UNIVERSITÄT KARLSRUHE (TH)
Institut für Wissenschaftliches Rechnen
und Mathematische Modellbildung (IWRMM)
HDoz. Dr. W. Krämer Dipl.-Math.oec. W. Hofschuster

Numerik mit Ergebnisverifikation III – WS 1998/99

Bedienungsanleitung zum Programm AWA

Das Programm AWA von R. Lohner (siehe [1, 2]) berechnet Einschließungen für die Lösungen von gewöhnlichen **A**nfangs**W**ert**A**ufgaben. Das Programm arbeitet entweder vollständig interaktiv, d. h. die Differentialgleichung, die numerischen Parameter wie Anfangswerte, Startschrittweite und Verfahrensordnung etc. können am Bildschirm eingegeben werden, oder aber es wird eine Textdatei verwendet, welche das zu lösende Problem beschreibt.

1. Eingabe der Differentialgleichung

Das Programm erwartet stets ein System von Differentialgleichungen erster Ordnung. Gleichungen höherer Ordnung müssen also durch die übliche Transformation $u_1 := u$, $u_2 := u'$, ... , $u_n := u^{(n-1)}$ in ein System erster Ordnung überführt werden.

Z.B. werden für eine Anfangswertaufgabe vierter Ordnung mit der Differentialgleichung

$$u^{(4)}(t) = f(t, u(t), u'(t), u''(t), u^{(3)}(t)) \tag{1}$$

(f sei hierbei eine allgemeine nichtlineare Funktion) und den Anfangsbedingungen

$$u(t_0) = u_0,\ u'(t_0) = u_0',\ u''(t_0) = u_0'',\ u^{(3)}(t_0) = u_0^{(3)}$$

hierzu die (Hilfs-)Funktionen

$$u_1(t) := u(t),\ u_2(t) := u'(t),\ u_3(t) := u''(t),\ u_4(t) := u^{(3)}(t)$$

definiert. Dann läßt sich diese Anfangswertaufgabe als System von vier Differentialgleichungen erster Ordnung für die vier Funktionen $u_1(t)$, $u_2(t)$, $u_3(t)$, $u_4(t)$ mit

$$\left.\begin{array}{rcl} u_1'(t) &=& u_2(t) \\ u_2'(t) &=& u_3(t) \\ u_3'(t) &=& u_4(t) \\ u_4'(t) &=& f(t, u_1(t), u_2(t), u_3(t), u_4(t)) \end{array}\right\} \tag{2}$$

und den Anfangsbedingungen

$$u_1(t_0) := u_0,\ u_2(t_0) := u_0',\ u_3(t_0) := u_0'',\ u_4(t_0) := u_0^{(3)}$$

formulieren (die Differentialgleichung sei explizit nach der höchsten Ableitung auflösbar).

Die Lösungskomponente u_1 von (2) entspricht dann der gesuchten Lösung u von (1).

Das Programm AWA erlaubt zur Spezifikation der einzelnen Differentialgleichungen 1. Ordnung folgende Operationen und Funktionen:

> $+, -, *, /$, `sqr`, `sqrt`, `exp`, `log`, `sin`, `cos`, `arctan`, `sinh`, `cosh`, sowie `^` (Exponentiation mit konstanten Exponenten) und die Funktion `sico` (im Zusammenhang mit `sin`, `cos`, `sinh`, `cosh`, siehe Syntaxdefinition weiter unten).

Alle Konstanten werden intern als Intervalle dargestellt, weshalb bei der Eingabe alle Konstanten auch Intervalle sein dürfen. Die gesuchten Funktionen müssen mit `U1, U2,..., UN` (`N` = max. Dimension) bezeichnet werden und die rechten Seiten der Differentialgleichung mit `F1, F2,..., FN` (alles groß geschrieben). Kommt die Zeit explizit in der Differentialgleichung vor, muß sie mit `T` bezeichnet werden. Wird z. B. die Gleichung $y'' + y = 0$ durch Einführen der dann gesuchten Hilfsfunktionen $u_1 := y$, $u_2 := y'$ in das System

$$\left.\begin{aligned} u_1' &= u_2 \\ u_2' &= -u_1 \end{aligned}\right\} \qquad (3)$$

bzw. vektoriell

$$u' = \begin{pmatrix} u_1' \\ u_2' \end{pmatrix} = f(t,u) = \begin{pmatrix} f_1(t,u_1,u_2) \\ f_2(t,u_1,u_2) \end{pmatrix} = \begin{pmatrix} u_2 \\ -u_1 \end{pmatrix}$$

umgeschrieben, so kann dieses in der Form

```
F1 =  U2
F2 = -U1
;;
```

eingegeben werden.

Bei der Eingabe können jederzeit Hilfsvariablen definiert werden, die als definiert gelten, sobald sie auf der linken Seite einer Zuweisung das erste Mal aufgetreten sind. Das System

$$\begin{aligned} y' &= y + y \cdot z \\ z' &= z - y \cdot z \end{aligned}$$

kann z. B. wie folgt spezifiziert werden

```
y  = U1   z = U2   $ Hilfsvariablen fuer U1,U2 (umbenannt)
yz = y*z           $ Hilfsvariable fuer y*z = U1*U2
F1 = y + yz        $ y' = y + y*z
F2 = z - yz        $ z' = z - y*z
;;                 $ Ende DGL, keine zusaetzlichen Ausgaben
```

In jeder Zeile kann ein Kommentar stehen, der mit dem \$-Zeichen eingeleitet wird und dann bis zum Zeilenende gilt. Dies wurde oben bereits verwendet.

Die Eingabe der Differentialgleichung wird mit einem Strichpunkt beendet. Danach kann eine Liste von zuvor definierten Variablen stehen (durch Leerzeichen getrennt), deren Werte man sich während der Integration ausgeben lassen kann. Z. B. kann man mit

```
  F1 =  U2                        $  Differential-
  F2 = -U1                        $     gleichung
norm = sqrt( sqr(U1) + sqr(U2) )  $  Hilfsvariable
;                                 $  DGL Ende
U1 U2 norm                        $  Ausgabeliste
;
```

die Lösungskomponenten `U1, U2` und die Norm des Vektors (`U1,U2`) der Lösung des Systems (3) ausgeben lassen.

Die Differentialgleichung (inklusive der Angabe der auszugebenen Größen) kann auch mit Hilfe einer Textdatei spezifiziert werden. Beim Programmlauf muß dann der Name dieser Textdatei angegeben werden. Wird stattdessen nur `<CR>` (= RETURN) eingegeben, so wird die interaktive Eingabe der DGL erwartet.

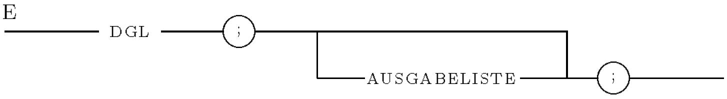
Bemerkungen:

1. Alle Funktionen `F1`, `F2` usw., die nicht explizit definiert werden, werden automatisch mit dem Wert 0 initialisiert.
2. Es gibt nur eine sehr einfache Behandlung von Eingabefehlern; größere DGL-Systeme sollten besser mittels einer Textdatei eingelesen werden. Bei Tippfehlern mit unklaren Konsequenzen empfiehlt sich i. a. ein Neustart des Programms.
3. Namen dürfen keine Sonderzeichen enthalten. Groß- und Kleinbuchstaben werden unterschieden.

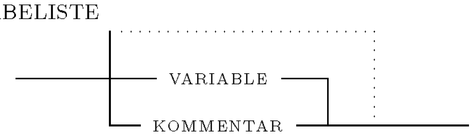
2. Syntaxregeln der Eingabe

Die Syntax der zugelassenen Formelausdrücke ist im folgenden durch Syntaxdiagramme angegeben. Hierin dürfen durchgezogene Linien nur von links nach rechts sowie von oben nach unten und punktierte Linien nur von rechts nach links sowie von unten nach oben durchlaufen werden. Im Eingabetext zu verwendende Zeichen sind mit Kreisen ◯ bzw. abgerundeten Rechtecken ▭ eingerahmt.

EINGABE

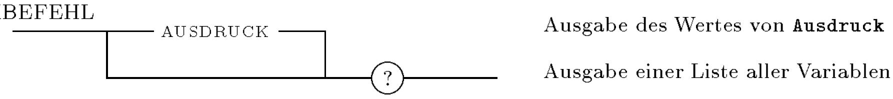


AUSGABELISTE

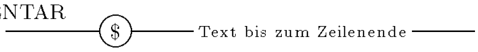


DGL

DRUCKBEFEHL

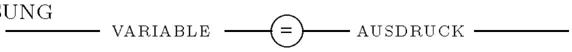


KOMMENTAR



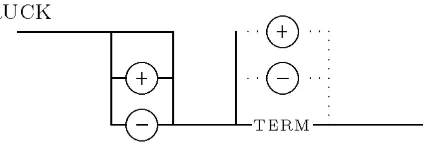
(„;“ innerhalb eines Kommentars wird nicht als Eingabeende gewertet.)

ZUWEISUNG

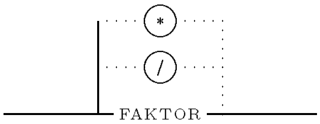


(Bei der Eingabe wird auch `AUSDRUCK = AUSDRUCK` ohne Fehlermeldung akzeptiert, wenn der links vom Gleichheitszeichen stehende Ausdruck mit einer Variablen beginnt. In diesem Fall wird dieser ersten Variablen der rechts vom Gleichheitszeichen stehende Ausdruck zugewiesen, der Rest des links stehenden Ausdrucks wird ignoriert.)

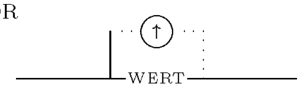
AUSDRUCK



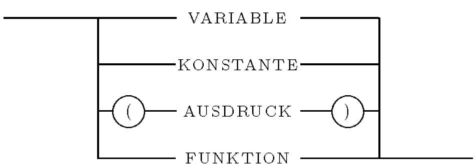
TERM



FAKTOR

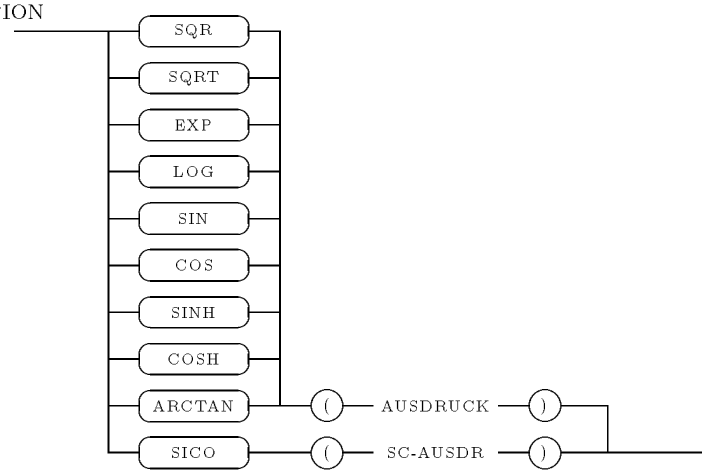


WERT



(Eine `VARIABLE` darf erst dann als `WERT` auftreten, wenn sie mindestens einmal auf der linken Seite einer Zuweisung gestanden hat. Ausnahmen sind die `U1`, ...,`UN`, siehe `RESERVIERTE VARIABLE`.)

FUNKTION



(Alle Funktionsnamen dürfen auch klein geschrieben werden, Groß- und Klein-Buchstaben dürfen sogar beliebig gemischt werden.)

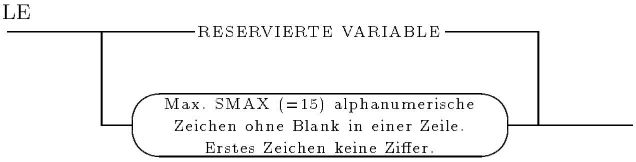
`SC-AUSDR:` Funktionsaufruf von `SIN`, `COS`, `SINH`, `COSH` oder eine Variable, die durch einen solchen Funktionsaufruf definiert wurde.

`sico:` liefert den „Partner-Knoten“ des Arguments, also z. B.

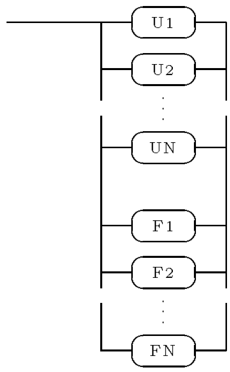
$$\left.\begin{array}{rl} S = & \sin(2*U1) \\ C = & \operatorname{sico}(S) \end{array}\right\} \Rightarrow C = \cos(2*U1)$$

$$\left.\begin{array}{rl} ch = & \cosh(U1*U2) \\ sh = & \operatorname{sico}(ch) \end{array}\right\} \Rightarrow sh = \sinh(U1*U2)$$

VARIABLE



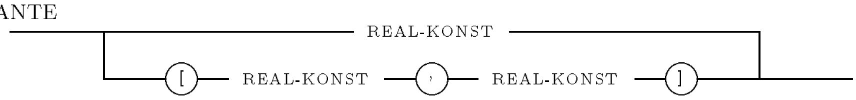
RESERVIERTE VARIABLE



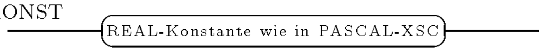
Die U_i, $1 \le i \le N$, stehen für die N gesuchten Funktionen und dürfen nicht auf der linken Seite einer Zuweisung stehen.

Die F_i, $1 \le i \le N$, stehen für die N Funktionen der rechten Seite der DGL und dürfen wie normale Variablen verwendet werden. Zu Beginn der Eingabe sind alle F_i mit dem Wert 0 initialisiert. (Dies gilt auch nach jedem Teilabschnitt, wenn während der Integration die DGL geändert wird.)

KONSTANTE



REAL-KONST



3. Eingabe der Anfangswerte und numerischen Parameter

Nach der Eingabe von “;“ am Ende der (evtl. leeren) Ausgabeliste bzw. nach dem Einlesen der DGL von einem File, fragt das Programm nach, ob die nun folgende Ein- und Ausgabe auf einer Datei protokolliert werden soll. Wird dies gewünscht, so ist anschließend noch ein Filename für diese Protokolldatei einzugeben. Daran anschließend werden dann die numerischen Parameter sowie der Anfangsvektor erwartet. Zunächst sind einzugeben:

aus Ausgabeparameter (`INTEGER` ≥ 1); dieser Wert gibt an, nach wieviel Integrationsschritten jeweils eine Ausgabe gemacht werden soll. $aus = 1$: jeden Schritt ausgeben; $aus = 2$: jeden 2. Schritt ausgeben, etc.

T_{start} Startpunkt der Integration.

T_{end} Endpunkt der Integration.

$h_{initial}$ Startschrittweite mit der die Integration durchzuführen ist. Für $h_{initial} = 0$ versucht AWA selbst eine geeignete Startschrittweite zu bestimmen.

p Verfahrensordnung, d. h. es wird in jedem Zeitschritt eine Taylorentwicklung der Lösung bis zur Ordnung p berechnet.

Danach ist ein Wert für die Steuerung der Einschließungsart einzugeben. Die Werte haben folgende Bedeutung:

0: (Intervallvektor) Die Lösung wird nach jedem Schritt in einen Intervallvektor eingeschlossen und dieser dient als Anfangswert für den nächsten Integrationsschritt. (Ergibt starke Aufblähungen z.B. bei oszillierenden Lösungen.)

1: (Parallelepiped) Die Lösung wird nach jedem Schritt in ein Parallelepiped eingeschlossen, welches als Anfangsmenge für den nächsten Zeitschritt genommen wird. (I. a. gut bei oszillierenden Lösungen, schlecht wenn das Fundamentalsystem schlecht konditioniert ist, was häufig der Fall ist!)

2: (QR-Zerlegung) Die Lösung wird nach jedem Schritt in einen Quader eingeschlossen (nicht notwendigerweise achsenparallel); dieser dient als Anfangsmenge für den nächsten Integrationsschritt. (I. a. auch bei schlecht konditionierten Fundamentalsystemen gut.)

3: Es werden Durchschnitte der Einschließungen der Verfahren 0 und 1 berechnet.

4: Es werden Durchschnitte der Einschließungen der Verfahren 0 und 2 berechnet.

Als nächstes kann gewählt werden, was auszugeben ist (in jedem aus-ten Schritt):

0: Die Lösungseinschließungen an den Stützstellen

7: Es werden Einschließungen für die in der Ausgabeliste aufgezählten Variablen ausgegeben

Anmerkung: In der aktuellen Version sind nur die Werte 0 oder 7 zulässig.

Nun ist der Anfangsvektor einzugeben, und zwar in der Reihenfolge
`U1(0), U2(0), ..., UN(0)`. Es können sowohl `REAL`- als auch `INTERVAL`-Werte eingegeben

werden (Intervalle in der üblichen Notation, z. B. $[0.999, 1.001]$). Wurde mindestens ein echtes Intervall eingegeben, so wird noch abgefragt, ob die Überschätzung der Lösungsmenge (komponentenweise) ausgegeben werden soll, was mit `j` oder `n` zu beantworten ist.

Als letztes sind nun Werte zur Kontrolle des absoluten bzw. relativen Fehlers E_a bzw. E_r bei der Schrittweitensteuerung anzugeben. Es handelt sich hierbei um eine heuristische Fehlerkontrolle.

Die Ausgabe der darauf folgenden Integration sieht z. B. für den Parameter 0 (nur Lösungseinschließung) wie in Tabelle 1 aus.

	`t = 0.000000000000E+000`		
$u_1(t)$	`[ 1.00000000000000E+000, 1.00000000000000E+000]`	`0.00E+000`	`0.0E+000`
$u_2(t)$	`[ 0.00000000000000E+000, 0.00000000000000E+000]`	`0.00E+000`	`---`
	`h = 5.395622253417969E-001`		
	`t = 5.395622253417969E-001`		
$u_1(t)$	`[ 8.60442337839063E-001, 8.60442337839064E-001]`	`2.23E-016`	`2.6E-016`
$u_2(t)$	`[-5.00147339760091E-001,-5.00147339760089E-001]`	`2.23E-016`	`4.5E-016`
	`h = 6.047134399414062E-001`		
	`t = 1.144275665283203E+000`		
$u_1(t)$	`[ 4.34945594914051E-001, 4.34945594914053E-001]`	`4.45E-016`	`1.1E-015`
$u_2(t)$	`[-8.60300683789632E-001,-8.60300683789630E-001]`	`6.67E-016`	`7.8E-016`
	`h = 5.856504440307617E-001`		
	Jeweils: Zeitpunkt, Einschließung der Lösung $u(t)$ (wobei jede Komponente u_i in einer neuen Zeile steht), aktuelle Schrittweite h	Durchmesser der Einschließungen	Oberschranke des relativen Fehlers

Tabelle 1: Beispiel eines Ausgabeprotokolls

Ist die Integration bei $t = T_{end}$ angekommen, so kann man sie mit neuen numerischen Parametern weiterlaufen oder wieder mit einem neuen Startvektor bei $t = T_{start}$ beginnen lassen. Läßt man die Integration weiter laufen, so hat man die Möglichkeit, eine neue Differentialgleichung einzugeben, die dann mit der augenblicklichen Einschließungsmenge als Anfangsmenge integriert wird. Integriert man nicht weiter und gibt auch keinen neuen Anfangsvektor ein, so kann eine neue Differentialgleichung eingegeben werden, was einem Neustart des Programms entspricht. Gibt man auch keine neue Differentialgleichung ein, so wird das Programm beendet.

4. Hinweise

1. Ist a priori nicht klar, welche Einschließungsart zu wählen ist (0,...,4), so ist i. a. 4 (d. h. Durchschnitte von 0 und 2) oder 3 (d. h. Durchschnitte von 0 und 1) eine geeignete Wahl.

2. In früheren Versionen von AWA war nur eine konstante Schrittweite zugelassen. Jetzt besitzt AWA eine automatische Schrittweitensteuerung. Hierbei ist eine Startschrittweite $h_{initial}$ anzugeben. Falls der Wert $h_{initial} = 0$ vorgegeben wird, versucht AWA selbst eine geeignete Startschrittweite zu finden. In bestimmten Fällen ist dies nicht möglich und es kommt dann zu einem Programmabbruch. Hier muß dann unbedingt eine Startschrittweite explizit angegeben werden (die Vorgabe einer sehr kleinen Schrittweite von z.B. 10^{-300} kann sinnvoll sein, wenn man das Schaubild einer Lösungskurve erzeugen möchte (siehe auch Abschnitt 6)).

3. Eine Vorgabe $T_{end} < T_{start}$ ist erlaubt. Es wird dann von „rechts nach links“ integriert. die Startschrittweite wird auch in diesem Fall nicht negativ vorgegeben.

4. Nach der Eingabe der Differentialgleichung (vor Eingabe der numerischen Parameter) fragt das Programm nach, ob die Ausgabe auf eine Datei protokolliert werden soll. Wird die Frage bejaht, so muß anschließend ein Dateiname für dieses Protokoll angegeben werden. Diese Ausgabedatei enthält dann alle Daten, die während der Integration eingegeben und berechnet wurden. Lediglich die Differentialgleichung selbst erscheint nicht in diesem Protokoll. Werden während eines Programmlaufes mehrere Differentialgleichungen eingegeben, deren Ergebnisse protokolliert werden sollen, so muß für jede Gleichung ein neuer Protokoll-Filename eingegeben werden, da sonst die bereits bestehende Datei gelöscht würde.

5. Beispiel: Lorenzsche Differentialgleichung

Das dreidimensionale Differentialgleichungssystem

$$\begin{aligned} x'(t) &= \sigma(y(t) - x(t)) \\ y'(t) &= \rho x(t) - y(t) - x(t)z(t) \\ z'(t) &= -\beta z(t) + x(t)y(t) \end{aligned}$$

mit $\sigma, \rho, \beta > 0$ wurde 1963 von E.N. Lorenz als mathematisches Modell einer konvektiven (vorwiegend vertikalen) Strömung, realisiert durch eine von unten erwärmte und von oben gekühlte Flüssigkeit angegeben (siehe z.B. [3]).

Im folgenden ist eine geeignete Eingabedatei für dieses System unter Verwendung der Anfangswerte

$$x(0) := 15,\ y(0) := 15 \text{ und } z(0) := 36$$

angegeben. Beachte, daß alle Zeilen, die mit einem $-Zeichen beginnen, Kommentarzeilen sind.

```
$  Lorenz differential equation :
$
$  x' = sigma*( y - x )
$  y' = rho*x - y - x*z
$  z' = -beta*z + x*y
$
$  sigma,rho,beta > 0
$
$
$ The data after the differential equation (terminated with ;;) means:
$
$ 1 0 20 0 16  :             out = 1,        each 1-st step
$                                            will be printed out
$                        T_start = 0,        start of integration interval
$                        T_end   = 20,       end of integration interval
$                      h_initial = 0,        initial step size chosen by AWA
$                           p    = 16,       order of method
$
$ 4             : enclosure method 4, i.e. intersection of interval-vector
$                 and QR-decomposition
$
$ 0             : output values : 0 = enclosures of the function values
$                                     of the solution
$
$ 15 15 36      : initial values for the solution
$                 x(0) = 15
$                 y(0) = 15
$                 z(0) = 36
$                 these initial values are close to the strange
$                 attractor of the Lorenz equations.
$
$ 1e-16 1e-16   : Error tolerances E_a = 1e-16 and E_r = 1e-16
$
```

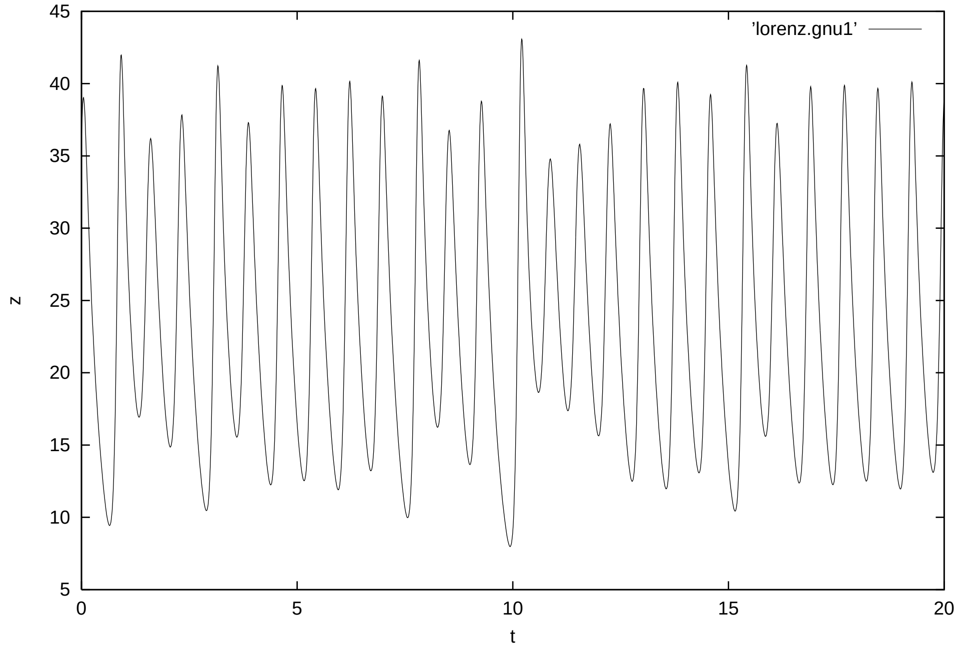
```
$
$ n             : do not continue integration
$ n             : no further initial values
$ n             : no other differential equation
$


x = U1   y = U2   z = U3    $ Renaming of U1, U2, U3

sigma = 10                  $ Auxiliary variables
 beta = 8/3
  rho = 28

F1 = sigma*(y-x)            $ Right hand side of ODE
F2 = (rho-z)*x - y
F3 = x*y - beta*z
;;                          $ End ODE, no additional output

1 0 20 0 16

4 0

15 15 36

1e-16 1e-16

n                           $ do not continue integration
n                           $ no further initial values
n                           $ no other differential equation
```

Die mit diesem Eingabefile erzeugten Ausgabedaten von AWA werden verwendet, um die Graphiken im Abschnitt 6 zu erstellen.

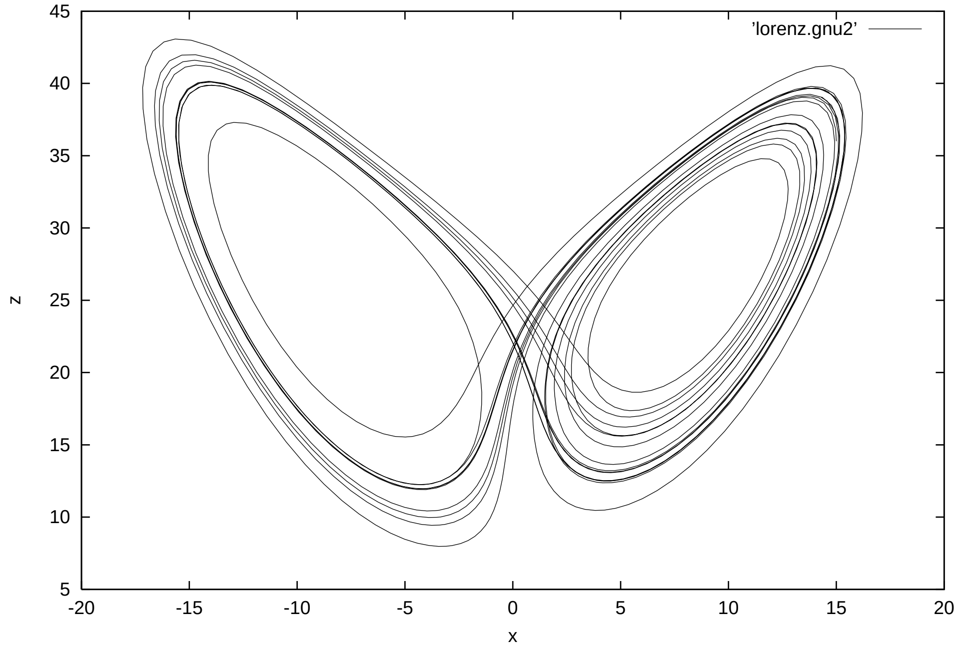
6. Graphische Ausgabe

Mit dem interaktiven Hilfsprogramm `AWAPLOT` können aus den Ausgabeprotokolldateien neue Datenfiles erzeugt werden. Diese Datenfiles können dann zusammen mit Zeichenprogrammen wie z.B. `GNUPLOT` zur graphischen Ausgabe verwendet werden.

Dies wird anhand der Lösungsfunktion der Lorenzschen Differentialgleichung (siehe Abschnitt 5) erläutert.



z-Komponente in Abhängigkeit von der Zeit t



Phasendiagramm z-Komponente gegen x-Komponente

Im folgenden wird beispielhaft das Vorgehen zur Generierung der letzten beiden Bilder beschrieben:

Zunächst wird das Programm `AWA` mit der Eingabedatei aus Abschnitt 5 gestartet. Die Ausgaben von `AWA` werden in die Ausgabeprotokolldatei `lorenz.out` geschrieben.

Anschließend wird `AWAPLOT` gestartet, als „input file name“ wird `lorenz.out` eingegeben, als Systemdimension muß in diesem Fall $n = 3$ gewählt werden. Zum Erstellen des ersten Datenfiles wird nun der „output file name“ `lorenz.gnu1` gewählt und für `i` und `j` werden die Werte 0 und 3 verwendet (Plot: Zeit t gegen 3. Koordinatenrichtung z). Zum Erstellen des zweiten Datenfiles wird der „output file name“ `lorenz.gnu2` gewählt und für `i` und `j` werden nun die Werte 1 und 3 verwendet (Plot: 1. Koordinatenrichtung x gegen 3. Koordinatenrichtung z). Danach wird das Programm `AWAPLOT` durch zweimaliges Drücken der Enter-Taste verlassen.

Das Programm `GNUPLOT` wird aufgerufen und mit

```
gnuplot> plot 'lorenz.gnu1' with lines
```

bzw.

```
gnuplot> plot 'lorenz.gnu2' with lines
```

können die entsprechenden Schaubilder ausgegeben werden.

Literatur

[1] Lohner, R.: *Einschließung der Lösung gewöhnlicher Anfangs- und Randwertaufgaben und Anwendungen.* Dissertation, Universität Karlsruhe, 1988.

[2] Lohner, R.: *Einschließungen bei Anfangs- und Randwertaufgaben gewöhnlicher Differentialgleichungen.* in: U. Kulisch (ed.), Wissenschaftliches Rechnen mit Ergebnisverifikation. Reihe Mathematical Research, Vol. 58, Akademie-Verlag, Berlin, 1989, und Vieweg, Braunschweig, 1989.
Speziell: *Praktikum „Einschließungen bei Differentialgleichungen“,* S. 209-223.

[3] Walter, W.: *Gewöhnliche Differentialgleichungen*, Springer Verlag, Berlin, Heidelberg, New York, 6. Aufl., 1996.