Gnuplot-Helfer

Jan Krieger

Operatoren: `+-*/`: Standard-Operatoren,
`**`: Potenzen `2**3`=2^3,
`~a`: Einerkomplement,
`!a`: logische Negation,
`a!`: Fakultät,
`& / &&`: bitweises/logisches UND, `^`: bitweises XOR,
`| / ||`: bitweises/logisches ODER,
`%`: Modulo (Rest bei Ganzzahl-Division),
`== / != / >= / <= / < / >`: Vergleichsopertoren,
`(cond)?a:b` C-style Ausfahloperator: if (cond) is true then return a, else return b

Konstantendefinition: Konstanten können mit dem Zuweisungsoperator `=` definiert werden. Die Konstante π ist als `pi` bereits definiert. Die Zahl e erhält man über `e=exp(1)`.

Funktionsdefinition: Funktionen einer oder mehrerer Abhängiger werden ebenfalls mittel `=` definiert:

```
g(x)=a*x+b
f(x,y)= a*x+b*y+2.5e-3
```

Berechnungen (`print`): Mit dem `print`-Befehl können Ergebnisse von Berechnungen auf der Gnuplot-Konsole ausgegeben werden.

Umleitung der Ausgabe/Drucken: Um ein mit Gnuplot erstelltes Diagramm in eine datei zu speichern verwendet man folgende Befehle:

```
set terminal postscript
set output 'filename.ps'
plot-Befehle ...
set terminal windows
```

Der Befehl `set terminal xxx` schaltet die AUsgabe ins Format `xxx` um. Hier sind eine reihe Möglichkeiten gegeben, die man über `set terminal` auflisten kann. wichtig sind folgende: `postscript`, `latex`, `windows`. Das letztere steht für die Ausgabe auf dem Bildschirm. `latex` erzeugt eine `.tex`-Datei (praktisch um kleine Kurven in LaTeX-Dokumente einzufügen. Dies geht dann über den Befehl `\input{file.tex}`). `set output 'filename.ps'` setzt die Datei, in die ausgegeben werden soll. Sie findet sich im aktuellen Arbeitsverzeichnis (Pfadangaben sind nicht erlaubt). Danach folgen die Ausgabebefehel, oder ein `replot`. Das umschalten auf die Bildschirmausgabe zum Schluss schließt die Datei wieder. Gibt man `set output "PRN"` an, so werden die Daten an den Drucker geschickt. Dann muss natürlich auch der richtige Treiber angegeben werden (z.B. `hpljii, pcl5` oä.). Drucken kann man aber auch aus dem Ausgabefenster heraus (Rechts-Klick).

Verändern der Größe der Ausgabe: Mit `set size <Zahl>,<Zahl>` kann man die Größe des Ausgabebereiches anpassen. Die Standardänge/-breite wird dann mit dem angegebenen Zahlenwert multipliziert. `set size 0.5,0.5` nutzt z.B. nur die halbe mögliche Fläche. Dies ist praktisch, wenn man Abbildungen kleiner machen will (vor Allem beim LaTeX-Export).

plotten einfacher Funktionen (2D/3D): 2D-Plots:

```
plot sin(x)
plot sin(x), cos(x)
plot [0:5] [-1:1] exp(-x)*sin(100*x) title "gedaempfter Oszilator"
```

Mit `plot` können Funktionen der Form $f(x)$ gezeichnet werden. Die angabe mehrerer Funktionen ist durch die ABtrennung durch Kommas möglich. Den einzelnen Kurven können Titel zugewiesen werden. Mit `[0:5] [-1:1]` wird der Bereich, in dem das Diagramm gezeichnet wird angegeben. DIe ersten Klammern enthalten den Bereich auf der x-Achse, die zweiten denjenigen der y-Achse. Einzelne Werte können auch frei gelassen werden (z.B. `[:5] [:1]`). Dann werden die entsprechenden alten Werte verwendet bzw. die Standard-Einstellungen.

3D-Plots:

```
splot x**2*y**3
```

Der Befehl `splot` zeichnet Flächen, die aus Stützpunkten über die xy-Ebene bestehen. Die Funktionen folgen also der Form $f(x, y)$.

Daten plotten: Das `plot`-Komando kann auch Daten aus einer Datei einlesen. Diese muss dazu folgendes Format haben:

- Standard-Textdatei
- Zahlen werden in der Form `2.5e-8` ($=2.5 \cdot 10^{-8}$) angegeben, also mit dem Punkt als Dezimaltrennzeichen
- unterschiedliche Spalten in der Datentabelle werden durch Kommas getrennt.
- Kommentare werden mit einem `#` eingeleitet und gehen bis zum Ende der Zeile
- Beispiel:

```
# Leerlaufversuch:
#
#   1. Spalte: U_primr               +/- 0.5 %
#   2. Spalte: U_sekundr/U_primr   +/- 1.0 %
#
10, 0.460
20, 0.475
30, 0.477
40, 0.475
```

Mittels `plot "file.dat" using <datafile-modifiers> <Optionen> with <style>` können die Daten ausgegeben werden. Falls ein Titel zugeordnet werden soll funktioniert das mittels `title "Text"`, wie vorher erklärt. Diese Klausel muss dabei vor

dem `with`-Abschnitt stehen.
Die `with`-Klausel gibt an, wie die Ausgabe erfolgen soll. Je nach dem Parameter `<style>` muss die übergebene Datei eine bestimmte Form haben. Hier einige Möglichkeiten (es sind noch sehr viel mehr definiert, siehe dazu die Online-Hilfe):

`<style>`	**Bedeutung**	**Datei**
`lines`	verbindet die Punkte durch Linien	`<x>, <y>`
`points`	zeichnet einzelne Punkte	`<x>, <y>`
`xerrorbars`/	zeichnet Fehlerbalken nur in x/y-	`<x>, <y>, <xdelta>` bzw.
`yerrorbars`	Richtung	`<x>, <y>, <xlow>, <xhigh>`
`xyerrorbars`	zeichnet Fehlerbalken in x- und y-	`<x>, <y>, <xdelta>, <ydelta>`
	Richtung	bzw. `<x>, <y>,`
		`<xlow>, <xhigh>,`
		`<ylow>, <yhigh>`

Mit der optionalen `using`-Klausel kann angegeben werden, wie das Datenfile einzulesen ist. Dazu wird als Parameter ein Ausdruck der Form `(<Ausdruck>):(<Ausdruck>)...` zu übergeben (Klammern beachten!!!). Als `<Ausdruck>` kommen normale Gnuplot-Ausdrücke incl. aller Funktionen und Operatoren in Frage. Um den Wert aus einer Spalte der Datei einzulesen kann man `($1)`, `($2)`... verwenden, um den Wert in der ersten Spalte, in der zweiten Spalte ... einzufügen.
Beispiele:

- `plot '1a.csv' using ($1):($3):($2):($4) title 'Rg = 274k' with xyerrorbars` gibt Werte mit xy-Fehlerbalken aus. dabei stehen die x-Werte in der ersten, die y-Werte in der dritten und die enstprechenden Fehler in der zweiten und vierten Spalte der Datei `1a.csv`.

- `plot '1a.csv' using (($1)):(1/($3)) with points` gibt Werte als Punkte $\left(x, \frac{1}{y}\right)$ aus.

Um die Werte nicht mit geraden Linien, sondern mit Splines zu verbinden benutzt man unter `<options>` die Anweisung `smooth <Verfahren>`. Als Verfahren kommen in Frage: `unique`, `csplines`, `acsplines`, `bezier`, `sbezier`.
Beispiel: `plot '1a.csv' using (($1)):(1/($3)) smooth bezier` zeichnet die Werte aus der angegebenen Datei (siehe letzter Punkt) und legt durch diese eine Bézier-Kurve.

parametrische Plots: Mit `set parametric`/`set noparametric` setzt man den parametrischen Modus. Die drei Variabeln `t` bzw. `u, v` dienen als Parameter für Kurven (`plot`) bzw. Flächen (`splot`). Ihre Wertebereiche können mit `set trange [0:10]` oä. gesetzt werden. Beispiele:

```
plot sin(t),t**2

splot cos(u)*cos(v),cos(u)*sin(v),sin(u)

x0 = 10
y0 = 10
omega = 628318530717959.0
set urange[0:0.5e-13]
splot x0*cos(omega*u), y0*sin(omega*u), u title "Raum-Zeit-Diagramm eines Rotors"
```

Polar-Koordinaten: Mit `set polar` werden ebene Polar-Koordinaten aktiviert. Der Parameter ist dann statt `x`, `t`. Beispiel:

```
set polar
plot cos(t)*sin(t)
```

Achsen-Beschriftungen uä.: Die Beschriftungen der Achsen erfolgen mit `set xlabel "Name"`, `set ylabel "Name"`, `set zlabel "Name"`. Den Diagramm-Titel setzt man mit `set title "Text"`. Mit `set notitle` wird kein Titel ausgegeben (andere entsprechend. In Strings in doppelten Anführungszeichen kann mit `\n` ein Zeilenumbruch eingefügt werden. Mit `set logscale <axes> <base>` bzw. `set nologscale <axes>` kann eine logarithmische Skala eingeschlatet/ausgeschaltet werden (Bsp: `set logscale x,y` setzt x- und y-Achse auf logarithmische Teilung).
Mit dem Befehl `set xtics <Parameter>` (bzw. `set ytics` und `set ztics`) kann die Achseneinteilung variieren. Als Parameter kommen unter Anderen in Frage (auch Kombinationen sind möglich):

- `axes`/`border`: zeigt die Einteilung auf den Achsen bzw. auf der Umrandung an
- `(<Zahl>, <Zahl>, ...)`: erstellt Achsenbeschriftungen nur an den angegebenen Zahlenwerten (falls diese im sichtbaren Bereich sind)
- `"Name" (<Zahl>, "Name" <Zahl>, ...)`: wie vorheriges, allerdings werden statt den Zahlen die entsprechenden Bezeichner angezeigt (Bsp: `set xtics ("pi" -pi, "-pi/2"`
- `autofreq`: Setzt die angezeigten Beschriftungen automatisch

Mit `set noxtics` können die Achsenbeschriftungen ausgeschaltet werden.

Sample-Rate einstellen/Wie plottet Gnuplot?: Gnuplot berechnet von einer Funktion eine reihe von Stützpunkten und verbindet diese dann durch einen Spline bzw. Linien-Elemente. Besonders bei 3-dimensionalen Plots ist dann die Darstellung in der Standard-Einstellung sehr grob. Abhilfe schafft da `set sample 10000`. Dieser Befehl setzt die Anzahl von Stützpunkten, die berechnet werden.

Fits: Gnuplot kann eine beliebige Funktion an Daten anfitten. Der eingesetzte Algorithmus versuch die Summe der Abstandsquadrate zu minimieren und so die generierte Funktion an die gegebenen Daten anzugleichen. Die Messdaten werden als date übergeben, die im aktuellen Arbeitsverzeichnis liegen muss (Wechesel mit `cd '<Pfad>'`, Pfad mit Slash /, nicht mit Backslash \ !!!). Die Datei enthält Wertepaare/Punkte (x, y), die mit `"."` als Dezimaltrennzeichen und durch Komma `","` getrennt aufgeführt werden. Es wird dann eine Funktion definiert, die einige (undefinierte) Konstanten enthält. Danach wird das Kommando `fit` aufgerufen. danach sind die Parameter entsprechend berechnet. Fehlerabschätzungen gibt Gnuplot nach der Berechnung mit aus. Beispiel:

```
g(x)a*x+b
fit g(x) 'dateiname.dat' via a,b
```

Die Gerade $g(x) = Ax + b$ wird an die Daten in `dateiname.dat` angefittet und zwar über die Parameter a und b. Mittels der `using`-Klausel kann das Format der Datei definiert werden (siehe plotten von Messdaten). Es ist auch möglich mehrdimensionale Funktionen anzupassen.

einige umfangreicherfe Beispiele:

- Ausgabe von Sinus- und Cosinus-Funktion in eine LaTeX-Datei, mit formatierten Achsen. `cd` wechselt das Verzeichnis. Danach werden die Achsenbeschriftungen und sonstigen EInstellungen für das Erscheinungsbild getätigt.

```
cd 'e:\texte\uni\allgemein\formeln_new'
set samples 200
set size 0.6, 0.6
set key Right
set key -pi/2, 0.8
set xtics ("$\\scriptstyle -2\\pi$" -2*pi, "$\\scriptstyle -\\pi$" -pi,\
           "$\\scriptstyle -\\frac{\\pi}{2}$" -pi/2, 0,\
           "$\\scriptstyle\\frac{\\pi}{2}$" pi/2,\
           "$\\scriptstyle\\pi$" pi,\
           "$\\scriptstyle\\frac{3\\pi}{2}$" 3*pi/2,\
           "$\\scriptstyle 2\\pi$" 2*pi)
set ytics (-1, 1)
set noborder
set nogrid
set xtics axis
set ytics axis
set xzeroaxis
set yzeroaxis
set terminal latex roman 8
set output "sincos_fig.tex"
plot [-pi:2*pi][-1:1] sin(x) title "$\\scriptstyle\\sin(x)$",\
                      cos(x) title "$\\scriptstyle\\cos(x)$"
set output "a"
set terminal windows
replot
```

Ergebnis: