ℝ>
text="<Text an der Stelle (x|0)>"

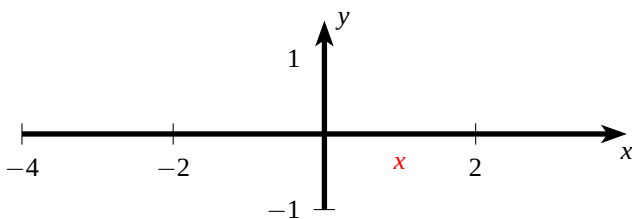
```

yachsenpunktohne

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
y=<y ∈ ℝ>
text="<Text an der Stelle (0|y)>"

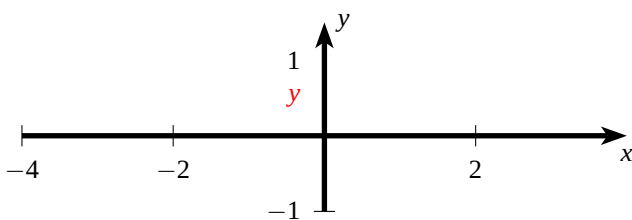
```

Abb. 5.14.: Der Befehl `xachsenpunktohne`Listing 5.13: Der Befehl `xachsenpunktohne`

```

1 koordinatensystem x=-4,4 y=-1,1.5
  einteilung=2,1 linie=2,solid,black
  skala=normal text="$x$","$y$"
2 xachsenpunktohne farbe=red x=1 text="
  $x$"

```

Abb. 5.15.: Der Befehl `yachsenpunktohne`Listing 5.14: Der Befehl `yachsenpunktohne`

```

1 koordinatensystem x=-4,4 y=-1,1.5
  einteilung=2,1 linie=2,solid,black
  skala=normal text="$x$","$y$"
2 yachsenpunktohne farbe=red y=0.5 text="
  $y$"

```

5.5.2. Pfeile an den Achsen

Eine weitere Kennzeichnungsmöglichkeit sind Pfeile, die mit den Befehlen `xpfeil` und `ypfeil` möglich sind.

xpfeil

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
p=<x ∈ ℝ>[,<Pfeillänge dy ∈ ℝ>]
text="<Text an der Stelle (x|dy)>"
```

ypfeil

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
p=[<Pfeillänge dx ∈ ℝ>,<y ∈ ℝ>]
text="<Text an der Stelle (dx|y)>"
```

Standardmäßig sind die Pfeile eine Einheit lang.

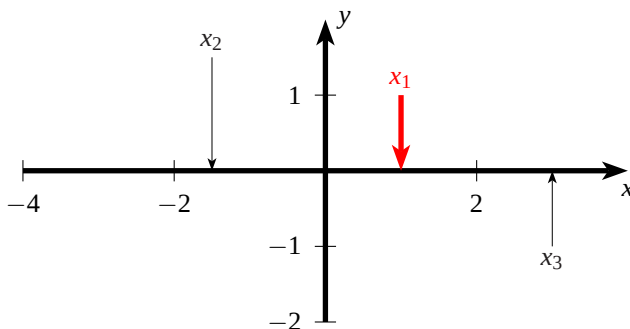


Abb. 5.16.: Der Befehl `xpfeil`

Listing 5.15: Der Befehl `xpfeil`

```
1 koordinatensystem x=-4,4 y=-2,2
   einteilung=2,1 linie=2,solid,black
   skala=normal text="$x$","$y$"
2 xpfeil farbe=red linie=2,solid p=1 text
 ="$x_1$"
3 xpfeil p=-1.5,1.5 text="$x_2$"
4 xpfeil p=3,-1 text="$x_3$"
```

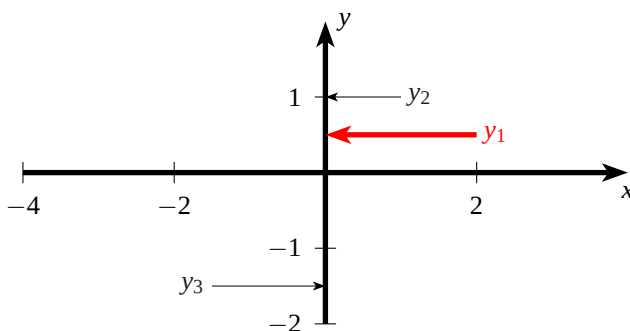


Abb. 5.17.: Der Befehl `ypfeil`

Listing 5.16: Der Befehl `ypfeil`

```
1 koordinatensystem x=-4,4 y=-2,2
   einteilung=2,1 linie=2,solid,black
   skala=normal text="$x$","$y$"
2 ypfeil farbe=red linie=2,solid p=2,0.5
  text="$y_1$"
3 ypfeil p=1 text="$y_2$"
4 ypfeil p=-1.5,-1.5 text="$y_3$"
```

5.5.3. Pfeile an den Achsen mit Kästen

Eine weitere Kennzeichnungsmöglichkeit sind Pfeile, die mit den Befehlen `xpfeilkasten` und `ypfeilkasten` möglich sind.

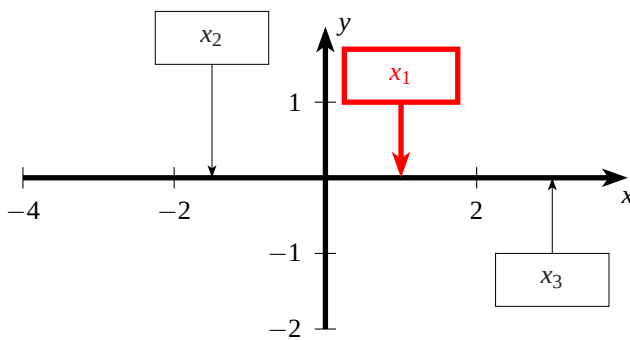
xpfeilkasten

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
p=<x ∈ ℝ>[,<Pfeillänge dy ∈ ℝ>]
text="<Text an der Stelle (x|dy)>"
```

ypfeilkasten

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
p=[<Pfeillänge dx ∈ ℝ>,<y ∈ ℝ>]
text="<Text an der Stelle (dx|y)>"
```

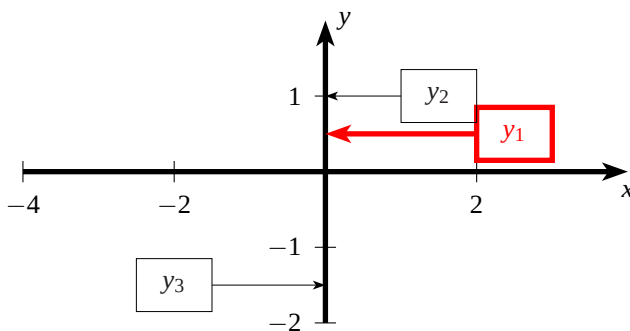
Standardmäßig sind die Pfeile eine Einheit lang.

Abb. 5.18.: Der Befehl `xpfeilkasten`Listing 5.17: Der Befehl `xpfeilkasten`

```

1 koordinatensystem x=-4,4 y=-2,2
  einteilung=2,1 linie=2,solid,black
  skala=normal text="$x$","$y$"
2 xpfeilkasten farbe=red linie=2,solid p
  =1 text="$x_1$"
3 xpfeilkasten p=-1,5,1,5 text="$x_2$"
4 xpfeilkasten p=3,-1 text="$x_3$"

```

Abb. 5.19.: Der Befehl `ypfeilkasten`Listing 5.18: Der Befehl `ypfeilkasten`

```

1 koordinatensystem x=-4,4 y=-2,2
  einteilung=2,1 linie=2,solid,black
  skala=normal text="$x$","$y$"
2 ypfeilkasten farbe=red linie=2,solid p
  =2,0,5 text="$y_1$"
3 ypfeilkasten p=1 text="$y_2$"
4 ypfeilkasten p=-1,5,-1,5 text="$y_3$"

```

5.6. Gitter zeichnen

Es sind mehrere Gittertypen in meinem Skript realisiert. Als eines der einfachsten ist dies der Befehl `kariert`, mit dem ein einfaches Gitter gezeichnet werden kann. Die Syntax lautet

kariert

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
dx=<dx1 ∈ ℝ>0>,<dx2 ∈ ℝ>0>,<dx3 ∈ ℝ>0>
dy=<dy1 ∈ ℝ>0>,<dy2 ∈ ℝ>0>,<dy3 ∈ ℝ>0>
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
y=<ymin ∈ ℝ>,<ymax ∈ ℝ>

```

Dabei geben die Werte in `dx` und `dy` an, in welchen Abständen Linien gezeichnet werden sollen. Sind die hinteren beiden Werte auf Werte ungleich Null gesetzt, so werden Linien zwei- und dreimal so dick wie die definierte Linienstärke gezeichnet.

5.6.1. Beispiele

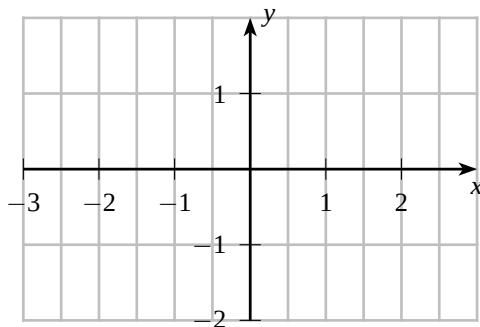


Abb. 5.20.: Der Befehl kariert

Listing 5.19: Der Befehl kariert

```
1 achsen
2 kariert farbe=lightgray linie=1,solid dx
   =0.5,0,0 dy=1,0,0 x=-3,3 y=-2,2
```

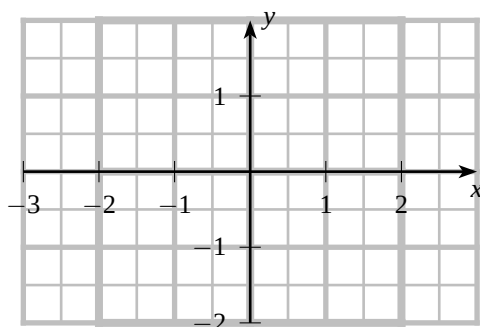


Abb. 5.21.: Ein weiteres Gitter

Listing 5.20: Ein weiteres Gitter

```
1 achsen
2 kariert farbe=lightgray linie=1,solid dx
   =0.5,1,2 dy=0.5,1,2 x=-3,3 y=-2,2
```

Setzt man übrigens `gitter` in eine Datei, wo bereits ein Koordinatensystem gezeichnet wird, so wird ein Gitter mit den gleichen Ausmaßen wie das Koordinatensystem gezeichnet. Analog gibt es noch den Befehl `matheheft`.

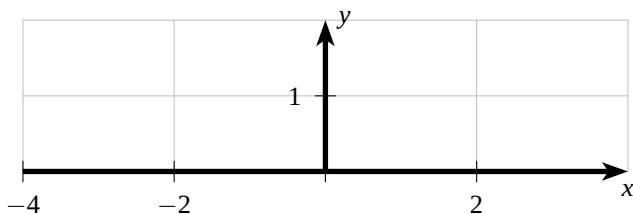


Abb. 5.22.: Der Befehl gitter

Listing 5.21: Der Befehl gitter

```
1 gitter
2 koordinatensystem x=-4,4 y=0,2
   einteilung=2,1 linie=2,solid,black
   skala=normal text="$x$","$y$"
```

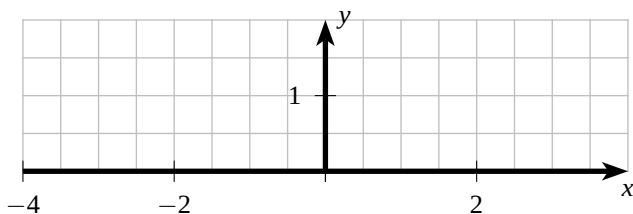


Abb. 5.23.: Der Befehl matheheft

Listing 5.22: Der Befehl matheheft

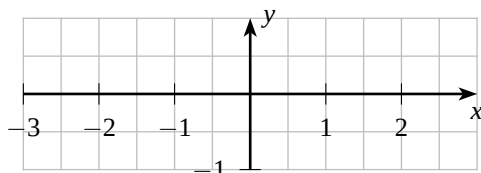
```
1 matheheft
2 koordinatensystem x=-4,4 y=0,2
   einteilung=2,1 linie=2,solid,black
   skala=normal text="$x$","$y$"
```

5.7. Matheheft und Millimeterpapier

Als weitere Befehle gibt es noch das `matheheft` und `millimeterpapier`, welches einem einmal ein kariertes Muster in 0,5 cm Abständen liefert und das andere ein Blatt Millimeterpapier. Die Syntax ist

matheheft

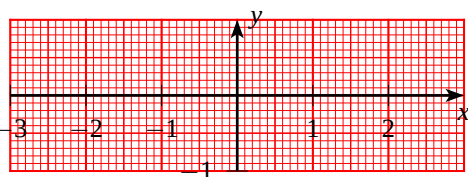
```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ> y=<ymin ∈ ℝ>,<ymax ∈ ℝ>
```

Abb. 5.24.: Der Befehl `matheheft`**millimeterpapier**

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ> y=<ymin ∈ ℝ>,<ymax ∈ ℝ>
```

Listing 5.23: Der Befehl `matheheft`

```
1 achsen
2 matheheft farbe=lightgray x=-3,3 y=-1,1
```

Abb. 5.25.: Der Befehl `millimeterpapier`Listing 5.24: Der Befehl `millimeterpapier`

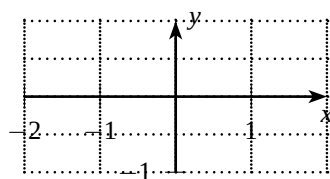
```
1 achsen
2 millimeterpapier farbe=red x=-3,3 y=-1,1
```

5.8. Grids

Zu guter Letzt gibt es noch die Befehle `grid` und `punkte`. Diese erzeugen einmal ein Gitter mit Unterpunkten und zum anderen ein reines Punktegitter. Die Syntax ist:

grid

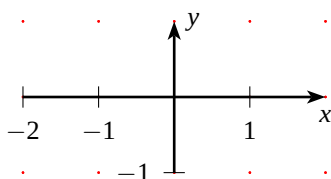
```
[layer=<n ∈ [0;100]>] [farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
dx=<Punkteabstand dx ∈ ℝ>0>,<Unterpunkteanzahl in x-Richtung nx ∈ ℕ>
dy=<Punkteabstand dy ∈ ℝ>0>,<Unterpunkteanzahl in y-Richtung ny ∈ ℕ>
x=<xmin ∈ ℝ>,<xmax ∈ ℝ> y=<ymin ∈ ℝ>,<ymax ∈ ℝ>
```

Abb. 5.26.: Der Befehl `grid`Listing 5.25: Der Befehl `grid`

```
1 achsen
2 grid farbe=black linie=1,solid,black dx=1,10 dy=0,5,4 x=-2,2 y=-1,1
```

punkte

```
[layer=<n ∈ [0;100]>] [farbe=<Linienfarbe>] abstand=<Punkteabstand d ∈ ℝ>0>
x=<xmin ∈ ℝ>,<xmax ∈ ℝ> y=<ymin ∈ ℝ>,<ymax ∈ ℝ>
```

Listing 5.26: Der Befehl `punkte`

```
1 achsen
2 punkte farbe=red abstand=1 x=-2,2 y=-1,1
```

Abb. 5.27.: Der Befehl `punkte`

6. Plotten von Funktionen und Daten

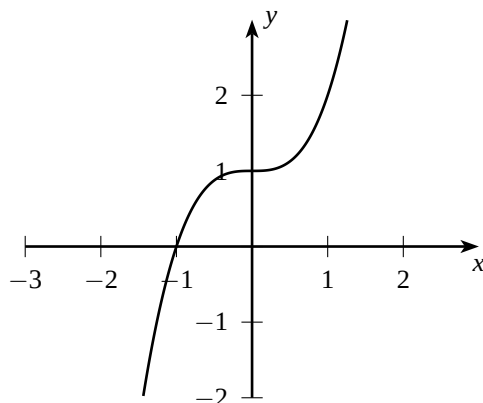
In meinem Skript habe ich die Möglichkeit realisiert Funktionen und Daten zu zeichnen. Dabei werden im Hintergrund einige Berechnungen angestellt, die einem das Leben vereinfachen.

6.1. Funktionen zeichnen

Um Funktionen zu zeichnen habe ich den Befehl `graph` geschrieben. Die Syntax ist die folgende:

```
graph
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
  [parameter="<Pstricksbefehle>"]
  x=<xmin ∈ ℝ>,<xmax ∈ ℝ> [y=<ymin ∈ ℝ>,<ymax ∈ ℝ>]
  <Funktionsterm f(x)>
```

Hierbei wird der Funktionsterm mit der Variablen x in der algebraischen Notation erwartet. Sind y_{min} und y_{max} nicht gesetzt, so wird die Funktion in den Grenzen x_{min} und x_{max} gezeichnet. Ist dies angegeben, so ist die Funktion auf diesen Bereich beschränkt.



Listing 6.1: Der Befehl `graph`

```
1 achsen
2 graph layer=50 farbe=black linie=1,solid
   parameter="" x=-3,3 y=-2,3 x^3+1
```

Abb. 6.1.: Der Befehl `graph`

Da der Funktionsterm in die umgekehrt-polnische-Notation (UPN) übersetzt wird, ist es sinnvoll den Term vollständig zu klammern. Ansonsten kann man auch als `PParameter algebraic` angeben, dann erfolgt diese Übersetzung nicht.

6.2. Markierungen an Graphen

Möchte man eine Markierung an einer Funktion oder eine generelle Markierung von einem Wertepaar $(x|y)$ zeichnen, so steht einem der Befehl `graphmarkierung` zur Verfügung.

```
graphmarkierung
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>,"<x | y | xy>"]
  [parameter="<Pstricksbefehle>"]
  x=<x ∈ ℝ>
```

[$y = \langle y \in \mathbb{R} \rangle$]
 [text=<"Text an der x-Achse">,<"Text an der y-Achse">]
 [<Funktionsterm $f(x)$ >] wenn **y** nicht definiert

Standardmäßig wird der Befehl `xy` bei der `linie` genutzt.

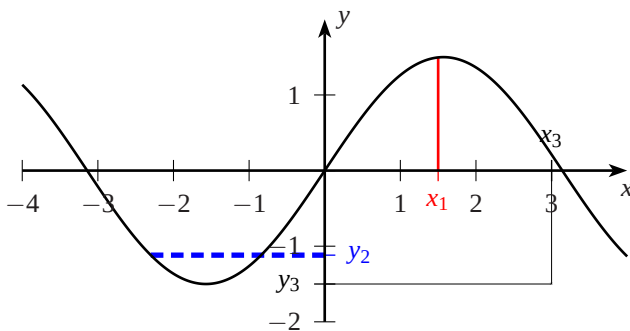


Abb. 6.2.: Der Befehl `graphmarkierung`

Listing 6.2: Der Befehl `graphmarkierung`

```

1 achsen
2 graph layer=50 farbe=black linie=1,
   solid parameter="" x=-4,4 y=-2,2 1.5*
   sin(x)
3 graphmarkierung farbe=red linie=1,
   solid,x x=1.5 text="$x_1$","$y_1$" 1.5*
   sin(x)
4 graphmarkierung farbe=blue linie=2,
   dashed,y x=-2.3 text="$x_2$","$y_2$"
   1.5*sin(x)
5 graphmarkierung x=3 y=-1.5 text="$x_3$
   ","$y_3$"
    
```

6.3. Ableitungen zeichnen

Um Ableitungen $f^{(n)}$ zu zeichnen gibt es den Befehl `ableitung`, der fast analog zum Befehl `graph` ist. Die Syntax lautet:

ableitung
 [layer=< $n \in [0; 100]$ >]
 [farbe=<Linienfarbe>]
 [**linie**=<Linienstärke in pt ($l \in \mathbb{R}_{>0}$)>,<Linienart>]
 [parameter="<Pstricksbefehle>"]
 $x = \langle x_{\min} \in \mathbb{R} \rangle, \langle x_{\max} \in \mathbb{R} \rangle$ [$y = \langle y_{\min} \in \mathbb{R} \rangle, \langle y_{\max} \in \mathbb{R} \rangle$]
 <Funktionsterm $f(x)$ >[,<Grad der Ableitung $n > 1$ >]

Hierbei kann der Grad der Ableitung angegeben werden. Standardmäßig ist der Grad auf 1 gesetzt. Zu beachten ist, dass wenn n größer 1 gewählt wird, kann die Funktion über die y -Achse hinaus gezeichnet werden.

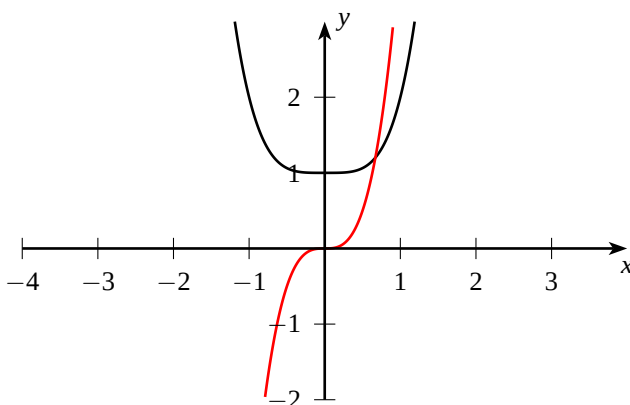


Abb. 6.3.: Der Befehl `ableitung`

Listing 6.3: Der Befehl `ableitung`

```

1 achsen
2 graph layer=50 farbe=black linie=1,
   solid parameter="" x=-4,4 y=-2,3 x
   ^4+1
3 ableitung layer=50 farbe=red linie=1,
   solid parameter="" x=-4,4 y=-2,3 x
   ^4+1
    
```

6.4. Tangenten und Tangentendreiecke

6.4.1. Zeichnen einer Tangente

Die Steigung einer Funktion in einem Punkt wird häufig durch das Zeichnen eines kurzen Tangentenstückes verdeutlicht. Dies kann durch die Ableitung der Funktion oder durch die Berechnung einer genügend kleinen Sekante geschehen. Die Gleichung einer Tangente an der Stelle x_0 für die Funktion f ist definiert durch

$$y = f(x_0) + f'(x_0) \cdot (x - x_0) \quad (6.1)$$

Eine **Tangente** kann nun mit dem Befehl `tangente` gezeichnet werden. Die Syntax ist dabei die folgende:

```
tangente
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
  [parameter="<Pstricksbefehle>"]
  x=<x0 ∈ ℝ>
  dx=<dx ∈ ℝ>0>
  <Funktionsterm f(x)>
```

Dabei ist x_0 die Stelle, für die die Tangente berechnet werden soll und dx die Länge der einzuzeichnenden Tangente.

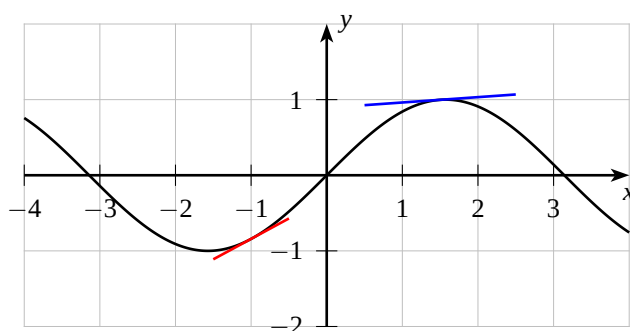


Abb. 6.4.: Der Befehl `tangente`

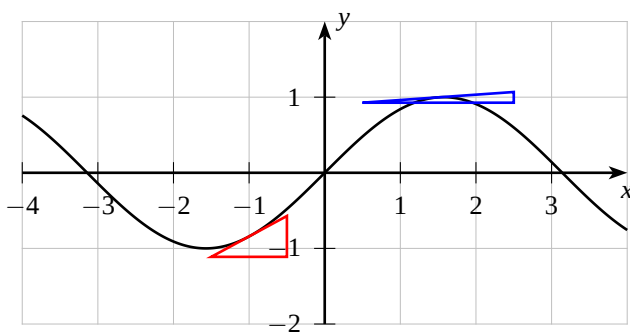
Listing 6.4: Der Befehl `tangente`

```
1 achsen
2 gitter
3 graph linie=1,solid x=-4,4 y=-2,2 sin(x)
4 tangente farbe=red linie=1,solid x=-1
   dx=1 sin(x)
5 tangente farbe=blue linie=1,solid x=1.5
   dx=2 sin(x)
```

6.4.2. Zeichnen eines Tangentendreiecks

Analog zur Tangente gibt es denn Befehl des **Tangentendreiecks**, der anstelle der Tangente ein vollständiges Steigungsdreieck einzeichnet.

```
tangentendreieck
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
  [parameter="<Pstricksbefehle>"]
  x=<x0 ∈ ℝ>
  dx=<dx ∈ ℝ>0>
  <Funktionsterm f(x)>
```


 Abb. 6.5.: Der Befehl `tangentendreieck`

 Listing 6.5: Der Befehl `tangentendreieck`

```

1  achsen
2  gitter
3  graph linie=1,solid x=-4,4 y=-2,2 sin(x)
4  tangentendreieck farbe=red linie=1,
    solid x=-1 dx=1 sin(x)
5  tangentendreieck farbe=blue linie=1,
    solid x=1.5 dx=2 sin(x)
    
```

6.5. Sekanten und Sekantendreiecke

6.5.1. Zeichnen von Sekanten

Eine Gerade die durch zwei verschiedene Punkte einer Funktion f geht, nennt man **Sekante**. Die Steigung der Sekante durch zwei verschiedene Punkte $(x_1|f(x_1))$ und $(x_2|f(x_2))$ des Graphen ist gegeben durch

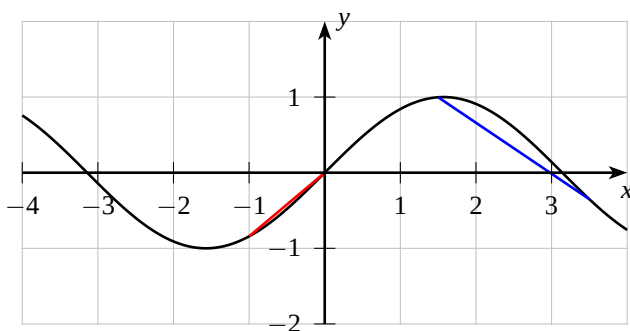
$$m = \frac{f(x_2) - f(x_1)}{x_2 - x_1} \quad (6.2)$$

Für mein Skript ist die Syntax der `sekante` die Folgende:

```

sekante
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
  [parameter="<Pstricksbefehle>"]
  x=<x1 ∈ ℝ>
  dx=<dx ∈ ℝ>0>
  <Funktionsterm f(x)>
    
```

Dabei ergibt sich x_2 durch $x_2 = x_1 + dx$.


 Abb. 6.6.: Der Befehl `sekante`

 Listing 6.6: Der Befehl `sekante`

```

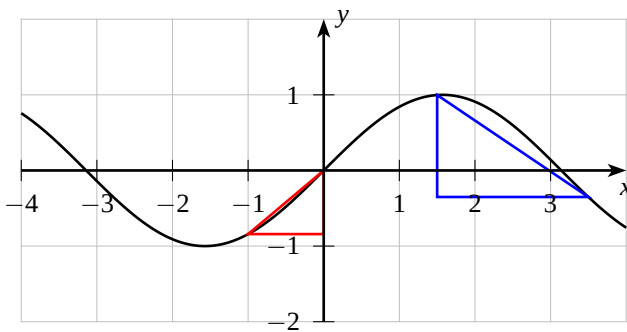
1  achsen
2  gitter
3  graph linie=1,solid x=-4,4 y=-2,2 sin(x)
4  sekante farbe=red linie=1,solid x=-1 dx
    =1 sin(x)
5  sekante farbe=blue linie=1,solid x=1.5
    dx=2 sin(x)
    
```

6.5.2. Zeichnen eines Sekantendreiecks

Analog zur Sekante gibt es denn Befehl des **Sekantendreiecks**, der anstelle der Sekante ein vollständiges Steigungsdreieck zeichnet.

sekantendreieck

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
x=<x1 ∈ ℝ>
dx=<dx ∈ ℝ>0>
<Funktionsterm f(x)>
```

Abb. 6.7.: Der Befehl `sekantendreieck`**Listing 6.7:** Der Befehl `sekantendreieck`

```
1 achsen
2 gitter
3 graph linie=1,solid x=-4,4 y=-2,2 sin(x)
4 sekantendreieck farbe=red linie=1,
   solid x=-1 dx=1 sin(x)
5 sekantendreieck farbe=blue linie=1,
   solid x=1.5 dx=2 sin(x)
```

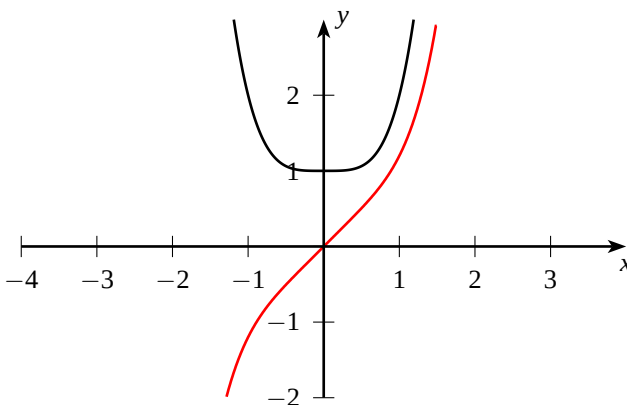
6.6. Stammfunktionen und Integrale**6.6.1. Stammfunktionen**

Um Stammfunktionen zu zeichnen gibt es den Befehl `stammfunktion`, der fast analog zum Befehl `graph` ist. Die Syntax lautet:

stammfunktion

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
[y=<ymin ∈ ℝ>,<ymax ∈ ℝ>]
[nullwert=<z ∈ ℝ>]
<Funktionsterm f(x)>
```

Dabei stellt `nullwert` die additive Konstante der Stammfunktion dar.

Abb. 6.8.: Der Befehl `stammfunktion`**Listing 6.8:** Der Befehl `stammfunktion`

```
1 achsen
2 graph linie=1,solid x=-4,4 y=-2,3 x^4+1
3 stammfunktion farbe=red linie=1,solid
   x=-4,4 y=-2,3 x^4+1
```

6.6.2. Flächen unter einer Funktion

Hinführend auf das Integral ergibt sich natürlich auch die **Obersumme** und **Untersumme** einer Funktion. Dabei betrachtet man die (beschränkte) Funktion $f : [a, b] \rightarrow \mathbb{R}$, wobei $a, b \in \mathbb{R}$ und $a < b$ gilt. Wird nun das Intervall $[a, b]$ in n Teile zerlegt mit $a = x_0 < x_1 < \dots < x_n = b$, so ist die Ober- bzw. Untersumme definiert als

$$O(Z) = \sum_{k=1}^n (x_k - x_{k-1}) \cdot \sup \{f(x) \mid x_k \leq x \leq x_{k+1}\} \quad (6.3)$$

$$U(Z) = \sum_{k=1}^n (x_k - x_{k-1}) \cdot \inf \{f(x) \mid x_k \leq x \leq x_{k+1}\} \quad (6.4)$$

Dies ist nun mit dem Befehl `oberuntersumme` realisiert. Die Syntax sieht dabei wie folgt aus:

```
oberuntersumme
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
[y=<ymin ∈ ℝ>,<ymax ∈ ℝ>]
n=<Einteilung n ∈ ℕ>
summen=<ober|unter|beide>
<Funktionsterm f(x)>
```

Dabei steht `ober` dafür, dass die Obersumme gezeichnet wird, `unter` zeichnet die Untersumme und `beide` zeichnet beide Treppenfunktionen. Ist `y` nicht gesetzt, so wird die Treppenfunktion zwischen `xmin` und `xmax` direkt gezeichnet. Ansonsten werden diese Werte aufgrund der `y`-Werte angepasst.

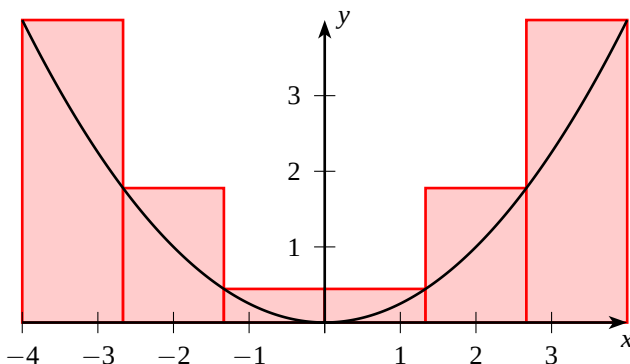


Abb. 6.9.: Die Obersumme von $f(x) = x^2$

Listing 6.9: Die Obersumme von $f(x) = x^2$

```
1 achsen
2 graph layer=61 farbe=black linie=1,
   solid x=-4,4 y=0,4 (0.5*x)^2
3 oberuntersumme farbe=red,solid,red!20
   linie=1,solid x=-4,4 y=0,4 n=6 summen
   =ober (0.5*x)^2
```

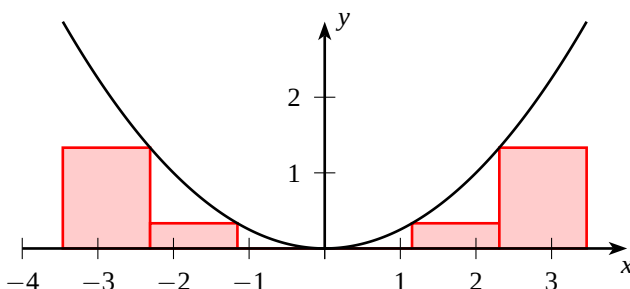


Abb. 6.10.: Die Untersumme von $f(x) = x^2$

Listing 6.10: Die Untersumme von $f(x) = x^2$

```
1 achsen
2 graph layer=61 farbe=black linie=1,
   solid x=-4,4 y=0,3 (0.5*x)^2
3 oberuntersumme farbe=red,solid,red!20
   linie=1,solid x=-4,4 y=0,3 n=6 summen
   =unter (0.5*x)^2
```

Möchte man anstelle der Stammfunktion oder Obersumme bzw. Untersumme das zugehörige Integral einer Funktion zeichnen, so gibt es den Befehl `integral`. Dieser zeichnet die eingeschlossene Fläche zwischen der Funktion und der x -Achse in dem angegebenen Intervall. Die Syntax ist:

integral

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
[y=<ymin ∈ ℝ>,<ymax ∈ ℝ>]
<Funktionsterm f(x)>
```

Ist y nicht angegeben, so werden die Grenzen automatisch berechnet.

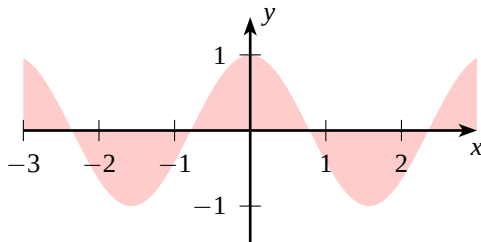


Abb. 6.11.: Der Befehl `integral`

Listing 6.11: Der Befehl `integral`

```
1 achsen
2 integral layer=5 farbe=red!20,solid,red
   !20 linie=1,solid x=-3,3 y=-1.5,1.5
   cos(2*x)
```

6.6.3. Einschluß zwischen zwei Funktionen

Beschäftigt man sich mit dem Integral, so soll natürlich auch die Fläche zwischen zwei Funktionen gezeichnet werden können. Dies ist mit dem Befehl `einschluss` möglich. Die Syntax ist die folgende:

einschluss

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
[y=<ymin ∈ ℝ>,<ymax ∈ ℝ>]
schnitt=<links|rechts|beide|keine>
<Funktionsterm f1(x)>,<Funktionsterm f2(x)>
```

Ist der `schnitt` auf `beide` gesetzt, so werden automatisch die zwei Schnittpunkte einer Funktion berechnet. Analog wird bei `links` der linke Schnittpunkt berechnet und der rechte Zeichenbereich ist der $x_{\max}$ Wert.

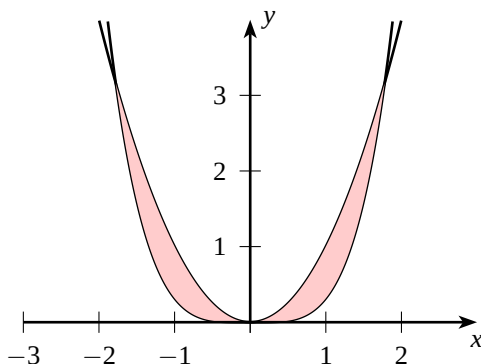


Abb. 6.12.: Der Befehl `einschluss`

Listing 6.12: Der Befehl `einschluss`

```
1 achsen
2 einschluss layer=50 farbe=black,solid,
   red!20 linie=1,solid parameter="
   plotstyle=curve" x=-3,3 y=0,4 schnitt=
   beide (0.75*x)^4,(x)^2
3 graph x=-3,3 y=0,4 (0.75*x)^4
4 graph x=-3,3 y=0,4 (x)^2
```

6.7. Rotationskörper um die x -Achse

Um einen **Rotationskörper**, der durch Rotation der Fläche, die durch den Graphen der Funktion f im Intervall $[x_{\min}, x_{\max}]$, die x -Achse und die beiden Geraden $x = x_{\min}$ und $x = x_{\max}$ begrenzt wird, um die x -Achse zu zeichnen, gibt es den Befehl `graphrot`

graphrot

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ> [y=<ymin ∈ ℝ>,<ymax ∈ ℝ>]
r=<Einteilung r ∈ ℕ>
<Funktionsterm f(x)>
```

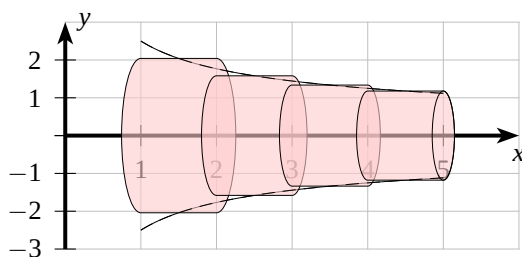


Abb. 6.13.: Der Befehl `graphrot`

Listing 6.13: Der Befehl `graphrot`

```
1 groesse x=6 y=3
2 gitter
3 koordinatensystem x=0,6 y=-3,3
  einteilung=1,1 skala=normal text=
  "$x$","$y$"
4 graphrot farbe=black,shape,red!20 x=1,5 y
  =-3,3 r=4 2.5/(x^0.5)
```

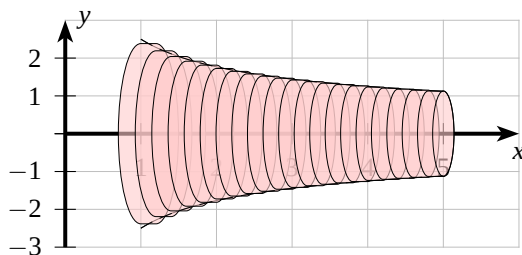


Abb. 6.14.: Der Befehl `graphrot` mit feineren Einteilung

Listing 6.14: Der Befehl `graphrot` mit feineren Einteilung

```
1 groesse x=6 y=3
2 gitter
3 koordinatensystem x=0,6 y=-3,3
  einteilung=1,1 skala=normal text=
  "$x$","$y$"
4 graphrot farbe=black,shape,red!20 x=1,5 y
  =-3,3 r=20 2.5/(x^0.5)
```

6.8. Parametrisierte Kurven

Unter einer **Parameterdarstellung** (auch **Parametrisierung** oder **Parametrierung**) einer Kurve versteht man eine Darstellung, bei der die Punkte der Kurve über einen einzigen Parameter abgelaufen werden können. Ein Beispiel hierfür ist die Darstellung einer Schleife in der Form

$$(\sin(t), \sin(t) \cdot \cos(t)) \quad \text{für } 0 \leq t \leq 2\pi \quad (6.5)$$

Die Syntax des Befehls ist die folgende:

parametricplot

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
t=<tmin ∈ ℝ>,<tmax ∈ ℝ>
<Funktionsterm x(t)>,<Funktionsterm y(t)>
```

Wobei $[t_{min}; t_{max}]$ das jeweilige Definitionsintervall beschreibt.

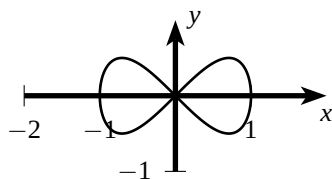


Abb. 6.15.: Der Befehl `parametricplot`

Listing 6.15: Der Befehl `parametricplot`

```
1 koordinatensystem x=-2,2 y=-1,1 einteilung
  =1,1 linie=2,solid,black skala=normal text="
  $x$","$y$"
```

```
2 parametricplot layer=50 farbe=black linie=1,
  solid t=0,360 sin(t),sin(t)*cos(t)
```

6.9. Zeichnen von Funktionsfolgen

Eine **Funktionsfolge** ist eine Folge f_n , $n \in \mathbb{N}$ von Funktionen $f_i : \mathbb{R} \rightarrow \mathbb{R}$. Diese kann mit dem Befehl `folge` gezeichnet werden.

folge

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
[r=<dx ∈ ℝ>,<r ∈ ℝ>]
<Funktionsterm f(x)>
```

dx gibt die Abstände der Punktfolgen an. Standardmäßig ist dies aus 1 gesetzt. r definiert den Radius der Punkte und ist standardmäßig auf 5 pt gesetzt.

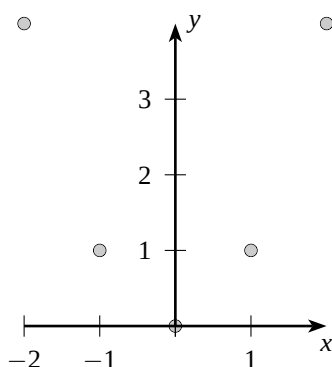


Abb. 6.16.: Der Befehl `folge`

Listing 6.16: Der Befehl `folge`

```
1 achsen
```

```
2 folge layer=50 farbe=black linie=1,
  solid parameter="" x=-2,2 r=1,5 x^2
```

6.10. Plotten von Daten

Um Daten zu visualisieren gibt es in meinem Skript mehrere Befehle. Um dies zu veranschaulichen nutze ich im folgenden die Tabelle 6.1.

t in s	1	2	3	4	5	6	7	8	9	10	15	20
I in mA	130	100	80	70	55	45	40	30	25	20	7	3

Tab. 6.1.: Entladestrom eines Kondensators

Es gibt die drei Befehle `listplot`, `fileplot` und `dataplot`. Diese realisieren die gleichnamigen Funktionen aus Pstricks¹. Die Syntax ist dabei bei allen Funktionen die folgende:

¹Siehe (Voss, 2008).

```
[<fileplot | listplot | dataplot>]
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
  [parameter="<Pstricksbefehle>"]
  [lese <Dateiname>] oder <x ∈ ℝ>,<y ∈ ℝ> beliebig oft
ende
```

Es gibt mit dem Befehl `lese` die Möglichkeit eine Komma oder Semikol getrennte Datei einzulesen, oder direkt entsprechende Werte einzulesen.

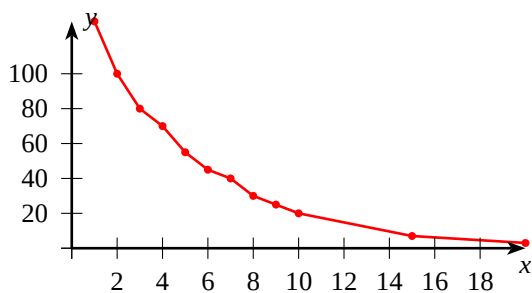


Abb. 6.17.: Der Befehl `listplot`

Listing 6.17: Der Befehl `listplot`

```
1 groesse x=6 y=3
2 achsen
3 listplot farbe=red linie=1,solid
  parameter="showpoints=true"
4 1,130
5 2,100
6 3,80
7 4,70
8 5,55
9 6,45
10 7,40
11 8,30
12 9,25
13 10,20
14 15,7
15 20,3
16 ende
```

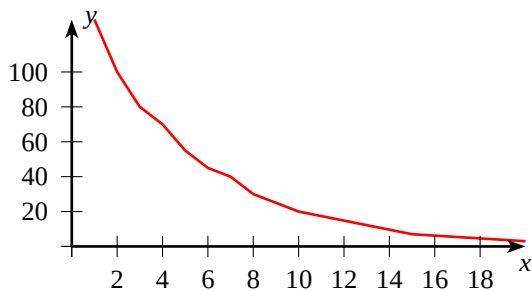


Abb. 6.18.: Der Befehl `fileplot`

Listing 6.18: Der Befehl `fileplot`

```
1 groesse x=6 y=3
2 achsen
3 fileplot farbe=red linie=1,solid
  parameter=""
4 lese Abbildungen/kondensator-entladung.
  dat
5 ende
```

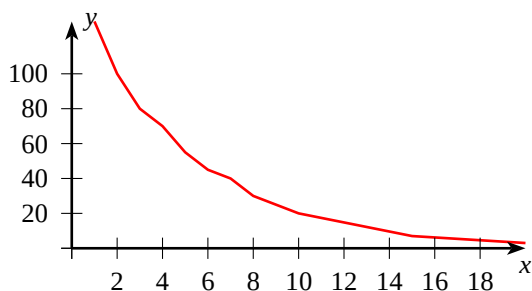


Abb. 6.19.: Der Befehl `dataplot`

Listing 6.19: Der Befehl `dataplot`

```
1 groesse x=6 y=3
2 achsen
3 dataplot farbe=red linie=1,solid
  parameter=""
4 lese Abbildungen/kondensator-entladung.
  dat
5 ende
```

Mithilfe des Befehls `barplot` kann man die Daten auch als Säulendiagramm darstellen. Dabei ist standardmäßig eine Breite von 1 eingestellt.

```
barplot
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
  [linie=<Linienstärke in pt ( $l \in \mathbb{R}_{>0}$ )>,<Linienart>]
  [parameter="<Pstricksbefehle>"]
  [r=<Säulenbreite  $b \in \mathbb{R}_{>0}$ >]
  [lese <Dateiname>] oder < $x \in \mathbb{R}$ >,< $y \in \mathbb{R}$ > beliebig oft
ende
```

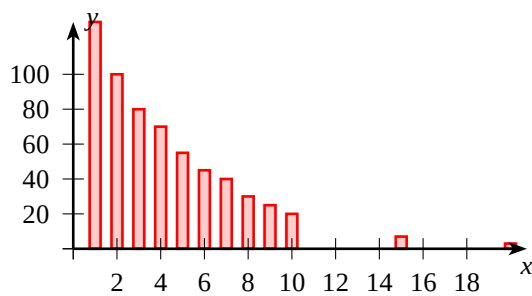


Abb. 6.20.: Der Befehl `barplot`

Listing 6.20: Der Befehl `barplot`

```
1 groesse x=6 y=3
2 achsen
3 barplot farbe=red,solid,red!20 linie=1,
    solid r=0.5
4 lese Abbildungen/kondensator-entladung.
    dat
5 ende
```


7. Diagramme

Da im Unterricht doch häufiger verschiedene Diagrammtypen auftauchen, habe ich vor etwa einem Jahr mehrere Funktionalitäten in mein Skript eingebaut. Hierzu gehören u.a. die Säulen- und Balkendiagramme, sowie Piktogramme und weitere Diagrammtypen.

7.1. Säulen- und Balkendiagramme

Um Zahlen auf einen Blick zu zeigen und zu erfassen, werden verschiedene Diagramme verwendet. Hierzu werden häufig Säulen- oder Balkendiagramme genutzt, die sich nur darin unterscheiden, dass die Datenreihe 90° gedreht ist. Im Folgenden wird zunächst nur das **Säulendiagramm** beschrieben. Analoges gilt für das **Balkendiagramm**. Über die jeweilige Merkmalsausprägung wird eine Säule gezeichnet, deren Höhe der absoluten oder relativen Häufigkeit entspricht.¹

7.1.1. Säulendiagramme

Zunächst einmal die beiden Möglichkeiten für das `saeulendiagramm`:

saeulendiagramm

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
[r=<Standardbreite der Säulen b ∈ ℝ>0>,<Standardabstand zur nächsten Säule a ∈ ℝ>0>]
ausrichtung=<Textwinkel in Grad>,<Textabstand in y-Richtung>
    <Absoluter Wert n ∈ ℝ> (beliebig oft)
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
[r=<Standardbreite der Säule b ∈ ℝ>0>,<Standardabstand zur nächsten Säule a ∈ ℝ>0>]
text="<Text>"
```

ende

Dabei steht `p` für die Startposition der ersten Säule, die zweite Säule wird im Abstand `abstand` nach rechts gezeichnet. `breite` gibt die Breite der Säule an. Ist bei dem absoluten Wert keine Farbe, etc. gesetzt, so wird die Säule mit den vordefinierten Parametern gezeichnet.

Hierbei fallen zwei Parameter auf. Ist `achsen` gesetzt, so wird automatisch ein Koordinatensystem mit gezeichnet. Ist `relativ` gesetzt, so werden nicht die absoluten Häufigkeiten dargestellt, sondern die relativen Häufigkeiten. Der Befehl `mitte` setzt die Achsenbeschriftung jeweils in die Mitte der einzelnen Achsen. Der Befehl `achsenmarkierung` setzt die Achsenbeschriftung für die einzelnen Achsen.

Im einzelnen lautet er wie folgt:

achsenmarkierung

```
[x="<x – Achsenbeschriftung>"]
[y="<y – Achsenbeschriftung>"]
```

Alternativ kann man die Position einer einzelnen Säule angeben:

¹Siehe (Div., 2010)

saeulendiagramm

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
[r=<Standardbreite der Säulen b ∈ ℝ>0>]
ausrichtung=<Textwinkel in Grad>,<Textabstand in y-Richtung>
<x ∈ ℝ><Absoluter Wert n ∈ ℝ> (beliebig oft)
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
[r=<Standardbreite der Säulen b ∈ ℝ>0>]
text="<Text>"

```

ende

Hier einmal die Beschreibung für das Zeichnen der Säule:

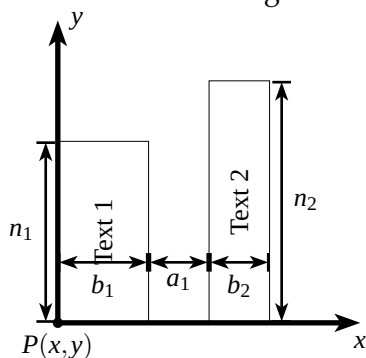


Abb. 7.1.: Die Definition des Säulendiagramms

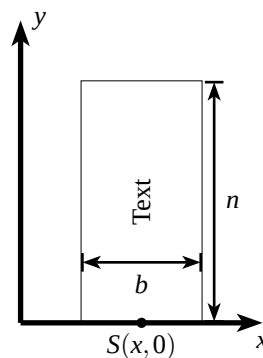


Abb. 7.2.: Die zweite Definition des Säulendiagramms

Mit `farbverlauf` werden die Säulen in Abstufungen mit der gesetzten Füllfarbe gezeichnet.

Als Beispiel zeigt die folgende Tabelle die Verkehrsmittel mit denen einzelne Kinder einer Klasse zur Schule kommen und anschließend mehrere Beispiele für Säulendiagramme.

Verkehrsmittel	zu Fuß	Fahrrad	U-Bahn	Bus
Absolute Häufigkeiten	5	4	3	6
Relative Häufigkeiten	5/18	4/18	3/18	6/18

Tab. 7.1.: Schulweg

Beispiele

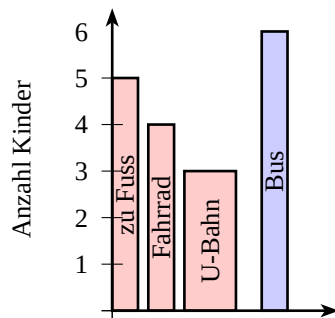


Abb. 7.3.: Säulendiagramm

Listing 7.1: Säulendiagramm

```

1 groesse x=3 y=4
2 achsenmarkierung y="Anzahl Kinder"
3 saeulendiagramm farbe=black,solid,red!20 linie=1,solid
   parameter="" p=0,0 r=0.5,0.2 ausrichtung=90,
   mitte
4 5 text="zu Fuss"
5 4 text="Fahrrad"
6 3 r=1,0.5 text="U-Bahn"
7 6 farbe=black,solid,blue!20 text="Bus"
8 ende
9 achsen
10 mitte

```

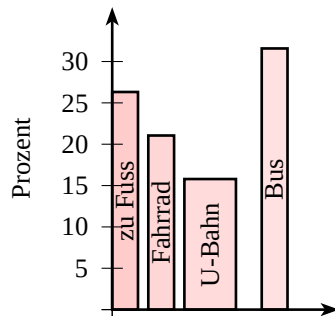


Abb. 7.4.: Säulendiagramm Nr. 2

Listing 7.2: Säulendiagramm Nr. 2

```

1 relativ
2 farbverlauf
3 groesse x=3 y=4
4 achsenmarkierung y="Anzahl Kinder"
5 saeulendiagramm farbe=black,solid,red!20 linie=1,solid
   parameter="" p=0,0 r=0.5,0.2 ausrichtung=90,
   mitte
6 5 text="zu Fuss"
7 4 text="Fahrrad"
8 3 r=1,0.5 text="U-Bahn"
9 6 farbe=black,solid,blue!20 text="Bus"
10 ende
11 achsen
12 mitte

```

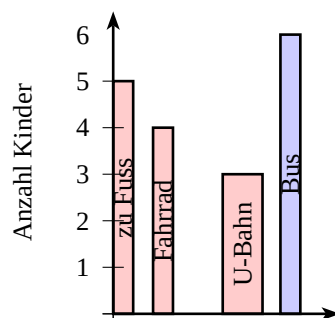


Abb. 7.5.: Säulendiagramm Nr. 3

Listing 7.3: Säulendiagramm Nr. 3

```

1 groesse x=3 y=4
2 achsenmarkierung y="Anzahl Kinder"
3 saeulendiagramm farbe=black,solid,red!20 linie=1,solid
   parameter="" r=0.5 ausrichtung=90,mitte
4 1,5 text="zu Fuss"
5 2,4 text="Fahrrad"
6 4,3 r=1 text="U-Bahn"
7 5.2,6 farbe=black,solid,blue!20 text="Bus"
8 ende
9 achsen
10 mitte

```

7.1.2. Balkendiagramme

Analog zum Säulendiagramm gibt es das **Balkendiagramm** Hier die erste Syntax des balkendiagramm:

balkendiagramm

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<Standardhöhe der Säulen b ∈ ℝ>0>0>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
r=<Standardhöhe der Säulen b ∈ ℝ>0>0

```

ende

Dabei steht p für die Startposition des ersten Balkens, die zweite Säule wird im Abstand `abstand` nach oben gezeichnet. `hoehe` gibt die Höhe des Balkens an.

Die zweite Syntax lautet

balkendiagramm

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
r=<Standardhöhe der Säulen b ∈ ℝ>0>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
r=<Standardhöhe der Säulen b ∈ ℝ>0

```

ende

Beispiele

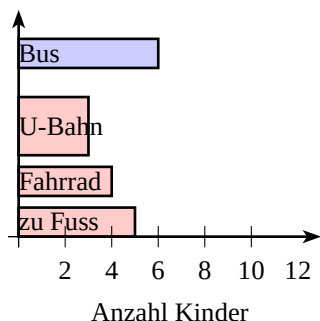


Abb. 7.6.: Balkendiagramm

Listing 7.4: Balkendiagramm

```
1 groesse x=4 y=3
2 achsenmarkierung x="Anzahl Kinder"
3 balkendiagramm farbe=black,solid,red!20 linie=1,solid
   parameter="" p=0,0 r=0.5,0.2 ausrichtung=0,auf
4 5 text="zu Fuss"
5 4 text="Fahrrad"
6 3 r=1,0.5 text="U-Bahn"
7 6 farbe=black,solid,blue!20 text="Bus"
8 ende
9 achsen
10 mitte
```

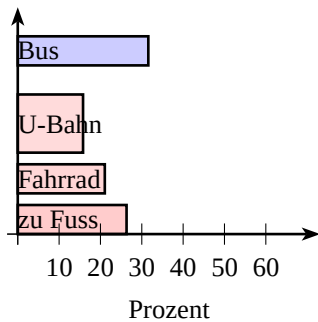


Abb. 7.7.: Balkendiagramm Nr. 2

Listing 7.5: Balkendiagramm Nr. 2

```

1  relativ
2  farbverlauf
3  groesse x=4 y=3
4  achsenmarkierung x="Anzahl Kinder"
5  balkendiagramm farbe=black,solid,red!20 linie=1,solid
   parameter="" p=0,0 r=0,5,0,2 ausrichtung=0,auf
6  5 text="zu Fuss"
7  4 text="Fahrrad"
8  3 r=1,0,5 text="U-Bahn"
9  6 farbe=black,solid,blue!20 text="Bus"
10 ende
11 achsen
12 mitte

```

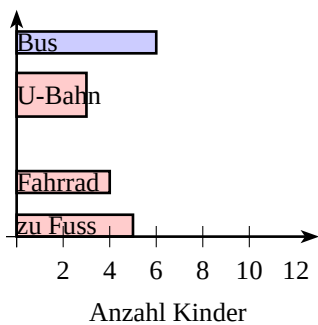


Abb. 7.8.: Balkendiagramm Nr. 2

Listing 7.6: Balkendiagramm Nr. 2

```

1  groesse x=4 y=3
2  achsenmarkierung x="Anzahl Kinder"
3  balkendiagramm farbe=black,solid,red!20 linie=1,solid
   parameter="" r=0,5 ausrichtung=0,auf
4  5,1 text="zu Fuss"
5  4,2 text="Fahrrad"
6  3,4 r=1 text="U-Bahn"
7  6,5,2 farbe=black,solid,blue!20 text="Bus"
8  ende
9  achsen
10 mitte

```

7.1.3. Gestapelte Säulen- und Balkendiagramme

In einem gestapelten Diagramm werden Werte mehrerer Datenreihen jeweils aufeinander oder nebeneinander gestapelt dargestellt. Der Gesamtbalken ist jeweils der Gesamtwert. Dabei sind die Änderungen zum Säulen- und Balkendiagramm minimal.

Anstelle von `saeulendiagramm` wird `saeulendiagrammgestapelt` genutzt. Analog ist dies beim `balkendiagramm` der Befehl `balkendiagrammgestapelt`.

Beispiele dazu sind in den Abbildungen 7.9 und 7.10 zu sehen.

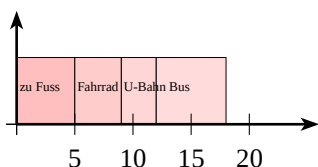


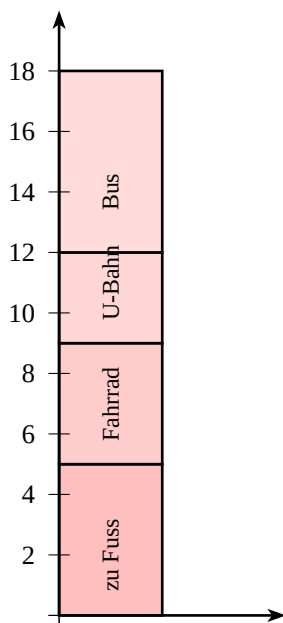
Abb. 7.9.: Gestapeltes Balkendiagramm

Listing 7.7: Gestapeltes Balkendiagramm

```

1  beschriftunggroesse "tiny"
2  farbverlauf
3  groesse x=4 y=1,5
4  balkendiagrammgestapelt farbe=black,solid,red!22
   linie=0,2,solid p=0,0 r=1,5 ausrichtung=0,2
5  5 text="zu Fuss"
6  4 text="Fahrrad"
7  3 text="U-Bahn"
8  6 text="Bus"
9  ende
10 achsen

```



Listing 7.8: Gestapeltes Säulendiagramm

```

1  beschriftungsgroesse "small"
2  groesse x=3 y=8
3  farbverlauf
4  saeulendiagrammgestapelt farbe=black,solid,red!20
   linie=1,solid p=0,0 r=1,5 ausrichtung=90,2
5  5 text="zu Fuss"
6  4 text="Fahrrad"
7  3 text="U-Bahn"
8  6 text="Bus"
9  ende
10 achsen

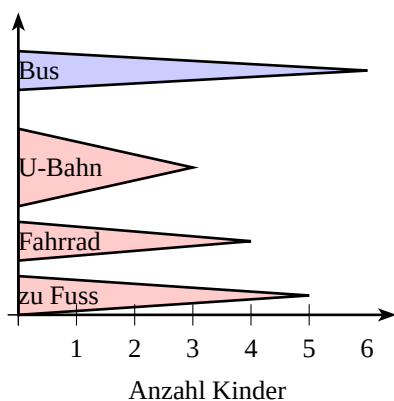
```

Abb. 7.10.: Gestapeltes Säulendiagramm

Die Quellcodeänderungen bei dem gestapelten Säulendiagramm sind nur minimal. Es ist effektiv nur eine Zeile geändert. `groesse` dient dazu, dass die Ausmaße des Bilder gesetzt werden.

7.1.4. Weitere Säulen- und Balkendiagrammtypen

Zu guter Letzt gibt es noch die Säulen- und Balkendiagramme in Form von Dreiecken. Dazu wird fast der gleiche Befehl genutzt wie in 7.1 und 7.3, nur dass statt `saeulendiagramm` in Zeile 2 der Befehl `saeulendiagrammdreieck` steht. `r` steht dabei für die Breite eines gleichschenkligen Dreiecks.



Listing 7.9: Balkendiagramm in Dreiecksform Nr. 1

```

1  groesse x=5 y=4
2  achsenmarkierung x="Anzahl Kinder"
3  balkendiagrammdreieck farbe=black,solid,red!20
   linie=1,solid parameter="" p=0,0 r=0.5,0.2
   ausrichtung=0,auf
4  5 text="zu Fuss"
5  4 text="Fahrrad"
6  3 r=1,0.5 text="U-Bahn"
7  6 farbe=black,solid,blue!20 text="Bus"
8  ende
9  achsen
10 mitte

```

Abb. 7.11.: Balkendiagramm in Dreiecksform Nr. 1

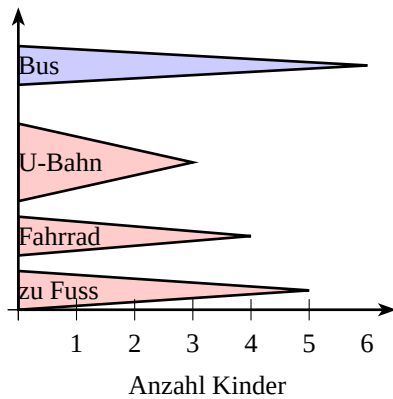


Abb. 7.12.: Balkendiagramm in Dreiecksform Nr. 2

Listing 7.10: Balkendiagramm in Dreiecksform Nr. 2

```

1 relativ
2 farbverlauf
3 groesse x=5 y=4
4 achsenmarkierung x="Anzahl Kinder"
5 balkendiagrammdreieck farbe=black,solid,red!20
   linie=1,solid parameter="" p=0,0 r=0.5,0.2
   ausrichtung=0,auf
6 5 text="zu Fuss"
7 4 text="Fahrrad"
8 3 r=1,0.5 text="U-Bahn"
9 6 farbe=black,solid,blue!20 text="Bus"
10 ende
11 achsen
12 mitte

```

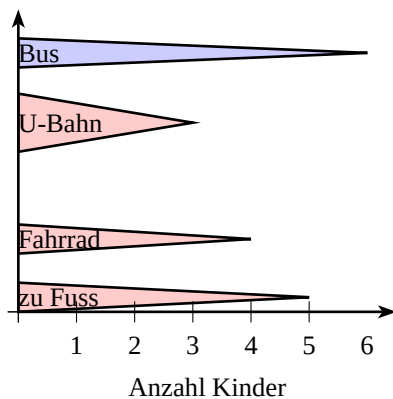


Abb. 7.13.: Balkendiagramm in Dreiecksform Nr. 3

Listing 7.11: Balkendiagramm in Dreiecksform Nr. 3

```

1 groesse x=5 y=4
2 achsenmarkierung x="Anzahl Kinder"
3 balkendiagrammdreieck farbe=black,solid,red!20
   linie=1,solid parameter="" r=0.5 ausrichtung
   =0,auf
4 5,1 text="zu Fuss"
5 4,2 text="Fahrrad"
6 3,4 r=1 text="U-Bahn"
7 6,5.2 farbe=black,solid,blue!20 text="Bus"
8 ende
9 achsen
10 mitte

```

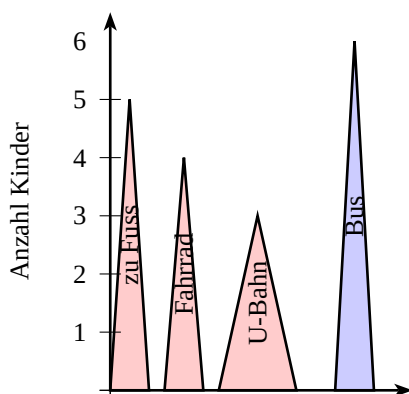


Abb. 7.14.: Säulendiagramm in Dreiecksform Nr. 1

Listing 7.12: Säulendiagramm in Dreiecksform Nr. 1

```

1 groesse x=4 y=5
2 achsenmarkierung y="Anzahl Kinder"
3 saeulendiagrammdreieck farbe=black,solid,red!20
   linie=1,solid parameter="" p=0,0 r=0.5,0.2
   ausrichtung=90,mitte
4 5 text="zu Fuss"
5 4 text="Fahrrad"
6 3 r=1,0.5 text="U-Bahn"
7 6 farbe=black,solid,blue!20 text="Bus"
8 ende
9 achsen
10 mitte

```

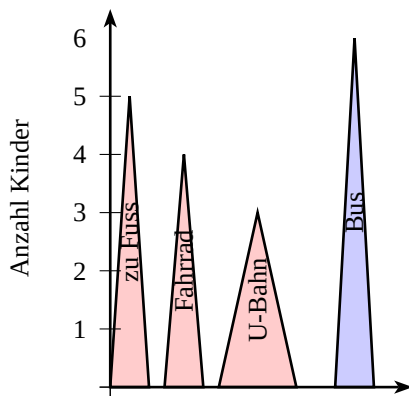


Abb. 7.15.: Säulendiagramm in Dreiecksform Nr. 2

Listing 7.13: Säulendiagramm in Dreiecksform Nr. 2

```

1 relativ
2 farbverlauf
3 groesse x=4 y=5
4 achsenmarkierung y="Anzahl Kinder"
5 saeulendiagrammdreieck farbe=black,solid,red!20
   linie=1,solid parameter="" p=0,0 r=0.5,0.2
   ausrichtung=90,mitte
6 5 text="zu Fuss"
7 4 text="Fahrrad"
8 3 r=1,0.5 text="U-Bahn"
9 6 farbe=black,solid,blue!20 text="Bus"
10 ende
11 achsen
12 mitte

```

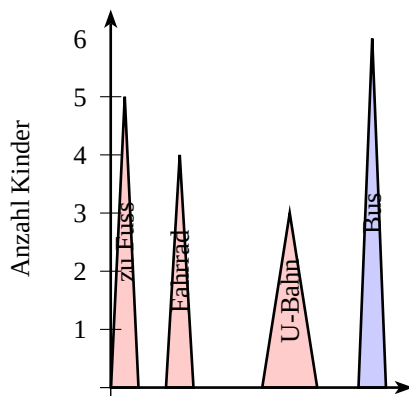


Abb. 7.16.: Säulendiagramm in Dreiecksform Nr. 3

Listing 7.14: Säulendiagramm in Dreiecksform Nr. 3

```

1 groesse x=4 y=5
2 achsenmarkierung y="Anzahl Kinder"
3 saeulendiagrammdreieck farbe=black,solid,red!20
   linie=1,solid parameter="" r=0.5 ausrichtung
   =90,mitte
4 1,5 text="zu Fuss"
5 2,4 text="Fahrrad"
6 4,3 r=1 text="U-Bahn"
7 5,5,6 farbe=black,solid,blue!20 text="Bus"
8 ende
9 achsen
10 mitte

```

7.2. Kreis- und Ringdiagramme

Die bisherigen Diagrammtypen waren alles Achsendiagramme, nun sollen noch weitere Diagrammtypen behandelt werden.

7.2.1. Torten- oder Kreisdiagramm

Zunächst soll ein **Tortendiagramm** oder auch **Kreisdiagramm** behandelt werden.² Dabei steht dann ein Kresisektor für einen Teilwert und der gesamte Kreis für die Gesamtsumme aller Teilwerte. Der Befehl `tortendiagramm` wird wie gewohnt aufgerufen.

tortendiagramm

```

[layer=<n ∈ [0; 100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<Radius r ∈ ℝ>0>0>0

```

²Im Folgenden wird sich weiterhin auf die Tabelle 7.1 bezogen.

```
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt ( $l \in \mathbb{R}_{>0}$ )>,<Linienart>]
[parameter="<Pstricksbefehle>"]
[r=<Radius  $r \in \mathbb{R}_{>0}$ >,<Abstand Mittelpunkt  $a \in \mathbb{R}_{>0}$ >]
text="<Text>"
```

ende

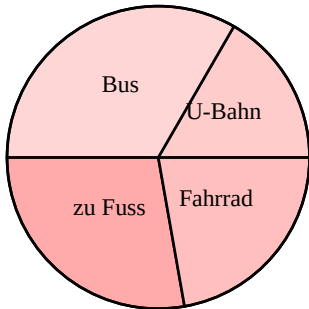


Abb. 7.17.: Ein Kreisdiagramm

p steht dabei für den Mittelpunkt des Kreises. $r=2,0,180$ steht dabei für den Radius des Kreises mit 2 und 0 steht dafür, wie weit der Abstand vom Mittelpunkt ist. 180 ist der Winkel, auf dem der erste Kreissektor beginnt. Die weiteren Sektoren werden immer gegen den Uhrzeigersinn angeordnet.

Listing 7.15: Ein Kreisdiagramm

```
1 farbverlauf
2 beschriftungsgroesse "small"
3 tortendiagramm farbe=black,solid,red!20 linie=1,
  solid p=0,0 r=2,0,180
4 5 text="zu Fuss"
5 4 text="Fahrrad"
6 3 text="U-Bahn"
7 6 text="Bus"
8 ende
```

Beispiele

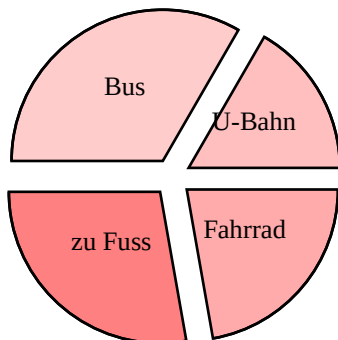


Abb. 7.18.: Ein Kreisdiagramm mit $r=2,0.25,0$

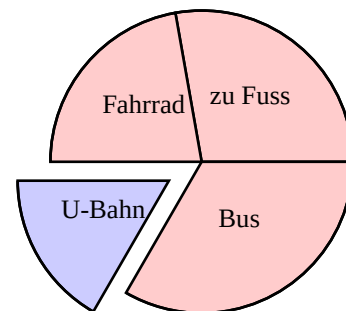


Abb. 7.19.: Veränderte Zeile 3
 farbe=blue,solid,blue!20
 r=2,0.5 text="U-Bahn"

7.2.2. Ringdiagramm

Ein weiterer Diagrammtyp ist das **Ringdiagramm**.

ringdiagramm

```
[layer=< $n \in [0;100]$ >]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt ( $l \in \mathbb{R}_{>0}$ )>,<Linienart>]
[parameter="<Pstricksbefehle>"]
p=< $x \in \mathbb{R}$ >,< $y \in \mathbb{R}$ >
r=<Außenradius  $r \in \mathbb{R}_0$ >,<Ringbreite  $b \in \mathbb{R}_0$ >,<Startwinkel in Grad>
<Absoluter Anteil  $a \in \mathbb{R}_{>0}$ >
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt ( $l \in \mathbb{R}_{>0}$ )>,<Linienart>]
[parameter="<Pstricksbefehle>"]
```

[r=<Radius $r \in \mathbb{R}_{>0}a \in \mathbb{R}_{>0}text=<Text>$

ende

Dies wird mit dem Befehl `ringdiagramm` aufgerufen, der ansonsten analog zum `Kreisdiagramm`-befehl ist. Hierbei ist allerdings im Befehl `r` die erste Zahl der Außenradius des Kreises und die zweite Zahl die Breite des Ringes dar.

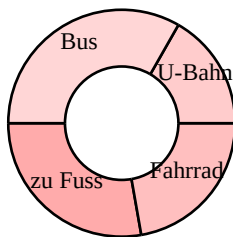


Abb. 7.20.: Ein Ringdiagramm

Listing 7.16: Ein Ringdiagramm

```
1 farbverlauf
2 beschriftungsgroesse "small"
3 ringdiagramm farbe=black,solid,red!20 linie=1,
  solid p=0,0 r=1.5,0.75,180
4 5 text="zu Fuss"
5 4 text="Fahrrad"
6 3 text="U-Bahn"
7 6 text="Bus"
8 ende
```

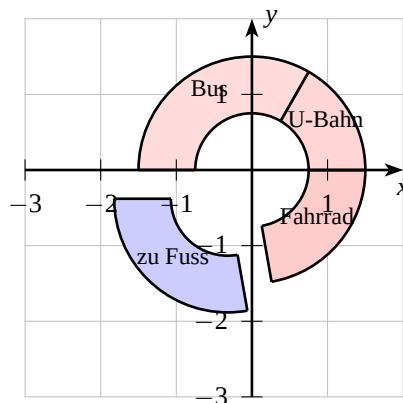


Abb. 7.21.: Veränderte Zeile 5 `farbe=blue,solid,blue!20 r=1.5,0.75,0.5 text="zu Fuss"`

In der Abbildung 7.21 sieht man eine kleine Änderung im Befehl `r`. Auch hier ist die erste Zahl der Außenradius und die zweite Zahl die Breite des Ringausschnittes. Die dritte Zahl steht für den Abstand vom Mittelpunkt `p`.

7.3. Piktogramme und Strichlisten

Als zwei weitere Diagrammtypen habe ich noch ein **Piktogramm** und eine **Strichliste** realisiert.

7.3.1. Piktogramm

In einem **Piktogramm** werden absolute Häufigkeiten und Größen visualisiert. Dabei entspricht jedes Symbol einer bestimmten Anzahl bzw. Größe. Mit dem Befehl `bilddiagramm` wird ein Piktogramm erstellt.

piktogramm

p=< $x \in \mathbb{R}y \in \mathbb{R}$

r=<Breite des Bildes in pt>,<Winkel des Bildes in Grad>,<Abstand in x-Richtung>,<Abstand in y-Richtung zum nächsten Bild>

ausrichtung=<Textwinkel in Grad>,<Ausrichtung in y-Richtung>

```

<Anzahl der Bilder  $n \in \mathbb{N}$ > beliebig oft
  [r=<Breite des Bildes in pt>,<Winkel des Bildes in Grad>,<Abstand in x-Richtung>,<Abstand in
    y-Richtung zum nächsten Bild>]
  text="<Text>"
  [ausrichtung=<Textwinkel in Grad>,<Ausrichtung in y-Richtung>]
  "<Dateiname>"
ende

```

Dabei steht p für den Startpunkt des ersten Bildes. In r steht zunächst die Breite eines Bildes, der Offset und der Abstand der Bilder in x und y Richtung. In der `ausrichtung` kann der Winkel des Textes und der Abstand in y -Richtung definiert werden.

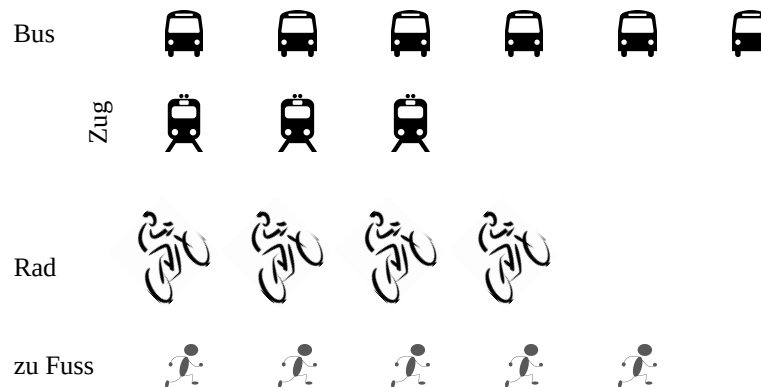


Abb. 7.22.: Ein Piktogramm

Listing 7.17: Ein Piktogramm

```

1 piktogramm p=0.2,0.5 r=0.5,0,1,0.5 ausrichtung=0,-2
2   5 text="zu Fuss" "Abbildungen/fuss.eps"
3   4 r=1,45,0.5,1 text="Rad" "Abbildungen/rad.eps"
4   3 text="Zug" ausrichtung=90,-1 "Abbildungen/zug.eps"
5   6 text="Bus" "Abbildungen/bus.eps"
6 ende

```

7.3.2. Eine Strichliste

Auch eine **Strichliste** habe ich in meinem Skript realisiert. Mit dem Befehl `strichliste` fängt dieser an.

```

strichliste
  [layer=< $n \in [0; 100]$ >]
  p=< $x \in \mathbb{R}$ >,< $y \in \mathbb{R}$ >
  r=<Höhe des Striches  $h \in \mathbb{R}_{>0}$ >,<Abstand in x-Richtung  $a \in \mathbb{R}_{>0}$ ><Anzahl Striche  $n \in \mathbb{N}$ >

```

In r wird die Höhe und der horizontale Abstand definiert. Die Anzahl gibt an, wie viele Striche gezeichnet werden sollen.



Abb. 7.23.: Eine Strichliste

Listing 7.18: Eine Strichliste

```

1 strichliste linie=2,solid,black p=0,0 r=1,0.2 17

```

7.4. Eine Legende

Wenn man ein Diagramm zeichnen möchte, darf natürlich keine **Legende** fehlen. Dies wird mit dem Befehl `legende` aufgerufen. Die genaue Syntax lautet:

legende

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>
r=<Breite b ∈ ℝ>0>0y ∈ ℝ>0>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
```

ende

`p` steht für die Startposition des ersten Eintrages. Die Breite und Höhe gibt die Ausmaße des Rechteckes an. Der letzte Eintrag steht für den Abstand zwischen den Rechtecken in y -Richtung. Die weiteren Rechtecke werden von oben nach unten gezeichnet.



Abb. 7.24.: Eine Legende

Listing 7.19: Eine Legende

```
1 legende farbe=red,solid,red!20 linie=1,solid p=0,0 r=0.5,0.5,0.2
2   text="zu Fuss" farbe=black,solid,green!20 linie=2,solid
3   text="Rad" farbe=black,solid,red!20
4   text="Zug" farbe=black,solid,blue!20
5   text="Bus" farbe=black,solid,orange!20
6 ende
```



Abb. 7.25.: Eine zweite Legende

Listing 7.20: Eine zweite Legende

```
1 farbverlauf
2 legende farbe=black,solid,red!20 linie=1,solid p=0,0 r=0.5,0.5,0.2
3   text="zu Fuss"
4   text="Rad"
5   text="Zug"
6   text="Bus"
7 ende
```

7.5. Die Binomialverteilung

In meinem Skript habe ich ebenfalls die **Binomialverteilung** realisiert. Eine Zufallsgröße heißt **binomialverteilt** mit den Parametern n und p , wenn für alle k ($k = 0, \dots, n$) gilt:

$$P(X = k) = \binom{n}{k} p^k \cdot (1 - p)^{n-k} \quad (7.1)$$

7.5.1. Die Binomialverteilung als Säulendiagramm

Die Syntax des Skriptes sieht dabei wie folgt aus:

```
binomial
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter=<Pstricksbefehle>]
np=<n ∈ ℕ>,<p ∈ [0;1]>,<ja|nein>
b=<Säulenbreite b ∈ ℝ>0Skala ∈ ℝ>0>
```

Dabei steht **ja** oder **nein** dafür, ob die Binomialverteilung aufsummiert werden soll oder nicht. Setzt man **ja**, so wird die **kumulierte Binomialverteilung** dargestellt. Desweiteren kann die Säulenbreite und der Abstand der Achsenmarkierung festgelegt werden.

Setzt man auch hier erneut **achsen** mit als Befehl, so werden die Achsen mit gezeichnet.

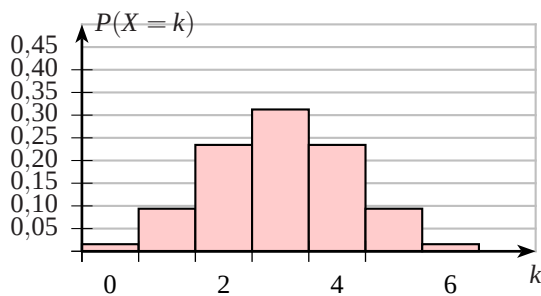


Abb. 7.26.: Die Binomialverteilung

Listing 7.21: Die Binomialverteilung

```
1 achsenmarkierung x="$k$" y="$P(X=k)$"
2 groesse x=6 y=3
3 achsen
4 gitter
5 binomial farbe=black,solid,red!20 linie=0.8,
   solid np=6,0.5,nein b=1,2
```

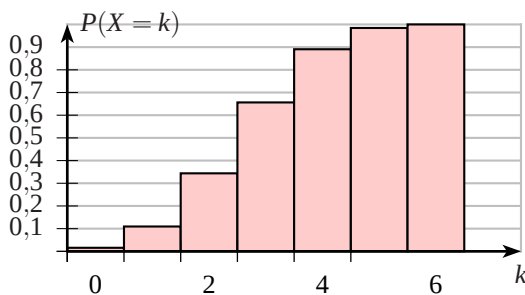


Abb. 7.27.: Die kumulierte Binomialverteilung

Möchte man nur einzelne Teile der Verteilung angezeigt haben, so gibt es den Befehl **binomialteil**, der wie folgt aufgerufen wird:

```
binomialteil
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter=<Pstricksbefehle>]
np=<n ∈ ℕ>,<p ∈ [0;1]>,<Startwert x1 ∈ ℕ>,<Endwert x2 ∈ ℕ>,<ja|nein>
```

$b = \langle \text{Säulenbreite } b \in \mathbb{R}_{>0}, \langle x_{\text{Skala}} \in \mathbb{R}_{>0} \rangle \rangle$

Hierbei wird k von startwert bis endwert dargestellt.

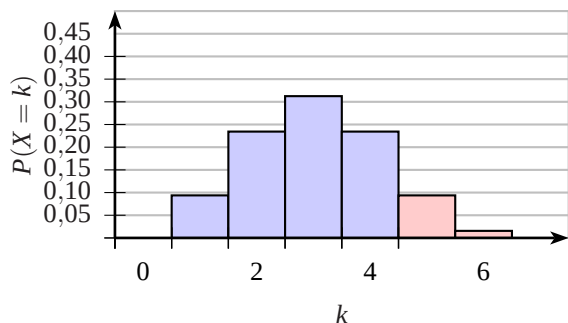


Abb. 7.28.: Die Binomialverteilung teilweise dargestellt

Listing 7.23: Die Binomialverteilung teilweise dargestellt

```

1 achsenmarkierung x="$k$" y="$P(X=k)$"
2 groesse x=6 y=3
3 achsen
4 gitter
5 mitte
6 binomialteil farbe=black,solid,blue!20 linie
   =0.8,solid np=6,0.5,1,4,nein b=1,2
7 binomialteil farbe=black,solid,red!20 linie
   =0.8,solid np=6,0.5,5,6,nein b=1,2

```

7.5.2. Die Binomialverteilung als Strichdiagramm

Analog zu den Befehlen `binomial` und `binomialteil` gibt es die Befehle `binomialstrich` und `binomialteilstrich`, die anstelle von Säulen nur Striche zeichnen.

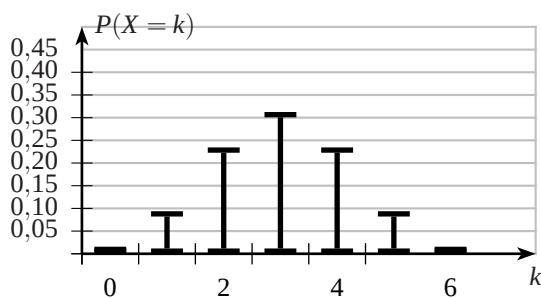


Abb. 7.29.: Die Binomialverteilung als Strichdiagramm

Listing 7.24: Die Binomialverteilung als Strichdiagramm

```

1 achsenmarkierung x="$k$" y="$P(X=k)$"
2 groesse x=6 y=3
3 achsen
4 gitter
5 binomialstrich linie=2,solid np=6,0.5,nein b
   =1,2

```

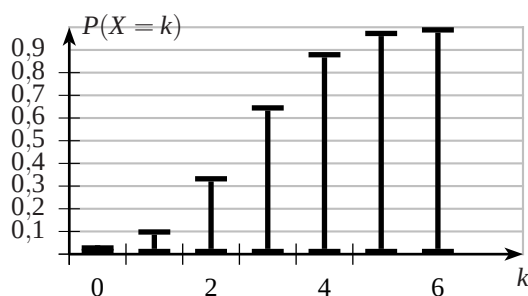


Abb. 7.30.: Die kumulierte Binomialverteilung als Strichdiagramm

Listing 7.25: Die kumulierte Binomialverteilung als Strichdiagramm

```

1 achsenmarkierung x="$k$" y="$P(X=k)$"
2 groesse x=6 y=3
3 achsen
4 gitter
5 binomialstrich linie=2,solid np=6,0.5,ja b
   =1,2

```

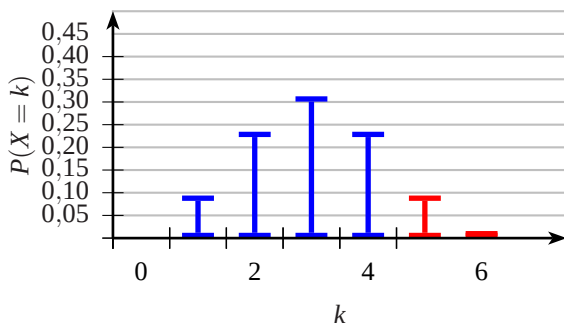


Abb. 7.31.: Die Binomialverteilung als Strichdiagramm teilweise dargestellt

Listing 7.26: Die Binomialverteilung als Strichdiagramm teilweise dargestellt

```

1 achsenmarkierung x="$k$" y="$P(X=k)$"
2 groesse x=6 y=3
3 achsen
4 gitter
5 mitte
6 binomialteilstrich linie=2,solid,blue np
   =6,0.5,1,4,nein b=1,2
7 binomialteilstrich linie=2,solid,red np
   =6,0.5,5,6,nein b=1,2

```


8. Dreidimensionale Abbildungen

Für die Oberstufenmathematik benötigt man in der Analytischen Geometrie einige dreidimensionale Zeichenelemente. So habe ich mein Skript auf mehrere Elemente im $\mathbb{R}^3$ erweitert. Ich habe mich bemüht so viele Zeichenroutinen wie möglich in den $\mathbb{R}^3$ zu übertragen, aber aus Zeitmangel ist diese Übertragung noch nicht vollständig. Die Zeichenelemente werden dabei durch die Parallelprojektion in die Bildebene transformiert.

8.1. Die Parallelprojektion

In der Abbildung 8.1 sieht man einen dreidimensionalen Punkt $P(x|y|z)$ im Koordinatensystem $(x|y|z)$, der mithilfe der Parallelprojektion in den Punkt $P^*(x^*|y^*)$ im System $(x_E|y_E)$ verwandelt wird. Der Winkel α stellt dabei die horizontale Rotation quer zu z-Achse dar. Der Winkel β ist die vertikale Rotation, bezogen auf die Papierebene. Die Transformation ist dabei die folgende:

$$x_E = -x \cos(\alpha) + y \sin(\alpha) \quad (8.1)$$

$$y_E = -(x \sin(\alpha) + y \cos(\alpha)) \cdot \sin(\beta) + z \cos(\beta) \quad (8.2)$$

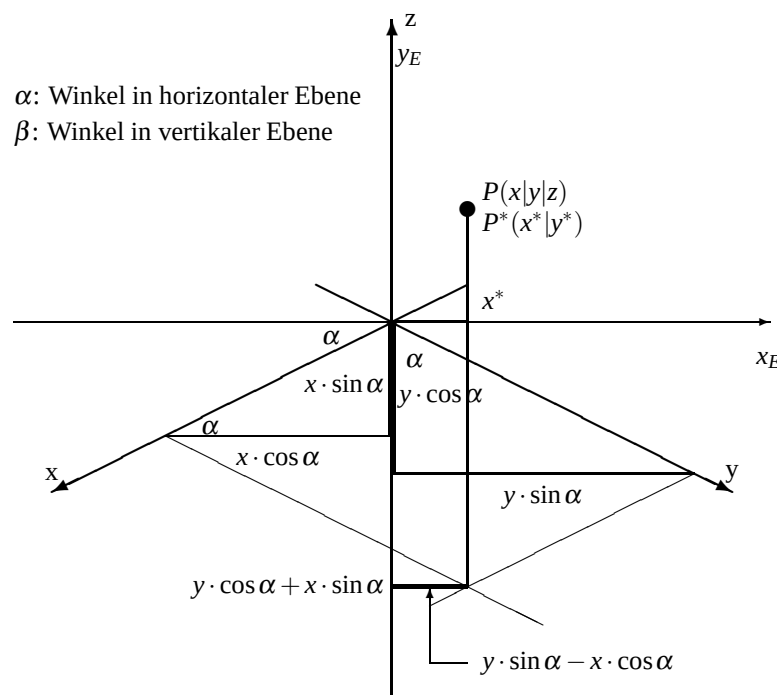


Abb. 8.1.: Koordinaten in einem dreidimensionalen Koordinatensystem (aus : (Voss, 2008, S. 355))

Diese Winkel kann man dabei mit dem Befehl `einstellung global` setzen.

einstellung

```
[layer=<n ∈ [0;100]>]
[alpha=<Winkel in Grad>]
[beta=<Winkel in Grad>]
[coortype=<0|1|2|3>]
[pstricks]
```

Dabei ist α auf 45, β auf 30 und coortype auf 2 gesetzt. Ist pstricks gesetzt, so werden alle Elemente mit dem pst-3plot -Paket¹ aus PsTricks gesetzt. Ansonsten werden alle Zeichenelemente intern transformiert.

Der Befehl coortype beschreibt weitere Projektionsverfahren, die die folgenden sind:

coortype=0 Die Parallelprojektion wie oben beschrieben.

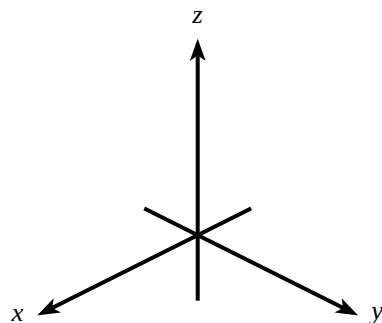


Abb. 8.2.: $\text{coortype}=0$

Listing 8.1: $\text{coortype}=0$

```
1 einstellung alpha=45 beta=30 coortype
  =0 pstricks
2 koordinatensystem x=-1,3 y=-1,3 z=-1,3
  einteilung=1,1,1 skala=keine text="
  $x$","$y$","$z$"
```

coortype=1 Die $y - z$ -Achsen sind orthogonal und der Winkel zwischen der x - und y -Achse ist durch α gegeben. Der Winkel β ist bedeutungslos.

$$x_E = y - x \cdot \sin(\alpha) \quad (8.3)$$

$$y_E = z - x \cdot \cos(\alpha - 90) \quad (8.4)$$

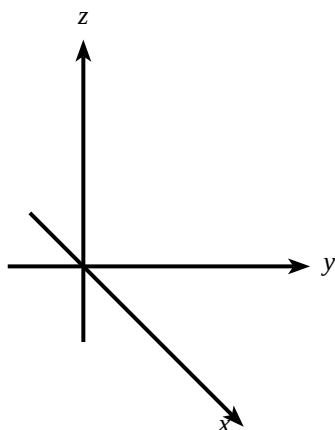


Abb. 8.3.: $\text{coortype}=1$

Listing 8.2: $\text{coortype}=1$

```
1 einstellung alpha=45 beta=30 coortype
  =1 pstricks
2 koordinatensystem x=-1,3 y=-1,3 z=-1,3
  einteilung=1,1,1 skala=keine text="
  $x$","$y$","$z$"
```

coortype=2 Die $y - z$ -Achsen sind orthogonal und der Winkel zwischen der x - und y -Achse ist 135° und die x -Achse ist um den Wert $\frac{1}{\sqrt{2}}$ verkürzt. Die Winkel α und β sind bedeutungslos.

$$x_E = y - 0,5x \quad (8.5)$$

$$y_E = z - 0,5x \quad (8.6)$$

¹Siehe (Voss, 2008, S. 338 ff)

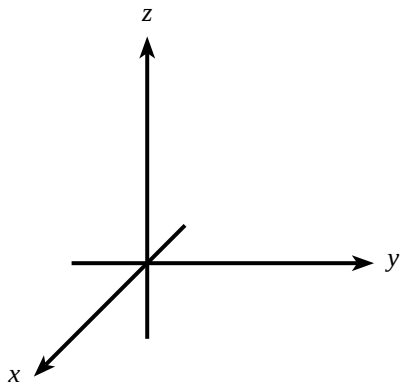


Abb. 8.4.: coortype=2

Listing 8.3: coortype=2

```

1 einstellung alpha=45 beta=30 coortype
  =2 pstricks
2 koordinatensystem x=-1,3 y=-1,3 z=-1,3
  einteilung=1,1,1 skala=keine text="
    $x$","$y$","$z$"

```

coortype=3 Die $y-z$ -Achsen sind orthogonal und der Winkel zwischen der x - und y -Achse ist 45° und die x -Achse ist um den Wert $\frac{1}{\sqrt{2}}$ verkürzt. Die Winkel α und β sind bedeutungslos.

$$x_E = y + 0,5x \quad (8.7)$$

$$y_E = z + 0,5x \quad (8.8)$$

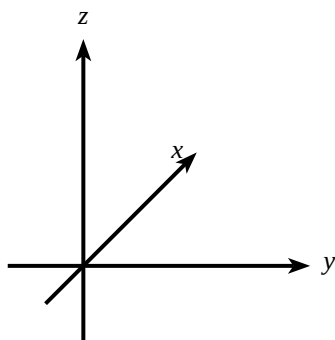


Abb. 8.5.: coortype=3

Listing 8.4: coortype=3

```

1 einstellung alpha=45 beta=30 coortype
  =3 pstricks
2 koordinatensystem x=-1,3 y=-1,3 z=-1,3
  einteilung=1,1,1 skala=keine text="
    $x$","$y$","$z$"

```

8.2. Zeichenbefehle für den $\mathbb{R}^3$

8.2.1. Linien, Vektoren und Polygone

Linien zeichnen

Auch im $\mathbb{R}^3$ werden viele Zeichenobjekten aus Linien aufgebaut. So ist die Linie erweitert worden:

linie

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,"<Pfeiltyp>"]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ,<y1 ∈ ℝ,<z1 ∈ ℝ>
p=<x2 ∈ ℝ,<y2 ∈ ℝ,<z2 ∈ ℝ>
[text="<Text>"]<Textdrehwinkel in Grad>,<Textausrichtung>,<x-Richtung>,<y-Richtung>,<z-Richtung>]
```

In p stehen die Koordinaten des Anfangs und Endpunktes der Linie. Zusätzlich kann noch ein Text mit **text** an die Linie geschrieben werden mit dem Befehl **text**, der optional ist. Wird nur ein Text für sich gesetzt, so wird der mittig von der Linie platziert.

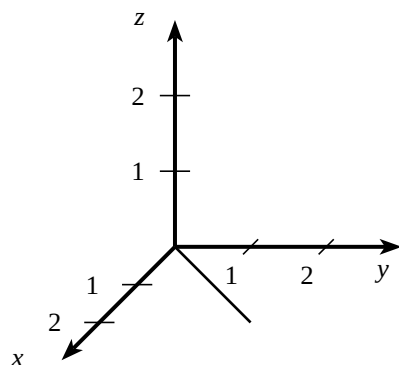


Abb. 8.6.: Der Befehl **linie** im $\mathbb{R}^3$

Listing 8.5: Der Befehl **linie** im $\mathbb{R}^3$

```
1 koordinatensystem x=0,3 y=0,3 z=0,3 einteilung
   =1,1,1 skala=normal text="$x$","$y$","$z$"
2 linie farbe=black linie=1,solid p=0,0,0 p=2,2,0
```

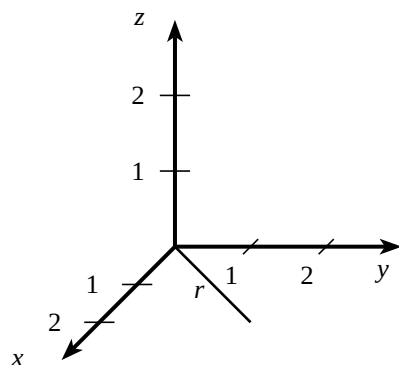


Abb. 8.7.: Der Befehl **linie** im $\mathbb{R}^3$ mit Text

Listing 8.6: Der Befehl **linie** im $\mathbb{R}^3$ mit Text

```
1 koordinatensystem x=0,3 y=0,3 z=0,3 einteilung
   =1,1,1 skala=normal text="$x$","$y$","$z$"
2 linie farbe=black linie=1,solid p=0,0,0 p=2,2,0
   text="$r$",0,180,0.2,0,0
```

Man kann auch an eine Linie mehrere Werte übergeben, damit diese Koordinaten miteinander verbunden werden. Dazu wird der Befehl **linie** etwas abgewandelt, wie man in Abbildung 8.8 sieht.

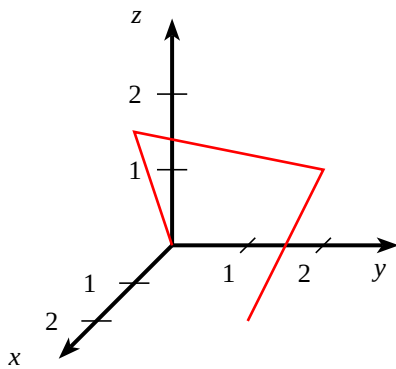


Abb. 8.8.: Der Befehl `linie` im $\mathbb{R}^3$ mit mehreren Wertepaaren

Listing 8.7: Der Befehl `linie` im $\mathbb{R}^3$ mit mehreren Wertepaaren

```

1 koordinatensystem x=0,3 y=0,3 z=0,3
   einteilung=1,1,1 skala=normal text="
   $x$","$y$","$z$"
2 linie farbe=red linie=1,solid
3 0,0,0
4 1,0,2
5 0,2,1
6 2,2,0
7 ende

```

Linien mit Verschiebungen

Ein neuer Befehl ist der Linienbefehl `vlinie`. Bei diesem gibt man von einem Startpunkt die Verschiebung in x -, y - und z -Richtung an.

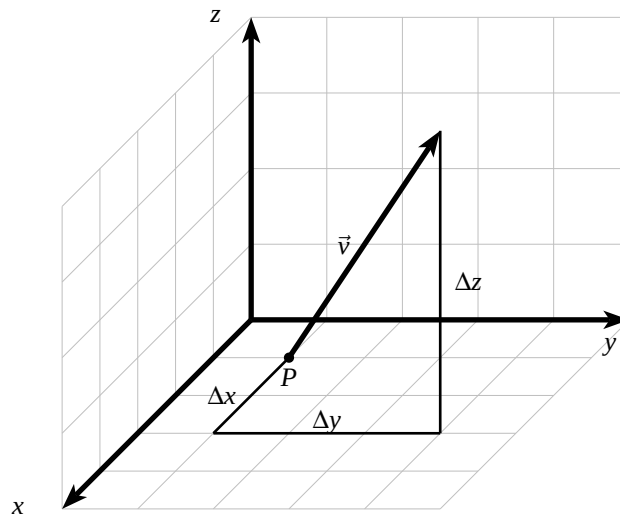


Abb. 8.9.: Eine Linie wird über seinen Startpunkt und die Verschiebung definiert.

vlinie

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Pfeiltyp>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ>,<z1 ∈ ℝ> r=<Δx ∈ ℝ>,<Δy ∈ ℝ>,<Δz ∈ ℝ>
[text="<Text>[,<Drehwinkel des Textes in Grad>,<Textausrichtung>,<x-Richtung>,<y-Richtung>]]

```

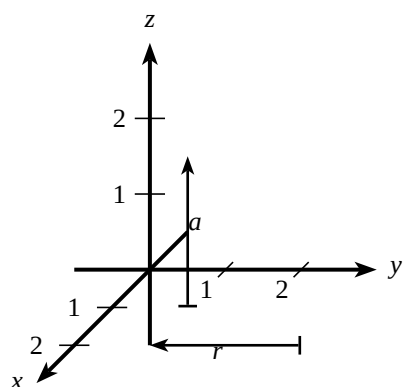


Abb. 8.10.: Der Befehl `vlinie` im $\mathbb{R}^3$

Listing 8.8: Der Befehl `vlinie` im $\mathbb{R}^3$

```

1 einstellung pstricks
2 koordinatensystem x=-1,3 y=-1,3 z=-1,3
   einteilung=1,1,1 skala=normal text="$x$","$y$","$z$"
3 vlinie linie=1,solid,"| ->" p=2,3,0 r=2,-1,1 text="$r$"
   0,180,0,2,0,0
4 vlinie farbe=black linie=1,solid,"| ->" p=1,1,0 r
   =2,1,3 text="$a$"

```

Vektoren zeichnen

Vektoren kann man natürlich wie unter 4.1.2 bereits im $\mathbb{R}^2$ beschrieben zeichnen, indem man den Pfeiltyp auf "`->`" setzt. Dies erledigt allerdings auch schon der Befehl `vektor` selbst, der analog zum Befehl der Linie ist.

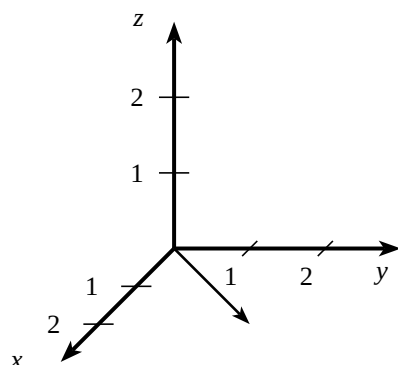


Abb. 8.11.: Der Befehl `vektor` im $\mathbb{R}^3$

Listing 8.9: Der Befehl `vektor` im $\mathbb{R}^3$

```

1 koordinatensystem x=0,3 y=0,3 z=0,3
   einteilung=1,1,1 skala=normal text="$x$","$y$","$z$"
2 vektor farbe=black linie=1,solid p=0,0,0 p
   =2,2,0

```

Polygonzüge im $\mathbb{R}^3$ zeichnen

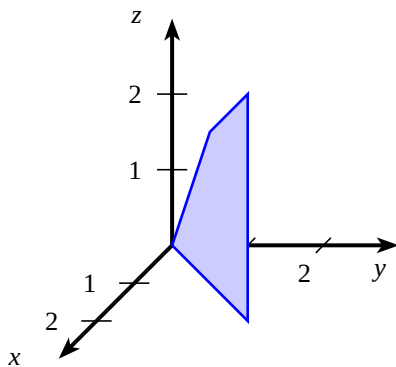
Möchte man anstelle von aneinandergereihten Linien einen geschlossenen Polygonzug zeichnen, so nutzt man den Befehl `polygon` wie im $\mathbb{R}^2$, der fast ähnlich zum Linienbefehl aussieht. Zusätzlich zu der Linienfarbe kann mit `farbe` die Linienfarbe und Füllfarbe definiert werden.

`polygon`

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
[lese <Dateiname>] oder [<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>] beliebig oft
ende

```

Abb. 8.12.: Der Befehl `polygon` im $\mathbb{R}^3$ Listing 8.10: Der Befehl `polygon` im $\mathbb{R}^3$

```

1 koordinatensystem x=0,3 y=0,3 z=0,3
   einteilung=1,1,1 skala=normal text="
   $x$","$y$","$z$"
2 polygon farbe=blue,solid,blue!20 linie=1,solid,
   blue parameter=""
3 0,0,0
4 1,1,2
5 0,1,2
6 2,2,0
7 ende

```

8.2.2. Dreiecke und Vierecke im $\mathbb{R}^3$ zeichnen

Dreiecke

Wie im $\mathbb{R}^2$ gibt es auch hier den Befehl des Dreiecks, der analog zum Befehl unter Abschnitt 4.2.1 ist. Die Syntax ist nur um die z-Koordinate erweitert.

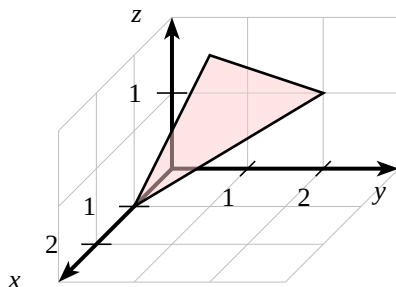
dreieck

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ>,<z1 ∈ ℝ>
p=<x2 ∈ ℝ>,<y2 ∈ ℝ>,<z2 ∈ ℝ>
p=<x3 ∈ ℝ>,<y3 ∈ ℝ>,<z3 ∈ ℝ>

```

Beim Dreieck werden die Koordinaten der drei Eckpunkte angegeben.

Abb. 8.13.: Der Befehl `dreieck` im $\mathbb{R}^3$ Listing 8.11: Der Befehl `dreieck` im $\mathbb{R}^3$

```

1 koordinatensystem x=0,3 y=0,3 z=0,2 einteilung
   =1,1,1 skala=normal text="$x$","$y$","$z$"
2 gitter
3 dreieck farbe=black,solid,red!20 linie=1,solid
   parameter="opacity=0.5" p=1,0,0 p=1,1,2 p=2,3,2

```

Vierecke

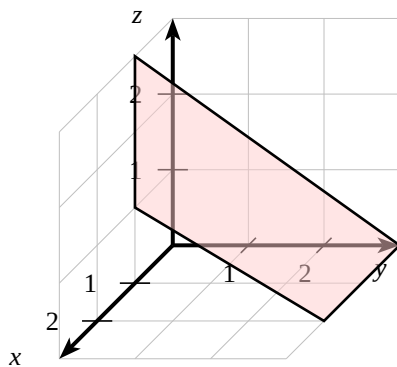
Bei einem **Viereck** übergibt man wie unter 4.2.3 an den Startwert und die drei weiteren Koordinaten im $\mathbb{R}^3$.

viereck

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ>,<z1 ∈ ℝ> p=<x2 ∈ ℝ>,<y2 ∈ ℝ>,<z2 ∈ ℝ>
p=<x3 ∈ ℝ>,<y3 ∈ ℝ>,<z3 ∈ ℝ> p=<x4 ∈ ℝ>,<y4 ∈ ℝ>,<z4 ∈ ℝ>

```

Abb. 8.14.: Der Befehl `viereck` im $\mathbb{R}^3$ Listing 8.12: Der Befehl `viereck` im $\mathbb{R}^3$

```

1 koordinatensystem x=0,3 y=0,3 z=0,3
   einteilung=1,1,1 skala=normal
   text="$x$", "$y$", "$z$"
2 gitter
3 viereck farbe=black,solid,red!20 linie=1,
   solid parameter="opacity=0.5" p=1,0,1 p
   =2,3,0 p=0,3,0 p=1,0,3

```

8.2.3. Ellipsen und Kreise

Ausgehend von einer Ellipse im $\mathbb{R}^2$ lautet die Gleichung für eine **Ellipse** im $\mathbb{R}^3$

$$\vec{x} = \vec{m} + \cos(\alpha) \cdot \vec{u} + \sin(\alpha) \cdot \vec{v}; \quad 0^\circ \leq \alpha \leq 360^\circ \quad (8.9)$$

wobei $\vec{m}$ der Mittelpunkt der Ellipse ist und $\vec{u}$ und $\vec{v}$ die senkrecht aufeinander stehenden Vektoren der Halbachsen sind. Die Syntax zur Erstellung einer Ellipse ist dann

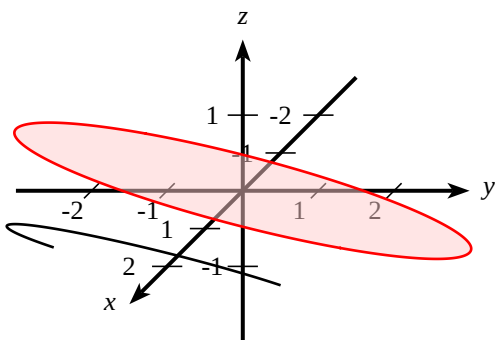
ellipse

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<mx ∈ ℝ>,<my ∈ ℝ>,<mz ∈ ℝ>
r=<ux ∈ ℝ>,<uy ∈ ℝ>,<uz ∈ ℝ>
r=<vx ∈ ℝ>,<vy ∈ ℝ>,<vz ∈ ℝ>
[r=<Startwinkel für α>,<Endwinkel für α>]

```

Standardmäßig wird die Ellipse zwischen 0 und 360 Grad gezeichnet.

Abb. 8.15.: Der Befehl `ellipse` im $\mathbb{R}^3$

Ein **Kreis** ist der Spezialfall der Ellipse mit $|\vec{u}| = |\vec{v}| = r$. Somit braucht man nur einen Vektor anzugeben, um einen Kreis zu zeichnen:

kreis

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<mx ∈ ℝ>,<my ∈ ℝ>,<mz ∈ ℝ>

```

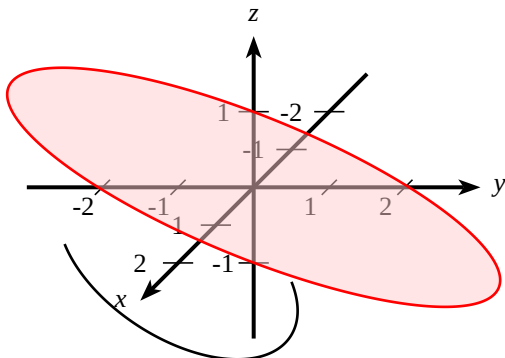
Listing 8.13: Der Befehl `ellipse` im $\mathbb{R}^3$

```

1 einstellung pstricks
2 koordinatensystem x=-3,3 y=-3,3 z
   =-2,2 einteilung=1,1,1 skala=
   normal text="$x$", "$y$", "$z$"
3 ellipse farbe=black,solid,red!20 linie=1,
   solid,red parameter="opacity=0.5" p
   =0,0,0 r=0.5,-1,1 r=1,-2.25,1
4 ellipse p=2,0,0 r=1,-1,0.75 r=1,-1,1 r
   =0,180

```

$r = \langle u_x \in \mathbb{R}, \langle u_y \in \mathbb{R}, \langle u_z \in \mathbb{R} \rangle \rangle$
 $[r = \langle \text{Startwinkel für } \alpha \rangle, \langle \text{Endwinkel für } \alpha \rangle]$

Abb. 8.16.: Der Befehl `kreis` im $\mathbb{R}^3$ Listing 8.14: Der Befehl `kreis` im $\mathbb{R}^3$

```

1 einstellung coortype=2 pstricks
2 koordinatensystem x=-3,3 y=-3,3 z
  =-2,2 einteilung=1,1,1 skala=
  normal text="$x$", "$y$", "$z$"
3 kreis farbe=black,solid,red!20 linie=1,
  solid,red parameter="opacity=0.5" p
  =0,0,0 r=1,-2.25,1
4 kreis p=2,0,0 r=1,-1,0.75 r=0,180

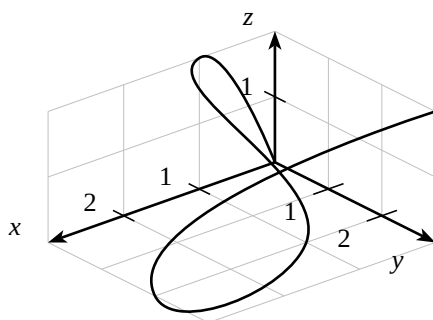
```

8.2.4. Kurven zeichnen

Ähnlich zum Linienbefehl gibt es noch den Befehl der **Kurve** ebenfalls im dreidimensionalen Raum. Hier wird der Befehl `kurve` einfach um die z-Koordinate erweitert.

kurve
 $[layer = \langle n \in [0; 100] \rangle]$
 $[farbe = \langle \text{Linienfarbe} \rangle]$
 $[linie = \langle \text{Linienstärke in pt } (l \in \mathbb{R}_{>0}) \rangle, \langle \text{Linienart} \rangle, \langle \text{Pfeiltyp} \rangle]$
 $[parameter = \langle \text{Pstricksbefehle} \rangle]$
 $\langle x \in \mathbb{R} \rangle, \langle y \in \mathbb{R} \rangle, \langle z \in \mathbb{R} \rangle$ beliebig oft
ende

Dabei ist zu beachten, dass bei diesem Befehl die Kurve durch alle angegebenen Punkte geht.

Abb. 8.17.: Eine Kurve im $\mathbb{R}^3$ Listing 8.15: Eine Kurve im $\mathbb{R}^3$

```

1 einstellung coortype=0
2 achsen
3 gitter
4 kurve farbe=black linie=1,solid
5 0,0,0
6 1,0,2
7 1,2,0
8 3,2,0
9 0,3,2
10 ende

```

Zusätzlich zur Kurve gibt es noch die Bezierkurve auch im dreidimensionalen. (Siehe 4.1) Dabei ist der Befehl `bezier` um die z-Koordinate erweitert. Dabei werden intern die Punkte in zweidimensionale Punkte übertragen und dann weiter verarbeitet.

bezier
 $[layer = \langle n \in [0; 100] \rangle]$
 $[farbe = \langle \text{Linienfarbe} \rangle]$
 $[linie = \langle \text{Linienstärke in pt } (l \in \mathbb{R}_{>0}) \rangle, \langle \text{Linienart} \rangle, \langle \text{Pfeiltyp} \rangle]$
 $[parameter = \langle \text{Pstricksbefehle} \rangle]$
 $\langle x \in \mathbb{R} \rangle, \langle y \in \mathbb{R} \rangle, \langle z \in \mathbb{R} \rangle$ beliebig oft
ende

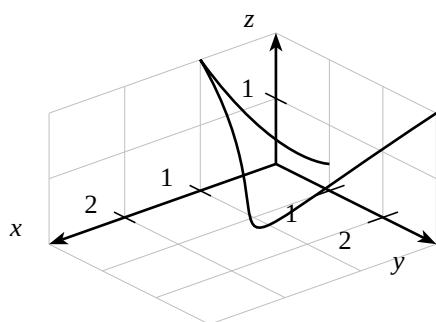


Abb. 8.18.: Eine Bezierkurve im $\mathbb{R}^3$

Listing 8.16: Eine Bezierkurve im $\mathbb{R}^3$

```

1 einstellung coortype=0 pstricks
2 achsen
3 gitter
4 bezier farbe=black linie=1,solid
5 0,0,0
6 1,0,2
7 1,2,0
8 3,2,0
9 0,3,2
10 ende

```

8.2.5. Beschriftungen, Punkte und Kreuze im $\mathbb{R}^3$ zeichnen

Beschriftungen

Auch im $\mathbb{R}^3$ gibt es denn Befehl der **Beschriftung**, der wie folgt aussieht:

beschriftung

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
text="<Text>",<Textwinkel in Grad>,<Drehung in Grad>

```

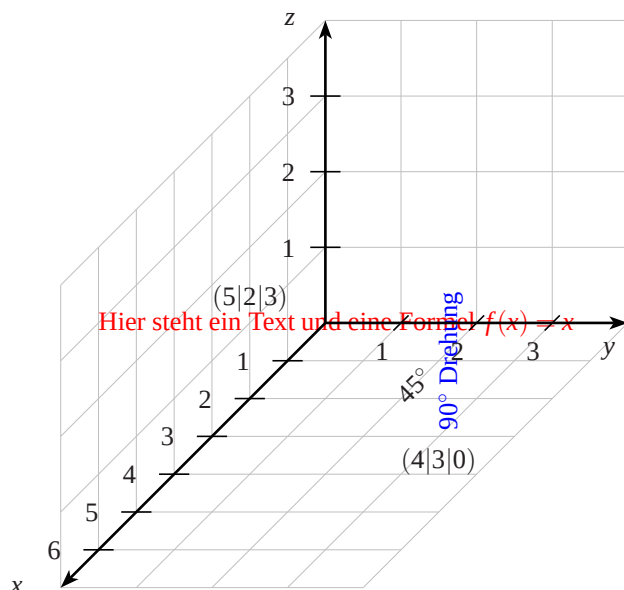


Abb. 8.19.: Der Befehl beschriftung im $\mathbb{R}^3$

Listing 8.17: Der Befehl beschriftung im $\mathbb{R}^3$

```

1 grenzgitter x=0,7 y=0,4 z=0,4
2 achsen
3 beschriftung farbe=red p=6,0,3 text="
  Hier steht ein Text und eine Formel $f(x)=
  x$",0,r
4 beschriftung farbe=blue p=1,2,0 text="
  $90^{\circ}$ Drehung",90,0
5 beschriftung p=4,3,1 text="$45^{\circ}$"
  ,45,45
6 beschriftung p=4,3,0 text="$ (4|3|0)$"
  ,0,45
7 beschriftung p=5,2,3 text="$ (5|2|3)$"
  ,0,225

```

Punkte zeichnen

Möchte man einen Koordinatenpunkt $(x|y|z)$ mit Beschriftung versehen, so gibt es den Befehl **punkt**, an den man einen Text übergeben kann. Die Syntax sieht wie folgt aus:

punkt

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]

```

```
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<Radius r ∈ ℝ>>0 des Punktes>
[text="<Text>"],<Drehung in Grad>]]
```

Der Radius eines Punktes ist standardmäßig auf 0,05 gesetzt.

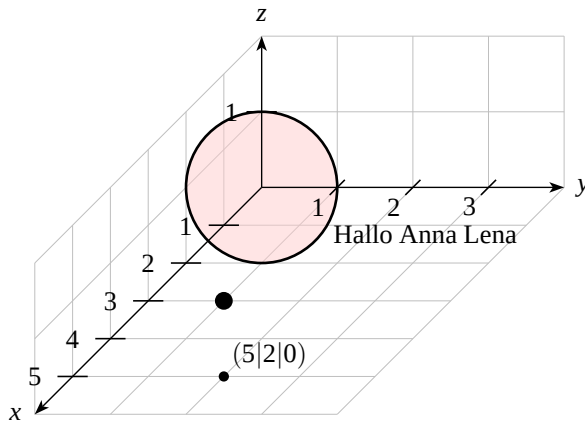


Abb. 8.20.: Der Befehl `punkt` im $\mathbb{R}^3$

Listing 8.18: Der Befehl `punkt` im $\mathbb{R}^3$

```
1 einstellung alpha=45 beta=30
   coortype=2 pstricks
2 grenzgitter x=0,6 y=0,4 z=0,2
3 achsen
4 punkt farbe=black,solid,red!20 linie=1,
   solid parameter="opacity=0.5" p=0,0,0 r
   =1 text="Halo Anna Lena",-30
5 punkt p=5,2,0 text="$(5|2|0)$"
6 punkt p=3,1,0 r=0.1
```

Kreuze zeichnen

Möchte man anstelle eines Koordinatenpunkt $(x|y|z)$ ein **Kreuz** zeichnen, so gibt es den Befehl `kreuz`, an den man einen Text übergeben kann. Die Syntax sieht wie folgt aus:

kreuz

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<Länge des Kreuzes l ∈ ℝ>>0>,<Drehwinkel des Kreuzes in Grad>]
[text="<Text>"],<Drehung in Grad>]]
```

Standardmäßig ist die Länge des Kreuzes auf 0,5 LE gesetzt und der Drehwinkel beträgt 45° . Die Drehung des Textes ist als Standard ebenfalls 45° .

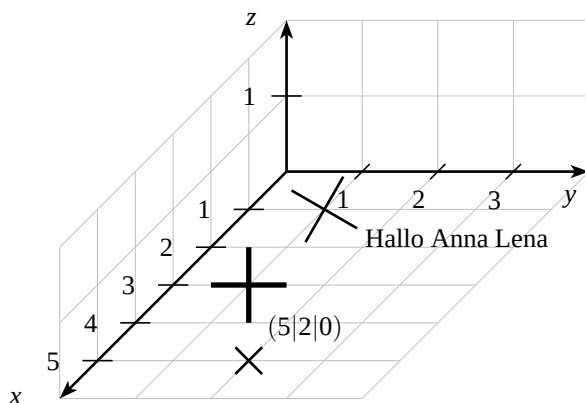


Abb. 8.21.: Der Befehl `kreuz` im $\mathbb{R}^3$

Listing 8.19: Der Befehl `kreuz` im $\mathbb{R}^3$

```
1 einstellung coortype=2
2 grenzgitter x=0,6 y=0,4 z=0,2
3 achsen
4 kreuz farbe=black linie=1,solid
   parameter="" p=1,1,0 r=1,30 text="
   Hallo Anna Lena",-30
5 kreuz p=5,2,0 text="$(5|2|0)$"
6 kreuz linie=2,solid p=3,1,0 r=1,0
7 r
```

8.2.6. Grafiken einfügen

Möchte man eine **Grafik** in eine Zeichnung mit einfügen, so gibt es wie unter 4.5 beschrieben den Befehl `bild`, der einem dies ermöglicht. Dieser ist ebenfalls für dreidimensionale Objekte erweitert.

bild

```
[layer=<n ∈ [0;100]>]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<Breite des Bildes l ∈ ℝ>>[,<Drehwinkel des Bildes in Grad>]
<"Dateiname">
```

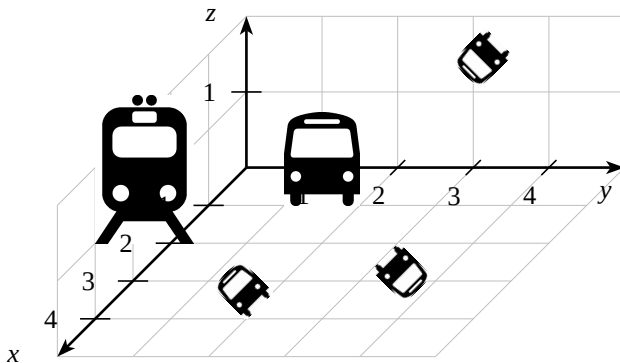


Abb. 8.22.: Eine Grafik einfügen

Listing 8.20: Eine Grafik einfügen

```
1 einstellung coortype=2
2 grenzgitter x=0,5 y=0,5 z=0,2
3 achsen
4 bild p=1,1,0 r=1 Abbildungen/bus.eps
5 bild p=4,2,0 r=0.5,45 Abbildungen/bus.eps
6 bild p=1,4,2 r=0.5,135 Abbildungen/bus.eps
7 bild p=2,3,0 r=0.5,225 Abbildungen/bus.eps
8 bild p=4,0,1 r=1.3 Abbildungen/zug.eps
```

8.3. Koordinatensysteme und Gitter im $\mathbb{R}^3$

8.3.1. Einfache Koordinatensysteme

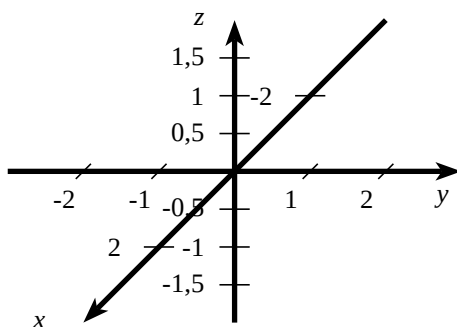
Die Syntax für ein dreidimensionales **koordinatensystem** ist:

koordinatensystem

```
[layer=<n ∈ [0;100]>]
x=<xmin ∈ ℝ>,<xmax ∈ ℝ> y=<ymin ∈ ℝ>,<ymax ∈ ℝ> z=<zmin ∈ ℝ>,<zmax ∈ ℝ>
einteilung=<xSkala ∈ ℝ>>,<ySkala ∈ ℝ>>,<zSkala ∈ ℝ>>
[linie=<Linienstärke in pt (l ∈ ℝ>>,<Linienart>[,<Linienfarbe>]]
skala=<keine|nurstrichel|nureins|normal|strichundeins>
text="Text an der x-Achse","Text an der y-Achse","Text an der z-Achse"
```

Einige Beispiele

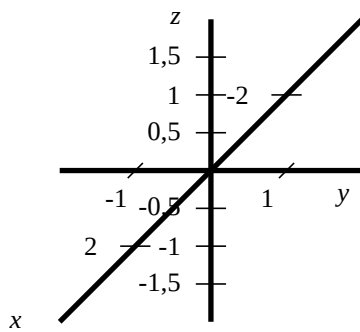
Hier nun ein paar einfache Beispiele für Koordinatenachsen:



Listing 8.21: Ein erstes Koordinatensystem im $\mathbb{R}^3$

```
1 koordinatensystem x=-4,4 y=-3,3 z=-2,2
   einteilung=2,1,0.5 linie=2,solid,black
   skala=normal text="$x$","$y$","$z$"
```

Abb. 8.23.: Ein erstes Koordinatensystem im $\mathbb{R}^3$



Listing 8.22: Ein zweites Koordinatensystem im $\mathbb{R}^3$

```
1 koordinatensystem x=-4,4 y=-2,2 z=-2,2
   einteilung=2,1,0.5 linie=2,solid,black,"-"
   skala=normal text="$x$","$y$","$z$"
```

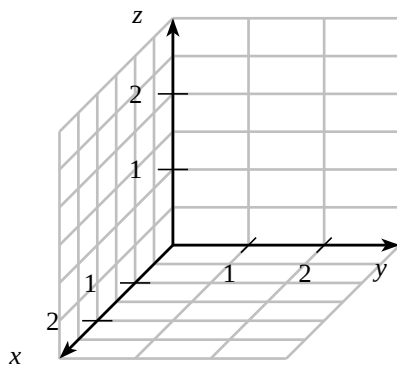
Abb. 8.24.: Ein zweites Koordinatensystem im $\mathbb{R}^3$

8.3.2. Gitter zeichnen im $\mathbb{R}^3$

Die Syntax für ein Gitter im $\mathbb{R}^3$ lautet

kariert

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
dx=<dx1 ∈ ℝ>0>,<dx2 ∈ ℝ>0>,<dx3 ∈ ℝ>0>
dy=<dy1 ∈ ℝ>0>,<dy2 ∈ ℝ>0>,<dy3 ∈ ℝ>0>
dz=<dz1 ∈ ℝ>0>,<dz2 ∈ ℝ>0>,<dz3 ∈ ℝ>0>
x=<xmin ∈ ℝ>,<xmax ∈ ℝ>
y=<ymin ∈ ℝ>,<ymax ∈ ℝ>
z=<zmin ∈ ℝ>,<zmax ∈ ℝ>
```



Listing 8.23: Der Befehl `kariert` im $\mathbb{R}^3$

```
1 achsen
2 kariert farbe=lightgray linie=1,solid dx
   =0.5,0,0 dy=1,0,0 dz=0.5,0,0 x=-0,3 y=0,3
   z=0,3
```

Abb. 8.25.: Der Befehl `kariert` im $\mathbb{R}^3$

8.4. Plotten von Funktionen und Daten im $\mathbb{R}^3$

Analog zu Abschnitt 6 gibt es auch für den $\mathbb{R}^3$ die Möglichkeit Funktionen zu zeichnen.

8.4.1. Funktionen zeichnen im $\mathbb{R}^3$

Um Funktionen der Form $f(x, y)$ zu zeichnen habe ich den Befehl `graph` erweitert. Die Syntax ist die folgende:

graph

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
```

```
[linie=<Linienstärke in pt ( $l \in \mathbb{R}_{>0}$ )>,<Linienart>]
[parameter="<Pstricksbefehle>"]
x=< $x_{\min} \in \mathbb{R}$ >,< $x_{\max} \in \mathbb{R}$ >
y=< $y_{\min} \in \mathbb{R}$ >,< $y_{\max} \in \mathbb{R}$ >
[z=< $z_{\min} \in \mathbb{R}$ >,< $z_{\max} \in \mathbb{R}$ >]
<Funktionsterm  $f(x, y)$ >
```

Hierbei wird der Funktionsterm mit den Variablen x und y in der algebraischen Notation erwartet. Werden $z_{\min}$ und $z_{\max}$ nicht angegeben, so wird die Funktion nicht ohne Beschränkung gezeichnet. Werden diese Werte angegeben, so werden intern die Werte $y_{\min}$ und $y_{\max}$ beschränkt.

Im Folgenden betrachte ich einmal die Funktion

$$f(x, y) = 15 \cdot (x^3 - x \cdot y^2 - \frac{x}{4}) \cdot \exp(-(x^2 + y^2)) \quad (8.10)$$

genauer.

Es sind dabei die gleichen Parameter wie in PStricks implementiert². Es gibt dabei die folgenden Parameter.

Name	Vorgabe
xPlotpoints	25
yPlotpoints	25
drawStyle	xLines

Tab. 8.1.: Parameter für dreidimensionale Funktionen

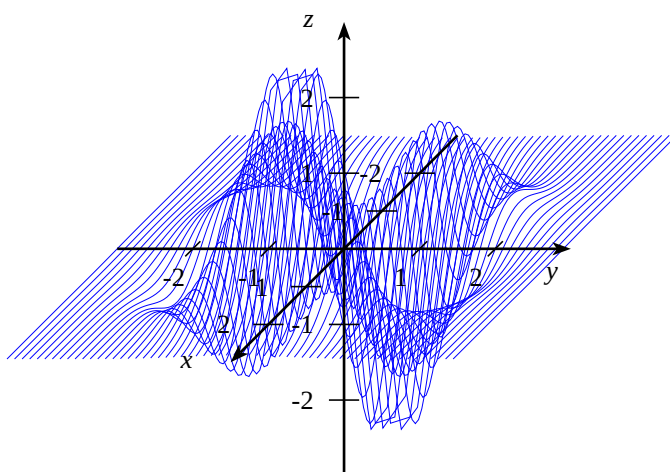


Abb. 8.26.: Der Befehl graph im $\mathbb{R}^3$

Listing 8.24: Der Befehl graph im $\mathbb{R}^3$

```
1 einstellung coortype=2
2 achsen
3 graph layer=50 farbe=blue linie
   =0.1,solid parameter="xPlotpoints
   =70,yPlotpoints=60" x=-3,3 y=-3,3
   z=-3,3 10*(x^3-x*y^2-x/4)*exp
   (-(x^2+y^2))
```

Setzt man bei den Einstellungen den Parameter pstricks, so wird die interne Pstricks Funktion pstplotThreeD genutzt. Dann kann auch der Parameter hiddenLine genutzt werden.

²Siehe (Voss, 2008, S. 379 ff)

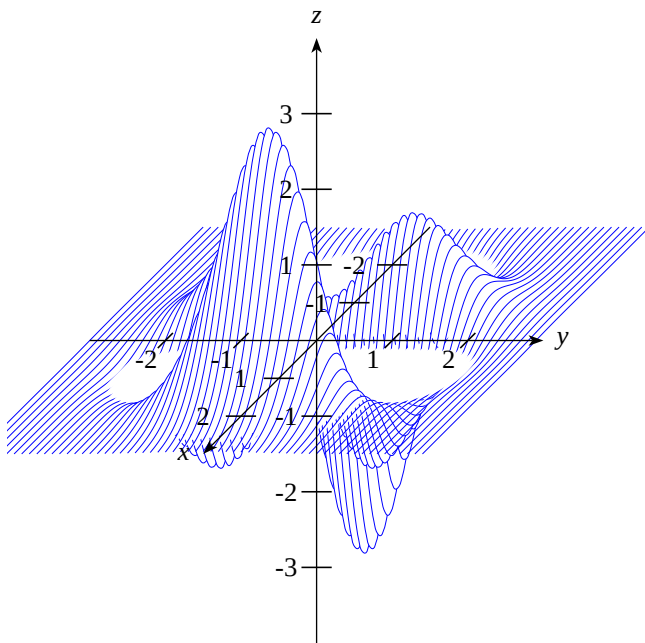


Abb. 8.27.: Der Befehl `graph` mit `pstricks` und `hiddenLine`

Listing 8.25: Der Befehl `graph` mit `pstricks` und `hiddenLine`

```
1 einstellung coortype=2 pstricks
2 achsen
3 graph layer=50 farbe=blue linie
   =0.1,solid parameter="xPlotpoints
   =70,yPlotpoints=60,hiddenLine=true
   " x=-3,3 y=-3,3 10*(x^3-x*y^2-x
   /4)*exp(-(x^2+y^2))
```

8.4.2. Parametrisierte Kurven im $\mathbb{R}^3$

Im $\mathbb{R}^3$ kann ebenfalls eine Parametrisierung einer Kurve dargestellt werden. Ein Beispiel hierfür ist die Darstellung einer Spirale in der Form

$$f(x, y, z) = \left(\sin(t), \cos(t), \frac{t}{500} \right) \quad (8.11)$$

Die Syntax des Befehls ist die folgende:

```
parametricplot
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
[parameter="<Pstricksbefehle>"]
t=<tmin ∈ ℕ>,<tmax ∈ ℕ>
[u=<umin ∈ ℕ>,<umax ∈ ℕ>]
<Funktionsterm x(t,u)>,<Funktionsterm y(t,u)>,<Funktionsterm z(t,u)>
```

Wobei $[t_{\min}; t_{\max}]$ und $[u_{\min}; u_{\max}]$ die jeweiligen Definitionsintervalle beschreiben.

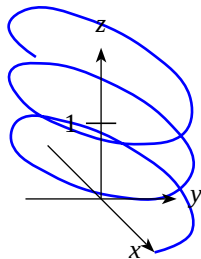


Abb. 8.28.: Der Befehl `parametricplot` im $\mathbb{R}^3$

Listing 8.26: Der Befehl `parametricplot` im $\mathbb{R}^3$

```
1 einstellung coortype=1 pstricks
2 achsen
3 parametricplot layer=50 farbe=blue
   linie=1,solid parameter="plotstyle=
   curve" t=0,1000 cos(t),sin(t),t/500
```

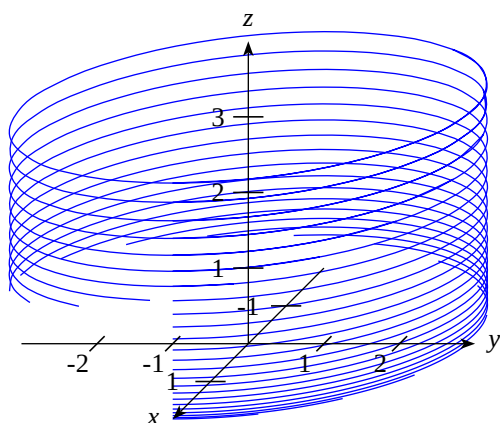


Abb. 8.29.: Der Befehl `parametricplot` mit dem Intervall u

Listing 8.27: Der Befehl `parametricplot` mit dem Intervall u

```
1 achsen
2 parametricplot layer=50 farbe=blue
   linie=0.5,solid parameter="plotstyle=
   curve" t=0,100 u=0,5 2*cos(t)*u,3*sin(t)
   )*u,(u^2)/8
```

8.4.3. Plotten von Daten im $\mathbb{R}^3$

Das Plotten von Daten im dreidimensionalen Raum läuft effektiv analog zum zweidimensionalen Raum ab. Siehe dazu Abschnitt 6.10. Es wird jeweils nur die z-Koordinate ergänzt. Die Syntax ist dabei bei allen Funktionen die folgende:

```
[<fileplot | listplot | dataplot>]
  [layer=<n ∈ [0;100]>]
  [farbe=<Linienfarbe>]
  [linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>]
  [parameter="<Pstricksbefehle>"]
  [lese <Dateiname>] oder <x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ> beliebig oft
ende
```

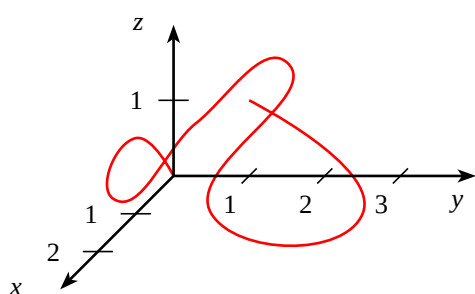


Abb. 8.30.: Der Befehl `listplot` im $\mathbb{R}^3$

Listing 8.28: Der Befehl `listplot` im $\mathbb{R}^3$

```
1 achsen
2 listplot farbe=red linie=1,solid
   parameter="plotstyle=curve"
3 0,0,0
4 1,0,1
5 2,0,2,0.7
6 0,0,3,0.7
7 1,2,2
8 1,1,0
9 3,4,1
10 2,2,2
11 ende
```

8.5. Objekte für die analytische Geometrie

Für meinen Mathematikunterricht in der Oberstufe brauche ich Elemente wie Geraden, Ebenen und Kugeln. Um dies einfach eingeben zu können, habe ich mir ein paar Zeichenelemente definiert.

8.5.1. Geraden im Raum

Die vektorielle Parametergleichung einer **Geraden** ist gegeben mit

$$g : \vec{x} = \vec{a} + r \cdot \vec{m} \quad (8.12)$$

wobei $\vec{a}$ der Stützvektor und $\vec{m} \neq 0$ der Richtungsvektor der Geraden g sind.

Die Syntax ist dabei ähnlich zur Linie:

gerade

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>₀)>,<Linienart>[,<Pfeiltyp>]]
[parameter="<Pstricksbefehle>"]
p=<aₓ ∈ ℝ>,<aᵧ ∈ ℝ>,<a_z ∈ ℝ>
p=<mₓ ∈ ℝ>,<mᵧ ∈ ℝ>,<m_z ∈ ℝ>
r=<r_Start ∈ ℝ>,<r_End ∈ ℝ>
```

r_{Start} und r_{Ende} sind dabei die Werte, für die der Start- und Endpunkt berechnet werden.

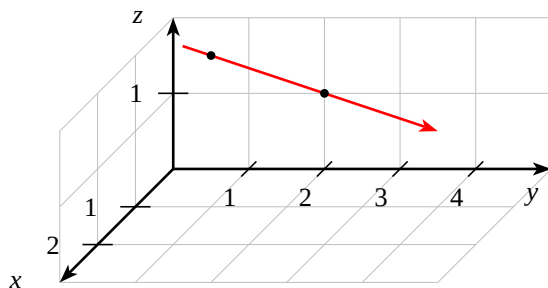


Abb. 8.31.: Der Befehl gerade

Listing 8.29: Der Befehl gerade

```
1 einstellung coortype=2
2 achsen
3 gitter
4 gerade farbe=red linie=1,solid,"->" p
   =1,1,2 p=1,2,0 r=-0.25,2
5 punkt p=1,1,2 r=0.04
6 punkt p=2,3,2 r=0.04
```

8.5.2. Ebenengleichungen

Neben der Geraden ist die **Ebene** ein wichtiges Objekt in der analytischen Geometrie.

Dabei gibt es verschiedene Darstellungsformen einer Ebene. Ich habe in meinem Skript die Parametergleichung und Koordinatengleichung realisiert. Die anderen Formen sind jeweils einfach umwandelbar.

Parametergleichung einer Ebene

Die vektorielle Parametergleichung einer **Ebene** ist gegeben mit

$$E : \vec{x} = \vec{a} + r \cdot \vec{u} + s \cdot \vec{v} \quad (8.13)$$

wobei $\vec{a}$ der Stützvektor und $\vec{u}, \vec{v} \neq 0$ die Richtungsvektoren und $r, s \in \mathbb{R}$ die Ebenenparameter sind.

Die Syntax ist dabei die folgende:

ebene

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>₀)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<aₓ ∈ ℝ>,<aᵧ ∈ ℝ>,<a_z ∈ ℝ>
p=<uₓ ∈ ℝ>,<uᵧ ∈ ℝ>,<u_z ∈ ℝ>
p=<vₓ ∈ ℝ>,<vᵧ ∈ ℝ>,<v_z ∈ ℝ>
[r=<r_Start ∈ ℝ>,<r_End ∈ ℝ>]
[s=<s_Start ∈ ℝ>,<s_End ∈ ℝ>]
```

Sind die Ebenenparameter nicht gesetzt, so ist der Startwert 0 und der Endwert 1.

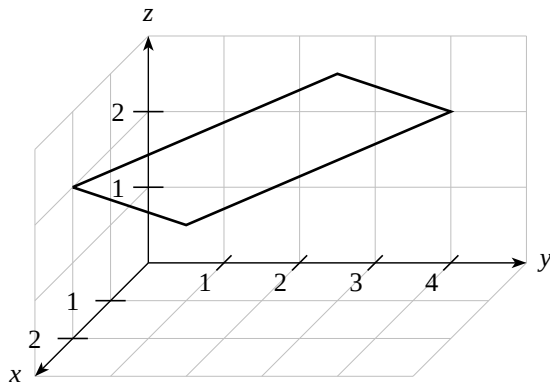


Abb. 8.32.: Der Befehl `ebene` in der Parametergleichung

Listing 8.30: Der Befehl `ebene` in der Parametergleichung

```
1 einstellung coortype=2 pstricks
2 achsen
3 gitter
4 ebene linie=1,solid p=2,0,2 p=1,2,0 p
   =-1,3,1
```

Die Koordinatengleichung einer Ebene

Eine Ebene im dreidimensionalen Anschauungsraum lässt sich stets durch eine lineare Gleichung der Form

$$ax + by + cz = d \quad (8.14)$$

darstellen, die man als **Koordinatengleichung** bezeichnet. Die Syntax ist dabei die Folgende:

ebene

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
r=<a ∈ ℝ>,<b ∈ ℝ>,<c ∈ ℝ>,<d ∈ ℝ>[,<n1 ∈ ℝ>][,<n2 ∈ ℝ>]
```

Wobei n_1 und n_2 standardmäßig auf 1 gesetzt sind. Diese Werte stehen dafür, wie weit eine Ebene gezeichnet werden soll.

Beispiele Im Folgenden sind drei Ebenen in der Koordinatenform dargestellt.

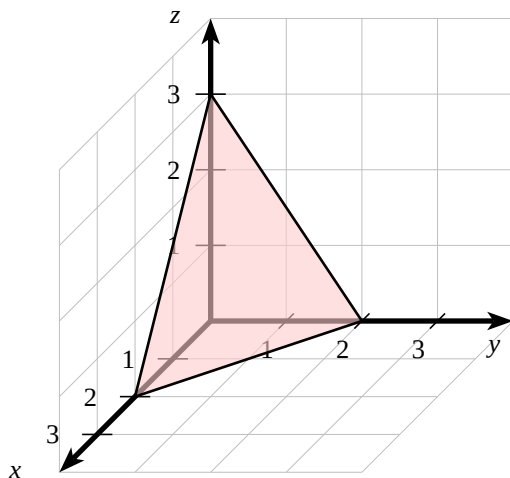
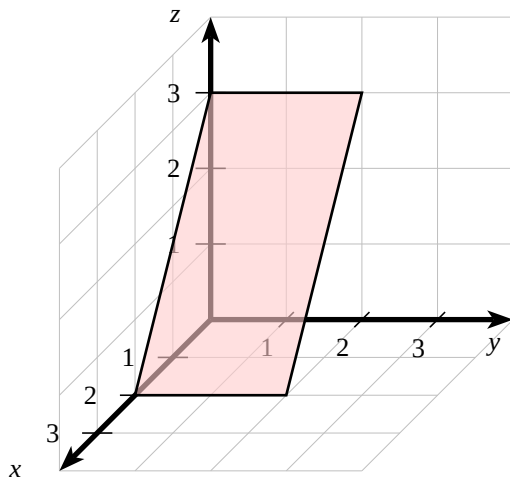


Abb. 8.33.: Der Befehl `ebene` in der Koordinatengleichung

Listing 8.31: Der Befehl `ebene` in der Koordinatengleichung

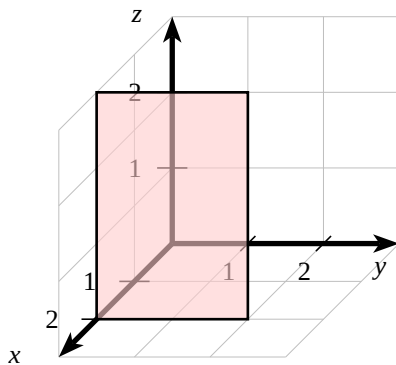
```
1 gitter
2 koordinatensystem x=0,4 y=0,4 z=0,4
   einteilung=1,1,1 linie=2,solid,black
   skala=normal text="$x$","$y$","$z$"
3 ebene layer=11 farbe=black,shape,red!20
   linie=1,solid r=6,6,4,12
```

Abb. 8.34.: Die Ebene $6x + 4z = 12$ Listing 8.32: Die Ebene $6x + 4z = 12$

```

1 gitter
2 koordinatensystem x=0,4 y=0,4 z=0,4
   einteilung=1,1,1 linie=2,solid,black
   skala=normal text="$x$","$y$","$z$"
3 ebene layer=11 farbe=black,shape,red!20
   linie=1,solid r=6,0,4,12,2

```

Abb. 8.35.: Die Ebene $6x + 4z = 12$ Listing 8.33: Die Ebene $6x + 4z = 12$

```

1 einstellung coortype=2
2 gitter
3 koordinatensystem x=0,3 y=0,3 z=0,3
   einteilung=1,1,1 linie=2,solid,black
   skala=normal text="$x$","$y$","$z$"
4 ebene layer=11 farbe=black,shape,red!20
   linie=1,solid r=6,0,0,12,2,3

```

Die Normalengleichung einer Ebene

Die Normalengleichung einer Ebene hat die Form

$$E: (\vec{x} - \vec{a}) \cdot \vec{n} = 0 \quad (8.15)$$

oder

$$E: \vec{x} \cdot \vec{n} - \vec{a} \cdot \vec{n} = 0 \quad (8.16)$$

wobei $\vec{n}$ ein Normalenvektor der Ebene, $\vec{a}$ der Ortsvektor eines beliebigen Punktes ist, der in der

Ebene liegt und $\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$ der Vektor der Unbekannten ist.

Die Syntax ist dabei die Folgende:

ebene

```

[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<ax ∈ ℝ>,<ay ∈ ℝ>,<az ∈ ℝ>
p=<nx ∈ ℝ>,<ny ∈ ℝ>,<nz ∈ ℝ>
[r=<rStart ∈ ℝ>,<rEnde ∈ ℝ>]
[r=<sStart ∈ ℝ>,<sEnde ∈ ℝ>]

```

Wobei n_1 und n_2 standardmäßig auf 1 gesetzt sind. Diese Werte stehen dafür, wie weit eine Ebene gezeichnet werden soll.

In der Abbildung 8.36 ist die Ebene $E : \left[\vec{x} - \begin{pmatrix} 1 \\ 3 \\ 2 \end{pmatrix} \right] \cdot \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} = 0$ dargestellt.

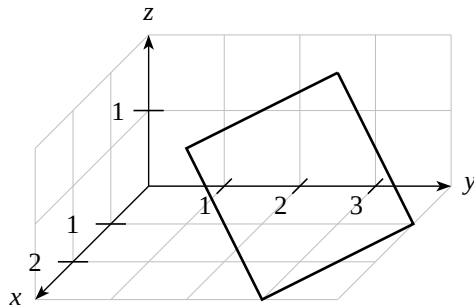


Abb. 8.36.: Eine Ebene in Normalenform

Listing 8.34: Eine Ebene in Normalenform

```
1 einstellung coortype=2 pstricks
2 achsen
3 gitter
4 ebene linie=1,solid p=1,3,2 p=1,2,1
```

8.5.3. Winkel im $\mathbb{R}^3$

Analog zum Abschnitt 4.6.1 gibt es auch im dreidimensionalen Winkelmarkierungen. Die Syntax ist dann:

winkel

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x1 ∈ ℝ>,<y1 ∈ ℝ>,<z1 ∈ ℝ>
p=<x2 ∈ ℝ>,<y2 ∈ ℝ>,<z2 ∈ ℝ>
p=<x3 ∈ ℝ>,<y3 ∈ ℝ>,<z3 ∈ ℝ>
r=<Radius r ∈ ℝ>0 des Bogens>,<ali>
text="<Text>"
```

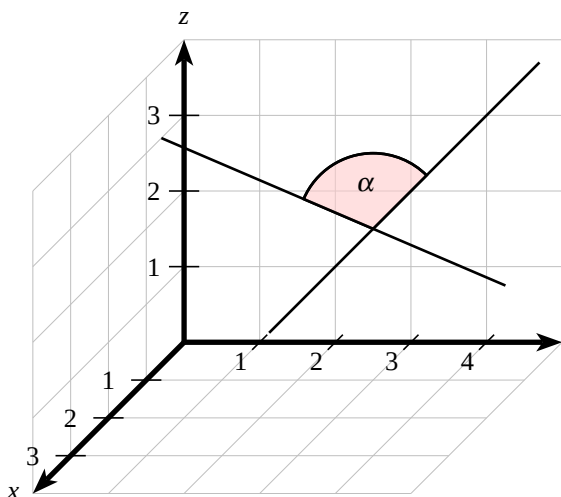


Abb. 8.37.: Ein Winkel im $\mathbb{R}^3$

Listing 8.35: Ein Winkel im $\mathbb{R}^3$

```
1 einstellung coortype=2 pstricks
2 gitter
3 koordinatensystem x=0,4 y=0,5 z=0,4
   einteilung=1,1,1 linie=2,solid,black
   skala=normal text="$x$","$y$","$z$"
4 gerade p=-2,7,6 p=-3,4,4 r=-1.25,-0.6
5 gerade p=1,-4,5 p=0,-7,3 r=-1.25,-0.6
6 winkel farbe=black,shape,red!20 linie=1,solid
   p=-2,7,6 p=1,3,2 p=1,-4,5 r=1,i text="$\alpha$"
```

8.5.4. Kugeln

Eine **Kugel** wird einfach mit dem Befehl `kugel` erstellt, wobei $(x|y|z)$ der Mittelpunkt der Kugel ist.

kugel

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<r ∈ ℝ>0>
```

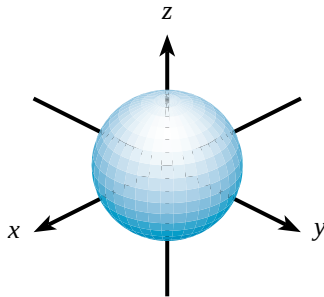


Abb. 8.38.: Eine Kugel

Listing 8.36: Eine Kugel

```
1 einstellung coortype=0 pstricks
2 koordinatensystem x=-2.5,2.5 y=-2.5,2.5 z
  =-2.2 einteilung=1,1,1 skala=keine
  text="$x$","$y$","$z$"
3 kugel farbe=blue,shape,blue!20 linie=1,solid
  parameter="SegmentColor={{cmyk
    ]{0.8,0.2,0.11,0}}}" p=0,0,0 r=1
```

8.6. Körper im $\mathbb{R}^3$

Neben der Kugel gibt es auch noch weitere Körper im dreidimensionalen Raum zu zeichnen. Dazu gehören die folgenden Objekte:

8.6.1. Ein Quader

Ein **Quader** ist definiert über einen Eckpunkt $(x|y|z)$ und seine Länge, Breite und Höhe definiert. Die Syntax ist dabei die folgende:

quader

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<a ∈ ℝ>0>0>0>
```

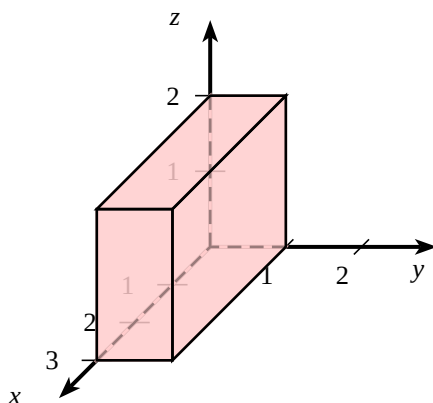


Abb. 8.39.: Ein Quader

Listing 8.37: Ein Quader

```
1 koordinatensystem x=0,4 y=0,3 z=0,3
  einteilung=1,1,1 skala=normal text="
  $x$","$y$","$z$"
2 quader farbe=black,shape,red!20 linie=1,solid
  parameter="" p=0,0,0 r=3,1,2
```

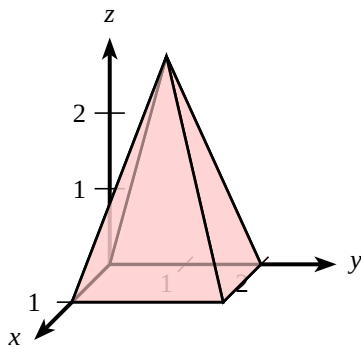
8.6.2. Eine Pyramide und ein Pyramidenstumpf

Eine **Pyramide** ist definiert über einen Eckpunkt $(x|y|z)$ und seine Länge, Breite und Höhe definiert. Die Syntax ist dabei die folgende:

pyramide

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<a ∈ ℝ>0>0>0

```



Listing 8.38: Eine Pyramide

```
1 einstellung coortype=2 pstricks
2 koordinatensystem x=0,2 y=0,3 z=0,3
   einteilung=1,1,1 skala=normal text="
   $x$","$y$","$z$"
3 pyramide farbe=black,shape,red!20 linie=1,
   solid p=1,0,0 r=1,2,3
```

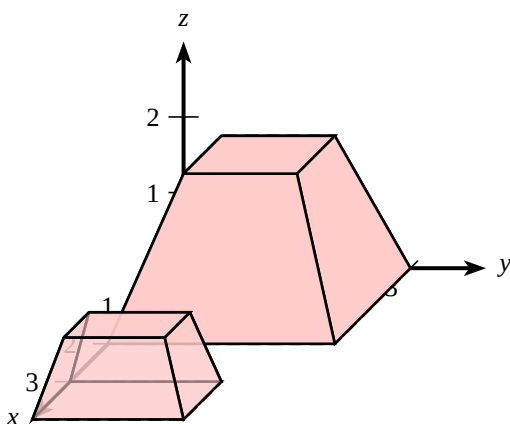
Abb. 8.40.: Eine Pyramide

Zusätzlich zur Pyramide gibt es noch den Befehl des **Pyramidenstumpfes**, dem zusätzlich zur Pyramide noch die abgeschnittene Höhe angegeben wird.

pyramidestumpf

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<a ∈ ℝ>0>0>0abgeschnitten ∈ ℝ>0

```



Listing 8.39: Ein Pyramidenstumpf

```
1 einstellung coortype=2 pstricks
2 koordinatensystem x=0,4 y=0,4 z=0,3
   einteilung=1,1,1 skala=normal text="
   $x$","$y$","$z$"
3 pyramidestumpf farbe=black,solid,red!20
   linie=1,solid p=2,0,0 r=2,3,4,2
4 pyramidestumpf farbe=black,shape,red!20
   linie=1,solid p=4,0,0 r=1,2,3,1
```

Abb. 8.41.: Ein Pyramidenstumpf

8.6.3. Ein Kreiszylinder

Ein **Kreiszylinder** ist definiert über seinen Mittelpunkt $(x|y|z)$, seinen Radius und seine Höhe. Die Syntax ist dabei die folgende:

zylinder

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<r ∈ ℝ>0>0

```

Dabei gibt w an, in welchen Abständen die Seitenlinien gezeichnet werden. Der Standardwert ist dabei 30.

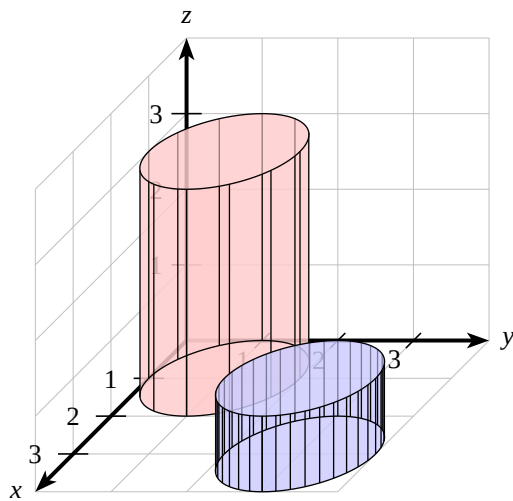


Abb. 8.42.: Ein Kreiszylinder

Listing 8.40: Ein Kreiszylinder

```
1 einstellung pstricks
2 gitter
3 koordinatensystem x=0,4 y=0,4 z=0,4
   einteilung=1,1,1 skala=normal text="
   $x$","$y$","$z$"
4 zylinder farbe=black,shape,red!20 linie=0.5,
   solid p=1,1,0 r=1,3
5 zylinder farbe=black,shape,blue!20 linie=0.5,
   solid p=3,3,0 r=1,1,10
```

8.6.4. Ein Kegel und ein Kegelstumpf

Ein Kegel ist definiert über seinen Mittelpunkt $(x|y|z)$, seinen Radius und seine Höhe. Die Syntax ist die folgende:

kegel

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>[,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<r ∈ ℝ>0>0

```

Dabei gibt w an, in welchen Abständen die Seitenlinien gezeichnet werden. Der Standardwert ist dabei 30.

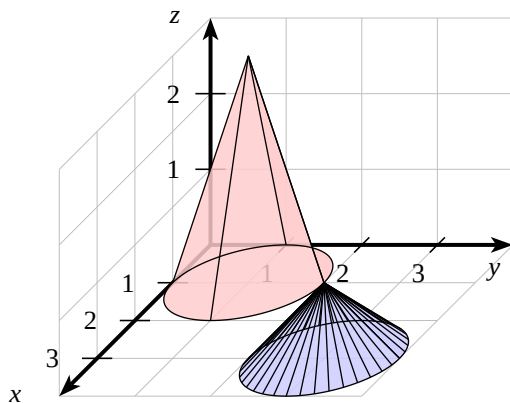


Abb. 8.43.: Ein Kegel

Zusätzlich zum Kegel gibt es noch den Befehl des **Kegelstumpfs**, dem wie beim Kegel der Mittelpunkt, der Radius, die Höhe und die abgeschnittene Höhe übergeben wird.

kegelstumpf

```
[layer=<n ∈ [0;100]>]
[farbe=<Linienfarbe>,<Füllart>,<Füllfarbe>]
[linie=<Linienstärke in pt (l ∈ ℝ>0)>,<Linienart>,<Linienfarbe>]]
[parameter="<Pstricksbefehle>"]
p=<x ∈ ℝ>,<y ∈ ℝ>,<z ∈ ℝ>
r=<r ∈ ℝ>0>0abgeschnitten ∈ ℝ>0

```

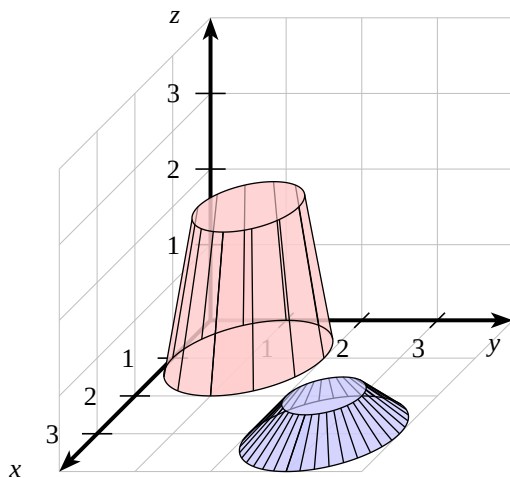


Abb. 8.44.: Ein Kegelstumpf

Listing 8.41: Ein Kegel

```
1 gitter
2 koordinatensystem x=0,4 y=0,4 z=0,3
   einteilung=1,1,1 skala=normal text="
   $x$","$y$","$z$"
3 kegel farbe=black,shape,red!20 linie=0.5,solid
   p=1,1,0 r=1,3
4 kegel farbe=black,shape,blue!20 linie=0.5,
   solid p=3,3,0 r=1,1,10
```

Listing 8.42: Ein Kegelstumpf

```
1 gitter
2 koordinatensystem x=0,4 y=0,4 z=0,4
   einteilung=1,1,1 skala=normal text="
   $x$","$y$","$z$"
3 kegelstumpf farbe=black,shape,red!20 linie
   =0.5,solid p=1,1,0 r=1,3,1
4 kegelstumpf farbe=black,shape,blue!20 linie
   =0.5,solid p=3,3,0 r=1,1,0.5,10
```

9. Spielfeld

In diesem Abschnitt sind die Neuheiten dargestellt, die noch unausgereift sind und deshalb auch noch nicht vollständig dokumentiert sind.

9.1. Uhرداریstellung

Da ich für die fünfte Klasse ab und an Uhren benötigte, habe ich mal angefangen die Funktion `uhr` zu implementieren.

Hier ein paar Beispiele:

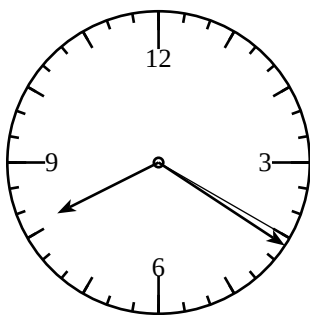


Abb. 9.1.: Eine erste Uhr

Listing 9.1: Eine erste Uhr

```
1 uhr linie=1,solid p=0,0 r=2,0.5  
2 uhrzeit linie=1,solid,black p=0,0 r  
   =2,20,20,20
```

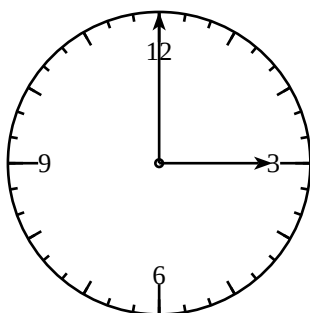


Abb. 9.2.: Eine zweite Uhr

Listing 9.2: Eine zweite Uhr

```
1 uhr linie=1,solid p=0,0 r=2,0.4  
2 uhrzeit linie=1,solid,black p=0,0 r=2,15,0
```

9.2. Regression

Mein Plan ist es, eine vollständige Regressionsanalyse zu implementieren. Bisher habe ich nur die lineare Regression geschafft. Hier ein Beispiel:

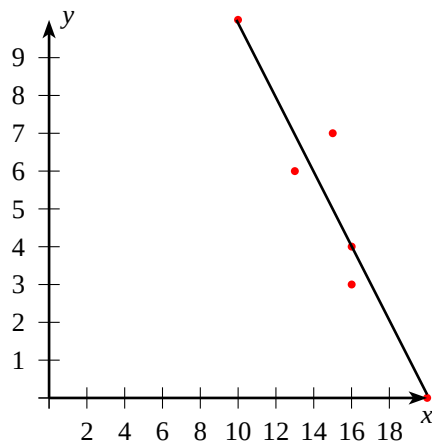


Abb. 9.3.: Die lineare Regression

Listing 9.3: Die lineare Regression

```
1 groesse x=5 y=5
2 achsen
3 listplot farbe=red linie=1,solid
   parameter="showpoints=true,plotstyle=
   dots"
4 20,0
5 16,3
6 15,7
7 16,4
8 13,6
9 10,10
10 ende
11 regression
12 20,0
13 16,3
14 15,7
15 16,4
16 13,6
17 10,10
18 ende
```

Anhang

A. Die Quell-Dateien

Im Folgenden finden sich die gesamten Perl-Dateien. Dabei handelt es sich nicht immer um den saubersten Code, da dies Skript ein lange gewachsenes Projekt ist. Mittlerweile würde ich einiges anders programmieren, aber dazu fehlt mir die Zeit, sodass einiges nur schnelle Lösungen sind und mit Sicherheit auch noch einige Fehler enthält.

A.1. Plotter.pl

Hier die Hauptdatei `plotter.pl` zum Einlesen meiner Dateien.

Listing A.1: `plotter.pl`

```
#!/usr/bin/perl -w
#####
# Funktionsplotter Skript zum erstellen von Grafiken in Pstricks
# Copyright (C) 2008 Xenia Rendtel
#
# Letzte Aenderung: 13.09.2011
#
# Dieses Programm ist freie Software. Sie koennen es unter den Bedingungen der
# GNU General Public License, wie von der Free Software Foundation
# veröffentlicht , weitergeben und/oder modifizieren , entweder gemaess Version 2
# der Lizenz oder (nach Ihrer Option) jeder spaeteren Version .
#
# Die Veroeffentlichung dieses Programms erfolgt in der Hoffnung, dass es Ihnen
# von Nutzen sein wird, aber OHNE IRGENDNEINE GARANTIE, sogar ohne die implizite
# Garantie der MARKTREIFE oder der VERWENDBARKEIT
# ZWECK. Details finden Sie in der GNU General Public License.
#
# Sie sollten ein Exemplar der GNU General Public License zusammen mit diesem
# Programm erhalten haben. Falls nicht , schreiben Sie an die Free Software
# Foundation, Inc. , 51 Franklin St, Fifth Floor, Boston, MA 02110, USA.
#
# Bei Fehlern schreiben Sie diese einfach an
# webmaster@rendtel.de
#####
###Weitere Pakete
use Math::Trig;
use strict ;
use warnings;
use FindBin;

use lib "$FindBin::Bin";
use lib '/home/xenia/Schule/Programme/Plotter';
use funktionen;
use funktionsparser;

#use Image::Magick; #ImageMagick muss hierfuer installiert werden
use Image::Size;
use Image::ExifTool;

use bildbearbeitung;
use dateierzeugen;
use gitter ;
```

use achsen;				43
use graphen;				44
use uhr;				45
use knoten;				46
use regression;				47
				48
<i>#use baum; # Muss noch bearbeitet werden</i>				49
				50
use zeichnen;				51
use paar;				52
use diagramme;				53
use dreidimensional;				54
				55
my (				56
\$hoeheersatz,	\$kommandodatei,	\$texdatei,	\$i,	57
\$j,	\$zeile,	\$farbverlauf,	\$farbverlauf2,	58
\$zeilei ,	\$zeilej ,	\$layer,	\$linienstaerke ,	59
\$linienart ,	\$farbe,	\$fuellart ,	\$fuellfarbe ,	60
\$text ,	\$radius,	\$winkel,	\$linienstaerkeersatz ,	61
\$linienartersatz ,	\$farbeersatz ,	\$fuellartersatz ,	\$fuellfarbeersatz ,	62
\$textersatz ,	\$radiusersatz ,	\$winkelersatz ,	\$breite ,	63
\$hoehe,	\$skala,	\$func,	\$nullwert,	64
\$plotart ,	\$aufsummiert,	\$markierung,	\$abstandx,	65
\$dickx,	\$ganzdickx,	\$abstandy,	\$dicky,	66
\$ganzdicky,	\$xmin,	\$xmax,	\$dx,	67
\$ymin,	\$ymax,	\$dy,	\$abstand,	68
\$xlaenge,	\$xschritt ,	\$xskala,	\$xtext ,	69
\$x,	\$yschritt ,	\$ylaenge,	\$yskala,	70
\$ytext ,	\$y,	\$ersatzvektor,	\$pfeil ,	71
\$breitetechnik,	\$bild ,	\$nwert,	\$pwert,	72
\$skalierungspunkt,	\$punktzeichnen,	\$punktart,	\$enderreicht,	73
\$hilfszeile ,	\$inzeile ,	\$absolutwert,	\$merkei,	74
\$winkelstart,	\$laenge,	\$datei,	\$schachtelung,	75
\$summen,	\$func1,	\$func2,	\$func3,	76
\$schnittpunkte,	\$linie ,	\$x1,	\$x2,	77
\$y1,	\$y2,	\$verbunden,	\$ausrichtung,	78
\$dxersatz ,	\$abstandersatz ,	\$textwinkersatz ,	\$textausrichtungersatz ,	79
\$wert,	\$dyersatz ,	\$l1 ,	\$w1,	80
\$l2 ,	\$w2,	\$l3 ,	\$w3,	81
\$l4 ,	\$pfeillaenge ,	\$w4,	\$l5 ,	82
\$w5,	\$l6 ,	\$w6,	\$radius1,	83
\$radius2,	\$winkel1,	\$winkel2,	\$pfeilart ,	84
\$leer ,	\$eingabedateizeilen,	@eingabedatei,	@ausgabedatei,	85
@ausgabedateizeilen,	\$kommandodateinummer,	\$startwinkel,	\$x3,	86
\$y3,	\$dicke,	\$dickeersatz ,	\$polygonzeichnen,	87
\$anfang,	\$ende,	\$winkelrichtung,	\$z,	88
\$z1,	\$z2,	\$z3,	\$alpha,	89
\$beta,	\$coor,	\$pstricks ,	\$abstandz,	90
\$dickz,	\$ganzdickz,	\$zmin,	\$zmax,	91
\$ztext ,	\$zschritt ,	\$parameter,	\$tmin,	92
\$tmax,	\$umin,	\$umax,	\$parameterersatz ,	93
\$x4,	\$y4,	\$z4,	\$r1 ,	94
\$r2,	\$r3,	\$c,	\$d,	95
\$a,	\$b,	\$r4,	\$n,	96
\$r,	\$abgeschnitten,	\$r5,	\$offset ,	97
\$r6,	\$xbruch,	\$rad,	\$bezeichnung,	98
\$variable,	\$schritt ,	\$spalten,	\$kopf,	99
\$nachkommastellen,	\$textwinkel,	\$geschlossen,	\$textausrichtung,	100
\$name,	\$anzahl,	\$breiteersatz ,	\$abstandxersatz ,	101
\$abstandyersatz ,	\$grad,	\$absolut,	\$datenlesenstatus,	102
\$einlesen,	\$n1,	\$n2,	\$stunde,	103

```

$minute,      $sekunde,      $art,      $groesse,
$name1,        $name2,        $parameter1, $parameter2
);
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164

```

my \$pstrickersatz3d;
my \$pstrickersatz;

\$absolut = 1;
\$datenlesenstatus = \$einlesen = 0;

Arrays und Layer
\$eingabedateizeilen = 0;
@ausgabedatei = ();
@ausgabedateizeilen = ();
\$kommandodateinummer = 0;
for (\$i = 0 ; \$i <= 150 ; \$i++) { \$ausgabedateizeilen[\$i] = 0; }
setskalierung(1, "x");
setskalierung(1, "y");
setschnitt(0, "x");
setschnitt(0, "y");
setskala("normal", "x");
setskala("normal", "y");
setfarbe("black");
setfuellart ("");
setpfeilart ("");
setfuellfarbe ("black!20");
setlinienstaerke (1);
setlinienart ("solid");
setmarkierung("\\\$x\\\$", "x");
setmarkierung("\\\$y\\\$", "y");
setmarkierung("\\\$z\\\$", "z");
setachsenzeichnen(0, "x");
setachsenzeichnen(0, "y");
\$datei = 0;
setextern(0);
\$farbverlauf = 0;

Die Eingabedatei wird gelesen

sub standard {
 my (\$layer1, \$linienart1, \$linienstaerke1, \$farbe1, \$fuellart1, \$fuellfarbe1, \$pfeilart1,
 \$parameter1)
 = @_;
 \$layer = \$layer1;
 \$linienart = \$linienart1;
 \$linienstaerke = \$linienstaerke1;
 \$farbe = \$farbe1;
 \$fuellart = \$fuellart1;
 \$fuellfarbe = \$fuellfarbe1;
 \$pfeilart = \$pfeilart1;
 \$parameter = \$parameter1;
}

sub leseingabedatei {
 my \$dateiname = **shift**(@_);
 my \$zeile;
 open(**my** \$EINGABEDATEI, \$dateiname) || **die** \$dateiname . ": \$!";
 while (\$zeile = <\$EINGABEDATEI>) {

```

# Zeilenenden beseitigen , Kommentare und Leerzeilen ignorieren
$zeile =~ s/[\r\n]//g;
$zeile =~ s/ +/ /g;
$zeile =~ s/ +$/ /g;
$zeile =~ s/^ +//g;
next if ( $zeile =~ /^#\// );
next if ( $zeile =~ /^$/ );
if ( $zeile =~ /^lesedatei (.*)$/i ) { leseeingabedatei($1); }
elsif ( $zeile =~ /^fileplot (.*)$/i ) {
    $einlesen = 1;
    $eingabedatei[$eingabedateizeilen] = $zeile;
    $eingabedateizeilen++;
}
elsif ( $zeile =~ /^listplot (.*)$/i ) {
    $einlesen = 1;
    $eingabedatei[$eingabedateizeilen] = $zeile;
    $eingabedateizeilen++;
}

elsif ( $zeile =~ /^regression (.*)$/i ) {
    $einlesen = 1;
    $eingabedatei[$eingabedateizeilen] = $zeile;
    $eingabedateizeilen++;
}

elsif ( $zeile =~ /^lese (.*)$/i ) {
    if ( $einlesen == 0 ) { leseeingabedatei($1); }
    else {
        $eingabedatei[$eingabedateizeilen] = $zeile;
        $eingabedateizeilen++;
    }
}
elsif ( $zeile =~ /^ende (.*)$/i ) {
    $einlesen = 0;
    $eingabedatei[$eingabedateizeilen] = $zeile;
    $eingabedateizeilen++;
}

else {

    #Einzelne Zeilen in ein Array speichern
    $eingabedatei[$eingabedateizeilen] = $zeile;
    $eingabedateizeilen++;
}
}
close($EINGABEDATEI);
}

# In ein array wird geschrieben
sub schreibeinarray {
    my ( $layer, $zeile ) = @_ ;
    $zeile =~ s/, ]/]/;
    $zeile =~ s/, ]/]/;
    $zeile =~ s/, ]/]/;
    $zeile =~ s/ / / /;
    $zeile =~ s/ / / /;
    $ausgabedatei[$layer][$ausgabedateizeilen[$layer] ] = $zeile;
    $ausgabedateizeilen[$layer]++;
}

```

```

### Funktionen fuer das Plotterskript
sub kopfdatei {
    kommentar( 101, "Autor: X. Rendtel" );
    kommentar( 101, "Letzte Aenderung: 2011 \n" );

    if ( getkopf() == 1 ) {
        schreibeinarray( 103, sprintf("\\documentclass{ttctexa}") );
        schreibeinarray( 103, sprintf("\\setlength\\parindent{0pt}") );
        schreibeinarray( 103, sprintf("\\StartShownPreambleCommands") );
        schreibeinarray( 105, sprintf("\\StopShownPreambleCommands") );
        usepackage( "",
            "pstricks,pst-pdf,pst-node,xcolor,pst-circ,pst-func,pst-math,pstricks-add" );
        usepackage( "utf8x", "inputenc" );
        usepackage( "LGR,T1", "fontenc" );

        pstricks( 110, "\\pagestyle{empty} \n \\begin{document} " );
    }
    my $parameterweiter =
        "dotstyle=*, plotpoints=1000, dotsize=3pt, arrowsize=3pt 2, arrowinset=0.25,";

    if ( get3dpstricks() == 1 ) {
        pstricks( 109, "\\makeatletter" );
        pstricks( 109, "\\def\\psxyzlabel\\#1{\\small\\expandafter\\@stripDecimals\\#1..\\@
            @nil}" );
        pstricks( 109, "\\makeatother" );

        usepackage( "", "pst-gr3d,pst-3dplot,pst-ob3d" );

        $parameterweiter =
            $parameterweiter
            . "Alpha="
            . get3dwinkel("alpha")
            . ", Beta="
            . get3dwinkel("beta")
            . ", coordType="
            . get3dkoor()
            . ", xThreeDunit="
            . getskalierung("x")
            . ", yThreeDunit="
            . getskalierung("y")
            . ", zThreeDunit="
            . getskalierung("z");
    }
    else {
        $parameterweiter =
            $parameterweiter
            . " xunit="
            . getskalierung("x")
            . "cm, yunit="
            . getskalierung("y") . "cm";
    }

    einstellungenkopf( 110, getfarbe(), getfuellart(), getfuellfarbe(), getlinienart(),
        getlinienstaerke(), $parameterweiter );
    beginpicture(111);
}

sub fussdatei {

```

```

endpicture(100);
if ( getkopf() == 1 ) { pstricks( 100, "\\end{document}"); }
}

### Einlesen der Datei
while ( exists $ARGV[$kommandodateinummer] ) {
    $kommandodatei = $ARGV[$kommandodateinummer];
    $texdatei      = $kommandodatei;
    $texdatei =~ s/\.ptxt$/ /i;
    $texdatei = $texdatei . " . xtex ";
    $kommandodateinummer++;
    $eingabedateizeilen = 0;
    leseeingabedatei($kommandodatei);
    for ( $j = 0 ; $j < $eingabedateizeilen ; $j++ ) {

        for ( $i = $j ; $i <= $j ; $i++ ) {
            $zeile = $eingabedatei[$i];
            if ( $zeile =~ /^gitter$/i ) { setgitter(1); }
            elsif ( $zeile =~ /^matheheft$/i ) { setmatheheft(1); }
            elsif ( $zeile =~ /^knoten$/i ) { setknoten(1); }
            elsif ( $zeile =~ /^farbverlauf$/i ) { $farbverlauf = 1; }
            elsif ( $zeile =~ /^text$/i ) { setbeschriftung(1); }
            elsif ( $zeile =~ /^punkt$/i ) { setpunkt(1); }
            elsif ( $zeile =~ /^relativ$/i ) { setrelativ(); }
            elsif ( $zeile =~ /^mitte$/i ) { setmitte(1); }
            elsif ( $zeile =~ /^hypothenuse$/i ) { sethyp(1); }
            elsif ( $zeile =~ /^ohnekopf$/i ) { setkopf(0); }
            elsif ( $zeile =~ /^schriftgroesse ([0-9]+)$/i ) {
                setschriftgroesse ($1);
            }
            elsif ( $zeile =~ /^beschriftunggroesse \"([^\"]*)\"$/i ) {
                setbeschriftunggroesse($1);
            }
            elsif ( $zeile =~ /^ticks ([0-9]+)$/i ) { setsubticks($1); }
            elsif ( $zeile =~
/^grenzen x=([-0-9\\.]+),([-0-9\\.]+) y=([-0-9\\.]+),([-0-9\\.]+) z=([-0-9\\.]+),([-0-9\\.]+)
)?$/i
)
            {
                minmax( $1, $3, $2, $4 );
                setachsengraph( $1, $3, $2, $4 );
                if ( defined $5 ) {
                    minmax3d( $1 - 1, $3 - 1, $6 - 1, $2 + 1, $4 + 1, $7 + 1 );
                }
            }
            elsif ( $zeile =~ /^achsen$/i ) {
                setachsen(1);
                $hilfszeile = $eingabedatei[$i];
                $enderreicht = 0;
                while ( ( !$enderreicht )
                    && ( $i < $eingabedateizeilen - 1 ) )
                {
                    $i++;
                    $hilfszeile = $eingabedatei[$i];
                    if ( $hilfszeile =~ /^gitter$/i ) {
                        $enderreicht = 1;
                        setgitter(1);
                    }
                    else { $enderreicht = 1; }
                }
            }
        }
    }
}

```

```

    }
  }
}

for ( $i = 0 ; $i < $eingabedateizeilen ; $i++ ) {
  $zeile = $eingabedatei[$i];

  # Befehl " skalierung "
  if ( $zeile =~ /^skalierung ([0-9\.]*)$/i ) {
    skalierungsetzen( $1, "x" );
    skalierungsetzen( $1, "y" );
    skalierungsetzen( $1, "z" );
  }

  elseif ( $zeile =~ /^skalierung( x=([0-9\.]*)?( y=([0-9\.]*)?( z=([0-9\.]*)?)?$/i ) {
    skalierungsetzen( $2, "x" ) if ( defined $1 );
    skalierungsetzen( $4, "y" ) if ( defined $3 );
    skalierungsetzen( $6, "z" ) if ( defined $5 );
  }

  elseif ( $zeile =~ /^ticks ([0-9]+)$/i ) { setsubticks( $1 + 1 ); }
  elseif ( $zeile =~ /^pfeile ([^"]*)$/i ) { setpfeilart( $1 ); }
  elseif (
    $zeile =~ /^achsenmarkierung( x=\"([^\"]*)\" )?( y=\"([^\"]*)\" )?( z=\"([^\"]*)\" )?$/i )
  {
    setmarkierung( $2, "x" ) if ( defined $1 );
    setmarkierung( $4, "y" ) if ( defined $3 );
    setmarkierung( $6, "z" ) if ( defined $5 );
  }

  elseif ( $zeile =~ /^gitter$/i ) { }
  elseif ( $zeile =~ /^achsen$/i ) { }
  elseif ( $zeile =~ /^knoten$/i ) { }
  elseif ( $zeile =~ /^farbverlauf$/i ) { }
  elseif ( $zeile =~ /^text$/i ) { }
  elseif ( $zeile =~ /^punkt$/i ) { }
  elseif ( $zeile =~ /^relativ$/i ) { }
  elseif ( $zeile =~ /^mitte$/i ) { }
  elseif ( $zeile =~ /^ohnekopf$/i ) { }
  elseif ( $zeile =~ /^schriftgroesse ([0-9]+)$/i ) { }
  elseif ( $zeile =~ /^beschriftungsgroesse \"([^\"]*)\"$/i ) { }
  elseif ( $zeile =~
    /\^grenzen x=([-0-9\.]*) ,([-0-9\.]*) y=([-0-9\.]*) ,([-0-9\.]*) ( z=([-0-9\.]*) ,([-0-9\.]*)
    )?$/i
  )
  {
    minmax( $1, $3, $2, $4 );
    if ( defined $5 ) {
      minmax3d( $1 - 1, $3 - 1, $6 - 1, $2 + 1, $4 + 1, $7 + 1 );
    }
  }
}

# Achsen fuer Diagramme setzen

elseif ( $zeile =~ /^neuesbild( layer=([0-9]+) )?$/i ) {
  standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
  $layer = $2 if ( defined $1 );
  endpicture($layer);
  abstand($layer);
  abstand($layer);
  pstricks( $layer, "\\newpage" );
}

```

abstand(\$layer);	406
beginpicture(\$layer);	407
setgitterbild (\$layer);	408
setachsenbild(\$layer);	409
	410
}	411
	412
elseif (\$zeile =~ /^groesse(x=([0-9\.]+))?(y=([0-9\.]+))?(z=([0-9\.]+))?\$ /i) {	413
setgross(1);	414
setgroesse(\$2, "x") if (defined \$1);	415
setgroesse(\$4, "y") if (defined \$3);	416
setgroesse(\$6, "z") if (defined \$5);	417
}	418
	419
# Grenzen setzen	420
	421
elseif (\$zeile =~	422
/^grenzgitter x=([0-9\.]+),([0-9\.]+) y=([0-9\.]+),([0-9\.]+)(z=([0-9\.]+)	423
,([0-9\.]+))?\$ /i	
)	424
{	425
setgitter (1);	426
setachsen(1);	427
if (defined \$5) {	428
minmax3d(\$1, \$3, \$6, \$2, \$4, \$7);	429
setachsengraph3d(\$1, \$3, \$6, \$2, \$4, \$7);	430
}	431
else {	432
minmax(\$1, \$3, \$2, \$4);	433
setachsengraph(\$1, \$3, \$2, \$4);	434
}	435
}	436
	437
# Befehl "Achsen Schnittpunkte"	438
elseif (\$zeile =~ /^achsenschnitt(x=([0-9\.]+))?(y=([0-9\.]+))?(z=([0-9\.]+))?\$ /i	439
) {	
schnittsetzen(\$2, "x") if (defined \$1);	440
schnittsetzen(\$4, "y") if (defined \$3);	441
schnittsetzen(\$5, "z") if (defined \$4);	442
}	443
elseif (\$zeile =~	444
/^einstellung(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	445
=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?\$ /i	
)	446
{	447
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	448
"");	449
\$layer = \$2 if (defined \$1);	450
if (defined \$3) {	451
\$farbe = \$4;	452
\$fuellart = \$5;	453
\$fuellfarbe = \$6;	454
}	455
if (defined \$7) {	456
\$linienstaerke = \$8;	457
\$linienart = \$9;	458
}	459
\$farbe = \$11 if (defined \$10);	460
setfarbe(\$farbe);	461
setfuellart (\$fuellart);	462
setfuellfarbe (\$fuellfarbe);	463

```

setlinienstaerke ($linienstaerke);
setlinienart ($linienart);

}
elseif ( $zeile =~
/^einstellung( layer=([0-9]+))?( alpha=([-0-9\\.]+))?( beta=([-0-9\\.]+))?( coortype
=([-0-9\\.]+))?( pstricks)?$/i
)
{
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
$layer = $2 if ( defined $1 );
$alpha = 45;
$alpha = $4 if ( defined $3 );
$beta = 30;
$beta = $6 if ( defined $5 );
$coor = 2;
$coor = $8 if ( defined $7 );
set3dwinkel( $alpha, "alpha" );
set3dwinkel( $beta, "beta" );
set3dkoor($coor);
$pstricks = 0;
$pstricks = 1 if ( defined $9 );
set3dpstricks($pstricks);
}

# Befehl "kariert"
elseif ( $zeile =~
/^kariert( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+))?( linie=([0-9\\.]+),([a-zA-Z0-9!]+))?(,[a-zA-Z0-9!]+))?(
dx=([0-9\\.]+),([0-9\\.]+),([0-9\\.]+) dy=([0-9\\.]+),([0-9\\.]+),([0-9\\.]+)
(dz=([0-9\\.]+),([0-9\\.]+),([0-9\\.]+))?( x|p)=([-0-9\\.]+),([-0-9\\.]+) [y|p]=([-0-9\\.]+)
,([-0-9\\.]+)( z=([-0-9\\.]+),([-0-9\\.]+)))?$/i
)
{
standard( 1, getlinienart (), 0.2, "lightgray", "", getfuellfarbe (), "", "" );
$layer = $2 if ( defined $1 );
$farbe = $4 if ( defined $3 );
if ( defined $5 ) {
$linienstaerke = $6;
$linienart = $7;
}
$farbe = $9 if ( defined $8 );
$abstandx = $10;
$dickx = $11;
$ganzdickx = $12;
$abstandy = $13;
$dicky = $14;
$ganzdicky = $15;

if ( defined $16 ) {
$abstandz = $17;
$dickz = $18;
$ganzdickz = $19;
}
$xmin = $20;
$xmax = $21;
$ymin = $22;
$ymax = $23;
if ( defined $24 ) {
$zmin = $25;
$zmax = $26;
}

```

}	521
setachsengraph(\$xmin, \$ymin, \$xmax, \$ymax);	522
	523
	524
if ((defined \$16) && (defined \$24)) {	525
set3dzeichnen(1);	526
kariert3d(	527
\$layer, \$farbe, \$linienstaerke, \$linienart, \$xmin,	528
\$ymin, \$zmin, \$xmax, \$ymax, \$zmax,	529
\$abstandx, \$dickx, \$ganzdickx, \$abstandy, \$dicky,	530
\$ganzdicky, \$abstandz, \$dickz, \$ganzdickz	531
);	532
}	533
else {	534
kariert(	535
\$layer, \$farbe, \$linienstaerke, \$linienart, \$xmin,	536
\$ymin, \$xmax, \$ymax, \$abstandx, \$dickx,	537
\$ganzdickx, \$abstandy, \$dicky, \$ganzdicky	538
);	539
}	540
next;	541
}	542
	543
	544
# Befehl "punkte"	545
elseif (\$zeile =~	546
/^punkte(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))? abstand=([0-9\.]+)? [x p]	547
=([-0-9\.]+),([-0-9\.]+) [y p]=([-0-9\.]+),([-0-9\.]+)\$/i	
)	548
{	549
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	550
"");	551
\$layer = \$2 if (defined \$1);	552
\$farbe = \$4 if (defined \$3);	553
\$abstand = \$5;	554
\$xmin = \$6;	555
\$xmax = \$7;	556
\$ymin = \$8;	557
\$ymax = \$9;	558
setachsengraph(\$xmin, \$ymin, \$xmax, \$ymax);	559
grid(\$layer, \$farbe, getlinienstaerke (), getlinienart (), \$xmin, \$ymin, \$xmax, \$ymax,	560
\$abstand, \$abstand, \$abstand, \$abstand, \$abstand, \$abstand);	561
	562
}	563
	564
elseif (\$zeile =~	565
/^grid(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.]+),([a-zA-Z0-9!]+))?([a-zA-Z0-9!]+)? dx=([0-9\.]+),([0-9\.]+) dy=([0-9\.]+),([0-9\.]+) [x p]=([-0-9\.]+)	566
,([-0-9\.]+) [y p]=([-0-9\.]+),([-0-9\.]+)\$/i	
)	567
{	568
standard(1, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	569
"");	570
\$layer = \$2 if (defined \$1);	571
\$farbe = \$4 if (defined \$3);	572
if (defined \$5) {	573
\$linienstaerke = \$6;	574
\$linienart = \$7;	575
}	576
\$farbe = \$9 if (defined \$8);	577
\$abstandx = \$10;	578

```

$dickx   = $11;
$abstandy = $12;
$dicky   = $13;
$xmin    = $14;
$xmax    = $15;
$ymin    = $16;
$ymax    = $17;
setachsengraph( $xmin, $ymin, $xmax, $ymax );
grid( $layer, $farbe, $linienstaerke, $linienart, $xmin, $ymin, $xmax, $ymax,
      $abstandx,
      $dickx, 0, $abstandy, $dicky, 0 );
}

elseif ( $zeile =~
/^grid( layer=([0-9]+) )?( farbe=([a-zaeoeue0-9!]+) )?( linie=([0-9\.] +), ([a-z0-9!]+) )?( ([a-zaeoeue0-9!]+) )? dx=([0-9\.] +), ([0-9\.] +), ([0-9\.] +) dy=([0-9\.] +), ([0-9\.] +), ([0-9\.] +)
[x|p]=([0-9\.] +), ([0-9\.] +) [y|p]=([0-9\.] +), ([0-9\.] +) )$/i
)
{
  standard( 1, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
  $layer = $2 if ( defined $1 );
  $farbe = $4 if ( defined $3 );
  if ( defined $5 ) {
    $linienstaerke = $6;
    $linienart     = $7;
  }
  $farbe = $9 if ( defined $8 );
  $abstandx = $10;
  $dickx    = $11;
  $ganzdickx = $12;
  $abstandy = $13;
  $dicky    = $14;
  $ganzdicky = $15;
  $xmin     = $16;
  $xmax     = $17;
  $ymin     = $18;
  $ymax     = $19;
  setachsengraph( $xmin, $ymin, $xmax, $ymax );
  grid(
    $layer, $farbe, $linienstaerke, $linienart, $xmin,
    $ymin, $xmax, $ymax, $abstandx, $dickx,
    $ganzdickx, $abstandy, $dicky, $ganzdicky
  );
}

# Befehl "millimeterpapier"
elseif ( $zeile =~
/^millimeterpapier( layer=([0-9]+) )?( farbe=([a-zaeoeue0-9!]+) )? [x|p]=([0-9\.] +)
, ([0-9\.] +) [y|p]=([0-9\.] +), ([0-9\.] +) )$/i
)
{
  standard( 10, getlinienart(), getlinienstaerke(), "lightgray", "", getfuellfarbe(), "",
    "" );
  $layer = $2 if ( defined $1 );
  $farbe = $4 if ( defined $3 );
  $xmin = $5;
  $xmax = $6;
  $ymin = $7;
  $ymax = $8;

```

setachsengraph(\$xmin, \$ymin, \$xmax, \$ymax);	636
millimeterpapier(\$layer, \$farbe, \$xmin, \$ymin, \$xmax, \$ymax);	637
	638
}	639
	640
# Befehl "matheheft"	641
elseif (\$zeile =~	642
/^matheheft(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+))? [x p]=([0-9\..]+),([0-9\..]+) [y	643
p]=([0-9\..]+),([0-9\..]+)\$ /i	
)	644
{	645
standard(10, getlinienart (), getlinienstaerke (), "lightgray", "", getfuellfarbe (), "",	646
" ");	647
\$layer = \$2 if (defined \$1);	648
\$farbe = \$4 if (defined \$3);	649
\$xmin = \$5;	650
\$xmax = \$6;	651
\$ymin = \$7;	652
\$ymax = \$8;	653
setachsengraph(\$xmin, \$ymin, \$xmax, \$ymax);	654
matheheft(\$layer, \$farbe, \$xmin, \$ymin, \$xmax, \$ymax);	655
	656
}	657
	658
# Befehl fuer das Koordinatensystem	659
elseif (\$zeile =~	660
/^achse(layer=([0-9]+))? [x p]=([0-9\..]+),([0-9\..]+),([0-9\..]+)(linie=([0-9\..]+),([a-	661
z0-9!]+))?(,[a-zaeoue0-9!]+))? skala=(keine nurstriche nureins normal strichundeins)	
text="\^[^"]*)" \$ /i	662
)	663
{	664
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "->",	665
" ");	666
	667
\$layer = \$2 if (defined \$1);	668
\$xmin = \$3;	669
\$xmax = \$4;	670
\$xschritt = \$5;	671
if (defined \$6) {	672
\$linienstaerke = \$7;	673
\$linienart = \$8;	674
}	675
\$farbe = \$10 if (defined \$9);	676
\$xskala = \$11;	677
\$xtext = \$12;	678
setachszeichnen(1, "x");	679
setachszeichnen(0, "y");	680
setminmaxachsen(\$xmin, "x", "min");	681
	682
setminmaxachsen(\$xmax, "x", "max");	683
setminmaxachsen(0, "y", "min");	684
setminmaxachsen(0, "y", "max");	685
setachsengraph(\$xmin, 0, \$xmax, 0);	686
setschritt (\$xschritt, "x");	687
schnittsetzen(\$xmin, "x");	688
setschritt (0, "y");	689
setskala(\$xskala, "x");	690
settext(\$xtext, "x");	691
	692
for (\$j = 0 ; \$j < \$eingabedateizeilen ; \$j++) {	693
\$zeilej = \$eingabedatei[\$j];	

```

        if ( $zeilej =~
/^achse( layer=([0-9]+) )? [x|p]=([0-9\..]+),([0-9\..]+),([0-9\..]+)( linie=([0-9\..]+),([a-
z0-9!]+) )? ,([a-zaeooue0-9!]+) )? skala=(keine | nurstriche | nureins | normal | strichundeins)
text="\([^\"]*\)\"$/i
        )
    {
        standard( 10, getlinienart (), getlinienstaerke (), getfarbe (), "",
            getfuellfarbe (), "->", "" );

        $layer    = $2 if ( defined $1 );
        $ymin      = $3;
        $ymax      = $4;
        $yschritt  = $5;

        if ( defined $6 ) {
            $linienstaerke = $7;
            $linienart      = $8;
        }
        $farbe = $10 if ( defined $9 );
        $yskala = $11;
        $ytext  = $12;
        setachsenzeichnen( 1, "y" );
        setminmaxachsen( $ymin, "y", "min" );
        setminmaxachsen( $ymax, "y", "max" );
        setschritt ( $yschritt, "y" );
        setskala( $yskala, "y" );
        schnittsetzen( $ymin, "y" );
        settext( $ytext, "y" );
        setachsengraph( 0, $ymin, 0, $ymax );

        achsen(
            $layer, $farbe, $linienstaerke, $linienart, $pfeilart,
            $xmin, $xmax, $ymin, $ymax, $xschritt,
            $yskala, $xtext, $yschritt, $yskala, $ytext
        );
        next;
    }
}
if ( achsengezeichnet("y") == 0 ) {
    setpfeilart ( $pfeilart );
    xachse(
        $layer, $farbe, $linienstaerke, $linienart, $xmin,
        $xmax, $xschritt, $yskala, $xtext
    );
}

}

# Befehl fuer das Koordinatensystem mit Pi
elseif ( $zeile =~
/^achserad( layer=([0-9]+) )? [x|p]=([0-9\..]+),([0-9\..]+),([0-9\..]+)( linie=([0-9\..]+),([a-
z0-9!]+) )? ,([a-zaeooue0-9!]+) )? ( text="\([^\"]*\)\" )?$/i
    )
{
    standard( 10, getlinienart (), 0.8, getfarbe (), "", getfuellfarbe (), "->", "" );
    $layer    = $2 if ( defined $1 );
    $xmin      = degtorad($3);
    $xmax      = degtorad($4);
    $xschritt  = degtorad($5);
    if ( defined $6 ) {
        $linienstaerke = $7;
    }
}

```

\$linienart = \$8;	752
}	753
\$farbe = \$10 if (defined \$9);	754
\$xtext = "";	755
\$xtext = \$12 if (defined \$11);	756
setachsenzeichnen(1, "x");	757
setachsenzeichnen(0, "y");	758
setminmaxachsen(\$xmin, "x", "min");	759
setminmaxachsen(\$xmax, "x", "max");	760
setminmaxachsen(0, "y", "min");	761
setminmaxachsen(0, "y", "max");	762
setachsengraph(\$xmin, 0, \$xmax, 0);	763
setschritt (\$xschritt, "x");	764
setskala(\$xskala, "x");	765
settext(\$xtext, "x");	766
	767
for (\$j = 0 ; \$j < \$eingabedateizeilen ; \$j++) {	768
\$zeilej = \$eingabedatei[\$j];	769
if (\$zeilej =~	770
/^yachse(layer=([0-9]+)?)? [y p]=([0-9\.\.]+),([0-9\.\.]+),([0-9\.\.]+)(linie=([0-9\.\.]+),([a-	771
z0-9!]+)?)?([a-zaeoeue0-9!]+)?)? skala=(keine nurstriche nureins normal strichundeins)	
text="\([^"]*\)"\$/i	
)	772
{	773
standard(10, getlinienart (), 0.8, getfarbe(), "", getfuellfarbe (), "->", "");	774
	775
\$layer = \$2 if (defined \$1);	776
\$ymin = \$3;	777
\$ymax = \$4;	778
\$yschritt = \$5;	779
if (defined \$6) {	780
\$linienstaerke = \$7;	781
\$linienart = \$8;	782
}	783
	784
\$farbe = \$10 if (defined \$9);	785
\$yskala = \$11;	786
\$ytext = \$12;	787
setachsenzeichnen(1, "y");	788
setminmaxachsen(\$ymin, "y", "min");	789
setminmaxachsen(\$ymax, "y", "max");	790
setschritt (\$yschritt, "y");	791
setskala(\$yskala, "y");	792
settext(\$ytext, "y");	793
setachsengraph(0, \$ymin, 0, \$ymax);	794
setpfeilart (\$pfeilart);	795
	796
xyachserad(	797
\$layer, \$farbe, \$linienstaerke, \$linienart, \$xmin,	798
\$xmax, \$ymin, \$ymax, \$xschritt, \$xtext,	799
\$yschritt, \$yskala, \$ytext	800
);	801
	802
}	803
}	804
if (achsengezeichnet("y") == 0) {	805
setpfeilart (\$pfeilart);	806
xachserad(\$layer, \$farbe, \$linienstaerke, \$linienart, \$xmin, \$xmax, \$xschritt,	807
\$xtext);	808
}	809
	810

```

    }
    elseif ( $zeile =~
/^yachse( layer=([0-9]+))?[ y | p]=([0-9\.]+),([0-9\.]+),([0-9\.]+)( linie=([0-9\.]+),([a-
z0-9!]+))?(,[a-zaeoeue0-9!]+)? skala=(keine | nurstriche | nureins | normal | strichundeins)
text="\([^\"]*\)"\$/i
    )
    {
        standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "->",
        "" );
        $layer    = $2 if ( defined $1 );
        $ymin     = $3;
        $ymax     = $4;
        $yschritt = $5;
        if ( defined $6 ) {
            $linienstaerke = $7;
            $linienart     = $8;
        }

        $farbe = $10 if ( defined $9 );
        $yskala = $11;
        $ytext  = $12;
        setachszeichnen( 1, "y" );

        setminmaxachsen( $ymin, "y", "min" );
        setminmaxachsen( $ymax, "y", "max" );
        setminmaxachsen( 0,    "x", "min" );
        setminmaxachsen( 0,    "x", "max" );
        setschritt ( $yschritt, "y" );
        setskala ( $yskala, "y" );
        settext ( $ytext, "y" );
        setachsengraph( 0, $ymin, 0, $ymax );

        if ( achsengezeichnet("x") == 0 ) {
            setpfeilart ( $pfeilart );
            yachse(
                $layer, $farbe, $linienstaerke, $linienart, $ymin,
                $ymax, $yschritt, $yskala, $ytext
            );
        }
    }
    elseif ( $zeile =~
/^koordinatensystem( layer=([0-9]+))?[ x | p]=([0-9\.]+),([0-9\.]+) [y | p]=([0-9\.]+)
,([0-9-9\.]+)( z=([0-9-9\.]+),([0-9-9\.]+))?( einteilung=([0-9-9\.]+),([0-9-9\.]+)
,([0-9-9\.]+))?( linie=([0-9-9\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(,\([^\"]*\)"\)?(
skala=(keine | nurstriche | nureins | normal | strichundeins))?( text="\([^\"]*\)"\",\([^\"]*\)"\")
,\([^\"]*\)"\")?( mitte)?$/i
    )
    {
        standard( 10, getlinienart (), 1.5, getfarbe(), "", getfuellfarbe (), "->", "" );
        $layer = $2 if ( defined $1 );
        $xmin = $3;
        $xmax = $4;
        $ymin = $5;
        $ymax = $6;
        if ( defined $7 ) {
            $zmin = $8;
            $zmax = $9;
        }
        $xschritt = 1;

```

```

$yschritt = 1;
$zschritt = 1;
if ( defined $10 ) {
    $xschritt = $11;
    $yschritt = $12;
    if ( defined $13 ) {
        $zschritt = $14;
    }
}

if ( defined $15 ) {
    $linienstaerke = $16;
    $linienart = $17;
}
$farbe = $19 if ( defined $18 );
$pfeilart = $21 if ( defined $20 );
$skala = "normal";
if ( defined $22 ) {
    $skala = $23;
}
$txt = $ytxt = $ztxt = "";
if ( defined $24 ) {
    $txt = $25;
    $ytxt = $26;
    if ( defined $27 ) {
        $ztxt = $28;
    }
}
if ( defined $29 ) { setmitte(); }
setminmaxachsen( $xmin, "x", "min" );
setminmaxachsen( $xmax, "x", "max" );
setschritt ( $xschritt, "x" );
setskala( $skala, "x" );
settext( $txt, "x" );
setminmaxachsen( $ymin, "y", "min" );
setminmaxachsen( $ymax, "y", "max" );
setschritt ( $yschritt, "y" );
setskala( $skala, "y" );
settext( $ytxt, "y" );
setachsengraph( $xmin, $ymin, $xmax, $ymax );

if ( ( defined $13 ) && ( defined $25 ) ) {
    achsen3d(
        $layer, $farbe, $linienstaerke, $linienart, $pfeilart,
        $xmin, $xmax, $ymin, $ymax, $zmin,
        $zmax, $xschritt, $skala, $txt, $yschritt,
        $skala, $ytxt, $zschritt, $skala, $ztxt
    );
}
else {
    achsen(
        $layer, $farbe, $linienstaerke, $linienart, $pfeilart,
        $xmin, $xmax, $ymin, $ymax, $xschritt,
        $skala, $txt, $yschritt, $skala, $ytxt
    );
}
}
elsif (
    $zeile =~

```

```

/^xyachserad( layer=([0-9]+))?( [x|p]=([0-9\..]+),([0-9\..]+) [y|p]=([0-9\..]+)
,([0-9\..]+)( einteilung=([0-9\..]+),([0-9\..]+)(,[0-9\..]+))?(,(rad|deg|degtorad|
radtodeg))?( linie=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( skala=(keine|
nurstriche|nureins|normal|strichundeins))?( text="\^[^"]*"\" ,"\^[^"]*"\" )?$/i
)
{
  standard( 10, getlinienart(), 0.8, getfarbe(), "", getfuellfarbe(), "->", "" );
  $layer = $2 if ( defined $1 );
  $xmin = $3;
  $xmax = $4;
  $ymin = $5;
  $ymax = $6;
  $xschritt = 3.14;
  $yschritt = 1;
  $xbruch = 1;
  $rad = "degtorad";
  setachsengraph( $xmin, $ymin, $xmax, $ymax );

  if ( defined $7 ) {
    $xschritt = $8;
    $yschritt = $9;
  }

  if ( defined $10 ) { $xbruch = $11; }
  if ( defined $12 ) { $rad = $13; }

  if ( defined $14 ) {
    $linienstaerke = $15;
    $linienart = $16;
  }
  $farbe = getfarbe();
  $farbe = $18 if ( defined $17 );
  $skala = "normal";
  if ( defined $19 ) { $skala = $20; }
  $xtext = $ytext = "";
  if ( defined $21 ) {
    $xtext = $22;
    $ytext = $23;
  }

  setminmaxachsen( degtorad($xmin), "x", "min" );
  setminmaxachsen( degtorad($xmax), "x", "max" );
  setschritt ( $xschritt, "x" );
  setskala( $skala, "x" );
  settext( $xtext, "x" );
  setminmaxachsen( $ymin, "y", "min" );
  setminmaxachsen( $ymax, "y", "max" );
  setschritt ( $yschritt, "y" );
  setskala( $skala, "y" );
  settext( $ytext, "y" );
  xyachserad(
    $layer, $farbe, $linienstaerke, $linienart, $xmin,
    $xmax, $ymin, $ymax, $xschritt, $xtext,
    $yschritt, $skala, $ytext, $xbruch, $rad
  );
}

# Zeichnen von Funktionen
elsif ( $zeile =~
/^graph( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+))?( linie=([0-9\..]+),([a-z0-9!]+))?(

```

```

parameter="\([^\""]*\)"? [x|p]=([-0-9\\.]+),([-0-9\\.]+) ([y|p]=([-0-9\\.]+),([-0-9\\.]+)
)?( z=([-0-9\\.]+),([-0-9\\.]+))?.*)$/i
)
{
  standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
  "" );
  $layer = $2 if ( defined $1 );

  $farbe = $4 if ( defined $3 );
  if ( defined $5 ) {
    $linienstaerke = $6;
    $linienart = $7;
  }
  if ( defined $8 ) { $parameter = $9; }

  $xmin = $10;
  $xmax = $11;
  $func = $18;

  if ( defined $12 ) {
    $ymin = $13;
    $ymax = $14;
  }
  if ( !defined $13 ) { $ymin = $ymax = 0; }

  if ( defined $15 ) {
    $zmin = $16;
    $zmax = $17;
  }
  else { $zmin = $zmax = 0; }

  graph(
    $layer,      $func,      $xmin, $ymin,
    $zmin,      $xmax,      $ymax, $zmax,
    $linienart, $linienstaerke, $farbe, $parameter
  );
}

## Rotationsfunktion um x-Achse zeichnen

elseif ( $zeile =~
/^graphrot( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\\.]+),([a-z0-9!]+))?(,[a-zA-Z0-9!]+))?( parameter="\([^\""]*\)"? [x|p
]=([-0-9\\.]+),([-0-9\\.]+) ([y|p]=([-0-9\\.]+),([-0-9\\.]+))?.*)$/i
)
{
  standard( 10, getlinienart (), 0.2, getfarbe(), "", getfuellfarbe (), "", "" );
  $layer = $2 if ( defined $1 );

  if ( defined $3 ) {
    $farbe = $4;
    $fuellart = $5;
    $fuellfarbe = $6;
  }

  if ( defined $7 ) {
    $linienstaerke = $8;
    $linienart = $9;
  }
  $farbe = $11 if ( defined $10 );

```

```

if ( defined $12 ) { $parameter = $13; }
$xmin = $14;
$xmax = $15;
if ( defined $16 ) {
    $ymin = $17;
    $ymax = $18;
}
$n = $19;
$func = $20;
if ( !defined $16 ) {
    $ymin = calcymin( $func, $xmin, $xmax );
    $ymax = calcymin( $func, $xmin, $xmax );
}
graphrot(
    $layer,      $func,      $xmin, $ymin,  $xmax,
    $ymax,      $n,         $farbe, $fuellart, $fuellfarbe,
    $linienart, $linienstaerke, $parameter
);
}

elseif ( $zeile =~
/^folge( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+))?( linie=([0-9\.\.]+),([a-zA-Z0-9!]+))?(
parameter="\([^"]*\)" )? [x|p]=([0-9\.\.]+),([0-9\.\.]+)( r=([0-9\.\.]+),([0-9\.\.]+))?(
.*)$/i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart = $7;
    }
    if ( defined $8 ) { $parameter = $9; }

    $xmin = $10;
    $xmax = $11;
    $func = $15;
    $dx = 1;
    $groesse = 5;

    if ( defined $12 ) {
        $dx = $13;
        $groesse = $14;
    }

    folge(
        $layer, $func, $xmin, $xmax, $dx,
        $groesse, $linienart, $linienstaerke, $farbe, $parameter
    );
}

elseif ( $zeile =~
/^parametricplot( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+))?( linie=([0-9\.\.]+),([a-zA-Z0-9!]+)
)?( parameter="\([^"]*\)" )? t=([0-9\.\.]+),([0-9\.\.]+)( u=([0-9\.\.]+),([0-9\.\.]+))?( ([^
]*),([^\.]*)([, ]*))?$/i
)

```

)	1099
{	1100
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	1101
"");	1102
\$layer = \$2 if (defined \$1);	1103
\$farbe = \$4 if (defined \$3);	1104
	1105
if (defined \$5) {	1106
\$linienstaerke = \$6;	1107
\$linienart = \$7;	1108
}	1109
if (defined \$8) { \$parameter = \$9; }	1110
	1111
\$tmin = \$10;	1112
\$tmax = \$11;	1113
	1114
if (defined \$12) {	1115
\$umin = \$13;	1116
\$umax = \$14;	1117
	1118
}	1119
else { \$umin = \$umax = 0; }	1120
\$func1 = \$15;	1121
\$func2 = \$16;	1122
if (defined \$17) { \$func3 = \$18; }	1123
else { \$func3 = ""; }	1124
	1125
parametricplot(	1126
\$layer, \$func1, \$func2, \$func3,	1127
\$tmin, \$umin, \$tmax, \$umax,	1128
\$linienart, \$linienstaerke, \$farbe, \$parameter	1129
);	1130
	1131
}	1132
	1133
# Zeichnen der Ableitungsfunktion	1134
elsif (\$zeile =~	1135
/^ableitung(layer=([0-9\.\+])?(farbe=([a-zA-Z0-9\.\+])?(linie=([0-9\.\+])([a-zA-Z0-9\.\+])?(	1136
parameter=\"([^\"]*)\")? [x p]=([0-9\.\+])([0-9\.\+])([y p]=([0-9\.\+])([0-9\.\+])	
)? ([^,]*)?([^\"]*)?\$/i	
)	1137
{	1138
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	1139
"");	1140
\$layer = \$2 if (defined \$1);	1141
\$farbe = \$4 if (defined \$3);	1142
	1143
if (defined \$5) {	1144
\$linienstaerke = \$6;	1145
\$linienart = \$7;	1146
}	1147
	1148
if (defined \$8) { \$parameter = \$9; }	1149
	1150
\$xmin = \$10;	1151
\$xmax = \$11;	1152
\$func = \$15;	1153
	1154
if (defined \$12) {	1155
\$ymin = \$13;	1156
\$ymax = \$14;	1157

```

}
if ( !defined $13 ) {
    $ymin = calcymin( $func, $xmin, $xmax );
    $ymax = calcymax( $func, $xmin, $xmax );
}
if ( defined $16 ) { $grad = $17; }
else { $grad = 1; }
hilfe( $layer, "grad", $grad );
ableitung(
    $layer,      $func,  $grad, $xmin,
    $ymin,      $xmax, $ymax, $linienart,
    $linienstaerke, $farbe, $parameter
);
}

# Zeichnen der Stammfunktion
elsif ( $zeile =~
/^stammfunktion( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+))?( linie=([0-9\.\.]+),([a-z0
-9!]+))?( [x|p]=([0-9\.\.]+),([0-9\.\.]+)( [y|p]=([0-9\.\.]+),([0-9\.\.]+))?( parameter
=\\("([^\"]*)\\")?( nullwert=([0-9\.\.]+))?(.*)$/i
)
{
    standard( 10, getlinienart(), 0.2, getfarbe(), "", getfuellfarbe(), "", "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart = $7;
    }
    $xmin = $8;
    $xmax = $9;

    $func = $17;
    $nullwert = 0;
    $nullwert = $16 if ( defined $15 );
    $ymin = calcymin( $func, $xmin, $xmax );
    $ymax = calcymax( $func, $xmin, $xmax );

    if ( defined $10 ) {
        $ymin = $11;
        $ymax = $12;
    }

    if ( defined $13 ) { $parameter = $14; }

    stammfunktion(
        $layer, $func, $xmin, $ymin,
        $xmax, $ymax, $linienart, $linienstaerke,
        $farbe, $nullwert, $parameter
    );
}
elsif ( $zeile =~
/^integral( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.\.]+),([a-z0-9!]+))?(,([a-zA-Z0-9!]+))?( parameter=\\("([^\"]*)\\")?( [x|p
]=([0-9\.\.]+),([0-9\.\.]+)( [y|p]=([0-9\.\.]+),([0-9\.\.]+))?(.*)$/i
)
{

```

standard(10, getlinienart (), 0.2, getfarbe(), "", getfuellfarbe (), "", "");	1215
\$layer = \$2 if (defined \$1);	1216
	1217
if (defined \$3) {	1218
\$farbe = \$4;	1219
\$fuellart = \$5;	1220
\$fuellfarbe = \$6;	1221
}	1222
	1223
if (defined \$7) {	1224
\$linienstaerke = \$8;	1225
\$linienart = \$9;	1226
}	1227
\$farbe = \$11 if (defined \$10);	1228
	1229
if (defined \$12) { \$parameter = \$13; }	1230
	1231
\$xmin = \$14;	1232
\$xmax = \$15;	1233
if (defined \$16) {	1234
\$ymin = \$17;	1235
\$ymax = \$18;	1236
}	1237
\$func = \$19;	1238
if (!defined \$16) {	1239
\$ymin = calcymin(\$func, \$xmin, \$xmax);	1240
\$ymax = calcymax(\$func, \$xmin, \$xmax);	1241
}	1242
integral (	1243
\$layer, \$func, \$xmin, \$ymin,	1244
\$xmax, \$ymax, \$farbe, \$fuellart ,	1245
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter	1246
);	1247
	1248
}	1249
	1250
# Zeichnen von Funktionen	1251
elseif (\$zeile =~	1252
/^wertezeichnen(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.\.]+),([a-zA-Z0-9!]+)	1253
)?(parameter=\"([^\"]*)\")?\$/i	
)	1254
{	1255
settexthier ("");	1256
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	1257
"");	1258
\$layer = \$2 if (defined \$1);	1259
\$farbe = \$4 if (defined \$3);	1260
	1261
if (defined \$5) {	1262
\$linienstaerke = \$6;	1263
\$linienart = \$7;	1264
}	1265
\$enderreicht = 0;	1266
\$inzeile = 0;	1267
\$datei++;	1268
\$name = "datei" . \$datei;	1269
if (defined \$8) { \$parameter = \$9; }	1270
	1271
while ((!\$enderreicht)	1272
&& (\$i < \$eingabedateizeilen - 1))	1273
{	1274

```

    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
    elsif ( $hilfszeile =~ /^([-0-9\.])([l;|,|; ])([-0-9\.]$)/i ) {
        $x = $1;
        $y = $2;
        paar( $x, $y );
        setachsengraph( $x, $y, $x, $y );
    }
    elsif ( $hilfszeile =~
        /^([-0-9\.])([l;|,|; ])([-0-9\.])([l;|,|; ])([-0-9\.]$)/i )
    {
        $x = $1;
        $y = $2;
        $z = $3;
        tripel( $x, $y, $z );
    }
    else {
        kommentar( $layer, "Kaputte Zeile" );
        $enderreicht = 1;
    }
}
if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
else {
    liniekopf( $layer, $linienart, $linienstaerke, $farbe, $parameter, "",
        gettexthier() );
    settexthier( "" );
}

}
elsif ( $zeile =~
/^polypunkte( layer=([-0-9]+) )?( farbe=([a-zaeoue0-9!]+) )?( linie=([-0-9\.])([a-z]+) )?(
parameter="\([^"]*\)" )?$/i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
        "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart = $7;
    }

    if ( defined $8 ) { $parameter = $9; }

    $enderreicht = 0;
    $inzeile = 0;
    $polygonzeichnen = 0;
    $datei++;
    $name = "datei" . $datei;
    while ( ( !$enderreicht )
        && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) {
            $enderreicht = 1;
            if ( $polygonzeichnen == 1 ) { schliessedatei( $name ); }
        }
    }
}

```

```
elseif ( $hilfszeile =~ /^lese (.*)$/i ) {
    leseingabedatei($1);
    $datei = $1;

    $inzeile++;
}
elseif ( $hilfszeile =~ /^([-0-9\\.]+)[,|;| , | ; ]([-0-9\\.]+)$/i ) {
    $x = $1;
    $y = $2;
    setachsengraph( $x, $y, $x, $y );
    if ( $inzeile == 0 ) { oeffnedatei($name); }
    $inzeile++;

    schreibedaten( $name, $x, $y );
    $polygonzeichnen = 1;
}
elseif ( $hilfszeile =~
    /^([-0-9\\.]+)[,|;| , | ; ]([-0-9\\.]+)[,|;| , | ; ]([-0-9\\.]+)$/i )
{
    $x = $1;
    $y = $2;
    $z = $3;
    if ( $inzeile == 0 ) {
        oeffnedatei($name);
    }
    $inzeile++;
    minmax( $x, $x, $y, $y );
    schreibedaten3d( $name, $x, $y, $z );
    $polygonzeichnen = 1;
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;
}
}
if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
else {
    readdata( $layer, $name );
    daten( $layer, $name, $linienart, $linienstaerke, $farbe, $parameter );
}
}
}
elseif ( $zeile =~ /^schreibe=\"([^\"]*)\"$/i ) {
    setextern(1);
    $name = $1;
    $enderreicht = 0;
    $inzeile = 0;
    while ( ( !$enderreicht )
        && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
        elseif ( $hilfszeile =~ /^([-0-9\\.]+)[,|;| , | ; ]([-0-9\\.]+)$/i ) {
            $x = $1;
            $y = $2;
            setachsengraph( $x, $y, $x, $y );
            if ( $inzeile == 0 ) { oeffnedatei($name); }
            $inzeile++;
            schreibedaten( $name, $x, $y );
        }
    }
}
```

```

elseif ( $hilfszeile =~
    /^( [-0-9\.] + [ , | ; ] [ (-0-9\.) + ] [ , | ; ] [ (-0-9\.) + ] $ /i )
{
    $x = $1;
    $y = $2;
    $z = $3;
    if ( $inzeile == 0 ) { oeffnedei($name); }
    $inzeile++;
    schreibedaten3d( $name, $x, $y, $z );
}

else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;
}
}
if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
else {
    schliessedatei( $name );
}
}

elseif ( $zeile =~ /^schreibeindei="\^[^"]*"$/i ) {

    $name = $1;
    setextern(1);
    $enderreicht = 0;
    $inzeile = 0;
    while ( ( !$enderreicht )
        && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
        elseif ( $hilfszeile =~ /^( [-0-9\.] + [ , | ; ] [ (-0-9\.) + ] $ /i ) {
            $x = $1;
            $y = $2;
            setachsengraph( $x, $y, $x, $y );
            if ( $inzeile == 0 ) { oeffnedei($name); }
            $inzeile++;
            schreibedaten( $name, $x, $y );
        }

        elseif ( $hilfszeile =~
            /^( [-0-9\.] + [ , | ; ] [ (-0-9\.) + ] [ , | ; ] [ (-0-9\.) + ] $ /i )
        {
            $x = $1;
            $y = $2;
            $z = $3;
            if ( $inzeile == 0 ) { oeffnedei($name); }
            $inzeile++;
            schreibedaten3d( $name, $x, $y, $z );
        }

        else {
            kommentar( $layer, "Kaputte Zeile" );
            $enderreicht = 1;
        }
    }
}

```

}	1457
if (!\$deerreicht) { printf("Keine sinnvollen Daten\n"); }	1458
else {	1459
schliessedatei (\$name);	1460
setextern(0);	1461
}	1462
}	1463
	1464
elseif (\$zeile =~	1465
/^dataplot(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+))?(linie=([0-9\.]+),([a-z]+))?(	1466
parameter="\([^\""]*\)"?)\$/i	
)	1467
{	1468
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	1469
"");	1470
\$layer = \$2 if (defined \$1);	1471
\$farbe = \$4 if (defined \$3);	1472
	1473
if (defined \$5) {	1474
\$linienstaerke = \$6;	1475
\$linienart = \$7;	1476
	1477
}	1478
if (defined \$8) { \$parameter = \$9; }	1479
	1480
\$deerreicht = 0;	1481
\$inzeile = 0;	1482
\$polygonzeichnen = 0;	1483
\$datei++;	1484
\$name = "datei" . \$datei;	1485
while ((!\$deerreicht)	1486
&& (\$i < \$eingabedateizeilen - 1))	1487
{	1488
\$i++;	1489
\$hilfszeile = \$eingabedatei[\$i];	1490
if (\$hilfszeile =~ /^ende\$/i) {	1491
\$deerreicht = 1;	1492
if (\$polygonzeichnen == 1) { schliessedatei (\$name); }	1493
}	1494
elseif (\$hilfszeile =~ /^lese (.*)\$/i) {	1495
leseingabedatei(\$1);	1496
\$name = \$1;	1497
if (\$inzeile == 0) { }	1498
\$inzeile++;	1499
}	1500
elseif (\$hilfszeile =~ /^([-0-9\.]+)[, ; , ;]([-0-9\.]+)\$/i) {	1501
\$x = \$1;	1502
\$y = \$2;	1503
setachsengraph(\$x, \$y, \$x, \$y);	1504
if (\$inzeile == 0) { oeffnedatei(\$name); }	1505
\$inzeile++;	1506
	1507
schreibedaten(\$name, \$x, \$y);	1508
\$polygonzeichnen = 1;	1509
}	1510
elseif (\$hilfszeile =~	1511
/^([-0-9\.]+)[, ; , ;]([-0-9\.]+)[, ; , ;]([-0-9\.]+)\$/i)	1512
{	1513
\$x = \$1;	1514
\$y = \$2;	1515
\$z = \$3;	1516

```

    if ( $inzeile == 0 ) { oeffnedei($name); }
    $inzeile++;

    schreibedaten3d( $name, $x, $y, $z );
    $polygonzeichnen = 1;
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;
}
}
if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
else {
    readdata( $layer, $name );
    daten( $layer, $name, $linienart, $linienstaerke, $farbe, $parameter );
    settexthier ( "" );
}
}
elseif ( $zeile =~ /^schreibe=\"([^\"]*)\"$/i ) {

    $datei      = $1;
    $enderreicht = 0;
    $inzeile    = 0;
    while ( ( !$enderreicht )
        && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
        elseif ( $hilfszeile =~ /^([-0-9\\.]+)[,|;| | ; ]([-0-9\\.]+)$/i ) {
            $x = $1;
            $y = $2;
            setachsengraph( $x, $y, $x, $y );
            if ( $inzeile == 0 ) { oeffnedei($name); }
            $inzeile++;
            schreibedaten( $name, $x, $y );
        }

        elseif ( $hilfszeile =~
            /^([-0-9\\.]+)[,|;| | ; ]([-0-9\\.]+)[,|;| | ; ]([-0-9\\.]+)$/i )
        {
            $x = $1;
            $y = $2;
            $z = $3;
            if ( $inzeile == 0 ) { oeffnedei($name); }
            $inzeile++;
            schreibedaten3d( $name, $x, $y, $z );
        }

        else {
            kommentar( $layer, "Kaputte Zeile" );
            $enderreicht = 1;
        }
    }
    if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
    else { schliessedatei( $name ); }
}

elseif ( $zeile =~
/^fileplot( layer=([0-9]+) )?( farbe=([a-zA-Z0-9!]+) )?( linie=([0-9\\.]+),([a-z]+) )?(
parameter=\"([^\"]*)\" )?$/i

```

)	1577
{	1578
\$einlesen = 1;	1579
standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",	1580
"");	1581
\$layer = \$2 if (defined \$1);	1582
\$farbe = \$4 if (defined \$3);	1583
	1584
if (defined \$5) {	1585
\$linienstaerke = \$6;	1586
\$linienart = \$7;	1587
	1588
}	1589
if (defined \$8) { \$parameter = \$9; }	1590
	1591
\$enderreicht = 0;	1592
\$inzeile = 0;	1593
\$polygonzeichnen = 0;	1594
\$datei++;	1595
\$name = "datei" . \$datei;	1596
while ((!\$enderreicht)	1597
&& (\$i < \$eingabedateizeilen - 1))	1598
{	1599
\$i++;	1600
\$hilfszeile = \$eingabedatei[\$i];	1601
if (\$hilfszeile =~ /^ende\$/i) {	1602
\$enderreicht = 1;	1603
if (\$polygonzeichnen == 1) { schliessdatei(\$name); }	1604
}	1605
elsif (\$hilfszeile =~ /^lese(.*)\$/i) {	1606
	1607
\$name = \$1;	1608
if (\$inzeile == 0) { }	1609
\$inzeile++;	1610
leseeingabedatei(\$1);	1611
}	1612
elsif (\$hilfszeile =~ /^([0-9\.]+)[, ; , ;]([0-9\.]+)\$/i) {	1613
\$x = \$1;	1614
\$y = \$2;	1615
	1616
setachsengraph(\$x, \$y, \$x, \$y);	1617
if (\$inzeile == 0) { oeffnedatei(\$name); }	1618
\$inzeile++;	1619
schreibedaten(\$name, \$x, \$y);	1620
\$polygonzeichnen = 1;	1621
}	1622
elsif (\$hilfszeile =~	1623
/^([0-9\.]+)[, ; , ;]([0-9\.]+)[, ; , ;]([0-9\.]+)\$/i)	1624
{	1625
\$x = \$1;	1626
\$y = \$2;	1627
\$z = \$3;	1628
if (\$inzeile == 0) { oeffnedatei(\$name); }	1629
\$inzeile++;	1630
	1631
schreibedaten3d(\$name, \$x, \$y, \$z);	1632
\$polygonzeichnen = 1;	1633
}	1634
else {	1635
kommentar(\$layer, "Kaputte Zeile");	1636
\$enderreicht = 1;	1637

```

    }
  }
  if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
  else {
    fileplot ( $layer, $name, $linienart, $linienstaerke, $farbe, $parameter );
    settexthier ("" );
  }
}
elseif ( $zeile =~ /^schreibe=\"([^\"]*)\"$/i ) {
  $name = $1;
  $enderreicht = 0;
  $inzeile = 0;
  while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
  {
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) {
      $enderreicht = 1;
    }
    elseif ( $hilfszeile =~ /^([-0-9\.]|[,|;| ]|([-0-9\.]|([a-z]+)))$/i ) {
      $x = $1;
      $y = $2;
      setachsengraph( $x, $y, $x, $y );
      if ( $inzeile == 0 ) { oeffnedatei($name); }
      $inzeile++;
      schreibedaten( $name, $x, $y );
    }
    else {
      kommentar( $layer, "Kaputte Zeile" );
      $enderreicht = 1;
    }
  }
  if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
  else { schliessedatei( $name ); }
}

elseif ( $zeile =~
/^listplot( layer=([-0-9]+))?( farbe=([a-zaeoeue0-9!]+))?( linie=([-0-9\.]|([a-z]+)))?(
parameter=\"([^\"]*)\" )?$/i
)
{
  $einlesen = 1;
  settexthier ("" );
  standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
  $layer = $2 if ( defined $1 );
  $farbe = $4 if ( defined $3 );

  if ( defined $5 ) {
    $linienstaerke = $6;
    $linienart = $7;
  }
  if ( defined $8 ) { $parameter = $9; }
  $enderreicht = 0;
  $inzeile = 0;
  $polygonzeichnen = 0;
  $datei++;
  $name = "datei" . $datei;
  while ( ( !$enderreicht )

```

```

    && ( $i < $eingabedateizeilen - 1 ) )
{
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) {
        $enderreicht = 1;
        if ( $polygonzeichnen == 1 ) { schliessedatei($name); }
    }
    elseif ( $hilfszeile =~ /^lese (.*)$/i ) {
        $name = $1;
        if ( $inzeile == 0 ) { }
        $inzeile++;
        leseeingabedatei($1);
    }
    elseif ( $hilfszeile =~ /^([-0-9\.] +)[,| ; ]([-0-9\.] +)$/i ) {
        $x = $1;
        $y = $2;
        setachsengraph( $x, $y, $x, $y );
        if ( $inzeile == 0 ) { oeffnedatei($name); }
        $inzeile++;
        schreibedaten( $name, $x, $y );
        $polygonzeichnen = 1;
    }

    elseif ( $hilfszeile =~
        /^([-0-9\.] +)[,| ; ]([-0-9\.] +)[,| ; ]([-0-9\.] +)$/i )
    {
        $x = $1;
        $y = $2;
        $z = $3;
        setachsengraph3d( $x, $y, $z, $x, $y, $z );
        minmax3d( $x, $y, $z, $x, $y, $z );
        if ( $inzeile == 0 ) { oeffnedatei($name); }
        $inzeile++;
        schreibedaten3d( $name, $x, $y, $z );
        $polygonzeichnen = 1;
    }
    else {
        kommentar( $layer, "Kaputte Zeile" );
        $enderreicht = 1;
    }
}
if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
else {
    readdata( $layer, $name );
    listplot ( $layer, $name, $linienart, $linienstaerke, $farbe, $parameter );
    settexthier ("" );
}

elseif ( $zeile =~ /^schreibe=\"([^\"]*)\"$/i ) {
    $name = $1;
    $enderreicht = 0;
    $inzeile = 0;
    while ( !$enderreicht )
        && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) {
            $enderreicht = 1;

```

```

    }
    elseif ( $hilfszeile =~ /^([-0-9\.][,|,| ; ]([-0-9\.]+)$)/i ) {
        $x = $1;
        $y = $2;
        setachsengraph( $x, $y, $x, $y );
        if ( $inzeile == 0 ) { oeffnedei($name); }
        $inzeile++;
        schreibedaten( $name, $x, $y );
    }
    else {
        kommentar( $layer, "Kaputte Zeile" );
        $enderreicht = 1;
    }
}
if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
else { schliessedatei( $name); }
}

elseif ( $zeile =~
/^regression( layer=([0-9]+)?) ( farbe=([a-zA-Z0-9!]+)?) ( linie=([0-9\.]+),([a-z]+)?) (
parameter="\([^"]*\)" )? )$/i
)
{
    $einlesen = 1;
    settexthier( "" );
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart = $7;
    }
    if ( defined $8 ) { $parameter = $9; }

    $enderreicht = 0;
    $inzeile = 0;
    $polygonzeichnen = 0;
    while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) {
            $enderreicht = 1;
            if ( $polygonzeichnen == 1 ) {
                schliesseregression( $inzeile );
            }
        }
    }
    elseif ( $hilfszeile =~ /^([-0-9\.][,|,| ; ]([-0-9\.]+)$)/i ) {

        $x = $1;
        $y = $2;
        setachsengraph( $x, $y, $x, $y );
        if ( $inzeile == 0 ) { oeffneregression(); }
        $inzeile++;
        schreiberegression( $x, $y, "", $inzeile );
        $polygonzeichnen = 1;
    }
}

```

elseif (\$hilfszeile =~	1819
/^[(-0-9\.]+)[, ; , ;]((-0-9\.)+)\$/i)	1820
{	1821
\$x = \$1;	1822
\$y = \$2;	1823
\$z = \$3;	1824
setachsengraph3d(\$x, \$y, \$z, \$x, \$y, \$z);	1825
minmax3d(\$x, \$y, \$z, \$x, \$y, \$z);	1826
if (\$inzeile == 0) { oeffneregression(); }	1827
\$inzeile++;	1828
	1829
schreiberegression(\$x, \$y, \$z, \$inzeile);	1830
\$polygonzeichnen = 1;	1831
}	1832
else {	1833
kommentar(\$layer, "Kaputte Zeile");	1834
\$endeerreicht = 1;	1835
}	1836
}	1837
if (!\$endeerreicht) { printf("Keine sinnvollen Daten\n"); }	1838
else {	1839
regression(\$layer, 1, \$linienart, \$linienstaerke, \$farbe, \$parameter);	1840
}	1841
}	1842
elseif (\$zeile =~ /^schreibe=\"([^\"]*)\"\$/i) {	1843
\$name = \$1;	1844
\$endeerreicht = 0;	1845
\$inzeile = 0;	1846
while ((!\$endeerreicht)	1847
&& (\$i < \$eingabedateizeilen - 1))	1848
{	1849
\$i++;	1850
\$hilfszeile = \$eingabedatei[\$i];	1851
if (\$hilfszeile =~ /^ende\$/i) {	1852
\$endeerreicht = 1;	1853
}	1854
elseif (\$hilfszeile =~ /^[(-0-9\.]+)[, ; , ;]((-0-9\.)+)\$/i) {	1855
\$x = \$1;	1856
\$y = \$2;	1857
setachsengraph(\$x, \$y, \$x, \$y);	1858
if (\$inzeile == 0) { oeffnedei(\$name); }	1859
\$inzeile++;	1860
schreibedaten(\$name, \$x, \$y);	1861
}	1862
else {	1863
kommentar(\$layer, "Kaputte Zeile");	1864
\$endeerreicht = 1;	1865
}	1866
}	1867
if (!\$endeerreicht) { printf("Keine sinnvollen Daten\n"); }	1868
else { schliessedatei(\$name); }	1869
}	1870
	1871
elseif (	1872
\$zeile =~	1873
/^(bezier(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.]+),([a-zA-Z0-9!]+))	1874
?,\"([^\"]*)\"?)?(parameter=\"([^\"]*)\"?)?\$/i	1875
)	1876
{	1877
	1878

settexthier ("");	1879
\$pstrickersatz3d = get3dpstricks();	1880
\$pstrickersatz = getpstricks();	1881
	1882
setpstricks(0);	1883
set3dpstricks(0);	1884
	1885
	1886
	1887
standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",	1888
""));	1889
\$layer = \$2 if (defined \$1);	1890
\$farbe = \$4 if (defined \$3);	1891
	1892
if (defined \$5) {	1893
\$linienstaerke = \$6;	1894
\$linienart = \$7;	1895
}	1896
\$pfeilart = \$9 if (defined \$8);	1897
	1898
if (defined \$10) { \$parameter = \$11; }	1899
	1900
\$sendeerreicht = 0;	1901
\$inzeile = 0;	1902
\$polygonzeichnen = 0;	1903
\$datei++;	1904
\$name = "datei" . \$datei;	1905
while ((!\$sendeerreicht)	1906
&& (\$i < \$eingabedateizeilen - 1))	1907
{	1908
\$i++;	1909
\$hilfszeile = \$eingabedatei[\$i];	1910
if (\$hilfszeile =~ /^ende\$/i) { \$sendeerreicht = 1; }	1911
elsif (\$hilfszeile =~ /^lese (.*)\$/i) {	1912
leseeingabedatei(\$1);	1913
\$name = \$1;	1914
\$inzeile++;	1915
}	1916
elsif (\$hilfszeile =~ /^([-0-9\.] +)[, ;]+([-0-9\.] +)\$/i) {	1917
\$x = \$1;	1918
\$y = \$2;	1919
setachsengraph(\$x, \$y, \$x, \$y);	1920
minmax(1.5 * \$x, \$1, 5 * \$y, 1.5 * \$x, \$1, 5 * \$y);	1921
\$inzeile++;	1922
paar(\$x, \$y);	1923
setachsengraph(\$x, \$y, \$x, \$y);	1924
}	1925
elsif (\$hilfszeile =~	1926
/^([-0-9\.] +)[, ;]+([-0-9\.] +)[, ;]+([-0-9\.] +)\$/i)	1927
{	1928
\$x = \$1;	1929
\$y = \$2;	1930
\$z = \$3;	1931
\$inzeile++;	1932
setachsengraph3d(\$x, \$y, \$z, \$x, \$y, \$z);	1933
tripel(\$x, \$y, \$z);	1934
}	1935
	1936
	1937
else {	1938
kommentar(\$layer, "Kaputte Zeile");	1939

\$enderreicht = 1;	1940
}	1941
}	1942
if (!\$enderreicht) { printf("Keine sinnvollen Daten\n"); }	1943
else {	1944
if (\$parameter =~ /showpoints/) {	1945
\$parameter =~ s/showpoints//g;	1946
\$parameter =~ s/=true//g;	1947
liniekopf(\$layer, \$linienart, \$linienstaerke, \$farbe, \$parameter, \$pfeilart,	1948
gettexthier());	1949
}	1950
bezierkopf(\$layer, \$linienart, \$linienstaerke, \$farbe, \$parameter, \$pfeilart,	1951
gettexthier());	1952
	1953
settexthier("");	1954
setpstricks(\$pstrickersatz);	1955
set3dpstricks(\$pstrickersatz3d);	1956
	1957
}	1958
}	1959
elseif (\$zeile =~ /^schreibe="\([^"]*)\"\$/i) {	1960
\$name = \$1;	1961
\$enderreicht = 0;	1962
\$inzeile = 0;	1963
while ((!\$enderreicht)	1964
&& (\$i < \$eingabedateizeilen - 1))	1965
{	1966
\$i++;	1967
\$hilfszeile = \$eingabedatei[\$i];	1968
if (\$hilfszeile =~ /^ende\$/i) { \$enderreicht = 1; }	1969
elsif (\$hilfszeile =~ /^[(-0-9\\.]+)[, ;]+([(-0-9\\.]+)\$/i) {	1970
\$x = \$1;	1971
\$y = \$2;	1972
setachsengraph(\$x, \$y, \$x, \$y);	1973
if (\$inzeile == 0) { oeffnedeitei(\$name); }	1974
\$inzeile++;	1975
schreibedaten(\$name, \$x, \$y);	1976
}	1977
elsif (\$hilfszeile =~	1978
/^([(-0-9\\.]+)[, ;]+[(-0-9\\.]+)[, ;]+[(-0-9\\.]+)\$/i)	1979
{	1980
\$x = \$1;	1981
\$y = \$2;	1982
\$z = \$3;	1983
if (\$inzeile == 0) { oeffnedeitei(\$name); }	1984
\$inzeile++;	1985
schreibedaten3d(\$name, \$x, \$y, \$z);	1986
}	1987
	1988
} else {	1989
kommentar(\$layer, "Kaputte Zeile");	1990
\$enderreicht = 1;	1991
}	1992
}	1993
if (!\$enderreicht) { printf("Keine sinnvollen Daten\n"); }	1994
else { schliessedeitei(\$name); }	1995
}	1996
	1997
elseif (	1998
\$zeile =~	1999
	2000

```

/^linie( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+))?( linie=([0-9\.]+),([a-z0-9!]+))
?(\\"([^\"]*)\\")?( parameter=\\"([^\"]*)\\")?$/i
)
{
    settexthier("");

    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart      = $7;
    }

    $pfeilart = $9 if ( defined $8 );

    if ( defined $10 ) { $parameter = $11; }

    $enderreicht = 0;
    $inzeile      = 0;
    $polygonzeichnen = 0;
    $datei++;
    $name = "datei" . $datei;
    while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
        elsif ( $hilfszeile =~ /^lese (.*)$/i ) {
            leseingabedatei($1);
            $name = $1;
            $inzeile++;
        }
        elsif ( $hilfszeile =~ /^([-0-9\.]+)[,|;| , | ; ]([-0-9\.]+)$/i ) {
            $x = $1;
            $y = $2;
            setachsengraph( $x, $y, $x, $y );
            paar( $x, $y );
        }

        elsif ( $hilfszeile =~
        /^([-0-9\.]+)[,|;| , | ; ]([-0-9\.]+)[,|;| , | ; ]([-0-9\.]+)$/i )
        {
            $x = $1;
            $y = $2;
            $z = $3;
            tripel( $x, $y, $z );
        }

        else {
            kommentar( $layer, "Kaputte Zeile" );
            $enderreicht = 1;
        }
    }
    if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
    else {
        liniekopf( $layer, $linienart, $linienstaerke, $farbe, $parameter, $pfeilart,
        gettexthier() );
    }
}

```

settexthier ("");	2061
}	2062
}	2063
elseif (\$zeile =~ /^schreibe=\"([^\"]*)\"\$/i) {	2064
\$name = \$1;	2065
\$enderreicht = 0;	2066
\$inzeile = 0;	2067
while ((!\$enderreicht)	2068
&& (\$i < \$eingabedateizeilen - 1))	2069
{	2070
\$i++;	2071
\$hilfszeile = \$eingabedatei[\$i];	2072
if (\$hilfszeile =~ /^ende\$/i) { \$enderreicht = 1; }	2073
elseif (\$hilfszeile =~ /^([-0-9\\.]+)[, ;]([-0-9\\.]+)\$/i) {	2074
\$x = \$1;	2075
\$y = \$2;	2076
setachsengraph(\$x, \$y, \$x, \$y);	2077
if (\$inzeile == 0) { oeffnedei(\$name); }	2078
\$inzeile++;	2079
schreibedaten(\$name, \$x, \$y);	2080
}	2081
else {	2082
kommentar(\$layer, "Kaputte Zeile");	2083
\$enderreicht = 1;	2084
}	2085
}	2086
if (!\$enderreicht) { printf("Keine sinnvollen Daten\n"); }	2087
else { schliessedatei(\$name); }	2088
}	2089
	2090
elseif (	2091
\$zeile =~	2092
/^kurve(layer=([-0-9]+)?(farbe=([a-zaeoeue0-9!]+)?(linie=([-0-9\\.]+),([a-z0-9!]+))	2093
?,\"([^\"]*)\")?(parameter=\"([^\"]*)\")?\$/i	2094
)	2095
{	2096
	2097
settexthier ("");	2098
	2099
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	2100
"");	2101
\$layer = \$2 if (defined \$1);	2102
\$farbe = \$4 if (defined \$3);	2103
	2104
	2105
if (defined \$5) {	2106
\$linienstaerke = \$6;	2107
\$linienart = \$7;	2108
}	2109
	2110
\$pfeilart = \$9 if (defined \$8);	2111
	2112
if (defined \$10) { \$parameter = \$11; }	2113
	2114
\$enderreicht = 0;	2115
\$inzeile = 0;	2116
\$polygonzeichnen = 0;	2117
\$datei++;	2118
\$name = "datei" . \$datei;	2119
while ((!\$enderreicht)	2120

&& (\$i < \$eingabedateizeilen - 1))	2121
{	2122
\$i++;	2123
\$hilfszeile = \$eingabedatei[\$i];	2124
if (\$hilfszeile =~ /^ende\$/i) { \$enderreicht = 1; }	2125
elsif (\$hilfszeile =~ /^lese (.*)\$/i) {	2126
leseeingabedatei(\$1);	2127
\$name = \$1;	2128
	2129
\$inzeile++;	2130
}	2131
elsif (\$hilfszeile =~ /^([0-9\\.]+)[, ; , ;]([0-9\\.]+)\$/i) {	2132
\$x = \$1;	2133
\$y = \$2;	2134
setachsengraph(\$x, \$y, \$x, \$y);	2135
\$inzeile++;	2136
minmax(1.5 * \$x, \$1, 5 * \$y, 1.5 * \$x, \$1, 5 * \$y);	2137
paar(\$x, \$y);	2138
	2139
}	2140
	2141
elsif (\$hilfszeile =~	2142
/^([0-9\\.]+)[, ; , ;]([0-9\\.]+)[, ; , ;]([0-9\\.]+)\$/i)	2143
{	2144
\$x = \$1;	2145
\$y = \$2;	2146
\$z = \$3;	2147
\$inzeile++;	2148
setachsengraph3d(\$x, \$y, \$z, \$x, \$y, \$z);	2149
tripel(\$x, \$y, \$z);	2150
	2151
}	2152
	2153
else {	2154
kommentar(\$layer, "Kaputte Zeile");	2155
\$enderreicht = 1;	2156
}	2157
}	2158
if (!\$enderreicht) {	2159
printf("Keine sinnvollen Daten\n");	2160
}	2161
else {	2162
kurvekopf(\$layer, \$linienart, \$linienstaerke, \$farbe, \$parameter, \$pfeilart,	2163
gettexthier());	2164
	2165
settexthier("");	2166
}	2167
}	2168
elsif (\$zeile =~ /^schreibe=\\("[^"]*"\\)\$\$/i) {	2169
\$name = \$1;	2170
\$enderreicht = 0;	2171
\$inzeile = 0;	2172
while ((!\$enderreicht)	2173
&& (\$i < \$eingabedateizeilen - 1))	2174
{	2175
\$i++;	2176
\$hilfszeile = \$eingabedatei[\$i];	2177
if (\$hilfszeile =~ /^ende\$/i) {	2178
\$enderreicht = 1;	2179
}	2180
elsif (\$hilfszeile =~ /^([0-9\\.]+)[, ; , ;]([0-9\\.]+)\$/i) {	2181

\$x = \$1;	2182
\$y = \$2;	2183
setachsengraph(\$x, \$y, \$x, \$y);	2184
if (\$inzeile == 0) {	2185
oeffnedei(\$name);	2186
}	2187
\$inzeile++;	2188
schreibedaten(\$name, \$x, \$y);	2189
}	2190
else {	2191
kommentar(\$layer, "Kaputte Zeile");	2192
\$enderreicht = 1;	2193
}	2194
}	2195
if (!\$enderreicht) {	2196
printf("Keine sinnvollen Daten\n");	2197
}	2198
else {	2199
schliessedatei(\$name);	2200
}	2201
}	2202
}	2203
elseif (\$zeile =~	2204
/^polygon(layer=([0-9]+)?) (farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+)?) (linie	2205
=([0-9\.] +),([a-z0-9!]+)?) (,([a-zaeoue0-9!]+)?) (parameter="\([^"]*\)")?\$/i	2206
)	2207
{	2208
settexthier("");	2209
\$anfang = 0;	2210
standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",	2211
"");	2212
\$layer = \$2 if (defined \$1);	2213
}	2214
if (defined \$3) {	2215
\$farbe = \$4;	2216
\$fuellart = \$5;	2217
\$fuellfarbe = \$6;	2218
}	2219
if (defined \$7) {	2220
\$linienstaerke = \$8;	2221
\$linienart = \$9;	2222
}	2223
\$farbe = \$11 if (defined \$10);	2224
if (defined \$12) {	2225
\$parameter = \$13;	2226
}	2227
else {	2228
\$parameter = "";	2229
}	2230
\$enderreicht = 0;	2231
}	2232
while ((!\$enderreicht)	2233
&& (\$i < \$eingabedateizeilen - 1))	2234
{	2235
\$i++;	2236
\$hilfszeile = \$eingabedatei[\$i];	2237
if (\$hilfszeile =~ /^ende\$/i) {	2238
\$enderreicht = 1;	2239
if (\$anfang == 1) { paar(\$x, \$y); }	2240
else { tripel(\$x, \$y, \$z); }	2241

```

    }
    elseif ( $hilfszeile =~ /^([-0-9\.]+),([-0-9\.]+)$/i ) {
        if ( $anfang == 0 ) {
            $x = $1;
            $y = $2;
            setachsengraph( $x, $y, $x, $y );
            $anfang = 1;
        }
        paar( $1, $2 );
    }

    elseif ( $hilfszeile =~ /^([-0-9\.]+),([-0-9\.]+),([-0-9\.]+)$/i ) {
        if ( $anfang == 0 ) {
            $x = $1;
            $y = $2;
            $z = $3;
            $anfang = 2;
        }
        tripel( $1, $2, $3 );
    }

    else {
        printf("Kaputte Zeile beim Polygon\n");
        $enderreicht = 1;
    }
}
if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
else {

    polygon($layer, $linienart, $linienstaerke, $farbe, $fuellart, $fuellfarbe,
        $parameter, gettexthier() );

    settexthier("");
}
}

elseif ( $zeile =~
/^barplot( layer=([0-9]+)?( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoue0-9!]+)?( parameter="\^[^"]*)" )?( r
=([-0-9\.]+))?$ /i
)
{
    setachsengraph(0, 0, 0, 0 );
    settexthier("");
    $anfang = 0;
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe = $4;
        $fuellart = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart = $9;
    }
}

```

\$farbe = \$11 if (defined \$10);	2301
if (defined \$12) { \$parameter = \$13; }	2302
else { \$parameter = ""; }	2303
	2304
if (defined \$14) { \$breite = \$15 }	2305
else { \$breite = 1; }	2306
	2307
\$enderreicht = 0;	2308
	2309
while ((!\$enderreicht)	2310
&& (\$i < \$eingabedateizeilen - 1))	2311
{	2312
\$i++;	2313
\$hilfszeile = \$eingabedatei[\$i];	2314
if (\$hilfszeile =~ /^ende\$/i) {	2315
\$enderreicht = 1;	2316
}	2317
elsif (\$hilfszeile =~ /^([-0-9\.]+),([-0-9\.]+)\$/i) {	2318
saeule(\$layer, \$1 - \$breite / 2,	2319
0, \$breite, \$2, \$farbe, \$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke,	2320
"", 0, 0, "s", \$parameter);	2321
setachsengraph(\$1, \$2, \$1, \$2);	2322
	2323
# setachsengraph(\$1-\$breite/2, \$2, \$1+\$breite/2, \$2);	2324
}	2325
	2326
else {	2327
printf("Kaputte Zeile beim Balkendiagramm\n");	2328
\$enderreicht = 1;	2329
	2330
}	2331
}	2332
if (!\$enderreicht) { printf("Keine sinnvollen Daten\n"); }	2333
else {	2334
	2335
settexthier ("");	2336
	2337
}	2338
}	2339
	2340
# Ober und Untersumme werden gezeichnet	2341
elseif (\$zeile =~	2342
/^oberuntersumme(layer=([0-9]+))?(farbe=([a-zAEOeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(	2343
linie=([0-9\.]+),([a-z0-9!]+))?(,[a-zAEOeue0-9!]+)? [x p]=([0-9\.]+),([0-9\.]+)([y	
p]=([0-9\.]+),([0-9\.]+))? n=([1-9][0-9]*) (summen=(ober unter beide))?(.*)\$/i	
)	2344
{	2345
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	2346
"");	2347
\$layer = \$2 if (defined \$1);	2348
	2349
if (defined \$3) {	2350
\$farbe = \$4;	2351
\$fuellart = \$5;	2352
\$fuellfarbe = \$6;	2353
}	2354
\$linienstaerke = 0.2;	2355
	2356
if (defined \$7) {	2357
\$linienstaerke = \$8;	2358
\$linienart = \$9;	2359

```

}
$farbe = $11 if ( defined $10 );
$xmin = $12;
$xmax = $13;
if ( defined $14 ) {
    $ymin = $15;
    $ymax = $16;
}
$schachtelung = $17;
$summen = "beide";
$summen = $19 if ( defined $18 );
$func = $20;
if ( !defined $14 ) {
    $ymin = calcymin( $func, $xmin, $xmax );
    $ymax = calcymin( $func, $xmin, $xmax );
}
oberuntersumme(
    $layer,          $func,          $xmin,    $ymin,    $xmax,
    $ymax,          $farbe,          $fuellart, $fuellfarbe, $linienart,
    $linienstaerke, $schachtelung, $summen
);
}

# Zeichnen von Flaechen zwischen zwei Funktionen
elsif ( $zeile =~
/^einschluss( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter=\("[^"]*" )? [x|p
]=([0-9\.\.]+),([0-9\.\.]+)( [y|p]=([0-9\.\.]+),([0-9\.\.]+))?( schnitt=(links|rechts|beide
|keine))?( ([^,]*)(,[^,]*))?$ /i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe = $4;
        $fuellart = $5;
        $fuellfarbe = $6;
    }
    $linienstaerke = 0.2;

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart = $9;
    }
    $farbe = $11 if ( defined $10 );

    if ( defined $12 ) { $parameter = $13; }

    $xmin = $14;
    $xmax = $15;
    if ( defined $16 ) {
        $ymin = $17;
        $ymax = $18;
    }
    $schnittpunkte = "keine";
    $schnittpunkte = $20 if ( defined $19 );
    $func1 = $21;
    $func2 = "0";

```

2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386

2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417

```

$func2      = $23 if ( defined $22 );
if ( !defined $16 ) {
    $ymin = calcfymin( $func1, $func2, $xmin, $xmax );
    $ymax = calcfymax( $func1, $func2, $xmin, $xmax );
}
setminmaxachsen( $xmin, "x", "min" );
setminmaxachsen( $xmax, "x", "max" );
setminmaxachsen( $ymin, "y", "min" );
setminmaxachsen( $ymax, "y", "max" );
einschluss(
    $layer,      $func1,      $func2,      $xmin, $ymin,
    $xmax,      $ymax,      $schnittpunkte, $farbe, $fuellart ,
    $fuellfarbe, $linienart, $linienstaerke, $parameter
);
}

elseif ( $zeile =~
/^einschlussanfang( layer=([0-9]+) )?( farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+) )?(
    linie=([0-9\.] +),([a-z0-9!]+) )?(,[a-zA-Z0-9!]+) )?( parameter="\([^"]*\)" )?$/i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe  = $6;
    }
    $linienstaerke = 0.2;

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    einschlussanfang( $layer, $farbe, $fuellart, $fuellfarbe, $linienart, $linienstaerke,
        $parameter );
}

elseif ( $zeile =~
/^einschlussende( layer=([0-9]+) )?( farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+) )?(
    linie=([0-9\.] +),([a-z0-9!]+) )?(,[a-zA-Z0-9!]+) )?( parameter="\([^"]*\)" )?$/i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe  = $6;
    }
    $linienstaerke = 0.2;

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }

```

<pre> } \$farbe = \$11 if (defined \$10); einschliessende(\$layer, \$farbe, \$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke, \$parameter); } elsif (\$zeile =~ /^achsenpunkt(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))? [x p]=([0-9\.]+) (text =\"([^\"]*)\")? \\$/i) { standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "", ""); \$layer = \$2 if (defined \$1); \$farbe = \$4 if (defined \$3); \$x = \$5; \$text = ""; \$text = \$7 if (defined \$6); if (get3dzeichnen() == 1) { xmarkierung3d(\$layer, \$x, \$text, \$farbe); } else { xmarkierung(\$layer, \$x, \$text, \$farbe); } } elsif (\$zeile =~ /^achsenpunktohne(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))? [x p]=([0-9\.]+) (text =\"([^\"]*)\")? \\$/i) { standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "", ""); \$layer = \$2 if (defined \$1); \$farbe = \$4 if (defined \$3); \$x = \$5; \$text = ""; \$text = \$7 if (defined \$6); if (get3dzeichnen() == 1) { xmarkierung3d(\$layer, \$x, \$text, \$farbe); } else { xmarkierungohne(\$layer, \$x, \$text, \$farbe); } } elsif (\$zeile =~ /^yachsenpunkt(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))? [y p]=([0-9\.]+) (text =\"([^\"]*)\")? \\$/i) { standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "", ""); \$layer = \$2 if (defined \$1); \$farbe = \$4 if (defined \$3); \$y = \$5; \$text = ""; \$text = \$7 if (defined \$6); if (get3dzeichnen() == 1) { ymarkierung3d(\$layer, \$y, \$text, \$farbe); </pre>	<pre> 2477 2478 2479 2480 2481 2482 2483 2484 2485 2486 2487 2488 2489 2490 2491 2492 2493 2494 2495 2496 2497 2498 2499 2500 2501 2502 2503 2504 2505 2506 2507 2508 2509 2510 2511 2512 2513 2514 2515 2516 2517 2518 2519 2520 2521 2522 2523 2524 2525 2526 2527 2528 2529 2530 2531 2532 2533 2534 </pre>
---	--

}	2535
else { ymarkierung(\$layer,\$y, \$text, \$farbe); }	2536
}	2537
	2538
elseif (\$zeile =~	2539
/^achsenpunktohne(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+))?[y p]=([0-9\..]+)(text	2540
=\"([^\"]*)\"?)\"\$/i	2541
)	2542
{	2543
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	2544
"");	2545
\$layer = \$2 if (defined \$1);	2546
\$farbe = \$4 if (defined \$3);	2547
\$y = \$5;	2548
\$text = "";	2549
\$text = \$7 if (defined \$6);	2550
	2551
if (get3dzeichnen() == 1) {	2552
ymarkierung3d(\$layer,\$y,\$text, \$farbe);	2553
}	2554
else { ymarkierungohne(\$layer,\$y,\$text, \$farbe); }	2555
	2556
}	2557
	2558
elseif (\$zeile =~	2559
/^achsenpunkt(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+))? z=([0-9\..]+)(text=\"([^\"]*)	2560
?\"\$/i	
)	2561
{	2562
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	2563
"");	2564
\$layer = \$2 if (defined \$1);	2565
\$farbe = \$4 if (defined \$3);	2566
\$z = \$5;	2567
\$text = "";	2568
\$text = \$7 if (defined \$6);	2569
	2570
zmarkierung3d(\$layer,\$z,\$text, \$farbe);	2571
	2572
}	2573
	2574
elseif (\$zeile =~	2575
/^xpfeil(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+))?(linie=([0-9\..]+)([a-z0-9!]+))? p	2576
=([0-9\..]+)([0-9\..]+))?(text=\"([^\"]*)\"?)\"\$/i	
)	2577
{	2578
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	2579
"");	2580
\$layer = \$2 if (defined \$1);	2581
\$farbe = \$4 if (defined \$3);	2582
\$linienstaerke = 0.2;	2583
	2584
if (defined \$5) {	2585
\$linienstaerke = \$6;	2586
\$linienart = \$7;	2587
}	2588
\$x = \$8;	2589
\$y = 1;	2590
\$y = \$10 if (defined \$9);	2591
\$text = "";	2592

```

$text = $12 if ( defined $11 );
xpfeil( $layer, $x, $y, $text, $farbe, $linienstaerke, $linienart );
}

elsif ( $zeile =~
/^xpfeilkasten( layer=([0-9]+)?)? ( farbe=([a-zaeoeue0-9!]+)?)? ( linie=([0-9\.\.]+),([a-z0-9!]+)?)?
  p=([0-9\.\.]+),([0-9\.\.]+)?)? ( text="\([^"]*\)")? "$/i
)
{
  standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
    "" );
  $layer = $2 if ( defined $1 );
  $farbe = $4 if ( defined $3 );
  $linienstaerke = 0.2;

  if ( defined $5 ) {
    $linienstaerke = $6;
    $linienart = $7;
  }
  $x = $8;
  $y = 1;
  $y = $10 if ( defined $9 );
  $text = "";
  $text = $12 if ( defined $11 );
  xpfeilkasten( $layer, $x, $y, $text, $farbe, $linienstaerke, $linienart );
}

elsif ( $zeile =~
/^ypfeilkasten( layer=([0-9]+)?)? ( farbe=([a-zaeoeue0-9!]+)?)? ( linie=([0-9\.\.]+),([a-z0-9!]+)?)? p
  =([0-9\.\.]+),([0-9\.\.]+)?)? ( text="\([^"]*\)")? "$/i
)
{
  standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
    "" );
  $layer = $2 if ( defined $1 );
  $farbe = $4 if ( defined $3 );
  $linienstaerke = 0.2;

  if ( defined $5 ) {
    $linienstaerke = $6;
    $linienart = $7;
  }
  $y = $10;
  $x = 1;
  $x = $9 if ( defined $8 );
  $text = "";
  $text = $12 if ( defined $11 );
  ypfeilkasten( $layer, $x, $y, $text, $farbe, $linienstaerke, $linienart );
}

elsif ( $zeile =~
/^ypfeilkasten( layer=([0-9]+)?)? ( farbe=([a-zaeoeue0-9!]+)?)? ( linie=([0-9\.\.]+),([a-z0-9!]+)?)?
  p=([0-9\.\.]+),([0-9\.\.]+)?)? ( text="\([^"]*\)")? "$/i
)
{
  standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
    "" );
  $layer = $2 if ( defined $1 );

```

\$farbe = \$4 if (defined \$3);	2651
\$linienstaerke = 0.2;	2652
	2653
if (defined \$5) {	2654
\$linienstaerke = \$6;	2655
\$linienart = \$7;	2656
}	2657
\$y = \$10;	2658
\$x = 1;	2659
\$x = \$9 if (defined \$8);	2660
\$text = "";	2661
\$text = \$12 if (defined \$11);	2662
ypfeilkasten(\$layer, \$x, \$y, \$text, \$farbe, \$linienstaerke, \$linienart);	2663
	2664
}	2665
	2666
elseif (\$zeile =~	2667
/^zpfeil(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.\.]+),([a-z0-9!]+))? p	2668
=([0-9\.\.]+),([0-9\.\.]+))?(text=\"([^\"]*)\")?\$/i	
)	2669
{	2670
\$layer = 90;	2671
\$layer = \$2 if (defined \$1);	2672
\$farbe = \$4 if (defined \$3);	2673
\$linienstaerke = 0.2;	2674
	2675
if (defined \$5) {	2676
\$linienstaerke = \$6;	2677
\$linienart = \$7;	2678
}	2679
\$z = \$8;	2680
\$y = 1;	2681
\$y = \$10 if (defined \$9);	2682
\$text = "";	2683
\$text = \$12 if (defined \$11);	2684
zpfeil(\$layer, \$z, \$y, \$text, \$farbe, \$linienstaerke, \$linienart);	2685
	2686
}	2687
elseif (\$zeile =~	2688
/^graphmarkierung(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.\.]+),([a-z]+),	2689
x y xy))?(parameter=\"([^\"]*)\")? [x p]=([0-9\.\.]+)([y p]=([0-9\.\.]+))?(text	
=\"([^\"]*)\" ,\"([^\"]*)\")? (([^]*))?\$/i	
)	2690
{	2691
standard(10, getlinienart (), getlinienstaerke (), getfarbe (), "", getfuellfarbe (), "",	2692
"");	2693
\$layer = \$2 if (defined \$1);	2694
\$farbe = \$4 if (defined \$3);	2695
\$linienstaerke = 0.2;	2696
	2697
\$linie = "xy";	2698
if (defined \$5) {	2699
\$linienstaerke = \$6;	2700
\$linienart = \$7;	2701
\$linie = \$8;	2702
}	2703
	2704
if (defined \$9) { \$parameter = \$10; }	2705
	2706
\$x = \$11;	2707
\$xtext = "";	2708

```

$ytext = "";
if ( defined $14 ) {
    $xtext = $15;
    $ytext = $16;
}
$func = 0;
$func = $18 if ( defined $17 );
if ( dreidimensional( $func, "y" ) == 0 ) {
    $y = calcfunk( $func, "x", $x );
}
if ( defined $12 ) { $y = $13; }
else { $y = calcfunk( $func, "x", $x ); }

markierung(
    $layer,      $func, $x,    $y,
    $xtext,      $ytext, $farbe, $linienart,
    $linienstaerke, $linie, $parameter
);

}
elseif ( $zeile =~
/^punkt( layer=([0-9!+])?( farbe=([a-zaeoeue0-9!+]),([a-z0-9!+]),([a-z0-9!+])?( linie
=([0-9!\.]+),([a-z0-9!+])?)(,[a-zaeoeue0-9!+])?( parameter="\([^"]*\)"? p
=([0-9!\.]+),([0-9!\.]+),([0-9!\.]+))?( r=([0-9!\.]+))?( text="\([^"]*\)"
)?,([0-9!\.]+))?$ /i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
        "" );
    $layer = $2 if ( defined $1 );

    $fuellart = "solid";
    $fuellfarbe = "black";
    if ( defined $3 ) {
        $farbe = $4;
        $fuellart = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart = $9;
    }
    $farbe = $11 if ( defined $10 );

    if ( defined $12 ) { $parameter = $13; }

    $x = $14;
    $y = $15;
    if ( defined $16 ) { $z = $17; }
    else { $z = ""; }
    $radius = 0.05;
    $radius = $19 if ( defined $18 );
    $text = "";
    $text = $21 if ( defined $20 );
    $ausrichtung = 45;
    $ausrichtung = $23 if ( defined $22 );
    punkzeichnen(
        $layer,      $x,    $y,      $z,      $radius,
        $ausrichtung, $farbe, $fuellart, $fuellfarbe, $linienart,
        $linienstaerke, $text, $parameter

```

);	2767
	2768
}	2769
	2770
elseif (\$zeile =~	2771
/^kreuz(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.]�+),([a-z0-9!]+))?(	2772
parameter=\"([^\"]*)\"?) p=([-0-9\.]�+),([-0-9\.]�+),([-0-9\.]�+)?(r=([-0-9\.]�+))	
? ,([-0-9\.]�+))?(text=\"([^\"]*)\"?)? ,([-0-9\.]�+))?\$ /i	
)	2773
{	2774
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	2775
"");	2776
\$layer = \$2 if (defined \$1);	2777
if (defined \$3) {	2778
\$farbe = \$4;	2779
}	2780
	2781
if (defined \$5) {	2782
\$linienstaerke = \$6;	2783
\$linienart = \$7;	2784
}	2785
if (defined \$8) { \$parameter = \$9; }	2786
	2787
\$x = \$10;	2788
\$y = \$11;	2789
if (defined \$12) { \$z = \$13; }	2790
else { \$z = ""; }	2791
\$radius = 0.5;	2792
\$radius = \$15 if (defined \$14);	2793
\$winkel = 45;	2794
\$winkel = \$17 if (defined \$16);	2795
\$text = "";	2796
\$text = \$19 if (defined \$18);	2797
\$ausrichtung = 45;	2798
\$ausrichtung = \$21 if (defined \$20);	2799
	2800
kreuzzeichnen(	2801
\$layer, \$x, \$y, \$z,	2802
\$radius, \$winkel, \$farbe, \$linienart ,	2803
\$linienstaerke, \$text, \$ausrichtung, \$parameter	2804
);	2805
	2806
}	2807
	2808
elseif (\$zeile =~	2809
/^kreuz(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.]�+),([a-z0-9!]+))?(	2810
parameter=\"([^\"]*)\"?) p=([-0-9\.]�+),\"([^\"]*)\"(r=([-0-9\.]�+))?(,([-0-9\.]�+))?(text	
=\"([^\"]*)\"?)? ,([-0-9\.]�+))?\$ /i	
)	2811
{	2812
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	2813
"");	2814
\$layer = \$2 if (defined \$1);	2815
	2816
if (defined \$3) {	2817
\$farbe = \$4;	2818
}	2819
	2820
if (defined \$5) {	2821
\$linienstaerke = \$6;	2822
\$linienart = \$7;	2823

```

}
if ( defined $8 ) { $parameter = $9; }

$х      = $10;
$func   = $11;
$radius = 0.5;
$radius = $13 if ( defined $12 );
$winkel = 45;
$winkel = $15 if ( defined $14 );
$text   = "";
$text   = $17 if ( defined $16 );
$ausrichtung = 45;
$ausrichtung = $19 if ( defined $18 );

kreuzzeichnen( $layer, $х, calcfunk( $func, "x", $х ),
    "", $radius, $winkel, $farbe, $linienart, $linienstaerke, $text, $ausrichtung,
    $parameter );
}

elseif ( $zeile =~
/^punkt( layer=([0-9!+])?( farbe=([a-zaeoeue0-9!+]),([a-z0-9!+]),([a-z0-9!+])?( linie
=([0-9!.+]),([a-z0-9!+])?(,([a-zaeoeue0-9!+])?( parameter="\("[^"]"*)\")? p
=([0-9!.+]),\("[^"]"*)\")( r=([0-9!.+])?( text="\("[^"]"*)\")(,([0-9!.+])?)$/i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
        "" );
    $layer = $2 if ( defined $1 );

    $fuellart = "solid";
    $fuellfarbe = "black";
    if ( defined $3 ) {
        $farbe = $4;
        $fuellart = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart = $9;
    }
    $farbe = $11 if ( defined $10 );

    if ( defined $12 ) { $parameter = $13; }

    $х = $14;
    $func = $15;

    $radius = 0.05;
    $radius = $17 if ( defined $16 );
    $text = "";
    $text = $19 if ( defined $18 );
    $ausrichtung = 45;
    $ausrichtung = $21 if ( defined $20 );
    punktzeichnen( $layer, $х, calcfunk( $func, "x", $х ),
        "", $radius, $ausrichtung, $farbe, $fuellart, $fuellfarbe, $linienart,
        $linienstaerke, $text, $parameter );
}

```

elseif (\$zeile =~	2883
/^beschriftung(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+))? p=([-0-9\.]+),\"([^\"]*)\" text	2884
=\"([^\"]*)\" \"([0-9\.]+))?(,[a-z0-9!]+))?(,[0-9\.]+))?\$ /i	
)	2885
{	2886
\$layer = 10;	2887
\$layer = \$2 if (defined \$1);	2888
\$farbe = "black";	2889
\$farbe = \$4 if (defined \$3);	2890
\$x = \$5;	2891
\$func = \$6;	2892
\$text = \$7;	2893
\$winkel = 0;	2894
\$ausrichtung = "c";	2895
\$dy = 0;	2896
	2897
\$winkel = \$9 if (defined \$8);	2898
\$ausrichtung = \$11 if (defined \$10);	2899
\$dy = \$13 if (defined \$12);	2900
beschriftung(\$layer, \$farbe, \$x, calcfunk(\$func, "x", \$x) + \$dy,	2901
\$winkel, \$ausrichtung, \$text);	2902
}	2903
	2904
elseif (\$zeile =~	2905
/^beschriftung(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+))? p=([-0-9\.]+),([0-9\.]+)	2906
,([0-9\.]+) text=\"([^\"]*)\" \"([0-9\.]+))?(,[a-z0-9!]+))?\$ /i	2907
)	2908
{	2909
standard(10, getlinienart (), getlinienstaerke (), getfarbe (), "", getfuellfarbe (), "",	2910
"");	2911
\$layer = \$2 if (defined \$1);	2912
	2913
\$farbe = \$4 if (defined \$3);	2914
\$x = \$5;	2915
\$y = \$6;	2916
\$z = \$7;	2917
\$text = \$8;	2918
	2919
\$winkel = 0;	2920
\$ausrichtung = "c";	2921
\$winkel = \$10 if (defined \$9);	2922
\$ausrichtung = \$12 if (defined \$11);	2923
beschriftung3d(\$layer, \$farbe, \$x, \$y, \$z, \$winkel, \$ausrichtung, \$text);	2924
	2925
}	2926
	2927
elseif (\$zeile =~	2928
/^beschriftung(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+))? p=([-0-9\.]+),([0-9\.]+) text	2929
=\"([^\"]*)\" \"([0-9\.]+))?(,[a-z0-9!]+))?\$ /i	
)	2930
{	2931
\$layer = 10;	2932
\$layer = \$2 if (defined \$1);	2933
\$farbe = "black";	2934
\$farbe = \$4 if (defined \$3);	2935
\$x = \$5;	2936
\$y = \$6;	2937
\$text = \$7;	2938
\$winkel = 0;	2939
\$ausrichtung = "c";	2940

```

$winkel      = $9 if ( defined $8 );
$ausrichtung = $11 if ( defined $10 );
beschriftung( $layer, $farbe, $x, $y, $winkel, $ausrichtung, $text );
}
2941
2942
2943
2944
2945
2946
2947
elseif ( $zeile =~
/^beschriftungskreis( layer=([0-9]+) )?( farbe=([a-zaeoue0-9!]+) )? p=([0-9\.]+),([0-9\.]+)
,([0-9\.]+),([0-9\.]+) text="\([^"]*\)"([0-9\.]+),([a-z0-9!]+) )?$/i
)
2948
2949
{
2950
$layer      = 10;
2951
$layer      = $2 if ( defined $1 );
2952
$farbe      = "black";
2953
$farbe      = $4 if ( defined $3 );
2954
$x          = $5;
2955
$y          = $6;
2956
$radius     = $7;
2957
$winkel     = $8;
2958
$text      = $9;
2959
$winkelrichtung = 0;
2960
$ausrichtung = "c";
2961
if ( defined $10 ) {
2962
$winkelrichtung = $11;
2963
$ausrichtung    = $12;
2964
}
2965
2966
kreistext ( $layer, $x, $y, $radius, $winkel, $farbe, $text, $winkelrichtung,
$ausrichtung );
2967
2968
}
2969
2970
2971
elseif ( $zeile =~
/^tangente( layer=([0-9]+) )?( farbe=([a-zaeoue0-9!]+) )?( linie=([0-9\.]+),([a-z0-9!]+) )?(
parameter="\([^"]*\)" )? [x|p]=([0-9\.]+)( dx=([0-9\.]+) )? (.)$/i
)
2972
2973
{
2974
2975
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
2976
2977
$layer = $2 if ( defined $1 );
2978
$farbe = $4 if ( defined $3 );
2979
2980
if ( defined $5 ) {
2981
$linienstaerke = $6;
2982
$linienart     = $7;
2983
}
2984
2985
if ( defined $8 ) { $parameter = $9; }
2986
2987
2988
$x      = $10;
2989
$dx     = 0.5;
2990
$dx     = $12 if ( defined $11 );
2991
$func   = $13;
2992
tangente( $layer, $func, $x, $dx, $farbe, $linienart, $linienstaerke, $parameter );
2993
2994
}
2995
elseif ( $zeile =~
/^tangendentreieck( layer=([0-9]+) )?( farbe=([a-zaeoue0-9!]+) )?( linie=([0-9\.]+),([a-z0
-9!]+) )?( parameter="\([^"]*\)" )? [x|p]=([0-9\.]+)( dx=([0-9\.]+) )? (.)$/i
)
2996
2997
{
2998

```

standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	2999
"");	3000
\$layer = \$2 if (defined \$1);	3001
\$farbe = \$4 if (defined \$3);	3002
	3003
if (defined \$5) {	3004
\$linienstaerke = \$6;	3005
\$linienart = \$7;	3006
}	3007
if (defined \$8) { \$parameter = \$9; }	3008
	3009
\$x = \$10;	3010
\$dx = 0.5;	3011
\$dx = \$12 if (defined \$11);	3012
\$func = \$13;	3013
kommentar(\$layer, "Ein Tangentendreieck wird gezeichnet");	3014
tangentendreieck(\$layer, \$func, \$x, \$dx, \$farbe, \$linienart, \$linienstaerke,	3015
\$parameter);	3016
	3017
}	3018
	3019
elseif (\$zeile =~	3020
/^sekante(layer=([0-9]+))?(farbe=([a-zAEOUE0-9!]+))?(linie=([0-9\.]�+)([a-z0-9!]+))?(	3021
parameter="\([^\""]*)\""? [x p]=([0-9\.]�+)(dx=([0-9\.]�+)? (.)\$ /i	
)	3022
{	3023
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	3024
"");	3025
\$layer = \$2 if (defined \$1);	3026
\$farbe = \$4 if (defined \$3);	3027
	3028
if (defined \$5) {	3029
\$linienstaerke = \$6;	3030
\$linienart = \$7;	3031
}	3032
	3033
if (defined \$8) { \$parameter = \$9; }	3034
	3035
\$x = \$10;	3036
\$dx = 0.5;	3037
\$dx = \$12 if (defined \$11);	3038
\$func = \$13;	3039
sekante(\$layer, \$func, \$x, \$dx, \$farbe, \$linienart, \$linienstaerke, \$parameter);	3040
	3041
}	3042
	3043
elseif (\$zeile =~	3044
/^sekantendreieck(layer=([0-9]+))?(farbe=([a-zAEOUE0-9!]+))?(linie=([0-9\.]�+)([a-z0	3045
-9!]+))?(parameter="\([^\""]*)\""? [x p]=([0-9\.]�+)(dx=([0-9\.]�+)? (.)\$ /i	
)	3046
{	3047
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	3048
"");	3049
\$layer = \$2 if (defined \$1);	3050
\$farbe = \$4 if (defined \$3);	3051
	3052
if (defined \$5) {	3053
\$linienstaerke = \$6;	3054
\$linienart = \$7;	3055
}	3056
if (defined \$8) { \$parameter = \$9; }	3057

```

$х      = $10;
$dx     = 0.5;
$dx     = $12 if ( defined $11 );
$func   = $13;
sekantendreieck( $layer, $func, $х, $dx, $farbe, $linienart, $linienstaerke,
    $parameter );

}
elseif ( $zeile =~
/^saeule( layer=([0-9]+) )?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+) )?( linie
=([0-9\..]+),([a-z0-9!]+) )?( ([a-zaeoeue0-9!]+) )?( parameter="\([^"]*\)" )? p
=([0-9\..]+),([0-9\..]+) r=([0-9\..]+),([0-9\..]+) )?( text="\([^"]*\)" )?( ausrichtung
=([0-9\..]+),([a-z]+) | ([0-9\..]+) ) )?$/i
)
{
    setsaeule();
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe = $6;
    }
    $linienstaerke = 0.2;

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $х      = $14;
    $у      = $15;
    $dx     = $16;
    $dy     = $17;
    $text    = "";
    $text    = $19 if ( defined $18 );
    $textwinkel = 0;
    $textausrichtung = "auf";

    if ( defined $20 ) {
        $textwinkel = $21;
        $textausrichtung = $22;
    }
    saeule( $layer, $х, $у, $dx, $dy, $farbe, $fuellart, $fuellfarbe, $linienart,
        $linienstaerke, $text, $textwinkel, $textausrichtung, "s", $parameter );
}
elseif ( $zeile =~
/^balken( layer=([0-9]+) )?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+) )?( linie
=([0-9\..]+),([a-z0-9!]+) )?( ([a-zaeoeue0-9!]+) )?( parameter="\([^"]*\)" )? p
=([0-9\..]+),([0-9\..]+) r=([0-9\..]+),([0-9\..]+) )?( text="\([^"]*\)" )?( ausrichtung
=([0-9\..]+),([a-z]+) | ([0-9\..]+) ) )?$/i
)
{
    setbalken();
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",

```

"");	3113
\$layer = \$2 if (defined \$1);	3114
	3115
if (defined \$3) {	3116
\$farbe = \$4;	3117
\$fuellart = \$5;	3118
\$fuellfarbe = \$6;	3119
}	3120
\$linienstaerke = 0.2;	3121
	3122
if (defined \$7) {	3123
\$linienstaerke = \$8;	3124
\$linienart = \$9;	3125
}	3126
\$farbe = \$11 if (defined \$10);	3127
	3128
if (defined \$12) { \$parameter = \$13; }	3129
	3130
\$x = \$14;	3131
\$y = \$15;	3132
\$dy = \$16;	3133
\$dx = \$17;	3134
\$text = "";	3135
\$text = \$19 if (defined \$18);	3136
\$textwinkel = 0;	3137
\$textausrichtung = "auf";	3138
	3139
if (defined \$20) {	3140
\$textwinkel = \$21;	3141
\$textausrichtung = \$22;	3142
}	3143
saeule(\$layer, \$x, \$y, \$dx, \$dy, \$farbe, \$fuellart, \$fuellfarbe, \$linienart,	3144
\$linienstaerke, \$text, \$textwinkel, \$textausrichtung, "b", \$parameter);	3145
	3146
}	3147
	3148
# Zeichnen von mehreren Saehlen	3149
elsif (\$zeile =~	3150
/^saeulendiagramm(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(	3151
linie=([0-9\.\.]+),([a-z0-9!]+))?(,([a-zA-Z0-9!]+))?(parameter="\([^"]*\)")?(p	
=([0-9\.\.]+),([0-9\.\.]+))?(r=([0-9\.\.]+),([0-9\.\.]+))?(ausrichtung=([0-9\.\.]+),([a-z	
]+) ([0-9\.\.]+))?)?\$ /i	
)	3152
{	3153
\$n = \$j = 0;	3154
setsaeule();	3155
	3156
standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",	3157
"");	3158
\$layer = \$2 if (defined \$1);	3159
	3160
if (defined \$3) {	3161
\$farbe = \$4;	3162
\$fuellart = \$5;	3163
\$fuellfarbe = \$6;	3164
}	3165
\$linienstaerke = 0.2;	3166
	3167
if (defined \$7) {	3168
\$linienstaerke = \$8;	3169
\$linienart = \$9;	3170

```

}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$x = 0;
$y = 0;
if ( defined $14 ) {
    $x = $15;
    $y = $16;
}
$dx      = 1;
$abstand = 1;
if ( defined $17 ) {
    $dx      = $18;
    $abstand = $19;
}
$textwinkel      = 90;
$textausrichtung = "auf";
if ( defined $20 ) {
    $textwinkel      = $21;
    $textausrichtung = $22;
}
$x2      = $x1;
$y2      = $y1;
$enderreicht = 0;
$inzeile   = 0;
$merkei    = $i;
while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
{
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) {
        $enderreicht = 1;
    }
    elsif ( $hilfszeile =~
/^[(-0-9\.]+)( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\.]+),([a-
z0-9!]+))?(,[a-zaeoue0-9!]+))?( parameter="\([^"]*" )?( r=([(-0-9\.]+),([(-0-9\.]+))?(
text="\([^"]*" )?( ausrichtung=([0-9\.]+),([([a-z]+) | ([(-0-9\.]+)))?$/i
)
    {
        if ( getrelativ () == 1 ) {
            $absolut += $1;
            $j++;
            setmarkierung( "Prozent", "y" );
        }
        else {
            $absolut = $absolut;
            $j++;
        }
    }
    else {
        kommentar( $layer, "Kaputte Zeile" );
        $enderreicht = 1;
    }
}
}

if ( $absolut == 1 ) { $absolut = $absolut; }
else { $absolut = $absolut / 100; }

```

\$enderreicht = 0;	3230
\$inzeile = 0;	3231
\$i = \$merkei;	3232
	3233
while ((!\$enderreicht)	3234
&& (\$i < \$eingabedateizeilen - 1))	3235
{	3236
\$i++;	3237
\$hilfszeile = \$eingabedatei[\$i];	3238
if (\$hilfszeile =~ /^ende\$/i) {	3239
\$enderreicht = 1;	3240
}	3241
elseif (\$hilfszeile =~	3242
/^[-0-9\.] + (farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie=([0-9\.] +),([a-	3243
z0-9!]+))?(,[a-zaeoue0-9!]+)?(parameter="\([^\"]*\)")?(r=([-0-9\.] +),([-0-9\.] +))?(	3244
text="\([^\"]*\)")?(ausrichtung=([0-9\.] +),((([a-z]+) ([-0-9\.] +))))?\$/i	
)	3245
{	3246
\$dy = \$1;	3247
\$farbeersatz = \$farbe;	3248
\$fuellartersatz = \$fuellart ;	3249
\$fuellfarbeersatz = \$fuellfarbe ;	3250
\$parameterersatz = \$parameter;	3251
if (defined \$2) {	3252
\$farbeersatz = \$3;	3253
\$fuellartersatz = \$4;	3254
\$fuellfarbeersatz = \$5;	3255
}	3256
if (\$farbverlauf == 1) {	3257
\$fuellfarbeersatz = farbverlauf(\$fuellfarbe , \$i);	3258
}	3259
else { \$fuellfarbeersatz = \$fuellfarbeersatz ; }	3260
	3261
\$linienstaerkeersatz = \$linienstaerke;	3262
\$linienartersatz = \$linienart ;	3263
if (defined \$6) {	3264
\$linienstaerkeersatz = \$7;	3265
\$linienartersatz = \$8;	3266
}	3267
\$farbe = \$10 if (defined \$9);	3268
if (defined \$11) { \$parameterersatz = \$12; }	3269
	3270
\$dxersatz = \$dx;	3271
\$abstandersatz = \$abstand;	3272
if (defined \$13) {	3273
\$dxersatz = \$14;	3274
\$abstandersatz = \$15;	3275
}	3276
\$text = "";	3277
\$text = \$17 if (defined \$16);	3278
\$textwinkersatz = \$textwinkel;	3279
\$textausrichtungersatz = \$textausrichtung;	3280
if (defined \$18) {	3281
\$textwinkersatz = \$19;	3282
\$textausrichtungersatz = \$20;	3283
}	3284
\$inzeile++;	3285
saeule(	3286
\$lager, \$x, \$y,	3287
\$dxersatz, \$dy / \$absolut, \$farbeersatz,	3288

```

        $fuellartersatz ,      $fuellfarbersatz , $linienartersatz ,
        $linienstaerkeersatz , $text,      $textwinkelpersatz,
        $textausrichtungersatz, "s",      $parameterersatz
    );
    $x = $x + $dxersatz + $abstandersatz;
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;

}
}
}

if ( !$enderreicht ) {
    printf( "%Keine sinnvollen Daten\n" );
}
}

# Zeichnen von mehreren Saeulen
elsif ( $zeile =~
/^saeulendiagramm( layer=([0-9]+) )?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+) )?(
    linie=([0-9\..]+),([a-z0-9!]+) )?(,([a-zaeoeue0-9!]+) )?( parameter="\([^\"]*\)" )?( r
    =([0-9\..]+) )?( ausrichtung=([0-9\..]+),([a-z]+) ! ([0-9\..]+) ) )?$/i
)
{
    $n = $j = 0;
    setsaeule();

    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe  = $6;
    }
    $linienstaerke = 0.2;

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $breite = 1;
    if ( defined $14 ) {
        $breite = $15;
    }
    $textwinkel      = 90;
    $textausrichtung = "auf";
    if ( defined $16 ) {
        $textwinkel      = $17;
        $textausrichtung = $18;
    }
    $enderreicht = 0;
    $inzeile      = 0;
    $merkei       = $i;
    while ( ( !$enderreicht )
        && ( $i < $eingabedateizeilen - 1 ) )

```

{	3348
\$i++;	3349
\$hilfszeile = \$eingabedatei[\$i];	3350
if (\$hilfszeile =~ /^ende\$/i) {	3351
\$endeerreicht = 1;	3352
}	3353
elsif (\$hilfszeile =~	3354
/^[0-9\.\.]+)([0-9\.\.]+)(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	3355
=([0-9\.\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\^[^"]*")?(r	3356
=([0-9\.\.]+))?(text="\^[^"]*")?(ausrichtung=([0-9\.\.]+),([a-z]+) ([0-9\.\.]+))?)?\$/i	
)	3357
{	3358
if (getrelativ () == 1) {	3359
\$absolut += \$2;	3360
\$j++;	3361
setmarkierung("Prozent", "y");	3362
}	3363
else {	3364
\$absolut = \$absolut;	3365
\$j++;	3366
}	3367
}	3368
else {	3369
kommentar(\$layer, "Kaputte Zeile");	3370
\$endeerreicht = 1;	3371
}	3372
}	3373
	3374
	3375
if (\$absolut == 1) { \$absolut = \$absolut; }	3376
else { \$absolut = \$absolut / 100; }	3377
\$endeerreicht = 0;	3378
\$inzeile = 0;	3379
\$i = \$merkei;	3380
	3381
while ((!\$endeerreicht)	3382
&& (\$i < \$eingabedateizeilen - 1))	3383
{	3384
\$i++;	3385
\$hilfszeile = \$eingabedatei[\$i];	3386
if (\$hilfszeile =~ /^ende\$/i) {	3387
\$endeerreicht = 1;	3388
}	3389
elsif (\$hilfszeile =~	3390
/^[0-9\.\.]+)([0-9\.\.]+)(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	3391
=([0-9\.\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\^[^"]*")?(r	3392
=([0-9\.\.]+))?(text="\^[^"]*")?(ausrichtung=([0-9\.\.]+),([a-z]+) ([0-9\.\.]+))?)?\$/i	
)	3393
{	3394
\$x = \$1;	3395
\$dy = \$2;	3396
\$farbeersatz = \$farbe;	3397
\$fuellartersatz = \$fuellart ;	3398
\$fuellfarbeersatz = \$fuellfarbe ;	3399
\$parameterersatz = \$parameter;	3400
if (defined \$3) {	3401
\$farbeersatz = \$4;	3402
\$fuellartersatz = \$5;	3403
\$fuellfarbeersatz = \$6;	3404

```

    }
    if ( $farbverlauf == 1 ) {
        $fuellfarbeersatz = farbverlauf( $fuellfarbe, $i );
    }
    else { $fuellfarbeersatz = $fuellfarbeersatz ; }

    $linienstaerkeersatz = $linienstaerke;
    $linienartersatz     = $linienart ;
    if ( defined $7 ) {
        $linienstaerkeersatz = $8;
        $linienartersatz     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameterersatz = $13; }

    $breiteersatz = $breite;
    if ( defined $14 ) {
        $breiteersatz = $15;
    }
    $text          = "";
    $text          = $17 if ( defined $16 );
    $textwinklersatz = $textwinkel;
    $textausrichtungsersatz = $textausrichtung;
    if ( defined $18 ) {
        $textwinklersatz = $19;
        $textausrichtungsersatz = $20;
    }
    $inzeile++;
    saeule(
        $layer,          $x - $breiteersatz / 2,
        0,               $breiteersatz ,
        $dy / $absolut,  $farbeersatz ,
        $fuellartersatz , $fuellfarbeersatz ,
        $linienartersatz , $linienstaerkeersatz ,
        $text,           $textwinklersatz,
        $textausrichtungsersatz, "s",
        $parameterersatz
    );
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $senderreicht = 1;

}
}

if ( !$senderreicht ) {
    printf("%Keine sinnvollen Daten\n");
}

elseif ( $zeile =~
/^saeulendiagrammdreieck( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9!]+),([a-z0-9!]+))?(,[a-zA-Z0-9!]+)?( parameter="\([^\)]*\)" )
?( p=([0-9!]+),([0-9!]+))?( r=([0-9!]+),([0-9!]+))?( ausrichtung=([0-9!]+),(( [a-z]+ ) | ([0-9!]+)))?$/i
)
{
    setsaeule();
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
}

```

\$layer = \$2 if (defined \$1);	3463
	3464
if (defined \$3) {	3465
\$farbe = \$4;	3466
\$fuellart = \$5;	3467
\$fuellfarbe = \$6;	3468
}	3469
\$linienstaerke = 0.2;	3470
	3471
if (defined \$7) {	3472
\$linienstaerke = \$8;	3473
\$linienart = \$9;	3474
}	3475
\$farbe = \$11 if (defined \$10);	3476
if (defined \$12) { \$parameter = \$13; }	3477
	3478
\$x = 0;	3479
\$y = 0;	3480
if (defined \$14) {	3481
\$x = \$15;	3482
\$y = \$16;	3483
}	3484
\$dx = 1;	3485
\$abstand = 1;	3486
if (defined \$17) {	3487
\$dx = \$18;	3488
\$abstand = \$19;	3489
}	3490
\$textwinkel = 90;	3491
\$textausrichtung = "auf";	3492
if (defined \$20) {	3493
\$textwinkel = \$21;	3494
\$textausrichtung = \$22;	3495
}	3496
\$x2 = \$x1;	3497
\$y2 = \$y1;	3498
\$enderreicht = 0;	3499
\$inzeile = 0;	3500
while ((!\$enderreicht)	3501
&& (\$i < \$eingabedateizeilen - 1))	3502
{	3503
\$i++;	3504
\$hilfszeile = \$eingabedatei[\$i];	3505
if (\$hilfszeile =~ /^ende\$/i) {	3506
\$enderreicht = 1;	3507
	3508
}	3509
elsif (\$hilfszeile =~	3510
/^([0-9\.])(farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie=([0-9\.])([a-z0-9!]+))?(,[a-zaeoue0-9!]+)?(parameter="\([^"]*")?(r=([0-9\.])([0-9\.])?(	3511
text="\([^"]*")?(ausrichtung=([0-9\.])(,[a-z]+) ([0-9\.])))?\$/i	
)	3512
{	3513
\$dy = \$1;	3514
\$farbeersatz = \$farbe;	3515
\$fuellartersatz = \$fuellart;	3516
\$fuellfarbeersatz = \$fuellfarbe;	3517
\$parameterersatz = \$parameter;	3518
if (defined \$2) {	3519
\$farbeersatz = \$3;	3520
\$fuellartersatz = \$4;	3521

```

    $fuellfarbersatz = $5;
}
$linienstaerkeersatz = $linienstaerke;
$linienartersatz = $linienart;
if ( defined $6 ) {
    $linienstaerkeersatz = $7;
    $linienartersatz = $8;
}
$farbe = $10 if ( defined $9 );
if ( defined $11 ) { $parameterersatz = $12; }

$dxersatz = $dx;
$abstandersatz = $abstand;
if ( defined $13 ) {
    $dxersatz = $14;
    $abstandersatz = $15;
}
$text = "";
$text = $17 if ( defined $16 );
$textwinklersatz = $textwinkel;
$textausrichtungersatz = $textausrichtung;
if ( defined $18 ) {
    $textwinklersatz = $19;
    $textausrichtungersatz = $20;
}
$inzeile++;
saeuledreieck(
    $layer,          $x,          $y,
    $dxersatz,       $dy,         $farbersatz,
    $fuellartersatz , $fuellfarbersatz , $linienartersatz ,
    $linienstaerkeersatz , $text,   $textwinklersatz,
    $textausrichtungersatz, "s",    $parameterersatz
);
$x = $x + $dxersatz + $abstandersatz;
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;
}
}
}
if ( !$enderreicht ) { printf("%Keine sinnvollen Daten\n"); }
}

elseif ( $zeile =~
/^saeulendiagrammdreieck( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\.\.]+),([a-z0-9!]+))?(,[a-zA-Z0-9!]+))?( parameter="\([^"]*\)" )
?( r=([0-9\.\.]+))?( ausrichtung=([0-9\.\.]+),([a-z]+)|([0-9\.\.]+))?)$/i
)
{
    $n = $j = 0;
    setsaeule();

    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe = $4;
        $fuellart = $5;
        $fuellfarbe = $6;

```

}	3581
\$linienstaerke = 0.2;	3582
	3583
if (defined \$7) {	3584
\$linienstaerke = \$8;	3585
\$linienart = \$9;	3586
}	3587
\$farbe = \$11 if (defined \$10);	3588
if (defined \$12) { \$parameter = \$13; }	3589
	3590
\$breite = 1;	3591
if (defined \$14) {	3592
\$breite = \$15;	3593
}	3594
\$textwinkel = 90;	3595
\$textausrichtung = "auf";	3596
if (defined \$16) {	3597
\$textwinkel = \$17;	3598
\$textausrichtung = \$18;	3599
}	3600
\$enderreicht = 0;	3601
\$inzeile = 0;	3602
\$merkei = \$i;	3603
while ((!\$enderreicht)	3604
&& (\$i < \$eingabedateizeilen - 1))	3605
{	3606
\$i++;	3607
\$hilfszeile = \$eingabedatei[\$i];	3608
if (\$hilfszeile =~ /^ende\$/i) {	3609
\$enderreicht = 1;	3610
	3611
}	3612
elsif (\$hilfszeile =~	3613
/^([0-9\\.]+),([0-9\\.]+)(farbe=([a-zAEOeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	3614
=([0-9\\.]+),([a-z0-9!]+))?(,([a-zAEOeue0-9!]+))?(parameter="\([^"]*\)")?(r	
=([0-9\\.]+))?(text="\([^"]*\)")?(ausrichtung=([0-9\\.]+),([a-z]+) ([0-9\\.]+))?)?\$/i	
)	3615
{	3616
if (getrelativ () == 1) {	3617
\$absolut += \$2;	3618
\$j++;	3619
setmarkierung("Prozent", "y");	3620
}	3621
else {	3622
\$absolut = \$absolut;	3623
\$j++;	3624
}	3625
}	3626
else {	3627
kommentar(\$layer, "Kaputte Zeile");	3628
\$enderreicht = 1;	3629
	3630
}	3631
}	3632
	3633
if (\$absolut == 1) { \$absolut = \$absolut; }	3634
else { \$absolut = \$absolut / 100; }	3635
\$enderreicht = 0;	3636
\$inzeile = 0;	3637
\$i = \$merkei;	3638
	3639

```

while ( ( !$endeerreicht )
  && ( $i < $eingabedateizeilen - 1 ) )
{
  $i++;
  $hilfszeile = $eingabedatei[$i];
  if ( $hilfszeile =~ /^ende$/i ) {
    $endeerreicht = 1;
  }
  elsif ( $hilfszeile =~
/^[(-0-9\\.]+)([(-0-9\\.]+)( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter=\\"([^\"]*)\\")?( r
=([(-0-9\\.]+))?( text=\\"([^\"]*)\\")?( ausrichtung=([0-9\\.]+),([a-z]+)|([(-0-9\\.]+)))?$/i
)
{
  $x          = $1;
  $dy         = $2;
  $farbeersatz = $farbe;
  $fuellartersatz = $fuellart ;
  $fuellfarbeersatz = $fuellfarbe ;
  $parameterersatz = $parameter;
  if ( defined $3 ) {
    $farbeersatz = $4;
    $fuellartersatz = $5;
    $fuellfarbeersatz = $6;
  }
  if ( $farbverlauf == 1 ) {
    $fuellfarbeersatz = farbverlauf( $fuellfarbe , $i );
  }
  else { $fuellfarbeersatz = $fuellfarbeersatz ; }

  $linienstaerkeersatz = $linienstaerke;
  $linienartersatz = $linienart ;
  if ( defined $7 ) {
    $linienstaerkeersatz = $8;
    $linienartersatz = $9;
  }
  $farbe = $11 if ( defined $10 );
  if ( defined $12 ) { $parameterersatz = $13; }

  $breiteersatz = $breite ;
  if ( defined $14 ) {
    $breiteersatz = $15;
  }
  $text          = "";
  $text          = $17 if ( defined $16 );
  $textwinkersatz = $textwinkel;
  $textausrichtungsersatz = $textausrichtung;
  if ( defined $18 ) {
    $textwinkersatz = $19;
    $textausrichtungsersatz = $20;
  }
  $inzeile++;
  saeuledreieck(
    $layer,          $x - $breiteersatz / 2,
    0,              $breiteersatz ,
    $dy / $absolut, $farbeersatz ,
    $fuellartersatz , $fuellfarbeersatz ,
    $linienartersatz , $linienstaerkeersatz ,
    $text,          $textwinkersatz ,
    $textausrichtungsersatz, "s",

```

\$parameterersatz	3699
);	3700
}	3701
else {	3702
kommentar(\$layer, "Kaputte Zeile");	3703
\$endeerreicht = 1;	3704
}	3705
}	3706
}	3707
}	3708
if (!\$endeerreicht) {	3709
printf("%Keine sinnvollen Daten\n");	3710
}	3711
}	3712
}	3713
elseif (\$zeile =~	3714
/^saeulendiagrammgestapelt(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie=([0-9\.\.]+),([a-z0-9!]+))?(,[a-zA-Z0-9!]+))?(parameter="\([^"]*\)"	3715
? (p=([0-9\.\.]+),([0-9\.\.]+))?(r=([0-9\.\.]+),([0-9\.\.]+))?(ausrichtung=([0-9\.\.]+),([a-	
-z]+) ([0-9\.\.]+))?\$ /i	
)	3716
{	3717
setsaeule();	3718
standard(10, getlinienart(), 0.2, getfarbe(), "", getfuellfarbe(), "", "");	3719
\$layer = \$2 if (defined \$1);	3720
}	3721
if (defined \$3) {	3722
\$farbe = \$4;	3723
\$fuellart = \$5;	3724
\$fuellfarbe = \$6;	3725
}	3726
}	3727
if (defined \$7) {	3728
\$linienstaerke = \$8;	3729
\$linienart = \$9;	3730
}	3731
\$farbe = \$11 if (defined \$10);	3732
if (defined \$12) { \$parameter = \$13; }	3733
}	3734
\$х = 0;	3735
\$у = 0;	3736
if (defined \$14) {	3737
\$х = \$15;	3738
\$у = \$16;	3739
}	3740
\$dx = 1;	3741
\$hoehe = 10;	3742
if (defined \$17) {	3743
\$dx = \$18;	3744
\$hoehe = \$19;	3745
}	3746
\$textwinkel = 90;	3747
\$textausrichtung = "auf";	3748
if (defined \$20) {	3749
\$textwinkel = \$21;	3750
\$textausrichtung = \$22;	3751
}	3752
\$х2 = \$х1;	3753
\$у2 = \$у1;	3754
\$endeerreicht = 0;	3755
\$absolutwert = \$n = \$j = 0;	3756

```

$inzeile      = 0;
$merkei       = $i;
while ( ( !$enderreicht )
  && ( $i < $eingabedateizeilen - 1 ) )
{
  $i++;
  $hilfszeile = $eingabedatei[$i];
  if ( $hilfszeile =~ /^ende$/i ) {
    $enderreicht = 1;
  }
  elsif ( $hilfszeile =~
/^([-0-9\\.]+)( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\\.]+),([a-
z0-9!]+))?(,[a-za0-9!]+))?( parameter="\([^\\"]*\)"?( text="\([^\\"]*\)"?( ausrichtung
=([0-9\\.]+),([a-z]+) | ([0-9\\.]+)))?$/i
  )
  {
    if ( getrelativ () == 1 ) {
      $absolutwert = $absolutwert + $1;
      setmarkierung( "Prozent", "y" );
      $j++;
    }
    else {
      $absolutwert = 1;
      $j++;
    }
  }
  else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;
  }
}
$i          = $merkei;
$enderreicht = 0;
$inzeile     = 0;
if ( $absolutwert == 1 ) { $absolutwert = $absolutwert; }
else { $absolutwert = $absolutwert / 100; }
while ( ( !$enderreicht )
  && ( $i < $eingabedateizeilen - 1 ) )
{
  $i++;
  $hilfszeile = $eingabedatei[$i];
  if ( $hilfszeile =~ /^ende$/i ) {
    $enderreicht = 1;
  }
  elsif ( $hilfszeile =~
/^([-0-9\\.]+)( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\\.]+),([a-
z0-9!]+))?(,[a-za0-9!]+))?( parameter="\([^\\"]*\)"?( text="\([^\\"]*\)"?( ausrichtung
=([0-9\\.]+),([a-z]+) | ([0-9\\.]+)))?$/i
  )
  {
    $dy          = $1 / $absolutwert;
    $farbeersatz = $farbe;
    $fuellartersatz = $fuellart ;
    $fuellfarbeersatz = $fuellfarbe ;
    $parameterersatz = $parameter;
    if ( defined $2 ) {
      $farbeersatz = $3;
      $fuellartersatz = $4;
      $fuellfarbeersatz = $5;
    }
  }
}

```

if (\$fuellfarbeersatz eq \$fuellfarbe) {	3814
if (\$farbverlauf == 1) {	3815
\$fuellfarbeersatz = farbverlauf(\$fuellfarbe , \$i);	3816
}	3817
else { \$fuellfarbeersatz = \$fuellfarbeersatz ; }	3818
}	3819
\$linienstaerkeersatz = \$linienstaerke;	3820
\$linienartersatz = \$linienart;	3821
if (defined \$6) {	3822
\$linienstaerkeersatz = \$7;	3823
\$linienartersatz = \$8;	3824
}	3825
\$farbe = \$10 if (defined \$9);	3826
if (defined \$11) { \$parameterersatz = \$12; }	3827
	3828
	3829
\$text = "";	3830
\$text = \$14 if (defined \$13);	3831
\$textwinkersatz = \$textwinkel;	3832
\$textausrichtungsersatz = \$textausrichtung;	3833
if (defined \$15) {	3834
\$textwinkersatz = \$16;	3835
\$textausrichtungsersatz = \$17;	3836
}	3837
\$inzeile++;	3838
achsengraph(\$x + \$dx + \$dx / 2, \$y + \$dy);	3839
saeule(	3840
\$layer, \$x, \$y,	3841
\$dx, \$dy, \$farbeersatz,	3842
\$fuellartersatz , \$fuellfarbeersatz , \$linienartersatz ,	3843
\$linienstaerkeersatz , \$text, \$textwinkersatz,	3844
\$textausrichtungsersatz, "s", \$parameterersatz	3845
);	3846
\$y = \$y + \$dy;	3847
}	3848
else {	3849
kommentar(\$layer, "Kaputte Zeile");	3850
\$enderreicht = 1;	3851
}	3852
}	3853
if (!\$enderreicht) { printf ("%Keine sinnvollen Daten\n"); }	3854
}	3855
elseif (\$zeile =~	3856
/^tortendiagramm(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(	3857
linie=([0-9\.\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+)?(parameter="\^[^"]*")?(p	
=([0-9\.\.]+),([0-9\.\.]+))?(r=([0-9\.\.]+),([0-9\.\.]+))?(,[a-z0-9\.\.]+))?(ausrichtung	
=([0-9\.\.]+),([a-z]+) ([0-9\.\.]+))?\$ /i	
)	3858
{	3859
standard(10, getlinienart (), 0.2, getfarbe(), "", getfuellfarbe (), "", "");	3860
\$n = 0;	3861
\$j = 0;	3862
\$layer = \$2 if (defined \$1);	3863
	3864
if (defined \$3) {	3865
\$farbe = \$4;	3866
\$fuellart = \$5;	3867
\$fuellfarbe = \$6;	3868
}	3869
	3870
if (defined \$7) {	3871

```

$linienstaerke = $8;
$linienart      = $9;
}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$x1 = 0;
$y1 = 0;
if ( defined $14 ) {
    $x1 = $15;
    $y1 = $16;
}
$radius = 2;
$abstand = 0;
if ( defined $17 ) {
    $radius = $18;
    $abstand = $19;
}
$startwinkel = 0;
$startwinkel = $21 if ( defined $20 );
$textwinkel = 0;
$textausrichtung = "auf";
if ( defined $22 ) {
    $textwinkel = $23;
    $textausrichtung = $24;
}
$x2 = $x1;
$y2 = $y1;
$enderreicht = 0;
$sinzeile = 0;
$absolutwert = $n = $j = 0;
$merkei = $i;
while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
{
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
    elsif ( $hilfszeile =~
/^([0-9\.]+)( farbe=([a-zAEOeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\.]+),([a-
z0-9!]+))?(,([a-zAEOeue0-9!]+))?( parameter="\([^"]*\)" )?( r=([0-9\.]+),([0-9\.]+))?(
text="\([^"]*\)" )?( ausrichtung=([0-9\.]+),([a-z]+) | ([0-9\.]+)) )?$/i
    )
    {
        $absolutwert = $absolutwert + $1;
        $j++;
    }
    else { $enderreicht = 1; }
}
$enderreicht = 0;
$sinzeile = 0;
$i = $merkei;

$winkelstart = $startwinkel;
while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
{
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
    elsif ( $hilfszeile =~

```

/^([0-9\\.]+)(farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie=([0-9\\.]+),([a-z0-9!]+))?(,[a-zaeoue0-9!]+)?(parameter="\([^"]*\)"?(r=([0-9\\.]+),([0-9\\.]+))?(text="\([^"]*\)"?(ausrichtung=([0-9\\.]+),((([a-z]+) ([0-9\\.]+)))?\$/i	3931
)	3932
{	3933
\$wert = \$1;	3934
\$farbeersatz = \$farbe;	3935
\$fuellartersatz = \$fuellart ;	3936
	3937
\$fuellfarbeersatz = \$fuellfarbe ;	3938
\$parameterersatz = \$parameter;	3939
if (defined \$2) {	3940
\$farbeersatz = \$3;	3941
\$fuellartersatz = \$4;	3942
\$fuellfarbeersatz = \$5;	3943
}	3944
	3945
if (\$fuellfarbeersatz eq \$fuellfarbe) {	3946
if (\$farbverlauf == 1) {	3947
\$fuellfarbeersatz = farbverlauf(\$fuellfarbe , \$i);	3948
}	3949
else { \$fuellfarbeersatz = \$fuellfarbeersatz ; }	3950
}	3951
	3952
\$linienstaerkeersatz = \$linienstaerke;	3953
\$linienartersatz = \$linienart ;	3954
if (defined \$6) {	3955
\$linienstaerkeersatz = \$7;	3956
\$linienartersatz = \$8;	3957
}	3958
\$farbe = \$10 if (defined \$9);	3959
if (defined \$11) { \$parameterersatz = \$12; }	3960
	3961
\$radiusersatz = \$radius;	3962
\$abstandersatz = \$abstand;	3963
if (defined \$13) {	3964
\$radiusersatz = \$14;	3965
\$abstandersatz = \$15;	3966
}	3967
\$text = "";	3968
\$text = \$17 if (defined \$16);	3969
\$textwinklersatz = \$textwinkel;	3970
\$textausrichtungsersatz = \$textausrichtung;	3971
if (defined \$18) {	3972
\$textwinklersatz = \$19;	3973
\$textausrichtungsersatz = \$20;	3974
}	3975
\$winkel = sprintf("%0.4f", ((\$wert / \$absolutwert) * 360));	3976
\$inzeile++;	3977
	3978
torte(	3979
\$layer, \$x2, \$y2,	3980
\$winkel, \$winkelstart, \$radiusersatz,	3981
\$abstandersatz, \$farbe, \$fuellartersatz ,	3982
\$fuellfarbeersatz , \$linienartersatz , \$linienstaerkeersatz ,	3983
\$text, \$textwinklersatz, \$textausrichtungsersatz,	3984
\$parameterersatz	3985
);	3986
\$winkelstart = \$winkelstart + \$winkel;	3987
}	3988
else {	3989

```

        kommentar( $layer, "Kaputte Zeile" );
        $senderreicht = 1;
    }
}
if ( !$senderreicht ) { printf("%Keine sinnvollen Daten\n"); }
}
elseif ( $zeile =~
/^ringdiagramm( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(
linie=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoue0-9!]+))?( parameter="\([^"]*\)" )?( p
=([0-9\.]+),([0-9\.]+))?( r=([0-9\.]+),([0-9\.]+))?(,[a-z0-9!]+))?( ausrichtung
=([0-9\.]+),([a-z]+)|([0-9\.]+))?)$/i
)
{
    $abstand = 0;
    standard( 10, getlinienart(), 0.2, getfarbe(), "", getfuellfarbe(), "", "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe  = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart      = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $x1 = 0;
    $y1 = 0;
    if ( defined $14 ) {
        $x1 = $15;
        $y1 = $16;
    }
    $radius = 2;
    $dicke = 0;
    if ( defined $17 ) {
        $radius = $18;
        $dicke = $19;
    }
    $startwinkel = 0;
    $startwinkel = $21 if ( defined $20 );
    $textwinkel = 0;
    $textausrichtung = "auf";
    if ( defined $22 ) {
        $textwinkel = $23;
        $textausrichtung = $24;
    }
    $x2 = $x1;
    $y2 = $y1;
    $senderreicht = 0;
    $inzeile = 0;
    $absolutwert = $n = $j = 0;
    $merkei = $i;
    while ( ( !$senderreicht )
        && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];

```

if (\$hilfszeile =~ /^ende\$/i) {	4048
\$endeerreicht = 1;	4049
}	4050
elseif (\$hilfszeile =~	4051
/^([0-9\.]+)(farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie=([0-9\.]+),([a-	4052
z0-9!]+))?,([a-zaeoue0-9!]+))?(parameter="\([^\\"]*\)")?(r=([0-9\.]+),([0-9\.]+)	
,([0-9\.]+))?(text="\([^\\"]*\)")?(ausrichtung=([0-9\.]+),([a-z]+) ! ([0-9\.]+)))?\$ /i	
)	4053
{	4054
\$absolutwert = \$absolutwert + \$1;	4055
\$j++;	4056
}	4057
else { \$endeerreicht = 1; }	4058
}	4059
\$endeerreicht = 0;	4060
\$inzeile = 0;	4061
\$i = \$merkei;	4062
\$winkelstart = \$startwinkel;	4063
while ((!\$endeerreicht)	4064
&& (\$i < \$eingabedateizeilen - 1))	4065
{	4066
\$i++;	4067
\$hilfszeile = \$eingabedatei[\$i];	4068
if (\$hilfszeile =~ /^ende\$/i) {	4069
\$endeerreicht = 1;	4070
}	4071
elseif (\$hilfszeile =~	4072
/^([0-9\.]+)(farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie=([0-9\.]+),([a-	4073
z0-9!]+))?,([a-zaeoue0-9!]+))?(parameter="\([^\\"]*\)")?(r=([0-9\.]+),([0-9\.]+)	
,([0-9\.]+))?(text="\([^\\"]*\)")?(ausrichtung=([0-9\.]+),([a-z]+) ! ([0-9\.]+)))?\$ /i	
)	4074
{	4075
\$wert = \$1;	4076
\$farbeersatz = \$farbe;	4077
\$fuellartersatz = \$fuellart ;	4078
\$fuellfarbeersatz = \$fuellfarbe ;	4079
\$parameterersatz = \$parameter;	4080
if (defined \$2) {	4081
\$farbeersatz = \$3;	4082
\$fuellartersatz = \$4;	4083
\$fuellfarbeersatz = \$5;	4084
}	4085
	4086
if (\$fuellfarbeersatz eq \$fuellfarbe) {	4087
if (\$farbverlauf == 1) {	4088
\$fuellfarbeersatz = farbverlauf(\$fuellfarbe , \$i);	4089
}	4090
else { \$fuellfarbeersatz = \$fuellfarbeersatz ; }	4091
}	4092
	4093
\$linienstaerkeersatz = \$linienstaerke;	4094
\$linienartersatz = \$linienart ;	4095
if (defined \$6) {	4096
\$linienstaerkeersatz = \$7;	4097
\$linienartersatz = \$8;	4098
}	4099
\$farbe = \$10 if (defined \$9);	4100
if (defined \$11) { \$parameterersatz = \$12; }	4101
	4102
\$radiusersatz = \$radius;	4103
\$dickeersatz = \$dicke;	4104

```

$abstandersatz = $abstand;
if ( defined $13 ) {
    $radiusersatz = $14;
    $dickeersatz = $15;
    $abstandersatz = $16;
}
$text = "";
$text = $18 if ( defined $17 );
$textwinkersatz = $textwinkel;
$textausrichtungersatz = $textausrichtung;
if ( defined $19 ) {
    $textwinkersatz = $20;
    $textausrichtungersatz = $21;
}
$winkel = sprintf( "%0.4f", ( ( $wert / $absolutwert ) * 360 ) );
$inzeile++;
ringdiagramm(
    $layer,          $x2,          $y2,
    $winkel,         $winkelstart, $radiusersatz,
    $dickeersatz,    $abstandersatz, $farbe,
    $fuellartersatz , $fuellfarbeersatz , $linienartersatz ,
    $linienstaerkeersatz , $text,    $textwinkersatz,
    $textausrichtungersatz, $parameterersatz
);
$winkelstart = $winkelstart + $winkel;
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;
}
}
if ( !$enderreicht ) {
    printf( "%Keine sinnvollen Daten\n" );
}
}
elseif ( $zeile =~
/^balkendiagramm( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(
linie=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoue0-9!]+))?( parameter="\([^"]*\)" )?( p
=([0-9-9\.]+),([0-9-9\.]+))?( r=([0-9-9\.]+),([0-9-9\.]+))?( ausrichtung=([0-9-9\.]+),([a-z
]+)|([0-9-9\.]+))?)$/i
)
{
    setbalken();
    standard( 10, getlinienart(), 0.2, getfarbe(), "", getfuellfarbe(), "", "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe = $4;
        $fuellart = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $x1 = 0;
    $y1 = 0;

```

```

if ( defined $14 ) {
    $x1 = $15;
    $y1 = $16;
}
$dy      = 1;
$abstand = 1;
if ( defined $17 ) {
    $dy      = $18;
    $abstand = $19;
}
$textwinkel      = 0;
$textausrichtung = "auf";
if ( defined $20 ) {
    $textwinkel      = $21;
    $textausrichtung = $22;
}
}
$x      = $x1;
$y      = $y1;
$enderreicht = 0;
$inzeile  = 0;
$merkei   = $i;
while ( ( !$enderreicht ) && ( $i < $eingabedateizeilen - 1 ) ) {
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
    elsif ( $hilfszeile =~
/^([0-9\.]+)( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\.]+),([a-
z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\([^\"]*\)")?( r=([0-9\.]+),([0-9\.]+))?(
text="\([^\"]*\)")?( ausrichtung=([0-9\.]+),([a-z]+) | ([0-9\.]+)))?$/i
    )
    {
        if ( getrelativ () == 1 ) {
            $absolut += $1;
            $j++;
            setmarkierung( "Prozent", "x" );
        }
        else {
            $absolut = $absolut;
            $j++;
        }
    }
    else {
        kommentar( $layer, "Kaputte Zeile" );
        $enderreicht = 1;
    }
}
$enderreicht = 0;
$inzeile      = 0;
$i            = $merkei;
if ( $absolut == 1 ) { $absolut = $absolut; }
else { $absolut = $absolut / 100; }
while ( ( !$enderreicht ) && ( $i < $eingabedateizeilen - 1 ) ) {
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
    elsif ( $hilfszeile =~
/^([0-9\.]+)( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\.]+),([a-
z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\([^\"]*\)")?( r=([0-9\.]+),([0-9\.]+))?(
text="\([^\"]*\)")?( ausrichtung=([0-9\.]+),([a-z]+) | ([0-9\.]+)))?$/i
    )

```

```

{
    $dx          = $1;
    $farbeersatz = $farbe;
    $fuellartersatz = $fuellart ;
    $fuellfarbeersatz = $fuellfarbe ;
    $parameterersatz = $parameter;
    if ( defined $2 ) {
        $farbeersatz = $3;
        $fuellartersatz = $4;
        $fuellfarbeersatz = $5;
    }

    if ( $fuellfarbeersatz eq $fuellfarbe ) {
        if ( $farbverlauf == 1 ) {
            $fuellfarbeersatz = farbverlauf( $fuellfarbe , $i );
        }
        else { $fuellfarbeersatz = $fuellfarbeersatz ; }
    }

    $linienstaerkeersatz = $linienstaerke;
    $linienartersatz = $linienart ;
    if ( defined $6 ) {
        $linienstaerkeersatz = $7;
        $linienartersatz = $8;
    }
    $farbe = $10 if ( defined $9 );
    if ( defined $11 ) { $parameterersatz = $12; }
    $dyersatz = $dy;
    $abstandersatz = $abstand;
    if ( defined $13 ) {
        $dyersatz = $14;
        $abstandersatz = $15;
    }
    $text = "";
    $text = $17 if ( defined $16 );
    $textwinklersatz = $textwinkel;
    $textausrichtungersatz = $textausrichtung;

    if ( defined $18 ) {
        $textwinklersatz = $19;
        $textausrichtungersatz = $20;
    }
    $inzeile++;
    saeule(
        $layer,          $x,          $y,
        $dx / $absolut,   $dyersatz,   $farbeersatz,
        $fuellartersatz , $fuellfarbeersatz , $linienartersatz ,
        $linienstaerkeersatz , $text,   $textwinklersatz,
        $textausrichtungersatz, "b",    $parameterersatz
    );
    $y = $y + $dyersatz + $abstandersatz;
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;
}
}

if ( !$enderreicht ) {
    printf("%Keine sinnvollen Daten\n");
}

```

\$absolut = 1;	4281
}	4282
	4283
elseif (\$zeile =~	4284
/^balkendiagramm(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(	4285
linie=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\^[^"]*")?(r	
=([0-9\..]+))?(ausrichtung=([0-9\..]+),([a-z]+) ! ([0-9\..]+)))?\$/i	
)	4286
{	4287
setbalken();	4288
standard(10, getlinienart (), 0.2, getfarbe(), "", getfuellfarbe (), "", "");	4289
\$layer = \$2 if (defined \$1);	4290
	4291
if (defined \$3) {	4292
\$farbe = \$4;	4293
\$fuellart = \$5;	4294
\$fuellfarbe = \$6;	4295
}	4296
	4297
if (defined \$7) {	4298
\$linienstaerke = \$8;	4299
\$linienart = \$9;	4300
}	4301
\$farbe = \$11 if (defined \$10);	4302
if (defined \$12) { \$parameter = \$13; }	4303
	4304
\$hoehe = 1;	4305
if (defined \$14) { \$hoehe = \$15; }	4306
\$textwinkel = 0;	4307
\$textausrichtung = "auf";	4308
if (defined \$16) {	4309
\$textwinkel = \$17;	4310
\$textausrichtung = \$18;	4311
}	4312
\$enderreicht = 0;	4313
\$inzeile = 0;	4314
\$merkei = \$i;	4315
while ((!\$enderreicht) && (\$i < \$eingabedateizeilen - 1)) {	4316
\$i++;	4317
\$hilfszeile = \$eingabedatei[\$i];	4318
if (\$hilfszeile =~ /^ende\$/i) { \$enderreicht = 1; }	4319
elseif (\$hilfszeile =~	4320
/^([0-9\..]+),([0-9\..]+)(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	4321
=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\^[^"]*")?(r	
=([0-9\..]+))?(text="\^[^"]*")?(ausrichtung=([0-9\..]+),([a-z]+) ! ([0-9\..]+)))?\$/i	
)	4322
{	4323
if (getrelativ () == 1) {	4324
\$absolut += \$1;	4325
\$j++;	4326
setmarkierung("Prozent", "x");	4327
}	4328
else {	4329
\$absolut = \$absolut;	4330
\$j++;	4331
}	4332
	4333
}	4334
else {	4335
kommentar(\$layer, "Kaputte Zeile");	4336
\$enderreicht = 1;	4337

```

    }
  }
  $enderreicht = 0;
  $inzeile      = 0;
  $i            = $merkei;
  if ( $absolut == 1 ) { $absolut = $absolut; }
  else          { $absolut = $absolut / 100; }
  while ( ( !$enderreicht ) && ( $i < $eingabedateizeilen - 1 ) ) {
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
    elsif ( $hilfszeile =~
/^([-0-9\\.]+),([-0-9\\.]+)(farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie
=([-0-9\\.]+),([a-z0-9!]+))?(,[a-zA-Z0-9!]+))?(parameter=\"([^\"]*)\"?( r
=([-0-9\\.]+))?(text=\"([^\"]*)\"?(ausrichtung=([-0-9\\.]+),([a-z]+)|([-0-9\\.]+)))?$/i
)
    {
      $dx          = $1;
      $y           = $2;
      $farbeersatz = $farbe;
      $fuellartersatz = $fuellart ;
      $fuellfarbeersatz = $fuellfarbe ;
      $parameterersatz = $parameter;
      if ( defined $3 ) {
        $farbeersatz = $4;
        $fuellartersatz = $5;
        $fuellfarbeersatz = $6;
      }

      if ( $fuellfarbeersatz eq $fuellfarbe ) {
        if ( $farbverlauf == 1 ) {
          $fuellfarbeersatz = farbverlauf( $fuellfarbe , $i );
        }
        else { $fuellfarbeersatz = $fuellfarbeersatz ; }
      }

      $linienstaerkeersatz = $linienstaerke;
      $linienartersatz     = $linienart ;
      if ( defined $7 ) {
        $linienstaerkeersatz = $8;
        $linienartersatz     = $9;
      }
      $farbe = $11 if ( defined $10 );
      if ( defined $12 ) { $parameterersatz = $13; }
      $hoeheersatz = $hoehe;
      if ( defined $14 ) {
        $hoeheersatz = $15;
      }
      $text          = "";
      $text           = $17 if ( defined $16 );
      $textwinkersatz = $textwinkel;
      $textausrichtungsersatz = $textausrichtung;

      if ( defined $18 ) {
        $textwinkersatz = $19;
        $textausrichtungsersatz = $20;
      }
      $inzeile++;
      saeule(
        $layer,
        $y - $hoeheersatz / 2, $dx / $absolut,

```

```

        $hoeheersatz,          $farbeersatz,
        $fuellartersatz ,      $fuellfarbeersatz ,
        $linienartersatz ,     $linienstaerkeersatz ,
        $text,                  $textwinkelerersatz,
        $textausrichtungsersatz, "b",
        $parameterersatz
    );
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $enderreicht = 1;
}
}

if ( !$enderreicht ) {
    printf ("%Keine sinnvollen Daten\n");
}
$absolut = 1;
}

elseif ( $zeile =~
/^balkendiagrammdreieck( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0
-9!]+))?( linie=([0-9\.] +),([a-z0-9!]+))?(,([a-zaeoue0-9!]+))?( parameter="\([^"]*\)" )
?( r=([0-9\.] +))?( ausrichtung=([0-9\.] +),([a-z]+) | ([0-9\.] +)) )?$/i
)
{
    setbalken();
    standard( 10, getlinienart(), 0.2, getfarbe(), "", getfuellfarbe(), "", "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe  = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $hoehe = 1;
    if ( defined $14 ) { $hoehe = $15; }
    $textwinkel      = 0;
    $textausrichtung = "auf";
    if ( defined $16 ) {
        $textwinkel      = $17;
        $textausrichtung = $18;
    }
    $enderreicht = 0;
    $sinzeile     = 0;
    $merkei       = $i;
    while ( ( !$enderreicht ) && ( $i < $eingabedateizeilen - 1 ) ) {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
        elseif ( $hilfszeile =~
/^([0-9\.] +),([0-9\.] +)( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.] +),([a-z0-9!]+))?(,([a-zaeoue0-9!]+))?( parameter="\([^"]*\)" )?( r

```

```

=([0-9\.]+))?( text="\([^"]*\)" )?( ausrichtung=([0-9\.]+),([a-z]+)|([0-9\.]+)))?$/i
)
{
  if ( getrelativ () == 1 ) {
    $absolut += $1;
    $j++;
    setmarkierung( "Prozent", "x" );
  }
  else {
    $absolut = $absolut;
    $j++;
  }
}
else {
  kommentar( $layer, "Kaputte Zeile" );
  $senderreicht = 1;
}
}
$senderreicht = 0;
$inzeile = 0;
$i = $merkei;
if ( $absolut == 1 ) { $absolut = $absolut; }
else { $absolut = $absolut / 100; }
while ( ( !$senderreicht ) && ( $i < $eingabedateizeilen - 1 ) ) {
  $i++;
  $hilfszeile = $eingabedatei[$i];
  if ( $hilfszeile =~ /^ende$/i ) { $senderreicht = 1; }
  elsif ( $hilfszeile =~
/^([0-9\.]+),([0-9\.]+)( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoue0-9!]+))?( parameter="\([^"]*\)" )?( r
=([0-9\.]+))?( text="\([^"]*\)" )?( ausrichtung=([0-9\.]+),([a-z]+)|([0-9\.]+)))?$/i
)
{
  $dx = $1;
  $y = $2;
  $farbeersatz = $farbe;
  $fuellartersatz = $fuellart ;
  $fuellfarbeersatz = $fuellfarbe ;
  $parameterersatz = $parameter;
  if ( defined $3 ) {
    $farbeersatz = $4;
    $fuellartersatz = $5;
    $fuellfarbeersatz = $6;
  }

  if ( $fuellfarbeersatz eq $fuellfarbe ) {
    if ( $farbverlauf == 1 ) {
      $fuellfarbeersatz = farbverlauf( $fuellfarbe , $i );
    }
    else { $fuellfarbeersatz = $fuellfarbeersatz ; }
  }

  $linienstaerkeersatz = $linienstaerke;
  $linienartersatz = $linienart ;
  if ( defined $7 ) {
    $linienstaerkeersatz = $8;
    $linienartersatz = $9;
  }
  $farbe = $11 if ( defined $10 );
  if ( defined $12 ) { $parameterersatz = $13; }
}

```

\$hoeheersatz = \$hoehe;	4513
if (defined \$14) {	4514
\$hoeheersatz = \$15;	4515
}	4516
\$text = "";	4517
\$text = \$17 if (defined \$16);	4518
\$textwinklersatz = \$textwinkel;	4519
\$textausrichtungsersatz = \$textausrichtung;	4520
	4521
if (defined \$18) {	4522
\$textwinklersatz = \$19;	4523
\$textausrichtungsersatz = \$20;	4524
}	4525
\$inzeile++;	4526
saeuledreieck(	4527
\$layer, 0,	4528
\$y - \$hoeheersatz / 2, \$dx / \$absolut,	4529
\$hoeheersatz, \$farbeersatz,	4530
\$fuellartersatz , \$fuellfarbeersatz ,	4531
\$linienartersatz , \$linienstaerkeersatz ,	4532
\$text, \$textwinklersatz,	4533
\$textausrichtungsersatz, "b",	4534
\$parameterersatz	4535
);	4536
}	4537
else {	4538
kommentar(\$layer, "Kaputte Zeile");	4539
\$senderreicht = 1;	4540
}	4541
}	4542
	4543
if (!\$senderreicht) {	4544
printf ("%Keine sinnvollen Daten\n");	4545
}	4546
\$absolut = 1;	4547
}	4548
	4549
elseif (\$zeile =~	4550
/^balkendiagrammdreieck(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0	4551
-9!]+))?(linie=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\([^\"]*\)"	
?(p=([0-9\..]+),([0-9\..]+))?(r=([0-9\..]+),([0-9\..]+))?(ausrichtung=([0-9\..]+),([a	
-z]+) ([0-9\..]+))?)?\$/i	
)	4552
{	4553
setbalken();	4554
standard(10, getlinienart (), 0.2, getfarbe(), "", getfuellfarbe (), "", "");	4555
\$layer = \$2 if (defined \$1);	4556
	4557
if (defined \$3) {	4558
\$farbe = \$4;	4559
\$fuellart = \$5;	4560
\$fuellfarbe = \$6;	4561
}	4562
	4563
if (defined \$7) {	4564
\$linienstaerke = \$8;	4565
\$linienart = \$9;	4566
}	4567
\$farbe = \$11 if (defined \$10);	4568
	4569
if (defined \$12) { \$parameter = \$13; }	4570

```

4571 $x1 = 0;
4572 $y1 = 0;
4573 if ( defined $14 ) {
4574     $x1 = $15;
4575     $y1 = $16;
4576 }
4577 $dy      = 1;
4578 $abstand = 1;
4579 if ( defined $17 ) {
4580     $dy      = $18;
4581     $abstand = $19;
4582 }
4583 $textwinkel      = 0;
4584 $textausrichtung = "auf";
4585 if ( defined $20 ) {
4586     $textwinkel      = $21;
4587     $textausrichtung = $22;
4588 }
4589 $x      = $x1;
4590 $y      = $y1;
4591 $enderreicht = 0;
4592 $inzeile     = 0;
4593 while ( ( !$enderreicht )
4594     && ( $i < $eingabedateizeilen - 1 ) )
4595 {
4596     $i++;
4597     $hilfszeile = $eingabedatei[$i];
4598     if ( $hilfszeile =~ /^ende$/i ) {
4599         $enderreicht = 1;
4600     }
4601     elseif ( $hilfszeile =~
4602 / ^ ( [ - 0 - 9 \ . ] + ) ( farbe = ( [ a - z aeoeue 0 - 9 ! ] + ) , ( [ a - z 0 - 9 ! ] + ) , ( [ a - z 0 - 9 ! ] + ) ) ? ( linie = ( [ 0 - 9 \ . ] + ) , ( [ a -
4603 z 0 - 9 ! ] + ) ) ? ( , ( [ a - z aeoeue 0 - 9 ! ] + ) ) ? ( parameter = \ " ( [ ^ \ " ] * ) \ " ) ? ( r = ( [ - 0 - 9 \ . ] + ) , ( [ - 0 - 9 \ . ] + ) ) ? (
4604 text = \ " ( [ ^ \ " ] * ) \ " ) ? ( ausrichtung = ( [ 0 - 9 \ . ] + ) , ( ( [ a - z ] + ) ! ( [ - 0 - 9 \ . ] + ) ) ) ? $ / i
4605 )
4606 {
4607     $dx      = $1;
4608     $farbersatz = $farbe;
4609     $fuellartersatz = $fuellart ;
4610     $fuellfarbersatz = $fuellfarbe ;
4611     $parameterersatz = $parameter;
4612     if ( defined $2 ) {
4613         $farbersatz = $3;
4614         $fuellartersatz = $4;
4615         $fuellfarbersatz = $5;
4616     }
4617     $linienstaerkeersatz = $linienstaerke ;
4618     $linienartersatz = $linienart ;
4619     if ( defined $6 ) {
4620         $linienstaerkeersatz = $7;
4621         $linienartersatz = $8;
4622     }
4623     $farbe = $10 if ( defined $9 );
4624     if ( defined $11 ) { $parameterersatz = $12; }
4625
4626     $dyersatz = $dy;
4627     $abstandersatz = $abstand;
4628     if ( defined $13 ) {
4629         $dyersatz = $14;

```

```

        $abstandersatz = $15;
    }
    $text          = "";
    $text          = $17 if ( defined $16 );
    $textwinkersatz = $textwinkel;
    $textausrichtungersatz = $textausrichtung;
    if ( defined $18 ) {
        $textwinkersatz = $19;
        $textausrichtungersatz = $20;
    }
    $inzeile++;
    saeuledreieck(
        $layer,          $x,          $y,
        $dx,             $dyersatz,   $farbeersatz,
        $fuellartersatz , $fuellfarbeersatz , $linienartersatz ,
        $linienstaerkeersatz , $text,   $textwinkersatz,
        $textausrichtungersatz, "b",    $parameterersatz
    );
    $y = $y + $dyersatz + $abstandersatz;
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $endeerreicht = 1;
}
}
if ( !$endeerreicht ) { printf ("%Keine sinnvollen Daten\n"); }
}
elseif ( $zeile =~
/^balkendiagrammgestapelt( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0
-9!]+))?( linie=([0-9\.\.]+),([a-z0-9!]+))?(,[a-zaeoue0-9!]+))?( parameter="\([^"]*\)" )
?( p=([0-9\.\.]+),([0-9\.\.]+))?( r=([0-9\.\.]+),([0-9\.\.]+))?( ausrichtung=([0-9\.\.]+),([a
-z]+)!( [0-9\.\.]+))?)$/i
)
{
    setbalken();
    standard( 10, getlinienart (), 0.2, getfarbe(), "", getfuellfarbe (), "", "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $x = 0;
    $y = 0;
    if ( defined $14 ) {
        $x = $15;
        $y = $16;
    }
    $dy      = 1;
    $breite  = 10;
    if ( defined $17 ) {
        $dy      = $18;

```

```

    $breite = $19;
}
$textwinkel = 90;
$textausrichtung = "auf";
if ( defined $20 ) {
    $textwinkel = $21;
    $textausrichtung = $22;
}
$x2 = $x1;
$y2 = $y1;
$enderreicht = 0;
$absolutwert = $n = $j = 0;
$inzeile = 0;
$merkei = $i;
while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
{
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) {
        $enderreicht = 1;
    }
    elsif ( $hilfszeile =~
/^([0-9\.]+)( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\.]+),([a-
z0-9!]+))?(,[a-zA0-9!]+))?( parameter="\^[^"]*" )?( text="\^[^"]*" )?( ausrichtung
=([0-9\.]+),([a-z]+) ! ([0-9\.]+)))?$/i
    )
    {
        if ( getrelativ () == 1 ) {
            $absolutwert = $absolutwert + $1;
            setmarkierung( "Prozent", "x" );
            $j++;
        }
        else {
            $absolutwert = 1;
            $j++;
        }
    }
    else {
        kommentar( $layer, "Kaputte Zeile" );
        $enderreicht = 1;
    }
}
$i = $merkei;
$enderreicht = 0;
$inzeile = 0;

if ( $absolutwert == 1 ) { $absolutwert = $absolutwert; }
else {
    $absolutwert = $absolutwert / 100;
}
while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
{
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) {
        $enderreicht = 1;
    }
    elsif ( $hilfszeile =~

```

```

/^([0-9\.]+)( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\.]+),([a-
z0-9!]+))?(,[a-za0-9!]+))?( parameter="\^[^"]*" )?( text="\^[^"]*" )?( ausrichtung
=([0-9\.]+),([a-z]+) | ([0-9\.]+)))?$/i
)
{
    $dx          = $1 / $absolutwert;
    $farbeersatz = $farbe;
    $fuellartersatz = $fuellart ;
    $fuellfarbeersatz = $fuellfarbe ;
    $parameterersatz = $parameter;
    if ( defined $2 ) {
        $farbeersatz = $3;
        $fuellartersatz = $4;
        $fuellfarbeersatz = $5;
    }

    if ( $fuellfarbeersatz eq $fuellfarbe ) {
        if ( $farbverlauf == 1 ) {
            $fuellfarbeersatz = farbverlauf( $fuellfarbe , $i );
        }
        else { $fuellfarbeersatz = $fuellfarbeersatz ; }
    }
    $linienstaerkeersatz = $linienstaerke;
    $linienartersatz = $linienart ;
    if ( defined $6 ) {
        $linienstaerkeersatz = $7;
        $linienartersatz = $8;
    }
    $farbe = $10 if ( defined $9 );
    if ( defined $11 ) { $parameterersatz = $12; }

    $text          = "";
    $text          = $14 if ( defined $13 );
    $textwinklersatz = $textwinkel;
    $textausrichtungsersatz = $textausrichtung;
    if ( defined $15 ) {
        $textwinklersatz = $16;
        $textausrichtungsersatz = $17;
    }
    $inzeile++;
    achsengraph( $x + $dx, $y + $dy + $dy / 2 );
    saeule(
        $layer,          $x,          $y,
        $dx,             $dy,         $farbeersatz,
        $fuellartersatz , $fuellfarbeersatz , $linienartersatz ,
        $linienstaerkeersatz , $text,   $textwinklersatz,
        $textausrichtungsersatz, "b",   $parameterersatz
    );
    $x = $x + $dx;
}
else {
    kommentar( $layer, "Kaputte Zeile" );
    $senderreicht = 1;
}
}
if ( !$senderreicht ) { printf ("%Keine sinnvollen Daten\n"); }
}

elseif ( $zeile =~
/^piktogramm( layer=([0-9]+))?( parameter="\^[^"]*" )?( p=([0-9\.]+),([0-9\.]+))?( r
=([0-9\.]+),([0-9\.]+),([0-9\.]+),([0-9\.]+))?( ausrichtung=([0-9\.]+),([a-z0

```

```

-9!]+))?$ /i
)
{
  setpikto();

  standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
  $layer = $2 if ( defined $1 );
  kommentar( $layer, "Ein Bilddiagramm zeichnen" );

  if ( defined $3 ) { $parameter = $4; }

  $x1 = 0;
  $y1 = 0;
  if ( defined $5 ) {
    $x1 = $6;
    $y1 = $7;
  }
  $breite = 1;
  $winkel = 1;
  $abstandx = 1;
  $abstandy = 1;
  if ( defined $8 ) {
    $breite = $9;
    $winkel = $10;
    $abstandx = $11;
    $abstandy = $12;
  }
  $textwinkel = 0;
  $textausrichtung = "auf";
  if ( defined $13 ) {
    $textwinkel = $14;
    $textausrichtung = $15;
  }

  $x = $x1;
  $y = $y1;
  $enderreicht = 0;

  while ( ( !$enderreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
  {
    $i++;
    $hilfszeile = $eingabedatei[$i];

    if ( $hilfszeile =~ /^ende$/i ) {
      $enderreicht = 1;
    }
    elsif ( $hilfszeile =~
/^( [-0-9\\.]+ ) ( r= ( [-0-9\\.]+ ), ( [-0-9\\.]+ ), ( [-0-9\\.]+ ), ( [-0-9\\.]+ ) ) ? ( text= \" [^\" ]* \" ) ? (
ausrichtung= ( [-0-9\\.]+ ), ( [-a-z0-9!]+ ) ) ? \" [^\" ]* \" $ /i
)
    {
      $anzahl = $1;
      $breiteersatz = $breite;
      $winkelersatz = $winkel;
      $abstandxersatz = $abstandx;
      $abstandyersatz = $abstandy;

      if ( defined $2 ) {

```

4805
4806
4807
4808
4809
4810
4811
4812
4813
4814
4815
4816
4817
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840
4841
4842
4843
4844
4845
4846
4847
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857
4858
4859
4860
4861
4862
4863

\$breiteersatz = \$3;	4864
\$winkelpersatz = \$4;	4865
\$abstandxersatz = \$5;	4866
\$abstandyersatz = \$6;	4867
}	4868
	4869
	4870
\$text = "";	4871
\$text = \$8 if (defined \$7);	4872
\$textwinkelpersatz = \$textwinkel;	4873
\$textausrichtungersatz = \$textausrichtung;	4874
if (defined \$9) {	4875
\$textwinkelpersatz = \$10;	4876
\$textausrichtungersatz = \$11;	4877
}	4878
\$bild = \$12;	4879
	4880
setachsengraph(\$x, \$y, \$x + \$breiteersatz, \$y + \$abstandyersatz);	4881
	4882
bilddiagramm(\$layer, \$bild, \$anzahl, \$x, \$y, \$breiteersatz, \$winkelpersatz,	4883
\$abstandxersatz, \$text, \$textwinkelpersatz, \$textausrichtungersatz,	4884
\$parameter);	4885
\$y = \$y + \$abstandyersatz + bildhoeheberechnet(\$bild, \$breiteersatz);	4886
	4887
}	4888
else {	4889
kommentar(\$layer, "Kaputte Zeile");	4890
\$senderreicht = 1;	4891
}	4892
}	4893
if (!\$senderreicht) {	4894
printf("%Keine sinnvollen Daten\n");	4895
}	4896
}	4897
	4898
elseif (\$zeile =~	4899
/^legende(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	4900
=([0-9\.] +),([a-z0-9!]+))?(([a-zaeoue0-9!]+))?(parameter="\^[^"]*")?(p	
=([0-9\.] +),([0-9\.] +))?(r=([0-9\.] +),([0-9\.] +),([0-9\.] +))?\$/i	4901
)	4902
{	4903
standard(10, getlinienart(), 0.2, getfarbe(), "", getfuellfarbe(), "", "");	4904
\$layer = \$2 if (defined \$1);	4905
	4906
if (defined \$3) {	4907
\$farbe = \$4;	4908
\$fuellart = \$5;	4909
\$fuellfarbe = \$6;	4910
}	4911
if (defined \$7) {	4912
\$linienstaerke = \$8;	4913
\$linienart = \$9;	4914
}	4915
\$farbe = \$11 if (defined \$10);	4916
if (defined \$12) { \$parameter = \$13; }	4917
	4918
\$x = 0;	4919
\$y = 0;	4920
if (defined \$14) {	4921
\$x = \$15;	4922
\$y = \$16;	

```

}
$dx      = 1;
$dy      = 1;
$abstand = 1;
if ( defined $17 ) {
    $dx      = $18;
    $dy      = $19;
    $abstand = $20;
}
$endeerreicht = 0;
$inzeile      = 0;
while ( ( !$endeerreicht )
    && ( $i < $eingabedateizeilen - 1 ) )
{
    $i++;
    $hilfszeile = $eingabedatei[$i];
    if ( $hilfszeile =~ /^ende$/i ) {
        $endeerreicht = 1;
    }
    elsif ( $hilfszeile =~
/^text=\"([^\"]*)\"( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie=([0-9\.]+)
,[a-z0-9!]+))?(,[a-z0-9!]+))?( parameter=\"([^\"]*)\" )?$/i
    )
    {
        $text          = $1;
        $farbeersatz    = $farbe;
        $fuellartersatz = $fuellart ;
        $fuellfarbeersatz = $fuellfarbe ;
        $parameterersatz = $parameter;
        if ( defined $2 ) {
            $farbeersatz    = $3;
            $fuellartersatz  = $4;
            $fuellfarbeersatz = $5;
        }

        if ( $fuellfarbeersatz eq $fuellfarbe ) {

            if ( $farbverlauf == 1 ) {
                $fuellfarbeersatz = farbverlauf( $fuellfarbe , $i );
            }
            else { $fuellfarbeersatz = $fuellfarbeersatz ; }
        }

        $linienstaerkeersatz = $linienstaerke;
        $linienarterersatz    = $linienart ;
        if ( defined $6 ) {
            $linienstaerkeersatz = $7;
            $linienarterersatz    = $8;
        }
        $farbe = $10 if ( defined $9 );
        if ( defined $11 ) {
            $parameterersatz = $12;
        }

        $inzeile++;
        saeule(
            $layer,          $x,
            $y,              $dx,
            $dy,             $farbeersatz,
            $fuellartersatz , $fuellfarbeersatz ,

```

4923
4924
4925
4926
4927
4928
4929
4930
4931
4932
4933
4934
4935
4936
4937
4938
4939
4940
4941
4942
4943
4944
4945
4946
4947
4948
4949
4950
4951
4952
4953
4954
4955
4956
4957
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969
4970
4971
4972
4973
4974
4975
4976
4977
4978
4979
4980
4981
4982

<pre> \$linienartersatz , \$linienstaerkeersatz , \$text, 0, "rechts", "b", \$parameterersatz); \$y = \$y - \$dy - \$abstand; } else { kommentar(\$layer, "Kaputte Zeile"); \$endeerreicht = 1; } } if (!\$endeerreicht) { printf ("%Keine sinnvollen Daten\n"); } } elseif (\$zeile =~ /^linie(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+))?(linie=([0-9\.]+),([a-z0-9!]+)) ?(\\"([^\"]*)\")?(parameter=\\"([^\"]*)\")? p=([0-9\.]+),([0-9\.]+),([0-9\.]+))? p =([0-9\.]+),([0-9\.]+),([0-9\.]+))?\$ /i) { standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "", ""); \$layer = \$2 if (defined \$1); \$farbe = \$4 if (defined \$3); if (defined \$5) { \$linienstaerke = \$6; \$linienart = \$7; } \$ Pfeilart = \$9 if (defined \$8); if (defined \$10) { \$parameter = \$11; } \$x1 = \$12; \$y1 = \$13; \$z1 = ""; \$z1 = \$15 if (defined \$14); \$x2 = \$16; \$y2 = \$17; \$z2 = ""; \$z2 = \$19 if (defined \$18); linie(\$layer, \$x1, \$y1, \$z1, \$x2, \$y2, \$z2, \$linienart, \$linienstaerke, \$farbe, \$parameter, \$Pfeilart); } elseif (\$zeile =~ /^linie(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+))?(linie=([0-9\.]+),([a-z0-9!]+)) ?(\\"([^\"]*)\")?(parameter=\\"([^\"]*)\")? p=([0-9\.]+),([0-9\.]+),([0-9\.]+))? p =([0-9\.]+),([0-9\.]+),([0-9\.]+))? text=\\"([^\"]*)\" ,([0-9\.]+),([0-9!]+)))? ,([0-9\.]+)? ,([0-9\.]+)? ,([0-9\.]+)? \$ /i) { standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "", ""); \$layer = \$2 if (defined \$1); \$farbe = \$4 if (defined \$3); </pre>	4983 4984 4985 4986 4987 4988 4989 4990 4991 4992 4993 4994 4995 4996 4997 4998 4999 5000 5001 5002 5003 5004 5005 5006 5007 5008 5009 5010 5011 5012 5013 5014 5015 5016 5017 5018 5019 5020 5021 5022 5023 5024 5025 5026 5027 5028 5029 5030 5031 5032 5033 5034 5035 5036 5037 5038
--	---

```

if ( defined $5 ) {
    $linienstaerke = $6;
    $linienart      = $7;
}

$pfeilart = $9 if ( defined $8 );

if ( defined $10 ) { $parameter = $11; }

$x1      = $12;
$y1      = $13;
$z1      = "";
$z1      = $15 if ( defined $14 );
$x2      = $16;
$y2      = $17;
$z2      = "";
$z2      = $19 if ( defined $18 );
$text    = $20;
$winkel  = 0;
$winkel  = $22 if ( defined $21 );
$ausrichtung = 0;
$ausrichtung = $24 if ( defined $23 );
$abstandx = 0.2;
$abstandx = $26 if ( defined $25 );
$abstandy = $abstandx;

$abstandy = $28 if ( defined $27 );
$abstandz = $abstandx;
$abstandz = $30 if ( defined $29 );

linietext (
    $layer, $x1,      $y1,      $z1,      $x2,
    $y2,    $z2,      $abstandx, $abstandy, $abstandz,
    $text,  $winkel,  $ausrichtung, $linienart, $linienstaerke,
    $farbe, $parameter, $pfeilart
);
}

elsif ( $zeile =~
/^linie( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+))?( linie=([0-9\.]+),([a-z0-9!]+))
?(\^[^\"]*)\")?( parameter="\^[^\"]*")? p=([0-9\.]+),([0-9\.]+),([0-9\.]+))? p
=([0-9\.]+),([0-9\.]+),([0-9\.]+))? r=([0-9\.]+)( text="\^[^\"]*")\")?(\^[^\"]*)\")?
?(\^[^\"]*")\")?(\^[^\"]*")\")?(\^[^\"]*")\")?(\^[^\"]*")\")?$/i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
    "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart      = $7;
    }

    $pfeilart = $9 if ( defined $8 );

    if ( defined $10 ) { $parameter = $11; }

    $x1      = $12;

```

\$y1	= \$13;	5097
\$z1	= "";	5098
\$z1	= \$15 if (defined \$14);	5099
\$x2	= \$16;	5100
\$y2	= \$17;	5101
\$z2	= "";	5102
\$z2	= \$19 if (defined \$18);	5103
\$abstand	= \$20;	5104
\$text	= "";	5105
\$abstandx	= 0.2;	5106
\$ausrichtung	= 0;	5107
\$winkel	= 0;	5108
\$abstandy	= \$abstandz = \$abstandx;	5109
if (defined \$21) {		5110
\$text = \$22;		5111
		5112
\$winkel = \$24 if (defined \$23);		5113
		5114
\$ausrichtung = \$26 if (defined \$25);		5115
		5116
\$abstandx = \$28 if (defined \$27);		5117
		5118
\$abstandy = \$30 if (defined \$29);		5119
		5120
\$abstandz = \$32 if (defined \$31);		5121
}		5122
		5123
linietextabstand(		5124
\$layer,	\$x1, \$y1, \$z1, \$x2,	5125
\$y2,	\$z2, \$abstand, \$abstandx, \$abstandy,	5126
\$abstandz,	\$text, \$winkel, \$ausrichtung, \$linienart,	5127
\$linienstaerke, \$farbe, \$parameter, \$pfeilart		5128
);		5129
		5130
}		5131
		5132
elsif (\$zeile =~		5133
/^linie(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\\.]+),([a-z0-9!]+))		5134
?(\\"([^\"]*)\\")?(parameter=\\"([^\"]*)\\")? p=([0-9\\.]+),([0-9\\.]+),([0-9\\.]+))? r		
=([0-9\\.]+),([0-9\\.]+),([0-9\\.]+))? r=([0-9\\.]+)(text=\\"([^\"]*)\\")?([0-9\\.]+))		
?(\\"([a-z0-9!]+))?(\\([0-9\\.]+))?(\\([0-9\\.]+))?(\\([0-9\\.]+))?)?\$/i		
)		5135
{		5136
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",		5137
"");		5138
\$layer = \$2 if (defined \$1);		5139
\$farbe = \$4 if (defined \$3);		5140
		5141
if (defined \$5) {		5142
\$linienstaerke = \$6;		5143
\$linienart = \$7;		5144
}		5145
		5146
\$pfeilart = \$9 if (defined \$8);		5147
		5148
if (defined \$10) { \$parameter = \$11; }		5149
		5150
\$x1	= \$12;	5151
\$y1	= \$13;	5152
\$z1	= "";	5153
\$z1	= \$15 if (defined \$14);	5154

```

$laenge      = $16;
$w1          = $17;
$w2          = 0;
$w2          = $19 if ( defined $18 );
$abstand     = $20;
$text       = "";
$abstandx    = 0.2;
$ausrichtung = 0;
$winkel      = 0;
$abstandy    = $abstandz = $abstandx;
if ( defined $21 ) {
    $text = $22;

    $winkel = $24 if ( defined $23 );

    $ausrichtung = $26 if ( defined $25 );

    $abstandx = $28 if ( defined $27 );

    $abstandy = $30 if ( defined $29 );

    $abstandz = $32 if ( defined $31 );
}

liniepolartextabstand(
    $layer,      $x1,   $y1,      $z1,      $laenge,
    $w1,         $w2,   $abstand, $abstandx, $abstandy,
    $abstandz,   $text, $winkel,  $ausrichtung, $linienart,
    $linienstaerke, $farbe, $parameter, $pfeilart
);

}

elsif ( $zeile =~
/^linie( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+))?( linie=([0-9\.]+),([a-z0-9!]+))
?(\\"([^\"])*\\")?( parameter=\\"([^\"])*\\")? p=([0-9\.]+),([0-9\.]+),([0-9\.]+))? r
=([0-9\.]+),([0-9\.]+),([0-9\.]+))? text=\\"([^\"])*\\"([0-9\.]+))?(,[a-z0-9!]+)
)?,(,[0-9\.]+))?(,[0-9\.]+))?(,[0-9\.]+))?$ /i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe (), "", getfuellfarbe (), "",
    "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart      = $7;
    }

    $pfeilart = $9 if ( defined $8 );

    if ( defined $10 ) { $parameter = $11; }

    $x1 = $12;
    $y1 = $13;
    $z1 = "";
    $z1 = $15 if ( defined $14 );

    $laenge = $16;
    $w1      = $17;

```

\$w2 = 0;	5213
\$w2 = \$19 if (defined \$18);	5214
	5215
\$text = \$20;	5216
\$winkel = 0;	5217
\$winkel = \$22 if (defined \$21);	5218
\$ausrichtung = 0;	5219
\$ausrichtung = \$24 if (defined \$23);	5220
\$abstandx = 0.2;	5221
\$abstandx = \$26 if (defined \$25);	5222
\$abstandy = \$abstandx;	5223
	5224
\$abstandy = \$28 if (defined \$27);	5225
\$abstandz = \$abstandx;	5226
\$abstandz = \$30 if (defined \$29);	5227
	5228
linietextpolar (	5229
\$layer, \$x1, \$y1, \$z1, \$laenge,	5230
\$w1, \$w2, \$abstandx, \$abstandy, \$abstandz,	5231
\$text, \$winkel, \$ausrichtung, \$linienart, \$linienstaerke,	5232
\$farbe, \$parameter, \$pfeilart	5233
);	5234
}	5235
	5236
elseif (\$zeile =~	5237
/^linie(layer=([0-9]+))?(farbe=([a-zAEOEUE0-9!]+))?(linie=([0-9\.]�),([a-z0-9!]+))	5238
?(\\"([^\"]*)\\")?(parameter=\\"([^\"]*)\\")? p=([0-9\.]�),([0-9\.]�),([0-9\.]�))? r	
=([0-9\.]�),([0-9\.]�),([0-9\.]�))?\$ /i	
)	5239
{	5240
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	5241
"");	5242
\$layer = \$2 if (defined \$1);	5243
\$farbe = \$4 if (defined \$3);	5244
	5245
if (defined \$5) {	5246
\$linienstaerke = \$6;	5247
\$linienart = \$7;	5248
}	5249
	5250
\$pfeilart = \$9 if (defined \$8);	5251
if (defined \$10) { \$parameter = \$11; }	5252
	5253
\$x1 = \$12;	5254
\$y1 = \$13;	5255
\$z1 = "";	5256
\$z1 = \$15 if (defined \$14);	5257
\$laenge = \$16;	5258
\$w1 = \$17;	5259
\$w2 = "";	5260
\$w2 = \$19 if (defined \$18);	5261
liniepolar (	5262
\$layer, \$x1, \$y1, \$z1,	5263
\$laenge, \$w1, \$w2, \$linienart,	5264
\$linienstaerke, \$farbe, \$parameter, \$pfeilart	5265
);	5266
}	5267
	5268
elseif (\$zeile =~	5269
/^vlinie(layer=([0-9]+))?(farbe=([a-zAEOEUE0-9!]+))?(linie=([0-9\.]�),([a-z0-9!]+))	5270
?(\\"([^\"]*)\\")?(parameter=\\"([^\"]*)\\")? p=([0-9\.]�),([0-9\.]�),([0-9\.]�))? r	

```

=([0-9\..]+)([0-9\..]+)([0-9\..]+)? text=\"([^\"]*)\"([0-9\..]+)([0-9\..]+)([0-9\..]+)?([0-9\..]+)
)?([0-9\..]+)([0-9\..]+)([0-9\..]+)?([0-9\..]+)?$/i
)
{
  standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
  $layer = $2 if ( defined $1 );
  $farbe = $4 if ( defined $3 );

  if ( defined $5 ) {
    $linienstaerke = $6;
    $linienart      = $7;
  }

  $pfeilart = $9 if ( defined $8 );

  if ( defined $10 ) { $parameter = $11; }

  $x1 = $12;
  $y1 = $13;
  $z1 = "";
  $z2 = "";
  $z1 = $15 if ( defined $14 );

  $x2 = $x1 + $16;
  $y2 = $y1 + $17;
  $z2 = $z1 + $19 if ( defined $18 );

  $text      = $20;
  $winkel    = 0;
  $winkel    = $22 if ( defined $21 );
  $ausrichtung = 0;
  $ausrichtung = $24 if ( defined $23 );
  $abstandx   = 0.2;
  $abstandx   = $26 if ( defined $25 );
  $abstandy   = $abstandx;

  $abstandy = $28 if ( defined $27 );
  $abstandz = $abstandx;
  $abstandz = $30 if ( defined $29 );

  linietext (
    $layer, $x1,      $y1,      $z1,      $x2,
    $y2,      $z2,      $abstandx, $abstandy, $abstandz,
    $text, $winkel, $ausrichtung, $linienart, $linienstaerke,
    $farbe, $parameter, $pfeilart
  );
}

elseif ( $zeile =~
/^vektor(layer=([0-9\..]+))(farbe=([a-zaeoue0-9\..]+))(linie=([0-9\..]+)([a-z0-9\..]+)
?\"([^\"]*)\")?(parameter=\"([^\"]*)\")?p=([0-9\..]+)([0-9\..]+)([0-9\..]+)?p
=([0-9\..]+)([0-9\..]+)([0-9\..]+)?$/i
)
{
  standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "->",
    "" );
  $layer = $2 if ( defined $1 );
  $farbe = $4 if ( defined $3 );

  if ( defined $5 ) {

```

5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283
5284
5285
5286
5287
5288
5289
5290
5291
5292
5293
5294
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304
5305
5306
5307
5308
5309
5310
5311
5312
5313
5314
5315
5316
5317
5318
5319

5320
5321
5322
5323
5324
5325
5326
5327

\$linienstaerke = \$6;	5328
\$linienart = \$7;	5329
}	5330
\$pfeilart = \$9 if (defined \$8);	5331
	5332
if (defined \$10) { \$parameter = \$11; }	5333
	5334
\$x1 = \$12;	5335
\$y1 = \$13;	5336
\$z1 = "";	5337
\$z1 = \$15 if (defined \$14);	5338
\$x2 = \$16;	5339
\$y2 = \$17;	5340
\$z2 = "";	5341
\$z2 = \$19 if (defined \$18);	5342
	5343
linie(\$layer, \$x1, \$y1, \$z1, \$x2, \$y2, \$z2, \$linienart, \$linienstaerke, \$farbe,	5344
\$parameter, \$pfeilart);	5345
	5346
}	5347
	5348
elseif (\$zeile =~	5349
/^vektor(layer=([0-9]+)?)? farbe=([a-zaeoue0-9!]+)?)? linie=([0-9\.] +, ([a-z0-9!]+))	5350
?(\\"([^\"]*)\")? parameter=\\\"([^\"]*)\")? p=([0-9\.] +, ([0-9\.] +, ([0-9\.] +)?)? p	
=([0-9\.] +, ([0-9\.] +, ([0-9\.] +)?)? text=\\\"([^\"]*)\\"([0-9\.] +)?)?(\\"([a-z0-9!]+)	
)?(\\"([0-9\.] +)?)?(\\"([0-9\.] +)?)?(\\"([0-9\.] +)?)?\$/i	5351
)	5352
{	5353
standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), ">",	5354
"");	5355
\$layer = \$2 if (defined \$1);	5356
\$farbe = \$4 if (defined \$3);	5357
	5358
if (defined \$5) {	5359
\$linienstaerke = \$6;	5360
\$linienart = \$7;	5361
}	5362
\$pfeilart = \$9 if (defined \$8);	5363
	5364
if (defined \$10) { \$parameter = \$11; }	5365
	5366
\$x1 = \$12;	5367
\$y1 = \$13;	5368
\$z1 = "";	5369
\$z1 = \$15 if (defined \$14);	5370
\$x2 = \$16;	5371
\$y2 = \$17;	5372
\$z2 = "";	5373
\$z2 = \$19 if (defined \$18);	5374
\$text = \$20;	5375
\$winkel = 0;	5376
\$winkel = \$22 if (defined \$21);	5377
\$ausrichtung = 0;	5378
\$ausrichtung = \$24 if (defined \$23);	5379
\$abstandx = 0.2;	5380
\$abstandx = \$26 if (defined \$25);	5381
\$abstandy = \$abstandx;	5382
	5383
\$abstandy = \$28 if (defined \$27);	5384
\$abstandz = \$abstandx;	5385
\$abstandz = \$30 if (defined \$29);	

```

linietext (
    $layer, $x1,      $y1,      $z1,      $x2,
    $y2,  $z2,      $abstandx, $abstandy, $abstandz,
    $text, $winkel,  $ausrichtung, $linienart, $linienstaerke,
    $farbe, $parameter, $pfeilart
);

}

elseif ( $zeile =~
/^vektor(layer=([0-9]+))?(farbe=([a-zaeoue0-9!]+))?(linie=([0-9\.]+),([a-z0-9!]+))
?(\\"([^\"]*)\\")?parameter=\\"([^\"]*)\\")?p=([0-9\.]+),([0-9\.]+),([0-9\.]+))?p
=([0-9\.]+),([0-9\.]+),([0-9\.]+))?r=([0-9\.]+)(text=\\"([^\"]*)\\"([0-9\.]+))
?(\\"([a-z0-9!]+))?(\\([0-9\.]+))?(\\([0-9\.]+))?(\\([0-9\.]+))?$ /i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "->",
    "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart      = $7;
    }
    $pfeilart = $9 if ( defined $8 );

    if ( defined $10 ) { $parameter = $11; }

    $x1      = $12;
    $y1      = $13;
    $z1      = "";
    $z1      = $15 if ( defined $14 );
    $x2      = $16;
    $y2      = $17;
    $z2      = "";
    $z2      = $19 if ( defined $18 );
    $abstand = $20;
    $text     = "";
    $abstandx = 0.2;
    $ausrichtung = 0;
    $winkel    = 0;
    $abstandy  = $abstandz = $abstandx;
    if ( defined $21 ) {
        $text = $22;

        $winkel = $24 if ( defined $23 );

        $ausrichtung = $26 if ( defined $25 );

        $abstandx = $28 if ( defined $27 );

        $abstandy = $30 if ( defined $29 );

        $abstandz = $32 if ( defined $31 );
    }

    liniertextabstand(
        $layer,      $x1,  $y1,      $z1,      $x2,
        $y2,          $z2,  $abstand, $abstandx, $abstandy,

```

<pre> \$abstandz, \$text, \$winkel, \$ausrichtung, \$linienart, \$linienstaerke, \$farbe, \$parameter, \$pfeilart); } elseif (\$zeile =~ /^vektor(layer=([0-9]+)?) (farbe=([a-zA-Z0-9!]+)?) (linie=([0-9\.\.]+),([a-z0-9!]+) ?(\\"([^\"]*)\\")? (parameter=\\"([^\"]*)\\")? p=([0-9\.\.]+),([0-9\.\.]+),([0-9\.\.]+)?) r =([0-9\.\.]+),([0-9\.\.]+),([0-9\.\.]+)?) r=([0-9\.\.]+) (text=\\"([^\"]*)\\")? (([0-9\.\.]+) ?(\\"([a-z0-9!]+)?)?(\\"([0-9\.\.]+)?)?(\\"([0-9\.\.]+)?)?(\\"([0-9\.\.]+)?)?)?\$/i) { standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "->", ""); \$layer = \$2 if (defined \$1); \$farbe = \$4 if (defined \$3); if (defined \$5) { \$linienstaerke = \$6; \$linienart = \$7; } \$pfeilart = \$9 if (defined \$8); if (defined \$10) { \$parameter = \$11; } \$xl = \$12; \$yl = \$13; \$zl = ""; \$zl = \$15 if (defined \$14); \$laenge = \$16; \$w1 = \$17; \$w2 = 0; \$w2 = \$19 if (defined \$18); \$abstand = \$20; \$text = ""; \$abstandx = 0.2; \$ausrichtung = 0; \$winkel = 0; \$abstandy = \$abstandz = \$abstandx; if (defined \$21) { \$text = \$22; \$winkel = \$24 if (defined \$23); \$ausrichtung = \$26 if (defined \$25); \$abstandx = \$28 if (defined \$27); \$abstandy = \$30 if (defined \$29); \$abstandz = \$32 if (defined \$31); } liniepolarabstand(\$layer, \$xl, \$yl, \$zl, \$laenge, \$w1, \$w2, \$abstand, \$abstandx, \$abstandy, \$abstandz, \$text, \$winkel, \$ausrichtung, \$linienart, \$linienstaerke, \$farbe, \$parameter, \$pfeilart); </pre>	5444 5445 5446 5447 5448 5449 5450 5451 5452 5453 5454 5455 5456 5457 5458 5459 5460 5461 5462 5463 5464 5465 5466 5467 5468 5469 5470 5471 5472 5473 5474 5475 5476 5477 5478 5479 5480 5481 5482 5483 5484 5485 5486 5487 5488 5489 5490 5491 5492 5493 5494 5495 5496 5497 5498 5499 5500 5501
--	---

```

}
5502
5503
    elsif ( $zeile =~
5504
/^vektor (layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+))?( linie=([0-9\.]�+),([a-z0-9!]+))
5505
?(\\"([^\"]*)\\")?( parameter=\\"([^\"]*)\\")? p=([0-9\.]�+),([0-9\.]�+),([0-9\.]�+))? r
=([0-9\.]�+),([0-9\.]�+),([0-9\.]�+))? text=\\"([^\"]*)\\"([0-9\.]�+)?(,[a-z0-9!]+)
)?(,[0-9\.]�+)?(,[0-9\.]�+)?(,[0-9\.]�+))?$ /i
5506
)
5507
{
5508
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "->",
5509
        "" );
5510
    $layer = $2 if ( defined $1 );
5511
    $farbe = $4 if ( defined $3 );
5512
5513
    if ( defined $5 ) {
5514
        $linienstaerke = $6;
5515
        $linienart      = $7;
5516
    }
5517
    $pfeilart = $9 if ( defined $8 );
5518
5519
    if ( defined $10 ) { $parameter = $11; }
5520
5521
    $x1 = $12;
5522
    $y1 = $13;
5523
    $z1 = "";
5524
    $z1 = $15 if ( defined $14 );
5525
5526
    $laenge = $16;
5527
    $w1      = $17;
5528
    $w2      = 0;
5529
    $w2      = $19 if ( defined $18 );
5530
5531
    $text      = $20;
5532
    $winkel    = 0;
5533
    $winkel    = $22 if ( defined $21 );
5534
    $ausrichtung = 0;
5535
    $ausrichtung = $24 if ( defined $23 );
5536
    $abstandx   = 0.2;
5537
    $abstandx   = $26 if ( defined $25 );
5538
    $abstandy   = $abstandx;
5539
5540
    $abstandy = $28 if ( defined $27 );
5541
    $abstandz = $abstandx;
5542
    $abstandz = $30 if ( defined $29 );
5543
5544
    liniertextpolar (
5545
        $layer, $x1,      $y1,      $z1,      $laenge,
5546
        $w1,   $w2,      $abstandx, $abstandy, $abstandz,
5547
        $text, $winkel,  $ausrichtung, $linienart, $linienstaerke,
5548
        $farbe, $parameter, $pfeilart
5549
    );
5550
}
5551
5552
    elsif ( $zeile =~
5553
/^vektor (layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+))?( linie=([0-9\.]�+),([a-z0-9!]+))
?(\\"([^\"]*)\\")?( parameter=\\"([^\"]*)\\")? p=([0-9\.]�+),([0-9\.]�+),([0-9\.]�+))? r
=([0-9\.]�+),([0-9\.]�+),([0-9\.]�+))?$ /i
5554
)
5555
{
5556
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "->",
5557
        "" );

```

\$layer = \$2 if (defined \$1);	5558
\$farbe = \$4 if (defined \$3);	5559
	5560
if (defined \$5) {	5561
\$linienstaerke = \$6;	5562
\$linienart = \$7;	5563
}	5564
	5565
\$pfeilart = \$9 if (defined \$8);	5566
if (defined \$10) { \$parameter = \$11; }	5567
	5568
\$x1 = \$12;	5569
\$y1 = \$13;	5570
\$z1 = "";	5571
\$z1 = \$15 if (defined \$14);	5572
\$laenge = \$16;	5573
\$w1 = \$17;	5574
\$w2 = "";	5575
\$w2 = \$19 if (defined \$18);	5576
liniepolar (	5577
\$layer, \$x1, \$y1, \$z1,	5578
\$laenge, \$w1, \$w2, \$linienart ,	5579
\$linienstaerke, \$farbe, \$parameter, \$pfeilart	5580
);	5581
}	5582
	5583
elseif (\$zeile =~	5584
/^vlinie(layer=([0-9]+)?) (farbe=([a-zaeoue0-9!]+)?) (linie=([0-9\.] +)([a-z0-9!]+))	5585
? (\\"([^\"]*)\\")? (parameter=\\"([^\"]*)\\")? p=([0-9\.] +)([0-9\.] +)([0-9\.] +)? r	
=([0-9\.] +)([0-9\.] +)([0-9\.] +)? text=\\"([^\"]*)\\" ([0-9\.] +)? ([a-z0-9!]+)	
)? ([0-9\.] +)? ([0-9\.] +)? ([0-9\.] +)? \$/i	
)	5586
{	5587
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	5588
"");	5589
\$layer = \$2 if (defined \$1);	5590
\$farbe = \$4 if (defined \$3);	5591
	5592
if (defined \$5) {	5593
\$linienstaerke = \$6;	5594
\$linienart = \$7;	5595
}	5596
	5597
\$pfeilart = \$9 if (defined \$8);	5598
	5599
if (defined \$10) { \$parameter = \$11; }	5600
	5601
\$x1 = \$12;	5602
\$y1 = \$13;	5603
\$z1 = "";	5604
\$z1 = \$15 if (defined \$14);	5605
	5606
\$x2 = \$x1 + \$16;	5607
\$y2 = \$y1 + \$17;	5608
\$z2 = \$z1 + \$19 if (defined \$18);	5609
	5610
\$text = \$20;	5611
\$winkel = 0;	5612
\$winkel = \$22 if (defined \$21);	5613
\$ausrichtung = 0;	5614
\$ausrichtung = \$24 if (defined \$23);	5615

```

$abstandx = 0.2;
$abstandx = $26 if ( defined $25 );
$abstandy = $abstandx;

$abstandy = $28 if ( defined $27 );
$abstandz = $abstandx;
$abstandz = $30 if ( defined $29 );

linietext (
    $layer, $x1,      $y1,      $z1,      $x2,
    $y2,  $z2,      $abstandx, $abstandy, $abstandz,
    $text, $winkel,  $ausrichtung, $linienart, $linienstaerke,
    $farbe, $parameter, $pfeilart
);
}

elsif ( $zeile =~
/^vlinie( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+))?( linie=([0-9\.\.]+),([a-z0-9!]+))
?(\\"([^\"]*)\\")?( parameter=\\"([^\"]*)\\")? p=([-0-9\.\.]+),([-0-9\.\.]+),([-0-9\.\.]+))? r
=([-0-9\.\.]+),([-0-9\.\.]+),([-0-9\.\.]+))?$ /i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
        "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {
        $linienstaerke = $6;
        $linienart      = $7;
    }

    $pfeilart = $9 if ( defined $8 );

    if ( defined $10 ) { $parameter = $11; }

    $x1 = $12;
    $y1 = $13;
    $z1 = "";
    $z1 = $15 if ( defined $14 );

    $x2 = $x1 + $16;
    $y2 = $y1 + $17;
    $z2 = $z1 + $19 if ( defined $18 );

    linie ( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $linienart, $linienstaerke, $farbe,
        $parameter, $pfeilart );
}

elsif ( $zeile =~
/^liniepolar( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+))?( linie=([0-9\.\.]+),([a-z0-9!]+))
?(\\"([^\"]*)\\")?( parameter=\\"([^\"]*)\\")? p=([-0-9\.\.]+),([-0-9\.\.]+),([-0-9\.\.]+))? r
=([-0-9\.\.]+),([-0-9\.\.]+),([-0-9\.\.]+))?$ /i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "-",
        "" );
    $layer = $2 if ( defined $1 );
    $farbe = $4 if ( defined $3 );

    if ( defined $5 ) {

```

\$linienstaerke = \$6;	5673
\$linienart = \$7;	5674
}	5675
\$pfeilart = \$9 if (defined \$8);	5676
if (defined \$10) { \$parameter = \$11; }	5677
	5678
\$x1 = \$12;	5679
\$y1 = \$13;	5680
\$z1 = "";	5681
\$z1 = \$15 if (defined \$14);	5682
\$laenge = \$16;	5683
\$w1 = \$17;	5684
\$w2 = "";	5685
\$w2 = \$19 if (defined \$18);	5686
liniepolar (	5687
\$layer, \$x1, \$y1, \$z1,	5688
\$laenge, \$w1, \$w2, \$linienart ,	5689
\$linienstaerke, \$farbe, \$parameter, \$pfeilart	5690
);	5691
}	5692
	5693
elseif (\$zeile =~	5694
/^gerade(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+))?(linie=([0-9\.]+),([a-z0-9!]+))	5695
?(\\"([^\"]*)\\")?(parameter=\\"([^\"]*)\\")? p=([0-9\.]+),([0-9\.]+),([0-9\.]+))? p	
=([0-9\.]+),([0-9\.]+),([0-9\.]+))? r=([0-9\.]+),([0-9\.]+)\$/i	
)	5696
{	5697
standard(10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",	5698
"");	5699
\$layer = \$2 if (defined \$1);	5700
\$farbe = \$4 if (defined \$3);	5701
	5702
if (defined \$5) {	5703
\$linienstaerke = \$6;	5704
\$linienart = \$7;	5705
}	5706
	5707
\$pfeilart = \$9 if (defined \$8);	5708
	5709
if (defined \$10) { \$parameter = \$11; }	5710
	5711
\$x1 = \$12;	5712
\$y1 = \$13;	5713
\$z1 = "";	5714
\$z1 = \$15 if (defined \$14);	5715
\$x2 = \$16;	5716
\$y2 = \$17;	5717
\$z2 = "";	5718
\$z2 = \$19 if (defined \$18);	5719
\$r1 = \$20;	5720
\$r2 = \$21;	5721
setachsengraph3d(0, 0, 0, 0, 0, 0);	5722
minmax3d(0, 0, 0, 0, 0, 0);	5723
gerade(\$layer, \$x1, \$y1, \$z1, \$x2, \$y2, \$z2, \$r1, \$r2, \$farbe, \$linienart ,	5724
\$linienstaerke, \$parameter, \$pfeilart);	5725
	5726
}	5727
	5728
elseif (\$zeile =~	5729
/^ebene(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	5730
=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter=\\"([^\"]*)\\")? p	

```

=([−0−9\\.]+),([−0−9\\.]+),([−0−9\\.]+))? p=([−0−9\\.]+),([−0−9\\.]+),([−0−9\\.]+))? p
=([−0−9\\.]+),([−0−9\\.]+),([−0−9\\.]+))?( r=([−0−9\\.]+),([−0−9\\.]+))?( r=([−0−9\\.]+)
,[−0−9\\.]+))?$ /i
)
{

setachsengraph3d( 0, 0, 0, 0, 0, 0 );
standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
"" );
$layer = $2 if ( defined $1 );

if ( defined $3 ) {
    $farbe      = $4;
    $fuellart    = $5;
    $fuellfarbe = $6;
}

if ( defined $7 ) {
    $linienstaerke = $8;
    $linienart     = $9;
}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$x1 = $14;
$y1 = $15;
$z1 = "";
$z1 = $17 if ( defined $16 );
$x2 = $18;
$y2 = $19;
$z2 = "";
$z2 = $21 if ( defined $20 );

$x3 = $22;
$y3 = $23;
$z3 = "";
$z3 = $25 if ( defined $24 );

if ( defined $26 ) {
    $r1 = $27;
    $r2 = $28;
}
else {
    $r1 = 0;
    $r2 = 1;
}

if ( defined $29 ) {
    $r3 = $30;
    $r4 = $31;
}
else {
    $r3 = 0;
    $r4 = 1;
}

}

kommentar( $layer, "Eine Ebene zeichnen" );

ebene( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $x3, $y3, $z3, $r1, $r2, $r3, $r4, $farbe,
    $fuellart, $fuellfarbe, $linienart, $linienstaerke, $parameter );

```

5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788

	5789
}	5790
	5791
elseif (\$zeile =~	5792
/^ebene(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	5793
=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\("[^"]*")? p	
=([0-9\..]+),([0-9\..]+),([0-9\..]+))? p=([0-9\..]+),([0-9\..]+),([0-9\..]+))?(r	
=([0-9\..]+),([0-9\..]+))?(r=([0-9\..]+),([0-9\..]+))?\$ /i	
)	5794
{	5795
	5796
setachsengraph3d(0, 0, 0, 0, 0, 0);	5797
standard(10, getlinienart (), getlinienstaerke (), getfarbe (), "", getfuellfarbe (), "",	5798
"");	5799
\$layer = \$2 if (defined \$1);	5800
	5801
if (defined \$3) {	5802
\$farbe = \$4;	5803
\$fuellart = \$5;	5804
\$fuellfarbe = \$6;	5805
}	5806
	5807
if (defined \$7) {	5808
\$linienstaerke = \$8;	5809
\$linienart = \$9;	5810
}	5811
\$farbe = \$11 if (defined \$10);	5812
if (defined \$12) { \$parameter = \$13; }	5813
	5814
\$x1 = \$14;	5815
\$y1 = \$15;	5816
\$z1 = "";	5817
\$z1 = \$17 if (defined \$16);	5818
\$x2 = \$18;	5819
\$y2 = \$19;	5820
\$z2 = "";	5821
\$z2 = \$21 if (defined \$20);	5822
if (defined \$22) {	5823
\$r1 = \$23;	5824
\$r2 = \$24;	5825
}	5826
else {	5827
\$r1 = 0;	5828
\$r2 = 1;	5829
}	5830
	5831
if (defined \$25) {	5832
\$r3 = \$26;	5833
\$r4 = \$27;	5834
}	5835
else {	5836
\$r3 = 0;	5837
\$r4 = 1;	5838
}	5839
	5840
	5841
kommentar(\$layer, "Eine Ebene zeichnen");	5842
	5843
ebenenormal(\$layer, \$x1, \$y1, \$z1, \$x2, \$y2, \$z2, \$r1, \$r2, \$r3, \$r4, \$farbe,	5844
\$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke, \$parameter);	5845
	5846

```

}
elseif ( $zeile =~
/^ebene( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.] +),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\([^"]*\)" )? [k l r
]=([0-9\.] +),([0-9\.] +),([0-9\.] +),([0-9\.] +),([0-9\.] +))?(,[0-9\.] +))?$ /i
)
{
setachsengraph3d( 0, 0, 0, 0, 0, 0 );
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
$layer = $2 if ( defined $1 );

if ( defined $3 ) {
$farbe = $4;
$fuellart = $5;
$fuellfarbe = $6;
}

if ( defined $7 ) {
$linienstaerke = $8;
$linienart = $9;
}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$a = $14;
$b = $15;
$c = $16;
$d = $17;
$n1 = $n2 = 1;
if ( defined $18 ) { $n1 = $19; }
if ( defined $20 ) { $n2 = $21; }

kommentar( $layer,
"Die Ebene " . $a . "x+" . $b . "y+" . $c . "z=" . $d . " zeichnen" );

ebenekform(
$layer, $a, $b, $c, $d, $n1, $n2, $farbe, $fuellart,
$fuellfarbe, $linienart, $linienstaerke, $parameter

);
}

elseif ( $zeile =~
/^quader( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.] +),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\([^"]*\)" )? p
=([0-9\.] +),([0-9\.] +),([0-9\.] +) r=([0-9\.] +),([0-9\.] +),([0-9\.] +)
,([0-9\.] +),([0-9\.] +))?$ /i
)
{
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
$layer = $2 if ( defined $1 );

if ( defined $3 ) {
$farbe = $4;
$fuellart = $5;

```

\$fuellfarbe = \$6;	5903
}	5904
if (defined \$7) {	5905
\$linienstaerke = \$8;	5906
\$linienart = \$9;	5907
}	5908
\$farbe = \$11 if (defined \$10);	5909
if (defined \$12) { \$parameter = \$13; }	5910
	5911
	5912
\$x1 = \$14;	5913
\$y1 = \$15;	5914
\$z1 = \$16;	5915
\$a = \$17;	5916
\$b = \$18;	5917
\$c = \$19;	5918
	5919
\$w1 = \$w2 = 0;	5920
if (defined \$20) {	5921
\$w1 = \$21;	5922
\$w2 = \$22;	5923
}	5924
	5925
quader(	5926
\$layer, \$x1, \$y1, \$z1, \$a, \$b, \$c, \$w1, \$w2, \$farbe,	5927
\$fuellart ,	5928
\$fuellfarbe , \$linienart , \$linienstaerke ,	5929
\$parameter	5930
)	5931
);	5932
	5933
}	5934
	5935
elseif (\$zeile =~	5936
/^pyramide(layer=([0-9]+)?) (farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	5937
=([0-9\.]+),([a-z0-9!]+))?(([a-zaeoeue0-9!]+))?(parameter=\"([^\"]*)\")? p	
=([0-9\.]+),([0-9\.]+),([0-9\.]+) r=([0-9\.]+),([0-9\.]+),([0-9\.]+)	
(,[0-9\.]+),([0-9\.]+))?\$/i	
)	5938
{	5939
	5940
standard(10, getlinienart () , getlinienstaerke () , getfarbe () , "" , getfuellfarbe () , "" ,	5941
"");	5942
\$layer = \$2 if (defined \$1);	5943
	5944
if (defined \$3) {	5945
\$farbe = \$4;	5946
\$fuellart = \$5;	5947
\$fuellfarbe = \$6;	5948
}	5949
	5950
if (defined \$7) {	5951
\$linienstaerke = \$8;	5952
\$linienart = \$9;	5953
}	5954
\$farbe = \$11 if (defined \$10);	5955
if (defined \$12) { \$parameter = \$13; }	5956
	5957
	5958
\$x1 = \$14;	5959
\$y1 = \$15;	5960
\$z1 = \$16;	

```

$a = $17;
$b = $18;
$c = $19;
$w1 = $w2 = 0;

if ( defined $20 ) {
    $w1 = $21;
    $w2 = $22;
}

kommentar( $layer, "Eine Pyramide zeichnen" );

pyramide(
    $layer, $x1, $y1, $z1, $a, $b, $c, $w1, $w2, $farbe,
    $fuellart,
    $fuellfarbe, $linienart, $linienstaerke, $parameter

);

}

elsif ( $zeile =~
/^pyramidestumpf( layer=([0-9]+))?( farbe=([a-zAEOeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(
    linie=([0-9\.]+),([a-z0-9!]+))?(,[a-zAEOeue0-9!]+))?( parameter="\([^"]*\)" )? p
=([0-9-9\.]+),([0-9-9\.]+),([0-9-9\.]+) r=([0-9-9\.]+),([0-9-9\.]+),([0-9-9\.]+)
,([0-9-9\.]+),([0-9-9\.]+),([0-9-9\.]+))?$ /i
)
{

    standard( 10, getlinienart (), getlinienstaerke (), getfarbe (), "", getfuellfarbe (), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $x1      = $14;
    $y1      = $15;
    $z1      = $16;
    $a       = $17;
    $b       = $18;
    $c       = $19;
    $abgeschnitten = $20;
    $w1      = $w2 = 0;

    if ( defined $21 ) {
        $w1 = $22;
        $w2 = $23;
    }

    kommentar( $layer, "Einen Pyramidenstumpf zeichnen" );

```

	6019
pyramidestumpf(	6020
\$layer, \$x1, \$y1, \$z1, \$a, \$b, \$c, \$abgeschnitten, \$w1, \$w2,	6021
\$farbe,	6022
\$fuellart ,	6023
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter	6024
	6025
);	6026
	6027
}	6028
	6029
elseif (\$zeile =~	6030
/^kegel(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	6031
=([0-9\.]+),([a-z0-9!]+))?(,([a-zaeoeue0-9!]+))?(parameter="\([^"]*\)"? p	
=([0-9\.]+),([0-9\.]+),([0-9\.]+) r=([0-9\.]+),([0-9\.]+),([0-9\.]+))?\$ /i	
)	6032
{	6033
	6034
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	6035
"");	6036
\$layer = \$2 if (defined \$1);	6037
	6038
if (defined \$3) {	6039
\$farbe = \$4;	6040
\$fuellart = \$5;	6041
\$fuellfarbe = \$6;	6042
}	6043
	6044
if (defined \$7) {	6045
\$linienstaerke = \$8;	6046
\$linienart = \$9;	6047
}	6048
\$farbe = \$11 if (defined \$10);	6049
if (defined \$12) { \$parameter = \$13; }	6050
	6051
\$x1 = \$14;	6052
\$y1 = \$15;	6053
\$z1 = \$16;	6054
\$a = \$17;	6055
\$b = \$18;	6056
	6057
\$w1 = 90;	6058
if (defined \$19) { \$w1 = \$20; }	6059
	6060
kommentar(\$layer, "Einen Kegel zeichnen");	6061
	6062
kegel(	6063
\$layer, \$x1, \$y1, \$z1, \$a, \$b, \$w1, \$farbe,	6064
\$fuellart ,	6065
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter	6066
	6067
);	6068
	6069
}	6070
	6071
elseif (\$zeile =~	6072
/^kegelstumpf(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	6073
=([0-9\.]+),([a-z0-9!]+))?(,([a-zaeoeue0-9!]+))?(parameter="\([^"]*\)"? p	
=([0-9\.]+),([0-9\.]+),([0-9\.]+) r=([0-9\.]+),([0-9\.]+),([0-9\.]+)	
,([0-9\.]+))?\$ /i	
)	6074

```

{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart   = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $x1      = $14;
    $y1      = $15;
    $z1      = $16;
    $r       = $17;
    $c       = $18;
    $abgeschnitten = $19;

    $w1 = 30;
    if ( defined $20 ) { $w1 = $21; }

    kommentar( $layer, "Einen Kegelstumpf zeichnen" );

    kegelstumpf(
        $layer, $x1, $y1, $z1, $r, $c, $abgeschnitten, $w1, $farbe,
        $fuellart,
        $fuellfarbe, $linienart, $linienstaerke, $parameter

    );
}

elsif ( $zeile =~
/^zylinder( layer=([0-9]+))?( farbe=([a-zA-Z0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.\.]+),([a-z0-9!]+))?(,[a-zA-Z0-9!]+))?( parameter="\([^"]*\)" )? p
=([0-9\.\.]+),([0-9\.\.]+),([0-9\.\.]+) r=([0-9\.\.]+),([0-9\.\.]+),([0-9\.\.]+))?$ /i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart   = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
}

```

\$farbe = \$11 if (defined \$10);	6134
if (defined \$12) { \$parameter = \$13; }	6135
	6136
\$x1 = \$14;	6137
\$y1 = \$15;	6138
\$z1 = \$16;	6139
\$a = \$17;	6140
\$b = \$18;	6141
	6142
\$w1 = 30;	6143
if (defined \$19) { \$w1 = \$20; }	6144
kommentar(\$layer, "Einen Zylinder zeichnen");	6145
	6146
zylinder(	6147
\$layer, \$x1, \$y1, \$z1, \$a, \$b, \$w1, \$farbe,	6148
\$fuellart ,	6149
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter	6150
)	6151
);	6152
	6153
}	6154
	6155
elseif (\$zeile =~	6156
/^kugel(layer=([0-9]+))?(farbe=([a-zaeooue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	6157
=([0-9\.]+),([a-z0-9!]+))?(,([a-zaeooue0-9!]+))?(parameter="\^[^"]*)")? p	
=([0-9\.]+),([0-9\.]+),([0-9\.]+) r=([0-9\.]+))\$/i	
)	6158
{	6159
set3dpstricks(1);	6160
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	6161
"");	6162
\$layer = \$2 if (defined \$1);	6163
	6164
if (defined \$3) {	6165
\$farbe = \$4;	6166
\$fuellart = \$5;	6167
\$fuellfarbe = \$6;	6168
}	6169
	6170
if (defined \$7) {	6171
\$linienstaerke = \$8;	6172
\$linienart = \$9;	6173
}	6174
\$farbe = \$11 if (defined \$10);	6175
if (defined \$12) { \$parameter = \$13; }	6176
	6177
\$x = \$14;	6178
\$y = \$15;	6179
\$z = \$16;	6180
\$r = \$17;	6181
	6182
kommentar(\$layer, "Eine Kugel zeichnen");	6183
	6184
kugel(	6185
\$layer, \$x, \$y, \$z, \$r, \$farbe, \$fuellart ,	6186
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter	6187
)	6188
);	6189
	6190
}	6191
	6192

```

elseif ( $zeile =~
/^kreis( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoue0-9!]+))?( parameter="\([^"]*\)" )? p
=([0-9\.]+),([a-z0-9!]+)( r=([0-9\.]+))?( text="\([^"]*\)" ,([a-z0-9!]+))?(,[a-z0
-9!]+))?)?$/i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
    "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart      = $9;
    }
    $farbe = $11 if ( defined $10 );

    if ( defined $12 ) { $parameter = $13; }

    $x      = $14;
    $y      = $15;
    $radius = 1;
    $radius = $17 if ( defined $16 );

    kreis(
        $layer, $x, $y, $radius, $farbe,
        $fuellart, $fuellfarbe, $linienart, $linienstaerke, $parameter
    );
    $winkel      = 0;
    $ausrichtung = "c";
    if ( defined $18 ) {
        $text = $19;

        $winkel = $21 if ( defined $20 );

        $ausrichtung = $23 if ( defined $22 );

        beschriftung( $layer, $farbe, $x, $y, $winkel, $ausrichtung, $text );
    }
}

elseif ( $zeile =~
/^kreisskala( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+))?( linie=([0-9\.]+),([a-z0-9!]+))
? ,("\([^"]*\)" )? parameter="\([^"]*\)" )? p=([0-9\.]+),([a-z0-9!]+)( r=([0-9\.]+)
,[a-z0-9!]+),([a-z0-9!]+))?)?$/i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "- ",
    "" );
    $layer = $2 if ( defined $1 );

    $farbe = $4 if ( defined $3 );

```

if (defined \$5) {	6249
\$linienstaerke = \$6;	6250
\$linienart = \$7;	6251
}	6252
\$pfeilart = \$9 if (defined \$8);	6253
if (defined \$10) { \$parameter = \$11; }	6254
	6255
\$x = \$12;	6256
\$y = \$13;	6257
\$radius = 1;	6258
\$laenge = 0.2;	6259
\$winkel = 180;	6260
if (defined \$14) {	6261
\$radius = \$15;	6262
\$laenge = \$16;	6263
\$winkel = \$17;	6264
}	6265
	6266
kreisskala(	6267
\$layer, \$x, \$y, \$radius,	6268
\$laenge, \$winkel, \$farbe, \$linienart ,	6269
\$linienstaerke, \$parameter, \$pfeilart	6270
);	6271
	6272
}	6273
	6274
elseif (\$zeile =~	6275
/^ellipse(layer=([0-9!]+))(farbe=([a-zAEOEUE0-9!]+),([a-z0-9!]+),([a-z0-9!]+))(linie	6276
=([0-9!.]+),([a-z0-9!]+))(,([a-zAEOEUE0-9!]+))(parameter="\([^"]*\)")? p	
=([0-9!.]+),([0-9!.]+) (r=([0-9!.]+),([0-9!.]+))?\$ /i	
)	6277
{	6278
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	6279
"");	6280
\$layer = \$2 if (defined \$1);	6281
	6282
if (defined \$3) {	6283
\$farbe = \$4;	6284
\$fuellart = \$5;	6285
\$fuellfarbe = \$6;	6286
}	6287
	6288
if (defined \$7) {	6289
\$linienstaerke = \$8;	6290
\$linienart = \$9;	6291
}	6292
\$farbe = \$11 if (defined \$10);	6293
if (defined \$12) { \$parameter = \$13; }	6294
	6295
\$x = \$14;	6296
\$y = \$15;	6297
\$radius1 = 1;	6298
\$radius2 = 1;	6299
	6300
if (defined \$16) {	6301
\$radius1 = \$17;	6302
\$radius2 = \$18;	6303
}	6304
ellipse (	6305
\$layer, \$x, \$y, \$radius1,	6306
\$radius2, \$farbe, \$fuellart , \$fuellfarbe ,	6307

```

$linienart, $linienstaerke, $parameter
);
}

elseif ( $zeile =~
/^ellipse( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\^[^"]*)"?)? p
=([0-9\.]+),([0-9\.]+),([0-9\.]+) r=([0-9\.]+),([0-9\.]+),([0-9\.]+) r
=([0-9\.]+),([0-9\.]+),([0-9\.]+)( r=([0-9\.]+),([0-9\.]+))?)$/i
)
{
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
$layer = $2 if ( defined $1 );

if ( defined $3 ) {
$farbe = $4;
$fuellart = $5;
$fuellfarbe = $6;
}

if ( defined $7 ) {
$linienstaerke = $8;
$linienart = $9;
}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$x = $14;
$y = $15;
$z = $16;

$r1 = $17;
$r2 = $18;
$r3 = $19;
$r4 = $20;
$r5 = $21;
$r6 = $22;

$w1 = 0;
$w2 = 360;
if ( defined $23 ) {
$w1 = $24;
$w2 = $25;
}

ellipse3d( $layer, $x, $y, $z, $r1, $r2, $r3, $r4, $r5, $r6, $w1, $w2, $farbe,
$fuellart, $fuellfarbe, $linienart, $linienstaerke, $parameter );
}

elseif ( $zeile =~
/^kreis( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\^[^"]*)"?)? p
=([0-9\.]+),([0-9\.]+),([0-9\.]+) r=([0-9\.]+),([0-9\.]+),([0-9\.]+)( r
=([0-9\.]+),([0-9\.]+))?)$/i
)
{
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
}

```

\$layer = \$2 if (defined \$1);	6363
	6364
if (defined \$3) {	6365
\$farbe = \$4;	6366
\$fuellart = \$5;	6367
\$fuellfarbe = \$6;	6368
}	6369
	6370
if (defined \$7) {	6371
\$linienstaerke = \$8;	6372
\$linienart = \$9;	6373
}	6374
\$farbe = \$11 if (defined \$10);	6375
if (defined \$12) { \$parameter = \$13; }	6376
	6377
\$x = \$14;	6378
\$y = \$15;	6379
\$z = \$16;	6380
	6381
\$r1 = \$17;	6382
\$r2 = \$18;	6383
\$r3 = \$19;	6384
	6385
\$w1 = 0;	6386
\$w2 = 360;	6387
if (defined \$20) {	6388
\$w1 = \$21;	6389
\$w2 = \$22;	6390
}	6391
	6392
kreis3d(\$layer, \$x, \$y, \$z, \$r1, \$r2, \$r3, \$w1, \$w2, \$farbe, \$fuellart, \$fuellfarbe,	6393
\$linienart, \$linienstaerke, \$parameter);	6394
	6395
}	6396
	6397
elseif (\$zeile =~	6398
/^kreisbogen(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	6399
=([0-9\.\.]+),([a-z0-9!]+))?(,([a-zaeoeue0-9!]+))?(parameter="\([^"]*\)")? p	
=([0-9\.\.]+),([0-9\.\.]+)(r=([0-9\.\.]+),([0-9\.\.]+),([0-9\.\.]+))?(geschlossen)?(pfeil	
?\$/i	
)	6400
{	6401
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	6402
"");	6403
\$layer = \$2 if (defined \$1);	6404
	6405
if (defined \$3) {	6406
\$farbe = \$4;	6407
\$fuellart = \$5;	6408
\$fuellfarbe = \$6;	6409
}	6410
	6411
if (defined \$7) {	6412
\$linienstaerke = \$8;	6413
\$linienart = \$9;	6414
}	6415
\$farbe = \$11 if (defined \$10);	6416
if (defined \$12) { \$parameter = \$13; }	6417
	6418
\$x = \$14;	6419
\$y = \$15;	6420

```

$radius = 1;
$winkel1 = 0;
$winkel2 = 360;

if ( defined $16 ) {
    $radius = $17;
    $winkel1 = $18;
    $winkel2 = $19;
}
$geschlossen = "nein";
$geschlossen = "ja" if ( defined $20 );
$pfeil = "";
$pfeil = "->" if ( defined $21 );
kreisbogen(
    $layer,      $linienart,  $linienstaerke, $x,    $y,
    $radius,     $winkel1,    $winkel2,     $farbe, $fuellart,
    $fuellfarbe, $geschlossen, $parameter,    $pfeil
);
}

elseif ( $zeile =~
/^ellipsebogen( layer=([0-9]+) )?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+) )?( linie
=([0-9\..]+),([a-z0-9!]+) )?( ([a-zaeoeue0-9!]+) )?( parameter="\([^"]*\)" )? p
=([0-9\..]+),([0-9\..]+) )?( r=([0-9\..]+),([0-9\..]+),([0-9\..]+),([0-9\..]+) )?(
geschlossen)?( pfeil)?$/i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe = $4;
        $fuellart = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $x = $14;
    $y = $15;
    $radius1 = 1;
    $radius2 = 1;

    $winkel1 = 0;
    $winkel2 = 360;

    if ( defined $16 ) {
        $radius1 = $17;
        $radius2 = $18;
        $winkel1 = $19;
        $winkel2 = $20;
    }
    $geschlossen = "nein";
    $geschlossen = "ja" if ( defined $21 );

```

\$pfeil	= "";	6479
\$pfeil	= "->" if (defined \$22);	6480
ellipsebogen(		6481
\$layer,	\$linienart, \$linienstaerke, \$x,	6482
\$radius1,	\$radius2, \$winkel1, \$winkel2, \$farbe,	6483
\$fuellart,	\$fuellfarbe, \$geschlossen, \$parameter, \$pfeil	6484
);		6485
}		6486
		6487
elseif (\$zeile =~		6488
/^ring(layer=([0-9]+))?(farbe=([a-zAEOUE0-9!]+),([a-z0-9!]+),([a-zAEOUE0-9!]+))?(linie		6489
=([0-9\.] +),([a-z0-9!]+))?(,[a-zAEOUE0-9!]+))?(parameter="\^[^"]*)"? p		6490
=([0-9\.] +),([0-9\.] +)(r=([0-9\.] +),([0-9\.] +),([0-9\.] +),([0-9\.] +))?\$ /i		
)		6491
{		6492
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",		6493
"");		6494
\$layer = \$2 if (defined \$1);		6495
		6496
if (defined \$3) {		6497
\$farbe	= \$4;	6498
\$fuellart	= \$5;	6499
\$fuellfarbe	= \$6;	6500
}		6501
		6502
if (defined \$7) {		6503
\$linienstaerke	= \$8;	6504
\$linienart	= \$9;	6505
}		6506
\$farbe = \$11 if (defined \$10);		6507
if (defined \$12) { \$parameter = \$13; }		6508
		6509
\$x	= \$14;	6510
\$y	= \$15;	6511
\$radius	= 1;	6512
\$dicke	= 1;	6513
\$winkel1	= 0;	6514
\$winkel2	= 360;	6515
		6516
if (defined \$16) {		6517
\$radius	= \$17;	6518
\$dicke	= \$18;	6519
\$winkel1	= \$19;	6520
\$winkel2	= \$20;	6521
}		6522
		6523
kommentar(\$layer, "Einen Ring zeichnen");		6524
ring(		6525
\$layer,	\$linienart, \$linienstaerke, \$x,	6526
\$radius,	\$dicke, \$winkel1, \$winkel2, \$farbe,	6527
\$fuellart,	\$fuellfarbe, \$parameter	6528
);		6529
		6530
}		6531
elseif (\$zeile =~		6532
/^rechteck(layer=([0-9]+))?(farbe=([a-zAEOUE0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie		6533
=([0-9\.] +),([a-z0-9!]+))?(,[a-zAEOUE0-9!]+))?(parameter="\^[^"]*)"? p		
=([0-9\.] +),([0-9\.] +)(r=([0-9\.] +),([0-9\.] +))?(,[0-9\.] +))?\$ /i		
)		6534
{		6535

```

standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
$layer = $2 if ( defined $1 );

if ( defined $3 ) {
    $farbe      = $4;
    $fuellart    = $5;
    $fuellfarbe = $6;
}

if ( defined $7 ) {
    $linienstaerke = $8;
    $linienart      = $9;
}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$x      = $14;
$y      = $15;
$breite = 1;
$hoehe  = $breite;

if ( defined $16 ) {
    $breite = $17;
    $hoehe  = $breite;
    if ( defined $18 ) { $hoehe = $19; }
}
$winkel = 0;
if ( defined $20 ) { $winkel = $21; }
rechteck(
    $layer,      $x,      $y,      $breite,
    $hoehe,      $winkel, $farbe,  $fuellart ,
    $fuellfarbe, $linienart, $linienstaerke, $parameter
);
}

elsif ( $zeile =~
/^parallelogramm( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(
linie=([0-9\.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\^[^"]*" )? p
=([0-9\.]+),([0-9\.]+)( r=([0-9\.]+),([0-9\.]+))?(,[0-9\.]+))?(,[0-9\.]+))?$
/i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart      = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $x      = $14;

```

\$y = \$15;	6594
\$breite = 1;	6595
\$hoehe = \$breite;	6596
	6597
if (defined \$16) {	6598
\$breite = \$17;	6599
\$hoehe = \$breite;	6600
if (defined \$18) { \$hoehe = \$19; }	6601
}	6602
\$winkel1 = 0;	6603
\$winkel2 = 90;	6604
if (defined \$20) { \$winkel1 = \$21; }	6605
if (defined \$22) { \$winkel2 = \$23; }	6606
parallelogramm(	6607
\$layer, \$x, \$y, \$breite, \$hoehe,	6608
\$winkel1, \$winkel2, \$farbe, \$fuellart, \$fuellfarbe,	6609
\$linienart, \$linienstaerke, \$parameter	6610
);	6611
}	6612
	6613
elseif (\$zeile =~	6614
/^dreieck(layer=([0-9!+])?(farbe=([a-zaeoue0-9!+]),([a-z0-9!+]),([a-z0-9!+]))?(linie	6615
=([0-9!.]+),([a-z0-9!+]))?(,([a-zaeoue0-9!+]))?(parameter="\([^"]*\)")? p	
=([0-9!.]+),([0-9!.]+) (r=([0-9!.]+),([0-9!.]+),([0-9!.]+))?)?\$/i	
)	6616
{	6617
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	6618
"");	6619
\$layer = \$2 if (defined \$1);	6620
	6621
if (defined \$3) {	6622
\$farbe = \$4;	6623
\$fuellart = \$5;	6624
\$fuellfarbe = \$6;	6625
}	6626
	6627
if (defined \$7) {	6628
\$linienstaerke = \$8;	6629
\$linienart = \$9;	6630
}	6631
\$farbe = \$11 if (defined \$10);	6632
if (defined \$12) { \$parameter = \$13; }	6633
	6634
\$x = \$14;	6635
\$y = \$15;	6636
\$laenge = 1;	6637
\$winkel = 1;	6638
	6639
if (defined \$16) {	6640
\$laenge = \$17;	6641
\$winkel = \$18;	6642
}	6643
\$winkel2 = 0;	6644
if (defined \$19) { \$winkel2 = \$20; }	6645
	6646
dreieckrechtwinklig(	6647
\$layer, \$x, \$y, \$laenge,	6648
\$winkel, \$winkel2, \$farbe, \$fuellart,	6649
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter	6650
);	6651
	6652

```

}
elseif ( $zeile =~
/^dreieck( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\([^"]*\)" )? p
=([0-9\..]+),([0-9\..]+) p=([0-9\..]+),([0-9\..]+) r=([0-9\..]+))?)?$/i
)
{
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
$layer = $2 if ( defined $1 );

if ( defined $3 ) {
$farbe = $4;
$fuellart = $5;
$fuellfarbe = $6;
}

if ( defined $7 ) {
$linienstaerke = $8;
$linienart = $9;
}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$x = $14;
$y = $15;
$breite = 1;
$hoehe = 1;

if ( defined $16 ) {
$breite = $17;
$hoehe = $18;
}
$winkel1 = 90;
if ( defined $19 ) { $winkel1 = $20; }

$winkel2 = 0;
if ( defined $21 ) { $winkel2 = $22; }

dreieckbreitehoehe(
$layer, $x, $y, $breite, $hoehe,
$winkel1, $winkel2, $farbe, $fuellart, $fuellfarbe,
$linienart, $linienstaerke, $parameter
);
}

elseif ( $zeile =~
/^dreieck( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\([^"]*\)" )? p
=([0-9\..]+),([0-9\..]+) p=([0-9\..]+),([0-9\..]+) r=([0-9\..]+))$/i
)
{
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
$layer = $2 if ( defined $1 );

if ( defined $3 ) {
$farbe = $4;
$fuellart = $5;

```

\$fuellfarbe = \$6;	6710
}	6711
if (defined \$7) {	6712
\$linienstaerke = \$8;	6713
\$linienart = \$9;	6714
}	6715
\$farbe = \$11 if (defined \$10);	6716
if (defined \$12) { \$parameter = \$13; }	6717
	6718
	6719
\$x1 = \$14;	6720
\$y1 = \$15;	6721
	6722
\$x2 = \$16;	6723
\$y2 = \$17;	6724
	6725
\$winkel = \$18;	6726
	6727
if (gethyp() == 1) {	6728
	6729
dreieckrechtwinkligpunktehoehe(	6730
\$layer, \$x1, \$y1, \$x2,	6731
\$y2, \$winkel, \$farbe, \$fuellart ,	6732
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter	6733
);	6734
	6735
}	6736
else {	6737
dreieckrechtwinkligpunkte(	6738
\$layer, \$x1, \$y1, \$x2,	6739
\$y2, \$winkel, \$farbe, \$fuellart ,	6740
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter	6741
);	6742
	6743
}	6744
	6745
}	6746
	6747
elseif (\$zeile =~	6748
/^pfeil(layer=([0-9]+))?(farbe=([a-zaoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	6749
=([0-9\.]+),([a-z0-9!]+))?(,([a-zaoeue0-9!]+))?(parameter="\([^"]*\)")? p	
=([0-9\.]+),([0-9\.]+)(r=([0-9\.]+),([0-9\.]+),([0-9\.]+),([0-9\.]+))?)?\$/i	
)	6750
{	6751
standard(10, getlinienart () , getlinienstaerke () , getfarbe() , "", getfuellfarbe () , "" ,	6752
"");	6753
\$layer = \$2 if (defined \$1);	6754
	6755
if (defined \$3) {	6756
\$farbe = \$4;	6757
\$fuellart = \$5;	6758
\$fuellfarbe = \$6;	6759
}	6760
	6761
if (defined \$7) {	6762
\$linienstaerke = \$8;	6763
\$linienart = \$9;	6764
}	6765
\$farbe = \$11 if (defined \$10);	6766
if (defined \$12) { \$parameter = \$13; }	6767
	6768

```

$х      = $14;
$у      = $15;
$breite = 1;
$hoehe  = 1;

if ( defined $16 ) {
    $hoehe = $18;
    $breite = $17;
}

$pfeil      = 0.2;
$pfeillaenge = 1;

if ( defined $19 ) {
    $pfeil      = $20;
    $pfeillaenge = $21;
}

pfeil (
    $layer,      $х,      $у,      $breite,      $hoehe,
    $pfeil,      $pfeillaenge,      $farbe, $fuellart, $fuellfarbe,
    $linienart, $linienstaerke, $parameter
);
}

elseif ( $zeile =~
/^pfeilknick( layer=([0-9]+))?( farbe=([a-zaeoue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.] +),([a-z0-9!]+))?( ([a-zaeoue0-9!]+))?( parameter="\([^"]*\)" )? p
=([0-9\.] +),([0-9\.] +)( r=([0-9\.] +),([0-9\.] +),([0-9\.] +),([0-9\.] +)
,([0-9\.] +))?)?$ /i
)
{
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe  = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart      = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $х      = $14;
    $у      = $15;
    $breite  = 1;
    $hoehe   = 1;
    $breittechnik = 0.5;

    if ( defined $16 ) {
        $breite      = $17;
        $breittechnik = $18;
        $hoehe        = $19;
    }

```

\$pfeil	= 0.2;	6827		
\$pfeillaenge	= 1;	6828		
		6829		
if (defined \$20) {		6830		
\$pfeil	= \$21;	6831		
\$pfeillaenge	= \$22;	6832		
}		6833		
		6834		
pfeilknick(		6835		
\$layer,	\$x,	\$y,	\$breite, \$breiteknick,	6836
\$hoehe,	\$pfeil,	\$pfeillaenge,	\$farbe, \$fuellart ,	6837
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter				6838
);				6839
}				6840
				6841
elseif (\$zeile =~				6842
/^dreieck(layer=([0-9!]+))(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie				6843
=([0-9!.]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\([^"]*\)"? p				
=([0-9!.]+),([0-9!.]+),([0-9!.]+))? p=([0-9!.]+),([0-9!.]+),([0-9!.]+))? p				
=([0-9!.]+),([0-9!.]+),([0-9!.]+))?\$ /i				
)				6844
{				6845
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",				6846
"");				6847
\$layer = \$2 if (defined \$1);				6848
				6849
if (defined \$3) {				6850
\$farbe	= \$4;			6851
\$fuellart	= \$5;			6852
\$fuellfarbe	= \$6;			6853
}				6854
				6855
if (defined \$7) {				6856
\$linienstaerke	= \$8;			6857
\$linienart	= \$9;			6858
}				6859
\$farbe = \$11 if (defined \$10);				6860
if (defined \$12) { \$parameter = \$13; }				6861
				6862
\$x1 = \$14;				6863
\$y1 = \$15;				6864
\$z1 = "";				6865
				6866
if (defined \$16) { \$z1 = \$17; }				6867
				6868
\$x2 = \$18;				6869
\$y2 = \$19;				6870
				6871
\$z2 = "";				6872
				6873
if (defined \$20) { \$z2 = \$21; }				6874
				6875
\$x3 = \$22;				6876
\$y3 = \$23;				6877
				6878
\$z3 = "";				6879
				6880
if (defined \$24) { \$z3 = \$25; }				6881
				6882
kommentar(\$layer, "Ein Dreieck zeichnen");				6883
dreieck(\$layer, \$x1, \$y1, \$z1, \$x2, \$y2, \$z2, \$x3, \$y3, \$z3, \$farbe, \$fuellart ,				6884

```

$fuellfarbe, $linienart, $linienstaerke, $parameter );
}

elsif ( $zeile =~
/^viereck( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?( parameter="\([^"]*\)" )? p
=([0-9\..]+),([0-9\..]+),([0-9\..]+))? p=([0-9\..]+),([0-9\..]+),([0-9\..]+))? p
=([0-9\..]+),([0-9\..]+),([0-9\..]+))? p=([0-9\..]+),([0-9\..]+),([0-9\..]+))?$ /i
)
{
standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",
"" );
$layer = $2 if ( defined $1 );

if ( defined $3 ) {
$farbe = $4;
$fuellart = $5;
$fuellfarbe = $6;
}

if ( defined $7 ) {
$linienstaerke = $8;
$linienart = $9;
}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$x1 = $14;
$y1 = $15;
$z1 = "";

if ( defined $16 ) { $z1 = $17; }

$x2 = $18;
$y2 = $19;

$z2 = "";

if ( defined $20 ) { $z2 = $21; }

$x3 = $22;
$y3 = $23;

$z3 = "";

if ( defined $24 ) { $z3 = $25; }

$x4 = $26;
$y4 = $27;

$z4 = "";

if ( defined $28 ) { $z4 = $29; }

viereck( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $x3, $y3, $z3, $x4, $y4, $z4, $farbe,
$fuellart, $fuellfarbe, $linienart, $linienstaerke, $parameter );
}

elsif ( $zeile =~

```

/^neck(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	6943
=([0-9\.] +),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\^[^"]*)\")? p	
=([0-9\.] +),([0-9\.] +) r=([0-9\.] +),([0-9\.] +),([0-9\.] +))?\$ /i	
)	6944
{	6945
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	6946
"");	6947
\$layer = \$2 if (defined \$1);	6948
	6949
if (defined \$3) {	6950
\$farbe = \$4;	6951
\$fuellart = \$5;	6952
\$fuellfarbe = \$6;	6953
}	6954
	6955
if (defined \$7) {	6956
\$linienstaerke = \$8;	6957
\$linienart = \$9;	6958
}	6959
\$farbe = \$11 if (defined \$10);	6960
if (defined \$12) { \$parameter = \$13; }	6961
	6962
\$x1 = \$14;	6963
\$y1 = \$15;	6964
	6965
\$anzahl = \$16;	6966
\$laenge = \$17;	6967
	6968
\$winkel = 0;	6969
	6970
if (defined \$18) { \$winkel = \$19; }	6971
	6972
neck(	6973
\$layer, \$x1, \$y1, \$anzahl,	6974
\$laenge, \$winkel, \$farbe, \$fuellart ,	6975
\$fuellfarbe , \$linienart , \$linienstaerke, \$parameter	6976
);	6977
	6978
}	6979
	6980
elseif (\$zeile =~	6981
/^winkel(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	6982
=([0-9\.] +),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(p=([0-9\.] +),([0-9\.] +) p	
=([0-9\.] +),([0-9\.] +) p=([0-9\.] +),([0-9\.] +)(r=([0-9\.] +))?(,(i a))?(text	
=("\^[^"]*)\")?\$ /i	
)	6983
{	6984
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	6985
"");	6986
\$layer = \$2 if (defined \$1);	6987
	6988
if (defined \$3) {	6989
\$farbe = \$4;	6990
\$fuellart = \$5;	6991
\$fuellfarbe = \$6;	6992
}	6993
	6994
if (defined \$7) {	6995
\$linienstaerke = \$8;	6996
\$linienart = \$9;	6997
}	6998

```

$farbe = $11 if ( defined $10 );
$x      = $12;
$y      = $13;
$x2     = $14;
$y2     = $15;
$x3     = $16;
$y3     = $17;
$l1      = vektorbetrag( vektordifferenz( "x", $x, $y, 0, $x2, $y2, 0 ),
    vektordifferenz( "y", $x, $y, 0, $x2, $y2, 0 ), 0 );
$l2      = vektorbetrag( vektordifferenz( "x", $x3, $y3, 0, $x2, $y2, 0 ),
    vektordifferenz( "y", $x3, $y3, 0, $x2, $y2, 0 ), 0 );
$l3      = vektorbetrag( vektordifferenz( "x", $x, $y, 0, $x3, $y3, 0 ),
    vektordifferenz( "y", $x, $y, 0, $x3, $y3, 0 ), 0 );
$radius  = minimum( $l1, $l2, $l3 ) / 3;
$radius  = $19 if ( defined $18 );
$winkelrichtung = "i";
$winkelrichtung = $21 if ( defined $20 );
$text    = "";
$text    = $23 if ( defined $22 );
winkel( $layer, $x, $y, $x2, $y2, $x3, $y3, $radius, $text, $winkelrichtung, $farbe,
    $fuellart, $fuellfarbe, $linienart, $linienstaerke, "" );
}

elsif ( $zeile =~
/^winkel( layer=([0-9]+) )?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+) )?( linie
=([0-9\.\.]+),([a-z0-9!]+) )?( ([a-zaeoeue0-9!]+) )? p=([0-9\.\.]+),([0-9\.\.]+)
,([0-9\.\.]+) p=([0-9\.\.]+),([0-9\.\.]+),([0-9\.\.]+) p=([0-9\.\.]+),([0-9\.\.]+)
,([0-9\.\.]+) )? r=([0-9\.\.]+) )?( (i|a) )?( text="\^[^"]*" )? )$/i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
        "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe  = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    $x      = $12;
    $y      = $13;
    $z      = $14;
    $x2     = $15;
    $y2     = $16;
    $z2     = $17;

    $x3 = $18;
    $y3 = $19;
    $z3 = $20;

    $l1 = vektorbetrag(
        vektordifferenz(
            "x",
            x3d( $x, $y, $z ),

```

y3d(\$x, \$y, \$z),	7057
0,	7058
x3d(\$x2, \$y2, \$z2),	7059
y3d(\$x2, \$y2, \$z2),	7060
0	7061
),	7062
vektordifferenz(	7063
"y",	7064
x3d(\$x, \$y, \$z),	7065
y3d(\$x, \$y, \$z),	7066
0,	7067
x3d(\$x2, \$y2, \$z2),	7068
y3d(\$x2, \$y2, \$z2),	7069
0	7070
),	7071
0	7072
);	7073
\$l2 = vektorbetrag(	7074
vektordifferenz(	7075
"x",	7076
x3d(\$x3, \$y3, \$z3),	7077
y3d(\$x3, \$y3, \$z3),	7078
0,	7079
x3d(\$x2, \$y2, \$z2),	7080
y3d(\$x2, \$y2, \$z2),	7081
0	7082
),	7083
vektordifferenz(	7084
"y",	7085
x3d(\$x3, \$y3, \$z3),	7086
y3d(\$x3, \$y3, \$z3),	7087
0,	7088
x3d(\$x2, \$y2, \$z2),	7089
y3d(\$x2, \$y2, \$z2),	7090
0	7091
),	7092
0	7093
);	7094
\$l3 = vektorbetrag(	7095
vektordifferenz(	7096
"x",	7097
x3d(\$x, \$y, \$z),	7098
y3d(\$x, \$y, \$z),	7099
0,	7100
x3d(\$x3, \$y3, \$z3),	7101
y3d(\$x3, \$y3, \$z3),	7102
0	7103
),	7104
vektordifferenz(	7105
"y",	7106
x3d(\$x, \$y, \$z),	7107
y3d(\$x, \$y, \$z),	7108
0,	7109
x3d(\$x3, \$y3, \$z3),	7110
y3d(\$x3, \$y3, \$z3),	7111
0	7112
),	7113
0	7114
);	7115
\$radius = minimum(\$l1, \$l2, \$l3) / 3;	7116
\$radius = \$22 if (defined \$21);	7117

```

$winkelrichtung = "i";
$winkelrichtung = $24 if ( defined $23 );
$text           = "";
$text           = $26 if ( defined $25 );
winkel(
    $layer, x3d( $x, $y, $z ),
    y3d( $x, $y, $z ), x3d( $x2, $y2, $z2 ),
    y3d( $x2, $y2, $z2 ), x3d( $x3, $y3, $z3 ),
    y3d( $x3, $y3, $z3 ), $radius,
    $text,      $winkelrichtung,
    $farbe,     $fuellart,
    $fuellfarbe, $linienart,
    $linienstaerke, ""
);
}

elseif ( $zeile =~ /^pstricks( layer=([0-9]+) )?$/i ) {
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
        "" );
    $layer = $2 if ( defined $1 );
    my $enderreicht = 0;
    my $hilfszeile;
    my $inzeile = 0;
    while ( ( !$enderreicht )
        && ( $i < $eingabedateizeilen - 1 ) )
    {
        $i++;
        $hilfszeile = $eingabedatei[$i];
        if ( $hilfszeile =~ /^ende$/i ) { $enderreicht = 1; }
        elseif ( $hilfszeile ne "" ) {
            $inzeile++;
            pstricks( $layer, $hilfszeile );
        }
        else {
            kommentar( $layer, "Kaputte Zeile" );
            $enderreicht = 1;
        }
    }
    if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
    else { }
}

elseif ( $zeile =~
/^bild( layer=([0-9]+) )? p=([0-9\.]+),([0-9\.]+),([0-9\.]+) )? r=([0-9\.]+)
    (,([0-9\.]+) )?(,([0-9\.]+) )? ([^"]*)$/i
)
{
    $layer = 10;
    $layer = $2 if ( defined $1 );
    $x     = $3;
    $y     = $4;
    $z     = "";
    $z     = $6 if ( defined $5 );
    $breite = $7;
    $winkel = 0;
    $winkel = $9 if ( defined $8 );
    $offset = 0;
    $offset = $11 if ( defined $10 );
    $bild = $12;
    bild( $layer, $x, $y, $z, $breite, $winkel, $bild );
}

```

}	7178
elseif (	7179
\$zeile =~	7180
/^ strichliste (layer=([0-9]+))?(linie=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+)? p	7181
=([0-9\..]+),([0-9\..]+) r=([0-9\..]+),([0-9\..]+) ([^\"]*)\$/i	7182
)	7183
{	7184
settexthier ("");	7185
	7186
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	7187
"");	7188
\$layer = \$2 if (defined \$1);	7189
	7190
if (defined \$3) {	7191
\$linienstaerke = \$4;	7192
\$linienart = \$5;	7193
}	7194
	7195
if (defined \$6) { \$farbe = \$7; }	7196
\$x = \$8;	7197
\$y = \$9;	7198
\$hoehe = \$10;	7199
\$abstand = \$11, \$wert = \$12;	7200
strichliste (\$layer, \$x, \$y, \$wert, \$hoehe, \$abstand, \$farbe, \$linienstaerke,	7201
\$linienart);	7202
}	7203
	7204
elseif (\$zeile =~	7205
/^binomial(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	7206
=([0-9\..]+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(parameter="\([^"]*\)")? np	
=([0-9\..]+),([0-9\..]+),([a-z0-9!]+))?(b=([0-9\..]+),([0-9\..]+))\$/i	
)	7207
{	7208
	7209
setpfeilart ("->");	7210
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), ">	7211
"");	7212
\$layer = \$2 if (defined \$1);	7213
	7214
if (defined \$3) {	7215
\$farbe = \$4;	7216
\$fuellart = \$5;	7217
\$fuellfarbe = \$6;	7218
}	7219
	7220
if (defined \$7) {	7221
\$linienstaerke = \$8;	7222
\$linienart = \$9;	7223
}	7224
\$farbe = \$11 if (defined \$10);	7225
if (defined \$12) { \$parameter = \$13; }	7226
	7227
\$nwert = \$14;	7228
\$pwert = \$15;	7229
\$aufsummiert = "nein";	7230
if (defined \$16) { \$aufsummiert = \$17; }	7231
\$breite = 1;	7232
\$markierung = \$nwert;	7233
	7234
if (defined \$18) {	7235

```

    $breite      = $19;
    $markierung = $20;
}

kommentar( $layer, "Die Binomialverteilung" );

binomial(
    $layer,      $nwert,      $pwert,      $aufsummiert,
    $breite,     $farbe,     $fuellart,   $fuellfarbe,
    $linienart,  $linienstaerke, $markierung, $parameter
);

}

elsif ( $zeile =~
/^binomialteil( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.]+),([a-z0-9!]+))?(,([a-zaeoeue0-9!]+))?( parameter="\([^"]*\)" )? np
=([0-9\.]+),([0-9\.]+),([0-9\.]+),([0-9\.]+),([a-z0-9!]+))?( b=([0-9\.]+),([0-9\.]+))$/i
)
{
    setpfeilart (" ->");
    standard( 10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), " ->",
    "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart   = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $nwert      = $14;
    $pwert      = $15;
    $anfang     = $16;
    $sende      = $17;
    $aufsummiert = "nein";

    if ( defined $18 ) { $aufsummiert = $19; }
    $breite      = 1;
    $markierung = $nwert;
    if ( defined $20 ) {
        $breite      = $21;
        $markierung = $22;
    }
    kommentar( $layer, "Die Binomialverteilung teilweise gezeichnet" );
    binomialteil(
        $layer,      $nwert,      $pwert,      $anfang,   $sende,
        $aufsummiert, $breite,     $farbe,     $fuellart,   $fuellfarbe,
        $linienart,  $linienstaerke, $markierung, $parameter
    );
}

elsif ( $zeile =~
/^binomialstrich( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(

```

linie=([0-9\.[+]),([a-z0-9!+]))?([a-zaeoeue0-9!+]))?(parameter=\`"([^\"]*)\`"? np	
=([0-9\.[+]),([0-9\.[+]),([a-z0-9!+]))?(b=([0-9\.[+]),([0-9\.[+]))\$/i	
)	7295
{	7296
setpfeilart ("→");	7297
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "→",	7298
"");	7299
\$layer = \$2 if (defined \$1);	7300
	7301
if (defined \$3) {	7302
\$farbe = \$4;	7303
\$fuellart = \$5;	7304
\$fuellfarbe = \$6;	7305
}	7306
	7307
if (defined \$7) {	7308
\$linienstaerke = \$8;	7309
\$linienart = \$9;	7310
}	7311
\$farbe = \$11 if (defined \$10);	7312
if (defined \$12) { \$parameter = \$13; }	7313
	7314
\$nwert = \$14;	7315
\$pwert = \$15;	7316
\$aufsummiert = "nein";	7317
if (defined \$16) { \$aufsummiert = \$17; }	7318
\$breite = 1;	7319
\$markierung = \$nwert;	7320
	7321
if (defined \$18) {	7322
\$breite = \$19;	7323
\$markierung = \$20;	7324
}	7325
	7326
kommentar(\$layer, "Die Binomialverteilung");	7327
	7328
setbinostrich ();	7329
binomial(	7330
\$layer, \$nwert, \$pwert, \$aufsummiert,	7331
\$breite, \$farbe, \$fuellart, \$fuellfarbe,	7332
\$linienart, \$linienstaerke, \$markierung, \$parameter	7333
);	7334
	7335
}	7336
	7337
elseif (\$zeile =~	7338
/^binomialteilstrich(layer=([0-9!+]))? (farbe=([a-zaeoeue0-9!+]),([a-z0-9!+]),([a-z0-9!+]))	7339
? (linie=([0-9\.[+]),([a-z0-9!+]))?([a-zaeoeue0-9!+]))?(parameter=\`"([^\"]*)\`"? np	
=([0-9\.[+]),([0-9\.[+]),([0-9\.[+]),([0-9\.[+]),([a-z0-9!+]))?(b=([0-9\.[+]),([0-9\.[+]))\$/i	
)	7340
{	7341
setpfeilart ("→");	7342
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "→",	7343
"");	7344
\$layer = \$2 if (defined \$1);	7345
	7346
if (defined \$3) {	7347
\$farbe = \$4;	7348
\$fuellart = \$5;	7349
\$fuellfarbe = \$6;	7350
}	7351

```

if ( defined $7 ) {
    $linienstaerke = $8;
    $linienart     = $9;
}
$farbe = $11 if ( defined $10 );
if ( defined $12 ) { $parameter = $13; }

$nwert      = $14;
$pwert      = $15;
$anfang     = $16;
$sende      = $17;
$aufsummiert = "nein";

if ( defined $18 ) { $aufsummiert = $19; }
$breite     = 1;
$markierung = $nwert;
if ( defined $20 ) {
    $breite     = $21;
    $markierung = $22;
}
kommentar( $layer, "Die Binomialverteilung teilweise gezeichnet" );
setbinostrich ();
binomialteil(
    $layer,      $nwert,      $pwert,      $anfang, $sende,
    $aufsummiert, $breite,    $farbe,      $fuellart, $fuellfarbe,
    $linienart,  $linienstaerke, $markierung, $parameter
);
}

elsif ( $zeile =~
/^baumdiagramm( layer=([0-9]+))?( farbe=([a-zAEOeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(
linie=([0-9\.]+),([a-z0-9!]+))?(,[a-zAEOeue0-9!]+))?( parameter="\([^"]*\)" )?( p
=([-0-9\.]+),([-0-9\.]+))?$ /i
)
{
    standard( 10, getlinienart (), 0.2, getfarbe(), "", getfuellfarbe (), "", "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart     = $9;
    }
    $farbe = $11 if ( defined $10 );
    if ( defined $12 ) { $parameter = $13; }

    $x = 0;
    $y = 0;
    if ( defined $14 ) {
        $x = $15;
        $y = $16;
    }
    $text = "";

```

\$enderreicht = 0;	7411
\$inzeile = 0;	7412
while ((!\$enderreicht)	7413
&& (\$i < \$eingabedateizeilen - 1))	7414
{	7415
\$i++;	7416
\$hilfszeile = \$eingabedatei[\$i];	7417
if (\$hilfszeile =~ /^ende\$/i) {	7418
\$enderreicht = 1;	7419
	7420
}	7421
elsif (\$hilfszeile =~ /^(^\\")* -> ([^\\")*\$/i) {	7422
knotenzuordnung(\$1 . "->" . \$2);	7423
\$text = \$text . \$1 . "->" . \$2 . " ";	7424
	7425
\$inzeile++;	7426
	7427
}	7428
else {	7429
kommentar(\$layer, "Kaputte Zeile");	7430
\$enderreicht = 1;	7431
}	7432
}	7433
if (!\$enderreicht) { printf("%Keine sinnvollen Daten\n"); }	7434
else {	7435
baumdiagramm(	7436
\$layer, \$x, \$y, \$farbe, \$fuellart ,	7437
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter, \$text	7438
);	7439
}	7440
	7441
}	7442
	7443
elsif (\$zeile =~	7444
/^uhr(layer=([0-9]+))?(farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?(linie	7445
=([0-9\\.]+),([a-z0-9!]+))?(,([a-zaeoeue0-9!]+))?(parameter=\\"(^\\")*\\")? p	
=([0-9\\.]+),([0-9\\.]+)(r=([0-9\\.]+),([0-9\\.]+))?\$ /i	
)	7446
{	7447
standard(10, getlinienart () , getlinienstaerke () , getfarbe() , "" , getfuellfarbe () , "" ,	7448
"");	7449
\$layer = \$2 if (defined \$1);	7450
	7451
if (defined \$3) {	7452
\$farbe = \$4;	7453
\$fuellart = \$5;	7454
\$fuellfarbe = \$6;	7455
}	7456
	7457
if (defined \$7) {	7458
\$linienstaerke = \$8;	7459
\$linienart = \$9;	7460
}	7461
\$farbe = \$11 if (defined \$10);	7462
if (defined \$12) { \$parameter = \$13; }	7463
	7464
\$x = \$14;	7465
\$y = \$15;	7466
\$radius = 1;	7467
\$laenge = 0.2;	7468
	7469

```

if ( defined $16 ) {
    $radius = $17;
    $laenge = $18;
}
uhr(
    $layer,    $x,        $y,        $radius,
    $laenge,   $farbe,    $fuellart, $fuellfarbe,
    $linienart, $linienstaerke, $parameter
);
}

elseif ( $zeile =~
/^uhrzeit( layer=([0-9]+))?( linie=([0-9\.]�+),([a-z0-9!]+))?(,[a-zaeoeue0-9!]+))?(
parameter="\([^\"]*\)"? p=([0-9\.]�+),([0-9\.]�+) r=([0-9\.]�+)(,[0-9\.]�+)
)?,([0-9\.]�+))?(,[0-9\.]�+))?$ /i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $linienstaerke = $4;
        $linienart      = $5;
    }
    $farbe = $7 if ( defined $6 );

    if ( defined $8 ) { $parameter = $9; }

    $x = $10;
    $y = $11;

    $stunde = "";
    $minute = "";
    $sekunde = "";

    $radius = $12;
    $stunde = $14 if ( defined $13 );
    $minute = $16 if ( defined $15 );
    $sekunde = $18 if ( defined $17 );

    uhrzeit(
        $layer,    $x,        $y,        $radius,
        $stunde,   $minute,   $sekunde, $farbe,
        $linienart, $linienstaerke, $parameter
    );
}

elseif ( $zeile =~
/^knoten( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+))?( parameter="\([^\"]*\)"? p
=([0-9\.]�+),([0-9\.]�+),([0-9\.]�+))?( r="\([^\"]*\)"\,"([^\"]*\)"\,"([^\"]*\)"\)"?$/i
)
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );

    $farbe = $4 if ( defined $3 );
    if ( defined $6 ) { $parameter = $5; }

```

\$x = \$7;	7528
\$y = \$8;	7529
\$z = "";	7530
\$z = \$10 if (defined \$9);	7531
	7532
	7533
\$text = \$12;	7534
\$name = \$11;	7535
	7536
\$art = "rnode";	7537
\$art = \$14 if (defined \$13);	7538
	7539
knoten(\$layer, \$farbe, \$x, \$y, \$z, \$name, \$text, \$art, \$parameter);	7540
}	7541
	7542
elseif (\$zeile =~	7543
/^linie(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.\.]+),([a-z0-9!]+))	7544
?(\\"([^\"]*)\\")?(parameter=\\"([^\"]*)\\")? p=\\"([^\"]*)\\" p=\\"([^\"]*)\\"\$/i	7545
)	7546
{	7547
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	7548
"");	7549
\$layer = \$2 if (defined \$1);	7550
\$farbe = \$4 if (defined \$3);	7551
	7552
if (defined \$5) {	7553
linienstaerke = \$6;	7554
linienart = \$7;	7555
}	7556
	7557
\$pfeilart = \$9 if (defined \$8);	7558
	7559
if (defined \$10) {	7560
parameter1 = \$11;	7561
}	7562
else {	7563
parameter1 = "";	7564
}	7565
\$name1 = \$12;	7566
\$name2 = \$13;	7567
	7568
knotenverbindung(	7569
layer, name1, name2, "",	7570
linienart, linienstaerke, farbe, parameter1,	7571
"" ,	7572
pfeilart	7573
);	7574
}	7575
	7576
elseif (\$zeile =~	7577
/^linie(layer=([0-9]+))?(farbe=([a-zA-Z0-9!]+))?(linie=([0-9\.\.]+),([a-z0-9!]+))	
?(\\"([^\"]*)\\")?(parameter=\\"([^\"]*)\\")? p=\\"([^\"]*)\\" p=\\"([^\"]*)\\" text=\\"([^\"]*)	
(\\"([^\"]*)\\")?\$/i	
)	7578
{	7579
standard(10, getlinienart (), getlinienstaerke (), getfarbe(), "", getfuellfarbe (), "",	7580
"");	7581
\$layer = \$2 if (defined \$1);	7582
\$farbe = \$4 if (defined \$3);	7583
	7584
if (defined \$5) {	7585

```

$linienstaerke = $6;
$linienart      = $7;
}

$pfeilart = $9 if ( defined $8 );

if ( defined $10 ) {
    $parameter1 = $11;
}
else {
    $parameter1 = "";
}
$name1      = $12;
$name2      = $13;
$text       = $14;
$parameter2 = "";
$parameter2 = $16 if ( defined $15 );

knotenverbindung(
    $layer,      $name1, $name2,  $text,      $linienart,
    $linienstaerke, $farbe, $parameter1, $parameter2, $pfeilart
);
}

elsif ( $zeile =~
/^kreis( layer=([0-9]+))?( farbe=([a-zaeoeue0-9!]+),([a-z0-9!]+),([a-z0-9!]+))?( linie
=([0-9\.]+),([a-z0-9!]+))?(,([a-zaeoeue0-9!]+))?(,\'([^\']*)\')?( parameter=\'([^\']*)\')
? p=\'([^\']*)\' r=([0-9\.]+))?( text=\'([^\']*)\' ,([0-9\.]+))?(,([a-z0-9!]+))?)?$/i
)
{
    standard( 10, getlinienart(), getlinienstaerke(), getfarbe(), "", getfuellfarbe(), "",
    "" );
    $layer = $2 if ( defined $1 );

    if ( defined $3 ) {
        $farbe      = $4;
        $fuellart    = $5;
        $fuellfarbe  = $6;
    }

    if ( defined $7 ) {
        $linienstaerke = $8;
        $linienart      = $9;
    }
    $farbe = $11 if ( defined $10 );

    $pfeilart = $13 if ( defined $12 );

    if ( defined $14 ) { $parameter = $15; }

    $name1      = $16;
    $radius      = 1;
    $radius      = $18 if ( defined $17 );
    $text        = "";
    $winkel       = 0;
    $ausrichtung = 0;
    if ( defined $19 ) {
        $text = $20;

        $winkel = $22 if ( defined $21 );

```

```

7586
7587
7588
7589
7590
7591
7592
7593
7594
7595
7596
7597
7598
7599
7600
7601
7602
7603
7604
7605
7606
7607
7608
7609
7610
7611
7612
7613
7614
7615
7616
7617
7618
7619
7620
7621
7622
7623
7624
7625
7626
7627
7628
7629
7630
7631
7632
7633
7634
7635
7636
7637
7638
7639
7640
7641
7642
7643
7644

```

\$ausrichtung = \$24 if (defined \$23);	7645
}	7646
	7647
knotenkreis(	7648
\$layer, \$name1, \$text, \$winkel,	7649
\$radius, \$farbe, \$fuellart, \$fuellfarbe,	7650
\$linienart, \$linienstaerke, \$pfeilart, \$parameter	7651
);	7652
	7653
}	7654
	7655
elseif (\$zeile =~ /^usepackage=\"([^\"]*)\"\\\"([^\"]*)\"\$/i) {	7656
usepackage(\$1,\$2);	7657
}	7658
elseif (\$zeile =~ /^definition=\"([^\"]*)\"\$/i) {	7659
pstricks(105, \$1);	7660
}	7661
elseif (\$zeile =~ /^inhalt=\"([^\"]*)\"\$/i) { kommentar(102, \$1); }	7662
	7663
elseif (\$zeile =~ /^inhalt\$/i) {	7664
	7665
\$enderreicht = 0;	7666
\$inzeile = 0;	7667
while ((!\$enderreicht)	7668
&& (\$i < \$eingabedateizeilen - 1))	7669
{	7670
\$i++;	7671
\$hilfszeile = \$eingabedatei[\$i];	7672
if (\$hilfszeile =~ /^ende\$/i) { \$enderreicht = 1; }	7673
elseif (\$hilfszeile =~ /^(.*)\$/i) {	7674
kommentar(102, \$1);	7675
\$inzeile++;	7676
}	7677
else {	7678
kommentar(\$layer, "Kaputte Zeile");	7679
\$enderreicht = 1;	7680
}	7681
}	7682
	7683
if (!\$enderreicht) { printf("Keine sinnvollen Daten\\n"); }	7684
else {	7685
abstand(\$layer);	7686
	7687
}	7688
}	7689
	7690
elseif (\$zeile =~ /^schreibe=\"([^\"]*)\"\$/i) {	7691
\$name = \$1;	7692
\$enderreicht = 0;	7693
\$inzeile = 0;	7694
while ((!\$enderreicht)	7695
&& (\$i < \$eingabedateizeilen - 1))	7696
{	7697
\$i++;	7698
\$hilfszeile = \$eingabedatei[\$i];	7699
if (\$hilfszeile =~ /^ende\$/i) { \$enderreicht = 1; }	7700
elseif (\$hilfszeile =~ /^([-0-9\\.]+)[, ; ;]([-0-9\\.]+)\$/i) {	7701
\$x = \$1;	7702
\$y = \$2;	7703
setachsengraph(\$x,\$y, \$x, \$y);	7704
if (\$inzeile == 0) { oeffnedatei(\$name); }	7705
\$inzeile++;	

```

        schreibedaten( $name, $x, $y );
    }
    else {
        kommentar( $layer, "Kaputte Zeile" );
        $enderreicht = 1;
    }
}
if ( !$enderreicht ) { printf("Keine sinnvollen Daten\n"); }
else { schliessedatei( $name); }
}

elseif ( $zeile =~ /^([-0-9\\.]+)[,|;| , | ; ]([-0-9\\.]+)$/i ) {
    setachsengraph( $1, $2, $1, $2 );
    minmax( $1, $2, $1, $2 );
}

# Alle anderen Befehle sind Schrott
else { printf( "%% Unbekannter Befehl '%s'\n", $zeile ); }
}
}
setgitterbild( 1 );
setachsenbild(99);

$leer = 1;
for ( $layer = 0 ; $layer <= 100 ; $layer++ ) {
    for ( $i = 0 ; $i < $ausgabedateizeilen[$layer] ; $i++ ) { $leer = 0; }
}
if ( $leer == 0 ) {
    kopfdatei();
    fussdatei();
    for ( $layer = 101 ; $layer <= 113 ; $layer++ ) {
        for ( $i = 0 ; $i < $ausgabedateizeilen[$layer] ; $i++ ) {
            printf( "%s \n", $ausgabedatei[$layer][$i] );
        }
    }
    for ( $layer = 0 ; $layer <= 100 ; $layer++ ) {
        for ( $i = 0 ; $i < $ausgabedateizeilen[$layer] ; $i++ ) {
            printf( "%s \n", $ausgabedatei[$layer][$i] );
        }
    }
}
else { printf( "%% Die Datei ist leer \n"); }

```

A.2. Das Modul für Koordinatensysteme

Listing A.2: achsen.pm

```

#####
# Modul fuer Koordinaten-Achsen
#
# Letzte Aenderung: 31.08.2011
#####<>
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;
my (
    $xachsetrue, $yachsetrue, $xminachse, $yminachse, $xmaxachse,
    $ymaxachse, $xschritt, $yschritt, $xskala, $yskala,

```

```

    $xtext,    $ytext,    $xmark,    $ymark,    $zmark
);
13
14
15
$achsettrue = $yachsettrue = 0;
16
#####
17
# Achsen zeichnen:
18
# wurden die einzelnen Achsen schon aufgerufen?
19
# Wurden die Achsen gezeichnet?
20
#####
21
sub achsengezeichnet {
22
    my ($xodery) = @_ ;
23
    if ( $xodery eq "x" ) {
24
        if ( getachsenzeichnen("x") == 0 ) { return 0; }
25
        else { return 1; }
26
    }
27
    elsif ( $xodery eq "y" ) {
28
        if ( getachsenzeichnen("y") == 0 ) { return 0; }
29
        else { return 1; }
30
    }
31
    else { return "ERROR"; }
32
}
33
34
my $achsenart = "->";
35
36
sub setachsenart {
37
    my ($achsenarthier) = @_ ;
38
    $achsenart = $achsenarthier;
39
}
40
41
my $subticks = 0;
42
43
sub setsubticks {
44
    my ($ticks) = @_ ;
45
    $subticks = $ticks;
46
}
47
48
sub getsubticks { return $subticks; }
49
50
sub getachsenart { return $achsenart; }
51
52
sub setachsenzeichnen {
53
    my ( $wert, $xodery ) = @_ ;
54
    if ( $xodery eq "x" ) { $achsettrue = $wert; }
55
    if ( $xodery eq "y" ) { $yachsettrue = $wert; }
56
}
57
58
sub getachsenzeichnen {
59
    my ($xodery) = @_ ;
60
    if ( $xodery eq "x" ) { return $achsettrue; }
61
    if ( $xodery eq "y" ) { return $yachsettrue; }
62
}
63
64
$xmark = $ymark = $zmark = "";
65
66
sub setmarkierung {
67
    my ( $wert, $xyz ) = @_ ;
68
    if ( $xyz eq "x" ) { $xmark = $wert; }
69
    if ( $xyz eq "y" ) { $ymark = $wert; }
70
    if ( $xyz eq "z" ) { $zmark = $wert; }
71
}
72
73

```

```

sub getmarkierung {
  my ($xyz) = @_;
  if ( $xyz eq "x" ) { return $xmark; }
  if ( $xyz eq "y" ) { return $ymark; }
  if ( $xyz eq "z" ) { return $zmark; }
}

sub setschritt {
  my ( $schritt, $xodery ) = @_;
  if ( $xodery eq "x" ) { $xschritt = $schritt; }
  if ( $xodery eq "y" ) { $yschritt = $schritt; }
}

sub getschritt {
  my ($xodery) = @_;
  if ( $xodery eq "x" ) { return $xschritt; }
  if ( $xodery eq "y" ) { return $yschritt; }
}

sub setskala {
  my ( $schritt, $xodery ) = @_;
  if ( $xodery eq "x" ) { $xskala = $schritt; }
  if ( $xodery eq "y" ) { $yskala = $schritt; }
}

sub getskala {
  my ($xodery) = @_;
  if ( $xodery eq "x" ) { return $xskala; }
  if ( $xodery eq "y" ) { return $yskala; }
}

sub settext {
  my ( $schritt, $xodery ) = @_;
  if ( $xodery eq "x" ) { $xtext = $schritt; }
  if ( $xodery eq "y" ) { $ytext = $schritt; }
}

sub gettext {
  my ($xodery) = @_;
  if ( $xodery eq "x" ) { return $xtext; }
  if ( $xodery eq "y" ) { return $ytext; }
}

sub setminmaxachsen {
  my ( $min, $xodery, $minmax ) = @_;
  if ( $xodery eq "x" ) {
    if ( $minmax eq "min" ) { $xminachse = $min; }
    else { $xmaxachse = $min; }
  }
  if ( $xodery eq "y" ) {
    if ( $minmax eq "min" ) { $yminachse = $min; }
    else { $ymaxachse = $min; }
  }
}

#####
# Wie ist das Minimum und Maximum?
#####

sub getminmaxachsen {
  my ( $xodery, $minmax ) = @_;
  if ( $xodery eq "x" ) {
    if ( $minmax eq "min" ) { return $xminachse; }

```

74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134

```

    else                                { return $xmaxachse; }
}
if ( $xodery eq "y" ) {
    if ( $minmax eq "min" ) { return $yminachse; }
    else                                { return $ymaxachse; }
}
}

my ( $xmitte, $ymitte );
$xmitte = $ymitte = 0;

sub setmitte { $xmitte = $ymitte = 1; }

sub achsenbeschriftung {
    my ( $layer, $farbe, $xmax, $ymax, $xtext, $ytext ) = @_;

    if ( $xmitte == 0 ) {
        beschriftung( $layer, $farbe, $xmax, getschnitt("y") - 0.6 / getskalierung("y"),
            0, 180, $xtext );
    }
    else {
        beschriftung(
            $layer, $farbe,
            getschnitt("x") + ( $xmax - getschnitt("x") ) / 2,
            getschnitt("y") - 1 / getskalierung("y"),
            0, "c", $xtext
        );
    }
    if ( $ymitte == 0 ) {
        beschriftung(
            $layer, $farbe,
            getschnitt("x") + 0.2 / getskalierung("x"),
            $ymax - 0.2 / getskalierung("y"),
            0, 0, $ytext
        );
    }
    else {
        beschriftung(
            $layer, $farbe,
            getschnitt("x") - 1.2 / getskalierung("x"),
            getschnitt("y") + ( $ymax - getschnitt("y") ) / 2,
            90, "c", $ytext
        );
    }
}

my $pfeilartachsen = "->";

sub setpfeilart {
    my ($art) = @_;
    $pfeilartachsen = $art;
}

sub getpfeilart { return $pfeilartachsen; }

sub xachse {
    my (
        $layer,    $farbe,    $linienstaerke, $linienart, $xminachse,
        $xmaxachse, $xschritt, $xskala,      $xtext
    ) = @_;

```

```

if ( achsengezeichnet("y") == 0 ) {
    schnittsetzen( $xminachse, "x" );
}
minmax( $xminachse - $xschritt, -1, $xmaxachse + $xschritt, 1 );
achsen(
    $layer,    $farbe,    $linienstaerke, $linienart, getpfeilart(), $xminachse,
    $xmaxachse, 0,        0,        $xschritt, $xskala,    $xtext,
    0,         "keine", ""
);
}

#####
# die y-Achse wird gezeichnet
#####
sub yachse {
    my (
        $layer,    $farbe,    $linienstaerke, $linienart, $yminachse,
        $ymaxachse, $yschritt, $yskala,    $ytext
    ) = @_;

    minmax( -1, $yminachse - $yschritt, 1, $ymaxachse + $yschritt );

    achsen(
        $layer,    $farbe,    $linienstaerke, $linienart, getpfeilart(), 0,
        0,         $yminachse, $ymaxachse, 0,        $yskala,    "",
        $yschritt, $yskala,    $ytext
    );
}

my ( $ystep, $xstep );

$ystep = $xstep = 1;

sub achsen {
    my (
        $layer,    $farbe,    $linienstaerke, $linienart, $pfeilart,
        $xminachse, $xmaxachse, $yminachse,    $ymaxachse, $xschritt,
        $xskala,    $xtext,    $yschritt,    $yskala,    $ytext
    ) = @_;

    minmax(
        $xminachse - $xschritt,
        $yminachse - $yschritt,
        $xmaxachse + $xschritt,
        $ymaxachse + $yschritt
    );

    $ystep = $yschritt;
    $xstep = $xschritt;

    if ( getmatheheft() == 1 ) {
        setmatheheft(2);
        matheheft( $layer, "lightgray", $xminachse, $yminachse, $xmaxachse, $ymaxachse );
    }

    if ( getgitter() == 1 ) {
        setgitter( 2 );
        kariert(
            1,    "lightgray", 0.25,    "solid", $xminachse, $yminachse,

```

196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256

\$xmaxachse,\$ymaxachse,\$xschritt, 0,	257
0, 0	258
);	259
}	260
	261
my (\$parameterweiter, \$achsen);	262
	263
if (\$xminachse != \$xmaxachse) {	264
if (\$xschritt == 0) {	265
\$xschritt = aufrunden(\$xmaxachse * 10000, 1) / 10000;	266
\$xschritt = schoenetabelle(\$xmaxachse * 10000) / 10000;	267
}	268
}	269
	270
if (\$yminachse != \$ymaxachse) {	271
if (\$yschritt == 0) {	272
\$yschritt = aufrunden(\$ymaxachse * 10000, 1) / 10000;	273
\$yschritt = schoenetabelle(\$ymaxachse * 10000) / 10000;	274
}	275
}	276
	277
if (\$xminachse == \$xmaxachse) {	278
\$achsen = "xAxis=false, showorigin=true,";	279
}	280
elseif (\$yminachse == \$ymaxachse) {	281
\$achsen = "yAxis=false, showorigin=true,";	282
}	283
else { \$achsen = "showorigin=false,"; }	284
setgrossskalierung(\$xmaxachse - \$xminachse, \$ymaxachse - \$yminachse);	285
if (\$linienart eq "") { \$linienart = "solid"; }	286
if (\$xskala eq "keine") {	287
	288
\$parameterweiter =	289
\$achsen	290
. "comma, ticks=none, labels=none, subticks= "	291
. getsubticks() . ", Dx="	292
. \$xschritt . ", Dy="	293
. \$yschritt	294
. ", dx=0pt, dy=0pt, Ox="	295
. getschnitt("x") . ", Oy="	296
. getschnitt("y");	297
	298
}	299
elseif (\$xskala eq "nurstriche") {	300
\$parameterweiter =	301
\$achsen	302
. "comma, labels=none, subticks="	303
. getsubticks() . ", Dx="	304
. \$xschritt . ", Dy="	305
. \$yschritt	306
. ", dx=0pt, dy=0pt, Ox="	307
. getschnitt("x") . ", Oy="	308
. getschnitt("y");	309
	310
}	311
elseif (\$xskala eq "nureins") {	312
xmarkierung(\$layer, 1, 1, \$farbe);	313
ymarkierung(\$layer, 1, 1, \$farbe);	314
\$parameterweiter =	315
\$achsen	316
. "comma, labels=none, subticks=0, Dx="	317

```

        . $xschritt . " , Dy="
        . $yschritt
        . " , dx=0pt, dy=0pt, Ox="
        . getschnitt("x") . " , Oy="
        . getschnitt("y");
    }
    else {
        $parameterweiter =
            $achsen
            . "comma, subticks="
            . getsubticks() . " , Dx="
            . $xschritt . " , Dy="
            . $yschritt
            . " , dx=0pt, dy=0pt, Ox="
            . getschnitt("x") . " , Oy="
            . getschnitt("y");
    }

    if ( $xmitte == 1 ) {
        achsenbeschriftung( $layer, $farbe, $xmaxachse, $ymaxachse, $xtext, $ytext );
    }

    paar( getschnitt("x"), getschnitt("y") );
    paar( $xminachse, $yminachse );

    paar( $xmaxachse, $ymaxachse );

    if ( $xmitte == 0 ) {
        textanhaengen( "[" . $xtext . " , -90]" . $ytext . " , 0" );
    }

    if ( $linienstaerke > 0 ) {
        befehl erzeugen( $layer, "psaxes", $farbe, $linienart, $linienstaerke, $parameterweiter,
            $pfeilart, gettexthier() );
    }

    settexthier( "" );

    setachsengraph( $xminachse, $yminachse, $xmaxachse, $ymaxachse );

    if ( $xskala eq "keine" ) {
        minmax( $xminachse, $yminachse, $xmaxachse, $ymaxachse );
    }
    else {
        minmax(
            $xminachse - $xschritt * 2,
            $yminachse - $yschritt * 2,
            $xmaxachse + $xschritt * 2,
            $ymaxachse + $yschritt * 2
        );
    }
}

sub schoenetabelle {
    my ($obergrenze) = @_ ;

    my (
        $schoen1, $schoen2, $schoen3, $kriterium1, $kriterium2,
        $skalierung, $breite, $kandidat, $schritt, $skalenstrich,
        $basis, $skalierung2, $i, $schoenerschritt
    )

```

318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378

```

);
$ schoen1      = 1;
$ schoen2      = 2;
$ schoen3      = 5;
$ kriterium1    = 5;
$ kriterium2    = 12.5;
$ schoenerschritt = 0;

$ skalierung = 10**floor( logarithmus( $obergrenze, 10 ) );

$ breite = $obergrenze / $skalierung;

if ( $breite >= $schoen3 ) { $kandidat = $schoen3; }
else {
    if ( $breite >= $schoen2 ) { $kandidat = $schoen2; }
    else { $kandidat = $schoen1; }
}
$skalierung2 = $skalierung;
$basis = $kandidat;
for ( $i = 1 ; $i <= 15 ; $i++ ) {

    $schritt = $skalierung2 * $basis;
    $skalenstrich = $obergrenze / $schritt;

    if ( ( $skalenstrich > $kriterium1 )
        && ( $skalenstrich < $kriterium2 ) )
    {
        $schoenerschritt = $schritt;
    }

    if ( $basis == 1 ) { $skalierung2 = $skalierung2 / 10; }
    else { $skalierung2 = $skalierung2; }
    if ( $basis == $schoen3 ) { $basis = $schoen2; }
    else {
        if ( $basis == $schoen2 ) { $basis = $schoen1; }
        else { $basis = $schoen3; }
    }
}

return $schoenerschritt;
}

#####
# Achsen mit Bogenmass
#####
sub xachserad {
    my ( $layer, $farbe, $linienstaerke, $linienart, $xminachse, $xmaxachse, $xschritt, $xtext )
        =
        @_ ;
    minmax( $xminachse - $xschritt, -1, $xmaxachse + $xschritt, 1 );

    xyachserad( $layer, $farbe, $linienstaerke, $linienart, $xminachse, $xmaxachse, 0, 0,
        $xschritt,
        $xtext, 0, 0, "", 1, "rad" );
}

sub xyachserad {
    my (

```

379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427

428
429
430
431

432
433
434
435
436
437

```

$layer, $farbe, $linienstaerke, $linienart, $xminachse,
$xmaxachse, $yminachse, $ymaxachse, $xschritt, $xtext,
$yschritt, $yskala, $ytext, $xbruch, $rad
) = @_;
my ( $label, $i, $mark, $zahl, $pi, $grad );

if ( $rad eq "deg" ) {
    $label = degtorad($xschritt);
    $xminachse = degtorad($xminachse);
    $xmaxachse = degtorad($xmaxachse);
    $grad = "deg";
}
elseif ( $rad eq "rad" ) {
    $label = $xschritt;
    $xminachse = $xminachse;
    $xmaxachse = $xmaxachse;
    $grad = "rad";
}
elseif ( $rad eq "degtorad" ) {
    $label = degtorad($xschritt);
    $xminachse = degtorad($xminachse);
    $xmaxachse = degtorad($xmaxachse);
    $grad = "rad";
}
else {
    $label = $xschritt;
    $xminachse = $xminachse;
    $xmaxachse = $xmaxachse;
    $grad = "deg";
}

$pi = degtorad(180);

achsen(
    $layer, $farbe, $linienstaerke, $linienart,
    "->", $xminachse, $xmaxachse, $yminachse,
    $ymaxachse, $xmaxachse * 2, $yskala, $xtext,
    $yschritt, $yskala, $ytext
);

for ( $i = 0 ; $i < $xmaxachse ; $i = $i + $label ) {
    if ( $i != 0 ) {

        if ( $xbruch == 1 ) {
            if ( $grad eq "deg" ) {
                $zahl = radtodeg($i);
                $mark = "\$". $zahl . "\^\{\circ\}\$";
                $mark =~ tr/./,/s;
            }
            else {
                $zahl = $i / $pi;
                $mark = "\$". $zahl . "\pi\$";
                $mark =~ tr/./,/s;
            }
        }
        else {

            if ( $grad eq "deg" ) {
                $zahl = radtodeg( $i * $xbruch );
                $mark = "\$". $zahl . "\^\{\circ\}\$";
            }
        }
    }
}

```

```

        $mark =~ tr/./,/s;
    }
    else {
        $zahl = $i / $pi * $xbruch;
        $mark = "\$\\frac{" . $zahl . " }{" . $xbruch . "}\\pi\$";
        $mark =~ tr/./,/s;
    }

}
xmarkierung( $layer, $i, $mark, $farbe );
}
}

for ( $i = 0 ; $i > $xminachse ; $i = $i - $label ) {
    if ( $i != 0 ) {

        if ( $xbruch == 1 ) {
            if ( $grad eq "deg" ) {
                $zahl = abs( radtodeg($i) );
                $mark = "\$-". $zahl . "^{\circ}\$";
                $mark =~ tr/./,/s;
            }
            else {
                $zahl = abs( $i / $pi );
                $mark = "\$-". $zahl . "\\pi\$";
                $mark =~ tr/./,/s;
            }
        }
        else {

            if ( $grad eq "deg" ) {
                $zahl = abs( radtodeg( $i * $xbruch ) );
                $mark = "\$-". $zahl . "^{\circ}\$";
                $mark =~ tr/./,/s;
            }
            else {
                $zahl = abs( $i / $pi * $xbruch );
                $mark = "\$-\\frac{" . $zahl . " }{" . $xbruch . "}\\pi\$";
                $mark =~ tr/./,/s;
            }
        }

        xmarkierung( $layer, $i, $mark, $farbe );
    }
}

#####
### Markierungen an den Achsen
#####

my $xmarkierungkopf = 0;

sub xmarkierung {
    my ( $layer, $x, $text, $farbe ) = @_;
    minmax( $x, -$ystep / getskalierung("y"), $x, $ystep / getskalierung("y") );

    $text =~ tr/./,/s;
    if ( $xmarkierungkopf == 0 ) {
        pstricks(
            105, "\\makeatletter\\newcommand{\\psxpoint}[3][black]{%"

```

```

\\psline[linewidth=.5pt,linestyle=#1]{-}(#2,\\pst@xticksizB)(#2,\\pst@xticksizA)
\\rput[t]{! #2 \\pst@number\\pslabelsep \\pst@number\\pst@xticksizB add
\\pst@number\\psunit div neg )}\\color{#1}#3}
\\makeatother \\n"
);
$markierungskopf = 1;
}
if ( $farbe eq "black" ) {
  schreibeinarray( $layer, sprintf( "\\psxpoint{%g}{%s}", $x, $text ) );
}
else {
  schreibeinarray( $layer, sprintf( "\\psxpoint{%g}{%g}{%s}", $farbe, $x, $text ) );
}
}

my $markierungskopfohne = 0;

sub xmarkierungohne {
  my ( $layer, $x, $text, $farbe ) = @_;
  minmax( $x, -$ystep / getskalierung("y"), $x, $ystep / getskalierung("y") );

  $text =~ tr/./,/s;
  if ( $markierungskopfohne == 0 ) {
    pstricks(
      105, "\\makeatletter\\newcommand{\\psxpointohne}[3][black]{%
\\rput[t]{! #2 \\pst@number\\pslabelsep \\pst@number\\pst@xticksizB add
\\pst@number\\psunit div neg )}\\color{#1}#3}
\\makeatother \\n"
    );
    $markierungskopfohne = 1;
  }
  if ( $farbe eq "black" ) {
    schreibeinarray( $layer, sprintf( "\\psxpointohne{%g}{%s}", $x, $text ) );
  }
  else {
    schreibeinarray( $layer, sprintf( "\\psxpointohne{%s}{%g}{%s}", $farbe, $x, $text ) );
  }
}

sub xmarkierungstrahl {
  my ( $layer, $x, $ylaenge, $text, $linienart, $linienstaerke, $farbe, $parameter ) = @_;

  $text =~ tr/./,/s;

  linie (
    $layer, $x, $ylaenge / 2,
    "", $x, -$ylaenge / 2,
    "", $linienart, $linienstaerke, $farbe, $parameter, ""
  );
  beschriftung( $layer, $farbe, $x, -( $ylaenge / 2 + $ylaenge / 4 ), 0, 270, $text );
}

sub ymarkierungstrahl {
  my ( $layer, $y, $xlaenge, $text, $linienart, $linienstaerke, $farbe, $parameter ) = @_;

  $text =~ tr/./,/s;

  linie (
    $layer, $xlaenge / 2,

```

```

    $y, "", -$xlaenge / 2,
    $y, "", $linienart, $linienstaerke, $farbe, $parameter, ""
);
beschriftung( $layer, $farbe, -( $xlaenge / 2 + $xlaenge / 4 ), $y, 0, 180, $text );
}

my $xmarkierungkopfboben = 0;

sub xmarkierungobten {
    my ( $layer, $x, $text, $farbe ) = @_;
    minmax( $x, -1, $x, 1 );

    if ( $xmarkierungkopfboben == 0 ) {
        pstricks(
            105, "\\makeatletter\\newcommand{\\psxpointoben}[3][black]{%
\\psline[linewidth=.5pt,linecolor=#1]{-}(#2,\\pst\\@xticksizB)(#2,\\pst\\@xticksizA)
\\rput[b]{! #2 \\pst\\@number\\pslabelsep \\pst\\@number\\pst\\@xticksizB add
\\pst\\@number\\psyunit div }{\\color{#1}#3}}
\\makeatother \\n"
        );
        $xmarkierungkopfboben = 1;
    }
    if ( $farbe eq "black" ) {
        schreibeinarray( $layer, sprintf( "\\psxpointoben[%g]{%s}", $x, $text ) );
    }
    else {
        schreibeinarray( $layer, sprintf( "\\psxpointoben[%s]{%g}{%s}", $farbe, $x, $text ) );
    }
}

my $xmulti = 0;

sub xmarkierungmulti {
    my ( $layer, $x1, $x2, $schritt, $farbe ) = @_;
    minmax( $x1, 0, $x2, 0 );

    my $dx = $x2 - $x1;
    if ( $xmulti == 0 ) {
        pstricks(
            105, "\\makeatletter\\newcommand{\\psxpointstrich}[3][black]{%
\\psline[linewidth=.5pt,linecolor=#1]{-}(#2,\\pst\\@xticksizB)(#2,\\pst\\@xticksizA)
\\makeatother"
        );
        $xmulti = 1;
    }
    schreibeinarray(
        $layer,
        sprintf(
            "\\multido{\\n=%g+%g}{%i}{\\psxpointstrich[%s]{\\n}{\\n+1}}",
            $x1, $schritt, $dx, $farbe,
        )
    );
}

my $ymarkierungkopf = 0;

sub ymarkierung {
    my ( $layer, $ywert, $text, $farbe ) = @_;
    minmax( -$xstep / getskalierung("x"), $ywert, $xstep / getskalierung("x"), $ywert );

```

```

if ( $markierungkopf == 0 ) {
    pstricks(
        105, "\\makeatletter\\newcommand{\\psypoint}[3][black]{%
\\psline[linewidth=.5pt,linewidth=#1]{-}{\\pst\\@yticksB,#2)(\\pst\\@yticksA,#2)
\\rput[r](! \\pst\\@number\\pslabelsep \\pst\\@number\\pst\\@yticksA sub
\\pst\\@number\\psxunit div neg #2){\\color{#1}#3}}
\\makeatother
"
    );
    $markierungkopf = 1;
}
schreibeinarray( $layer, sprintf( "\\psypoint[%s][%g]{%s}", $farbe, $ywert, $text ) );
}

my $markierungkopfohne = 0;

sub ymarkierungohne {
    my ( $layer, $ywert, $text, $farbe ) = @_;
    minmax( -$xstep / getskalierung("x"), $ywert, $xstep / getskalierung("x"), $ywert );

    if ( $markierungkopfohne == 0 ) {
        pstricks(
            105, "\\makeatletter\\newcommand{\\psypointohne}[3][black]{%
\\rput[r](! \\pst\\@number\\pslabelsep \\pst\\@number\\pst\\@yticksA sub
\\pst\\@number\\psxunit div neg #2){\\color{#1}#3}}
\\makeatother
"
        );
        $markierungkopfohne = 1;
    }
    schreibeinarray( $layer, sprintf( "\\psypointohne[%s][%g]{%s}", $farbe, $ywert, $text ) );
}

my $markierungkopfneu = 0;

sub ymarkierunglinks {
    my ( $layer, $ywert, $text, $farbe ) = @_;
    minmax( 0, $ywert, 0, $ywert );

    if ( $markierungkopfneu == 0 ) {
        pstricks(
            105, "\\makeatletter\\newcommand{\\psypointlinks}[3][black]{%
\\psline[linewidth=.5pt,linewidth=#1]{-}{\\pst\\@yticksB,#2)(\\pst\\@yticksA,#2)
\\rput[l](! \\pst\\@number\\pslabelsep \\pst\\@number\\pst\\@yticksA sub
\\pst\\@number\\psxunit div #2){\\color{#1}#3}}
\\makeatother
"
        );
        $markierungkopfneu = 1;
    }
    schreibeinarray( $layer, sprintf( "\\psypointlinks[%s][%g]{%s}", $farbe, $ywert, $text ) );
}

sub markierung {
    my (
        $layer, $func, $x, $y, $xtext, $ytext,
        $farbe, $linienart, $linienstaerke, $striche, $parameter
    ) = @_;

```

```

minmax( $x, $y, $x, $y );
my $parameterweiter = "";
if ( dreidimensional( $func, "y" ) == 0 ) {
    if ( $striche =~ /x/ ) {
        linie( $layer, $x, $y, "", $x, 0, "", $linienart, $linienstaerke, $farbe, $parameter,
            "" );
        if ( $y < 0 ) { xmarkierungoben( $layer, $x, $xtext, $farbe ); }
        else { xmarkierung( $layer, $x, $xtext, $farbe ); }
    }
    if ( $striche =~ /y/ ) {
        linie( $layer, $x, $y, "", 0, $y, "", $linienart, $linienstaerke, $farbe, $parameter,
            "" );
        if ( $x < 0 ) { ymarkierunglinks( $layer, $y, $ytext, $farbe ); }
        else { ymarkierung( $layer, $y, $ytext, $farbe ); }
    }
}
if ( $parameter eq "" ) {
    $parameterweiter = "dotstyle=x, dotsize=1pt";
}
else { $parameterweiter = $parameter; }

paar( $x, $y );
befehlerzeugen( $layer, "psdot", $farbe, $linienart, $linienstaerke, $parameterweiter, "",
    gettexthier() );
}
else {
    tripel( $x, $y, calcfunk3d( $func, "x", $x, "y", $y ) );
    if ( get3dpstricks() == 1 ) {

        $parameterweiter = "drawCoor, dotstyle=x, dotsize=1pt,";
        $parameterweiter = $parameterweiter . " " . " " . $parameter;
        befehlerezeugen( $layer, "pstThreeDDot", $farbe, $linienart, $linienstaerke,
            $parameterweiter, "", gettexthier() );
    }
    else {
        $parameterweiter = "dotstyle=x, dotsize=1pt,";
        $parameterweiter = $parameterweiter . " " . " " . $parameter;
        befehlerezeugen( $layer, "psdot", $farbe, $linienart, $linienstaerke, $parameterweiter,
            "", gettexthier() );
    }
    xmarkierung3d( $layer, $x, $xtext, $farbe );
    ymarkierung3d( $layer, $y, $ytext, $farbe )
}
settexthier( "" );
}
#####
# Pfeile an den Achsen
#####
sub xpfeil {
    my ( $layer, $x, $ywert, $text, $farbe, $linienstaerke, $linienart ) = @_ ;
    my $ywerttext;

    if ( get3dzeichnen() == 1 ) {
        linie( $layer, $x, $ywert, 0, $x, getschnitt("y"), 0, $linienart, $linienstaerke, $farbe,
            "", "->" );
        if ( $ywert > 0 ) {
            $ywerttext = $ywert + 0.1;
            beschriftung(

```

```

    $layer, $farbe,
    x3d( $x, $ywerttext, 0 ),
    y3d( $x, $ywerttext, 0 ),
    0, 0, $text
);
}
else {
    $ywerttext = $ywert - 0.1;
    beschriftung(
        $layer, $farbe,
        x3d( $x, $ywerttext, 0 ),
        y3d( $x, $ywerttext, 0 ),
        0, 180, $text
    );
}
}
else {
    linie ( $layer, $x, $ywert, "", $x, getschnitt("y"), "", $linienart, $linienstaerke, $farbe,
    "", "->" );
    if ( $ywert > 0 ) {
        $ywerttext = $ywert + 0.1;
        beschriftung( $layer, $farbe, $x, $ywerttext, 0, 90, $text );
    }
    else {
        $ywerttext = $ywert - 0.3;
        beschriftung( $layer, $farbe, $x, $ywerttext, 0, 90, $text );
    }
}
}
}

sub xpfeilkasten {
    my ( $layer, $x, $y, $text, $farbe, $linienstaerke, $linienart ) = @_ ;

    my ( $breite, $hoehe, $laenge );

    $laenge = length($text);

    if ( $laenge == 0 ) { $breite = 0.15 * 3; }
    else { $breite = 0.15 * $laenge; }

    $hoehe = 0.7;
    xpfeil( $layer, $x, $y, "", $farbe, $linienstaerke, $linienart );

    if ( $y > 0 ) {
        rechtecktext(
            $layer,      $x - $breite, $y,      $x + $breite,
            $y + $hoehe, $text,      $farbe,    "",
            "",          $linienart,  $linienstaerke, ""
        );
    }
    else {
        rechtecktext(
            $layer, $x - $breite, $y - $hoehe, $x + $breite,
            $y,     $text,      $farbe,    "",
            "",     $linienart,  $linienstaerke, ""
        );
    }
}

sub ypfeil {

```

my (\$layer, \$x, \$ywert, \$text, \$farbe, \$linienstaerke, \$linienart) = @_;	865
my \$xtext;	866
if (get3dzeichnen() == 1) {	867
linie (\$layer, \$x, \$ywert, 0, getschnitt("x"), \$ywert, 0, \$linienart, \$linienstaerke,	868
\$farbe, "", ">");	869
if (\$x > 0) {	870
\$xtext = \$x + 0.1;	871
beschriftung(	872
\$layer, \$farbe,	873
x3d(\$xtext, \$ywert, 0),	874
y3d(\$xtext, \$ywert, 0),	875
0, 270, \$text	876
);	877
}	878
else {	879
\$xtext = \$x - 0.1;	880
beschriftung(	881
\$layer, \$farbe,	882
x3d(\$xtext, \$ywert, 0),	883
y3d(\$xtext, \$ywert, 0),	884
0, 90, \$text	885
);	886
}	887
}	888
else {	889
linie (\$layer, \$x, \$ywert, "", getschnitt("x"), \$ywert, "", \$linienart, \$linienstaerke,	890
\$farbe, "", ">");	891
if (\$x > 0) {	892
\$xtext = \$x + 0.1;	893
beschriftung(\$layer, \$farbe, \$xtext, \$ywert, 0, 0, \$text);	894
}	895
else {	896
\$xtext = \$x - 0.1;	897
beschriftung(\$layer, \$farbe, \$xtext, \$ywert, 0, 180, \$text);	898
}	899
}	900
	901
	902
	903
	904
sub ypfeilkasten {	905
my (\$layer, \$x, \$y, \$text, \$farbe, \$linienstaerke, \$linienart) = @_;	906
	907
ypfeil(\$layer, \$x, \$y, "", \$farbe, \$linienstaerke, \$linienart);	908
	909
my (\$breite, \$hoehe, \$laenge);	910
	911
\$laenge = length (\$text);	912
	913
if (\$laenge == 0) { \$breite = 0.2 * 3; }	914
else { \$breite = 0.2 * \$laenge; }	915
	916
\$hoehe = 0.7;	917
	918
if (\$x > 0) {	919
rechtecktext(	920
\$layer, \$x, \$y - 0.5 * \$hoehe, \$x + \$breite,	921
\$y + 0.5 * \$hoehe, \$text, \$farbe, "",	922
"", \$linienart, \$linienstaerke, ""	923
);	924
}	925

```

else {
    rechtecktext(
        $layer,          $x - $breite, $y - 0.5 * $hoehe, $x,
        $y + 0.5 * $hoehe, $text,      $farbe,          "",
        "",              $linienart,   $linienstaerke,   "",
    );
}
}

sub zpfeil {
    my ( $layer, $zwert, $ywert, $text, $farbe, $linienstaerke, $linienart ) = @_ ;
    my $ywerttext;

    linie ( $layer, 0, $ywert, $zwert, 0, getschnitt("y"), $zwert, $linienart, $linienstaerke,
        $farbe, "", "->" );
    if ( $ywert > 0 ) {
        $ywerttext = $ywert + 0.1;
        beschriftung(
            $layer, $farbe,
            x3d( 0, $ywerttext, $zwert ),
            y3d( 0, $ywerttext, $zwert ),
            0, 0, $text
        );
    }
    else {
        $ywerttext = $ywert - 0.1;
        beschriftung(
            $layer, $farbe,
            x3d( 0, $ywerttext, $zwert ),
            y3d( 0, $ywerttext, $zwert ),
            0, 180, $text
        );
    }
    abstand($layer);
}

1;

```

A.3. Modul zur Erzeugung von Dateien

Listing A.3: datelerzeugen.pm

```

#####
# Modul um Dateien in Latex zu erzeugen
# Letzte Änderung: 31.08.2011
#####
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;

# Variablen
my (
    $xmin,          $ymin,          $xmax,
    $ymax,          $scalex,        $scaley,
    $scalez,        $skalierungsvorgenommenx, $skalierungsvorgenommeny,
    $skalierungsvorgenommenz, $schnittx,      $schnitty,
    $schnittvorgenommenx, $schnittvorgenommeny, $farbe,

```

```

$linienart ,           $linienstaerke ,           $fuellfarbe ,
$fuellart ,            $tree ,                    $groessesetzen,
$groessex,             $groessey,                 $groessez,
$kopf,                 $schriftgroesse ,   $erstex1,
$erstex2,              $erstey1,                 $erstey2,
$schnittz ,            $schnittvorgenommenz
);
$tree = 0;

$groessesetzen = 0;
$groessex = $groessey = $groessez = 0;

# Min+Max des Bildes
$xmin = $ymin = $xmax = $ymax = 0;
$erstex1 = $erstex2 = $erstey1 = $erstey2 = 0;

#package dateierzeugen ;
# Minimum und Maximum setzen
sub setminmax {
    my ( $min, $xodery, $minmax ) = @_;

    if ( $xodery eq "x" ) {
        if ( $minmax eq "min" ) { $xmin = $min; }
        else { $xmax = $min; }
    }
    if ( $xodery eq "y" ) {
        if ( $minmax eq "min" ) { $ymin = $min; }
        else { $ymax = $min; }
    }
}

# Wie ist das Minimum und Maximum?
sub getminmax {
    my ( $xodery, $minmax ) = @_;
    if ( $xodery eq "x" ) {
        if ( $minmax eq "min" ) { return nachkommastellen( $xmin, 4 ); }
        else { return nachkommastellen( $xmax, 4 ); }
    }
    if ( $xodery eq "y" ) {
        if ( $minmax eq "min" ) { return nachkommastellen( $ymin, 4 ); }
        else { return nachkommastellen( $ymax, 4 ); }
    }
}

my $setmaxberechnung = 1;

sub setmaxberechnung {
    my ($wert) = @_;
    $setmaxberechnung = $wert;
}

sub getmaxberechnung { return $setmaxberechnung; }

# Minimum und Maximum des Bildes setzen
sub minmax {
    my ( $xminlokal, $yminlokal, $xmaxlokal, $ymaxlokal ) = @_;

    if ( ( istzahl($xminlokal) == 1 )
        && ( istzahl($yminlokal) == 1 )

```

```

&& ( istzahl($xmaxlokal) == 1 )
&& ( istzahl($ymaxlokal) == 1 )
{
    if ( getmaxberechnung == 1 ) {
        if ( $erstex1 == 0 ) {
            setminmax( $xminlokal, "x", "min" );
            $erstex1 = 1;
        }
        else {
            setminmax( min( $xminlokal, getminmax( "x", "min" ) ),
                "x", "min" );
        }
        if ( $erstex2 == 0 ) {
            setminmax( $xmaxlokal, "x", "max" );
            $erstex2 = 1;
        }
        else {
            setminmax( max( $xmaxlokal, getminmax( "x", "max" ) ),
                "x", "max" );
        }
        if ( $erstey1 == 0 ) {
            setminmax( $yminlokal, "y", "min" );
            $erstey1 = 1;
        }
        else {
            setminmax( min( $yminlokal, getminmax( "y", "min" ) ),
                "y", "min" );
        }
        if ( $erstey2 == 0 ) {
            setminmax( $ymaxlokal, "y", "max" );
            $erstey2 = 1;
        }
        else {
            setminmax( max( $ymaxlokal, getminmax( "y", "max" ) ),
                "y", "max" );
        }
    }
}
}
### Allgemeine Variablen
# Skalierung des Bildes
$scalex = $scaley = $scalez = 1;
$skalierungvorgenommenx = $skalierungvorgenommeny = $skalierungvorgenommenz = 0;

sub setskalierung {
    my ( $skale, $xyz ) = @_;

    if ( $skale != 0 ) {
        if ( $xyz eq "x" ) { $scalex = $skale; }
        if ( $xyz eq "y" ) { $scaley = $skale; }
        if ( $xyz eq "z" ) { $scalez = $skale; }
    }
}

sub getskalierung {
    my ( $xyz ) = @_;
    if ( $xyz eq "x" ) { return $scalex; }
    if ( $xyz eq "y" ) { return $scaley; }
    if ( $xyz eq "z" ) { return $scalez; }
}

```

```

}
# Ueberpruefen, ob die Skalierung schon vorgenommen wurde, wenn ja, bleibt es bei dem alten
  und wenn nein, wird die Skalieng neu gesetzt
sub skalierungsetzen {
  my ( $skalierung, $xoderyoderz ) = @_;
  if ( $xoderyoderz eq "x" ) {
    if ( $skalierungvorgenommenx == 0 ) {
      setskalierung( $skalierung, "x" );
      $skalierungvorgenommenx = 1;
    }
  }
  if ( $xoderyoderz eq "y" ) {
    if ( $skalierungvorgenommeny == 0 ) {
      setskalierung( $skalierung, "y" );
      $skalierungvorgenommeny = 1;
    }
  }
  if ( $xoderyoderz eq "z" ) {
    if ( $skalierungvorgenommenz == 0 ) {
      setskalierung( $skalierung, "z" );
      $skalierungvorgenommenz = 1;
    }
  }
}

sub setgross {
  my ($gross) = @_;
  $groessesetzen = $gross;
}

sub getgross { return $groessesetzen; }

sub setgroesse {
  my ( $skale, $xyz ) = @_;
  if ( $xyz eq "x" ) { $groessex = $skale; }
  if ( $xyz eq "y" ) { $groessey = $skale; }
  if ( $xyz eq "z" ) { $groessez = $skale; }
}

sub setausmasse {
  my ( $x, $y ) = @_;
  setgross(1);
  setgroesse( $x, "x" );
  setgroesse( $y, "y" );
}

sub setgrossskalierung {
  my ( $breitex, $breitey ) = @_;
  if ( $breitex != 0 ) { setskalierung( getgroesse("x") / $breitex, "x" ); }
  if ( $breitey != 0 ) { setskalierung( getgroesse("y") / $breitey, "y" ); }
}

sub setgrossskalierung3d {
  my ( $breitex, $breitey, $breitez ) = @_;
  if ( getgroesse("x") != 0 ) {
    $scalex = nachkommastellen( getgroesse("x") / $breitex, 6 );
    $scaley = nachkommastellen( getgroesse("y") / $breitey, 6 );
    $scalez = nachkommastellen( getgroesse("z") / $breitez, 6 );
  }
}

```

```

}
199
200
sub getgroesse {
201
202   my ($xyz) = @_ ;
203   if ( $xyz eq "x" ) { return $groessex; }
204   if ( $xyz eq "y" ) { return $groessey; }
205   if ( $xyz eq "z" ) { return $groessez; }
206
207 }
208
# Schnittpunkte des Bildes
209
$schnittx = $schnitty = $schnittvorgenommenx = $schnittvorgenommeny =
210
$schnittvorgenommenz = 0;
211
212
sub setschnitt {
213
214   my ( $schnitt, $xyz ) = @_ ;
215   if ( $xyz eq "x" ) { $schnittx = $schnitt; }
216   if ( $xyz eq "y" ) { $schnitty = $schnitt; }
217   if ( $xyz eq "z" ) { $schnittz = $schnitt; }
218
219 }
220
sub getschnitt {
221
222   my ($xyz) = @_ ;
223   if ( $xyz eq "x" ) { return $schnittx; }
224   if ( $xyz eq "y" ) { return $schnitty; }
225   if ( $xyz eq "z" ) { return $schnittz; }
226
227 }
228
sub schnittsetzen {
229
230   my ( $schnitt, $xyz ) = @_ ;
231   if ( $xyz eq "x" ) {
232       if ( $schnittvorgenommenx == 0 ) {
233           setschnitt( $schnitt, "x" );
234           $schnittvorgenommenx = 1;
235       }
236   }
237   if ( $xyz eq "y" ) {
238       if ( $schnittvorgenommeny == 0 ) {
239           setschnitt( $schnitt, "y" );
240           $schnittvorgenommeny = 1;
241       }
242   }
243   if ( $xyz eq "z" ) {
244       if ( $schnittvorgenommenz == 0 ) {
245           setschnitt( $schnitt, "z" );
246           $schnittvorgenommenz = 1;
247       }
248   }
249
250 }
251
$farbe = "black";
252
253
sub setfarbe {
254
255   my ($farbwahl) = @_ ;
256   $farbe = $farbwahl;
257
258 }
259
sub getfarbe { return $farbe; }
$fuellfarbe = "black!20";
sub setfuellfarbe {

```

my (\$farbwahl) = @_;	260
\$fuellfarbe = \$farbwahl;	261
}	262
	263
sub getfuellfarbe { return \$fuellfarbe; }	264
\$linienart = "solid";	265
	266
sub setlinienart {	267
my (\$linie) = @_;	268
\$linienart = \$linie;	269
}	270
	271
sub getlinienart { return \$linienart; }	272
\$linienstaerke = 1;	273
	274
sub setlinienstaerke {	275
my (\$linie) = @_;	276
\$linienstaerke = \$linie;	277
}	278
	279
sub getlinienstaerke { return \$linienstaerke; }	280
\$fuellart = "solid";	281
	282
sub setfuellart {	283
my (\$farbwahl) = @_;	284
\$fuellart = \$farbwahl;	285
}	286
	287
sub getfuellart { return \$fuellart; }	288
	289
<i>### Soll der Kopf der Datei mit ausgegeben werden? Standard ist ja.</i>	290
	291
\$kopf = 1;	292
	293
sub setkopf {	294
my (\$kopfhier) = @_;	295
\$kopf = \$kopfhier;	296
}	297
	298
sub getkopf { return \$kopf; }	299
	300
sub einstellungen {	301
my (\$layer, \$farbe, \$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke,	302
\$parameter)	303
= @_;	304
	305
my (	306
\$farbeersatz, \$fuellarersatz, \$fuellfarbeersatz,	307
\$linienarersatz, \$linienstaerkeersatz, \$parameterersatz	308
);	309
if ((\$farbe eq "") (\$farbe eq getfarbe())) { \$farbeersatz = ""; }	310
else { \$farbeersatz = "linecolor=" . \$farbe . ", "; }	311
if ((\$fuellart eq "") (\$fuellart eq getfuellart())) {	312
\$fuellarersatz = "";	313
}	314
else { \$fuellarersatz = " fillstyle=" . \$fuellart . ", "; }	315
if ((\$fuellfarbe eq "") (\$fuellfarbe eq getfuellfarbe())) {	316
\$fuellfarbeersatz = "";	317
}	318
else { \$fuellfarbeersatz = " fillcolor=" . \$fuellfarbe . ", "; }	319
if ((\$linienart eq "") (\$linienart eq getlinienart())) {	320

```

    $linienartersatz = "";
}
else { $linienartersatz = "linestyle=" . $linienart . ", "; }
if ( ( $linienstaerke eq "" ) || ( $linienstaerke eq getlinienstaerke() ) )
{
    $linienstaerkeersatz = "";
}
else { $linienstaerkeersatz = "linewidth=" . $linienstaerke . "pt, "; }
if ( $parameter eq "" ) { $parameterersatz = ""; }
else { $parameterersatz = $parameter; }
schreibeinarray(
    $layer,
    sprintf(
        "\\psset{%s%s%s%s%s%s}",
        $farbeersatz, $fuellartersatz, $fuellfarbeersatz,
        $linienartersatz, $linienstaerkeersatz, $parameterersatz
    )
);
}

sub einstellungenkopf {
    my ( $layer, $farbe, $fuellart, $fuellfarbe, $linienart, $linienstaerke,
        $parameter )
        = @_ ;

    my (
        $farbeersatz, $fuellartersatz, $fuellfarbeersatz,
        $linienartersatz, $linienstaerkeersatz, $parameterersatz
    );

    if ( ( $farbe eq "" ) ) { $farbeersatz = ""; }
    else { $farbeersatz = "linecolor=" . $farbe . ", "; }
    if ( ( $fuellart eq "" ) ) { $fuellartersatz = ""; }
    else { $fuellartersatz = " fillstyle=" . $fuellart . ", "; }
    if ( ( $fuellfarbe eq "" ) ) { $fuellfarbeersatz = ""; }
    else { $fuellfarbeersatz = " fillcolor=" . $fuellfarbe . ", "; }
    if ( ( $linienart eq "" ) ) { $linienartersatz = ""; }
    else { $linienartersatz = "linestyle=" . $linienart . ", "; }
    if ( ( $linienstaerke eq "" ) ) { $linienstaerkeersatz = ""; }
    else { $linienstaerkeersatz = "linewidth=" . $linienstaerke . "pt, "; }
    if ( $parameter eq "" ) { $parameterersatz = ""; }
    else { $parameterersatz = $parameter; }
    schreibeinarray(
        $layer,
        sprintf(
            "\\psset{%s%s%s%s%s%s}",
            $farbeersatz, $fuellartersatz, $fuellfarbeersatz,
            $linienartersatz, $linienstaerkeersatz, $parameterersatz
        )
    );
    abstand($layer);

    setfarbe($farbe);
    setfuellart ( $fuellart );
    setfuellfarbe ( $fuellfarbe );
    setlinienstaerke ( $linienstaerke );
    setlinienart ( $linienart );
}

# Befehl erzeugen
sub befehl erzeugen {

```

```

321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381

```

```

my ( $layer, $befehl, $farbe, $linienart, $linienstaerke, $parameter,
    $pfeilart, $anhang )
    = @_;
befehlfuellungerzeugen(
    $layer,    $befehl,    $farbe,    "",
    "",        $linienart, $linienstaerke, $parameter,
    $pfeilart, $anhang
);
}

sub befehlfuellungerzeugen {
    my (
        $layer,    $befehl,    $farbe,    $fuellart,
        $fuellfarbe, $linienart, $linienstaerke, $parameter,
        $pfeilart, $anhang
    ) = @_;

    my (
        $farbeersatz,    $fuellartersatz,    $fuellfarbeersatz,
        $linienartersatz, $linienstaerkeersatz, $parameterersatz
    );

    if ( ( $farbe eq "" ) || ( $farbe eq getfarbe() ) ) { $farbeersatz = ""; }
    else { $farbeersatz = "linecolor=" . $farbe . ", "; }
    if ( ( $fuellart eq "" ) || ( $fuellart eq getfuellart() ) ) {
        $fuellartersatz = "";
    }
    else { $fuellartersatz = " fillstyle=" . $fuellart . ", "; }
    if ( ( $fuellfarbe eq "" ) || ( $fuellfarbe eq getfuellfarbe() ) ) {
        $fuellfarbeersatz = "";
    }
    else { $fuellfarbeersatz = " fillcolor=" . $fuellfarbe . ", "; }
    if ( ( $linienart eq "" ) || ( $linienart eq getlinienart() ) ) {
        $linienartersatz = "";
    }
    else { $linienartersatz = "linestyle=" . $linienart . ", "; }
    if ( ( $linienstaerke eq "" ) || ( $linienstaerke eq getlinienstaerke() ) ) {
        $linienstaerkeersatz = "";
    }
    else { $linienstaerkeersatz = "linewidth=" . $linienstaerke . "pt, "; }
    if ( $parameter eq "" ) { $parameterersatz = ""; }
    else { $parameterersatz = $parameter; }

    if ( $pfeilart eq "" ) {
        if ( ( $farbeersatz eq "" )
            && ( $fuellartersatz eq "" )
            && ( $fuellfarbeersatz eq "" )
            && ( $linienartersatz eq "" )
            && ( $linienstaerkeersatz eq "" )
            && ( $parameterersatz eq "" ) )
        {
            schreibeinarray( $layer, sprintf( "\\%s%s%s%", $befehl, $anhang ) );
        }
        else {
            schreibeinarray(
                $layer,
                sprintf(
                    "\\%s[%s%s%s%s%s%s%s%s]",
                    $befehl,
                    $farbeersatz,
                    $fuellartersatz, $fuellfarbeersatz,

```

```

        $linienartersatz , $linienstaerkeersatz ,
        $parameterersatz, $anhang
    )
);
}
}
else {
    if ( ( $farbeersatz eq "" )
        && ( $fuellartersatz      eq "" )
        && ( $fuellfarbeersatz    eq "" )
        && ( $linienartersatz     eq "" )
        && ( $linienstaerkeersatz eq "" )
        && ( $parameterersatz     eq "" ) )
    {
        schreibeinarray( $layer,
            sprintf( "\\%s{%s}%s%%", $befehl, $pfeilart, $anhang ) );
    }
    else {
        schreibeinarray(
            $layer,
            sprintf(
                "\\%s[%s%s%s%s%s%s]{%s}%s%%",
                $befehl,
                $farbeersatz,
                $fuellartersatz , $fuellfarbeersatz ,
                $linienartersatz , $linienstaerkeersatz ,
                $parameterersatz, $pfeilart ,
                $anhang
            )
        );
    }
}
}
}
}
$chriftgroesse = 10;

sub setschriftgroesse {
    my ($groesse) = @_;
    $schriftgroesse = $groesse;
}

sub getschriftgroesse { return $schriftgroesse; }

my $beschriftunggroesse = "normalsize";

sub setbeschriftunggroesse {
    my ($groesse) = @_;
    $beschriftunggroesse = $groesse;
}

sub getbeschriftunggroesse { return $beschriftunggroesse; }

sub beschriftunggroesse {
    my ( $text, $groesse ) = @_;
    my $ausgabe;

    if ( $groesse eq "normalsize" ) { $ausgabe = $text; }
    else { $ausgabe = "\\\" . $groesse . \"{\" . $text . \"}\""; }
    return $ausgabe;
}

sub textfarbe {
    my ( $text, $farbe ) = @_;

```

```

my $ausgabe;
if ( $farbe eq getfarbe() ) { $ausgabe = $text; }
else { $ausgabe = "\\color{" . $farbe . "}" . $text . " "; }
return $ausgabe;
}

sub abstand {
my ($layer) = @_ ;
schreibeinarray( $layer, sprintf( "" ) );
}

sub kommentar {
my ( $layer, $text ) = @_ ;
schreibeinarray( $layer, sprintf( "%% %% %s", $text ) );
}

sub komm {
my ($text) = @_ ;
kommentar( 10, $text );
}

sub hilfe {
my ( $layer, $text, $var ) = @_ ;
schreibeinarray( $layer, sprintf( "%% %%s: %s", $text, $var ) );
}

sub usepackage {
my ( $parameter, $befehl ) = @_ ;

if ( getkopf() == 1 ) {
if ( $parameter eq "" ) {
schreibeinarray( 104, sprintf( "\\usepackage{%s}", $befehl ) );
}
else {
schreibeinarray( 104,
sprintf( "\\usepackage[%s]{%s}", $parameter, $befehl ) );
}
if ( $befehl eq "pst-tree" ) {
if ( $tree == 0 ) {
pstricks(
104,
"\\newcommand{\\knoten}[2][white]
{\\Tr{\\psframebox[fillcolor=#1!50,fillstyle=solid ]{#2}}}"
);
$tree = 1;
}
}
}
}

#### Pictureumgebung

sub beginpicture {
my ($layer) = @_ ;

schreibeinarray(
$layer,
sprintf(
"\\begin{pspicture}(%g,%g)(%g,%g)\n",
getminmax( "x", "min" ),

```

504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564

```

    getminmax( "y", "min" ),
    getminmax( "x", "max" ),
    getminmax( "y", "max" )
  )
);
}

sub setachsenbild {
  my ($layer) = @_ ;

  my ( $xeinteilung, $yeinteilung, $zeinteilung );

  if ( getachsen() == 1 ) {
    if ( get3dzeichnen() == 1 ) {
      if ( ( getachsengraph3d("xmax") == getachsengraph3d("xmin") )
        || ( getachsengraph3d("xmax") == 0 ) )
      {
        $xeinteilung = 0;
      }
      else {
        if ( getachsengraph3d("xmax") - getachsengraph3d("xmin") <= 1 )
        {
          $xeinteilung = schoenetabelle( getachsengraph3d("xmax") );
        }
        else {
          if ( schoenetabelle( getachsengraph3d("xmax") ) <= 1 ) {
            $xeinteilung = 1;
          }
          else {
            $xeinteilung =
              schoenetabelle( getachsengraph3d("xmax") );
          }
        }
      }
    }
    if ( ( getachsengraph3d("ymax") == getachsengraph3d("ymin") )
      || ( getachsengraph3d("ymax") == 0 ) )
    {
      $yeinteilung = 0;
    }
    else {
      if ( getachsengraph3d("ymax") - getachsengraph3d("ymin") <= 1 )
      {
        $yeinteilung = schoenetabelle( getachsengraph3d("ymax") );
      }
      else {
        if ( schoenetabelle( getachsengraph3d("ymax") ) <= 1 ) {
          $yeinteilung = 1;
        }
        else {
          $yeinteilung =
            schoenetabelle( getachsengraph3d("ymax") );
        }
      }
    }
  }

  if ( ( getachsengraph3d("zmax") == getachsengraph3d("zmin") )
    || ( getachsengraph3d("zmax") == 0 ) )
  {
    $zeinteilung = 0;
  }
}

```

```

}
else {
    if ( getachsengraph3d("zmax") - getachsengraph3d("zmin") <= 1 )
    {
        $zeinteilung = schoenetabelle( getachsengraph3d("zmax") );
    }
    else {
        if ( schoenetabelle( getachsengraph3d("zmax") ) <= 1 ) {
            $zeinteilung = 1;
        }
        else {
            $zeinteilung =
                schoenetabelle( getachsengraph3d("zmax") );
        }
    }
}

achsen3d(
    $layer,          getfarbe(),
    getlinienstaerke(), getlinienart(),
    "->",            getachsengraph3d("xmin"),
    getachsengraph3d("xmax"), getachsengraph3d("ymin"),
    getachsengraph3d("ymax"), getachsengraph3d("zmin"),
    getachsengraph3d("zmax"), $zeinteilung,
    "normal",         getmarkierung("x"),
    $yeinteilung,      "normal",
    getmarkierung("y"), $zeinteilung,
    "normal",         getmarkierung("z"),
);

}
else {
    if ( getschnitt("x") < getachsengraph("xmin") ) {
        schnittsetzen( getachsengraph("xmin"), "x" );
    }
    if ( getschnitt("y") < getachsengraph("ymin") ) {
        schnittsetzen( getachsengraph("ymin"), "y" );
    }
    if ( getbalken() == 1 ) {

        if ( getmarkierung("x") eq "$x$" ) {
            setmarkierung( "", "x" );
        }
        if ( getmarkierung("y") eq "$y$" ) {
            setmarkierung( "", "y" );
        }
    }

    if ( getrelativ() == 1 ) { setskalierung( 0.1, "x" ); }
    achsen(
        $layer,
        getfarbe(),
        getlinienstaerke(),
        getlinienart(),
        "->",
        getachsengraph("xmin"),
        getachsengraph("xmax") +
            schoenetabelle( getachsengraph("xmax") ),
        getachsengraph("ymin"),
        getachsengraph("ymax") +
            schoenetabelle( getachsengraph("ymax") ),
        schoenetabelle(

```

```

        getachsengraph("xmax") +
        schoenetabelle( getachsengraph("xmax") )
    ),
    "normal",
    getmarkierung("x"),
    getachsengraph("ymax") * 2,
    "normal",
    getmarkierung("y")
);
}

elsif ( getpikto() == 1 ) {
    if ( getmarkierung("x") eq "\\$x\\$" ) {
        setmarkierung( "", "x" );
    }
    if ( getmarkierung("y") eq "\\$y\\$" ) {
        setmarkierung( "", "y" );
    }
}

achsen(
    $layer,
    getfarbe(),
    getlinienstaerke(),
    getlinienart(),
    "->",
    getachsengraph("xmin"),
    getachsengraph("xmax") +
        schoenetabelle( getachsengraph("xmax") ),
    getachsengraph("ymin"),
    getachsengraph("ymax") +
        schoenetabelle( getachsengraph("ymax") ),
    schoenetabelle(
        getachsengraph("xmax") +
        schoenetabelle( getachsengraph("xmax") )
    ),
    "keine",
    getmarkierung("x"),
    getachsengraph("ymax") * 2,
    "keine",
    getmarkierung("y")
);
}

elsif ( getsaeule() == 1 ) {
    if ( getmarkierung("x") eq "\\$x\\$" ) {
        setmarkierung( "", "x" );
    }
    if ( getmarkierung("y") eq "\\$y\\$" ) {
        setmarkierung( "", "y" );
    }
}
if ( getrelativ() == 1 ) { setskalierung( 0.1, "y" ); }
achsen(
    $layer,
    getfarbe(),
    getlinienstaerke(),
    getlinienart(),
    "->",
    getachsengraph("xmin"),
    getachsengraph("xmax") +
        schoenetabelle( getachsengraph("xmax") ),

```

687
 688
 689
 690
 691
 692
 693
 694
 695
 696
 697
 698
 699
 700
 701
 702
 703
 704
 705
 706
 707
 708
 709
 710
 711
 712
 713
 714
 715
 716
 717
 718
 719
 720
 721
 722
 723
 724
 725
 726
 727
 728
 729
 730
 731
 732
 733
 734
 735
 736
 737
 738
 739
 740
 741
 742
 743
 744
 745
 746
 747

```
        getachsengraph("ymin"),
        getachsengraph("ymax") +
        schoenetabelle( getachsengraph("ymax") ),
        getachsengraph("xmax") * 2,
        "normal",
        getmarkierung("x"),
        schoenetabelle(
            getachsengraph("ymax") +
            schoenetabelle( getachsengraph("ymax") )
        ),
        "normal",
        getmarkierung("y")
    );
}
else {
    if (
        schoenetabelle(
            getachsengraph("xmax") - getachsengraph("xmin")
        ) <= 1
    )
    {
        $xeinteilung = 1;
    }
    else {
        $xeinteilung =
            schoenetabelle(
                getachsengraph("xmax") - getachsengraph("xmin") );
    }
    if (
        schoenetabelle(
            getachsengraph("ymax") - getachsengraph("ymin")
        ) <= 1
    )
    {
        $yeinteilung = 1;
    }
    else {
        $yeinteilung =
            schoenetabelle(
                getachsengraph("ymax") - getachsengraph("ymin") );
    }
    achsen(
        $layer,
        getlinienstaerke(),
        getfarbe(),
        getlinienart(),
        "→",
        getachsengraph("xmin"),
        getachsengraph("xmax"),
        getachsengraph("ymin"),
        getachsengraph("ymax"),
        $xeinteilung,
        "normal",
        getmarkierung("x"),
        $yeinteilung,
        "normal",
        getmarkierung("y")
    );
}
}
}
setachsen(2);
}

sub setgitterbild {
```

```
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
```

```

my ($layer) = @_ ;
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869

if ( getachsen() != 1 ) {
    settexthier("");
    my ($xeinteilung, $yeinteilung, $zeinteilung );
    if ( getschnitt("x") < getachsengraph("xmin") ) {
        schnittsetzen( getachsengraph("xmin"), "x" );
    }
    if ( getschnitt("y") < getachsengraph("ymin") ) {
        schnittsetzen( getachsengraph("ymin"), "y" );
    }

    if ( getgitter () == 1 ) {
        if ( get3dzeichnen() == 1 ) {
            if ( getachsengraph3d("xmax") == getachsengraph3d("xmin") ) {
                $xeinteilung = 0;
            }
            else {
                $xeinteilung =
                    schoenetabelle(
                        getachsengraph3d("xmax") – getachsengraph3d("xmin") );
            }
            if ( getachsengraph3d("ymax") == getachsengraph3d("ymin") ) {
                $yeinteilung = 0;
            }
            else {
                $yeinteilung =
                    schoenetabelle(
                        getachsengraph3d("ymax") – getachsengraph3d("ymin") );
            }
            if ( getachsengraph3d("zmax") == getachsengraph3d("zmin") ) {
                $zeinteilung = 0;
            }
            else {
                $zeinteilung =
                    schoenetabelle(
                        getachsengraph3d("zmax") – getachsengraph3d("zmin") );
            }
            kariert3d(
                $layer,                "lightgray",
                0.25,                  "solid",
                getachsengraph3d("xmin"), getachsengraph3d("ymin"),
                getachsengraph3d("zmin"), getachsengraph3d("xmax"),
                getachsengraph3d("ymax"), getachsengraph3d("zmax"),
                $xeinteilung,           0,
                0,                      $yeinteilung,
                0,                      0,
                $zeinteilung,           0,
                0
            );
        }
        else {
            $xeinteilung =
                schoenetabelle(
                    getachsengraph("xmax") – getachsengraph("xmin") );
            $yeinteilung =
                schoenetabelle(
                    getachsengraph("ymax") – getachsengraph("ymin") );
            kariert(

```

\$layer,	"lightgray",	870
0.25,	"solid",	871
getachsengraph("xmin"), getachsengraph("ymin"),		872
getachsengraph("xmax"), getachsengraph("ymax"),		873
\$xeinteilung,	0,	874
0,	\$yeinteilung,	875
0,	0	876
);		877
}		878
}		879
		880
if (getmatheheft() == 1) {		881
setmatheheft(2);		882
matheheft(\$layer, "lightgray", getachsengraph("xmin"),		883
getachsengraph("ymin"), getachsengraph("xmax"),		884
getachsengraph("ymax"));		885
}		886
		887
settexthier ("");		888
setgitter (2);		889
}		890
}		891
		892
sub endpicture {		893
my (\$layer) = @_;		894
pstricks(\$layer, "\\end{pspicture}");		895
}		896
		897
<i>### Pstrickstext</i>		898
sub pstricks {		899
my (\$layer, \$pstrickstext) = @_;		900
schreibeinarray(\$layer, sprintf ("%s", \$pstrickstext));		901
}		902
		903
sub latex {		904
my (\$layer, \$latextext) = @_;		905
pstricks(\$layer, \$latextext);		906
}		907
		908
sub schleife {		909
my (\$layer, \$variable, \$start, \$ende, \$schritt) = @_;		910
		911
my \$weitehier = abrunden(((\$ende - \$start) / \$schritt) + 1), 1);		912
my \$text =		913
"\\multido{\\\"		914
. \$variable . "="		915
. \$start . "+"		916
. \$schritt . "}{"		917
. \$weitehier . "}{%%";		918
pstricks(\$layer, \$text);		919
		920
}		921
		922
sub schleifeanzahl {		923
my (\$layer, \$variable, \$start, \$anzahl, \$schritt) = @_;		924
my \$text =		925
"\\multido{\\\"		926
. \$variable . "="		927
. \$start . "+"		928
. \$schritt . "}{"		929
. \$anzahl . "}{%%";		930

pstricks(\$layer, \$text);	931
}	932
	933
sub schleifenende {	934
my (\$layer) = @_;	935
pstricks(\$layer, "}");	936
}	937
	938
1;	939

dateierzeugen

A.4. Das Modul für Diagramme

Im Folgenden findet sich die Datei für die Erstellung von Diagrammen und Verteilungen.

Listing A.4: diagramme.pm

#####	1
# Modul fuer Diagramme	2
# Letzte Aenderung: 24.01.2011	3
#####	4
use POSIX qw /floor ceil/;	5
use Math::Trig;	6
use strict ;	7
use warnings;	8
	9
my \$achsen = 0;	10
	11
## Achsen werden gezeichnet	12
sub setachsen {	13
my (\$wert) = @_;	14
\$achsen = \$wert;	15
}	16
	17
sub getachsen { return \$achsen; }	18
	19
my (\$balken, \$saeule, \$relativ, \$pikto);	20
	21
\$balken = \$saeule = \$relativ = \$pikto = 0;	22
	23
sub setbalken { \$balken = 1; }	24
sub getbalken { return \$balken; }	25
	26
sub setsaeule { \$saeule = 1; }	27
sub getsaeule { return \$saeule; }	28
	29
sub setrelativ { \$relativ = 1; }	30
sub getrelativ { return \$relativ; }	31
sub setpikto { \$pikto = 1; }	32
sub getpikto { return \$pikto; }	33
	34
sub setymaxstat {	35
my (\$max) = @_;	36
setachsengraph(getachsengraph("xmin"), getachsengraph("ymin"),	37
getachsengraph("xmax"), \$max);	38
}	39
	40
sub saeule {	41
my (	42

```

$layer,      $x,      $y,      $dx,      43
$dy,      $farbe,      $fuellart , $fuellfarbe , 44
$linienart , $linienstaerke, $text,      $textwinkel, 45
$textausrichtung, $welcherbalken, $parameter 46
) = @_; 47
48
setachsengraph( $x, $y, $x + 2 * $dx, $y + $dy ); 49
50
#  achsengraph( $x,      $y ); 51
#  achsengraph( $x + $dx, $y + $dy ); 52
if ( $dx == 0 ) { 53
    linie ( $layer, $x, $y, "", $x, $y + $dy, "", $linienart , $linienstaerke , 54
        $farbe, $parameter, "-" ); 55
} 56
elseif ( $dy == 0 ) { 57
    linie ( $layer, $x, $y, "", $x + $dx, $y, "", $linienart , $linienstaerke , 58
        $farbe, $parameter, "-" ); 59
} 60
else { 61
    rechteck( 62
        $layer, nachkommastellen( $x, 5 ), 63
        nachkommastellen( $y, 5 ), nachkommastellen( $dx, 5 ), 64
        nachkommastellen( $dy, 5 ), 0, 65
        $farbe,      $fuellart , 66
        $fuellfarbe , $linienart , 67
        $linienstaerke, $parameter 68
    ); 69
} 70
71
if ( $welcherbalken eq "s" ) { 72
    if ( $textausrichtung eq "auf" ) { 73
        beschriftung( $layer + 1, "black", $x + $dx / 2, 74
            $y, $textwinkel, 90, $text ); 75
    } 76
    elseif ( $textausrichtung eq "mitte" ) { 77
        beschriftung( 78
            $layer + 1, "black", 79
            $x + $dx / 2, 80
            $y + $dy / 2, 81
            $textwinkel, "c", $text 82
        ); 83
    } 84
    elseif ( $textausrichtung eq "unten" ) { 85
        beschriftung( $layer + 1, "black", $x + $dx / 2, 86
            $y - 0.5, $textwinkel, 90, $text ); 87
    } 88
    elseif ( $textausrichtung eq "oben" ) { 89
        beschriftung( $layer + 1, "black", $x + $dx / 2, 90
            $y + $dy, $textwinkel, 90, $text ); 91
    } 92
    else { 93
        beschriftung( 94
            $layer + 1, "black", 95
            $x + $dx / 2, 96
            $y + $textausrichtung, 97
            $textwinkel, "c", $text 98
        ); 99
    } 100
} 101
if ( $welcherbalken eq "b" ) { 102
    if ( $textausrichtung eq "auf" ) { 103

```

```

    beschriftung( $layer + 1, "black", $x, $y + $dy / 2,
        $textwinkel, 0, $text );
}
elseif ( $textausrichtung eq "mitte" ) {
    beschriftung(
        $layer + 1, "black",
        $x + $dx / 2,
        $y + $dy / 2,
        $textwinkel, "c", $text
    );
}
elseif ( $textausrichtung eq "links" ) {
    beschriftung( $layer, "black", $x - 0.5, $y + $dy / 2,
        $textwinkel, "l", $text );
}
elseif ( $textausrichtung eq "rechts" ) {
    beschriftung(
        $layer + 1, "black",
        $x + $dx + $dx / 4,
        $y + $dy / 2,
        $textwinkel, "r", $text
    );
}
else {
    beschriftung(
        $layer + 1, "black",
        $x + $textausrichtung,
        $y + $dy / 2,
        $textwinkel, 90, $text
    );
}
}
}

sub saeuledreieck {
    my (
        $layer,      $x,      $y,      $dx,
        $dy,          $farbe,  $fuellart, $fuellfarbe,
        $linienart,   $linienstaerke, $text,   $textwinkel,
        $textausrichtung, $welcherbalken, $parameter
    ) = @_ ;

    achsengraph( $x,      $y );
    achsengraph( $x + $dx, $y + $dy );

    if ( $welcherbalken eq "s" ) {
        dreieck(
            $layer,      $x,      $y,      "",
            $x + $dx,    $y,      "",      $x + $dx / 2,
            $y + $dy,    "",      $farbe,   $fuellart,
            $fuellfarbe, $linienart, $linienstaerke, $parameter
        );
    }
    if ( $textausrichtung eq "auf" ) {
        beschriftung( $layer + 1, "black", $x + $dx / 2,
            $y, $textwinkel, 90, $text );
    }
    elseif ( $textausrichtung eq "mitte" ) {
        beschriftung(
            $layer + 1, "black",
            $x + $dx / 2,
            $y + $dy / 2,

```

\$textwinkel, "c", \$text	165
);	166
}	167
elseif (\$textausrichtung eq "unten") {	168
beschriftung(\$layer + 1, "black", \$x + \$dx / 2,	169
\$y - 0.5, \$textwinkel, 90, \$text);	170
}	171
elseif (\$textausrichtung eq "oben") {	172
beschriftung(\$layer, "black", \$x + \$dx / 2,	173
\$y + \$dy, \$textwinkel, 90, \$text);	174
}	175
else {	176
beschriftung(	177
\$layer + 1, "black",	178
\$x + \$dx / 2,	179
\$y + \$textausrichtung,	180
\$textwinkel, 90, \$text	181
);	182
}	183
}	184
if (\$welcherbalken eq "b") {	185
dreieck(	186
\$layer, \$x, \$y, "",	187
\$x + \$dx, \$y + \$dy / 2, "", \$x,	188
\$y + \$dy, "", \$farbe, \$fuellart ,	189
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter	190
);	191
if (\$textausrichtung eq "auf") {	192
beschriftung(\$layer + 1, "black", \$x, \$y + \$dy / 2,	193
\$textwinkel, 0, \$text);	194
}	195
elseif (\$textausrichtung eq "mitte") {	196
beschriftung(	197
\$layer + 1, "black",	198
\$x + \$dx / 2,	199
\$y + \$dy / 2,	200
\$textwinkel, "c", \$text	201
);	202
}	203
elseif (\$textausrichtung eq "links") {	204
beschriftung(\$layer + 1, "black", \$x - 0.5, \$y + \$dy / 2,	205
\$textwinkel, "l", \$text);	206
}	207
elseif (\$textausrichtung eq "rechts") {	208
beschriftung(	209
\$layer + 1, "black",	210
\$x + \$dx + \$dx / 4,	211
\$y + \$dy / 2,	212
\$textwinkel, "r", \$text	213
);	214
}	215
else {	216
beschriftung(	217
\$layer + 1, "black",	218
\$x + \$textausrichtung,	219
\$y + \$dy / 2,	220
\$textwinkel, 0, \$text	221
);	222
}	223
}	224
}	225

```

### Tortendiagramm
sub torte {
  my (
    $layer,      $x,      $y,      $winkel,
    $winkelanfang, $radius, $abstand, $farbe,
    $fuellart,    $fuellfarbe, $linienart, $linienstaerke,
    $text,        $textwinkel, $textausrichtung, $parameter
  ) = @_;

  my $xneu = $x + cos( degtorad( $winkelanfang + $winkel / 2 ) ) * $abstand;
  my $yneu = $y + sin( degtorad( $winkelanfang + $winkel / 2 ) ) * $abstand;
  my $xtext =
    $xneu + cos( degtorad( $winkelanfang + $winkel / 2 ) ) * $radius / 2;
  my $ytext =
    $yneu + sin( degtorad( $winkelanfang + $winkel / 2 ) ) * $radius / 2;
  kreisbogen(
    $layer,      $linienart,
    $linienstaerke, $xneu,
    $yneu,      $radius,
    $winkelanfang, $winkelanfang + $winkel,
    $farbe,      $fuellart,
    $fuellfarbe,  "ja",
    $parameter,   ""
  );
  beschriftung( $layer + 1, $farbe, $xtext, $ytext, $textwinkel, 90, $text );
}

### Ringdiagramm
sub ringdiagramm {
  my (
    $layer,      $x,      $y,      $winkel,
    $winkelanfang, $radius, $dicke, $abstand,
    $farbe,      $fuellart, $fuellfarbe, $linienart,
    $linienstaerke, $text,    $textwinkel, $textausrichtung,
    $parameter
  ) = @_;

  my $radiusmitte = $radius - $dicke / 2;
  my $xneu = nachkommastellen(
    $x + cos( degtorad( $winkelanfang + $winkel / 2 ) ) * $abstand, 4 );
  my $yneu = nachkommastellen(
    $y + sin( degtorad( $winkelanfang + $winkel / 2 ) ) * $abstand, 4 );
  my $xtext = nachkommastellen(
    $xneu + cos( degtorad( $winkelanfang + $winkel / 2 ) ) * $radiusmitte,
    4 );
  my $ytext = nachkommastellen(
    $yneu + sin( degtorad( $winkelanfang + $winkel / 2 ) ) * $radiusmitte,
    4 );
  setachsengraph(
    $xneu - $radius,
    $yneu - $radius,
    $xneu + $radius,
    $yneu + $radius
  );
  ring(
    $layer, $linienart, $linienstaerke,
    $xneu, $yneu, $radius,
    $dicke, $winkelanfang, $winkelanfang + $winkel,
    $farbe, $fuellart, $fuellfarbe,
    $parameter
  );
  beschriftung( $layer, "black", $xtext, $ytext, $textwinkel, 90, $text );
}

```

```

}
287
288
### Bilddiagramm
289
290
sub bilddiagramm {
291
  my (
292
    $layer, $bild,      $anzahl,      $xstart,
293
    $y,      $breite,   $winkel,      $abstand,
294
    $text,   $textwinkel, $textausrichtung, $parameter
295
  ) = @_;
296
  my ( $i, $xende );
297
298
  $xende = $xstart + ( $breite + $abstand ) * $anzahl - 1;
299
  minmax( $xstart, $y, $xende * 2, $y + $breite );
300
  setachsengraph( $xstart, $y, $xende, $y );
301
  schleife ( $layer, "nx", $xstart, $xende, $abstand + $breite );
302
  bild( $layer, "\\nx", $y, "", $breite, $winkel, $bild );
303
  schleifenende($layer);
304
  beschriftung(
305
    $layer, "black",
306
    $xstart + $textausrichtung,
307
    $y + bildhoeheberechnet( $bild, $breite ) / 2,
308
    $textwinkel, 0, $text
309
  );
310
}
311
312
### Binomialverteilung
313
sub fac {
314
  my ($n) = @_;
315
  my $ergebnis = 1;
316
  my $i;
317
  for ( $i = $n ; $i > 1 ; $i = $i - 1 ) { $ergebnis = $ergebnis * $i; }
318
  return $ergebnis;
319
}
320
321
sub binom {
322
  my ( $n, $k ) = @_;
323
  my ( $i, $j );
324
  my @temp;
325
  return 0 if ( ( $n < 0 ) || ( $k < 0 ) || ( $k > $n ) );
326
  for ( $i = 0 ; $i <= $n ; $i++ ) {
327
    for ( $j = $i ; $j >= 0 ; $j-- ) {
328
      $temp[$j] =
329
        ( ( $j == $i ) || ( $j == 0 ) )
330
        ? 1
331
        : $temp[$j] + $temp[ $j - 1 ];
332
    }
333
  }
334
  return $temp[$k];
335
}
336
337
my $strich = 0;
338
339
sub setbinostrich { $strich = 1; }
340
341
sub binomial {
342
  my (
343
    $layer,      $n,          $p,          $aufsummiert,
344
    $breite,     $farbe,      $fuellart,   $fuellfarbe,
345
    $linienart,  $linienstaerke, $markierung, $parameter
346
  ) = @_;
347

```

```

binomialteil(
    $layer,      $n,      $p,      0,
    $n,          $aufsummiert,$breite,  $farbe,
    $fuellart,   $fuellfarbe, $linienart, $linienstaerke,
    $markierung,$parameter
);
}

## Eine Strichliste wird erzeugt

sub strichliste {
    my (
        $layer, $xstart,      $ystart,
        $wert,  $hoehe,      $abstand,
        $farbe, $linienstaerke, $linienart
    ) = @_;
    my ( $x, $y, $i, $rest, $anzahl, $abstandfuenf );

    $anzahl = abrunden( $wert / 5, 1 );
    $rest   = $wert % 5;
    $y      = $ystart;

    for ( $i = 1 ; $i <= $wert ; $i++ ) {
        $x = $xstart + $i * $abstand;

        if ( $i % 5 == 0 ) {
            linie (
                $layer, $x - 4 * $abstand,
                $y, "",
                $x - 1 * $abstand,
                $y + $hoehe,
                "", $linienart, $linienstaerke, $farbe, "", ""
            );
        }
        else {
            if ( $i >= $wert - $rest ) {
                linie ( $layer, $x, $y, "", $x, $y + $hoehe,
                    "", $linienart, $linienstaerke, $farbe, "", "" );
            }
        }
    }

    for ( $i = 1 ; $i <= $anzahl ; $i++ ) {
        $x = $xstart + $i * $abstand * 5;
        schleifeanzahl( $layer, "nx", $x - 4 * $abstand, 4, $abstand );
        linie ( $layer, "\\nx", $y, "", "\\nx", $y + $hoehe,
            "", $linienart, $linienstaerke, $farbe, "", "" );
        schleifenende($layer);
    }
}

## Die Binomialverteilung fuer einen Teil zeichnen
sub binomialteil {
    my (
        $layer,      $n,      $p,      $anfang,
        $ende,       $aufsummiert,$breite,  $farbe,
        $fuellart,   $fuellfarbe, $linienart, $linienstaerke,
        $markierung,$parameter

```

```

) = @_;
#      setmitte ();
my $q      = 1 - $p;
my $bern2 = 0;
my $i      = 0;
my ( $bern, $bino, $max, $yschritte );
my $farbeersatz      = getfarbe();
my $fuellartersatz    = getfuellart();
my $fuellfarbeersatz  = getfuellfarbe();
my $linienstaerkeersatz = getlinienstaerke();
my $linienartersatz    = getlinienart();

minmax( -$n, -1, $n, 1 );

if ( $aufsummiert eq "ja" ) { setymaxstat(1); }
else {
  for ( $i = 0 ; $i <= $n ; $i++ ) {
    $bino = binom( $n, $i );
    $max = $bino * $p**($i) * $q**($n - $i );
    setymaxstat($max);
  }
}

if ( $markierung == 0 ) {
  if ( $n > 9 ) {
    $markierung = schoenetabelle( $n * $breite * 100 ) / 100;
  }
  else { $markierung = 1; }
}
else { $markierung = $markierung; }
setymaxstat( aufrunden( getachsengraph("ymax") * 10, 1 ) / 10 );
$yschritte = schoenetabelle( getachsengraph("ymax") * 10 ) / 10;
setymaxstat( $yschritte * 10 );

if ( getgitter () == 1 ) {
  kariert(
    $layer,          "lightgray",
    0.8,             "solid",
    0,               0,
    $n * $breite + $breite + 1, getachsengraph("ymax"),
    $n * $breite + $breite + 1, 0,
    0,               $yschritte,
    0,               0
  );
  setgitter (2);
}

if ( getachsen() == 1 ) {
  achsen(
    $layer + 1,      getfarbe(),
    getlinienstaerke(), $linienart,
    getpfeilart(),   0,
    $n * $breite + $breite + 1, 0,
    getachsengraph("ymax"), $n * $breite + $markierung,
    "normal",         getmarkierung("x"),
    $yschritte,        "normal",
    getmarkierung("y")
  );
}

```

```

if ( $n > 1 ) {
  pstricks( $layer, "" );
  einstellungen( $layer, $farbe, $fuellart, $fuellfarbe, $linienart,
    $linienstaerke, $parameter );

  for ( $i = 0 ; $i <= $n ; $i++ ) {
    $bern = binom( $n, $i );
    $bern = $bern * $p**( $i ) * $q**( $n - $i );
    if ( $aufsummiert ne "nein" ) {
      $bern2 = $bern2 + nachkommastellen( $bern, 4 );
    }
    else { $bern2 = nachkommastellen( $bern, 4 ); }
    if ( $p != 0 ) {
      if ( ( $i >= $anfang ) && ( $i <= $ende ) ) {
        if ( $strich == 1 ) {
          linie (
            $layer,
            $i * $breite + $breite / 2,
            0,
            "",
            $i * $breite + $breite / 2,
            $bern2,
            "",
            $linienart,
            $linienstaerke,
            $farbe,
            $parameter,
            "| - |"
          );
        }
        else {
          saeule(
            $layer,      $i * $breite,
            0,           $breite,
            $bern2,      $farbe,
            $fuellart,   $fuellfarbe,
            $linienart,  $linienstaerke,
            "",          0,
            0,           "s",
            $parameter
          );
        }
      }
    }
  }
  if ( $markierung == $n ) { }
  else {
    if ( ( $i % $markierung ) == 0 ) {
      if ( getachsen() == 1 ) {
        xmarkierungohne( $layer, $i * $breite + $breite / 2,
          $i, "black" );
      }
    }
  }
  pstricks( $layer, "}" );
}
else {
  for ( $i = 0 ; $i <= $n ; $i++ ) {
    $bern = binom( $n, $i );
    $bern = $bern * $p**( $i ) * $q**( $n - $i );
    if ( $aufsummiert ne "nein" ) {

```

```

    $bern2 = $bern2 + sprintf( "%0.4f", $bern );
}
else { $bern2 = nachkommastellen( $bern, 4 ); }
if ( $p != 0 ) {
    if ( ( $i >= $anfang ) && ( $i <= $ende ) ) {
        if ( $strich == 1 ) {
            linie (
                $layer,
                $i * $breite + $breite / 2,
                0,
                "",
                $i * $breite + $breite / 2,
                $bern2,
                "",
                $linienart,
                $linienstaerke,
                $farbe,
                $parameter,
                "| - |"
            );
        }
        else {
            saeule(
                $layer,    $i * $breite,
                0,        $breite,
                $bern2,    $farbe,
                $fuellart, $fuellfarbe,
                $linienart, $linienstaerke,
                "",        0,
                0,        "s",
                $parameter
            );
        }
    }
}
if ( $markierung == $n ) { }
else {
    if ( ( $i % $markierung ) == 0 ) {
        if ( getachsen() == 1 ) {
            xmarkierung( $layer, $i * $breite + $breite / 2,
                $i, "black" );
        }
    }
}
}
}
}

xmarkierungmulti( $layer, 0, $n, $breite, getfarbe() );

setachsen(2);
setfarbe($farbeersatz);
setfuellart ( $fuellartersatz );
setfuellfarbe ( $fuellfarbeersatz );
setlinienstaerke ( $linienstaerkeersatz );
setlinienart ( $linienartersatz );
}
1;

```

531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588

A.5. Das Modul für dreidimensionale Zeichnungen

Listing A.5: dreidimensional.pm

```

#####
# Modul fuer 3d-Zeichenroutinen
#
# Letzte Aenderung: 13.09.2011
#####
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;
my (
    $alpha,          $beta,          $coortype,      $alphatrue,
    $betatrue,        $coortypetrue, $dreidimensional, $pstricks,
    $pstrickstrue,    $strichx,      $strichy,        $strichz,
    $markierungx,     $markierungy, $markierungz,    $markierung,
    $markierungsdicke, $xmin,        $ymin,          $xmax,
    $ymax,            $zmin,        $zmax,          $erstex1,
    $erstex2,         $erstey1,     $erstey2,      $erstex1,
    $erstex2,
);
$dreidimensional = 0;
$alpha           = 45;
$beta            = 30;
$coortype        = 2;
$alphatrue       = $betatrue = $coortypetrue = 0;
$pstricks        = 0;
$pstrickstrue    = 0;
$strichx         = $strichy = $strichz = 0.2;
$markierungx     = -0.5;
$markierungy     = 0.5;
$markierungz     = -0.4;
$markierungsdicke = 0.6;
my $beschriftung = 0;

$markierung = 0;
#####
# Wird 3d gezeichnet ?
#####
sub set3dzeichnen {
    my ($wert) = @_ ;
    $dreidimensional = $wert;
}

sub get3dzeichnen { return $dreidimensional; }
#####
# Wie 3d zeichnen?
#####
sub setpstricks {
    my ($wert) = @_ ;
    $pstrickstrue = $wert;
}

sub setbeschriftung { $beschriftung = 1; }

my $punkt = 0;
sub setpunkt { $punkt = 1; }

```

```

sub getpstricks { return $pstrickstrue; }
58
59
sub set3dpstricks {
60
    my ($wert) = @_ ;
61
    if ( getpstricks() == 0 ) {
62
        $pstricks = $wert;
63
        setpstricks(1);
64
    }
65
    else { $pstricks = $pstricks; }
66
}
67
68
sub get3dpstricks { return $pstricks; }
69
#####
70
# Winkel und Koordinaten fuer 3d setzen
71
#####
72
sub set3dwinkel {
73
    my ( $wert, $alphaoderbeta ) = @_ ;
74
    set3dzeichnen(1);
75
    if ( $alphaoderbeta eq "alpha" ) {
76
        if ( get3d("alpha") == 0 ) {
77
            $alpha = $wert;
78
            set3d( 1, "alpha" );
79
        }
80
        else { $alpha = $alpha; }
81
    }
82
    elsif ( $alphaoderbeta eq "beta" ) {
83
        if ( get3d("beta") == 0 ) {
84
            set3d( 1, "beta" );
85
            $beta = $wert;
86
        }
87
        else { $beta = $beta; }
88
    }
89
    else { printf("%% Kein Winkel gesetzt \n"); }
90
}
91
92
sub get3dwinkel {
93
    my ($alphaoderbeta) = @_ ;
94
    if ( $alphaoderbeta eq "alpha" ) { return ($alpha); }
95
    elsif ( $alphaoderbeta eq "beta" ) { return ($beta); }
96
    else { return printf("%% Kein Winkel gesetzt \n"); }
97
}
98
99
sub set3dkoor {
100
    my ($wert) = @_ ;
101
    set3dzeichnen(1);
102
    if ( $wert > 3 ) { $wert = 2; }
103
    if ( get3d("coor") == 0 ) {
104
        $coortype = $wert;
105
        set3d( 1, "coor" );
106
    }
107
    else { $coortype = $coortype; }
108
}
109
110
sub get3dkoor { return $coortype; }
111
112
sub set3d {
113
    my ( $wert, $alphaoderbeta ) = @_ ;
114
    set3dzeichnen(1);
115
    if ( $alphaoderbeta eq "alpha" ) { $alphatruue = $wert; }
116
    elsif ( $alphaoderbeta eq "beta" ) { $betatruue = $wert; }
117
    elsif ( $alphaoderbeta eq "coor" ) { $coortypetruue = $wert; }
118
}

```

```

else                                     { printf("%% Kein Winkel gesetzt \n"); }
}
sub get3d {
my ($alphaoderbeta) = @_;
if ( $alphaoderbeta eq "alpha" ) { return $alphatrue; }
elsif ( $alphaoderbeta eq "beta" ) { return $betatrue; }
elsif ( $alphaoderbeta eq "coor" ) { return $coortypetrue; }
else { return printf("%% Kein Winkel gesetzt \n"); }
}

sub x3d {
my ( $x, $y, $z ) = @_;
set3dzeichnen(1);
if ( ( istzahl($x) == 1 ) && ( istzahl($y) == 1 ) && ( istzahl($z) == 1 ) )
{
if ( $coortype == 0 ) {
return nachkommastellen(
-(
$x +
getskalierung("x") *
cos( degtorad( get3dwinkel("alpha") ) )
) +
$y *
getskalierung("y") *
sin( degtorad( get3dwinkel("alpha") ) ),
4
);
}
elsif ( $coortype == 1 ) {
return nachkommastellen(
(
$y *
getskalierung("y") -
$x *
getskalierung("x") *
sin( degtorad( get3dwinkel("alpha") ) - degtorad(90) )
),
4
);
}
elsif ( $coortype == 2 ) {
return nachkommastellen(
( $y * getskalierung("y") - ( 0.5 * $x * getskalierung("x") ) ),
4
);
}
elsif ( $coortype == 3 ) {
return nachkommastellen(
$y * getskalierung("y") + ( 0.5 * $x * getskalierung("x") ),
4
);
}
else { return "ERROR"; }
}
else { return "ERROR"; }
}

sub y3d {
my ( $x, $y, $z ) = @_;
set3dzeichnen(1);

```

```

if ( ( istzahl($x) == 1 ) && ( istzahl($y) == 1 ) && ( istzahl($z) == 1 ) )
{
    if ( $coortype == 0 ) {
        return nachkommastellen(
            -(
                $x *
                getskalierung("x") *
                sin( degtorad( get3dwinkel("alpha") ) ) +
                $y *
                getskalierung("y") *
                cos( degtorad( get3dwinkel("alpha") ) )
            ) *
            sin( degtorad( get3dwinkel("beta") ) ) +
            $z *
            getskalierung("z") *
            cos( degtorad( get3dwinkel("beta") ) ),
            4
        );
    }
    elseif ( $coortype == 1 ) {
        return nachkommastellen(
            $z *
            getskalierung("z") -
            $x *
            getskalierung("x") *
            cos( degtorad( get3dwinkel("alpha") ) - degtorad(90) ),
            4
        );
    }
    elseif ( $coortype == 2 ) {
        return nachkommastellen(
            ( $z * getskalierung("z") - ( 0.5 * $x * getskalierung("x") ) ),
            4
        );
    }
    elseif ( $coortype == 3 ) {
        return nachkommastellen(
            $z * getskalierung("z") + ( 0.5 * $x * getskalierung("x") ),
            4
        );
    }
    else { return "ERROR"; }
}
else { return "ERROR"; }
}

#####
# Hilfsfunktionen fuer 3d
#####

sub calcfunk3d {
    my $funktionsausdruck = shift;
    my $variablex          = shift;
    my $wertx              = shift;
    my $variabley          = shift;
    my $werty              = shift;
    my @ergebnis           = regelwerk( tokenizer($funktionsausdruck) );

    return rechenbaum_ausrechnen(
        $ergebnis[0]{ 'Baum' },
        {
            $variablex => $wertx,

```

```

    $variabley => $wert,
    'pi'       => pi(),
    'e'        => exp(1)
  }
);
}

# Berechnet den maximalen und minimalen z-Wert
sub calczminmax {
  my ( $func, $xminwert, $yminwert, $xmaxwert, $ymaxwert, $minmax ) = @_;
  my ( $bereichmin, $bereichmax, $schrittx, $schritty, $xsuche, $ysuche,
    $zsuche, $schrittzahl );
  $schrittzahl = 50;

  $bereichmin = $bereichmax = 0;

  for ( $schrittx = 0 ; $schrittx < $schrittzahl ; $schrittx++ ) {
    $xsuche = $xminwert +
      ( $xmaxwert - $xminwert ) * ( $schrittx / ( $schrittzahl - 1 ) );
    for ( $schritty = 0 ; $schritty < $schrittzahl ; $schritty++ ) {
      $ysuche =
        $yminwert +
        ( $ymaxwert - $yminwert ) *
        ( $schritty / ( $schrittzahl - 1 ) );

      $zsuche = calcfunk3d( $func, "x", $xsuche, "y", $ysuche );
      if ( $zsuche < $bereichmin ) { $bereichmin = $zsuche; }
      if ( $zsuche > $bereichmax ) { $bereichmax = $zsuche; }
    }
  }

  return sprintf( "%0.4f", $bereichmin ) if ( $minmax eq "min" );
  return sprintf( "%0.4f", $bereichmax ) if ( $minmax eq "max" );
  printf( "calcxminmax: Unbekannter Parameter minmax='%s'\n", $minmax );
}

sub calczmin {
  my ( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert ) = @_;

  return sprintf( "%0.4f",
    calczminmax( $func, $xminwert, $yminwert, $xmaxwert, $ymaxwert, "min" )
  );
}

sub calczmax {
  my ( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert ) = @_;
  return sprintf( "%0.4f",
    calczminmax( $func, $xminwert, $yminwert, $xmaxwert, $ymaxwert, "max" )
  );
}

$xmin = $ymin = $xmax = $ymax = $zmin = $zmax = 0;
$erstex1 = $erstex2 = $erstey1 = $erstey2 = $erstez1 = $erstez2 = 0;

#package dateierzeugen ;
# Minimum und Maximum setzen
sub setminmax3d {
  my ( $min, $xyz, $minmax ) = @_;
  if ( $xyz eq "x" ) {
    if ( $minmax eq "min" ) { $xmin = $min; }

```

```

    else                { $xmax = $min; }
}
if ( $xyz eq "y" ) {
    if ( $minmax eq "min" ) { $ymin = $min; }
    else                { $ymax = $min; }
}
if ( $xyz eq "z" ) {
    if ( $minmax eq "min" ) { $zmin = $min; }
    else                { $zmax = $min; }
}
}

sub getminmax3d {
    my ( $xyz, $minmax ) = @_;
    if ( $xyz eq "x" ) {
        if ( $minmax eq "min" ) { return sprintf( "%0.4f", $xmin ); }
        else                { return sprintf( "%0.4f", $xmax ); }
    }
    if ( $xyz eq "y" ) {
        if ( $minmax eq "min" ) { return sprintf( "%0.4f", $ymin ); }
        else                { return sprintf( "%0.4f", $ymax ); }
    }
    if ( $xyz eq "z" ) {
        if ( $minmax eq "min" ) { return sprintf( "%0.4f", $zmin ); }
        else                { return sprintf( "%0.4f", $zmax ); }
    }
}

sub grenzen3d {
    my ( $x, $y, $z ) = @_;
    minmax3d( $x, $y, $z, $x, $y, $z );
}

sub minmax3d {
    my ( $x1, $y1, $z1, $x2, $y2, $z2 ) = @_;

    if ( istzahl($x1) == 1 ) {
        if ( $erstex1 == 0 ) {
            setminmax3d( $x1, "x", "min" );
            $erstex1 = 1;
        }
        else {
            if ( $x1 < getminmax3d( "x", "min" ) ) {
                setminmax3d( $x1, "x", "min" );
            }
        }
    }
    if ( istzahl($x2) == 1 ) {
        if ( $erstex2 == 0 ) {
            setminmax( $x2, "x", "max" );
            $erstex2 = 1;
        }
        else {
            if ( $x2 > getminmax3d( "x", "max" ) ) {
                setminmax3d( $x2, "x", "max" );
            }
        }
    }
    if ( istzahl($y1) == 1 ) {

```

```

if ( $erstey1 == 0 ) {
    setminmax3d( $y1, "y", "min" );
    $erstex1 = 1;
}
else {
    if ( $y1 < getminmax3d( "y", "min" ) ) {
        setminmax3d( $y1, "y", "min" );
    }
}
}
if ( istzahl($y2) == 1 ) {
    if ( $erstey2 == 0 ) {
        setminmax( $y2, "y", "max" );
        $erstey2 = 1;
    }
    else {
        if ( $y2 > getminmax3d( "y", "max" ) ) {
            setminmax3d( $y2, "y", "max" );
        }
    }
}
if ( istzahl($z1) == 1 ) {
    if ( $erstez1 == 0 ) {
        setminmax3d( $z1, "z", "min" );
        $erstez1 = 1;
    }
    else {
        if ( $z1 < getminmax3d( "z", "min" ) ) {
            setminmax3d( $z1, "z", "min" );
        }
    }
}
if ( istzahl($z2) == 1 ) {
    if ( $erstez2 == 0 ) {
        setminmax( $z2, "z", "max" );
        $erstez2 = 1;
    }
    else {
        if ( $z2 > getminmax3d( "z", "max" ) ) {
            setminmax3d( $z2, "z", "max" );
        }
    }
}
if ( ( istzahl($x1) == 0 )
    && ( istzahl($y1) == 1 )
    && ( istzahl($z1) == 1 ) )
{
    minmax(
        x3d( $x1, 0, 0 ),
        y3d( $x1, 0, 0 ),
        x3d( $x1, 0, $z1 ),
        y3d( $x1, 0, 0 )
    );
    minmax(
        x3d( 0, $y1, 0 ),
        y3d( 0, $y1, 0 ),
        x3d( 0, $y1, 0 ),
        y3d( 0, $y1, 0 )
    );
}

```

minmax(	424
x3d(0, 0, \$z1),	425
y3d(0, 0, \$z1),	426
x3d(0, 0, \$z1),	427
y3d(0, 0, \$z1)	428
);	429
minmax(	430
x3d(0, \$y1, \$z1),	431
y3d(0, \$y1, \$z1),	432
x3d(0, \$y1, \$z1),	433
y3d(0, \$y1, \$z1)	434
);	435
minmax(	436
x3d(\$x1, 0, \$z1),	437
y3d(\$x1, 0, \$z1),	438
x3d(\$x1, 0, \$z1),	439
y3d(\$x1, 0, \$z1)	440
);	441
minmax(	442
x3d(\$x1, \$y1, 0),	443
y3d(\$x1, \$y1, 0),	444
x3d(\$x1, \$y1, 0),	445
y3d(\$x1, \$y1, 0)	446
);	447
minmax(	448
x3d(\$x1, \$y1, \$z1),	449
y3d(\$x1, \$y1, \$z1),	450
x3d(\$x1, \$y1, \$z1),	451
y3d(\$x1, \$y1, \$z1)	452
);	453
}	454
if ((istzahl(\$x2) == 1)	455
&& (istzahl(\$y2) == 1)	456
&& (istzahl(\$z2) == 1))	457
{	458
minmax(	459
x3d(\$x2, 0, 0),	460
y3d(\$x2, 0, 0),	461
x3d(\$x2, 0, \$z2),	462
y3d(\$x2, 0, 0)	463
);	464
minmax(	465
x3d(0, \$y2, 0),	466
y3d(0, \$y2, 0),	467
x3d(0, \$y2, 0),	468
y3d(0, \$y2, 0)	469
);	470
minmax(	471
x3d(0, 0, \$z2),	472
y3d(0, 0, \$z2),	473
x3d(0, 0, \$z2),	474
y3d(0, 0, \$z2)	475
);	476
minmax(	477
x3d(0, \$y2, \$z2),	478
y3d(0, \$y2, \$z2),	479
x3d(0, \$y2, \$z2),	480
y3d(0, \$y2, \$z2)	481
);	482
minmax(	483
x3d(\$x2, 0, \$z2),	484

```

    y3d( $x2, 0, $z2 ),
    x3d( $x2, 0, $z2 ),
    y3d( $x2, 0, $z2 )
);
minmax(
    x3d( $x2, $y2, 0 ),
    y3d( $x2, $y2, 0 ),
    x3d( $x2, $y2, 0 ),
    y3d( $x2, $y2, 0 )
);
minmax(
    x3d( $x2, $y2, $z2 ),
    y3d( $x2, $y2, $z2 ),
    x3d( $x2, $y2, $z2 ),
    y3d( $x2, $y2, $z2 )
);
}
}

my (
    $xminachse, $xmaxachse, $yminachse, $ymaxachse, $zminachse, $zmaxachse,
    $xsetzen1, $xsetzen2, $ysetzen1, $ysetzen2, $zsetzen1, $zsetzen2
);

$xminachse = $xmaxachse = $yminachse = $ymaxachse = $zminachse = $zmaxachse =
$xsetzen1 = $xsetzen2 = $ysetzen1 = $ysetzen2 = $zsetzen1 = $zsetzen2 = 0;

sub setachsengraph3d {
    my ( $x1, $y1, $z1, $x2, $y2, $z2 ) = @_ ;

    if ( ( istzahl($x1) == 1 )
        && ( istzahl($y1) == 1 )
        && ( istzahl($z1) == 1 )
        && ( istzahl($x2) == 1 )
        && ( istzahl($y2) == 1 )
        && ( istzahl($z2) == 1 ) )
    {

        if ( $xsetzen1 == 0 ) {
            $xminachse = $x1;
            $xsetzen1 = 1;
        }
        else { $xminachse = abrunden( min( $xminachse, $x1 ), 1 ); }

        if ( $xsetzen2 == 0 ) {
            $xmaxachse = $x2;
            $xsetzen2 = 1;
        }
        else { $xmaxachse = aufrunden( max( $xmaxachse, $x2 ), 1 ); }

        if ( $ysetzen1 == 0 ) {
            $yminachse = $y1;
            $ysetzen1 = 1;
        }
        else { $yminachse = abrunden( min( $yminachse, $y1 ), 1 ); }
        if ( $ysetzen2 == 0 ) {
            $ymaxachse = $y2;
            $ysetzen2 = 1;
        }
        else { $ymaxachse = aufrunden( max( $ymaxachse, $y2 ), 1 ); }
    }
}

```

```

if ( $zsetzen1 == 0 ) {
    $zminachse = $z1;
    $zsetzen1 = 1;
}
else { $zminachse = abrunden( min( $zminachse, $z1 ), 1 ); }

if ( $zsetzen2 == 0 ) {
    $zmaxachse = $z2;
    $zsetzen2 = 1;
}
else { $zmaxachse = aufrunden( max( $zmaxachse, $z2 ), 1 ); }
}
}

sub getachsengraph3d {
    my ($text) = @_;
    if ( $text eq "xmin" ) { return $xminachse; }
    if ( $text eq "ymin" ) { return $yminachse; }
    if ( $text eq "zmin" ) { return $zminachse; }
    if ( $text eq "xmax" ) { return $xmaxachse; }
    if ( $text eq "ymax" ) { return $ymaxachse; }
    if ( $text eq "zmax" ) { return $zmaxachse; }
}

#####
##### Berechnungen fuer Vektoren
#####
sub calcendpunkt3d {
    my ( $x, $laenge, $w1, $w2 ) = @_;
    set3dzeichnen(1);
    return $x + cos( degtorad($w2) ) * $laenge * cos( degtorad($w1) );
}

sub calcendpunkty3d {
    my ( $y, $laenge, $w1, $w2 ) = @_;
    set3dzeichnen(1);
    return $y + sin( degtorad($w2) ) * $laenge * cos( degtorad($w1) );
}

sub calcendpunktz3d {
    my ( $z, $laenge, $w1, $w2 ) = @_;
    set3dzeichnen(1);
    return $z + sin( degtorad($w1) ) * $laenge;
}

#####
# Gitter im 3d
#####
sub kariert3d {
    my (
        $layer,    $farbe,    $linienstaerke, $linienart,
        $xmin,    $ymin,    $zmin,    $xmax,
        $ymax,    $zmax,    $abstandx,    $dickx,
        $ganzdickx, $abstandy, $dicky,    $ganzdicky,
        $abstandz, $dickz,    $ganzdickz,
    ) = @_;

    my $knotenersatz = getknoten();

```

```

setknoten(0);
my ( $i, $j, $k );

if ( get3dpstricks() == 1 ) {

    if ( $abstandx != 0 ) {
        for ( $i = $xmin ; $i <= $xmax ; $i = $i + $abstandx ) {
            minmax3d( $i, 0, $zmin, $i, 0, $zmax );
            minmax3d( $i, $ymin, 0, $i, $ymax, 0 );
        }

        schleife ( $layer, "nx", $xmin, $xmax, $abstandx );
        linie ( $layer, "\\nx", 0, $zmin, "\\nx", 0, $zmax, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie ( $layer, "\\nx", $ymin, 0, "\\nx", $ymax, 0, $linienart,
            $linienstaerke, $farbe, "", "" );
        schleifenende($layer);
    }

    if ( $dickx != 0 ) {
        schleife ( $layer, "nx", $xmin, $xmax, $dickx );
        linie ( $layer, "\\nx", 0, $zmin, "\\nx", 0, $zmax, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie ( $layer, "\\nx", $ymin, 0, "\\nx", $ymax, 0, $linienart,
            $linienstaerke, $farbe, "", "" );
        schleifenende($layer);
    }

    if ( $ganzdickx != 0 ) {
        schleife ( $layer, "nx", $xmin, $xmax, $ganzdickx );
        linie ( $layer, "\\nx", 0, $zmin, "\\nx", 0, $zmax, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie ( $layer, "\\nx", $ymin, 0, "\\nx", $ymax, 0, $linienart,
            $linienstaerke, $farbe, "", "" );
        schleifenende($layer);
    }

    if ( $abstandy != 0 ) {
        for ( $i = $ymin ; $i <= $ymax ; $i = $i + $abstandy ) {
            minmax3d( 0, $i, $zmin, 0, $i, $zmax );
            minmax3d( $xmin, $i, 0, $xmax, $i, 0 );
        }
        schleife ( $layer, "ny", $ymin, $ymax, $abstandy );
        linie ( $layer, 0, "\\ny", $zmin, 0, "\\ny", $zmax, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie ( $layer, $xmin, "\\ny", 0, $xmax, "\\ny", 0, $linienart,
            $linienstaerke, $farbe, "", "" );
        schleifenende($layer);
    }

    if ( $dicky != 0 ) {
        schleife ( $layer, "ny", $ymin, $ymax, $dicky );
        linie ( $layer, 0, "\\ny", $zmin, 0, "\\ny", $zmax, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie ( $layer, $xmin, "\\ny", 0, $xmax, "\\ny", 0, $linienart,
            $linienstaerke, $farbe, "", "" );
        schleifenende($layer);
    }
}

```

```

if ( $ganzdicky != 0 ) {
    schleife( $layer, "ny", $ymin, $ymax, $ganzdicky );
    linie( $layer, 0, "\\ny", $zmin, 0, "\\ny", $zmax, $linienart,
        $linienstaerke, $farbe, "", "" );
    linie( $layer, $xmin, "\\ny", 0, $xmax, "\\ny", 0, $linienart,
        $linienstaerke, $farbe, "", "" );
    schleifenende($layer);
}

if ( $abstandz != 0 ) {
    for ( $i = $zmin ; $i <= $zmax ; $i = $i + $abstandz ) {
        minmax3d( 0, $ymin, $i, 0, $ymax, $i );
        minmax3d( $xmin, 0, $i, $xmax, 0, $i );
    }
    schleife( $layer, "nz", $zmin, $zmax, $abstandz );
    linie( $layer, 0, $ymin, "\\nz", 0, $ymax, "\\nz", $linienart,
        $linienstaerke, $farbe, "", "" );
    linie( $layer, $xmin, 0, "\\nz", $xmax, 0, "\\nz", $linienart,
        $linienstaerke, $farbe, "", "" );
    schleifenende($layer);
}

if ( $dickz != 0 ) {
    schleife( $layer, "nz", $zmin, $zmax, $dickz );
    linie( $layer, 0, $ymin, "\\nz", 0, $ymax, "\\nz", $linienart,
        $linienstaerke, $farbe, "", "" );
    linie( $layer, $xmin, 0, "\\nz", $xmax, 0, "\\nz", $linienart,
        $linienstaerke, $farbe, "", "" );
    schleifenende($layer);
}

if ( $ganzdickz != 0 ) {
    schleife( $layer, "nz", $zmin, $zmax, $ganzdickz );
    linie( $layer, 0, $ymin, "\\nz", 0, $ymax, "\\nz", $linienart,
        $linienstaerke, $farbe, "", "" );
    linie( $layer, $xmin, 0, "\\nz", $xmax, 0, "\\nz", $linienart,
        $linienstaerke, $farbe, "", "" );
    schleifenende($layer);
}

}

else {
    if ( $abstandx != 0 ) {
        for ( $i = $xmin ; $i <= $xmax ; $i = $i + $abstandx ) {
            linie( $layer, $i, 0, $zmin, $i, 0, $zmax, $linienart,
                $linienstaerke, $farbe, "", "" );
            linie( $layer, $i, $ymin, 0, $i, $ymax, 0, $linienart,
                $linienstaerke, $farbe, "", "" );
        }
    }

    if ( $dickx != 0 ) {
        for ( $i = $xmin ; $i <= $xmax ; $i = $i + $dickx ) {
            linie( $layer, $i, 0, $zmin, $i, 0, $zmax, $linienart,
                $linienstaerke, $farbe, "", "" );
            linie( $layer, $i, $ymin, 0, $i, $ymax, 0, $linienart,
                $linienstaerke, $farbe, "", "" );
        }
    }

    if ( $ganzdickx != 0 ) {
        for ( $i = $xmin ; $i <= $xmax ; $i = $i + $ganzdickx ) {
            linie( $layer, $i, 0, $zmin, $i, 0, $zmax, $linienart,

```

```

        $linienstaerke, $farbe, "", "" );
linie( $layer, $i, $ymin, 0, $i, $ymax, 0, $linienart,
        $linienstaerke, $farbe, "", "" );
    }
}

if ( $abstandy != 0 ) {
    for ( $i = $ymin ; $i <= $ymax ; $i = $i + $abstandy ) {
        linie( $layer, 0, $i, $zmin, 0, $i, $zmax, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie( $layer, $xmin, $i, 0, $xmax, $i, 0, $linienart,
            $linienstaerke, $farbe, "", "" );
    }
}

if ( $dicky != 0 ) {
    for ( $i = $ymin ; $i <= $ymax ; $i = $i + $dicky ) {
        linie( $layer, 0, $i, $zmin, 0, $i, $zmax, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie( $layer, $xmin, $i, 0, $xmax, $i, 0, $linienart,
            $linienstaerke, $farbe, "", "" );
    }
}

if ( $ganzdicky != 0 ) {
    for ( $i = $ymin ; $i <= $ymax ; $i = $i + $ganzdicky ) {
        linie( $layer, 0, $i, $zmin, 0, $i, $zmax, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie( $layer, $xmin, $i, 0, $xmax, $i, 0, $linienart,
            $linienstaerke, $farbe, "", "" );
    }
}

if ( $abstandz != 0 ) {
    for ( $i = $zmin ; $i <= $zmax ; $i = $i + $abstandz ) {
        linie( $layer, 0, $ymin, $i, 0, $ymax, $i, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie( $layer, $xmin, 0, $i, $xmax, 0, $i, $linienart,
            $linienstaerke, $farbe, "", "" );
    }
}

if ( $dickz != 0 ) {
    for ( $i = $zmin ; $i <= $zmax ; $i = $i + $dickz ) {
        linie( $layer, 0, $ymin, $i, 0, $ymax, $i, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie( $layer, $xmin, 0, $i, $xmax, 0, $i, $linienart,
            $linienstaerke, $farbe, "", "" );
    }
}

if ( $ganzdickz != 0 ) {
    for ( $i = $zmin ; $i <= $zmax ; $i = $i + $ganzdickz ) {
        linie( $layer, 0, $ymin, $i, 0, $ymax, $i, $linienart,
            $linienstaerke, $farbe, "", "" );
        linie( $layer, $xmin, 0, $i, $xmax, 0, $i, $linienart,
            $linienstaerke, $farbe, "", "" );
    }
}
}

setknoten($knotenersatz);

#####

```

# Achsen im 3d	790
#####	791
	792
sub achsen3d {	793
my (	794
\$layer, \$farbe, \$linienstaerke, \$linienart,	795
\$pfeilart, \$xmin, \$xmax, \$ymin,	796
\$ymax, \$zmin, \$zmax, \$dx,	797
\$xskala, \$xtext, \$dy, \$yskala,	798
\$ytext, \$dz, \$zskala, \$ztext	799
) = @_;	800
	801
my \$knotenersatz = getknoten();	802
	803
if (getgitter () == 1) {	804
kariert3d(1, "lightgray", 0.25, "solid", \$xmin, \$ymin, \$zmin, \$xmax,	805
\$ymax, \$zmax, \$dx, 0, 0, \$dy, 0, 0, \$dz, 0, 0);	806
setgitter (2);	807
}	808
	809
setknoten(0);	810
	811
setgrossskalierung3d(\$xmax - \$xmin, \$ymax - \$ymin, \$zmax - \$zmin);	812
setachsengraph3d(\$xmin, \$ymin, \$zmin, \$xmax, \$ymax, \$zmax);	813
	814
if ((getgitter () == 1) (getgitter () == 3)) {	815
kariert3d(\$layer, "lightgray", 0.25, "solid", \$xmin, \$ymin, \$zmin,	816
\$xmax, \$ymax, \$zmax, \$dx, 0, 0, \$dy, 0, 0, \$dz, 0, 0);	817
setgitter (2);	818
}	819
	820
minmax3d(\$xmin, 0, 0, \$xmax, 0, 0);	821
minmax3d(0, \$ymin, 0, \$ymax, 0, 0);	822
minmax3d(0, 0, \$zmin, 0, 0, \$zmax);	823
	824
my (\$parameterweiter, \$i, \$j, \$k);	825
\$parameterweiter = "";	826
\$i = \$j = \$k = 0;	827
	828
if (\$xmin != \$xmax) {	829
for (\$i = \$xmin ; \$i <= \$xmax + \$dx ; \$i = \$i + \$dx) {	830
minmax3d(\$i, 0, 0, \$i, 0, 0);	831
}	832
}	833
if (\$ymin != \$ymax) {	834
for (\$i = \$ymin ; \$i <= \$ymax + \$dy ; \$i = \$i + \$dy) {	835
minmax3d(0, \$i, 0, 0, \$i, 0);	836
}	837
}	838
if (\$zmin != \$zmax) {	839
for (\$i = \$zmin ; \$i <= \$zmax + \$dz ; \$i = \$i + \$dz) {	840
minmax3d(0, 0, \$i, 0, 0, \$i);	841
}	842
}	843
	844
set3dzeichnen(1);	845
	846
if (get3dpstricks() == 1) {	847
	848
\$parameterweiter =	849
\$parameterweiter	850

```

. "comma,"
. "drawing, plane corr=normal,"
. "linecolor="
. $farbe
. ", xMin="
. nachkommastellen( $xmin, 4 )
. ", xMax="
. nachkommastellen( $xmax, 4 )
. ", yMin="
. nachkommastellen( $ymin, 4 )
. ", yMax="
. nachkommastellen( $ymax, 4 )
. ", zMin="
. nachkommastellen( $zmin, 4 )
. ", zMax="
. nachkommastellen( $zmax, 4 )
. ", nameX="
. $xtext
. ", nameY="
. $ytext
. ", nameZ="
. $ztext;

if ( get3dpstricks() == 1 ) {

my $hilfsdicke = $markierungsdicke;
$markierungsdicke = 0;

if ( $xmin != $xmax ) {
  for ( $i = ( 0 + $dx ) ; $i < $xmax ; $i = $i + $dx ) {
    xstrich3d( $layer, $i, $farbe );
  }
  for ( $i = ( 0 - $dx ) ; $i > $xmin ; $i = $i - $dx ) {
    xstrich3d( $layer, $i, $farbe );
  }
}
if ( $ymin != $ymax ) {
  for ( $j = ( 0 + $dy ) ; $j < $ymax ; $j = $j + $dy ) {
    ystrich3d( $layer, $j, $farbe );
  }
  for ( $j = ( 0 - $dy ) ; $j > $ymin ; $j = $j - $dy ) {
    ystrich3d( $layer, $j, $farbe );
  }
}
if ( $zmin != $zmax ) {
  for ( $k = ( 0 + $dz ) ; $k < $zmax ; $k = $k + $dz ) {
    zstrich3d( $layer, $k, $farbe );
  }
  for ( $k = ( 0 - $dz ) ; $k > $zmin ; $k = $k - $dz ) {
    zstrich3d( $layer, $k, $farbe );
  }
}
$markierungsdicke = $hilfsdicke;

if ( $xskala eq "normal" ) {

  if ( $xmin != $xmax ) {
    if ( $xmax != 0 ) {
      schleife( $layer, "nx", 0 + $dx, $xmax - $dx, $dx );
      xmarkierung3d( $layer, "\\nx", "\\nx", $farbe );
      schleifenende($layer);
    }
  }
}

```

```

    }
    if ( $xmin != 0 ) {
        schleife( $layer, "nx", 0 - $dx, $xmin + $dx, -$dx );
        xmarkierung3d( $layer, "\\nx", "\\nx", $farbe );
        schleifenende($layer);
    }
}
912
913
914
915
916
917
918
919
920
if ( $ymin != $ymax ) {
    if ( $ymax != 0 ) {
        schleife( $layer, "ny", 0 + $dy, $ymax - $dy, $dy );
        ymarkierung3d( $layer, "\\ny", "\\ny", $farbe );
        schleifenende($layer);
    }
    if ( $ymin != 0 ) {
        schleife( $layer, "ny", 0 - $dy, $ymin + $dy, -$dy );
        ymarkierung3d( $layer, "\\ny", "\\ny", $farbe );
        schleifenende($layer);
    }
}
921
922
923
924
925
926
927
928
929
930
931
932
933
if ( $zmin != $zmax ) {
    if ( $zmax != 0 ) {
        schleife( $layer, "nz", 0 + $dz, $zmax - $dz, $dz );
        zmarkierung3d( $layer, "\\nz", "\\nz", $farbe );
        schleifenende($layer);
    }
    if ( $zmin != 0 ) {
        schleife( $layer, "nz", 0 - $dz, $zmin + $dz, -$dz );
        zmarkierung3d( $layer, "\\nz", "\\nz", $farbe );
        schleifenende($layer);
    }
}
934
935
936
937
938
939
940
941
942
943
944
945
}
946
}
947
elseif ( $xskala eq "nurstriche" ) {
    if ( $xmin != $xmax ) {
        if ( $xmax != 0 ) {
            schleife( $layer, "nx", 0 + $dx, $xmax - $dx, $dx );
            xstrich3d( $layer, "\\nx", $farbe );
            schleifenende($layer);
        }
        if ( $xmin != 0 ) {
            schleife( $layer, "nx", 0 - $dx, $xmin + $dx, -$dx );
            xstrich3d( $layer, "\\nx", $farbe );
            schleifenende($layer);
        }
    }
    if ( $ymin != $ymax ) {
        if ( $ymax != 0 ) {
            schleife( $layer, "ny", 0 + $dy, $ymax - $dy, $dy );
            ystrich3d( $layer, "\\ny", $farbe );
            schleifenende($layer);
        }
        if ( $ymin != 0 ) {
            schleife( $layer, "ny", 0 - $dy, $ymin + $dy, -$dy );
            ystrich3d( $layer, "\\ny", $farbe );
            schleifenende($layer);
        }
    }
}
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972

```

```

}
if ( $zmin != $zmax ) {
  if ( $zmax != 0 ) {
    schleife( $layer, "nz", 0 + $dz, $zmax - $dz, $dz );
    zstrich3d( $layer, "\\nz", $farbe );
    schleifenende($layer);
  }
  if ( $zmin != 0 ) {
    schleife( $layer, "nz", 0 - $dz, $zmin + $dz, -$dz );
    zstrich3d( $layer, "\\nz", $farbe );
    schleifenende($layer);
  }
}
}
}
elseif ( $xskala eq "nureins" ) {
  if ( $xmin != $xmax ) {
    xmarkierung3d( $layer, 1, 1, $farbe );
  }
  if ( $ymin != $ymax ) {
    ymarkierung3d( $layer, 1, 1, $farbe );
  }
  if ( $zmin != $zmax ) {
    zmarkierung3d( $layer, 1, 1, $farbe );
  }
}
elseif ( $xskala eq "nurstrichundeins" ) {
  if ( $xmin != $xmax ) {
    if ( $xmax != 0 ) {
      schleife( $layer, "nx", 0 + $dx, $xmax - $dx, $dx );
      xstrich3d( $layer, "\\nx", "\\nx", $farbe );
      schleifenende($layer);
    }
    if ( $xmin != 0 ) {
      schleife( $layer, "nx", 0 - $dx, $xmin + $dx, -$dx );
      xstrich3d( $layer, "\\nx", "\\nx", $farbe );
      schleifenende($layer);
    }
  }
  if ( $ymin != $ymax ) {
    if ( $ymax != 0 ) {
      schleife( $layer, "ny", 0 + $dy, $ymax - $dy, $dy );
      ystrich3d( $layer, "\\ny", "\\ny", $farbe );
      schleifenende($layer);
    }
    if ( $ymin != 0 ) {
      schleife( $layer, "ny", 0 - $dy, $ymin + $dy, -$dy );
      ystrich3d( $layer, "\\ny", "\\ny", $farbe );
      schleifenende($layer);
    }
  }
  if ( $zmin != $zmax ) {
    if ( $zmax != 0 ) {
      schleife( $layer, "nz", 0 + $dz, $zmax - $dz, $dz );
      zstrich3d( $layer, "\\nz", "\\nz", $farbe );
      schleifenende($layer);
    }
    if ( $zmin != 0 ) {
      schleife( $layer, "nz", 0 - $dz, $zmin + $dz, -$dz );

```

```

        zstrich3d( $layer, "\\nz", "\\nz", $farbe );
        schleifenende($layer);
    }
}
if ( $xmin != $xmax ) {
    xmarkierung3d( $layer, 1, 1, $farbe );
}
if ( $ymin != $ymax ) {
    ymarkierung3d( $layer, 1, 1, $farbe );
}
if ( $zmin != $zmax ) {
    zmarkierung3d( $layer, 1, 1, $farbe );
}
}
}
else {
    if ( $xskala eq "normal" ) {
        if ( $xmin != $xmax ) {
            for ( $i = ( 0 + $dx ) ; $i < $xmax ; $i = $i + $dx ) {
                xmarkierung3d( $layer, $i, $i, $farbe );
            }
            for ( $i = ( 0 - $dx ) ; $i > $xmin ; $i = $i - $dx ) {
                xmarkierung3d( $layer, $i, $i, $farbe );
            }
        }
        if ( $ymin != $ymax ) {
            for ( $j = ( 0 + $dy ) ; $j < $ymax ; $j = $j + $dy ) {
                ymarkierung3d( $layer, $j, $j, $farbe );
            }
            for ( $j = ( 0 - $dy ) ; $j > $ymin ; $j = $j - $dy ) {
                ymarkierung3d( $layer, $j, $j, $farbe );
            }
        }
        if ( $zmin != $zmax ) {
            for ( $k = ( 0 + $dz ) ; $k < $zmax ; $k = $k + $dz ) {
                zmarkierung3d( $layer, $k, $k, $farbe );
            }
            for ( $k = ( 0 - $dz ) ; $k > $zmin ; $k = $k - $dz ) {
                zmarkierung3d( $layer, $k, $k, $farbe );
            }
        }
    }
}
elseif ( $xskala eq "nurstriche" ) {
    if ( $xmin != $xmax ) {
        for ( $i = ( 0 + $dx ) ; $i < $xmax ; $i = $i + $dx ) {
            xstrich3d( $layer, $i, $farbe );
        }
        for ( $i = ( 0 - $dx ) ; $i > $xmin ; $i = $i - $dx ) {
            xstrich3d( $layer, $i, $farbe );
        }
    }
    if ( $ymin != $ymax ) {
        for ( $j = ( 0 + $dy ) ; $j < $ymax ; $j = $j + $dy ) {
            ystrich3d( $layer, $j, $farbe );
        }
        for ( $j = ( 0 - $dy ) ; $j > $ymin ; $j = $j - $dy ) {
            ystrich3d( $layer, $j, $farbe );
        }
    }
    if ( $zmin != $zmax ) {

```

1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094

```

    for ( $k = ( 0 + $dz ) ; $k < $zmax ; $k = $k + $dz ) {
        zstrich3d( $layer, $k, $farbe );
    }
    for ( $k = ( 0 - $dz ) ; $k > $zmin ; $k = $k - $dz ) {
        zstrich3d( $layer, $k, $farbe );
    }
}
}
elseif ( $xskala eq "nureins" ) {
    if ( $xmin != $xmax ) {
        xmarkierung3d( $layer, 1, 1, $farbe );
    }
    if ( $ymin != $ymax ) {
        ymarkierung3d( $layer, 1, 1, $farbe );
    }
    if ( $zmin != $zmax ) {
        zmarkierung3d( $layer, 1, 1, $farbe );
    }
}
elseif ( $xskala eq "strichundeins" ) {
    for ( $i = ( 0 + $dx ) ; $i < $xmax ; $i = $i + $dx ) {
        if ( $i == 1 ) { xmarkierung3d( $layer, 1, 1, $farbe ); }
        else { xstrich3d( $layer, $i, $farbe ); }
    }
    for ( $i = ( 0 - $dx ) ; $i > $xmin ; $i = $i - $dx ) {
        xstrich3d( $layer, $i, $farbe );
    }
    for ( $j = ( 0 + $dy ) ; $j < $ymax ; $j = $j + $dy ) {
        if ( $j == 1 ) { ymarkierung3d( $layer, 1, 1, $farbe ); }
        else { ystrich3d( $layer, $j, $farbe ); }
    }
    for ( $j = ( 0 - $dy ) ; $j > $ymin ; $j = $j - $dy ) {
        ystrich3d( $layer, $j, $farbe );
    }
    for ( $k = ( 0 + $dz ) ; $k < $zmax ; $k = $k + $dz ) {
        if ( $k == 1 ) { zmarkierung3d( $layer, 1, 1, $farbe ); }
        else { zstrich3d( $layer, $k, $farbe ); }
    }
    for ( $k = ( 0 - $dz ) ; $k > $zmin ; $k = $k - $dz ) {
        zstrich3d( $layer, $k, $farbe );
    }
}
}
else { $parameterweiter = $parameterweiter; }
}

befehlerzeugen( $layer, "pstThreeDCoor", $farbe, $linienart,
    $linienstaerke, $parameterweiter, "", "" );
}
else {
    if ( $xmin != $xmax ) {
        if ( $xmax == 0 ) {
            linie( $layer, $xmin, 0, 0, $xmax, 0, 0, $linienart,
                $linienstaerke, $farbe, "", $pfeilart );
        }
    }
    else {
        linie( $layer, $xmin, 0, 0, $xmax, 0, 0, $linienart,
            $linienstaerke, $farbe, "", $pfeilart );
    }
}

```

```

    }
  }
  if ( $ymin != $ymax ) {
    if ( $ymax == 0 ) {
      linie( $layer, 0, $ymin, 0, 0, $ymax, 0, $linienart,
        $linienstaerke, $farbe, "", $pfeilart );
    }
    else {
      linie( $layer, 0, $ymin, 0, 0, $ymax, 0, $linienart,
        $linienstaerke, $farbe, "", $pfeilart );
    }
  }
  if ( $zmin != $zmax ) {
    if ( $zmax == 0 ) {
      linie( $layer, 0, 0, $zmin, 0, 0, $zmax, $linienart,
        $linienstaerke, $farbe, "", $pfeilart );
    }
    else {
      linie( $layer, 0, 0, $zmin, 0, 0, $zmax, $linienart,
        $linienstaerke, $farbe, "", $pfeilart );
    }
  }
}

if ( $xskala eq "normal" ) {
  if ( $xmin != $xmax ) {
    for ( $i = ( 0 + $dx ) ; $i < $xmax ; $i = $i + $dx ) {
      xmarkierung3d( $layer, $i, $i, $farbe );
    }
    for ( $i = ( 0 - $dx ) ; $i > $xmin ; $i = $i - $dx ) {
      xmarkierung3d( $layer, $i, $i, $farbe );
    }
  }
  if ( $ymin != $ymax ) {
    for ( $j = ( 0 + $dy ) ; $j < $ymax ; $j = $j + $dy ) {
      ymarkierung3d( $layer, $j, $j, $farbe );
    }
    for ( $j = ( 0 - $dy ) ; $j > $ymin ; $j = $j - $dy ) {
      ymarkierung3d( $layer, $j, $j, $farbe );
    }
  }
  if ( $zmin != $zmax ) {
    for ( $k = ( 0 + $dz ) ; $k < $zmax ; $k = $k + $dz ) {
      zmarkierung3d( $layer, $k, $k, $farbe );
    }
    for ( $k = ( 0 - $dz ) ; $k > $zmin ; $k = $k - $dz ) {
      zmarkierung3d( $layer, $k, $k, $farbe );
    }
  }
}

elseif ( $xskala eq "nurstriche" ) {
  if ( $xmin != $xmax ) {
    for ( $i = ( 0 + $dx ) ; $i < $xmax ; $i = $i + $dx ) {
      xstrich3d( $layer, $i, $farbe );
    }
    for ( $i = ( 0 - $dx ) ; $i > $xmin ; $i = $i - $dx ) {
      xstrich3d( $layer, $i, $farbe );
    }
  }
  if ( $ymin != $ymax ) {
    for ( $j = ( 0 + $dy ) ; $j < $ymax ; $j = $j + $dy ) {
      ystrich3d( $layer, $j, $farbe );
    }
  }
}

```

```

    }
    for ( $j = ( 0 - $dy ) ; $j > $ymin ; $j = $j - $dy ) {
        ystrich3d( $layer, $j, $farbe );
    }
}
if ( $zmin != $zmax ) {
    for ( $k = ( 0 + $dz ) ; $k < $zmax ; $k = $k + $dz ) {
        zstrich3d( $layer, $k, $farbe );
    }
    for ( $k = ( 0 - $dz ) ; $k > $zmin ; $k = $k - $dz ) {
        zstrich3d( $layer, $k, $farbe );
    }
}
}
elseif ( $xskala eq "nureins" ) {
    if ( $xmin != $xmax ) {
        xmarkierung3d( $layer, 1, 1, $farbe );
    }
    if ( $ymin != $ymax ) {
        ymarkierung3d( $layer, 1, 1, $farbe );
    }
    if ( $zmin != $zmax ) {
        zmarkierung3d( $layer, 1, 1, $farbe );
    }
}
elseif ( $xskala eq "strichundeins" ) {
    for ( $i = ( 0 + $dx ) ; $i < $xmax ; $i = $i + $dx ) {
        if ( $i == 1 ) { xmarkierung3d( $layer, 1, 1, $farbe ); }
        else { xstrich3d( $layer, $i, $farbe ); }
    }
    for ( $i = ( 0 - $dx ) ; $i > $xmin ; $i = $i - $dx ) {
        xstrich3d( $layer, $i, $farbe );
    }
    for ( $j = ( 0 + $dy ) ; $j < $ymax ; $j = $j + $dy ) {
        if ( $j == 1 ) { ymarkierung3d( $layer, 1, 1, $farbe ); }
        else { ystrich3d( $layer, $j, $farbe ); }
    }
    for ( $j = ( 0 - $dy ) ; $j > $ymin ; $j = $j - $dy ) {
        ystrich3d( $layer, $j, $farbe );
    }
    for ( $k = ( 0 + $dz ) ; $k < $zmax ; $k = $k + $dz ) {
        if ( $k == 1 ) { zmarkierung3d( $layer, 1, 1, $farbe ); }
        else { zstrich3d( $layer, $k, $farbe ); }
    }
    for ( $k = ( 0 - $dz ) ; $k > $zmin ; $k = $k - $dz ) {
        zstrich3d( $layer, $k, $farbe );
    }
}
else { $parameterweiter = $parameterweiter; }

if ( $xmin != $xmax ) {
    xmarkierung3dohne( $layer, $xmax, $xtext, $farbe );
}
if ( $ymin != $ymax ) {
    ymarkierung3dohne( $layer, $ymax, $ytext, $farbe );
}
if ( $zmin != $zmax ) {
    zmarkierung3dohne( $layer, $zmax, $ztext, $farbe );
}

```

}	1278
abstand(\$layer);	1279
	1280
setknoten(\$knotenersatz);	1281
	1282
}	1283
	1284
#####	1285
# Markierungen an der Achse	1286
#####	1287
	1288
sub xmarkierung3d {	1289
my (\$layer, \$x, \$text, \$farbe) = @_;	1290
	1291
if (istzahl(\$x) == 0) { set3dpstricks(1); }	1292
set3dzeichnen(1);	1293
	1294
xmarkierung3dohndlinie(\$layer,\$x,\$text, \$farbe);	1295
xstrich3d(\$layer, \$x, \$farbe);	1296
}	1297
	1298
sub xmarkierung3dohndlinie {	1299
my (\$layer, \$x, \$text, \$farbe) = @_;	1300
my (\$y, \$z, \$ausrichtung);	1301
	1302
\$y = \$markierungx;	1303
\$z = 0;	1304
set3dzeichnen(1);	1305
beschriftung3d(\$layer, \$farbe, \$x, \$y, \$z, 0, 180, \$text);	1306
	1307
}	1308
	1309
sub ymarkierung3d {	1310
my (\$layer, \$y, \$text, \$farbe) = @_;	1311
my (\$x, \$z);	1312
	1313
if (istzahl(\$y) == 0) { set3dpstricks(1); }	1314
set3dzeichnen(1);	1315
ymarkierung3dohndlinie(\$layer,\$y,\$text, \$farbe);	1316
ystrich3d(\$layer, \$y, \$farbe);	1317
}	1318
	1319
sub ymarkierung3dohndlinie {	1320
my (\$layer, \$y, \$text, \$farbe) = @_;	1321
my (\$x, \$z);	1322
\$x = \$markierungy;	1323
\$z = 0;	1324
set3dzeichnen(1);	1325
beschriftung3d(\$layer, \$farbe, \$x, \$y, \$z, 0, 270, \$text);	1326
	1327
}	1328
	1329
sub zmarkierung3d {	1330
my (\$layer, \$z, \$text, \$farbe) = @_;	1331
	1332
if (istzahl(\$z) == 0) { set3dpstricks(1); }	1333
set3dzeichnen(1);	1334
zmarkierung3dohndlinie(\$layer,\$z,\$text, \$farbe);	1335
zstrich3d(\$layer, \$z, \$farbe);	1336
}	1337
	1338
sub zmarkierung3dohndlinie {	

my (\$layer, \$z, \$text, \$farbe) = @_;	1339
my (\$x, \$y);	1340
	1341
\$x = 0;	1342
\$y = \$markierungz;	1343
	1344
set3dzeichnen(1);	1345
beschriftung3d(\$layer, \$farbe, \$x, \$y, \$z, 0, 180, \$text);	1346
}	1347
	1348
sub xstrich3d {	1349
my (\$layer, \$x, \$farbe) = @_;	1350
my (\$y, \$z);	1351
\$y = \$strichx;	1352
\$z = 0;	1353
set3dzeichnen(1);	1354
linie (\$layer, \$x, \$y, \$z, \$x, -\$y, \$z, "solid", \$markierungsdicke, \$farbe,	1355
"", "");	1356
}	1357
	1358
sub ystrich3d {	1359
my (\$layer, \$y, \$farbe) = @_;	1360
my (\$x, \$z);	1361
\$x = \$strichy;	1362
\$z = 0;	1363
set3dzeichnen(1);	1364
	1365
linie (\$layer, \$x, \$y, \$z, -\$x, \$y, \$z, "solid", \$markierungsdicke, \$farbe,	1366
"", "");	1367
}	1368
	1369
sub zstrich3d {	1370
my (\$layer, \$z, \$farbe) = @_;	1371
my (\$x, \$y);	1372
\$x = 0;	1373
\$y = \$strichz;	1374
set3dzeichnen(1);	1375
linie (\$layer, \$x, \$y, \$z, \$x, -\$y, \$z, "solid", \$markierungsdicke, \$farbe,	1376
"", "");	1377
}	1378
	1379
sub beschriftung3d {	1380
my (\$layer, \$farbe, \$x, \$y, \$z, \$winkel, \$ausrichtung, \$text) = @_;	1381
	1382
if (istzahl(\$x) == 1) { \$x = nachkommastellen(\$x, 4); }	1383
else { set3dpstricks(1); }	1384
if (istzahl(\$y) == 1) { \$y = nachkommastellen(\$y, 4); }	1385
else { set3dpstricks(1); }	1386
if (istzahl(\$z) == 1) { \$z = nachkommastellen(\$z, 4); }	1387
else { set3dpstricks(1); }	1388
\$text =~ tr/./,/s;	1389
	1390
if (get3dpstricks() == 1) {	1391
	1392
if (\$winkel == 0) {	1393
	1394
settexthier("");	1395
if ((istzahl(\$x) == 1)	1396
&& (istzahl(\$y) == 1)	1397
&& (istzahl(\$z) == 1))	1398
{	1399

```

    tripel ( $x, $y, $z );
}
else { tripelohneminmax( $x, $y, $z ); }
klammer( textfarbe( $text, $farbe ) );
befehlerzeugen( $layer, "pstThreeDPut", "", "", "", "", "",
    gettexthier ( ) );
settexthier ("" );
}
else {
    settexthier ("" );
    if ( ( istzahl($x) == 1 )
        && ( istzahl($y) == 1 )
        && ( istzahl($z) == 1 ) )
    {
        tripel ( $x, $y, $z );
    }
    else { tripelohneminmax( $x, $y, $z ); }

    klammer("\uput{0}["
        . $ausrichtung . "]{ "
        . $winkel
        . " }(0,0) {"
        . textfarbe( $text, $farbe )
        . " }" );
    befehlerezeugen( $layer, "pstThreeDPut", "", "", "", "", "",
        gettexthier ( ) );
    settexthier ("" );
}

}
else {
    beschriftung(
        $layer, $farbe,
        x3d( $x, $y, $z ),
        y3d( $x, $y, $z ),
        $winkel, $ausrichtung, $text
    );
}

if ( ( istzahl($x) == 1 ) && ( istzahl($y) == 1 ) && ( istzahl($z) == 1 ) )
{
    my $laenge = length($text);
    minmax3d(
        $x - ($laenge),
        $y - ($laenge),
        $z - ($laenge),
        $x + ($laenge),
        $y + ($laenge),
        $z + ($laenge)
    );
}

}

#####
# Geradenfunktion
#####

sub gerade {
    my ( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $r1, $r2, $farbe, $linienart,
        $linienstaerke, $parameter, $pfeilart )
        = @_ ;

```

```

if ( ( $z1 ne "" ) && ( $z2 ne "" ) ) {
    kommentar( $layer,
        "Die Gerade g: ("
        . $x1 . " | "
        . $y1 . " | "
        . $z1
        . ") + r("
        . $x2 . " | "
        . $y2 . " | "
        . $z2
        . ") " );
    linie (
        $layer,          $x1 + $r1 * $x2, $y1 + $r1 * $y2,
        $z1 + $r1 * $z2, $x1 + $r2 * $x2, $y1 + $r2 * $y2,
        $z1 + $r2 * $z2, $linienart,      $linienstaerke,
        $farbe,          $parameter,      $pfeilart
    );
}
else {
    kommentar( $layer,
        "Die Gerade g: ("
        . $x1 . " | "
        . $y1
        . ") + r("
        . $x2 . " | "
        . $y2
        . ") " );

    linie (
        $layer,          $x1 + $r1 * $x2,
        $y1 + $r1 * $y2, "",
        $x1 + $r2 * $x2, $y1 + $r2 * $y2,
        "",              $linienart,
        $linienstaerke,  $farbe,
        $parameter,      $pfeilart
    );
}
}

sub ebene {
    my (
        $layer,      $xauf,      $yauf,      $zauf,
        $xrichtung1, $yrichtung1, $zrichtung1, $xrichtung2,
        $yrichtung2, $zrichtung2, $r1,      $r2,
        $r3,          $r4,          $farbe,      $fuellart ,
        $fuellfarbe, $linienart,  $linienstaerke, $parameter
    ) = @_ ;

    my ( $x1, $y1, $z1, $x2, $y2, $z2, $x3, $y3, $z3, $x4, $y4, $z4 );

    $x1 = $xauf + $r3 * $xrichtung2 + $r1 * $xrichtung1;
    $y1 = $yauf + $r3 * $yrichtung2 + $r1 * $yrichtung1;
    $z1 = $zauf + $r3 * $zrichtung2 + $r1 * $zrichtung1;

    $x2 = $xauf + $r3 * $xrichtung2 + $r2 * $xrichtung1;
    $y2 = $yauf + $r3 * $yrichtung2 + $r2 * $yrichtung1;
    $z2 = $zauf + $r3 * $zrichtung2 + $r2 * $zrichtung1;

```

$\$x3 = \$x1 + (\$r4 - \$r3) * \$xrichtung2;$	1522
$\$y3 = \$y1 + (\$r4 - \$r3) * \$yrichtung2;$	1523
$\$z3 = \$z1 + (\$r4 - \$r3) * \$zrichtung2;$	1524
	1525
	1526
$\$x4 = \$x2 + (\$r4 - \$r3) * \$xrichtung2;$	1527
$\$y4 = \$y2 + (\$r4 - \$r3) * \$yrichtung2;$	1528
$\$z4 = \$z2 + (\$r4 - \$r3) * \$zrichtung2;$	1529
	1530
kommentar (\$layer,	1531
"Die Ebene E: ("	1532
. \$xauf . " "	1533
. \$yauf . " "	1534
. \$zauf	1535
. ") + r("	1536
. \$xrichtung1 . " "	1537
. \$yrichtung1 . " "	1538
. \$zrichtung1	1539
. ") + s("	1540
. \$xrichtung2 . " "	1541
. \$yrichtung2 . " "	1542
. \$zrichtung2	1543
. ")");	1544
	1545
viereck (\$layer, \$x1, \$y1, \$z1, \$x2, \$y2, \$z2, \$x4, \$y4, \$z4, \$x3, \$y3, \$z3,	1546
\$farbe, \$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke,	1547
\$parameter);	1548
	1549
setachsengraph3d (\$x1, \$y1, \$z1, \$x1, \$y1, \$z1);	1550
setachsengraph3d (\$x2, \$y2, \$z2, \$x2, \$y2, \$z2);	1551
setachsengraph3d (\$x3, \$y3, \$z3, \$x3, \$y3, \$z3);	1552
setachsengraph3d (\$x4, \$y4, \$z4, \$x4, \$y4, \$z4);	1553
	1554
}	1555
	1556
sub ebenenormal {	1557
my (\$layer, \$x1, \$y1, \$z1, \$xn, \$yn, \$zn, \$r1, \$r2, \$r3, \$r4, \$farbe,	1558
\$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke, \$parameter)	1559
= @_;	1560
	1561
kommentar (\$layer, "hier");	1562
	1563
my (\$x2, \$y2, \$z2, \$x3, \$y3, \$z3);	1564
	1565
if (\$zn == 0) {	1566
\$z2 = 1;	1567
\$x2 = 0;	1568
\$y2 = 0;	1569
\$z3 = 0;	1570
\$x3 = -\$y2;	1571
\$y3 = \$x3;	1572
}	1573
if (\$zn == 0) {	1574
\$z2 = 1;	1575
\$x2 = 0;	1576
\$y2 = 0;	1577
\$z3 = 0;	1578
\$x3 = -\$y2;	1579
\$y3 = \$x3;	1580
}	1581
elsif ((\$zn == 0) && (\$yn == 0)) {	1582

```

    $x2 = $y2 = $z3 = 0;
    $z2 = 1;
    $x3 = -1;
    $y3 = 2;
}
else {
    $x2 = $yn;
    $y2 = -$xn;
    $z2 = 0;
    $x3 = 0;
    $y3 = $zn;
    $z3 = -$yn;
}

ebene(
    $layer,      $x1,      $y1,      $z1,
    $x2,          $y2,      $z2,      $x3,
    $y3,          $z3,      $r1,      $r2,
    $r3,          $r4,      $farbe,    $fuellart ,
    $fuellfarbe , $linienart , $linienstaerke , $parameter
);

}

sub ebenekform {
    my ( $layer, $a, $b, $c, $d, $n1, $n2, $farbe, $fuellart , $fuellfarbe ,
        $linienart , $linienstaerke , $parameter )
        = @_;

    my ( $x1, $x2, $x3, $x4, $y1, $y2, $y3, $y4, $z1, $z2, $z3, $z4 );

    if ( ( $a == 0 ) && ( $b != 0 ) && ( $c != 0 ) ) {
        $x1 = 0;
        $y1 = $d / $b;
        $z1 = 0;

        $x2 = 0;
        $y2 = 0;
        $z2 = $d / $c;

        $x3 = $n1;
        $y3 = 0;
        $z3 = $d / $c;

        $x4 = $n1;
        $y4 = $d / $b;
        $z4 = 0;
        viereck(
            $layer,      $x1,          $y1,      $z1,
            $x2,          $y2,          $z2,      $x3,
            $y3,          $z3,          $x4,      $y4,
            $z4,          $farbe,        $fuellart , $fuellfarbe ,
            $linienart , $linienstaerke , $parameter
        );
    }
    elsif ( ( $a != 0 ) && ( $b == 0 ) && ( $c != 0 ) ) {
        $x1 = $d / $a;
        $y1 = 0;
        $z1 = 0;
        $x2 = $d / $a;

```

1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643

```

$y2 = $n1;
$z2 = 0;
$x3 = 0;
$y3 = $n1;
$z3 = $d / $c;
$x4 = 0;
$y4 = 0;
$z4 = $d / $c;
viereck(
    $layer,    $x1,        $y1,    $z1,
    $x2,        $y2,        $z2,    $x3,
    $y3,        $z3,        $x4,    $y4,
    $z4,        $farbe,    $fuellart, $fuellfarbe,
    $linienart, $linienstaerke, $parameter
);
}
elseif ( ( $a != 0 ) && ( $b != 0 ) && ( $c == 0 ) ) {
    $x1 = $d / $a;
    $y1 = 0;
    $z1 = 0;
    $x2 = 0;
    $y2 = $d / $b;
    $z2 = 0;
    $x3 = 0;
    $y3 = $d / $b;
    $z3 = $n1;
    $x4 = $d / $a;
    $y4 = 0;
    $z4 = $n1;
    viereck(
        $layer,    $x1,        $y1,    $z1,
        $x2,        $y2,        $z2,    $x3,
        $y3,        $z3,        $x4,    $y4,
        $z4,        $farbe,    $fuellart, $fuellfarbe,
        $linienart, $linienstaerke, $parameter
    );
}
elseif ( ( $a != 0 ) && ( $b == 0 ) && ( $c == 0 ) ) {
    $x1 = $d / $a;
    $y1 = $n1;
    $z1 = 0;

    $x2 = $d / $a;
    $y2 = 0;
    $z2 = 0;

    $x3 = $d / $a;
    $y3 = 0;
    $z3 = $n2;
    $x4 = $d / $a;
    $y4 = $n1;
    $z4 = $n2;
    viereck(
        $layer,    $x1,        $y1,    $z1,
        $x2,        $y2,        $z2,    $x3,
        $y3,        $z3,        $x4,    $y4,
        $z4,        $farbe,    $fuellart, $fuellfarbe,
        $linienart, $linienstaerke, $parameter
    );
}
elseif ( ( $a == 0 ) && ( $b != 0 ) && ( $c == 0 ) ) {

```

1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704

```

    $x1 = $n1;
    $y1 = ( $d / $b );
    $z1 = 0;

    $x2 = 0;
    $y2 = ( $d / $b );
    $z2 = 0;

    $x3 = 0;
    $y3 = ( $d / $b );
    $z3 = $n2;

    $x4 = $n1;
    $y4 = ( $d / $b );
    $z4 = $n2;
    viereck(
        $layer,      $x1,          $y1,      $z1,
        $x2,          $y2,          $z2,      $x3,
        $y3,          $z3,          $x4,      $y4,
        $z4,          $farbe,        $fuellart, $fuellfarbe,
        $linienart, $linienstaerke, $parameter
    );
}

elseif ( ( $a == 0 ) && ( $b == 0 ) && ( $c != 0 ) ) {

    $x1 = $n1;
    $y1 = 0;
    $z1 = $d / $c;
    $x2 = 0;
    $y2 = 0;
    $z2 = $d / $c;
    $x3 = 0;
    $y3 = $n2;
    $z3 = $d / $c;
    $x4 = $n1;
    $y4 = $n2;
    $z4 = $d / $c;
    viereck(
        $layer,      $x1,          $y1,      $z1,
        $x2,          $y2,          $z2,      $x3,
        $y3,          $z3,          $x4,      $y4,
        $z4,          $farbe,        $fuellart, $fuellfarbe,
        $linienart, $linienstaerke, $parameter
    );
}
else {
    $x1 = $d / $a;
    $y1 = 0;
    $z1 = 0;
    $x2 = 0;
    $y2 = $d / $b;
    $z2 = 0;
    $x3 = 0;
    $y3 = 0;
    $z3 = $d / $c;
    dreieck( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $x3, $y3, $z3, $farbe,
        $fuellart, $fuellfarbe, $linienart, $linienstaerke, $parameter );
}

```

```

1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765

```

```

}
sub quader {
  my ( $layer, $x, $y, $z, $a, $b, $c, $w1, $w2, $farbe, $fuellart,
    $fuellfarbe, $linienart, $linienstaerke, $parameter )
    = @_;

  my $liniehinten;

  if ( $fuellart ne "" ) { $liniehinten = "solid"; }
  else { $liniehinten = "dashed"; }

  settexthier("");

  minmax3d( $x, $y, $z, $x, $y, $z );
  minmax3d( $x + $a, $y, $z, $x + $a, $y, $z );
  minmax3d( $x + $a, $y + $b, $z, $x + $a, $y + $b, $z );
  minmax3d( $x + $a, $y + $b, $z + $c, $x + $a, $y + $b, $z + $c );

  if ( ( get3dpstricks() == 1 ) && ( $fuellart ne "" ) ) {
    tripel( $x, $y, $z );
    tripel( 0, $b, 0 );
    tripel( $a, 0, 0 );
    tripel( 0, 0, $c );

    befehlfuellungerzeugen(
      $layer, "pstThreeDBox", $farbe, $fuellart,
      $fuellfarbe, $linienart, $linienstaerke, $parameter,
      "", gettexthier()
    );
  }
  else {
    if ( $fuellart ne "" ) {
      viereck(
        $layer, $x + $a, $y, $z,
        $x + $a, $y, $z + $c, $x,
        $y, $z + $c, $x, $y,
        $z, $fuellfarbe, $fuellart, $fuellfarbe,
        $liniehinten, $linienstaerke, $parameter
      );

      viereck(
        $layer, $x, $y, $z,
        $x, $y + $b, $z, $x,
        $y + $b, $z + $c, $x, $y,
        $z + $c, $fuellfarbe, $fuellart, $fuellfarbe,
        $linienart, $linienstaerke, $parameter
      );

      viereck(
        $layer, $x + $a, $y + $b, $z,
        $x + $a, $y, $z, $x,
        $y, $z, $x, $y + $b,
        $z, $fuellfarbe, $fuellart, $fuellfarbe,
        $linienart, $linienstaerke, $parameter
      );
    }

    settexthier("");
    tripel( $x, $y + $b, $z );
  }
}

```

```

tripel ( $x,      $y,      $z );
tripel ( $x + $a, $y,      $z );
liniekopf( $layer, "dashed", $linienstaerke, $farbe, $parameter, "",
  gettexthier ( ) );
settexthier ("" );
tripel ( $x, $y, $z );
tripel ( $x, $y, $z + $c );
liniekopf( $layer, "dashed", $linienstaerke, $farbe, $parameter, "",
  gettexthier ( ) );
settexthier ("" );

viereck(
  $layer,      $x + $a,      $y + $b,  $z,
  $x + $a,      $y + $b,      $z + $c,  $x,
  $y + $b,      $z + $c,      $x,      $y + $b,
  $z,          $farbe,      $fuellart, $fuellfarbe,
  $linienart, $linienstaerke, $parameter
);

viereck(
  $layer,      $x + $a,      $y,      $z,
  $x + $a,      $y + $b,      $z,      $x + $a,
  $y + $b,      $z + $c,      $x + $a,  $y,
  $z + $c,      $farbe,      $fuellart, $fuellfarbe,
  $linienart, $linienstaerke, $parameter
);

viereck(
  $layer,      $x + $a,      $y + $b,  $z + $c,
  $x + $a,      $y,          $z + $c,  $x,
  $y,          $z + $c,      $x,      $y + $b,
  $z + $c,      $farbe,      $fuellart, $fuellfarbe,
  $linienart, $linienstaerke, $parameter
);
}
settexthier ("" );

if ( $punkt == 1 ) {
  if ( $beschriftung == 0 ) {
    punktzeichnen( $layer - 1, $x + $a, $y, $z, 0.05, 225, $farbe,
      "solid", $farbe, $linienart, $linienstaerke, "", "" );
    punktzeichnen( $layer - 1, $x + $a, $y + $b, $z, 0.05, 315, $farbe,
      "solid", $farbe, $linienart, $linienstaerke, "", "" );
    punktzeichnen( $layer - 1, $x, $y + $b, $z, 0.05, 45, $farbe,
      "solid", $farbe, $linienart, $linienstaerke, "", "" );
    punktzeichnen( $layer - 1, $x, $y, $z, 0.05, 135, $farbe, "solid",
      $farbe, $linienart, $linienstaerke, "", "" );
    punktzeichnen( $layer - 1, $x + $a, $y, $z + $c, 0.05, 225, $farbe,
      "solid", $farbe, $linienart, $linienstaerke, "", "" );
    punktzeichnen(
      $layer - 1, $x + $a,      $y + $b,      $z + $c,
      0.05,      315,      $farbe,      "solid",
      $farbe,      $linienart, $linienstaerke, "",
      ""
    );
  };
  punktzeichnen( $layer - 1, $x, $y + $b, $z + $c, 0.05, 45, $farbe,
    "solid", $farbe, $linienart, $linienstaerke, "", "" );
  punktzeichnen( $layer - 1, $x, $y, $z + $c, 0.05, 135, $farbe,
    "solid", $farbe, $linienart, $linienstaerke, "", "" );
}

```

}	1888
	1889
if (\$beschriftung == 1) {	1890
	1891
punktzeichnen(\$layer - 1, \$x + \$a, \$y, \$z, 0.05, 225, \$farbe, "solid",	1892
\$farbe, \$linienart, \$linienstaerke, "\\$A\$", "");	1893
punktzeichnen(	1894
\$layer - 1, \$x + \$a, \$y + \$b, \$z,	1895
0.05, 315, \$farbe, "solid",	1896
\$farbe, \$linienart, \$linienstaerke, "\\$B\$",	1897
""	1898
);	1899
punktzeichnen(\$layer - 1, \$x, \$y + \$b, \$z, 0.05, 45, \$farbe, "solid",	1900
\$farbe, \$linienart, \$linienstaerke, "\\$C\$", "");	1901
punktzeichnen(\$layer - 1, \$x, \$y, \$z, 0.05, 135, \$farbe, "solid",	1902
\$farbe, \$linienart, \$linienstaerke, "\\$D\$", "");	1903
punktzeichnen(	1904
\$layer - 1, \$x + \$a, \$y, \$z + \$c,	1905
0.05, 225, \$farbe, "solid",	1906
\$farbe, \$linienart, \$linienstaerke, "\\$E\$",	1907
""	1908
);	1909
punktzeichnen(	1910
\$layer - 1, \$x + \$a, \$y + \$b, \$z + \$c,	1911
0.05, 315, \$farbe, "solid",	1912
\$farbe, \$linienart, \$linienstaerke, "\\$F\$",	1913
""	1914
);	1915
punktzeichnen(	1916
\$layer - 1, \$x, \$y + \$b, \$z + \$c,	1917
0.05, 45, \$farbe, "solid",	1918
\$farbe, \$linienart, \$linienstaerke, "\\$G\$",	1919
""	1920
);	1921
punktzeichnen(\$layer - 1, \$x, \$y, \$z + \$c, 0.05, 135, \$farbe, "solid",	1922
\$farbe, \$linienart, \$linienstaerke, "\\$H\$", "");	1923
	1924
}	1925
	1926
}	1927
	1928
sub kugel {	1929
my (	1930
\$layer, \$x, \$y, \$z,	1931
\$r, \$farbe, \$fuellart, \$fuellfarbe,	1932
\$linienart, \$linienstaerke, \$parameter	1933
) = @_;	1934
my \$i;	1935
	1936
settexthier ("");	1937
set3dpstricks(1);	1938
tripel (\$x, \$y, \$z);	1939
klammer(\$r);	1940
befehlfuellungerzeugen(	1941
\$layer, "pstThreeDSphere", \$farbe, \$fuellart,	1942
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter,	1943
"", gettexthier ()	1944
);	1945
minmax3d(\$x - \$r, \$y, \$z, \$x - \$r, \$y, \$z);	1946
minmax3d(\$x, \$y - \$r, \$z, \$x, \$y - \$r, \$z);	1947
	1948

minmax3d(\$x, \$y, \$z - \$r, \$x, \$y, \$z - \$r);	1949
minmax3d(\$x + \$r, \$y, \$z, \$x + \$r, \$y, \$z);	1950
minmax3d(\$x, \$y + \$r, \$z, \$x, \$y + \$r, \$z);	1951
minmax3d(\$x, \$y, \$z + \$r, \$x, \$y, \$z + \$r);	1952
settexthier ("");	1953
}	1954
	1955
	1956
	1957
sub kreis3d {	1958
my (	1959
\$lager, \$x, \$y, \$z,	1960
\$u1, \$u2, \$u3, \$startwinkel,	1961
\$endwinkel, \$farbe, \$fuellart, \$fuellfarbe,	1962
\$linienart, \$linienstaerke, \$parameter	1963
) = @_;	1964
	1965
ellipse3d(	1966
\$lager, \$x, \$y, \$z,	1967
\$u1, \$u2, \$u3, \$u1,	1968
-\$u2, -\$u3, \$startwinkel, \$endwinkel,	1969
\$farbe, \$fuellart, \$fuellfarbe, \$linienart,	1970
\$linienstaerke, \$parameter	1971
);	1972
}	1973
	1974
	1975
sub ellipse3d {	1976
my (	1977
\$lager, \$x, \$y, \$z, \$u1, \$u2, \$u3, \$v1, \$v2, \$v3, \$startwinkel,	1978
\$endwinkel,	1979
\$farbe,	1980
\$fuellart,	1981
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter	1982
) = @_;	1983
settexthier ("");	1984
	1985
	1986
my \$i = 0;	1987
for (\$i = \$startwinkel ; \$i <= \$endwinkel ; \$i++) {	1988
setachsengraph3d(	1989
\$x + cos (degtorad(\$i)) * \$u1 + sin (degtorad(\$i)) * \$v1,	1990
\$y + cos (degtorad(\$i)) * \$u2 + sin (degtorad(\$i)) * \$v2,	1991
\$z + cos (degtorad(\$i)) * \$u3 + sin (degtorad(\$i)) * \$v3,	1992
\$x + cos (degtorad(\$i)) * \$u1 + sin (degtorad(\$i)) * \$v1,	1993
\$y + cos (degtorad(\$i)) * \$u2 + sin (degtorad(\$i)) * \$v2,	1994
\$z + cos (degtorad(\$i)) * \$u3 + sin (degtorad(\$i)) * \$v3	1995
);	1996
grenzen3d(	1997
\$x + cos (degtorad(\$i)) * \$u1 + sin (degtorad(\$i)) * \$v1,	1998
\$y + cos (degtorad(\$i)) * \$u2 + sin (degtorad(\$i)) * \$v2,	1999
\$z + cos (degtorad(\$i)) * \$u3 + sin (degtorad(\$i)) * \$v3	2000
);	2001
}	2002
	2003
if (get3dpstricks() == 1) {	2004
tripel (\$x, \$y, \$z);	2005
tripel (\$u1, \$u2, \$u3);	2006
tripel (\$v1, \$v2, \$v3);	2007
	2008
pstricks(\$lager,	2009

"\psset{beginAngle= "	2010
. \$startwinkel	2011
. ",endAngle= "	2012
. \$endwinkel	2013
. "}");	2014
befehlfuellungerzeugen(	2015
\$layer, "pstThreeDEllipse",	2016
\$farbe, \$fuellart ,	2017
\$fuellfarbe, \$linienart ,	2018
\$linienstaerke, \$parameter,	2019
"", gettexthier ()	2020
);	2021
}	2022
else {	2023
for (\$i = \$startwinkel ; \$i <= \$endwinkel ; \$i = \$i + 1) {	2024
tripel (	2025
\$x + cos(degtorad(\$i)) * \$u1 + sin(degtorad(\$i)) * \$v1,	2026
\$y + cos(degtorad(\$i)) * \$u2 + sin(degtorad(\$i)) * \$v2,	2027
\$z + cos(degtorad(\$i)) * \$u3 + sin(degtorad(\$i)) * \$v3	2028
);	2029
}	2030
polygon(\$layer, \$linienart , \$linienstaerke, \$farbe, \$fuellart ,	2031
\$fuellfarbe, \$parameter . " , plotstyle=curve",	2032
gettexthier ());	2033
}	2034
settexthier ("");	2035
}	2036
}	2037
}	2038
}	2039
sub rechteck3d {	2040
my (\$layer, \$x, \$y, \$z, \$r1, \$r2, \$r3, \$r4, \$r5, \$r6, \$farbe, \$fuellart ,	2041
\$fuellfarbe, \$linienart , \$linienstaerke, \$parameter)	2042
= @_;	2043
}	2044
settexthier ("");	2045
}	2046
if (get3dpstricks() == 1) {	2047
tripel (\$x, \$y, \$z);	2048
tripel (\$r1, \$r2, \$r3);	2049
tripel (\$r4, \$r5, \$r6);	2050
befehlfuellungerzeugen(	2051
\$layer, "pstThreeDSquare",	2052
\$farbe, \$fuellart ,	2053
\$fuellfarbe, \$linienart ,	2054
\$linienstaerke, \$parameter,	2055
"", gettexthier ()	2056
);	2057
}	2058
else {	2059
tripel (\$x, \$y, \$z);	2060
tripel (\$x + \$r1, \$y + \$r2, \$z + \$r3);	2061
tripel (\$x + \$r1 + \$r4, \$y + \$r2 + \$r5, \$z + \$r3 + \$r6);	2062
tripel (\$x + \$r4, \$y + \$r5, \$z + \$r6);	2063
tripel (\$x, \$y, \$z);	2064
befehlfuellungerzeugen(	2065
\$layer, "pspolygon", \$farbe, \$fuellart ,	2066
\$fuellfarbe, \$linienart , \$linienstaerke, \$parameter,	2067
"", gettexthier ()	2068
);	2069
}	2070

```

settexthier("");
}

sub pyramide {
  my ( $layer, $x, $y, $z, $a, $b, $c, $w1, $w2, $farbe, $fuellart,
        $fuellfarbe, $linienart, $linienstaerke, $parameter )
    = @_;

  my $n = 0;

  if ( $fuellart eq "" ) {

    linie (
      $layer,      $x - $a, $y + $b,    $z,
      $x - $a,     $y,     $z,         "dashed",
      $linienstaerke, $farbe, $parameter, ""
    );
    linie (
      $layer, $x - $a, $y, $z,
      $x - ( 0.5 * $a + $w1 ),
      $y + 0.5 * $b + $w2,
      $z + $c, "dashed", $linienstaerke, $farbe, $parameter, ""
    );

    linie ( $layer, $x - $a, $y, $z, $x, $y, $z, "dashed", $linienstaerke,
            $farbe, $parameter, "" );

    dreieck(
      $layer,      $x - $a,
      $y + $b,     $z,
      $x,          $y + $b,
      $z,          $x - ( 0.5 * $a + $w1 ),
      $y + 0.5 * $b + $w2, $z + $c,
      $farbe,      $fuellart,
      $fuellfarbe, $linienart,
      $linienstaerke, $parameter
    );

    dreieck(
      $layer,      $x,
      $y,          $z,
      $x,          $y + $b,
      $z,          $x - ( 0.5 * $a + $w1 ),
      $y + 0.5 * $b + $w2, $z + $c,
      $farbe,      $fuellart,
      $fuellfarbe, $linienart,
      $linienstaerke, $parameter
    );
  }
  else {

    rechteck3d( $layer, $x, $y, $z, -$a, 0, 0, 0, $b, 0, $farbe, $fuellart,
               $fuellfarbe, $linienart, $linienstaerke, $parameter );

    dreieck(
      $layer,      $x - $a,
      $y + $b,     $z,
      $x - $a,     $y,
      $z,          $x - ( 0.5 * $a + $w1 ),
      $y + 0.5 * $b + $w2, $z + $c,

```

```

2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131

```

\$farbe,	\$fuellart ,	2132
\$fuellfarbe ,	\$linienart ,	2133
\$linienstaerke ,	\$parameter	2134
);		2135
dreieck(		2136
\$layer,	\$x,	2137
\$y,	\$z,	2138
\$x - \$a,	\$y,	2139
\$z,	$\$x - (0.5 * \$a + \$w1)$,	2140
$\$y + 0.5 * \$b + \$w2$,	$\$z + \c ,	2141
\$farbe,	\$fuellart ,	2142
\$fuellfarbe ,	\$linienart ,	2143
\$linienstaerke ,	\$parameter	2144
);		2145
		2146
dreieck(		2147
\$layer,	$\$x - \a ,	2148
$\$y + \b ,	\$z,	2149
\$x,	$\$y + \b ,	2150
\$z,	$\$x - (0.5 * \$a + \$w1)$,	2151
$\$y + 0.5 * \$b + \$w2$,	$\$z + \c ,	2152
\$farbe,	\$fuellart ,	2153
\$fuellfarbe ,	\$linienart ,	2154
\$linienstaerke ,	\$parameter	2155
);		2156
		2157
dreieck(		2158
\$layer,	\$x,	2159
\$y,	\$z,	2160
\$x,	$\$y + \b ,	2161
\$z,	$\$x - (0.5 * \$a + \$w1)$,	2162
$\$y + 0.5 * \$b + \$w2$,	$\$z + \c ,	2163
\$farbe,	\$fuellart ,	2164
\$fuellfarbe ,	\$linienart ,	2165
\$linienstaerke ,	\$parameter	2166
);		2167
		2168
}		2169
}		2170
		2171
sub pyramidestumpf {		2172
my (		2173
\$layer, \$x, \$y, \$z, \$a, \$b, \$c, \$hoehe, \$w1, \$w2, \$farbe,		2174
\$fuellart ,		2175
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter		2176
		2177
) = @_;		2178
		2179
my (\$aneu, \$bneu, \$cneu);		2180
		2181
		2182
\$cneu = \$hoehe;		2183
		2184
\$aneu = nachkommastellen((\$c - \$cneu) / \$c * \$a, 3);		2185
\$bneu = nachkommastellen((\$c - \$cneu) / \$c * \$b, 3);		2186
		2187
if (\$cneu == \$c) {		2188
pyramide(		2189
\$layer, \$x, \$y, \$z, \$a, \$b, \$c, \$w1, \$w2, \$farbe,		2190
\$fuellart ,		2191
\$fuellfarbe , \$linienart , \$linienstaerke , \$parameter		2192

```

    );
}
else {

    if ( $fuellart eq "" ) {

        linie ( $layer, $x, $y, $z, $x - $a, $y, $z,, "dashed",
            $linienstaerke, $farbe, $parameter, "" );

        linie ( $layer, $x - $a, $y + $b, $z, $x - $a, $y, $z,, "dashed",
            $linienstaerke, $farbe, $parameter, "" );

        rechteck3d(
            $layer,                $x - 0.5 * $a + 0.5 * $aneu,
            $y + 0.5 * $b - 0.5 * $bneu, $z + $cneu,
            -$aneu,                0,
            0,                    0,
            $bneu,                0,
            $farbe,                $fuellart ,
            $fuellfarbe,           $linienart ,
            $linienstaerke,        $parameter
        );

        linie (
            $layer,                $x - $a,
            $y,                    $z,
            $x - 0.5 * $a - 0.5 * $aneu, $y + 0.5 * $b - 0.5 * $bneu,
            $z + $cneu,            "dashed",
            $linienstaerke,        $farbe,
            $parameter,            ""
        );

        viereck(
            $layer,                $x - 0.5 * $a - 0.5 * $aneu,
            $y + 0.5 * $b + 0.5 * $bneu, $z + $cneu,
            $x - 0.5 * $a + 0.5 * $aneu, $y + 0.5 * $b + 0.5 * $bneu,
            $z + $cneu,            $x,
            $y + $b,                $z,
            $x - $a,                $y + $b,
            $z,                      $farbe,
            $fuellart ,             $fuellfarbe ,
            $linienart ,            $linienstaerke ,
            $parameter
        );

        viereck(
            $layer,                $x - 0.5 * $a + 0.5 * $aneu,
            $y + 0.5 * $b - 0.5 * $bneu, $z + $cneu,
            $x,                    $y,
            $z,                    $x,
            $y + $b,                $z,
            $x - 0.5 * $a + 0.5 * $aneu, $y + 0.5 * $b + 0.5 * $bneu,
            $z + $cneu,            $farbe,
            $fuellart ,             $fuellfarbe ,
            $linienart ,            $linienstaerke ,
            $parameter
        );

    }
}
else {

```

2193
 2194
 2195
 2196
 2197
 2198
 2199
 2200
 2201
 2202
 2203
 2204
 2205
 2206
 2207
 2208
 2209
 2210
 2211
 2212
 2213
 2214
 2215
 2216
 2217
 2218
 2219
 2220
 2221
 2222
 2223
 2224
 2225
 2226
 2227
 2228
 2229
 2230
 2231
 2232
 2233
 2234
 2235
 2236
 2237
 2238
 2239
 2240
 2241
 2242
 2243
 2244
 2245
 2246
 2247
 2248
 2249
 2250
 2251
 2252
 2253

rechteck3d(\$layer, \$x, \$y, \$z, -\$a, 0, 0, 0, \$b, 0, \$farbe,	2254
\$fuellart , \$fuellfarbe , "dashed", \$linienstaerke, \$parameter);	2255
	2256
	2257
rechteck3d(	2258
\$layer,	2259
\$ $x - 0.5 * a + 0.5 * a_{neu}$,	2260
\$ $y + 0.5 * b - 0.5 * b_{neu}$, \$z + \$cneu,	2261
-\$aneu,	2262
0,	2263
0,	2264
\$bneu,	2265
\$farbe,	2266
\$fuellart ,	2267
\$fuellfarbe ,	2268
"ashed",	2269
\$linienstaerke ,	2270
\$parameter	2271
);	2272
	2273
	2274
viereck(	2275
\$layer,	2276
\$ $x - 0.5 * a - 0.5 * a_{neu}$,	2277
\$ $y + 0.5 * b - 0.5 * b_{neu}$, \$z + \$cneu,	2278
\$ $x - 0.5 * a + 0.5 * a_{neu}$, \$y + 0.5 * \$b - 0.5 * \$bneu,	2279
\$z + \$cneu,	2280
\$ x ,	2281
\$ y ,	2282
\$ $x - a$,	2283
\$ y ,	2284
\$ z ,	2285
\$farbe,	2286
\$fuellart ,	2287
\$fuellfarbe ,	2288
\$linienart ,	2289
\$linienstaerke ,	2290
\$parameter	2291
);	2292
	2293
	2294
viereck(	2295
\$layer,	2296
\$ $x - 0.5 * a - 0.5 * a_{neu}$,	2297
\$ $y + 0.5 * b + 0.5 * b_{neu}$, \$z + \$cneu,	2298
\$ $x - 0.5 * a + 0.5 * a_{neu}$, \$y + 0.5 * \$b + 0.5 * \$bneu,	2299
\$z + \$cneu,	2300
\$ $x - a$,	2301
\$ $y + b$,	2302
\$ z ,	2303
\$farbe,	2304
\$fuellart ,	2305
\$fuellfarbe ,	2306
\$linienart ,	2307
\$linienstaerke ,	2308
\$parameter	2309
);	2310
	2311
	2312
viereck(	2313
\$layer,	2314
\$ $x - 0.5 * a + 0.5 * a_{neu}$,	
\$ $y + 0.5 * b - 0.5 * b_{neu}$, \$z + \$cneu,	
\$ x ,	
\$ y ,	
\$ z ,	
\$ x ,	
\$ $y + b$,	
\$ z ,	
\$ $x - 0.5 * a + 0.5 * a_{neu}$, \$y + 0.5 * \$b + 0.5 * \$bneu,	

```

    $z + $cneu,          $farbe,
    $fuellart ,          $fuellfarbe ,
    $linienart ,         $linienstaerke,
    $parameter
);
2315
2316
2317
2318
2319
2320
viereck(
2321
    $layer,
2322
    $x - 0.5 * $a + 0.5 * $aneu, $y + 0.5 * $b - 0.5 * $bneu,
2323
    $z + $cneu,
2324
    $x - 0.5 * $a + 0.5 * $aneu, $y + 0.5 * $b + 0.5 * $bneu,
2325
    $z + $cneu,
2326
    $x - 0.5 * $a - 0.5 * $aneu, $y + 0.5 * $b + 0.5 * $bneu,
2327
    $z + $cneu,
2328
    $x - 0.5 * $a - 0.5 * $aneu,
2329
    $y + 0.5 * $b - 0.5 * $bneu,
2330
    $z + $cneu,
2331
    $farbe, $fuellart , $fuellfarbe , $linienart , $linienstaerke ,
2332
    $parameter
2333
);
2334
}
2335
}
2336
}
2337
}
2338
}
2339
sub kegel {
2340
    my ( $layer, $x, $y, $z, $r, $h, $weiter, $farbe, $fuellart , $fuellfarbe ,
2341
        $linienart , $linienstaerke , $parameter )
2342
        = @_ ;
2343
2344
    my ( $xmitte, $ymitte, $zmitte );
2345
2346
    my ( $i, $w1, $w2);
2347
2348
    $xmitte = $x;
2349
    $ymitte = $y;
2350
    $zmitte = $z + $h;
2351
2352
# ellipse3d ( $layer , $x, $y, $z, $r, 0, 0, 0, $r, 0, 0, 360, $farbe , $fuellart , $fuellfarbe ,
2353
    $linienart , $linienstaerke , $parameter );
2354
2355
for ( $i = 0 ; $i <= 360 ; $i = $i + $weiter ) {
2356
2357
    $w2 = 90 + $i;
2358
    $w1 = 90 + $i + $weiter;
2359
    if ( $fuellart ne "" ) {
2360
        befehlfuellungerzeugen(
2361
            $layer, "pscustom", $farbe, $fuellart ,
2362
            $fuellfarbe , $linienart , $linienstaerke, $parameter,
2363
            "", ""
2364
        );
2365
2366
        pstricks( $layer, "{\\gsave}");
2367
2368
        ellipse3d( $layer, $x, $y, $z, $r, 0, 0, 0, $r, 0, $w2, $w1, $farbe,
2369
            $fuellart , $fuellfarbe , $linienart , $linienstaerke ,
2370
            $parameter );
2371
2372
        linie (
2373
            $layer, $xmitte, $ymitte, $zmitte,
2374

```

nachkommastellen(\$x + cos(degtorad(\$w2)) * \$r, 4),	2375
nachkommastellen(\$y + sin(degtorad(\$w2)) * \$r, 4),	2376
\$z, "", "", "", "", ""	2377
);	2378
linie (	2379
\$layer,	2380
nachkommastellen(\$x + cos(degtorad(\$w1)) * \$r, 4),	2381
nachkommastellen(\$y + sin(degtorad(\$w1)) * \$r, 4),	2382
\$z, \$xmitte, \$ymitte, \$zmitte, "", "", "", "", ""	2383
);	2384
	2385
befehlfluellungerzeugen(\$layer, " fill ", "", \$fuellart, \$fuellfarbe,	2386
"", "", \$parameter, "", "");	2387
pstricks(\$layer, "\\grestore");	2388
}	2389
	2390
}	2391
	2392
ellipse3d(\$layer, \$x, \$y, \$z, \$r, 0, 0, 0, \$r, 0, 0, 360, \$farbe,	2393
\$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke, \$parameter);	2394
	2395
for (\$i = 0 ; \$i <= 360 ; \$i = \$i + \$weiter) {	2396
	2397
\$w2 = 90 + \$i;	2398
linie (	2399
\$layer,	2400
\$xmitte,	2401
\$ymitte,	2402
\$zmitte,	2403
nachkommastellen(\$x + cos(degtorad(\$w2)) * \$r, 4),	2404
nachkommastellen(\$y + sin(degtorad(\$w2)) * \$r, 4),	2405
\$z,	2406
\$linienart,	2407
\$linienstaerke,	2408
\$farbe,	2409
\$parameter,	2410
""	2411
);	2412
}	2413
	2414
}	2415
	2416
sub kegelstumpf {	2417
my (	2418
\$layer, \$x, \$y, \$z,	2419
\$r, \$h, \$hoehe, \$weiter,	2420
\$farbe, \$fuellart, \$fuellfarbe, \$linienart,	2421
\$linienstaerke, \$parameter	2422
) = @_;	2423
kreiskegelstumpf(	2424
\$layer, \$x, \$y,	2425
\$z, \$r, (\$h - \$hoehe) / \$h * \$r,	2426
(\$h - \$hoehe), \$weiter, \$farbe,	2427
\$fuellart, \$fuellfarbe, \$linienart,	2428
\$linienstaerke, \$parameter	2429
);	2430
}	2431
	2432
sub kreiskegelstumpf {	2433
my (\$layer, \$x, \$y, \$z, \$r1, \$r2, \$h, \$weiter, \$farbe, \$fuellart,	2434
	2435

```

    $fuellfarbe, $linienart, $linienstaerke, $parameter )
    = @_;
2436
2437
2438
set3dpstricks(1);
2439
my ( $xmitte, $ymitte, $zmitte, $i, $w1, $w2 );
2440
2441
$xmitte = $x;
2442
$ymitte = $y;
2443
$zmitte = $z + $h;
2444
2445
for ( $i = 0 ; $i <= 360 ; $i = $i + $weiter ) {
2446
    $w2 = 0 + $i;
2447
    $w1 = 0 + $i + $weiter;
2448
2449
    if ( $fuellart ne "" ) {
2450
        befehlfuellungerzeugen(
2451
            $layer,      "pscustom", $farbe,      $fuellart ,
2452
            $fuellfarbe, $linienart, $linienstaerke, $parameter,
2453
            "",          ""
2454
        );
2455
        pstricks( $layer, "{\\gsave" );
2456
        ellipse3d( $layer, $x, $y, $z, $r1, 0, 0, 0, $r1, 0, $w2, $w1,
2457
            $farbe, $fuellart, $fuellfarbe, $linienart, $linienstaerke,
2458
            $parameter );
2459
        linie (
2460
            $layer,
2461
            nachkommastellen( $x + cos( degtorad($w2) ) * $r1, 4 ),
2462
            nachkommastellen( $y + sin( degtorad($w2) ) * $r1, 4 ),
2463
            $z,
2464
            nachkommastellen( $xmitte + cos( degtorad($w2) ) * $r2, 4 ),
2465
            nachkommastellen( $ymitte + sin( degtorad($w2) ) * $r2, 4 ),
2466
            $zmitte,
2467
            "",
2468
            "",
2469
            "",
2470
            "",
2471
            "",
2472
        );
2473
        ellipse3d(
2474
            $layer,      $xmitte,  $ymitte,  $zmitte,
2475
            $r2,         0,        0,        0,
2476
            $r2,         0,        $w2,     $w1,
2477
            $farbe,      $fuellart, $fuellfarbe, $linienart,
2478
            $linienstaerke, $parameter
2479
        );
2480
        linie (
2481
            $layer,
2482
            nachkommastellen( $xmitte + cos( degtorad($w1) ) * $r2, 4 ),
2483
            nachkommastellen( $ymitte + sin( degtorad($w1) ) * $r2, 4 ),
2484
            $zmitte,
2485
            nachkommastellen( $x + cos( degtorad($w1) ) * $r1, 4 ),
2486
            nachkommastellen( $y + sin( degtorad($w1) ) * $r1, 4 ),
2487
            $z,
2488
            "",
2489
            "",
2490
            "",
2491
            "",
2492
            "",
2493
        );
2494
        befehlfuellungerzeugen(
2495
2496

```

\$layer, " fill ", "blue", \$fuellart ,	2497
\$fuellfarbe, "", "", \$parameter,	2498
"", ""	2499
);	2500
pstricks(\$layer, "\\grestore");	2501
}	2502
	2503
}	2504
ellipse3d(\$layer, \$x, \$y, \$z, \$r1, 0, 0, 0, \$r1, 0, 0, 360, \$farbe,	2505
\$fuellart , \$fuellfarbe , \$linienart , \$linienstaerke , \$parameter);	2506
	2507
for (\$i = 0 ; \$i <= 360 ; \$i = \$i + \$weiter) {	2508
\$w2 = \$i;	2509
linie (	2510
\$layer,	2511
nachkommastellen(\$xmitte + cos(degtorad(\$w2)) * \$r2, 4),	2512
nachkommastellen(\$ymitte + sin(degtorad(\$w2)) * \$r2, 4),	2513
\$zmitte,	2514
nachkommastellen(\$x + cos(degtorad(\$w2)) * \$r1, 4),	2515
nachkommastellen(\$y + sin(degtorad(\$w2)) * \$r1, 4),	2516
\$z,	2517
\$linienart ,	2518
\$linienstaerke ,	2519
\$farbe,	2520
\$parameter,	2521
""	2522
);	2523
}	2524
ellipse3d(\$layer, \$xmitte, \$ymitte, \$zmitte, \$r2, 0, 0, 0, \$r2, 0, 0, 360,	2525
\$farbe, \$fuellart , \$fuellfarbe , \$linienart , \$linienstaerke ,	2526
\$parameter);	2527
	2528
}	2529
	2530
<i># Ein Zylinder wird gezeichnet</i>	2531
sub zylinder {	2532
	2533
my (\$layer, \$x, \$y, \$z, \$r, \$h, \$weiter, \$farbe, \$fuellart , \$fuellfarbe ,	2534
\$linienart , \$linienstaerke , \$parameter)	2535
= @_;	2536
	2537
kreisegelstumpf(\$layer, \$x, \$y, \$z, \$r, \$r, \$h, \$weiter, \$farbe,	2538
\$fuellart , \$fuellfarbe , \$linienart , \$linienstaerke , \$parameter);	2539
	2540
}	2541
	2542
1;	2543

A.6. Das Modul für Funktionen

Listing A.6: funktionen.pm

#####	1
<i># Modul fuer allgemeine Funktionen</i>	2
<i>#</i>	3
<i># Letzte Aenderung: 18.08.2011</i>	4
#####	5
use POSIX qw /floor ceil/;	6
use Math::Trig;	7

```

use strict ;
use warnings;

#####
### Funktionen
### Hilfsfunktionen
#####

sub degtorad {
    my ($wert) = @_ ;
    return $wert * pi() / 180;
}

sub radtodeg {
    my ($wert) = @_ ;
    return $wert * 180 / pi();
}

sub abrunden {
    my ( $wert, $schritt ) = @_ ;
    return $schritt * floor( $wert / $schritt );
}

sub aufrunden {
    my ( $wert, $schritt ) = @_ ;
    return $schritt * ceil( $wert / $schritt );
}

sub nachkommastellen {
    my ( $wert, $nachkommastelle ) = @_ ;

    my $nachkommastellentext = "%0." . $nachkommastelle . "f";
    if ( $nachkommastelle >= 0 ) {
        $wert = sprintf( $nachkommastellentext, $wert );
    }
    else { $wert = $wert; }
    return $wert;
}

sub fact {
    my ($n) = shift;
    if ( ( $n == 1 ) || ( $n == 0 ) ) { return 1; }
    elsif ( $n < 0 ) { return "ERROR"; }
    else { return $n * fact( $n - 1 ); }
}

sub logarithmus { return log( $_[0] ) / log( $_[1] ); }

sub round {
    my ($number) = shift;
    return int( $number + .5 );
}

#####
##### Funktionen zum Berechnen von maximal und minimal
##### Werten im Werte- und Definitionsbereich.
#####

sub calcfunk {
    my ( $funktionsausdruck, $variable, $wert ) = @_ ;

    my @ergebnis = regelwerk( tokenizer($funktionsausdruck) );

```

```

return rechenbaum_ausrechnen(
    $ergebnis[0]{ 'Baum',
    {
        $variable => $wert,
        'pi'      => pi(),
        'e'       => exp(1)
    }
);
}

# Berechnet den ymin und ymax Wert fuer eine Funktion
sub calcyminmax {
    my ( $func, $xminwert, $xmaxwert, $minmax ) = @_;
    my ( $schritt, $xsuche, $ysuche, $yminwert, $ymaxwert );
    my $schrittzahl = 500;

    # Suche von ymin, indem der ganze Definitionsbereich durchgegangen wird und
    # der zugehoerige Wertebereich ausgerechnet wird.
    $yminwert = calcfunk( $func, "x", $xminwert );
    $ymaxwert = calcfunk( $func, "x", $xminwert );
    for ( $schritt = 0 ; $schritt < $schrittzahl ; $schritt++ ) {
        $xsuche = $xminwert +
            ( $xmaxwert - $xminwert ) * ( $schritt / ( $schrittzahl - 1 ) );
        $ysuche = calcfunk( $func, "x", $xsuche );
        $yminwert = $ysuche if ( $yminwert > $ysuche );
        $ymaxwert = $ysuche if ( $ymaxwert < $ysuche );
    }
    return $yminwert if ( $minmax eq "min" );
    return $ymaxwert if ( $minmax eq "max" );
    printf( "calcyminmax: Unbekannter Parameter minmax='%s'\n", $minmax );
}

sub calcymin {
    my ( $func, $xminwert, $xmaxwert ) = @_;
    return nachkommastellen( calcyminmax( $func, $xminwert, $xmaxwert, "min" ),
        4 );
}

sub calcymax {
    my ( $func, $xminwert, $xmaxwert ) = @_;
    return nachkommastellen( calcyminmax( $func, $xminwert, $xmaxwert, "max" ),
        4 );
}

# Berechnet den ymin und ymax Wert fuer zwei Funktion
sub calcfymin {
    my ( $func1, $func2, $xminwert, $xmaxwert ) = @_;
    my $ymin1 = calcymin( $func1, $xminwert, $xmaxwert );
    my $ymin2 = calcymin( $func2, $xminwert, $xmaxwert );
    if ( $ymin1 < $ymin2 ) { return nachkommastellen( $ymin1, 4 ); }
    else { return nachkommastellen( $ymin2, 4 ); }
}

sub calcfymax {
    my ( $func1, $func2, $xminwert, $xmaxwert ) = @_;
    my $ymax1 = calcymax( $func1, $xminwert, $xmaxwert );
    my $ymax2 = calcymax( $func2, $xminwert, $xmaxwert );
    if ( $ymax1 > $ymax2 ) { return nachkommastellen( $ymax1, 4 ); }
    else { return nachkommastellen( $ymax2, 4 ); }
}

```

```

# Verschiebt den xmin und xmax Wert so, dass Graph komplett in [ymin..ymax] liegt
sub calcxminmax {
  my ( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert, $minmax ) = @_;
  my ( $bereichmin, $bereichmax, $schritt, $xsuche, $ysuche, $schrittzahl,
    $gueltig );

  $schrittzahl = 1000;
  $gueltig = 0;
  $bereichmin = $xminwert;
  $bereichmax = $xminwert;

  for ( $schritt = 0 ; $schritt < $schrittzahl ; $schritt++ ) {
    $xsuche = $xminwert +
      ( $xmaxwert - $xminwert ) * ( $schritt / ( $schrittzahl - 1 ) );

    $ysuche = calcfunk( $func, "x", $xsuche );

    if ( ( $ysuche >= $yminwert ) && ( $ysuche <= $ymaxwert ) ) {

      $bereichmin = $xsuche if ( !$gueltig );
      $bereichmax = $xsuche;

      $gueltig = 1;
    }
    else { $gueltig = 0; }
  }

  return nachkommastellen( $bereichmin, 4 ) if ( $minmax eq "min" );
  return nachkommastellen( $bereichmax, 4 )
    if ( $minmax eq "max" );
  printf( "calcxminmax: Unbekannter Parameter minmax='%s'\n", $minmax );
}

sub calcxmin {
  my ( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert ) = @_;
  return nachkommastellen(
    calcxminmax( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert, "min" ),
    4
  );
}

sub calcxmax {
  my ( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert ) = @_;
  return nachkommastellen(
    calcxminmax( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert, "max" ),
    4
  );
}

# Verschiebt den xmin und xmax Wert so, dass Graph komplett in [zmin..zmax] liegt
sub calcxminmax3d {
  my (
    $func, $xminwert, $xmaxwert, $yminwert,
    $ymaxwert, $zminwert, $zmaxwert, $minmax
  ) = @_;
  my (
    $bereichmin, $bereichmax, $xsuche, $ysuche,
    $zsuche, $schrittzahl, $gueltig, $dx,
    $dy, $i, $j
  );

```

\$schrittanzahl = 10;	191
\$gueltig = 0;	192
\$bereichmin = \$yminwert;	193
\$bereichmax = \$yminwert;	194
	195
	196
for (\$i = 0 ; \$i < \$schrittanzahl ; \$i++) {	197
	198
\$xsuche = \$xminwert +	199
(\$xmaxwert - \$xminwert) * (\$i / (\$schrittanzahl - 1));	200
	201
for (\$j = 0 ; \$j < \$schrittanzahl ; \$j++) {	202
\$ysuche = \$yminwert +	203
(\$ymaxwert - \$yminwert) * (\$j / (\$schrittanzahl - 1));	204
\$zsuche = calcfunk3d(\$func, "x", \$xsuche, "y", \$ysuche);	205
if ((\$zsuche >= \$zminwert) && (\$zsuche <= \$zmaxwert)) {	206
	207
\$bereichmin = \$ysuche if (!\$gueltig);	208
\$bereichmax = \$ysuche;	209
	210
\$gueltig = 1;	211
}	212
else { \$gueltig = 0; }	213
}	214
	215
	216
return nachkommastellen(\$bereichmin, 4) if (\$minmax eq "min");	217
return nachkommastellen(\$bereichmax, 4) if (\$minmax eq "max");	218
printf("calcxminmax: Unbekannter Parameter minmax='%s'\n", \$minmax);	219
}	220
	221
sub calcymin3d {	222
my (	223
\$func, \$xminwert, \$xmaxwert, \$yminwert,	224
\$ymaxwert, \$zminwert, \$zmaxwert	225
) = @_;	226
return nachkommastellen(	227
calcxminmax3d(	228
\$func, \$xminwert, \$xmaxwert, \$yminwert,	229
\$ymaxwert, \$zminwert, \$zmaxwert, "min"	230
),	231
4	232
);	233
}	234
	235
sub calcymax3d {	236
my (	237
\$func, \$xminwert, \$xmaxwert, \$yminwert,	238
\$ymaxwert, \$zminwert, \$zmaxwert	239
) = @_;	240
return nachkommastellen(	241
calcxminmax3d(	242
\$func, \$xminwert, \$xmaxwert, \$yminwert,	243
\$ymaxwert, \$zminwert, \$zmaxwert, "max"	244
),	245
4	246
);	247
}	248
	249
sub calcparaminmax {	250
my (\$func1, \$func2, \$func3, \$t1, \$t2, \$u1, \$u2) = @_;	251

```

my ( $i, $j );
if ( $func3 eq "" ) {
    for ( $i = $t1 ; $i < $t2 ; $i = $i + 0.2 ) {
        minmax(
            calcfunk( $func1, "t", $i ),
            calcfunk( $func2, "t", $i ),
            calcfunk( $func1, "t", $i ),
            calcfunk( $func2, "t", $i )
        );
        setachsengraph(
            calcfunk( $func1, "t", $i ),
            calcfunk( $func2, "t", $i ),
            calcfunk( $func1, "t", $i ),
            calcfunk( $func2, "t", $i )
        );
    }
}
else {
    if ( $u1 == $u2 ) {
        for ( $i = $t1 ; $i < $t2 ; $i = $i + 0.1 ) {
            minmax3d(
                calcfunk3d( $func1, "t", $i, "u", 0 ),
                calcfunk3d( $func2, "t", $i, "u", 0 ),
                calcfunk3d( $func3, "t", $i, "u", 0 ),
                calcfunk3d( $func1, "t", $i, "u", 0 ),
                calcfunk3d( $func2, "t", $i, "u", 0 ),
                calcfunk3d( $func3, "t", $i, "u", 0 )
            );
            setachsengraph3d(
                calcfunk3d( $func1, "t", $i, "u", 0 ),
                calcfunk3d( $func2, "t", $i, "u", 0 ),
                calcfunk3d( $func3, "t", $i, "u", 0 ),
                calcfunk3d( $func1, "t", $i, "u", 0 ),
                calcfunk3d( $func2, "t", $i, "u", 0 ),
                calcfunk3d( $func3, "t", $i, "u", 0 )
            );
        }
    }
    else {
        for ( $j = $u1 ; $j < $u2 ; $j = $j + 0.1 ) {
            for ( $i = $t1 ; $i < $t2 ; $i = $i + 0.1 ) {
                minmax3d(
                    nachkommastellen(
                        calcfunk3d( $func1, "t", $i, "u", $j ), 4
                    ),
                    nachkommastellen(
                        calcfunk3d( $func2, "t", $i, "u", $j ), 4
                    ),
                    nachkommastellen(
                        calcfunk3d( $func3, "t", $i, "u", $j ), 4
                    ),
                    nachkommastellen(
                        calcfunk3d( $func1, "t", $i, "u", $j ), 4
                    ),
                    nachkommastellen(

```

```

        calcfunk3d( $func2, "t", $i, "u", $j ), 4
    ),
    nachkommastellen(
        calcfunk3d( $func3, "t", $i, "u", $j ), 4
    )
);
setachsengraph3d(
    nachkommastellen(
        calcfunk3d( $func1, "t", $i, "u", $j ), 4
    ),
    nachkommastellen(
        calcfunk3d( $func2, "t", $i, "u", $j ), 4
    ),
    nachkommastellen(
        calcfunk3d( $func3, "t", $i, "u", $j ), 4
    ),
    nachkommastellen(
        calcfunk3d( $func1, "t", $i, "u", $j ), 4
    ),
    nachkommastellen(
        calcfunk3d( $func2, "t", $i, "u", $j ), 4
    ),
    nachkommastellen(
        calcfunk3d( $func3, "t", $i, "u", $j ), 4
    )
);
}
}
}
}
}
my $n = 0;

# Die Aenderungsrate zwischen zwei Punkten wird ermittelt

sub calcaenderungsrate {
    my ( $x1, $y1, $x2, $y2 ) = @_;
    $n++;
    if ( ( $x2 - $x1 ) == 0 ) { return "ERROR"; }

    else { return ( $y2 - $y1 ) / ( $x2 - $x1 ); }
}

sub calcdiff {
    my ( $wert1, $wert2, $abstand ) = @_;
    my $wert = 1;
    if ( abs( $wert1 * $wert - $wert2 * $wert ) <= $abstand ) { return 1; }
    else { return 0; }
}

##Berechnungen fuer die Ableitungen
sub calcablxminmax {
    my ( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert, $minmax ) = @_;
    my ( $bereichmin, $bereichmax, $schritt, $xsuche, $ysuche );
    my $schrittzahl = 1000;
    my $gueltig = 0;
    $bereichmin = $bereichmax = $xminwert;
    for ( $schritt = 0 ; $schritt < $schrittzahl ; $schritt++ ) {

```

```

$xsuche = $xminwert +
    ( $xmaxwert - $xminwert ) * ( $schritt / ( $schrittzahl - 1 ) );
$ysuche = calcableitung( $func, $xsuche );
if ( ( $ysuche >= $yminwert ) && ( $ysuche <= $ymaxwert ) ) {
    $bereichmin = $xsuche if ( !$gueltig );
    $bereichmax = $xsuche;
    $gueltig = 1;
}
else { $gueltig = 0; }
}
return nachkommastellen( $bereichmin, 4 )
if ( $minmax eq "min" );
return nachkommastellen( $bereichmax, 4 )
if ( $minmax eq "max" );
printf( "calcblxminmax: Unbekannter Parameter minmax='%s'\n", $minmax );
}

sub calcablxmin {
    my ( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert ) = @_;
    return calcablxminmax( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert,
        "min" );
}

sub calcablxmax {
    my ( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert ) = @_;
    return calcablxminmax( $func, $xminwert, $xmaxwert, $yminwert, $ymaxwert,
        "max" );
}

# Berechnung der Ableitung ueber den Differenzenquotienten
sub calcableitung {
    my ( $func, $xwert ) = @_;
    my $hwert = 0.00000001;
    my $ywert =
        ( calcfunkt( $func, "x", $xwert + $hwert ) -
          calcfunkt( $func, "x", $xwert ) ) / $hwert;
    return nachkommastellen( $ywert, 4 );
}

#####
##### Berechnung des Funktionswertes t(x), wenn t die
##### Tangente an der Funktion func im Punkte festwert ist
#####
sub calctangente {
    my ( $func, $xwert, $festwert ) = @_;
    return nachkommastellen(
        calcfunkt( $func, "x", $festwert ) +
        ( calcableitung( $func, $festwert ) * ( $xwert - $festwert ) ),
        4
    );
}

#####
##### Berechnung der Schnittpunkte von zwei Funktionen
#####
sub calcschnittpunkte {
    my ( $func1, $func2, $xminwert, $xmaxwert, $linksrechts ) = @_;
    my (
        $xsuche,      $links,      $rechts,  @xpunkte,
        @xselbst,     $schritt,    $ysuche1, $ysuche2,
        $welchegroesser, $schrittzahl, $zaehler
    );
    $ysuche1 = calcfunkt( $func1, "x", getminmaxachsen( "x", "min" ) );

```

```

$ysuche2 = calcfunk( $func2, "x", getminmaxachsen( "x", "min" ) );
$welchegroesser = ( $ysuche1 > $ysuche2 ) ? 1 : 2;
$schrittanzahl = 9000;
$zaehler = 0;

for ( $schritt = 0 ; $schritt < $schrittanzahl ; $schritt++ ) {
    $xsuche = $xminwert +
        ( $xmaxwert - $xminwert ) * ( $schritt / ( $schrittanzahl - 1 ) );
    $ysuche1 = calcfunk( $func1, "x", $xsuche );
    $ysuche2 = calcfunk( $func2, "x", $xsuche );
    if ( ( $ysuche1 > $ysuche2 ) xor( $welchegroesser == 1 ) ) {
        $welchegroesser = ( $ysuche1 > $ysuche2 ) ? 1 : 2;
        $zaehler++;
        if ( $zaehler > 1 ) { $rechts = $xsuche; }
        else { $links = $xsuche; $rechts = $xsuche; }
    }
}
if ( $zaehler == 0 ) { $links = $xminwert; $rechts = $xmaxwert; }
return nachkommastellen( $links, 4 ) if ( $linksrechts eq "links" );
return nachkommastellen( $rechts, 4 ) if ( $linksrechts eq "rechts" );
printf( "calcschnittpunkte: Unbekannter Parameter linksrechts='%s'\n",
    $linksrechts );
}

sub calcschnittlinks {
    my ( $func1, $func2, $xminwert, $xmaxwert ) = @_;
    calcschnittpunkte( $func1, $func2, $xminwert, $xmaxwert, "links" );
}

sub calcschnittrechts {
    my ( $func1, $func2, $xminwert, $xmaxwert ) = @_;
    calcschnittpunkte( $func1, $func2, $xminwert, $xmaxwert, "rechts" );
}

#####
##### Berechnungen fuer Vektoren
#####

sub calcendpunkttx {
    my ( $x, $laenge, $winkel ) = @_;
    return nachkommastellen( $x + cos( degtorad( $winkel ) ) * $laenge, 4 );
}

sub calcendpunktty {
    my ( $y, $laenge, $winkel ) = @_;
    return nachkommastellen( $y + sin( degtorad( $winkel ) ) * $laenge, 4 );
}

sub calcsumlaenge {
    my ( $x1, $y1, $l1, $w1, $l2, $w2 ) = @_;
    my $xsum = calcendpunkttx( 0, $l1, $w1 ) + calcendpunkttx( 0, $l2, $w2 );
    my $ysum = calcendpunktty( 0, $l1, $w1 ) + calcendpunktty( 0, $l2, $w2 );
    return nachkommastellen( sqrt( ( $xsum * $xsum ) + ( $ysum * $ysum ) ), 4 );
}

# Winkel wird in Grad ausgegeben
sub calcneuerwinkel {
    my ( $x1, $y1, $l1, $w1, $l2, $w2 ) = @_;
    my $x12 = calcendpunkttx( $x1, $l1, $w1 );
    my $y12 = calcendpunktty( $y1, $l1, $w1 );
    my $x22 = calcendpunkttx( $x12, $l2, $w2 );
    my $y22 = calcendpunktty( $y12, $l2, $w2 );
    my $dx = $x22 - $x1;

```

```

my $dy = $y22 - $y1;
return nachkommastellen( radtodeg( atan2( $dy, $dx ) ), 4 );
}
#####
# Rechnen mit Vektoren
#####
#####$winkel2 = $winkel1 + $winkel
#####
# vektordifferenz (A,B) liefert den Vektor von A nach B
sub vektordifferenz {
my ( $xyoderz, $startx, $starty, $startz, $endex, $endey, $endez ) = @_;
if ( $xyoderz eq "x" ) { return $endex - $startx; }
elsif ( $xyoderz eq "y" ) { return $endey - $starty; }
elsif ( $xyoderz eq "z" ) { return $endez - $startz; }
else { return 0; }
}

sub vektorprodukt {
my ( $startx, $starty, $startz, $endex, $endey, $endez ) = @_;
return $startx * $endex + $starty * $endey + $startz * $endez;
}

sub vektorbetrag {
my ( $vektorx, $vektory, $vektorz ) = @_;
return
sqrt( $vektorx * $vektorx + $vektory * $vektory + $vektorz * $vektorz );
}

sub vektorwinkel {
my ( $vektor1x, $vektor1y, $vektor1z, $vektor2x, $vektor2y, $vektor2z ) =
@_;
return nachkommastellen(
radtodeg(
acos(
vektorprodukt(
$vektor1x, $vektor1y, $vektor1z,
$vektor2x, $vektor2y, $vektor2z
) / (
vektorbetrag( $vektor1x, $vektor1y, $vektor1z ) *
vektorbetrag( $vektor2x, $vektor2y, $vektor2z )
)
),
4
);
}

sub minimum {
my ( $x, $y, $z ) = @_;
my $hilf1;
if ( $x < $y ) { $hilf1 = $x; }
else { $hilf1 = $y; }
if ( $hilf1 < $z ) { return $hilf1; }
else { return $z; }
}

sub istzahl {
my ($zahl) = @_;
if ( defined $zahl ) {
if ( $zahl eq "" ) { return 0; }
elsif ( $zahl =~ /[a-zA-Z!+)]/ ) { return 0; }
}
}

```

496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555

<pre> else { return 1; } } else { return 0; } } sub max { my \$a = shift; my \$b = shift; return \$a > \$b ? \$a : \$b; } sub min { my \$a = shift; my \$b = shift; return \$a < \$b ? \$a : \$b; } sub farbverlauf { my (\$farbe, \$i) = @_ ; \$farbe =~ s/!/ /g; \$farbe =~ s/1/ /g; \$farbe =~ s/2/ /g; \$farbe =~ s/3/ /g; \$farbe =~ s/4/ /g; \$farbe =~ s/5/ /g; \$farbe =~ s/6/ /g; \$farbe =~ s/7/ /g; \$farbe =~ s/8/ /g; \$farbe =~ s/9/ /g; \$farbe =~ s/0/ /g; my \$n = abrunden(\$n + 100 / \$i, 1); return \$farbe . "!" . \$n; } 1; </pre>	<pre> 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 </pre>
--	--

A.7. Das Modul für den Funktionsparser

Listing A.7: funktionsparser.pm

<pre> ##### # Modul fuer einen Funktionsparser # Letzte Aenderung: 24.07.2010 ##### use POSIX qw /floor ceil/; use Math::Trig; use strict ; use warnings; use Data::Dumper; my \$hilfsstring = ""; sub tokenizer { my (\$text) = @_ ; my (\$token, \$vorzeichen); my @listevontokens; \$vorzeichen = 1; \$text =~ s/ //g; </pre>	<pre> 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 </pre>
--	---

```

$text = lc($text);
while ( $text ne "" ) {
  if ( ( $text =~ /^([-+]?[0-9]+(\.[0-9]+)?([eE][-+]?[0-9]+)?)/ )
    && ( $vorzeichen == 1 ) )
  {

    # Zahlen mit Vorzeichen
    $token = $1;
    $vorzeichen = 0;
    push @listevontokens,
    {
      'Inhalt' => $token,
      'Typ' => "Term",
      'Baum' => {
        'knoten' => $token,
        'links' => undef,
        'rechts' => undef
      }
    };
  }
  elsif ( ( $text =~ /^([0-9]+(\.[0-9]+)?([eE][-+]?[0-9]+)?)/ )
    && ( $vorzeichen == 0 ) )
  {

    # Zahlen ohne Vorzeichen
    $token = $1;
    $vorzeichen = 0;
    push @listevontokens,
    {
      'Inhalt' => $token,
      'Typ' => "Term",
      'Baum' => {
        'knoten' => $token,
        'links' => undef,
        'rechts' => undef
      }
    };
  }
  elsif ( $text =~ /^([-])/ ) {

    # Operatoren
    $token = $1;
    $vorzeichen = 1;
    push @listevontokens,
    {
      'Inhalt' => $token,
      'Typ' => "minus",
      'Baum' => {
        'knoten' => $token,
        'links' => undef,
        'rechts' => undef
      }
    };
  }
  elsif ( $text =~ /^[+]/ ) {

    # Operatoren
    $token = $1;
    $vorzeichen = 1;
    push @listevontokens,
    {

```

```

    'Inhalt' => $token,
    'Typ'    => "plus",
    'Baum'   => {
        'knoten' => $token,
        'links'  => undef,
        'rechts' => undef
    }
};
}
elseif ( $text =~ /^(.*)/ ) {
    80
    81
    82
    83
    84
    85
    86
    87
    88
    89
    90
    91
    92
    93
    94
    95
    96
    97
    98
    99
    100
    101
    102
    103
    104
    105
    106
    107
    108
    109
    110
    111
    112
    113
    114
    115
    116
    117
    118
    119
    120
    121
    122
    123
    124
    125
    126
    127
    128
    129
    130
    131
    132
    133
    134
    135
    136
    137
    138
    139
    140
    # Operatoren
    $token    = $1;
    $vorzeichen = 1;
    push @listevontokens,
    {
        'Inhalt' => $token,
        'Typ'    => "mal",
        'Baum'   => {
            'knoten' => $token,
            'links'  => undef,
            'rechts' => undef
        }
    };
}
elseif ( $text =~ /^([:\|\/])/ ) {
    # Operatoren
    $token    = $1;
    $vorzeichen = 1;
    push @listevontokens,
    {
        'Inhalt' => $token,
        'Typ'    => "geteilt",
        'Baum'   => {
            'knoten' => $token,
            'links'  => undef,
            'rechts' => undef
        }
    };
}
elseif ( $text =~
/^(sin|cos|tan|acos|asin|log|ln|ceiling|floor|truncate|round|sqrt|abs|fact|exp)/
)
{
    # Funktionen
    $token    = $1;
    $vorzeichen = 1;
    push @listevontokens,
    {
        'Inhalt' => $token,
        'Typ'    => "Funktion",
        'Baum'   => {
            'knoten' => $token,
            'links'  => undef,
            'rechts' => undef
        }
    };
}

```

```

elseif ( $text =~ /^(\()/ ) {
    # öffnende Klammer
    $token      = $1;
    $vorzeichen = 1;
    push @listevontokens,
    {
        'Inhalt' => $token,
        'Typ'    => "(",
        'Baum'   => {
            'knoten' => $token,
            'links'   => undef,
            'rechts'  => undef
        }
    };
}
elseif ( $text =~ /^(\))/ ) {
    # schließende Klammer
    $token      = $1;
    $vorzeichen = 0;
    push @listevontokens,
    {
        'Inhalt' => $token,
        'Typ'    => ")",
        'Baum'   => {
            'knoten' => $token,
            'links'   => undef,
            'rechts'  => undef
        }
    };
}
elseif ( $text =~ /^(\^)/ ) {
    # Hochzeichen
    $token      = $1;
    $vorzeichen = 1;
    push @listevontokens,
    {
        'Inhalt' => $token,
        'Typ'    => "hoch",
        'Baum'   => {
            'knoten' => $token,
            'links'   => undef,
            'rechts'  => undef
        }
    };
}
elseif ( $text =~ /^[a-z][0-9a-z]*/ ) {
    # Variable
    $token      = $1;
    $vorzeichen = 0;
    push @listevontokens,
    {
        'Inhalt' => $token,
        'Typ'    => "Term",
        'Baum'   => {
            'knoten' => $token,
            'links'   => undef,
            'rechts'  => undef
        }
    };
}

```

```

    }
    };
}
else {
    # Rest
    $token = $text;
    $vorzeichen = 1;
    push @listevontokens,
    {
        'Inhalt' => $token,
        'Typ' => "Rest",
        'Baum' => {
            'knoten' => $token,
            'links' => undef,
            'rechts' => undef
        }
    };
}

#print "Token " . $token . "\n";
$text = substr $text, length $token;
}
return @listevontokens;
}

my $ausgabe = 0;

sub ausgabeliste {
    my @liste = @_;
    my $listenlaenge = scalar @liste;
    my $i;
    if ( $ausgabe == 1 ) {
        printf("%%");
        for ( $i = 0 ; $i < $listenlaenge ; $i++ ) {
            printf( "%s", $liste[$i]{ 'Inhalt' } );
            printf(" ") if ( $i < $listenlaenge - 1 );
        }
        print "\n";
    }
}

sub regelwerk {
    my (@liste) = @_;
    my ( @klammerterm, $i, $j, $gefunden, $klammerauf, $klammerzu,
        $klammertiefe );

    do {
        $gefunden = 0;
        if ( !$gefunden ) {

            $klammerzu = $klammerauf = $klammertiefe = 0;

# Klammern:
# Wenn eine oeffnende Klammer gefunden wurde, dann wird der Zaehler klammertiefe erhoeht ,
# bei einer schliessenden Klammer erniedrigt.
# Bei korrekter Klammerung ist der Zaehler am Ende 0
# und es wird die regelwerk-Funktion auf den Inhalt der Klammer angewendet.

            for ( $i = 0 ; $i < scalar @liste ; $i++ ) {

```

202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262

```

if ( $liste[$i]{ 'Typ' } eq "(" ) {
  if ( ( $klammertiefe == 0 ) ) {
    $klammerauf = $i;
  }
  $klammertiefe++;
}
if ( $liste[$i]{ 'Typ' } eq ")" ) {
  $klammertiefe--;
  if ( $klammertiefe == 0 ) {
    $klammerz zu = $i;
  }
}

}
## Fehler wenn klammertiefe != 0 muss noch bearbeitet werden

if ( $klammerauf != $klammerz zu ) {

  @klammerterm = ();
  for ( $j = $klammerauf + 1 ; $j <= $klammerz zu - 1 ; $j++ ) {
    push @klammerterm,
      {
        'Inhalt' => $liste[$j]{ 'Inhalt' },
        'Typ'    => $liste[$j]{ 'Typ' },
        'Baum'   => {
          'knoten' => $liste[$j]{ 'Baum' }{ 'knoten' },
          'links'  => undef,
          'rechts' => undef
        }
      };
  }

  splice( @liste, $klammerauf, $klammerz zu - $klammerauf + 1,
    regelwerk(@klammerterm) );
  $gefunden = 1;
}

}
if ( !$gefunden ) {

  # Potenzieren
  for ( $i = 0 ; $i < scalar @liste - 2 ; $i++ ) {

    if ( ( $liste[$i]{ 'Typ' } eq "Term" )
      && ( $liste[ $i + 1 ]{ 'Typ' } eq "hoch" )
      && ( $liste[ $i + 2 ]{ 'Typ' } eq "Term" )
      && ( !$gefunden ) )
    {

      splice(
        @liste, $i, 3,
        {
          'Inhalt' => "("
            . $liste[$i]{ 'Inhalt' }
            . $liste[ $i + 1 ]{ 'Inhalt' }
            . $liste[ $i + 2 ]{ 'Inhalt' } . ")",
          'Typ'    => "Term",
          'Baum'   => {
            'knoten' => $liste[ $i + 1 ]{ 'Baum' }{ 'knoten' },
            'links'  => $liste[$i]{ 'Baum' },

```

```

        'rechts' => $liste [ $i + 2 ] { 'Baum' }
    }
}
);
$gefunden = 1;
}
}
}
if ( !$gefunden ) {

# Multiplikation und Division
for ( $i = 0 ; $i < scalar @liste - 2 ; $i++ ) {
    if (
        ( $liste [ $i ] { 'Typ' } eq "Term" )
        && ( ( $liste [ $i + 1 ] { 'Typ' } eq "mal" )
            || ( $liste [ $i + 1 ] { 'Typ' } eq "geteilt" ) )
        && ( $liste [ $i + 2 ] { 'Typ' } eq "Term" )
        && ( !$gefunden )
    )
    {
        splice(
            @liste, $i, 3,
            {
                'Inhalt' => "("
                . $liste [ $i ] { 'Inhalt' }
                . $liste [ $i + 1 ] { 'Inhalt' }
                . $liste [ $i + 2 ] { 'Inhalt' } . ")",
                'Typ' => "Term",
                'Baum' => {
                    'knoten' => $liste [ $i + 1 ] { 'Baum' } { 'knoten' },
                    'links' => $liste [ $i ] { 'Baum' },
                    'rechts' => $liste [ $i + 2 ] { 'Baum' }
                }
            }
        );
        $gefunden = 1;
    }
}
}
if ( !$gefunden ) {

# Funktion
for ( $i = 0 ; $i < scalar @liste - 1 ; $i++ ) {
    if ( ( $liste [ $i ] { 'Typ' } eq "Funktion" )
        && ( $liste [ $i + 1 ] { 'Typ' } eq "Term" )
        && ( !$gefunden ) )
    {
        if ( $liste [ $i ] { 'Inhalt' } eq "exp" ) {
            splice(
                @liste, $i, 2,
                {
                    'Inhalt' => "("
                    . exp(1) . "\^"
                    . $liste [ $i + 1 ] { 'Inhalt' } . ")",
                    'Typ' => "Term",
                    'Baum' => {
                        'knoten' => "^",
                        'links' => {
                            'knoten' => exp(1),
                            'links' => undef,
                            'rechts' => undef
                        }
                    }
                }
            );
            $gefunden = 1;
        }
    }
}
}

```

```

        },
        'rechts' => $liste [ $i + 1 ] { 'Baum' }
    }
}
);
}
else {
    splice(
        @liste, $i, 2,
        {
            'Inhalt' => "("
                . $liste [ $i ] { 'Inhalt' }
                . $liste [ $i + 1 ] { 'Inhalt' } . ")",
            'Typ' => "Term",
            'Baum' => {
                'knoten' => $liste [ $i ] { 'Baum' } { 'knoten' },
                'links' => $liste [ $i + 1 ] { 'Baum' },
                'rechts' => undef
            }
        }
    );
}
$gefunden = 1;
}
}
if ( !$gefunden ) {

    # Addition und Subtraktion
    for ( $i = 0 ; $i < scalar @liste - 2 ; $i++ ) {
        if (
            ( $liste [ $i ] { 'Typ' } eq "Term" )
            && ( ( $liste [ $i + 1 ] { 'Typ' } eq "plus" )
                || ( $liste [ $i + 1 ] { 'Typ' } eq "minus" ) )
            && ( $liste [ $i + 2 ] { 'Typ' } eq "Term" )
            && ( !$gefunden )
        ) {
            splice(
                @liste, $i, 3,
                {
                    'Inhalt' => "("
                        . $liste [ $i ] { 'Inhalt' }
                        . $liste [ $i + 1 ] { 'Inhalt' }
                        . $liste [ $i + 2 ] { 'Inhalt' } . ")",
                    'Typ' => "Term",
                    'Baum' => {
                        'knoten' => $liste [ $i + 1 ] { 'Baum' } { 'knoten' },
                        'links' => $liste [ $i ] { 'Baum' },
                        'rechts' => $liste [ $i + 2 ] { 'Baum' }
                    }
                }
            );
            $gefunden = 1;
        }
    }
}
if ( !$gefunden ) {

    # Term mit Vorzeichen
    for ( $i = 0 ; $i < scalar @liste - 1 ; $i++ ) {

```

```

if (
(
( $liste[$i]{ 'Typ' } eq "plus" )
|| ( $liste[$i]{ 'Typ' } eq "minus" )
)
)&& ( $liste[ $i + 1 ]{ 'Typ' } eq "Term" )
)
{
splice(
@liste, $i, 2,
{
'Inhalt' => "(0"
. $liste[$i]{ 'Inhalt' }
. $liste[ $i + 1 ]{ 'Inhalt' } . ")",
'Typ' => "Term",
'Baum' => {
'knoten' => $liste[$i]{ 'Baum' }{ 'knoten' },
'links' => {
'knoten' => 0,
'links' => undef,
'rechts' => undef
},
'rechts' => $liste[ $i + 1 ]{ 'Baum' }
}
}
);
$gefunden = 1;
}
}
}

} while ($gefunden);

# ausgabeliste ( @liste );
return @liste;
}

sub regelwerkupn {
my @liste = @_;
my ( @klammerterm, $i, $j, $gefunden, $klammerauf, $klammerzu,
$klammertiefe );

do {
$gefunden = 0;
if ( !$gefunden ) {

$klammerzu = $klammerauf = $klammertiefe = 0;

# Klammern:
# Wenn eine oeffnende Klammer gefunden wurde, dann wird der Zaehler klammertiefe erhoeht ,
# bei einer schliessenden Klammer erniedrigt .
# Bei korrekter Klammerung ist der Zaehler am Ende 0
# und es wird die regelwerk-Funktion auf den Inhalt der Klammer angewendet.

for ( $i = 0 ; $i < scalar @liste ; $i++ ) {

if ( $liste[$i]{ 'Typ' } eq "(" ) {
if ( ( $klammertiefe == 0 ) ) {
$klammerauf = $i;
}
$klammertiefe++;

```

```

    }
    if ( $liste[$i]{ 'Typ' } eq "" ) {
        $klammertiefe--;
        if ( $klammertiefe == 0 ) {
            $klammerzu = $i;
        }
    }
}

## Fehler wenn klammertiefe != 0 muss noch bearbeitet werden

if ( $klammerauf != $klammerzu ) {

    @klammerterm = ();
    for ( $j = $klammerauf + 1 ; $j <= $klammerzu - 1 ; $j++ ) {
        push @klammerterm,
        {
            'Inhalt' => $liste[$j]{ 'Inhalt' },
            'Typ'    => $liste[$j]{ 'Typ' },
            'Baum'   => {
                'knoten' => $liste[$j]{ 'Baum' }{ 'knoten' },
                'links'  => undef,
                'rechts' => undef
            }
        };
    }

    splice(
        @liste, $klammerauf,
        $klammerzu - $klammerauf + 1,
        regelwerkupn(@klammerterm)
    );
    $gefunden = 1;
}

}

if ( !$gefunden ) {

    # Potenzieren
    for ( $i = 0 ; $i < scalar @liste - 2 ; $i++ ) {
        if ( ( $liste[$i]{ 'Typ' } eq "Term" )
            && ( $liste[$i + 1]{ 'Typ' } eq "hoch" )
            && ( $liste[$i + 2]{ 'Typ' } eq "Term" ) )
        {

            if ( $liste[$i + 2]{ 'Inhalt' } == 2 ) {
                splice(
                    @liste, $i, 3,
                    {
                        'Inhalt' => "("
                            . $liste[$i]{ 'Inhalt' }
                            . $liste[$i + 1]{ 'Inhalt' }
                            . $liste[$i + 2]{ 'Inhalt' } . ")",
                        'Typ' => "Term",
                        'Baum' => {
                            'knoten' => "mul",
                            'links'  => $liste[$i]{ 'Baum' },
                            'rechts' => {
                                'knoten' => "dup",
                                'links'  => undef,

```

```

        'rechts' => undef
    },
    }
    );
}
else {

    splice(
        @liste, $i, 3,
        {
            'Inhalt' => "("
                . $liste[$i]{ 'Inhalt' }
                . $liste[$i + 1]{ 'Inhalt' }
                . $liste[$i + 2]{ 'Inhalt' } . ")",
            'Typ' => "Term",
            'Baum' => {
                'knoten' =>
                    $liste[$i + 1]{ 'Baum' }{ 'knoten' },
                'links' => $liste[$i]{ 'Baum' },
                'rechts' => $liste[$i + 2]{ 'Baum' }
            }
        }
    );
}
$gefunden = 1;

ausgabeliste(@liste);
}
}
if ( !$gefunden ) {

    # Multiplikation und Division
    for ( $i = 0 ; $i < scalar @liste - 2 ; $i++ ) {
        if (
            ( $liste[$i]{ 'Typ' } eq "Term" )
            && ( ( $liste[$i + 1]{ 'Typ' } eq "mal" )
                || ( $liste[$i + 1]{ 'Typ' } eq "geteilt" ) )
            && ( $liste[$i + 2]{ 'Typ' } eq "Term" )
        ) {
            splice(
                @liste, $i, 3,
                {
                    'Inhalt' => "("
                        . $liste[$i]{ 'Inhalt' }
                        . $liste[$i + 1]{ 'Inhalt' }
                        . $liste[$i + 2]{ 'Inhalt' } . ")",
                    'Typ' => "Term",
                    'Baum' => {
                        'knoten' => $liste[$i + 1]{ 'Baum' }{ 'knoten' },
                        'links' => $liste[$i]{ 'Baum' },
                        'rechts' => $liste[$i + 2]{ 'Baum' }
                    }
                }
            );
            $gefunden = 1;

            ausgabeliste(@liste);
        }
    }
}

```

```

    }
  }
  if ( !$gefunden ) {

    # Funktion
    for ( $i = 0 ; $i < scalar @liste - 1 ; $i++ ) {
      if ( ( $liste[$i]['Typ'] eq "Funktion" )
        && ( $liste[ $i + 1 ]['Typ'] eq "Term" ) )
      {
        if ( $liste[$i]['Inhalt'] eq "exp" ) {
          splice(
            @liste, $i, 2,
            {
              'Inhalt' => "("
                . exp(1) . "\^"
                . $liste[ $i + 1 ]['Inhalt'] . ")",
              'Typ' => "Term",
              'Baum' => {
                'knoten' => "^",
                'links' => {
                  'knoten' => exp(1),
                  'links' => undef,
                  'rechts' => undef
                },
                'rechts' => $liste[ $i + 1 ]['Baum']
              }
            }
          );
        }
        else {
          splice(
            @liste, $i, 2,
            {
              'Inhalt' => "("
                . $liste[$i]['Inhalt']
                . $liste[ $i + 1 ]['Inhalt'] . ")",
              'Typ' => "Term",
              'Baum' => {
                'knoten' => $liste[$i]['Baum']['knoten'],
                'links' => $liste[ $i + 1 ]['Baum'],
                'rechts' => undef
              }
            }
          );
        }
      }
      $gefunden = 1;

      ausgabeliste(@liste);
    }
  }
}

if ( !$gefunden ) {

  # Addition und Subtraktion
  for ( $i = 0 ; $i < scalar @liste - 2 ; $i++ ) {
    if ( ( $liste[$i]['Typ'] eq "Term" )
      && ( ( $liste[ $i + 1 ]['Typ'] eq "plus" ) )
      && ( $liste[ $i + 2 ]['Typ'] eq "Term" ) )
    {
      splice(
        @liste, $i, 3,

```

629
 630
 631
 632
 633
 634
 635
 636
 637
 638
 639
 640
 641
 642
 643
 644
 645
 646
 647
 648
 649
 650
 651
 652
 653
 654
 655
 656
 657
 658
 659
 660
 661
 662
 663
 664
 665
 666
 667
 668
 669
 670
 671
 672
 673
 674
 675
 676
 677
 678
 679
 680
 681
 682
 683
 684
 685
 686
 687
 688
 689

```

{
    'Inhalt' => "("
        . $liste[$i]{ 'Inhalt' }
        . $liste[ $i + 1 ]{ 'Inhalt' }
        . $liste[ $i + 2 ]{ 'Inhalt' } . ")",
    'Typ' => "Term",
    'Baum' => {
        'knoten' => $liste[ $i + 1 ]{ 'Baum' }{ 'knoten' },
        'links' => $liste[$i]{ 'Baum' },
        'rechts' => $liste[ $i + 2 ]{ 'Baum' }
    }
}
);
$gefunden = 1;

ausgabeliste(@liste);
}
}

if ( !$gefunden ) {

    # Addition und Subtraktion
    for ( $i = 0 ; $i < scalar @liste - 2 ; $i++ ) {
        if ( ( $liste[$i]{ 'Typ' } eq "Term" )
            && ( ( $liste[ $i + 1 ]{ 'Typ' } eq "minus" ) )
            && ( $liste[ $i + 2 ]{ 'Typ' } eq "Term" ) )
        {
            splice(
                @liste, $i, 3,
                {
                    'Inhalt' => "("
                        . $liste[$i]{ 'Inhalt' }
                        . $liste[ $i + 1 ]{ 'Inhalt' }
                        . $liste[ $i + 2 ]{ 'Inhalt' } . ")",
                    'Typ' => "Term",
                    'Baum' => {
                        'knoten' => "minus",
                        'links' => $liste[$i]{ 'Baum' },
                        'rechts' => $liste[ $i + 2 ]{ 'Baum' }
                    }
                }
            );
            $gefunden = 1;

            ausgabeliste(@liste);
        }
    }
}

if ( !$gefunden ) {

    # Term mit Vorzeichen
    for ( $i = 0 ; $i < scalar @liste - 1 ; $i++ ) {
        if (
            (
                ( $liste[$i]{ 'Typ' } eq "plus" )
                || ( $liste[$i]{ 'Typ' } eq "minus" )
            )
            && ( $liste[ $i + 1 ]{ 'Typ' } eq "Term" )
        )

```

```

{
    splice(
        @liste, $i, 2,
        {
            'Inhalt' => "(0"
                . $liste[$i]{ 'Inhalt' }
                . $liste[ $i + 1 ]{ 'Inhalt' } . ")",
            'Typ' => "Term",
            'Baum' => {
                'knoten' => "neg",
                'links' => undef,
                'rechts' => $liste[ $i + 1 ]{ 'Baum' }
            }
        }
    );
    $gefunden = 1;

    ausgabeliste(@liste);
}
}
} while ($gefunden);

ausgabeliste(@liste);
return @liste;
}

sub rechenbaum_ausrechnen {
    my $baum = shift;
    my $variablen = shift;

    #print $$baum{'knoten'} . "\n";
    if ( $$baum{'knoten'} eq "*" ) {
        return rechenbaum_ausrechnen( $$baum{'links'}, $variablen ) *
            rechenbaum_ausrechnen( $$baum{'rechts'}, $variablen );
    }
    elsif ( $$baum{'knoten'} eq ":" ) {
        return rechenbaum_ausrechnen( $$baum{'links'}, $variablen ) /
            rechenbaum_ausrechnen( $$baum{'rechts'}, $variablen );
    }
    elsif ( $$baum{'knoten'} eq "/" ) {
        return rechenbaum_ausrechnen( $$baum{'links'}, $variablen ) /
            rechenbaum_ausrechnen( $$baum{'rechts'}, $variablen );
    }
    elsif ( $$baum{'knoten'} eq "+" ) {
        return rechenbaum_ausrechnen( $$baum{'links'}, $variablen ) +
            rechenbaum_ausrechnen( $$baum{'rechts'}, $variablen );
    }
    elsif ( $$baum{'knoten'} eq "^" ) {
        return rechenbaum_ausrechnen( $$baum{'links'}, $variablen )
            **rechenbaum_ausrechnen( $$baum{'rechts'}, $variablen );
    }

    elsif ( $$baum{'knoten'} eq "-" ) {
        return rechenbaum_ausrechnen( $$baum{'links'}, $variablen ) -
            rechenbaum_ausrechnen( $$baum{'rechts'}, $variablen );
    }
    elsif ( defined $$variablen{ $$baum{'knoten'} } ) {
        return $$variablen{ $$baum{'knoten'} };
    }
}

```

```
}
elseif ( $$baum{'knoten'} eq "sin" ) {
    return sin( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "cos" ) {
    return cos( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "tan" ) {
    return tan( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "asin" ) {
    return arcsin( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "acos" ) {
    return acos( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "log" ) {
    return log( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "ceiling" ) {
    return aufrunden( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ),
        1 );
}
elseif ( $$baum{'knoten'} eq "floor" ) {
    return abrunden( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ),
        1 );
}
elseif ( $$baum{'knoten'} eq "truncate" ) {

    if ( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) < 0 ) {
        return aufrunden(
            rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ), 1 );
    }
    else {
        return abrunden(
            rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ), 1 );
    }

}
elseif ( $$baum{'knoten'} eq "round" ) {
    return round( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "sqrt" ) {
    return sqrt( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "abs" ) {
    return abs( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "fact" ) {
    return fact( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
elseif ( $$baum{'knoten'} eq "exp" ) {
    return exp( rechnenbaum_ausrechnen( $$baum{'links'}, $variablen ) );
}
else {

    # ACHTUNG: Keine Fehlerbehandlung!!!
    return $$baum{'knoten'};
}
}
```

```

sub polnisch {
  my $ref = shift;
  $hilfsstring = $$ref{'knoten'} . " " . " " . $hilfsstring;

  # suche rechts (rekursiv)
  if ( defined( $$ref{'rechts'} ) ) { polnisch( $$ref{'rechts'} ); }

  # suche links (rekursiv)
  if ( defined( $$ref{'links'} ) ) { polnisch( $$ref{'links'} ); }

  # Ausgabe des Knotens
  return $hilfsstring;
}

sub upn {
  my $funktionsausdruck = shift;
  my @ergebnis = regelwerkupn( tokenizer($funktionsausdruck) );
  my $funktion;
  $funktion = polnisch( $ergebnis[0]{'Baum'} );
  $funktion =~ s/\*/mul/g;
  $funktion =~ s/\+/add/g;
  $funktion =~ s/minus/sub/g;
  $funktion =~ s/\//div/g;
  $funktion =~ s/\:/div/g;
  $funktion =~ s/\^/exp/g;
  $funktion =~ s/pi/3.14159/g;

  if ( scalar @ergebnis != 1 ) {
    printf("Fehler \n");
    ausgabeliste(@ergebnis);
  }
  else {
    $hilfsstring = "";
    return $funktion;
  }
}

sub dreidimensional {
  my ( $funktion, $variable ) = @_ ;

  my $gefunden;
  $gefunden = 0;

  if ( $funktion =~ /$variable/ ) { return 1; }
  else { return 0; }
}

1;

```

873
 874
 875
 876
 877
 878
 879
 880
 881
 882
 883
 884
 885
 886
 887
 888
 889
 890
 891
 892
 893
 894
 895
 896
 897
 898
 899
 900
 901
 902
 903
 904
 905
 906
 907
 908
 909
 910
 911
 912
 913
 914
 915
 916
 917
 918
 919
 920
 921
 922
 923

A.8. Das Modul für Gitter

Listing A.8: gitter.pm

```

#####
#Modul fuer kariertes Papier
#Letzte Aenderung: 12.08.2011

```

1
 2
 3

```
#####
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;

my $gittersetzen = 0;
my $matheheftsetzen = 0;

sub setgitter {
    my ($wert) = @_;
    if ( $gittersetzen == 3 ) {
        if ( $wert == 1 ) { $gittersetzen = $gittersetzen; }
        else { $gittersetzen = $wert; }
    }
    else { $gittersetzen = $wert; }
}

sub getgitter { return $gittersetzen; }

sub setmatheheft {
    my ($wert) = @_;
    if ( $matheheftsetzen == 3 ) {
        if ( $wert == 1 ) { $matheheftsetzen = $matheheftsetzen; }
        else { $matheheftsetzen = $wert; }
    }
    else { $matheheftsetzen = $wert; }
    setgitter (2)
}

sub getmatheheft { return $matheheftsetzen; }

sub kariert {
    my (
        $layer,      $farbe,      $linienstaerke, $linienart ,
        $xminkariert, $yminkariert, $xmaxkariert,  $ymaxkariert,
        $abstandx,   $dickx,      $ganzdickx,    $abstandy,
        $dicky,      $ganzdicky
    ) = @_;
    my (
        $xminersatz,  $xmaxersatz, $yminersatz,
        $ymaxersatz,  $dickersatzx, $ganzdickersatzx,
        $abstandxersatz, $dickersatzy, $ganzdickersatzy,
        $abstandyersatz
    );

    settexthier ("");
    setgrossskalierung( $xmaxkariert - $xminkariert,
        $ymaxkariert - $yminkariert );

    setachsengraph( $xminkariert, $yminkariert, $xmaxkariert, $ymaxkariert );
    setmaxberechnung(0);

    if ( $abstandx == 0 ) {
        $abstandx = aufrunden( $xmaxkariert * 10000, 1 ) / 10000;
        $abstandx = schoenetabelle( $abstandx * 10000 ) / 10000;
    }
    if ( $abstandy == 0 ) {
```

```

$abstandy = aufrunden( $maxkariert * 10000, 1 ) / 10000;
$abstandy = schoenetabelle( $abstandy * 10000 ) / 10000;

}
if ( $dickx == 0 ) {
    $dickersatzx = $abstandx;
    $abstandxersatz = 0;
}
else {
    $dickersatzx = $dickx;
    $abstandxersatz = $abstandx;
}
if ( $dicky == 0 ) {
    $dickersatzy = $abstandy;
    $abstandyersatz = 0;
}
else {
    $dickersatzy = $dicky;
    $abstandyersatz = $abstandy;
}

$xminersatz = $xminkariert / $abstandx;
$xmaxersatz = $maxkariert / $abstandx;
$yminersatz = $yminkariert / $abstandy;
$ymaxersatz = $maxkariert / $abstandy;

paarohneminmax( getschnitt("x"), getschnitt("y") );
paarohneminmax( $xminersatz, $yminersatz );
paarohneminmax( $xmaxersatz, $ymaxersatz );
schreibeinarray(
    $layer,
    sprintf(
"\psgrid[xunit=%g, yunit=%g, gridwidth=%gpt, gridcolor=%s, griddots=0, gridlabels=0pt,
subgriddiv=0, subgridwidth=0pt, subgridcolor=%s, subgriddots=0]%%s%%s",
    $abstandx, $abstandy, $linienstaerke,
    $farbe, $farbe, gettexthier ()
    )
);
settexthier ("");

#Die zweite dicke Linie wird gezeichnet
if ( $dickersatzx == 0 ) {
    $xminersatz = $xminkariert;
    $xmaxersatz = $maxkariert;
}
else {
    if ( $dickersatzx > $maxkariert ) {
        $dickersatzx = 0;
        $xminersatz = $xminkariert;
        $xmaxersatz = $maxkariert;
    }
    else {
        $xminersatz = $xminkariert / $dickersatzx;
        $xmaxersatz = $maxkariert / $dickersatzx;
    }
}
if ( $dickersatzy == 0 ) {
    $yminersatz = $yminkariert;
    $ymaxersatz = $maxkariert;
}
else {

```

if (\$dickersatzy > \$ymaxkariert) {	125
\$dickersatzy = 0;	126
\$yminersatz = \$yminkariert;	127
\$ymaxersatz = \$ymaxkariert;	128
}	129
else {	130
\$yminersatz = \$yminkariert / \$dickersatzy;	131
\$ymaxersatz = \$ymaxkariert / \$dickersatzy;	132
}	133
}	134
if ((\$dickx != 0) (\$dicky != 0)) {	135
drei(	136
getschnitt("x"), getschnitt("y"),	137
\$xminersatz, \$yminersatz,	138
\$xmaxersatz, \$ymaxersatz	139
);	140
	141
schreibeinarray(	142
\$layer,	143
sprintf(	144
""\psgrid[xunit=%g, yunit=%g, gridwidth=%gpt, gridcolor=%s!100, griddots=0, gridlabels=0pt,	145
subgriddiv=0, subgridwidth=0pt, subgridcolor=%s,	
subgriddots=0]%%s%% ",	146
\$dickersatzx, \$dickersatzy, \$linienstaerke * 2, \$farbe, \$farbe,	147
gettexthier ()	148
)	149
);	150
	151
}	152
	153
# Die ganz dicke Linie wird gezeichnet	154
if (\$ganzdickx == 0) {	155
\$ganzdickersatzx = \$abstandx;	156
\$abstandxersatz = 0;	157
}	158
else {	159
\$ganzdickersatzx = \$ganzdickx;	160
\$abstandxersatz = \$abstandx;	161
}	162
if (\$ganzdicky == 0) {	163
\$ganzdickersatzy = \$abstandy;	164
\$abstandyersatz = 0;	165
}	166
else {	167
\$ganzdickersatzy = \$ganzdicky;	168
\$abstandyersatz = \$abstandy;	169
}	170
if (\$ganzdickersatzx == 0) {	171
\$xminersatz = \$xminkariert;	172
\$xmaxersatz = \$xmaxkariert;	173
}	174
else {	175
if (\$ganzdickersatzx > \$xmaxkariert) {	176
\$ganzdickersatzx = 0;	177
\$xminersatz = \$xminkariert;	178
\$xmaxersatz = \$xmaxkariert;	179
}	180
else {	181
\$xminersatz = \$xminkariert / \$ganzdickersatzx;	182
\$xmaxersatz = \$xmaxkariert / \$ganzdickersatzx;	183
}	184

```

}
if ( $ganzdickersatzy == 0 ) {
    $yminersatz = $yminkariert;
    $ymaxersatz = $ymaxkariert;
}
else {
    if ( $ganzdickersatzy > $ymaxkariert ) {
        $ganzdickersatzy = 0;
        $yminersatz = $yminkariert;
        $ymaxersatz = $ymaxkariert;
    }
    else {
        $yminersatz = $yminkariert / $ganzdickersatzy;
        $ymaxersatz = $ymaxkariert / $ganzdickersatzy;
    }
}
if ( ( $ganzdickx != 0 ) || ( $ganzdicky != 0 ) ) {
    drei(
        getschnitt("x"), getschnitt("y"),
        $xminersatz, $yminersatz,
        $xmaxersatz, $ymaxersatz
    );
    schreibeinarray(
        $layer,
        sprintf(
            "\psgrid[xunit=%g, yunit=%g, gridwidth=%gpt, gridcolor=%s!100, griddots=0, gridlabels=0pt,
            subgriddiv=0, subgridwidth=0pt, subgridcolor=%s, subgriddots=0]%s%%%",
            $ganzdickersatzx, $ganzdickersatzy, $linienstaerke * 3,
            $farbe, $farbe, gettexthier ()
        )
    );
}
setmaxberechnung(1);
settexthier ("");

minmax(
    $xminkariert - $dickx,
    $yminkariert - $dicky,
    $xmaxkariert + $dickx,
    $ymaxkariert + $dicky
);
minmax(
    $xminkariert + $dickx,
    $yminkariert + $dicky,
    $xmaxkariert - $dickx,
    $ymaxkariert - $dicky
);
}

sub grid {
    my (
        $layer, $farbe, $linienstaerke, $linienart,
        $xmingrid, $ymingrid, $xmaxgrid, $ymaxgrid,
        $abstandx, $dickx, $ganzdickx, $abstandy,
        $dicky, $ganzdicky
    ) = @_ ;

    drei(
        getschnitt("x"),

```

```

185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244

```

```

    getschnitt("y"),
    $xmingrid / $abstandx,
    $ymingrid / $abstandy,
    $xmaxgrid / $abstandx,
    $ymaxgrid / $abstandy
);
setgrossskalierung( $xmaxgrid - $xmingrid, $ymaxgrid - $ymingrid );
schreibeinarray(
    $layer,
    sprintf(
"\psgrid[xunit=%g, yunit=%g, gridwidth=%gpt, gridcolor=%s, griddots=%g, gridlabels=0pt,
    gridlabelcolor=%s, subgriddiv=0, subgriddots=0]%s%% ",
        $abstandx, $abstandy, $linienstaerke, $farbe,
        $dickx, $farbe, gettexthier ()
    )
);
settexthier ("");
}

sub millimeterpapier {
    my ( $layer, $farbe, $xminheft, $yminheft, $xmaxheft, $ymaxheft ) = @_;
    kariert(
        $layer, $farbe, 0.25, "solid", $xminheft, $yminheft,
        $xmaxheft, $ymaxheft, 0.1, 0.5, 1, 0.1,
        0.5, 1
    );
}

sub matheheft {
    my ( $layer, $farbe, $xminheft, $yminheft, $xmaxheft, $ymaxheft ) = @_;
    kariert(
        $layer, $farbe, 0.5, "solid", $xminheft, $yminheft,
        $xmaxheft, $ymaxheft, 0.5, 0, 0, 0.5,
        0, 0
    );
}
1;

```

A.9. Das Modul für Graphen

Listing A.9: graphen.pm

```

#####
# Modul fuer allgemeine Funktionen des Plotter – Skriptes in Pstricks
#
# Letzte Aenderung: 13.09.2011
#####
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;
my ( $zahler, $extern, $dateiname );

sub setextern {
    my ($ext) = @_;
    $extern = $ext;
}

```

sub getextern { return \$extern; }	17
	18
my (\$xminachse, \$xmaxachse, \$yminachse, \$ymaxachse, \$xsetzen1, \$xsetzen2,	19
\$ysetzen1, \$ysetzen2);	20
	21
\$xminachse = \$xmaxachse = \$yminachse = \$ymaxachse = \$xsetzen1 = \$xsetzen2 =	22
\$ysetzen1 = \$ysetzen2 = 0;	23
	24
sub achsengraph {	25
my (\$x1, \$y1) = @_;	26
setachsengraph(\$x1, \$y1, \$x1, \$y1);	27
}	28
	29
sub setachsengraphrueck {	30
\$xminachse=\$xmaxachse=\$yminachse=\$ymaxachse=0;	31
}	32
	33
sub setachsengraph {	34
my (\$xmin, \$ymin, \$xmax, \$ymax) = @_;	35
	36
	37
if ((istzahl(\$xmin) == 1)	38
&& (istzahl(\$ymin) == 1)	39
&& (istzahl(\$xmax) == 1)	40
&& (istzahl(\$ymax) == 1))	41
{	42
	43
	44
if (\$xsetzen1 == 0) {	45
\$xminachse = \$xmin;	46
\$xsetzen1 = 1;	47
}	48
else { \$xminachse = min(\$xminachse, \$xmin); }	49
if (\$xsetzen2 == 0) {	50
\$xmaxachse = \$xmax;	51
\$xsetzen2 = 1;	52
}	53
else { \$xmaxachse = max(\$xmaxachse, \$xmax); }	54
if (\$ysetzen1 == 0) {	55
\$yminachse = \$ymin;	56
\$ysetzen1 = 1;	57
}	58
else { \$yminachse = min(\$yminachse, \$ymin); }	59
if (\$ysetzen2 == 0) {	60
\$ymaxachse = \$ymax;	61
\$ysetzen2 = 1;	62
}	63
else { \$ymaxachse = max(\$ymaxachse, \$ymax); }	64
}	65
}	66
	67
sub getachsengraph {	68
my (\$text) = @_;	69
if (\$text eq "xmin") { return \$xminachse; }	70
if (\$text eq "ymin") { return \$yminachse; }	71
if (\$text eq "xmax") { return \$xmaxachse; }	72
if (\$text eq "ymax") { return \$ymaxachse; }	73
}	74
	75
\$dateiname = 0;	76
	77

```

## Zeichnen von Funktionen
sub graph {
my ( $layer, $func, $x1, $y1, $z1, $x2, $y2, $z2, $linienart ,
    $linienstaerke, $farbe, $parameter )
    = @_ ;

my ( $xmin, $xmax, $ymin, $ymax, $parameterweiter, $x, $y, $z, $i, $j ,
    $jzahler, $abstandx, $abstandy, $dx, $dy, $n, $xzeichnen, $yzeichnen,
    $innen, $aussen, $wechsel, $wechselhilf1, $wechselhilf2, $start );

$parameterweiter = "";
$xzeichnen      = 1;
$yzeichnen      = 0;
$innen          = $aussen = 0;
$wechselhilf1   = $wechselhilf2 = 0;
$wechsel        = 1;
$start          = 0;

settexthier ("" );

if ( dreidimensional( $func, "y" ) == 0 ) {

    setachsengraph( $x1, $y1, $x2, $y2 );
    minmax( $x1, $y1, $x2, $y2 );
    if ( $y1 != $y2 ) {

        for ( $j = $x1 ; $j <= $x2 ; $j = $j + ( $x2 - $x1 ) / 100 ) {
        if ( ( calcfunk( $func, "x", $j ) <= $y2 )
            && ( calcfunk( $func, "x", $j ) >= $y1 ) )
        {

            if ( $start == 0 ) {
            $wechsel      = 1;
            $wechselhilf1 = $wechsel;
            $start        = 1;
            }

            if ( $wechsel == $wechselhilf1 ) {$wechsel = $wechsel;}
            else {$wechsel++;}
            $wechselhilf1 = $wechsel;
            $innen++;

        }

        else {

            if ( $start == 0 ) {
            $wechsel      = 1;
            $wechselhilf2 = $wechsel;
            $start        = 1;
            }

            if ( $wechsel == $wechselhilf2 ) {$wechsel = $wechsel;}
            else {$wechsel++;}
            $wechselhilf2 = $wechsel;

            $aussen++;

        }
    }
}
}

```

78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138

```

    $xmin = calcxmin( $func, $x1, $x2, $y1, $y2 );
    $xmax = calcxmax( $func, $x1, $x2, $y1, $y2 );
    if ( $xmin == $xmax ) {
        $xmin = $x1;
        $xmax = $x2;
    }
}
else {
    $xmin = $x1;
    $xmax = $x2;
}

if ( getknoten() == 1 ) {
    if ( get3dzeichnen() == 1 ) { }
    else {
        knotensetzen( "\\pnode", $xmin, calcfunk( $func, "x", $xmin ),
            "" );
        knotensetzen( "\\pnode", $xmax, calcfunk( $func, "x", $xmax ),
            "" );
    }
}

# kommentar($layer,$wechsel);

# kommentar($layer,aufrunden( ( schoenetabelle ( ($x2-$x1) * 10000 ) / 10000), 1));
# kommentar($layer, schoenetabelle ( ($x2-$x1) * 10000 ) / 10000);

if ( $wechsel <= 3 ) {

    if ( ( $func =~ /cos/ )
        || ( $func =~ /sin/ )
        || ( $func =~ /tan/ ) )
    {
        if ( $parameter =~ /algebraic/ ) { $parameterweiter = ""; }
        else { $parameterweiter = "algebraic, "; }
    }
    $parameterweiter = $parameterweiter . $parameter;

    if ( $parameterweiter =~ /algebraic/ ) {
        klammer($xmin);
        klammer($xmax);
        klammer($func);
        befehl erzeugen( $layer, "psplot", $farbe, $linienart,
            $linienstaerke, $parameterweiter, "", gettexthier() );
    }
    else {
        klammer($xmin);
        klammer($xmax);
        klammer( upn($func) );
        befehl erzeugen( $layer, "psplot", $farbe, $linienart,
            $linienstaerke, $parameterweiter, "", gettexthier() );
    }
    minmax( $xmin, $y1, $xmax, $y2 );
}
else {
    settexthier( "" );
    for ( $j = $x1 ; $j <= $x2 ; $j = $j + aufrunden( ( schoenetabelle( ( $x2 - $x1 ) * 10000 ) /
        10000 ), 1 ) / 10 ) {
        if ( ( calcfunk( $func, "x", $j ) <= $y2 )
            && ( calcfunk( $func, "x", $j ) >= $y1 ) )
        {

```

```

        paarohneminmax( nachkommastellen( $j, 3 ),
            nachkommastellen( calcfunk( $func, "x", $j ), 3 ) );
    }
    else {
        liniekopf( $layer, $linienart, $linienstaerke, $farbe,
            $parameterweiter, "", gettexthier() );
        settexthier( "" );
    }
}

# listplot ( $layer, "", $linienart, $linienstaerke, $farbe, gettexthier() );
}
}
else {
    setachsengraph3d( $x1, $y1, $z1, $x2, $y2, $z2 );
    if ( $z1 == $z2 ) {
        $ymin = $y1;
        $ymax = $y2;
        if ( $parameter == ~/xPlotpoints=([0-9\.]+)/ ) {
            $n = $1;
            $dx = ( $x2 - $x1 ) / $n;
        }
        else { $dx = ( $x2 - $x1 ) / 25; }
        if ( $parameter == ~/yPlotpoints=([0-9\.]+)/ ) {
            $n = $1;
            $dy = ( $y2 - $y1 ) / $n;
        }
        else { $dy = ( $y2 - $y1 ) / 25; }

        for ( $i = $y1 ; $i <= $y2 ; $i = $i + $dy ) {
            for ( $j = $x1 ; $j <= $x2 ; $j = $j + $dx ) {
                $z = calcfunk3d( $func, "x", $j, "y", $i );
                setachsengraph3d( $j, $i, $z, $j, $i, $z );
                minmax3d( $j, $i, $z, $j, $i, $z );
            }
        }
    }
    else {
        $ymin = calcymin3d( $func, $x1, $x2, $y1, $y2, $z1, $z2 );
        $ymax = calcymax3d( $func, $x1, $x2, $y1, $y2, $z1, $z2 );
        setachsengraph3d( $x1, $y1, $z1, $x2, $y2, $z2 );
        minmax3d( $x1, $y1, $z1, $x2, $y2, $z2 );
    }

    set3dzeichnen(1);

    if ( get3dpstricks() == 1 ) {
        $parameterweiter =
            "plotstyle=curve, " . $parameterweiter . $parameter;

        if ( $parameterweiter == ~/algebraic/ ) {

            paar( $x1, $x2 );
            paar( $ymin, $ymax );
            klammer($func);
            befehlerezeugen( $layer, "psplotThreeD", $farbe, $linienart,
                $linienstaerke, $parameterweiter, "", gettexthier() );
        }
    }
}

```

```

else {
    paar( $x1, $x2 );
    paar( $ymin, $ymax );
    klammer( upn($func) );
    befehl erzeugen( $layer, "psplotThreeD", $farbe, $linienart,
        $linienstaerke, $parameterweiter, "", gettexthier() );
    }
    minmax3d( $x1, $ymin, $z1, $x1, $ymax, $z2 );
    settexthier( "" );
}
else {
    if ( $parameter == ~/xPlotpoints=([-0-9\.]+)/ ) {
        $n = $1;
        $parameter == s/xPlotpoints=$n//g;
        $dx = ( $x2 - $x1 ) / $n;
    }
    else { $dx = ( $x2 - $x1 ) / 25; }
    if ( $parameter == ~/yPlotpoints=([-0-9\.]+)/ ) {
        $n = $1;
        $parameter == s/yPlotpoints=$n//g;
        $dy = ( $ymax - $ymin ) / $n;
    }
    else { $dy = ( $ymax - $ymin ) / 25; }
    $parameterweiter = $parameterweiter . $parameter;
    settexthier( "" );

    if ( $parameterweiter == ~/drawStyle=xLines/ ) {
        $parameterweiter == s/drawStyle=xLines//g;
        $parameterweiter == s/drawStyle=xLines//g;
        $zeichnen = 1;
    }
    if ( $parameterweiter == ~/drawStyle=yLines/ ) {
        $parameterweiter == s/drawStyle=yLines//g;
        $parameterweiter == s/drawStyle=yLines//g;
        $yzeichnen = 1;
        $xzeichnen = 0;
    }
    if ( $parameterweiter == ~/drawStyle=xyLines/ ) {
        $parameterweiter == s/drawStyle=xyLines//g;
        $parameterweiter == s/drawStyle=xyLines//g;
        $yzeichnen = $xzeichnen = 1;
    }
    if ( $parameterweiter == ~/drawStyle=yxLines/ ) {
        $parameterweiter == s/drawStyle=yxLines//g;
        $parameterweiter == s/drawStyle=yxLines//g;
        $yzeichnen = $xzeichnen = 1;
    }
    $parameterweiter == s/hiddenLine=true//g;
    $parameterweiter == s/hiddenLine=false//g;
    $parameterweiter == s/hiddenLine=true//g;
    $parameterweiter == s/hiddenLine=false//g;
    $parameterweiter == s/hiddenLine//g;
    $parameterweiter == s/hiddenLine//g;
    $n = 0;

    if ( $zeichnen == 1 ) {
for ( $i = $ymin ; $i <= $ymax ; $i = $i + $dy ) {
    for ( $j = $x1 ; $j <= $x2 ; $j = $j + $dx ) {
        if ( $z1 == $z2 ) {

```

```

    $z = calcfunk3d( $func, "x", $j, "y", $i );
    tripel (
nachkommastellen( $j, 4 ),
nachkommastellen( $i, 4 ),
nachkommastellen( $z, 4 )
);
    $n++;
}
else {
    if ( ( calcfunk3d( $func, "x", $j, "y", $i ) > $z1 )
&& ( calcfunk3d( $func, "x", $j, "y", $i ) <
    $z2 ) )
    {
        $z = calcfunk3d( $func, "x", $j, "y", $i );
        tripel (
            nachkommastellen( $j, 4 ),
            nachkommastellen( $i, 4 ),
            nachkommastellen( $z, 4 )
        );
        $n++;
    }
}

}

if ( $n > 1 ) {
liniekopf( $layer, $linienart, $linienstaerke, $farbe,
    $parameterweiter, "", gettexthier() );
}
settexthier( "" );
$n = 0;
}
}

if ( $zeichnen == 1 ) {
for ( $j = $x1 ; $j <= $x2 ; $j = $j + $dx ) {
    for ( $i = $ymin ; $i <= $ymax ; $i = $i + $dy ) {
        if ( $z1 == $z2 ) {
            $z = calcfunk3d( $func, "x", $j, "y", $i );
            tripel (
                nachkommastellen( $j, 4 ),
                nachkommastellen( $i, 4 ),
                nachkommastellen( $z, 4 )
            );
            $n++;
        }
        else {
            if ( ( calcfunk3d( $func, "x", $j, "y", $i ) > $z1 )
&& ( calcfunk3d( $func, "x", $j, "y", $i ) <
            $z2 ) )
            {
                $z = calcfunk3d( $func, "x", $j, "y", $i );
                tripel (
                    nachkommastellen( $j, 4 ),
                    nachkommastellen( $i, 4 ),
                    nachkommastellen( $z, 4 )
                );
                $n++;
            }
        }
    }
}
}

```

```

}
}
if ( $n > 1 ) {

liniekopf(
    $layer,                                $linienart ,
    $linienstaerke,                        $farbe,
    "plotstyle=curve" . $parameterweiter, "",
    gettexthier ()
);
}
settexthier ("");
$n = 0;
}
}

}
}
settexthier ("");

$innen = $ausen = 0;
$wechselhilf1 = 0;
$wechselhilf2 = 0;
$wechsel      = 1;
$start        = 0;

}

sub graphrot {

my ( $layer, $func, $x1, $y1, $x2, $y2, $schritte, $farbe, $fuellart ,
    $fuellfarbe, $linienart, $linienstaerke, $parameter ) = @_ ;

setachsengraph( $x1, min( -$y1, -$y2 ), $x2, max( $y1, $y2 ) );
minmax( $x1, $y1, $x2, $y2 );
my ( $xmin, $xmax );

if ( $y1 == $y2 ) {
    $xmin = $x1;
    $xmax = $x2;
}
else {
    $xmin = calcxmin( $func, $x1, $x2, $y1, $y2 );
    $xmax = calcxmax( $func, $x1, $x2, $y1, $y2 );
}
settexthier ("");

paar( $xmin, $xmax );
klammer($schritte);
klammer( upn($func) );
befehlfuellungerzeugen(
    $layer,      "psVolume", $farbe,      $fuellart ,
    $fuellfarbe, $linienart, $linienstaerke, $parameter,
    "",          gettexthier ()
);

settexthier ("");

}

```

### Folgen zeichnen	443
	444
sub folge {	445
my (\$layer, \$func, \$x1, \$x2, \$dx, \$groesse, \$linienart, \$linienstaerke,	446
\$farbe, \$parameter) = @_;	447
	448
setachsengraph(\$x1, calcfunk(\$func, "x", \$x1),	449
\$x2, calcfunk(\$func, "x", \$x2));	450
my \$anzahlpunkte = aufrunden(abs(\$x2 - \$x1) / \$dx + 1, 1);	451
	452
my \$parameterweiter =	453
"showpoints=true, plotstyle=dots, dotstyle=o, plotpoints="	454
. \$anzahlpunkte	455
. ", dotsize="	456
. \$groesse . "pt";	457
	458
\$parameterweiter = \$parameterweiter . \$parameter;	459
graph(\$layer, \$func, \$x1, 0, 0, \$x2, 0, 0, \$linienart, \$linienstaerke,	460
\$farbe, \$parameterweiter);	461
}	462
	463
### Parametrisierte Funktionen	464
	465
sub parametricplot {	466
my (\$layer, \$func1, \$func2, \$func3, \$t1, \$u1, \$t2, \$u2, \$linienart,	467
\$linienstaerke, \$farbe, \$parameter) = @_;	468
	469
my \$parameterweiter = "plotpoints=200,";	470
\$parameterweiter = \$parameterweiter . "" . "" . \$parameter;	471
	472
my (\$i, \$j, \$n);	473
calcpaminmax(\$func1, \$func2, \$func3, \$t1, \$t2, \$u1, \$u2);	474
	475
	476
	477
settexthier ("");	478
	479
if (\$func3 eq "") {	480
klammer(\$t1);	481
klammer(\$t2);	482
if (\$parameterweiter =~ /algebraic/) {	483
klammer(\$func1 . " " . \$func2);	484
}	485
else {	486
klammer(upn(\$func1) . upn(\$func2));	487
}	488
	489
befehlerzeugen(\$layer, "parametricplot", \$farbe, \$linienart,	490
\$linienstaerke, \$parameterweiter, "", gettexthier ());	491
}	492
else {	493
	494
if (\$u1 == \$u2) {	495
paarohneminmax(\$t1, \$t2);	496
if (\$parameterweiter =~ /algebraic/) {	497
klammer(\$func1 . " " . \$func2 . " " . \$func3);	498
}	499
else {	500
klammer(upn(\$func1) . upn(\$func2) . upn(\$func3));	501
}	502
}	503

```

else {
    paarohneminmax( $t1, $t2 );
    paarohneminmax( $u1, $u2 );
    if ( $parameterweiter =~ /algebraic/ ) {
        klammer( $func1 . "|" . $func2 . "|" . $func3 );
    }
    else {
        klammer( upn($func1) . upn($func2) . upn($func3) );
    }
}

set3dzeichnen(1);
set3dpstricks(1);
befehlerzeugen( $layer, "parametricplotThreeD", $farbe, $linienart,
    $linienstaerke, $parameterweiter, "", gettexthier() );
}

settexthier( "");
}

### Zeichnen von allgemeinen Daten
sub fileplot {
my ( $layer, $datei, $linienart, $linienstaerke, $farbe, $parameter ) = @_ ;

    settexthier( "");
    setachsengraph( 0, 0, 0, 0 );
    setachsengraph3d( 0, 0, 0, 0, 0, 0 );
    klammer($datei);
    my $parameterweiter = "";

    $parameterweiter = $parameterweiter . " " . " " . $parameter;

    if ( get3dpstricks() == 1 ) {
        befehlzeugen( $layer, "fileplotThreeD", $farbe, $linienart,
            $linienstaerke, $parameterweiter, "", gettexthier() );
    }
    else {
        befehlzeugen( $layer, " fileplot ", $farbe, $linienart, $linienstaerke,
            $parameterweiter, "", gettexthier() );
    }
    settexthier( "");
}

sub listplot {
my ( $layer, $datei, $linienart, $linienstaerke, $farbe, $parameter ) = @_ ;
setachsengraph( 0, 0, 0, 0 );
setachsengraph3d( 0, 0, 0, 0, 0, 0 );
my $parameterweiter = "";

    $parameterweiter = $parameterweiter . " " . " " . $parameter;

    if ( get3dpstricks() == 1 ) {
        befehlzeugen( $layer, "listplotThreeD", $farbe, $linienart,
            $linienstaerke, $parameterweiter, "", "{\\datenlesen}" );
    }
    else {
        befehlzeugen( $layer, " listplot ", $farbe, $linienart, $linienstaerke,
            $parameterweiter, "", "{\\datenlesen}" );
    }
}

```

```

}
}

sub daten {
my ( $layer, $datei, $linienart, $linienstaerke, $farbe, $parameter ) = @_;
setachsengraph( 0, 0, 0, 0 );
setachsengraph3d( 0, 0, 0, 0, 0, 0 );
my $parameterweiter = "";

$parameterweiter = $parameterweiter . "" . "" . $parameter;

if ( get3dpstricks() == 1 ) {
    befehlerzeugen( $layer, "dataplotThreeD", $farbe, $linienart,
        $linienstaerke, $parameterweiter, "", "{\\datenlesen}" );
}
else {
    befehlerzeugen( $layer, "dataplot", $farbe, $linienart, $linienstaerke,
        $parameterweiter, "", "{\\datenlesen}" );
}
}

# Ableitungsfunktion zeichnen
sub ableitung {
my ( $layer, $func, $grad, $x1, $y1, $x2, $y2, $linienart, $linienstaerke,
    $farbe, $parameter )
    = @_;

setachsengraph( $x1, $y1, $x2, $y2 );
my ( $xmin, $xmax, $parameterweiter, $text );
settexthier ( "" );
$parameterweiter = " algebraic, ";
$parameterweiter = $parameterweiter . " " . " " . $parameter;
$xmin = sprintf( "%0.4f", calcablxmin( $func, $x1, $x2, $y1, $y2 ) );
$xmax = sprintf( "%0.4f", calcablxmax( $func, $x1, $x2, $y1, $y2 ) );
klammer($xmin);
klammer($xmax);
$text = "Derive(" . $grad . ", " . $func . ")";
klammer($text);
befehlerzeugen( $layer, "psplot", $farbe, $linienart, $linienstaerke,
    $parameterweiter, "", gettexthier() );

minmax( $x1, $y1, $x2, $y2 );
settexthier ( "" );
}

# Stammfunktion zeichnen
sub stammfunktion {
my ( $layer, $func, $x1, $y1, $x2, $y2, $linienart, $linienstaerke, $farbe,
    $verschiebungswert, $parameter )
    = @_;
setachsengraph( $x1, $y1, $x2, $y2 );
settexthier ( "" );
my (
    @xpunkte, @ypunkte, $schritt,      $i,
    @yintegral, $x,      $y,      $integral,
    $breite,    $element, $datei,    $merkex,
    $merkey,    $weite,   $schrittzahl, $schrittweite,
    $punkte,    $nullwert
    );

```

```

$dateiname++;
$weite = 3000;
$schrittzahl = ceil ( $x2 - $x1 ) * $weite;
$schrittweite = ceil ( ceil ( $x2 - $x1 ) / ( getskalierung("x") / 2 ) );
$punkte = 0;
$integral = 0;
$nullwert = 0;

# Integral wird berechnet
$breite = ( $x2 - $x1 ) / ($schrittzahl);
for ( $schritt = 0 ; $schritt < $schrittzahl + 1 ; $schritt++ ) {
    $x = $x1 + ( $x2 - $x1 ) * ( $schritt / $schrittzahl );
    $y = calcfunk( $func, "x", $x );
    $element = ( $breite * $y );
    $integral += $element;
    $nullwert = $integral if ( $x <= 0 );
    $yintegral[ $schritt ] = $integral;
}
$nullwert -= $verschiebungswert;

# oeffnedei ( $dateiname );

# Schleife zum zeichnen der Funktion
for (
    $schritt = 0 ;
    $schritt < $schrittzahl ;
    $schritt = $schritt + $schrittweite
)
{
    $x = nachkommastellen(
        $x1 + ( $x2 - $x1 ) * ( $schritt / ( $schrittzahl - 1 ) ), 3 );
    $y = nachkommastellen( $yintegral[ $schritt ] - $nullwert, 3 );

    # liegen die Funktionswerte innerhalb von ymin und ymax?
    if ( ( $y >= $y1 ) && ( $y <= $y2 ) ) {
        $xpunkte[ $punkte ] = $x;
        $ypunkte[ $punkte ] = $y;
        $punkte++;
    }
    else {

        # wenn nicht wird das bisherige Array ausgelesen und gezeichnet
        if ( $punkte > 1 ) {
            #schreibedaten ( $dateiname, $xpunkte[0], $ypunkte[0] );
            paar($xpunkte[0], $ypunkte[0]);
            for ( $i = 0 ; $i < $punkte - 2 ; $i = $i + 2 ) {

                if (
                    calcaenderungsrate(
                        $xpunkte[ $i ],      $ypunkte[ $i ],
                        $xpunkte[ $i + 1 ], $ypunkte[ $i + 1 ]
                    ) != calcaenderungsrate(
                        $xpunkte[ $i + 1 ],
                        $ypunkte[ $i + 1 ],
                        $xpunkte[ $i + 2 ],
                        $ypunkte[ $i + 2 ]
                    )
                )
                {
                    #schreibedaten (

```

```

# $dateiname,
# $xpunkte[ $i + 1 ],
# $ypunkte[ $i + 1 ]
#);
paar($xpunkte[$i+1], $ypunkte[$i+1]);

}
}
# schreibedaten (
# $dateiname,
# $xpunkte[ $i - 1 ],
# $ypunkte[ $i - 1 ]
#);

paar($xpunkte[$i+1], $ypunkte[$i-1]);

}
$punkte = 0;
}
}

# Restarray wird ausgelesen
if ( $punkte > 1 ) {
for ( $i = 0 ; $i < $punkte - 2 ; $i = $i + 2 ) {
if (
calcaenderungsrate(
$xpunkte[$i], $ypunkte[$i],
$xpunkte[ $i + 1 ], $ypunkte[ $i + 1 ]
) != calcaenderungsrate(
$xpunkte[ $i + 1 ],
$ypunkte[ $i + 1 ],
$xpunkte[ $i + 2 ],
$ypunkte[ $i + 2 ]
)
)
{
# schreibedaten (
# $dateiname,
# $xpunkte[ $i + 1 ],
# $ypunkte[ $i + 1 ]
#);
paar($xpunkte[$i+1], $ypunkte[$i+1]);
}
}
}
# schliessedatei ($dateiname);
# readdata( $layer, $dateiname );
liniekopf( $layer, $linienart, $linienstaerke, $farbe,
$parameter, "", gettexthier() );
# daten( $layer, $dateiname, $linienart, $linienstaerke, $farbe, $parameter );
$punkte = 0;
settexthier("");

}

# Zeichnen der Flaeche unter einer Funktion
sub integral {
my (
$layer, $func, $x1, $y1,
$x2, $y2, $farbe, $fuellart,

```

```

$fuellfarbe, $linienart, $linienstaerke, $parameter
) = @_;
748
749
750
setachsengraph( $x1, $y1, $x2, $y2 );
751
my ( $parameterweiter, $xmin, $xmax );
752
753
$parameterweiter = "algebraic" . $parameter;
754
755
if ( $y1 == $y2 ) {
756
    $xmin = $x1;
757
    $xmax = $x2;
758
759
}
760
else {
761
    $xmin = nachkommastellen( calcxmin( $func, $x1, $x2, $y1, $y2 ), 2 );
762
    $xmax = nachkommastellen( calcxmax( $func, $x1, $x2, $y1, $y2 ), 2 );
763
764
}
765
schreibeinarray(
766
    $layer,
767
    sprintf(
768
        "\\pscustom[fillcolor = %s]
769
    {\\gsave", $fuellfarbe
770
    )
771
    );
772
graph( $layer, $func, $xmin, $y1, 0, $xmax, $y2, 0, $linienart,
773
    $linienstaerke, $farbe, $parameter );
774
linie ( $layer, $xmax, 0, "", $xmin, 0, "", $linienart, $linienstaerke,
775
    $farbe, $parameter, "" );
776
befehlfuellungerzeugen(
777
    $layer, " fill ", $farbe, $fuellart,
778
    $fuellfarbe, $linienart, $linienstaerke, $parameter,
779
    "", ""
780
    );
781
schreibeinarray( $layer, sprintf("\\grestore") );
782
783
}
784
### Sichern von Daten
785
sub datensichern {
786
    my ($layer) = @_;
787
    schreibeinarray( $layer, sprintf("\\savedata{\\mydata}["");
788
}
789
790
sub readdata {
791
    my ($layer, $name) = @_;
792
    schreibeinarray( $layer, sprintf( "\\readdata{\\datenlesen}{%s}", $name ) );
793
}
794
795
sub setfilehandle { my ($file) = @_; }
796
my $AUSGABE;
797
798
my $filecontents = 0;
799
800
sub oeffnedei {
801
    my ($name) = @_;
802
    if ( getextern() == 0 ) {
803
        if ( $filecontents == 0 ) {
804
            usepackage( "", " filecontents " );
805
            $filecontents = 1;
806
        }
807
    }
808
    else {

```

```

schreibeinarray( 105,
    sprintf( "\n\\begin{filecontents*}{%s}", $name ) );
$zahler = 0;
settexthier ("" );
}
else {
    open( $AUSGABE, '>', $name );
    $zahler = 0;
}
}

sub schliessedatei {
my ($name) = @_ ;
if ( $extern == 0 ) {
    pstricks( 105, gettexthier() );
    pstricks( 105, "\\end{filecontents*}" );
    abstand(105);
    $zahler = 0;
}
else {
    close $AUSGABE;
    $zahler = 0;
}
}

sub schreibedaten {
my ($name, $x, $y ) = @_ ;
setachsengraph( $x, $y, $x, $y );
$zahler++;
if ( getextern() == 0 ) { paar( $x, $y ); }
else { printf( $AUSGABE "(%g,%g)", $x, $y ); }
minmax( $x, $y, $x, $y );
}

sub schreibedaten3d {
my ($name, $x, $y, $z ) = @_ ;
$zahler++;
setachsengraph3d( $x, $y, $z, $x, $y, $z );
my ($xersatz, $yersatz );
$xersatz = x3d( $x, $y, $z );
$yersatz = y3d( $x, $y, $z );
if ( getextern() == 0 ) { tripel( $x, $y, $z ); }
else {

    if ( get3dpstricks() == 1 ) {
        printf( $AUSGABE "(%g,%g,%g)", $x, $y, $z );
    }
    else { printf( $AUSGABE "(%g,%g)", $xersatz, $yersatz ); }
}

}

sub lesedaten {
my ($layer, $name) = @_ ;
my $zeile;
open( $AUSGABE, $name ) || die $name . ": $!";
while ( $zeile = <$AUSGABE> ) {
    schreibeinarray( $layer, sprintf( "%s", $zeile ) );
}
}

```

```

sub oberuntersumme {
my ( $layer, $func, $x1, $y1, $x2, $y2, $farbe, $fuellart, $fuellfarbe,
    $linienart, $linienstaerke, $n, $summen )
    = @_;
setachsengraph( $x1, $y1, $x2, $y2 );
my $farbeersatz = getfarbe();
my $fuellartersatz = getfuellart();
my $fuellfarbeersatz = getfuellfarbe();
my $linienstaerkeersatz = getlinienstaerke();
my $linienartersatz = getlinienart();
my (
    $yober, $yunter, $schritt, $xlinks,
    $xrechts, $flaecheober, $flaecheunter, $xmin,
    $xmax, $parameterweiter, $breite
    );

$parameterweiter = "";

if ( $y1 != $y2 ) {
    $xmin = calcxmin( $func, $x1, $x2, $y1, $y2 );
    $xmax = calcxmax( $func, $x1, $x2, $y1, $y2 );
}
else {
    $xmin = $x1;
    $xmax = $x2;
}

$breite = nachkommastellen( ( $xmax - $xmin ) / $n, 4 );

if ( $n > 2 ) {
    pstricks( $layer, "{" );

    einstellungen( $layer, $farbe, $fuellart, $fuellfarbe, $linienart,
        $linienstaerke, $parameterweiter );

    for ( $schritt = 0 ; $schritt < $n ; $schritt++ ) {
        $xlinks =
            nachkommastellen( $xmin + ( $xmax - $xmin ) * ( $schritt / ($n) ),
                4 );
        $xrechts = nachkommastellen(
            $xmin + ( $xmax - $xmin ) * ( ( $schritt + 1 ) / ($n) ), 4 );
        $yober = calcymax( $func, $xlinks, $xrechts );
        $yunter = calcymin( $func, $xlinks, $xrechts );
        if ( abs($yober) > abs($yunter) ) {
            rechteck( $layer, $xlinks, 0, $breite, $yober, 0, "", "", "",
                "", $linienstaerke, $parameterweiter )
            if ( ( $summen eq "ober" ) || ( $summen eq "beide" ) );
            rechteck( $layer, $xlinks, 0, $breite, $yunter, 0, "", "", "",
                "", $linienstaerke, $parameterweiter )
            if ( ( $summen eq "unter" ) || ( $summen eq "beide" ) );
        }
        else {
            rechteck( $layer, $xlinks, 0, $breite, $yober, 0, "", "", "",
                "", $linienstaerke, $parameterweiter )
            if ( ( $summen eq "ober" ) || ( $summen eq "beide" ) );
            rechteck( $layer, $xlinks, 0, $breite, $yunter, 0, "", "", "",
                "", $linienstaerke, $parameterweiter )
            if ( ( $summen eq "unter" ) || ( $summen eq "beide" ) );
        }
    }
    pstricks( $layer, "}" );
}

```

```

}
else {
  for ( $schritt = 0 ; $schritt < $n ; $schritt++ ) {
    $xlinks = $xmin + ( $xmax - $xmin ) * ( $schritt / ($n) );
    $xrechts = $xmin + ( $xmax - $xmin ) * ( ( $schritt + 1 ) / ($n) );
    $yober = calcymax( $func, $xlinks, $xrechts );
    $yunter = calcymin( $func, $xlinks, $xrechts );
    if ( abs($yober) > abs($yunter) ) {
      rechteck(
        $layer,      $xlinks,
        0,           $breite,
        $yober,      0,
        $farbe,      $fuellart,
        $fuellfarbe, $linienart,
        $linienstaerke, $parameterweiter
      ) if ( ( $summen eq "ober" ) || ( $summen eq "beide" ) );
      rechteck(
        $layer,      $xlinks,
        0,           $breite,
        $yunter,     0,
        $farbe,      $fuellart,
        $fuellfarbe, $linienart,
        $linienstaerke, $parameterweiter
      ) if ( ( $summen eq "unter" ) || ( $summen eq "beide" ) );
    }
    else {
      rechteck(
        $layer,      $xlinks,
        0,           $breite,
        $yober,      0,
        $farbe,      $fuellart,
        $fuellfarbe, $linienart,
        $linienstaerke, $parameterweiter
      ) if ( ( $summen eq "ober" ) || ( $summen eq "beide" ) );
      rechteck(
        $layer,      $xlinks,
        0,           $breite,
        $yunter,     0,
        $farbe,      $fuellart,
        $fuellfarbe, $linienart,
        $linienstaerke, $parameterweiter
      ) if ( ( $summen eq "unter" ) || ( $summen eq "beide" ) );
    }
  }
}
setfarbe($farbeersatz);
setfuellart ( $fuellartersatz );
setfuellfarbe ( $fuellfarbeersatz );
setlinienstaerke ( $linienstaerkeersatz );
setlinienart ( $linienartersatz );

}

# der Flaecheninhalt zwischen zwei Funktionen kann gezeichnet werden
sub einschluss {
my ( $layer, $func1, $func2, $x1, $y1, $x2, $y2, $schnittpunkte, $farbe,
    $fuellart, $fuellfarbe, $linienart, $linienstaerke, $parameter )
    = @_;
my ( $bereichxmin, $bereichxmax, $parameterweiter );
setachsengraph( $x1, $y1, $x2, $y2 );

```

931
 932
 933
 934
 935
 936
 937
 938
 939
 940
 941
 942
 943
 944
 945
 946
 947
 948
 949
 950
 951
 952
 953
 954
 955
 956
 957
 958
 959
 960
 961
 962
 963
 964
 965
 966
 967
 968
 969
 970
 971
 972
 973
 974
 975
 976
 977
 978
 979
 980
 981
 982
 983
 984
 985
 986
 987
 988
 989
 990
 991

```

kommentar( $layer,
    "Einschluss zwischen den Funktionen " . $func1 . " und " . $func2 );
$parameterweiter = $parameter;
if ( $schnittpunkte eq "links" ) {
    $bereichxmin =
        nachkommastellen( calcschnittlinks( $func1, $func2, $x1, $x2 ), 6 );
    $bereichxmax = nachkommastellen( $x2, 2 );
}
elseif ( $schnittpunkte eq "rechts" ) {
    $bereichxmax =
        nachkommastellen( calcschnittrechts( $func1, $func2, $x1, $x2 ), 6 );
    $bereichxmin = nachkommastellen( $x1, 2 );
}
elseif ( $schnittpunkte eq "beide" ) {
    $bereichxmin =
        nachkommastellen( calcschnittlinks( $func1, $func2, $x1, $x2 ), 6 );
    $bereichxmax =
        nachkommastellen( calcschnittrechts( $func1, $func2, $x1, $x2 ), 6 );
}
else {
    $bereichxmin = nachkommastellen( $x1, 2 );
    $bereichxmax = nachkommastellen( $x2, 2 );
}

einschlussanfang( $layer, $farbe, $fuellart, $fuellfarbe, $linienart,
    $linienstaerke, $parameter );
graph( $layer, $func1, $bereichxmin, 0, 0, $bereichxmax, 0, 0, $linienart,
    $linienstaerke, $farbe, $parameterweiter );

graph( $layer, $func2, $bereichxmax, 0, 0, $bereichxmin, 0, 0, $linienart,
    $linienstaerke, $farbe, $parameterweiter );
einschlussende( $layer, $farbe, $fuellart, $fuellfarbe, $linienart,
    $linienstaerke, $parameter );

# if ( calcfunk( $func1, "x", $bereichxmax ) != calcfunk( $func2, "x", $bereichxmax ) ) {
#
#     linie (
#         $layer, $bereichxmax, nachkommastellen( calcfunk( $func1, "x", $bereichxmax ), 6 ),
#         "", $bereichxmax, nachkommastellen( calcfunk( $func2, "x", $bereichxmax ), 6 ),
#         "", $linienart, $linienstaerke, $farbe, $parameter, ""
#     );
#
# }
#
# if ( calcfunk( $func1, "x", $bereichxmin ) != calcfunk( $func2, "x", $bereichxmin ) ) {
#     linie (
#         $layer, $bereichxmin, nachkommastellen( calcfunk( $func1, "x", $bereichxmin ), 6 ),
#         "", $bereichxmin, nachkommastellen( calcfunk( $func2, "x", $bereichxmin ), 6 ),
#         "", $linienart, $linienstaerke, $farbe, $parameter, ""
#     );
#
# }

}

sub punkteende {
my ( $layer, $farbe, $linienart, $linienstaerke ) = @_ ;
schreibeinarray( $layer, sprintf("]") );
schreibeinarray( $layer,
    sprintf("\dataplot[plotstyle=curve,showpoints=false,]");

```

erzeugen(\$layer, \$linienart, \$linienstaerke, \$farbe, "");	1053
schreibeinarray(\$layer,	1054
sprintf ("{\\mydata}", \$farbe, \$linienstaerke, \$linienart));	1055
}	1056
	1057
1;	1058

A.10. Das Modul für Tupel

Listing A.10: paar.pm

#####	1
# Modul fuer Paare, Tupel, etc Plotter –Skriptes in Pstricks	2
#	3
# VERSION 3 Arbeitsversion	4
# Letzte Aenderung: 26.08.2011	5
#####	6
use POSIX qw /floor ceil/;	7
use Math::Trig;	8
use strict ;	9
use warnings;	10
	11
my \$texthier = "";	12
	13
sub settexthier {	14
my (\$text) = @_;	15
\$texthier = \$text;	16
}	17
	18
sub gettexthier { return \$texthier; }	19
	20
sub textanhaengen {	21
my (\$text) = @_;	22
\$texthier = \$texthier . \$text;	23
}	24
	25
sub klammer {	26
my (\$x) = @_;	27
	28
my \$text = "{" . \$x . "}";	29
textanhaengen(\$text);	30
}	31
	32
sub paar {	33
my (\$x, \$y) = @_;	34
if ((\$x ne "ERROR") && (\$y ne "ERROR") && (\$x ne "") && (\$y ne "")) {	35
paarohnneminmax(\$x, \$y);	36
	37
if ((istzahl(\$x) == 1) && (istzahl(\$y) == 1)) {	38
minmax(\$x, \$y, \$x, \$y);	39
}	40
if (getknoten() == 1) {	41
if (get3dzeichnen() == 1) { }	42
else { knotensetzen("\\pnode", \$x, \$y, ""); }	43
}	44
}	45
}	46
	47
sub paarohnneminmax {	48

```

my ( $x, $y ) = @_;
my $text;

$text = sprintf( "(%s,%s)", $x, $y );

if ( $text =~ /[e!]+/ ) {
    $text = sprintf( "(%s,%s)",
        nachkommastellen( $x, 4 ),
        nachkommastellen( $y, 4 ) );
}
textanhaengen($text);
}

sub tupel {
    my ( $x1, $y1, $x2, $y2 ) = @_;
    settexthier( "" );
    paar( $x1, $y1 );
    paar( $x2, $y2 );
}

sub vier {
    my ( $x1, $y1, $x2, $y2, $x3, $y3, $x4, $y4 ) = @_;
    settexthier( "" );
    paar( $x1, $y1 );
    paar( $x2, $y2 );
    paar( $x3, $y3 );
    paar( $x4, $y4 );
}

sub drei {
    my ( $x1, $y1, $x2, $y2, $x3, $y3 ) = @_;
    settexthier( "" );
    paarohneminmax( $x1, $y1 );
    paarohneminmax( $x2, $y2 );
    paarohneminmax( $x3, $y3 );
}

sub tripel {
    my ( $x, $y, $z ) = @_;

    my $text;

    if ( get3dpstricks() == 1 ) { tripelohneminmax( $x, $y, $z ) }
    else { paar( x3d( $x, $y, $z ), y3d( $x, $y, $z ) ); }

    minmax3d( $x, $y, $z, $x, $y, $z );

    if ( getknoten() == 1 ) {
        knotensetzen( "\\pnode", $x, $y, $z );
    }
}

sub tripelohneminmax {
    my ( $x, $y, $z ) = @_;
    my $text;
    $text = "( " . $x . " , " . $y . " , " . $z . " )";
    textanhaengen($text);
}

```

49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109

1;

110
111

A.11. Das Modul für Zeichenroutinen

Listing A.11: zeichnen.pm

```
##### 1
# Modul fuer Zeichenroutinen 2
# 3
# Letzte Aenderung: 12.09.2011 4
##### 5
use POSIX qw /floor ceil/; 6
use Math::Trig; 7
use strict ; 8
use warnings; 9

# Linie zeichnen zwischen zwei Punkten 10
sub linie { 11
    my ( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $linienart, $linienstaerke, 12
        $farbe, $parameter, $pfeilart ) 13
        = @_; 14
    settexthier ("" ); 15
    16
    if ( $linienart ne "" ) { 17
        if ( ( istzahl($x1) == 1 ) 18
            && ( istzahl($y1) == 1 ) 19
            && ( istzahl($x2) == 1 ) 20
            && ( istzahl($y2) == 1 ) ) 21
        { 22
            minmax( $x1 - 1, $y1 - 1, $x2 + 1, $y2 + 1 ); 23
        } 24
    } 25
} 26

my $pstricksersatz3d = get3dpstricks(); 27
my $pstricksersatz = getpstricks(); 28

setachsengraph( $x1, $y1, $x1, $y1 ); 29
setachsengraph( $x2, $y2, $x2, $y2 ); 30

if ( istzahl($linienstaerke) == 1 ) { 31
    32
    if ( $linienstaerke > 0 ) { 33
        if ( ( $z1 eq "" ) || ( $z2 eq "" ) ) { 34
            paar( $x1, $y1 ); 35
            paar( $x2, $y2 ); 36
            setpstricks(0); 37
            set3dpstricks(0); 38
        } 39
        else { 40
            tripel( $x1, $y1, $z1 ); 41
            tripel( $x2, $y2, $z2 ); 42
            setachsengraph3d( $x1, $y1, $z1, $x1, $y1, $z1 ); 43
            setachsengraph3d( $x2, $y2, $z2, $x2, $y2, $z2 ); 44
        } 45
    } 46
    47
    liniekopf( $layer, $linienart, $linienstaerke, $farbe, $parameter, 48
    49
    50
    51
    52
```

```

    $pfeilart , gettexthier ( ) );
    settexthier ( "" );
}
else {
    if ( ( $z1 eq "" ) || ( $z2 eq "" ) ) {
        paar( $x1, $y1 );
        paar( $x2, $y2 );
        setpstricks ( 0 );
        set3dpstricks ( 0 );
    }
    else {
        tripel ( $x1, $y1, $z1 );
        tripel ( $x2, $y2, $z2 );
        setachsengraph3d( $x1, $y1, $z1, $x1, $y1, $z1 );
        setachsengraph3d( $x2, $y2, $z2, $x2, $y2, $z2 );
    }
}
liniekopf( $layer, $linienart, $linienstaerke, $farbe, $parameter,
    $pfeilart , gettexthier ( ) );
    settexthier ( "" );
}
setpstricks ( 0 );
set3dpstricks ( $pstrickersatz3d );
setpstricks ( $pstrickersatz );
}

sub liniertext {
    my (
        $layer,      $x1,      $y1,      $z1,
        $x2,          $y2,      $z2,      $abstandx,
        $abstandy,    $abstandz, $text,    $winkel,
        $ausrichtung, $linienart, $linienstaerke, $farbe,
        $parameter,   $pfeilart
    ) = @_;

    my ( $dx, $dy, $dz );

    if ( ( $z1 eq "" ) && ( $z2 eq "" ) ) {
        if ( $x1 == $x2 ) {
            if ( $abstandy == 0.2 ) {
                $dy = ( ( $y2 - $y1 ) / 2 );
            }
            else {
                $dy = ( ( $y2 - $y1 ) / 2 ) + $abstandy;
            }
        }
        else {
            $dy = ( ( $y2 - $y1 ) / 2 ) + $abstandy;
        }
        if ( $y1 == $y2 ) {
            if ( $abstandx == 0.2 ) {
                $dx = ( ( $x2 - $x1 ) / 2 );
            }
            else {
                $dx = ( ( $x2 - $x1 ) / 2 ) + $abstandx;
            }
        }
    }
}

```

```

    }
    else {
        $dx = ( ( $x2 - $x1 ) / 2 ) + $abstandx;
    }
}
else {
    $dy = ( ( $y2 - $y1 ) / 2 ) + $abstandy;
    $dx = ( ( $x2 - $x1 ) / 2 ) + $abstandx;
}

if ( ( $z1 ne "" ) && ( $z2 ne "" ) ) {
    $dz = ( ( $z2 - $z1 ) / 2 ) + $abstandz;
    beschriftung3d( $layer, $farbe, $x1 + $dx, $y1 + $dy, $z1 + $dz,
        $winkel, $ausrichtung, $text );

    minmax3d( $x1 - $dx, $y1 - $dy, $z1 - $dz, $x1 + $dx, $y1 + $dy,
        $z1 + $dz );

    minmax3d( $x1 + $dx, $y1 + $dy, $z1 + $dz, $x1 - $dx, $y1 - $dy,
        $z1 - $dz );

}
else {
    beschriftung( $layer, $farbe, $x1 + $dx, $y1 + $dy, $winkel,
        $ausrichtung, $text );
}

linie( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $linienart, $linienstaerke,
    $farbe, $parameter, $pfeilart );
}

sub liniertextabstand {
    my (
        $layer,    $x1,        $y1,        $z1,
        $x2,       $y2,        $z2,        $abstand,
        $abstandx, $abstandy,  $abstandz,  $text,
        $winkel,   $ausrichtung, $linienart, $linienstaerke,
        $farbe,    $parameter,  $pfeilart
    ) = @_ ;

    my (
        $winkelneu, $dx, $dy, $dz,  $laenge, $x1n,
        $y1n,       $x2n, $y2n, $x1neu, $y1neu
    );

    if ( ( $z1 ne "" ) && ( $z2 ne "" ) ) {
        kommentar( $layer, "hier" );
        $x1n = x3d( $x1, $y1, $z1 );
        $y1n = y3d( $x1, $y1, $z1 );

        $x2n = x3d( $x2, $y2, $z2 );
        $y2n = y3d( $x2, $y2, $z2 );

        $dx = $x2n - $x1n;
        $dy = $y2n - $y1n;
    }
    else {
        $x1n = $x1;
        $y1n = $y1;
    }
}

```

```

$х2n = $x2;
$у2n = $у2;

$dx = $x2 - $x1;
$dy = $у2 - $у1;
}

if ( $abstand == 0 ) {
  linietext (
    $layer,      $x1,      $y1,      $z1,
    $x2,         $y2,      $z2,      $abstandx,
    $abstandy,   $abstandz, $text,    $winkel,
    $ausrichtung, $linienart, $linienstaerke, $farbe,
    $parameter,  $pfeilart
  );
}
else {
  $winkelneu = radtodeg( atan2( $dy, $dx ) );
  $laenge = sqrt( $dx * $dx + $dy * $dy );

  $x1neu = calcendpunktх( $x1n, $abstand, $winkelneu + 90 );
  $y1neu = calcendpunktу( $y1n, $abstand, $winkelneu + 90 );

  linietextpolar (
    $layer,      $x1neu,   $y1neu,    "",
    $laenge,     $winkelneu, 0,        $abstandx,
    $abstandy,   $abstandz, $text,    $winkel,
    $ausrichtung, $linienart, $linienstaerke, $farbe,
    $parameter,  $pfeilart
  );
}
}

sub liniepolartextabstand {
  my (
    $layer,  $x,      $y,      $z,
    $laenge, $w1,     $w2,     $abstand,
    $abstandx, $abstandy, $abstandz, $text,
    $winkel,  $ausrichtung, $linienart, $linienstaerke,
    $farbe,   $parameter,  $pfeilart
  ) = @_ ;

  my ( $x2, $y2, $z2 );
  if ( ( $z eq "" ) || ( $w2 eq "" ) ) {
    $x2 = calcendpunktх( $x, $laenge, $w1 );
    $y2 = calcendpunktу( $y, $laenge, $w1 );
    linietextabstand(
      $layer,  $x,      $y,      "",
      $x2,     $y2,     "",      $abstand,
      $abstandx, $abstandy, $abstandz, $text,
      $winkel,  $ausrichtung, $linienart, $linienstaerke,
      $farbe,   $parameter,  $pfeilart
    );
  }
  else {
    $x2 = calcendpunktх3d( $x, $laenge, $w1, $w2 );
    $y2 = calcendpunktу3d( $y, $laenge, $w1, $w2 );
  }
}

```

\$z2 = calcendpunktz3d(\$z, \$laenge, \$w1, \$w2);	236
linietextabstand (	237
\$layer, \$x, \$y, \$z,	238
\$x2, \$y2, \$z2, \$abstand,	239
\$abstandx, \$abstandy, \$abstandz, \$text,	240
\$winkel, \$ausrichtung, \$linienart , \$linienstaerke ,	241
\$farbe, \$parameter, \$pfeilart	242
);	243
	244
}	245
	246
}	247
	248
#####	249
# Linie mit Laenge und Winkel	250
#####	251
sub liniepolar {	252
my (\$layer, \$x, \$y, \$z, \$laenge, \$w1, \$w2, \$linienart , \$linienstaerke ,	253
\$farbe, \$parameter, \$pfeilart)	254
= @_;	255
my (\$x2, \$y2, \$z2);	256
if ((\$z eq "") (\$w2 eq "")) {	257
\$x2 = calcendpunkt(\$x, \$laenge, \$w1);	258
\$y2 = calcendpunkt(\$y, \$laenge, \$w1);	259
linie (\$layer, \$x, \$y, "", \$x2, \$y2, "", \$linienart , \$linienstaerke ,	260
\$farbe, \$parameter, \$pfeilart);	261
}	262
else {	263
\$x2 = calcendpunkt3d(\$x, \$laenge, \$w1, \$w2);	264
\$y2 = calcendpunkt3d(\$y, \$laenge, \$w1, \$w2);	265
\$z2 = calcendpunkt3d(\$z, \$laenge, \$w1, \$w2);	266
linie (\$layer, \$x, \$y, \$z, \$x2, \$y2, \$z2, \$linienart , \$linienstaerke ,	267
\$farbe, \$parameter, \$pfeilart);	268
}	269
}	270
	271
sub linietextpolar {	272
my (	273
\$layer, \$x, \$y, \$z,	274
\$laenge, \$w1, \$w2, \$abstandx,	275
\$abstandy, \$abstandz, \$text, \$winkel,	276
\$ausrichtung, \$linienart , \$linienstaerke, \$farbe,	277
\$parameter, \$pfeilart	278
) = @_;	279
	280
my (\$x2, \$y2, \$z2, \$x3, \$y3);	281
if ((\$z eq "")) {	282
	283
if (\$w2 == 0) {	284
	285
\$x2 = calcendpunkt(\$x, \$laenge, \$w1);	286
\$y2 = calcendpunkt(\$y, \$laenge, \$w1);	287
linietext (	288
\$layer, \$x, \$y, "",	289
\$x2, \$y2, "", \$abstandx,	290
\$abstandy, \$abstandz, \$text, \$winkel,	291
\$ausrichtung, \$linienart , \$linienstaerke, \$farbe,	292
\$parameter, \$pfeilart	293
);	294
	295
}	296
}	

```

else {
    $x2 = calcendpunkttx( $x, $w2, $w1 + 90 );
    $y2 = calcendpunkty( $y, $w2, $w1 + 90 );
    $x3 = calcendpunkttx( $x2, $laenge, $w1 );
    $y3 = calcendpunkty( $y2, $laenge, $w1 );

    liniertext (
        $layer,      $x2,      $y2,      "",
        $x3,         $y3,         "",      $abstandx,
        $abstandy,   $abstandz, $text,    $winkel,
        $ausrichtung, $linienart, $linienstaerke, $farbe,
        $parameter,  $pfeilart
    );
}

}
else {
    $x2 = calcendpunkttx3d( $x, $laenge, $w1, $w2 );
    $y2 = calcendpunkty3d( $y, $laenge, $w1, $w2 );
    $z2 = calcendpunktz3d( $z, $laenge, $w1, $w2 );
    liniertext (
        $layer,      $x,      $y,      $z,
        $x2,         $y2,         $z2,      $abstandx,
        $abstandy,   $abstandz, $text,    $winkel,
        $ausrichtung, $linienart, $linienstaerke, $farbe,
        $parameter,  $pfeilart
    );
}
}

# Verschiedene Koepfe

sub liniekopf {
    my ( $layer, $linienart, $linienstaerke, $farbe, $parameter, $pfeilart,
        $anhang )
        = @_ ;

    if ( $anhang ne "" ) {

        if ( istzahl($linienstaerke) == 1 ) {

            if ( $linienstaerke > 0 ) {
                if ( get3dpstricks() == 1 ) {
                    set3dzeichnen(1);
                    befehlerzeugen( $layer, "pstThreeDLine", $farbe, $linienart,
                        $linienstaerke, $parameter, $pfeilart, $anhang );
                }
            }
            else {
                befehlerzeugen( $layer, "psline", $farbe, $linienart,
                    $linienstaerke, $parameter, $pfeilart, $anhang );
            }
        }
    }
}

else {
    if ( get3dpstricks() == 1 ) {
        set3dzeichnen(1);
        befehlerzeugen( $layer, "pstThreeDLine", $farbe, $linienart,

```

\$linienstaerke, \$parameter, \$pfeilart, \$anhang);	358
}	359
else {	360
befehlerzeugen(\$layer, "psline", \$farbe, \$linienart,	361
\$linienstaerke, \$parameter, \$pfeilart, \$anhang);	362
}	363
}	364
}	365
}	366
}	367
}	368
}	369
sub polygon {	370
my (\$layer, \$linienart, \$linienstaerke, \$farbe, \$fuellart, \$fuellfarbe,	371
\$parameter, \$anhang)	372
= @_;	373
}	374
if (\$anhang ne "") {	375
if (get3dpstricks() == 1) {	376
set3dzeichnen(1);	377
}	378
befehlfluellungerzeugen(	379
\$layer, "pstThreeDLine",	380
\$farbe, \$fuellart,	381
\$fuellfarbe, \$linienart,	382
\$linienstaerke, \$parameter,	383
"", \$anhang	384
);	385
}	386
else {	387
befehlfluellungerzeugen(	388
\$layer, "pspolygon", \$farbe, \$fuellart,	389
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter,	390
"", \$anhang	391
);	392
}	393
}	394
}	395
}	396
}	397
sub bezieckopf {	398
my (\$layer, \$linienart, \$linienstaerke, \$farbe, \$parameter, \$pfeilart,	399
\$anhang)	400
= @_;	401
}	402
my \$pstrickersatz3d = get3dpstricks();	403
my \$pstrickersatz = getpstricks();	404
}	405
if (\$anhang ne "") {	406
setpstricks(0);	407
set3dpstricks(0);	408
}	409
befehlerzeugen(\$layer, "psbezier", \$farbe, \$linienart, \$linienstaerke,	410
\$parameter, \$pfeilart, \$anhang);	411
}	412
}	413
}	414
setpstricks(\$pstrickersatz);	415
set3dpstricks(\$pstrickersatz3d);	416
}	417
}	418

```

}
419
420
421
sub kurvekopf {
422
my ( $layer, $linienart, $linienstaerke, $farbe, $parameter, $pfeilart,
423
    $anhang )
424
    = @_;
425
426
my $pstrickersatz3d = get3dpstricks();
427
my $pstrickersatz = getpstricks();
428
if ( $anhang ne "" ) {
429
    setpstricks(0);
430
    set3dpstricks(0);
431
    befehl erzeugen( $layer, "pscurve", $farbe, $linienart, $linienstaerke,
432
        $parameter, $pfeilart, $anhang );
433
    setpstricks(0);
434
    set3dpstricks($pstrickersatz3d);
435
    setpstricks($pstrickersatz);
436
437
    setpstricks(0);
438
    set3dpstricks($pstrickersatz3d);
439
    setpstricks($pstrickersatz);
440
}
441
}
442
443
# Punkte und Kreuze
444
sub punktzeichnen {
445
my (
446
    $layer,      $x,      $y,      $z,
447
    $radius,     $ausrichtung, $farbe, $fuellart,
448
    $fuellfarbe, $linienart, $linienstaerke, $text,
449
    $parameter
450
) = @_;
451
my ( $parameterweiter, $xneu, $yneu, $x2, $y2, $radiusx, $radiusy, $coor );
452
453
if ( $parameter =~ /drawCoor/ ) {
454
    $parameter =~ s/drawCoor//g;
455
    $parameter =~ s/drawCoor//g;
456
    $coor = 1;
457
}
458
else {
459
    $coor = 0;
460
}
461
462
$radiusx = $radius / getskalierung("x");
463
$radiusy = $radius / getskalierung("y");
464
465
if ( $z eq "" ) {
466
    $xneu = $x;
467
    $yneu = $y;
468
469
}
470
else {
471
    $xneu = x3d( $x, $y, $z );
472
    $yneu = y3d( $x, $y, $z );
473
}
474
475
$x2 =
476
    nachkommastellen(
477
        calcendpunkt( $xneu, ( $radiusx + 0.1 ), $ausrichtung ), 4 );
478
$y2 =
479

```

nachkommastellen(	480
calcendpunkty(\$yneu, (\$radiusy + 0.1), \$ausrichtung), 4);	481
	482
if (get3dpstricks() == 2) {	483
\$radius = 2 * \$radius;	484
tripel(\$x, \$y, \$z);	485
\$parameterweiter = \$parameter . ", dotsize=" . \$radius . "cm";	486
befehlfuellungerzeugen(	487
\$layer, "pstThreeDDot",	488
\$farbe, \$fuellart ,	489
\$fuellfarbe, \$linienart ,	490
\$linienstaerke, \$parameterweiter,	491
"", gettexthier ()	492
);	493
	494
# befehlerzeugen (\$layer, "pstThreeDDot", \$farbe, \$linienart , \$linienstaerke ,	495
\$parameterweiter, "", gettexthier ());	
settexthier ("");	496
if (\$coor == 1) {	497
linie (\$layer, \$x, \$y, \$z, \$x, \$y, 0, "dashed", \$linienstaerke,	498
\$farbe, "", "");	499
linie (\$layer, \$x, 0, 0, \$x, \$y, 0, "dashed", \$linienstaerke,	500
\$farbe, "", "");	501
linie (\$layer, 0, \$y, 0, \$x, \$y, 0, "dashed", \$linienstaerke,	502
\$farbe, "", "");	503
	504
# linie (\$layer, \$x, \$y, \$z, 0, \$y, \$z, "dashed", \$linienstaerke , \$farbe, "", "");	505
# linie (\$layer, 0, \$y, 0, 0, \$y, \$z, "dashed", \$linienstaerke , \$farbe, "", "");	506
# linie (\$layer, 0, 0, \$z, 0, \$y, \$z, "dashed", \$linienstaerke , \$farbe, "", "");	507
	508
# linie (\$layer, \$x, \$y, \$z, \$x, 0, \$z, "dashed", \$linienstaerke , \$farbe, "", "");	509
# linie (\$layer, \$x, 0, 0, \$x, 0, \$z, "dashed", \$linienstaerke , \$farbe, "", "");	510
# linie (\$layer, 0, 0, \$z, \$x, 0, \$z, "dashed", \$linienstaerke , \$farbe, "", "");	511
	512
}	513
}	514
else {	515
ellipse (	516
\$layer, \$xneu, \$yneu, \$radiusx,	517
\$radiusy, \$farbe, \$fuellart , \$fuellfarbe ,	518
\$linienart , \$linienstaerke, \$parameter	519
);	520
	521
}	522
	523
beschriftung(\$layer, \$farbe, \$x2, \$y2, 0, \$ausrichtung, \$text);	524
	525
}	526
	527
sub kreuzzeichnen {	528
my (	529
\$layer, \$x, \$y, \$z,	530
\$groesse, \$winkel, \$farbe, \$linienart ,	531
\$linienstaerke, \$text, \$ausrichtung, \$parameter	532
) = @_;	533
	534
my (\$xabstand, \$yabstand, \$xneu, \$yneu, \$radius);	535
\$xabstand =	536
nachkommastellen(sin(degtorad(\$winkel)) * (\$groesse / 2), 4);	537
\$yabstand =	538
nachkommastellen(cos(degtorad(\$winkel)) * (\$groesse / 2), 4);	539

```

$radius = sqrt( $xabstand * $xabstand + $yabstand * $yabstand );
if ( $z eq "" ) {
    #minmax( $x, $x, $y, $y );
    $xneu = $x;
    $yneu = $y;
}
else {
    # minmax3d( $x, $y, $z, $x, $y, $z );
    $xneu = x3d( $x, $y, $z );
    $yneu = y3d( $x, $y, $z );
}
setlabelabstand( $radius + 0.1 );
kreuz(
    $layer,      $xneu,      $yneu,
    $groesse,    $winkel,    $farbe,
    $linienart,  $linienstaerke, $parameter
);
beschriftung( $layer, $farbe, $xneu, $yneu, 0, $ausrichtung, $text );
setlabelabstand(0);
}

sub kreuz {
    my (
        $layer,      $x,      $y,
        $groesse,    $winkel,  $farbe,
        $linienart,  $linienstaerke, $parameter
    ) = @_ ;

    my $pstrickersatz3d = get3dpstricks();
    my $pstrickersatz   = getpstricks();

    setpstricks(0);
    set3dpstricks(0);

    my ( $xabstand1, $yabstand1, $xabstand2, $yabstand2 );
    $xabstand1 = sin( degtorad($winkel) ) * $groesse / 2;
    $yabstand1 = cos( degtorad($winkel) ) * $groesse / 2;
    $xabstand2 = sin( degtorad( $winkel + 90 ) ) * $groesse / 2;
    $yabstand2 = cos( degtorad( $winkel + 90 ) ) * $groesse / 2;
    linie (
        $layer,
        nachkommastellen( $x - $xabstand1, 4 ),
        nachkommastellen( $y - $yabstand1, 4 ),
        "",
        nachkommastellen( $x + $xabstand1, 4 ),
        nachkommastellen( $y + $yabstand1, 4 ),
        "",
        $linienart,
        $linienstaerke,
        $farbe,
        $parameter,
        "_"
    );

    linie (
        $layer,
        nachkommastellen( $x - $xabstand2, 4 ),

```

```

nachkommastellen( $y - $yabstand2, 4 ),
"",
nachkommastellen( $x + $xabstand2, 4 ),
nachkommastellen( $y + $yabstand2, 4 ),
"",
$linienart ,
$linienstaerke ,
$farbe,
$parameter,
"_"
);
setpstricks (0) ;
set3dpstricks($pstrickersatz3d);
setpstricks ( $pstrickersatz );
}

# Runde Sachen
sub ellipse {
  my (
    $layer,    $x,          $y,          $radius1,
    $radius2,  $farbe,      $fuellart , $fuellfarbe ,
    $linienart , $linienstaerke , $parameter
  ) = @_;
  my $text;
  settexthier ("" );
  if ( $radius1 == $radius2 ) {
    paar( $x, $y );
    klammer( $radius1 * getskalierung("x") );

    befehlfuellungerzeugen(
      $layer,    "pscircle", $farbe,      $fuellart ,
      $fuellfarbe , $linienart , $linienstaerke , $parameter,
      "",        gettexthier ()
    );

  }
  else {

    paar( $x,      $y );
    paar( $radius1, $radius2 );

    befehlfuellungerzeugen(
      $layer,    "psellipse", $farbe,      $fuellart ,
      $fuellfarbe , $linienart , $linienstaerke , $parameter,
      "",        gettexthier ()
    );

  }
  settexthier ("" );
  minmax( $x - $radius1, $y - $radius1, $x + $radius1, $y + $radius1 );
  minmax( $x - $radius2, $y - $radius2, $x + $radius2, $y + $radius2 );
  setachsengraph( $x - $radius1, $y - $radius1, $x + $radius1,
    $y + $radius1 );
  setachsengraph( $x - $radius2, $y - $radius2, $x + $radius2,
    $y + $radius2 );
}

sub kreis {
  my (
    $layer,    $x,          $y,          $radius,

```

```

    $farbe,      $fuellart , $fuellfarbe , $linienart ,
    $linienstaerke , $parameter
) = @_;
662
663
664
665
if ( ( $radius != 0 ) && ( $linienstaerke != 0 ) ) {
666
667
    ellipse (
668
        $layer,      $x,      $y,      $radius,
669
        $radius,      $farbe,      $fuellart , $fuellfarbe ,
670
        $linienart , $linienstaerke , $parameter
671
    );
672
}
673
}
674
675
sub kreisskala {
676
    my (
677
        $layer,      $x,      $y,      $radius,
678
        $laenge,      $winkel,      $farbe, $linienart ,
679
        $linienstaerke , $parameter, $pfeilart
680
    ) = @_;
681
682
    my ( $xstart , $ystart );
683
    $xstart = calcendpunkt( $x, $radius, $winkel );
684
    $ystart = calcendpunkt( $y, $radius, $winkel );
685
686
    liniepolar (
687
        $layer,      $xstart , $ystart ,      "",
688
        $laenge,      $winkel, "",      $linienart ,
689
        $linienstaerke , $farbe,      $parameter, $pfeilart
690
    );
691
692
}
693
694
sub kreistext {
695
    my (
696
        $layer, $x,      $y,      $radius, $winkel,
697
        $farbe, $text, $winkeltext, $ausrichtung
698
    ) = @_;
699
700
    my ( $xstart , $ystart );
701
    $xstart = calcendpunkt( $x, $radius, $winkel );
702
    $ystart = calcendpunkt( $y, $radius, $winkel );
703
704
    beschriftung( $layer, $farbe, $xstart , $ystart , $winkeltext, $ausrichtung,
705
        $text );
706
707
}
708
709
sub kreisbogen {
710
    my (
711
        $layer,      $linienart , $linienstaerke , $x1,
712
        $y1,      $r,      $w1,      $w2,
713
        $farbe,      $fuellart , $fuellfarbe ,      $geschlossen,
714
        $parameter, $pfeilart
715
    ) = @_;
716
717
    ellipsebogen(
718
        $layer,      $linienart ,
719
        $linienstaerke ,      $x1,
720
        $y1,      $r * getskalierung("x"),
721
        $r * getskalierung("y"), $w1,
722

```

\$w2,	\$farbe,	723
\$fuellart ,	\$fuellfarbe ,	724
\$geschlossen,	\$parameter,	725
\$pfeilart		726
)		727
		728
}		729
		730
sub ellipsebogen {		731
my (		732
\$layer, \$linienart , \$linienstaerke, \$x1,		733
\$y1, \$r1, \$r2, \$w1,		734
\$w2, \$farbe, \$fuellart , \$fuellfarbe ,		735
\$geschlossen, \$parameter, \$pfeilart		736
) = @_;		737
		738
my (\$x2, \$x3, \$y2, \$y3);		739
		740
if (\$r1 == \$r2) {		741
		742
paar(\$x1, \$y1);		743
klammer(\$r1);		744
klammer(\$w1);		745
klammer(\$w2);		746
		747
\$x2 = sprintf ("%0.4f", calcendpunkt(\$x1, \$r1, \$w1));		748
\$y2 = sprintf ("%0.4f", calcendpunkt(\$y1, \$r2, \$w1));		749
\$x3 = sprintf ("%0.4f", calcendpunkt(\$x1, \$r1, \$w2));		750
\$y3 = sprintf ("%0.4f", calcendpunkt(\$y1, \$r2, \$w2));		751
if (\$geschlossen eq "ja") {		752
		753
befehlfluellerzeugen(		754
\$layer, "pswedge", \$farbe, \$fuellart ,		755
\$fuellfarbe , \$linienart , \$linienstaerke, \$parameter,		756
"", gettexthier ()		757
);		758
		759
befehlerzeugen(\$layer, "psarc", \$farbe, \$linienart , \$linienstaerke,		760
\$parameter, \$pfeilart , gettexthier ());		761
		762
}		763
else {		764
befehlerzeugen(\$layer, "psarc", \$farbe, \$linienart , \$linienstaerke,		765
\$parameter, \$pfeilart , gettexthier ());		766
}		767
minmax(\$x1 - \$r1, \$y1 - \$r2, \$x1 - \$r1, \$y1 - \$r2);		768
minmax(\$x1 + \$r1, \$y1 + \$r2, \$x1 + \$r1, \$y1 + \$r2);		769
minmax(\$x1 - \$r1, \$y1 + \$r2, \$x1 - \$r1, \$y1 + \$r2);		770
minmax(\$x1 + \$r1, \$y1 - \$r2, \$x1 + \$r1, \$y1 - \$r2);		771
setachsengraph(\$x1 - \$r1, \$x1 - \$r1, \$y1 - \$r2, \$y1 - \$r2);		772
setachsengraph(\$x1 + \$r1, \$x1 + \$r1, \$y1 + \$r2, \$y1 + \$r2);		773
settexthier ("");		774
}		775
else {		776
		777
minmax(\$x1 - \$r1, \$y1 - \$r1, \$x1 + \$r1, \$y1 + \$r1);		778
minmax(\$x1 - \$r2, \$y1 - \$r2, \$x1 + \$r2, \$y1 + \$r2);		779
setachsengraph(\$x1 - \$r1, \$y1 - \$r1, \$x1 + \$r1, \$y1 + \$r1);		780
setachsengraph(\$x1 - \$r2, \$y1 - \$r2, \$x1 + \$r2, \$y1 + \$r2);		781
		782
my \$x2 = \$x1 + \$r1 * cos (degtorad(\$w1));		783

```

my $y2 = $y1 + $r2 * sin( degtorad($w1) );
my $x3 = $x1 + $r1 * cos( degtorad($w2) );
my $y3 = $y1 + $r2 * sin( degtorad($w2) );

if ( $geschlossen eq "ja" ) {

    einschlussanfang( $layer, $farbe, $fuellart, $fuellfarbe,
        $linienart, $linienstaerke, $parameter );
    linie( $layer, $x1, $y1, "", $x2, $y2, "", $linienart,
        $linienstaerke, $farbe, $parameter, "" );
    settexthier("");

    paar( $x1, $y1 );
    paarohneminmax( $r1, $r2 );

    klammer($w1);
    klammer($w2);

    befehlfuellungerzeugen( $layer, " psellipticarc ", "", "", "", "", "",
        "", "", gettexthier() );
    linie( $layer, $x3, $y3, "", $x1, $y1, "", $linienart,
        $linienstaerke, $farbe, $parameter, "" );

    einschlussende( $layer, $farbe, $fuellart, $fuellfarbe, $linienart,
        $linienstaerke, $parameter );
    paar( $x1, $y1 );
    paarohneminmax( $r1, $r2 );

    klammer($w1);
    klammer($w2);
    befehlrzeugen( $layer, " psellipticarc ", $farbe, $linienart,
        $linienstaerke, $parameter, $pfeilart, gettexthier() );

    linie( $layer, $x3, $y3, "", $x1, $y1, "", $linienart,
        $linienstaerke, $farbe, $parameter, "" );
    linie( $layer, $x1, $y1, "", $x2, $y2, "", $linienart,
        $linienstaerke, $farbe, $parameter, "" );

}
else {
    settexthier("");
    paar( $x1, $y1 );
    paarohneminmax( $r1, $r2 );
    klammer($w1);
    klammer($w2);
    befehlrzeugen( $layer, " psellipticarc ", $farbe, $linienart,
        $linienstaerke, $parameter, $pfeilart, gettexthier() );
}
}

sub ring {
    my (
        $layer, $linienart, $linienstaerke, $xzentrum,
        $yzentrum, $radius, $dicke, $winkell1,
        $winkel2, $farbe, $fuellart, $fuellfarbe,
        $parameter
    ) = @_;

    settexthier("");

```

784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844

```

my $radiusinnen = $radius - $dicke;
my $winkeldifferenz = abs( $winkel1 - $winkel2 );
my $x1 =
    nachkommastellen( calcendpunkt( $xzentrum, $radius, $winkel1 ), 2 );
my $y1 =
    nachkommastellen( calcendpunkt( $yzentrum, $radius, $winkel1 ), 2 );
my $x2 =
    nachkommastellen( calcendpunkt( $xzentrum, $radius, $winkel2 ), 2 );
my $y2 =
    nachkommastellen( calcendpunkt( $yzentrum, $radius, $winkel2 ), 2 );
my $x11 =
    nachkommastellen( calcendpunkt( $xzentrum, $radiusinnen, $winkel1 ), 2 );
my $y11 =
    nachkommastellen( calcendpunkt( $yzentrum, $radiusinnen, $winkel1 ), 2 );
my $x21 =
    nachkommastellen( calcendpunkt( $xzentrum, $radiusinnen, $winkel2 ), 2 );
my $y21 =
    nachkommastellen( calcendpunkt( $yzentrum, $radiusinnen, $winkel2 ), 2 );

my $hilfsfuellfarbe = getfuellfarbe ();
setfuellfarbe ("" );

befehlfuellungerzeugen(
    $layer, "pscustom", "", $fuellart, $farbe, "",
    "", $parameter, "", "{\\gsave"
);

paar( $xzentrum, $yzentrum );
klammer($radius);
klammer($winkel1);
klammer($winkel2);

befehlerzeugen( $layer, "psarc", $farbe, $linienart, $linienstaerke,
    $parameter, "", gettexthier () );

settexthier ("" );

paar( $xzentrum, $yzentrum );
klammer($radiusinnen);

klammer($winkel2);
klammer($winkel1);

befehlerzeugen( $layer, "psarcn", $farbe, $linienart, $linienstaerke,
    $parameter, "", gettexthier () );

settexthier ("" );

befehlfuellungerzeugen(
    $layer, " fill ", "", $fuellart,
    $fuellfarbe, "", "", $parameter,
    "", "\\grestore}"
);

paar( $xzentrum, $yzentrum );
klammer($radius);
klammer($winkel1);
klammer($winkel2);

befehlerzeugen( $layer, "psarc", $farbe, $linienart, $linienstaerke,
    $parameter, "", gettexthier () );

```

845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905

```

settexthier ("" );
906
907
908
paar( $xzentrum, $yzentrum );
909
klammer($radiusinnen);
910
911
klammer($winkel2);
912
klammer($winkel1);
913
914
befehlerzeugen( $layer, "psarcn", $farbe, $linienart, $linienstaerke,
915
$parameter, "", gettexthier () );
916
917
if ( $winkeldifferenz < 360 ) {
918
linie ( $layer, $x1, $y1, "", $x11, $y11, "", "solid", $linienstaerke,
919
$farbe, $parameter, "" );
920
linie ( $layer, $x2, $y2, "", $x21, $y21, "", "solid", $linienstaerke,
921
$farbe, $parameter, "" );
922
}
923
minmax(
924
$xzentrum - $radius,
925
$xzentrum - $radius,
926
$yzentrum - $radius,
927
$yzentrum - $radius
928
);
929
minmax(
930
$xzentrum + $radius,
931
$xzentrum + $radius,
932
$yzentrum + $radius,
933
$yzentrum + $radius
934
);
935
setachsengraph(
936
$xzentrum - $radius,
937
$xzentrum - $radius,
938
$yzentrum - $radius,
939
$yzentrum - $radius
940
);
941
setachsengraph(
942
$xzentrum + $radius,
943
$xzentrum + $radius,
944
$yzentrum + $radius,
945
$yzentrum + $radius
946
);
947
948
settexthier ("" );
949
setfuellfarbe ( $hilfsfuellfarbe );
950
}
951
952
# Eckige Dinge
953
sub rechteck {
954
my (
955
$layer,      $x,      $y,      $breite,
956
$hoehe,      $winkel, $farbe,  $fuellart,
957
$fuellfarbe, $linienart, $linienstaerke, $parameter
958
) = @_ ;
959
960
my ( $x2, $y2, $x3, $x4, $y3, $y4 );
961
962
$x2 = nachkommastellen( $x + cos( degtorad($winkel) ) * $breite, 4 );
963
$y2 = nachkommastellen( $y + sin( degtorad($winkel) ) * $breite, 4 );
964
$x3 = nachkommastellen( $x2 - sin( degtorad($winkel) ) * $hoehe, 4 );
965
$y3 = nachkommastellen( $y2 + cos( degtorad($winkel) ) * $hoehe, 4 );
966

```

\$x4 = nachkommastellen(\$x - sin(degtorad(\$winkel)) * \$hoehe, 4);	967
\$y4 = nachkommastellen(\$y + cos(degtorad(\$winkel)) * \$hoehe, 4);	968
	969
achsengraph(\$x, \$y);	970
achsengraph(\$x2, \$y2);	971
achsengraph(\$x3, \$y3);	972
achsengraph(\$x4, \$y4);	973
	974
settexthier ("");	975
if (\$linienstaerke != 0) {	976
	977
if (\$winkel == 0) {	978
paar(\$x, \$y);	979
paar(\$x + \$breite, \$y + \$hoehe);	980
	981
befehlfuellungerzeugen(	982
\$layer, "psframe", \$farbe, \$fuellart,	983
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter,	984
"", gettexthier ()	985
);	986
	987
}	988
else {	989
paar(\$x, \$y);	990
paar(\$x2, \$y2);	991
paar(\$x3, \$y3);	992
paar(\$x4, \$y4);	993
	994
polygon(\$layer, \$linienart, \$linienstaerke, \$farbe, \$fuellart,	995
\$fuellfarbe, \$parameter, gettexthier());	996
}	997
}	998
settexthier ("");	999
	1000
}	1001
	1002
sub parallelogramm {	1003
my (	1004
\$layer, \$x1, \$y1, \$breite,	1005
\$hoehe, \$w1, \$w2, \$farbe,	1006
\$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke,	1007
\$parameter	1008
) = @_;	1009
	1010
settexthier ("");	1011
my (\$x2, \$y2, \$x3, \$x4, \$y3, \$y4);	1012
if (\$w2 == 90) {	1013
rechteck(	1014
\$layer, \$x1, \$y1, \$breite,	1015
\$hoehe, \$w1, \$farbe, \$fuellart,	1016
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter	1017
);	1018
}	1019
else {	1020
\$x2 = calcendpunkt(\$x1, \$breite, \$w1);	1021
\$y2 = calcendpunkt(\$y1, \$breite, \$w1);	1022
\$x3 = calcendpunkt(\$x2, \$hoehe, \$w1 + \$w2);	1023
\$y3 = calcendpunkt(\$y2, \$hoehe, \$w1 + \$w2);	1024
\$x4 = calcendpunkt(\$x1, \$hoehe, \$w1 + \$w2);	1025
\$y4 = calcendpunkt(\$y1, \$hoehe, \$w1 + \$w2);	1026
	1027

paar(\$x1, \$y1);	1028
paar(\$x2, \$y2);	1029
paar(\$x3, \$y3);	1030
paar(\$x4, \$y4);	1031
	1032
polygon(\$layer, \$linienart, \$linienstaerke, \$farbe, \$fuellart ,	1033
\$fuellfarbe, \$parameter, gettexthier());	1034
	1035
}	1036
settexthier ("");	1037
}	1038
	1039
	1040
sub dreieckbreitehoehe {	1041
my (	1042
\$layer, \$x1, \$y1, \$breite,	1043
\$hoehe, \$winkel1, \$winkel2, \$farbe,	1044
\$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke,	1045
\$parameter	1046
) = @_;	1047
	1048
\$x1 = nachkommastellen(\$x1, 3);	1049
\$y1 = nachkommastellen(\$y1, 3);	1050
	1051
my (\$x2, \$y2, \$x3, \$y3);	1052
	1053
if (\$winkel1 == 90) {	1054
	1055
\$x2 = nachkommastellen(\$x1 + cos (degtorad(\$winkel2)) * \$breite, 3);	1056
\$y2 = nachkommastellen(\$y1 + sin (degtorad(\$winkel2)) * \$breite, 3);	1057
\$x3 = nachkommastellen(\$x2 - sin (degtorad(\$winkel2)) * \$hoehe, 3);	1058
\$y3 = nachkommastellen(\$y2 + cos (degtorad(\$winkel2)) * \$hoehe, 3);	1059
	1060
}	1061
else {	1062
	1063
\$x2 = nachkommastellen(\$x1 + cos (degtorad(\$winkel2)) * \$breite, 3);	1064
\$y2 = nachkommastellen(\$y1 + sin (degtorad(\$winkel2)) * \$breite, 3);	1065
\$x3 = nachkommastellen(	1066
\$x2 - sin (degtorad(\$winkel1 + \$winkel2)) * \$hoehe, 3);	1067
\$y3 = nachkommastellen(	1068
\$y2 + cos (degtorad(\$winkel1 + \$winkel2)) * \$hoehe, 3);	1069
}	1070
	1071
dreieck(\$layer, \$x1, \$y1, "", \$x2, \$y2, "", \$x3, \$y3, "", \$farbe,	1072
\$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke, \$parameter);	1073
	1074
}	1075
	1076
sub dreieckrechtwinklig {	1077
my (	1078
\$layer, \$x1, \$y1, \$laenge,	1079
\$winkelz, \$winkel, \$farbe, \$fuellart ,	1080
\$fuellfarbe, \$linienart, \$linienstaerke, \$parameter	1081
) = @_;	1082
	1083
\$x1 = nachkommastellen(\$x1, 3);	1084
\$y1 = nachkommastellen(\$y1, 3);	1085
	1086
my (\$schritt, \$winkelerg, \$x2, \$x3, \$x4, \$y2, \$y3, \$y4);	1087
	1088

\$winkelerg = 90 - \$winkelz - \$winkel;	1089
\$winkelerg = 90 - \$winkelz - \$winkel;	1090
\$winkelerg = 90 - \$winkelz - \$winkel;	1091
\$x2 = nachkommastellen(\$x1 + cos(degtorad(\$winkel)) * \$laenge, 3);	1092
\$y2 = nachkommastellen(\$y1 + sin(degtorad(\$winkel)) * \$laenge, 3);	1093
	1094
	1095
<i># Neuer Punkt im Dreieck,</i>	1096
<i># dazu werden die neuen Laengen</i>	1097
<i># und die naechste Koordinate berechnet .</i>	1098
my \$laengede = nachkommastellen(cos(degtorad(\$winkelz)) * \$laenge, 3);	1099
my \$laengeec =	1100
nachkommastellen(sqrt(\$laenge * \$laenge - \$laengede * \$laengede), 3);	1101
my \$laengedf =	1102
nachkommastellen(cos(degtorad(\$winkelerg)) * \$laengede, 3);	1103
my \$laengefe =	1104
nachkommastellen(sin(degtorad(\$winkelerg)) * \$laengede, 3);	1105
\$x3 = nachkommastellen(\$x1 + \$laengefe, 3);	1106
\$y3 = nachkommastellen(\$y1 + \$laengedf, 3);	1107
	1108
dreieck(\$layer, \$x1, \$y1, "", \$x2, \$y2, "", \$x3, \$y3, "", \$farbe,	1109
\$fuellart, \$fuellfarbe, \$linienart, \$linienstaerke, \$parameter);	1110
}	1111
	1112
my (\$hyp);	1113
\$hyp = 0;	1114
	1115
sub sethyp { \$hyp = 1; }	1116
sub gethyp { return \$hyp; }	1117
	1118
sub dreieckrechtwinkligpunkte {	1119
my (	1120
\$layer, \$x1, \$y1, \$x2,	1121
\$y2, \$winkel, \$farbe, \$fuellart ,	1122
\$fuellfarbe , \$linienart , \$linienstaerke, \$parameter	1123
) = @_;	1124
	1125
my \$dx = abs(\$x2 - \$x1);	1126
my \$dy = abs(\$y2 - \$y1);	1127
my \$laenge = sqrt(\$dx * \$dx + \$dy * \$dy);	1128
my \$winkeldrehung = radtodeg(atan2(\$dy, \$dx));	1129
	1130
dreieckrechtwinklig(	1131
\$layer, \$x1, \$y1, \$laenge,	1132
\$winkel, \$winkeldrehung, \$farbe, \$fuellart ,	1133
\$fuellfarbe , \$linienart , \$linienstaerke, \$parameter	1134
);	1135
	1136
}	1137
	1138
sub dreieckrechtwinkligpunktehoehe {	1139
my (	1140
\$layer, \$x1, \$y1, \$x2,	1141
\$y2, \$hoehe, \$farbe, \$fuellart ,	1142
\$fuellfarbe , \$linienart , \$linienstaerke, \$parameter	1143
) = @_;	1144
	1145
my (\$winkel, \$dx, \$dy, \$winkelerg, \$winkelb, \$winkelergb, \$x3, \$y3);	1146
	1147
\$dx = \$x2 - \$x1;	1148
\$dy = \$y2 - \$y1;	1149

```

if ( $dy <= 0 ) {
  if ( $dx <= 0 ) {
    $winkel    = radtodeg( atan( $dy / $dx ) );
    $winkelerg = 90 - $winkel;
    $winkelb   = degtorad($winkel);
    $winkelergb = degtorad($winkelerg);

    $x3 = nachkommastellen( $x1 + sin($winkelb) * $hoehe, 3 );
    $y3 = nachkommastellen( $y1 - cos($winkelb) * $hoehe, 3 );

  }
  else {
    $winkel    = radtodeg( atan( $dy / $dx ) );
    $winkelerg = 90 - $winkel;
    $winkelb   = degtorad($winkel);
    $winkelergb = degtorad($winkelerg);

    $x3 = nachkommastellen( $x1 - sin($winkelb) * $hoehe, 3 );
    $y3 = nachkommastellen( $y1 + cos($winkelb) * $hoehe, 3 );

  }
}
else {
  if ( $dx <= 0 ) {
    $winkel    = -radtodeg( atan( $dy / $dx ) );
    $winkelerg = 90 - $winkel;
    $winkelb   = degtorad($winkel);
    $winkelergb = degtorad($winkelerg);

    $x3 = nachkommastellen( $x1 - sin($winkelb) * $hoehe, 3 );
    $y3 = nachkommastellen( $y1 - cos($winkelb) * $hoehe, 3 );

  }
  else {
    $winkel    = radtodeg( atan( $dy / $dx ) );
    $winkelerg = 90 - $winkel;
    $winkelb   = degtorad($winkel);
    $winkelergb = degtorad($winkelerg);

    $x3 = nachkommastellen( $x1 - sin($winkelb) * $hoehe, 3 );
    $y3 = nachkommastellen( $y1 + cos($winkelb) * $hoehe, 3 );

  }
}

dreieck( $layer, $x1, $y1, "", $x2, $y2, "", $x3, $y3, "", $farbe,
  $fuellart, $fuellfarbe, $linienart, $linienstaerke, $parameter );

}

sub pfeil {
  my (
    $layer,    $x,    $y,    $breite,
    $hoehe,    $pfeil, $pfeillaenge, $farbe,
    $fuellart, $fuellfarbe, $linienart, $linienstaerke,
    $parameter
  ) = @_;

  settexthier ("" );

```

```

1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210

```

paar(\$x,	\$y);	1211			
paar(\$x + \$breite,	\$y);	1212			
paar(\$x + \$breite,	\$y - \$pfeil);	1213			
paar(\$x + \$breite + \$pfeillaenge,	\$y + 1 / 2 * \$hoehe);	1214			
paar(\$x + \$breite,	\$y + \$pfeil + \$hoehe);	1215			
paar(\$x + \$breite,	\$y + \$hoehe);	1216			
paar(\$x,	\$y + \$hoehe);	1217			
		1218			
polygon(\$layer, \$linienart, \$linienstaerke, \$farbe, \$fuellart, \$fuellfarbe,	\$parameter, gettexthier());	1219			
		1220			
settexthier ("");		1221			
		1222			
		1223			
}		1224			
		1225			
sub pfeilknick {		1226			
		1227			
my (		1228			
\$layer,	\$x,	\$y,	\$breite,		1229
\$breiteknick,	\$hoehe,	\$pfeil,	\$pfeillaenge,		1230
\$farbe,	\$fuellart,	\$fuellfarbe,	\$linienart,		1231
\$linienstaerke,	\$parameter				1232
) = @_;					1233
					1234
settexthier ("");					1235
					1236
paar(\$x, \$y);					1237
					1238
paar(\$x + \$breiteknick, \$y);					1239
paar(\$x + \$breiteknick, \$y - (\$breite - \$breiteknick) + \$hoehe);					1240
paar(\$x + \$breiteknick - \$pfeil,					1241
\$y - (\$breite - \$breiteknick) + \$hoehe);					1242
paar(\$x + \$breiteknick + 1 / 2 * \$hoehe,					1243
\$y - (\$breite - \$breiteknick) + \$hoehe - \$pfeillaenge / 2);					1244
paar(					1245
\$x + \$breiteknick + \$pfeil + \$hoehe,					1246
\$y - (\$breite - \$breiteknick) + \$hoehe					1247
);					1248
paar(\$x + \$breiteknick + \$hoehe,					1249
\$y - (\$breite - \$breiteknick) + \$hoehe);					1250
					1251
paar(\$x + \$breiteknick + \$hoehe, \$y + \$hoehe);					1252
					1253
paar(\$x, \$y + \$hoehe);					1254
polygon(\$layer, \$linienart, \$linienstaerke, \$farbe, \$fuellart, \$fuellfarbe,					1255
\$parameter, gettexthier());					1256
					1257
settexthier ("");					1258
					1259
}					1260
					1261
sub rechtecktext {					1262
my (					1263
\$layer,	\$x1,	\$y1,	\$x2,		1264
\$y2,	\$text,	\$farbe,	\$fuellart,		1265
\$fuellfarbe,	\$linienart,	\$linienstaerke,	\$parameter		1266
) = @_;					1267
					1268
achsengraph(\$x1, \$y1);					1269
achsengraph(\$x2, \$y2);					1270
tupel(\$x1, \$y1, \$x2, \$y2);					1271

```

if ( $farbe eq getfarbe() ) {
    textanhaengen("{}");
    textanhaengen($text);
    textanhaengen("{}");
}
else {
    textanhaengen("{\\color{");
    textanhaengen($farbe);
    textanhaengen("}");
    textanhaengen($text);
    textanhaengen("}");
}
befehlfuellungerzeugen(
    $layer,      "psTextFrame", $farbe,      $fuellart ,
    $fuellfarbe, $linienart ,    $linienstaerke, $parameter,
    "",          gettexthier ()
);
}

sub dreieck {
    my ( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $x3, $y3, $z3, $farbe, $fuellart ,
        $fuellfarbe, $linienart, $linienstaerke, $parameter )
        = @_;

    achsengraph( $x1, $y1 );
    achsengraph( $x2, $y2 );
    achsengraph( $x3, $y3 );
    settexthier ("" );
    if ( ( $z1 eq "" ) || ( $z2 eq "" ) || ( $z3 eq "" ) ) {

        paar( $x1, $y1 );
        paar( $x2, $y2 );
        paar( $x3, $y3 );

    }
    else {
        setachsengraph3d( $x1, $y1, $z1, $x1, $y1, $z1 );
        setachsengraph3d( $x2, $y2, $z2, $x2, $y2, $z2 );
        setachsengraph3d( $x3, $y3, $z3, $x3, $y3, $z3 );
        tripel ( $x1, $y1, $z1 );
        tripel ( $x2, $y2, $z2 );
        tripel ( $x3, $y3, $z3 );
        tripel ( $x1, $y1, $z1 );

    }

    polygon( $layer, $linienart, $linienstaerke, $farbe, $fuellart, $fuellfarbe,
        $parameter, gettexthier() );
    settexthier ("" );
}

sub viereck {
    my ( $layer, $x1, $y1, $z1, $x2, $y2, $z2, $x3, $y3, $z3, $x4, $y4, $z4,
        $farbe, $fuellart, $fuellfarbe, $linienart, $linienstaerke, $parameter )
        = @_;
    settexthier ("" );
    if ( ( $z1 eq "" ) || ( $z2 eq "" ) || ( $z3 eq "" ) ) {
        achsengraph( $x1, $y1 );

```

achsengraph(\$x2, \$y2);	1333
achsengraph(\$x3, \$y3);	1334
achsengraph(\$x4, \$y4);	1335
vier(\$x1, \$y1, \$x2, \$y2, \$x3, \$y3, \$x4, \$y4);	1336
	1337
}	1338
else {	1339
setachsengraph3d(\$x1, \$y1, \$z1, \$x1, \$y1, \$z1);	1340
setachsengraph3d(\$x2, \$y2, \$z2, \$x2, \$y2, \$z2);	1341
setachsengraph3d(\$x3, \$y3, \$z3, \$x3, \$y3, \$z3);	1342
setachsengraph3d(\$x4, \$y4, \$z4, \$x4, \$y4, \$z4);	1343
tripel (\$x1, \$y1, \$z1);	1344
tripel (\$x2, \$y2, \$z2);	1345
tripel (\$x3, \$y3, \$z3);	1346
tripel (\$x4, \$y4, \$z4);	1347
tripel (\$x1, \$y1, \$z1);	1348
	1349
}	1350
	1351
polygon(\$layer, \$linienart, \$linienstaerke, \$farbe, \$fuellart, \$fuellfarbe ,	1352
\$parameter, gettexthier());	1353
settexthier ("");	1354
	1355
}	1356
	1357
sub neck {	1358
my (	1359
\$layer, \$x1, \$y1, \$anzahl,	1360
\$laenge, \$winkel, \$farbe, \$fuellart ,	1361
\$fuellfarbe , \$linienart , \$linienstaerke, \$parameter	1362
) = @_;	1363
settexthier ("");	1364
	1365
my \$innenwinkel = 360 / \$anzahl;	1366
my \$x;	1367
	1368
my \$i = 0;	1369
for (\$i = 0 ; \$i <= \$anzahl ; \$i++) {	1370
paar(	1371
nachkommastellen(	1372
\$x1 + \$laenge * cos (degtorad(\$winkel + \$i * \$innenwinkel)),	1373
4	1374
),	1375
nachkommastellen(	1376
\$y1 + \$laenge * sin (degtorad(\$winkel + \$i * \$innenwinkel)),	1377
4	1378
)	1379
);	1380
achsengraph(	1381
nachkommastellen(	1382
\$x1 + \$laenge * cos (degtorad(\$winkel + \$i * \$innenwinkel)),	1383
4	1384
),	1385
nachkommastellen(	1386
\$y1 + \$laenge * sin (degtorad(\$winkel + \$i * \$innenwinkel)),	1387
4	1388
)	1389
);	1390
}	1391
	1392
polygon(\$layer, \$linienart, \$linienstaerke, \$farbe, \$fuellart, \$fuellfarbe ,	1393

```

    $parameter, gettexthier() );
    settexthier("");
}
sub winkel {
    my (
        $layer,      $x1,      $y1,      $x2,
        $y2,      $x3,      $y3,      $radius,
        $text,      $winkelrichtung, $farbe,      $fuellart ,
        $fuellfarbe , $linienart ,      $linienstaerke , $parameter
    ) = @_;
    my ( $winkel, $winkel1, $winkel2, $xtext, $ytext );
    my $zu;

    if ( getfuellfarbe() eq $fuellfarbe ) {
        $zu = "nein";
    }
    else {
        $zu = "ja";
    }

    $winkel = vektorwinkel(
        vektordifferenz( "x", $x2, $y2, 0, $x1, $y1, 0 ),
        vektordifferenz( "y", $x2, $y2, 0, $x1, $y1, 0 ),
        vektordifferenz( "z", $x2, $y2, 0, $x1, $y1, 0 ),
        vektordifferenz( "x", $x2, $y2, 0, $x3, $y3, 0 ),
        vektordifferenz( "y", $x2, $y2, 0, $x3, $y3, 0 ),
        vektordifferenz( "z", $x2, $y2, 0, $x3, $y3, 0 )
    );
    kommentar( $layer, "Der Winkel " . $winkel . " wird gezeichnet" );
    $winkel1 = nachkommastellen(
        radtodeg(
            atan2(
                vektordifferenz( "y", $x2, $y2, 0, $x1, $y1, 0 ),
                vektordifferenz( "x", $x2, $y2, 0, $x1, $y1, 0 )
            )
        ),
        3
    );
    $winkel2 = nachkommastellen(
        radtodeg(
            atan2(
                vektordifferenz( "y", $x2, $y2, 0, $x3, $y3, 0 ),
                vektordifferenz( "x", $x2, $y2, 0, $x3, $y3, 0 )
            )
        ),
        3
    );
    if ( $winkelrichtung eq "i" ) {
        kreisbogen(
            $layer,      $linienart ,
            $linienstaerke , $x2,
            $y2,      $radius * getskalierung("x"),
            $winkel1,      $winkel2,
            $farbe,      $fuellart ,
            $fuellfarbe , $zu,
            $parameter,      ""
        );
        $xtext = $x2 + cos( degtorad( $winkel1 + $winkel / 2 ) ) * $radius / 2;
        $ytext = $y2 + sin( degtorad( $winkel1 + $winkel / 2 ) ) * $radius / 2;

```

beschriftung(\$layer, "black", \$xtext, \$ytext, 0, 90, \$text);	1455
if ((\$winkel == 90) && (\$text eq " ")) {	1456
kreis(	1457
\$layer, \$xtext, \$ytext, 0.06, "black", "solid",	1458
"black", "solid", 0.01	1459
);	1460
}	1461
}	1462
elseif (\$winkelrichtung eq "a") {	1463
kreisbogen(	1464
\$layer, \$linienart,	1465
\$linienstaerke, \$x2,	1466
\$y2, \$radius * getskalierung("x"),	1467
\$winkel2, \$winkel1,	1468
\$farbe, \$fuellart,	1469
\$fuellfarbe, \$zu,	1470
\$parameter, ""	1471
);	1472
\$xtext = \$x2 -	1473
cos(degtorad(\$winkel2 - (360 + \$winkel) / 2)) * \$radius / 2;	1474
\$ytext = \$y2 +	1475
sin(degtorad(\$winkel2 + (360 - \$winkel) / 2)) * \$radius / 2;	1476
beschriftung(\$layer, "black", \$xtext, \$ytext, 0, 90, \$text);	1477
if ((\$winkel == 90) && (\$text eq " ")) {	1478
kreis(	1479
\$layer, \$xtext, \$ytext, 0.06, "black", "solid",	1480
"black", "solid", 0.01	1481
);	1482
}	1483
}	1484
else {	1485
kreisbogen(	1486
\$layer, \$linienart,	1487
\$linienstaerke, \$x2,	1488
\$y2, \$radius * getskalierung("x"),	1489
\$winkel1, \$winkel2,	1490
\$farbe, \$fuellart,	1491
\$fuellfarbe, \$zu,	1492
\$parameter, ""	1493
);	1494
\$xtext = \$x2 + cos(degtorad(\$winkel1 + \$winkel / 2)) * \$radius / 2;	1495
\$ytext = \$y2 + sin(degtorad(\$winkel1 + \$winkel / 2)) * \$radius / 2;	1496
beschriftung(\$layer, "black", \$xtext, \$ytext, 0, 90, \$text);	1497
}	1498
}	1499
}	1500
	1501
sub punktezeichnen {	1502
my (\$layer, \$x1, \$y1, \$x2, \$y2, \$farbe, \$verbunden) = @_;	1503
if (\$verbunden eq "jamitk") {	1504
kreis(\$layer, \$x1, \$y1, 0.15, \$farbe, "solid", \$farbe, "solid", 0.01);	1505
linie (	1506
\$layer, \$x1, \$y1, "", \$x2, \$y2,	1507
"", "solid", 0.15, \$farbe, "", "—"	1508
);	1509
kreis(\$layer, \$x2, \$y2, 0.15, \$farbe, "solid", \$farbe, "solid", 0.01);	1510
}	1511
if (\$verbunden eq "jaohne") {	1512
linie (	1513
\$layer, \$x1, \$y1, "", \$x2, \$y2,	1514
"", "solid", 0.15, \$farbe, "", "—"	1515

```

    );
}
if ( $verbunden eq "nein" ) {
    kreis( $layer, $x1, $y1, 0.15, $farbe, "solid", $farbe, "solid", 0.01 );
    kreis( $layer, $x2, $y2, 0.15, $farbe, "solid", $farbe, "solid", 0.01 );
}
}

# Tangenten, Steigungsdreiecke etc, werden gezeichnet
sub tangente {
    my ( $layer, $func, $x, $breite, $farbe, $linienart, $linienstaerke,
        $parameter )
        = @_;

    linie (
        $layer,
        $x - $breite / 2,
        calctangente( $func, $x - $breite / 2, $x ),
        "",
        $x + $breite / 2,
        calctangente( $func, $x + $breite / 2, $x ),
        "",
        $linienart,
        $linienstaerke,
        $farbe,
        $parameter,
        ""
    );
}

sub tangendendreieck {
    my ( $layer, $func, $x, $breite, $farbe, $linienart, $linienstaerke,
        $parameter )
        = @_;
    my $y = calcfunk( $func, "x", $x );

    my ( $xmin, $xmax, $ymin, $ymax, $m, $dx, $dy );
    $xmin = $x - $breite / 2;
    $xmax = $x + $breite / 2;
    $ymin = calctangente( $func, $xmin, $x );
    $ymax = calctangente( $func, $xmax, $x );
    $dy = $ymax - $ymin;
    $dx = $xmax - $xmin;

    $m = $dy / $dx;

    kommentar( $layer, "y/x = " . $dy . "/" . $dx . " = " . $m );

    if ( $m > 0 ) {
        dreieck( $layer, $xmin, $ymin, "", $xmax, $ymax, "", $xmax, $ymin, "",
            $farbe, "", "", $linienart, $linienstaerke, $parameter );
    }
    else {
        dreieck( $layer, $xmin, $ymin, "", $xmax, $ymax, "", $xmin, $ymax, "",
            $farbe, "", "", $linienart, $linienstaerke, $parameter );
    }
}

```

```

}
1577
1578
1579
sub sekante {
1580
  my ( $layer, $func, $x, $breite, $farbe, $linienart, $linienstaerke,
1581
    $parameter )
1582
    = @_;
1583
1584
    linie (
1585
      $layer, $x,
1586
      calcfunk( $func, "x", $x ), "",
1587
      $x + $breite, calcfunk( $func, "x", $x + $breite ),
1588
      "", $linienart,
1589
      $linienstaerke, $farbe,
1590
      $parameter, ""
1591
    );
1592
1593
}
1594
1595
sub sekantendreieck {
1596
  my ( $layer, $func, $x, $breite, $farbe, $linienart, $linienstaerke,
1597
    $parameter )
1598
    = @_;
1599
  my $y = calcfunk( $func, "x", $x );
1600
  my $xminwert = $x;
1601
  my $xmaxwert = $x + $breite;
1602
  my $yminwert = calcfunk( $func, "x", $xminwert );
1603
  my $ymaxwert = calcfunk( $func, "x", $xmaxwert );
1604
  my $yabl = ( $ymaxwert - $yminwert ) / ( $xmaxwert - $xminwert );
1605
  if ( $yminwert > $ymaxwert ) {
1606
    dreieck(
1607
      $layer, $xminwert, $yminwert, "",
1608
      $xminwert, $ymaxwert, "", $xmaxwert,
1609
      $ymaxwert, "", $farbe, "",
1610
      "", $linienart, $linienstaerke, $parameter
1611
    );
1612
  }
1613
  else {
1614
    dreieck(
1615
      $layer, $xminwert, $yminwert, "",
1616
      $xmaxwert, $yminwert, "", $xmaxwert,
1617
      $ymaxwert, "", $farbe, "",
1618
      "", $linienart, $linienstaerke, $parameter
1619
    );
1620
  }
1621
}
1622
1623
my $labelabstand = 0;
1624
1625
sub setlabelabstand {
1626
  my ($wert) = @_;
1627
  $labelabstand = $wert;
1628
}
1629
1630
sub getlabelabstand { return $labelabstand; }
1631
1632
### Text einfüegen
1633
sub beschriftung {
1634
  my ( $layer, $farbe, $x, $y, $winkel, $ausrichtung, $text ) = @_;
1635
  settexthier ("" );
1636
  if ( $text ne "" ) {
1637

```

if (\$ausrichtung eq "c") {	1638
klammer(\$winkel);	1639
paar(\$x, \$y);	1640
klammer(	1641
textfarbe(	1642
beschriftungsgroesse(\$text, getbeschriftungsgroesse()),	1643
\$farbe	1644
)	1645
);	1646
beschriftungskopf(\$layer, \$farbe, \$winkel, \$ausrichtung,	1647
gettexthier ());	1648
}	1649
else {	1650
	1651
klammer(getlabelabstand());	1652
textanhaengen("[");	1653
textanhaengen(\$ausrichtung);	1654
textanhaengen("]");	1655
klammer(\$winkel);	1656
paar(\$x, \$y);	1657
klammer(	1658
textfarbe(	1659
beschriftungsgroesse(\$text, getbeschriftungsgroesse()),	1660
\$farbe	1661
)	1662
);	1663
beschriftungskopf(\$layer, \$farbe, \$winkel, \$ausrichtung,	1664
gettexthier ());	1665
}	1666
}	1667
	1668
settexthier ("");	1669
my \$laenge = length(\$text);	1670
if ((istzahl(\$x) == 1) && (istzahl(\$y) == 1)) {	1671
minmax(	1672
\$x - (\$laenge) * 0.5 / getskalierung("x"),	1673
\$y - (\$laenge) * 0.5 / getskalierung("y"),	1674
\$x - (\$laenge) * 0.5 / getskalierung("x"),	1675
\$y - (\$laenge) * 0.5 / getskalierung("y")	1676
);	1677
minmax(	1678
\$x + (\$laenge) * 0.5 / getskalierung("x"),	1679
\$y + (\$laenge) * 0.5 / getskalierung("y"),	1680
\$x + (\$laenge) * 0.5 / getskalierung("x"),	1681
\$y + (\$laenge) * 0.5 / getskalierung("y")	1682
);	1683
}	1684
}	1685
	1686
sub beschriftungskopf {	1687
my (\$layer, \$farbe, \$winkel, \$ausrichtung, \$anhang) = @_;	1688
if (\$ausrichtung eq "c") {	1689
befehlerzeugen(\$layer, "rput", "", "", "", "", "", \$anhang);	1690
}	1691
	1692
else {	1693
befehlerzeugen(\$layer, "uput", "", "", "", "", "", \$anhang);	1694
}	1695
}	1696
	1697
sub einschlussanfang {	1698

```

my ( $layer, $farbe, $fuellart, $fuellfarbe, $linienart, $linienstaerke,
    $parameter )
    = @_;
    befehlfuellungerzeugen(
        $layer,      "pscustom", $farbe, "",
        $fuellfarbe, "",      "",      $parameter,
        "",          "{\gsave"
    );
}

sub einschliessende {
    my ( $layer, $farbe, $fuellart, $fuellfarbe, $linienart, $linienstaerke,
        $parameter )
        = @_;

    befehlfuellungerzeugen(
        $layer,      " fill ", $farbe, $fuellart,
        $fuellfarbe, "",      "",      $parameter,
        "",          "\\grestore}"
    );
}

1;

```

A.12. Das Modul für Grafiken

Listing A.12: bildbearbeitung.pm

```

#####
# Modul fuer Zeichenroutinen
#
# Letzte Aenderung: 12.09.2011
#####
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;

### Funktionen fuer die Bildbearbeitung

sub bildhoehe {
    my ($bilddatei) = @_;
    $bilddatei =~ s/eps/png/g;
    my ( $dx, $dy ) = imgsize($bilddatei);
    return $dy;
}

sub bildbreite {
    my ($bilddatei) = @_;
    $bilddatei =~ s/eps/png/g;
    my ( $dx, $dy ) = imgsize($bilddatei);
    return $dx;
}

sub bildhoeheberechnet {
    my ( $bilddatei, $breite ) = @_;
    return nachkommastellen(

```

```

    bildhoehe($bilddatei) * $breite / bildbreite($bilddatei), 4 );
}
# Ein Bild einfüegen

my $graphicx = 0;

sub bild {
    my ( $layer, $x1, $y1, $z1, $breite, $winkel, $bild ) = @_;

    my $pstrickersatz3d = get3dpstricks();
    my $pstrickersatz    = getpstricks();

    my ( $x, $y, $x2, $y2, $x3, $x4, $y3, $y4 );

    setpstricks(0);
    set3dpstricks(0);

    if ( $z1 eq "" ) {
        $x = $x1;
        $y = $y1;
    }
    else {
        if ( ( istzahl($x1) == 1 )
            && ( istzahl($y1) == 1 )
            && ( istzahl($z1) == 1 ) )
        {
            $x = x3d( $x1, $y1, $z1 );
            $y = y3d( $x1, $y1, $z1 );
        }
    }

    if ( ( istzahl($x) == 1 ) && ( istzahl($y) == 1 ) ) {

        if ( $winkel == 0 ) {
            minmax(
                $x, $y,
                $x + $breite,
                $y + bildhoeheberechnet( $bild, $breite )
            );
        }
        else {
            $x2 =
                nachkommastellen( $x + cos( degtorad($winkel) ) * $breite, 4 );
            $y2 =
                nachkommastellen( $y + sin( degtorad($winkel) ) * $breite, 4 );
            $x3 = nachkommastellen(
                $x2 -
                sin( degtorad($winkel) ) *
                bildhoeheberechnet( $bild, $breite ),
                4
            );
            $y3 = nachkommastellen(
                $y2 +
                cos( degtorad($winkel) ) *
                bildhoeheberechnet( $bild, $breite ),
                4
            );
            $x4 = nachkommastellen(
                $x -

```

```

30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90

```

```

        sin( degtorad($winkel) ) *
        bildhoeheberechnet( $bild, $breite ),
    4
);
$y4 = nachkommastellen(
    $y +
        cos( degtorad($winkel) ) *
        bildhoeheberechnet( $bild, $breite ),
    4
);
minmax( $x, $y, $x, $y );
minmax( $x2, $y2, $x2, $y2 );
minmax( $x3, $y3, $x3, $y3 );
minmax( $x4, $y4, $x4, $y4 );
}
}

if ( $graphicx == 0 ) {
    usepackage( "", "graphicx" );
    $graphicx = 1;
}
schreibeinarray(
    $layer,
    sprintf(
"\\psrotate(%s,%s){%g}\\rput[lb](%s,%s){\\includegraphics[width=%gcm]{%s}}}%%",
        $x, $y, $winkel, $x, $y, $breite, $bild
    )
);
setpstricks(0);
set3dpstricks($pstrickersatz3d);
setpstricks( $pstrickersatz );
}

1;

```

A.13. Die Spielfeld-Module

Listing A.13: uhr.pm

```

#####
# Modul fuer Uhren
#
# Letzte Aenderung: 15.02.2011
#####
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;

sub uhr {
my ( $layer, $x, $y, $radius, $laenge, $farbe, $fuellart, $fuellfarbe, $linienart,
    $linienstaerke, $parameter ) = @_ ;

kreis( $layer, $x, $y, $radius, $farbe, $fuellart, $fuellfarbe, $linienart, $linienstaerke,
    $parameter );
kreis( $layer, $x, $y, $laenge / 8, $farbe, $fuellart, $fuellfarbe, $linienart, $linienstaerke,
    $parameter );

```

```

my $i = 0;
for ( $i = 0 ; $i < 360 ; $i = $i + 90 ) {
    kreisskala( $layer, $x, $y, $radius - $laenge, $laenge, $i, $farbe, $linienart,
        $linienstaerke, $parameter, "-" );
}

for ( $i = 0 ; $i < 360 ; $i = $i + 30 ) {
    kreisskala( $layer, $x, $y, $radius - $laenge / 2, $laenge / 2, $i, $farbe, $linienart,
        $linienstaerke, $parameter, "-" );
}

for ( $i = 00 ; $i < 360 ; $i = $i + 10 ) {
    kreisskala( $layer, $x, $y, $radius - $laenge / 4, $laenge / 4, $i, $farbe, $linienart,
        $linienstaerke, $parameter, "-" );
}

kreistext( $layer, $x, $y, $radius - $laenge, 90, $farbe, 12, 0, 270 );
kreistext( $layer, $x, $y, $radius - $laenge, 0, $farbe, 3, 0, 180 );
kreistext( $layer, $x, $y, $radius - $laenge, 180, $farbe, 9, 0, 0 );
kreistext( $layer, $x, $y, $radius - $laenge, 270, $farbe, 6, 0, 90 );
}

sub uhrzeit {
my ( $layer, $x, $y, $radius, $stunde, $minute, $sekunde, $farbe, $linienart, $linienstaerke,
    $parameter ) = @_ ;

my $winkelstunde = 0;
my $winkelminute = 0;
my $winkelsekunde = 0;

if ( $sekunde ne "" ) {
    $winkelsekunde = 90 - 360 / 60 * $sekunde;
    liniepolar( $layer, $x, $y, "", $radius, $winkelsekunde, "", $linienart, $linienstaerke / 2,
        $farbe, $parameter, "" );
}

if ( $minute ne "" ) {
    $winkelminute = 90 - 360 / 60 * $minute;
    if ( $sekunde ne "" ) { $winkelminute = 90 - ( 360 / 60 * $minute + 10 / 60 * $sekunde ); }
    liniepolar( $layer, $x, $y, "", $radius, $winkelminute, "", $linienart, $linienstaerke,
        $farbe, $parameter, "->" );
}

if ( $stunde ne "" ) {
    if ( $stunde > 12 ) { $stunde = $stunde - 12; }
    $winkelstunde = 90 - 360 / 12 * $stunde;
    if ( $minute ne "" ) { $winkelstunde = 90 - ( 360 / 12 * $stunde + 10 / 60 * $minute ); }

    liniepolar( $layer, $x, $y, "", $radius * 3 / 4, $winkelstunde, "", $linienart,
        $linienstaerke, $farbe, $parameter, "->" );
}

}

1;

```

Listing A.14: knoten.pm

```
##### | 1
```

```

# Modul fuer Uhren
#
# Letzte Aenderung: 18.02.2011
#####
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;

my $knoten = 0;

sub getknoten { return $knoten; }

sub setknoten {
    my ($wert) = @_ ;
    $knoten = $wert;
}

my $knotenzaehler = 0;

my @knoteninhaltx = ();
my @knoteninhalty = ();
my $i = 0;
for ( $i = 0 ; $i <= 10000 ; $i++ ) {
    $knoteninhaltx[$i] = 1234.4321;
    $knoteninhalty[$i] = 1234.4321;
}

sub knotensetzen {
    my ( $knotenart, $x, $y, $z ) = @_ ;

    if ( $z eq "" ) {
        my $text = "(
            . nachkommastellen( $x, 2 ) . " , "
            . nachkommastellen( $y, 2 ) . " )";
        my $text2 = "\$(
            . nachkommastellen( $x, 2 ) . " | "
            . nachkommastellen( $y, 2 ) . " )\$";

        $text2 =~ s/\././g;
        my $textknoten =
            $knotenart
            . $text . "{A"
            . $knotenzaehler . "}"
            . "\\rput"
            . $text . "{"
            . $text2 . "}" ;
        my $i = 0;
        my $j = 0;
        $knotenzaehler++;

        pstricks( 10, $textknoten );
        my $laenge = length($text);
        minmax(
            $x - ( 0.5 * $laenge ) / getskalierung("x"),
            $y - ( 0.5 * $laenge ) / getskalierung("y"),
            $x - ( 0.5 * $laenge ) / getskalierung("x"),
            $y - ( 0.5 * $laenge ) / getskalierung("y")
        );

        minmax(

```

2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62

```

    $x + ( 0.5 * $laenge ) / getskalierung("x"),
    $y + ( 0.5 * $laenge ) / getskalierung("y"),
    $x + ( 0.5 * $laenge ) / getskalierung("x"),
    $y + ( 0.5 * $laenge ) / getskalierung("y")
  );
}
else {
  my $text = "(" . x3d( $x, $y, $z ) . "," . y3d( $x, $y, $z ) . ")";
  my $text2 = "\$( "
    . nachkommastellen( $x, 2 ) . "|"
    . nachkommastellen( $y, 2 ) . "|"
    . nachkommastellen( $z, 2 ) . ")\$";

  $text2 =~ s/\.\/./g;
  my $textknoten =
    $knotenart
    . $text . "{A"
    . $knotenzaehler . "}"
    . "\rput"
    . $text . "{"
    . $text2 . "}";
  my $i = 0;
  my $j = 0;
  $knotenzaehler++;

  pstricks( 10, $textknoten );
}
}

sub knoten {
  my (
    $layer, $farbe,      $x,      $y, $z,
    $name, $bezeichnung, $knotenart, $parameter
  ) = @_;

  settexthier ("" );

  if ( $knotenart eq "rnode" ) {
    if ( $z eq "" ) {
      paar( $x, $y );
      minmax( $x - 0.5, $y - 0.5, $x + 0.5, $y + 0.5 );
    }
    else {
      paar( x3d( $x, $y, $z ), y3d( $x, $y, $z ) );
      minmax(
        x3d( $x, $y, $z ) - 0.5,
        y3d( $x, $y, $z ) - 0.5,
        x3d( $x, $y, $z ) + 0.5,
        y3d( $x, $y, $z ) + 0.5
      );
    }
  }
  klammer( "\\\" . $knotenart . "{" . $bezeichnung . "}" . $name . " " );
  beschriftungskopf( $layer, $farbe, 0, "c", gettexthier() );
}
else {
  if ( $z eq "" ) {
    paar( $x, $y );
  }
  else {

```

paar(x3d(\$x, \$y, \$z), y3d(\$x, \$y, \$z));	124
}	125
if (\$knotenart eq "cnodeput") {	126
minmax(\$x - 1, \$y - 1, \$x + 1, \$y + 1);	127
}	128
klammer(\$bezeichnung);	129
klammer(\$name);	130
befehlerzeugen(\$layer, \$knotenart, \$farbe, "", "", \$parameter, "",	131
gettexthier ());	132
}	133
}	134
	135
	136
sub knotenverbindung {	137
my (	138
\$layer, \$name1, \$name2, \$text,	139
\$linienart , \$linienstaerke, \$farbe, \$parameter1,	140
\$parameter2, \$pfeilart	141
) = @_;	142
	143
settexthier ("");	144
	145
klammer(\$name1);	146
klammer(\$name2);	147
	148
befehlerzeugen(\$layer, "ncline", \$farbe, \$linienart , \$linienstaerke,	149
\$parameter1, \$pfeilart, gettexthier ());	150
settexthier ("");	151
if (\$text ne "") {	152
klammer(\$text);	153
befehlerzeugen(\$layer, "naput", \$farbe, \$linienart , \$linienstaerke,	154
\$parameter1, "", gettexthier ());	155
settexthier ("");	156
}	157
	158
}	159
	160
sub knotenkreis {	161
my (	162
\$layer, \$name1, \$text, \$winkel,	163
\$radius, \$farbe, \$fuellart , \$fuellfarbe ,	164
\$linienart , \$linienstaerke, \$pfeilart , \$parameter	165
) = @_;	166
	167
minmax(	168
getminmax("x", "min") - 2 * \$radius,	169
getminmax("y", "min") - 2 * \$radius,	170
getminmax("x", "max") + 2 * \$radius,	171
getminmax("y", "max") + 2 * \$radius	172
);	173
	174
settexthier ("");	175
klammer(\$name1);	176
klammer(\$radius);	177
	178
my \$parameterweiter = \$parameter . ", angle=" . \$winkel;	179
befehlerzeugen(\$layer, "nccircle", \$farbe, \$linienart , \$linienstaerke,	180
\$parameterweiter, \$pfeilart, gettexthier ());	181
settexthier ("");	182
if (\$text ne "") {	183
klammer(\$text);	184

```

    befehl erzeugen( $layer, "nbput", $farbe, $linienart, $linienstaerke, "",
        "", gettexthier() );
    settexthier("");
}

# \nccircle [angle=-180]{->}{E}{0.5}\nbput{$0,2$}

}

1;

```

```

185
186
187
188
189
190
191
192
193
194
195

```

Listing A.15: regression.pm

```

#####
# Modul fuer die Regressionsanalyse
# Letzte Aenderung: 29.05.2011
#####
use POSIX qw /floor ceil/;
use Math::Trig;
use strict ;
use warnings;

my $anzahlpunkte = 0;
my @regressionx = ();
my @regressiony = ();
my @regressionz = ();
my ( $xmax, $ymax, $zmax );

$xmax = $ymax = $zmax = 0;

sub oeffneregression {
    my $i = 0;
    for ( $i = 0 ; $i <= 150 ; $i++ ) {
        $regressionx[$i] = 0;
        $regressiony[$i] = 0;
        $regressionz[$i] = 0;
    }
}

sub schreiberegression {
    my ( $x, $y, $z, $n ) = @_ ;
    $regressionx[$n] = $x;
    $regressiony[$n] = $y;
    $regressionz[$n] = $z;
    $xmax = max( $xmax, $x );
    $ymax = max( $ymax, $y );
    if ( istzahl( $z ) == 1 ) {
        $zmax = max( $zmax, $z );
    }
}

sub leseregression {
    my ( $stelle, $n ) = @_ ;
    if ( $stelle eq "x" ) { return $regressionx[$n]; }
    elsif ( $stelle eq "y" ) { return $regressiony[$n]; }
    elsif ( $stelle eq "z" ) { return $regressionz[$n]; }
    else { return "ERROR"; }
}

sub schliesseregression {

```

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47

```

my (\$wert) = @_;	48
\$anzahlpunkte = \$wert;	49
}	50
sub regression {	51
my (\$layer, \$grad, \$linienart, \$linienstaerke, \$farbe, \$parameter) = @_;	52
	53
my (\$xsum, \$ysum, \$i, \$xavg, \$yavg, \$dx, \$dy, \$a, \$b, \$factor, \$offset);	54
	55
\$xsum = \$ysum = \$xavg = \$yavg = \$a = \$b = 0;	56
	57
for (\$i = 1 ; \$i <= \$anzahlpunkte ; \$i++) {	58
\$xsum = \$xsum + \$regressionx[\$i];	59
\$ysum = \$ysum + \$regressiony[\$i];	60
}	61
	62
\$xavg = \$xsum / \$anzahlpunkte;	63
\$yavg = \$ysum / \$anzahlpunkte;	64
	65
for (\$i = 1 ; \$i <= \$anzahlpunkte ; \$i++) {	66
\$dx = \$regressionx[\$i] - \$xavg;	67
\$dy = \$regressiony[\$i] - \$yavg;	68
\$a = \$a + (\$dx * \$dy);	69
\$b = \$b + (\$dx * \$dx);	70
}	71
	72
\$factor = \$a / \$b;	73
	74
\$offset = \$yavg - \$factor * \$xavg;	75
	76
my \$func = "";	77
	78
if (\$offset < 0) {	79
\$func = \$factor . "*x-" . abs (\$offset);	80
}	81
else {	82
\$func = \$factor . "*x+" . \$offset;	83
}	84
	85
graph (\$layer, \$func, 0, 0, "", \$xmax, \$ymax, "", \$linienart,	86
\$linienstaerke, \$farbe, \$parameter . ",algebraic");	87
	88
}	89
	90
1;	91
	92

Literaturverzeichnis

- [tug, 2009] (2009). /PSTricks/main. <http://tug.org/PSTricks/main.cgi/>.
- [BIGALKE und KÖHLER, 2007] BIGALKE, ANTON und N. KÖHLER (2007). Mathematik Sekundarstufe II - Allgemeine Ausgabe: Mathematik, Sekundarstufe II, Allgemeine Ausgabe, Bd.2 : Analytische Geometrie / Stochastik. Cornelsen.
- [BRONŠTEJN und SEMENDJAEV, 1985] BRONŠTEJN, IL'JA und K. A. SEMENDJAEV (1985). Taschenbuch der Mathematik. Teubner [u.a.], Leipzig [u.a.], 22 Aufl.
- [DIV., 2010] DIV. (2010). Säulendiagramm. <http://de.wikipedia.org/wiki/Saeulendiagramm>.
- [ERBRECHT, 2003] ERBRECHT, RÜDIGER (2003). Das große Tafelwerk interaktiv. Formelsammlung für die Sekundarstufen I und II. Westliche Bundesländer: Das große Tafelwerk interaktiv, inkl. CD-ROM: ... Chemie, Biologie - für das Abitur empfohlen.. Cornelsen, 1. Aufl.
- [KRIENKE, 2002] KRIENKE, RAINER (2002). Programmieren in Perl.. Hanser Fachbuchverlag, 2., erw Aufl.
- [VOSS, 2008] VOSS, HERBERT (2008). PSTricks. Lehmanns Media-Lob.de, 5. überarbeitete und erweiterte Aufl.
- [ZANDT et al., 2011] ZANDT, TIMOTHY VAN, M. SHARPE und H. VOSS (2011). pst-node: Nodes and node connections. <http://ctan.org/tex-archive/graphics/pstricks/contrib/pst-node>.

Index

<i>n</i> -Eck.....	26	Parametergleichung.....	95
binomial.....	75	einschluss.....	57
ebene.....	95	Einstellung.....	22
gerade.....	95	einstellung.....	22
kegel.....	101	Ellipse.....	32, 86
kugel.....	99	ellipse.....	32
pyramide.....	100	Ellipsenbogen.....	32
quader.....	99	ellipsenbogen.....	32
r.....	28, 29	farbe.....	25, 84
zylinder.....	100	farbverlauf.....	64
ableitung.....	52	fileplot.....	59
achsen.....	63, 75	folge.....	59
achsenmarkierung.....	63	Funktionenfolge.....	59
achsenschnitt.....	41	Gerade.....	95
ausrichtung.....	73	geschlossen.....	30, 32
Balkendiagramm.....	63, 66	gitter.....	48
balkendiagramm.....	66	Grafik.....	38, 90
balkendiagrammgestapelt.....	67	graph.....	51, 91
barplot.....	61	graphmarkierung.....	51
beide.....	57	graphrot.....	58
Beschriftung.....	34	Grenzen.....	21
beschriftung.....	34, 35	grenzen.....	21
beschriftungsgroesse.....	34	grid.....	49
bezier.....	33, 87	groesse.....	21
Bezierkurve.....	33	groesse.....	68
bild.....	38, 90	integral.....	56
bilddiagramm.....	72	kariert.....	47
binomialstrich.....	76	Kegel.....	101
binomialteil.....	75	Knoten.....	39
binomialteilstrich.....	76	knoten.....	39
binomialverteilt.....	75	Koordinatensystem.....	41
Binomialverteilung.....	75	koordinatensystem.....	41
kumuliert.....	75	Kreis.....	30, 86
dataplot.....	59	kreis.....	30, 39
Dreieck.....	27, 85	Kreisbogen.....	30
rechtwinklig.....	27	kreisbogen.....	30
dreieck.....	27	Kreisdiagramm.....	70
Ebene.....	95	kreisskala.....	31
Koordinatengleichung.....	96	Kreiszyylinder.....	100
Normalengleichung.....	97	Kreuz.....	36, 89

- kreuz.....36, 89
- Kugel.....99
- Kurve.....33, 87
- kurve.....33, 87
- Legende.....74
- legende.....74
- lese.....60
- Linie.....23
- Verschiebung.....83
- linie.....23
- linie.....23, 24, 26, 39, 82
- links.....57
- listplot.....59
- matheheft.....48
- millimeterpapier.....48
- mitte.....41, 63
- neuesbild.....40
- nullwert.....55
- Obersumme.....56
- oberuntersumme.....56
- ohnekopf.....22
- p.....23, 30–32, 82
- Parameterdarstellung.....58
- Parametrierung.....58
- Parametrisierung.....58
- pfeil.....30, 32
- pfeile.....43
- Piktogramm.....72
- Polygon.....25, 84
- polygon.....25, 84
- Punkt.....35, 88
- punkt.....35, 88
- punkte.....49
- Pyramide.....100
- Quader.....99
- r.....23, 30–32, 71, 72
- Rechteck.....27, 28
- rechteck.....27
- relativ.....63
- Ring.....31
- ring.....31
- Ringdiagramm.....71
- ringdiagramm.....72
- Rotationskörper.....58
- Säulendiagramm.....63
- saeulendiagramm.....63
- saeulendiagrammdreieck.....68
- saeulendigrammgestapelt.....67
- schnitt.....57
- Schriftgröße.....22
- schriftgroesse.....22
- Sekante.....54
- sekante.....54
- Sekantendreiecks.....54
- skala.....41
- skalierung.....21
- stammfunktion.....55
- Strichliste.....72, 73
- strichliste.....73
- Tangente.....53
- tangente.....53
- Tangentendreiecks.....53
- text.....23, 30, 82
- ticks.....43
- Tortendiagramm.....70
- tortendiagramm.....70
- uhr.....103
- Untersumme.....56
- Vektor.....25, 84
- vektor.....25, 84
- Viereck.....28, 85
- viereck.....28
- vlinie.....25, 83
- winkel.....38
- xachse.....42
- xachsenpunkt.....44
- xachsenpunktohne.....45
- xachserad.....44
- xpfeil.....46
- xpfeilkasten.....46
- xyachserad.....43
- yachse.....42
- yachsenpunkt.....44
- yachsenpunktohne.....45
- ypfeil.....46
- ypfeilkasten.....46