

## 5 Numerische Integration von Anfangswertproblemen

Bei einer Differentialgleichung  $n$ -ter Ordnung

$$y^{(n)} = f(x, y, y', y'', \dots, y^{(n-1)})$$

enthält die allgemeine Lösung  $n$  Integrationskonstanten. Es können  $n$  zusätzliche Bedingungen formuliert werden, mit denen diese Integrationskonstanten zu bestimmen sind, so daß sich die spezielle Lösung des durch Differentialgleichung und Zusatzbedingungen beschriebenen Problems ergibt.

Wenn alle Zusatzbedingungen für die gleichen Stelle  $x_0$  gegeben sind (der Funktionswert  $y$  und die Ableitungen bis zur  $(n - 1)$ -ten Ordnung sind an dieser Stelle vorgeschrieben), dann spricht man von einem **Anfangswertproblem** (im Gegensatz zum **Randwertproblem**, bei dem diese Bedingungen für unterschiedliche  $x$ -Werte gegeben sind). Die im Kapitel 3 mit dem Differenzenverfahren gelösten Aufgaben zur Biegetheorie sind typische lineare Randwertaufgaben (Differentialgleichungen **und** Randbedingungen sind linear), für die eine geschlossene Lösung prinzipiell möglich ist, auch wenn die komplizierten praxisnahen Probleme eine numerische Lösung nahelegen.

Für **nichtlineare Differentialgleichungen** ist eine geschlossene Lösung nur in ganz seltenen Ausnahmefällen möglich. Auch Näherungsmethoden wie das Differenzenverfahren sind nicht praktikabel, weil sich sehr große nichtlineare Gleichungssysteme ergeben würden. Wenn die Aufgabe jedoch als Anfangswertproblem formuliert ist, lassen sich auf numerischem Wege brauchbare Näherungslösungen gewinnen. Glücklicherweise sind gerade viele nichtlineare Probleme der Ingenieur-Mathematik Anfangswertprobleme, für die in diesem Kapitel geeignete Näherungsverfahren behandelt werden.

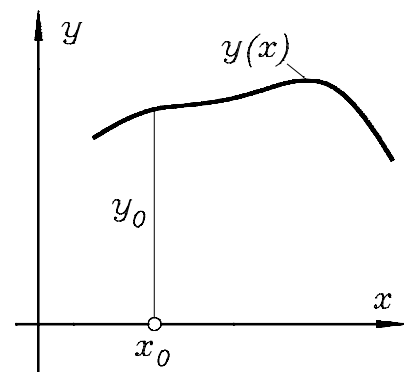
### 5.1 Das Verfahren von EULER-CAUCHY

Die Idee der numerischen Integration soll zunächst am einfachsten Anfangswertproblem mit dem einfachsten Verfahren vorgestellt werden. Für das **Anfangswertproblem 1. Ordnung**

$$y' = f(x, y) \quad , \quad y(x_0) = y_0$$

(eine Differentialgleichung 1. Ordnung und die dazugehörige Anfangsbedingung) ist die Funktion  $y(x)$  gesucht, die die Differentialgleichung und die Anfangsbedingung erfüllt (nebenstehende Skizze).

Ausgehend vom einzigen  $x$ -Wert, für den der gesuchte  $y$ -Wert bekannt ist, dem "Anfangspunkt"  $x_0$  mit dem Wert  $y_0$ , sucht man einen Wert  $y_1$  für die Stelle  $x_1 = x_0 + h$  ( $h$  ist die "Schrittweite"), um anschließend auf gleiche Weise zum nächsten Punkt zu kommen usw.



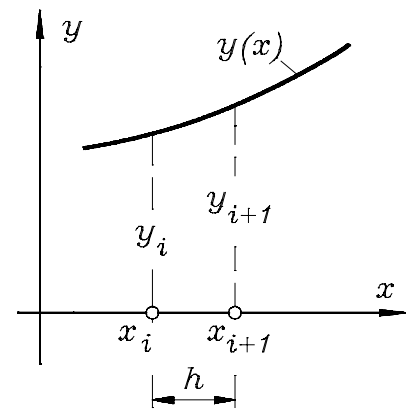
Dieser Prozeß sei bis zur Stelle  $x_i$  abgelaufen,  $y_i$  ist also bekannt. Dann ist das Berechnen von  $y_{i+1}$  für  $x_{i+1} = x_i + h$  der typische Integrationsschritt des Verfahrens.

Beide Seiten der Differentialgleichung des Anfangswertproblems werden über das Intervall  $h$  integriert:

$$\int_{x_i}^{x_{i+1}} y'(x) dx = \int_{x_i}^{x_{i+1}} f(x, y) dx \quad ,$$

$$[y(x)]_{x_i}^{x_{i+1}} = y_{i+1} - y_i = \int_{x_i}^{x_{i+1}} f(x, y) dx \quad ,$$

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} f(x, y) dx \quad .$$



Das verbleibende Integral auf der rechten Seite, das den Zuwachs des Funktionswertes vom Punkt  $i$  zum Punkt  $i+1$  repräsentiert, muß näherungsweise gelöst werden, weil die im Integranden enthaltene Funktion  $y(x)$  nicht bekannt ist. Die verschiedenen Verfahren der numerischen Integration von Anfangswertproblemen unterscheiden sich im wesentlichen in der Art und Qualität, wie dieses Integral angenähert wird.

Die gröbste Näherung für das Integral ist die Annahme, der Integrand  $f(x, y)$  sei im gesamten Integrationsintervall  $x_i \leq x \leq x_{i+1}$  konstant und kann durch den Wert  $f(x_i, y_i)$  am linken Rand des Integrationsintervalls ersetzt werden ( $x_i$  und  $y_i$  sind bekannt). Mit

$$\int_{x_i}^{x_{i+1}} f(x, y) dx \approx [x f(x_i, y_i)]_{x_i}^{x_{i+1}} = f(x_i, y_i) (x_{i+1} - x_i) = y_i' h$$

erhält man die

**Integrationsformel von EULER-CAUCHY:**

$$y_{i+1} = y_i + y_i' h \quad ; \quad x_{i+1} = x_i + h \quad .$$

Dies ist die einfachste Näherungsformel für die numerische Integration eines Anfangswertproblems, die Lösung  $y(x)$  wird durch einen Polygonzug approximiert. Die einfache Berechnungsvorschrift verdeutlicht in besonderer Schärfe das Problem aller Integrationsformeln für Anfangswertprobleme: Die Näherungslösung für das Integral erzeugt einen Fehler ("Quadraturfehler"), der in die Berechnung von  $y'$  für den nächsten Integrationsschritt eingeht und dabei einen weiteren Fehler (Steigungsfehler) erzeugt.

Andererseits wird auch die Stärke dieser Verfahren deutlich: Eine einfache, immer wieder auf die gerade berechneten Werte angewendete Formel kommt der Programmierung in hohem Maße entgegen. Weil jeder Schritt nur die Ergebnisse seines Vorgängers kennen muß, ist der Speicherplatzbedarf außerordentlich gering, so daß das Verfahren selbst auf programmierbaren Taschenrechnern realisierbar ist.

An einem einfachen Beispiel soll das Vorgehen demonstriert werden.

**Beispiel:** Das Problem, die Funktion  $y(x)$  zu bestimmen, die einen Spiegel definiert, der paralleles Licht so ablenkt, daß sich alle Lichtstrahlen in einem Punkt (Fokus) treffen, kann ohne Einschränkung der Allgemeinheit für Lichtstrahlen parallel zur  $x$ -Achse und mit dem Nullpunkt als Fokus formuliert werden.

Einige einfache geometrische Überlegungen (nachfolgende Skizze) führen auf die Differentialgleichung

$$y' = \frac{y}{x + \sqrt{x^2 + y^2}} ,$$

für die (mit einiger Mühe) die allgemeine Lösung berechnet werden kann, so daß sich die nichtlineare Differentialgleichung vorzüglich für eine Abschätzung der Genauigkeit einer numerischen Lösung eignet.

Ihre allgemeine Lösung

$$y^2 = C^2 + 2 C x$$

enthält die Integrationskonstante  $C$ , die mit einer ziemlich willkürlich festzulegenden

Anfangsbedingung bestimmt werden kann. Es gibt unendlich viele Funktionen, die die Forderung der Aufgabenstellung erfüllen, man darf einfach einen Punkt festlegen, durch den die Lösungskurve verlaufen soll. Wählt man z. B. als Anfangsbedingung

$$y(0) = 1 ,$$

dann erhält man mit  $C = 1$  als spezielle Lösung die quadratische Parabel

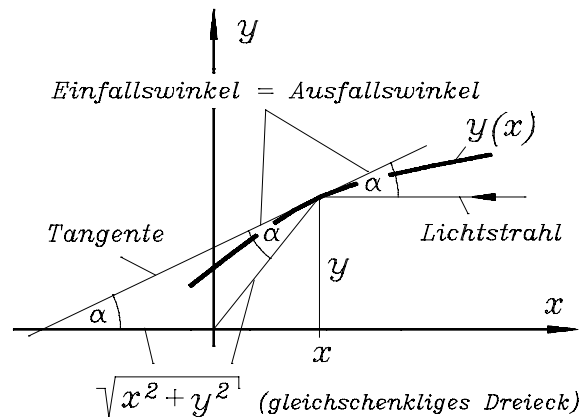
$$y = \pm \sqrt{1 + 2 x}$$

("Parabolspiegel"). Die nachfolgende Tabelle zeigt die numerische Lösung nach Euler-Cauchy für das nichtlineare Anfangswertproblem

$$y' = \frac{y}{x + \sqrt{x^2 + y^2}} , \quad y(0) = 1$$

mit der sehr groben Schrittweite  $h = 0,1$  im Vergleich mit der exakten Lösung:

$i$	$x_i$	$y_i$	$y'_i$	$y_{i, \text{exakt}}$
0	0	1,0000	1,0000	1,0000
1	0,1	1,1000	0,9132	1,0954
2	0,2	1,1913	0,8461	1,1832
3	0,3	1,2759	0,7921	1,2649
...	...	...	...	...
10	1,0	1,7560		1,7321



Schon am Ende des sehr kurzen Integrationsintervalls  $x = 0 \dots 1$  zeigt sich eine sichtbare Abweichung, die bei größeren Integrationsintervallen stärker wird, sich durch kleinere Schrittweiten jedoch verringern läßt. Die nachfolgende Tabelle zeigt die Ergebnisse, die sich bei verschiedenen Schrittweiten am Ende des Integrationsintervalls  $x = 0 \dots 5$  ergeben:

Schrittweite	$h =$	1,0	0,1	0,01	0,001	Exakte Lösung
Integrationsschritte	$N =$	5	50	500	5000	
$y(5) =$		3,9163	3,3723	3,3221	3,3172	3,3166

Natürlich kann man die Anzahl der Integrationsschritte nicht beliebig erhöhen, weil mit der Anzahl der Rechenoperationen auch der mit jeder Operation unvermeidlich verknüpfte Rundungsfehler das Ergebnis verfälschen wird.

Das nachfolgend angegebene kleine FORTRAN-Programm, mit dem diese Berechnung realisiert werden kann, zeigt, wie einfach das Verfahren zu programmieren ist:

```

C      ***** EULER-CAUCHY *****
PROGRAM ECSPGL
C      =====
      double precision  x0 , y0 , h , x , y , f
      integer          n , i
      print * , char (27) , '[2J'
      print * , 'Berechnung eines Anfangswertproblems nach EULER/CAUCHY'
      print * , '=====
      print *

100 print '(a$)' , ' Anfangsbedingung:      x0 = '
      read (* , * , ERR = 100) x0
200 print '(a$)' , '                          y0 = '
      read (* , * , ERR = 200) y0
300 print '(a$)' , ' Schrittweite:          h = '
      read (* , * , ERR = 300) h
400 print '(a$)' , ' Integrationsschritte: n = '
      read (* , * , ERR = 400) n
      print *
      print * , '                          x
      print *
      print * , x0 , y0

      x = x0
      y = y0
      do i = 1 , n
C      **** EULER-CAUCHY-Schritt: ***
          y = y + h * f ( x , y )
          x = x + h
C      *****
          print * , x , y
      enddo
      END

      DOUBLE PRECISION FUNCTION f ( x , y )
      double precision          x , y
C      Spiegel, der paralleles Licht in den Nullpunkt ablenkt:
          f = y / ( x + sqrt ( x*x + y*y ) )
      return
      END

```

Die meisten Programmzeilen beschäftigen sich mit der Eingabe, der gesamte EULER-CAUCHY-Algorithmus besteht aus den fettgedruckten Zeilen des Hauptprogramms. Die spezielle Differentialgleichung, die gelöst wird, ist in einer Zeile der Funktion  $\mathbf{f}(\mathbf{x}, \mathbf{y})$  definiert, die leicht für ein anderes Problem ersetzt werden kann.

- ◆ Bei Berechnung dieses Anfangswertproblem mit negativer Schrittweite (um die Spiegelform auch links vom Fokus zu bestimmen), ist für die Stelle  $x = -0,5$  ein Versagen der Rechnung zu erwarten, weil dort  $y'$  unendlich wird. Durch die unvermeidlichen Rundungsfehler bei der Rechnung äußert sich dies unter Umständen "nur" durch unsinnige Ergebnisse.

Dies ist ein generelles Problem bei der Lösung von Differentialgleichungen in Bereichen, in denen  $y' = f(x, y)$  sich "mit  $y$  sehr stark ändert". Für die Eindeutigkeit der Lösung einer Differentialgleichung muß gefordert werden, daß in dem betrachteten Bereich die partielle Ableitung von  $f(x, y)$  nach  $y$  entsprechend

$$\left| \frac{\partial f(x, y)}{\partial y} \right| \leq k$$

begrenzt ist (sogenannte "LIPSCHITZ-Bedingung").

Im Scheitelpunkt der betrachteten Lösungsfunktion ist diese Bedingung nicht erfüllt. Die exakte Lösung verzweigt sich dort auch in einen oberen und einen unteren Zweig.

- ◆ Das Verfahren von EULER-CAUCHY wird nur aus didaktischen Gründen hier so ausführlich behandelt. Für die weitaus meisten praktischen Probleme gibt es keinen Grund, nicht eines der im Abschnitt 5.3 behandelten genaueren Verfahren zu verwenden.

## 5.2 Differentialgleichungen höherer Ordnung, Differentialgleichungssysteme

Die EULER-CAUCHY-Formel ist problemlos auf Differentialgleichungssysteme 1. Ordnung übertragbar, indem sie für jede der zu berechnenden Funktionen aufgeschrieben wird. Für ein Anfangswertproblem mit zwei Differentialgleichungen 1. Ordnung

$$\begin{aligned} y_1' &= f_1(x, y_1, y_2) & , & & y_1(x_0) &= y_{1,0} & , \\ y_2' &= f_2(x, y_1, y_2) & , & & y_2(x_0) &= y_{2,0} \end{aligned}$$

wird die EULER-CAUCHY-Formel in jedem Integrationsschritt zweimal verwendet, so daß  $y_{1,i+1}$  und  $y_{2,i+1}$  aus  $y_{1,i}$  und  $y_{2,i}$  berechnet werden können.

Damit ist auch eine Möglichkeit der numerischen Integration von Differentialgleichungen (und Differentialgleichungssystemen) höherer Ordnung gegeben: Durch Einführen von zusätzlichen Variablen für die ersten Ableitungen werden Differentialgleichungen höherer Ordnung in ein Differentialgleichungssystem 1. Ordnung überführt.

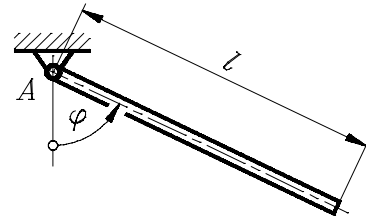
Dies soll am Beispiel einer einfachen Bewegungs-Differentialgleichung demonstriert werden. Bewegungs-Differentialgleichungen der Technischen Mechanik sind stets Differentialgleichungen 2. Ordnung (Beschleunigung ist 2. Ableitung der Wegkoordinate). Für die numeri-

sche Integration bedeutet das, daß neben der Wegkoordinate (z. B.:  $s$  oder bei Drehbewegungen  $\varphi$ ) auch noch die Geschwindigkeit (z. B.:  $v = \dot{s}$  bzw.  $\omega = \dot{\varphi}$ ) als Variable auftritt, so daß das Beschleunigungsglied durch die erste Ableitung der Geschwindigkeit ersetzt wird. Die unabhängige Variable in Bewegungs-Differentialgleichungen ist die Zeit  $t$ .

**Beispiel:**

Ein dünner Stab der Länge  $l$  mit konstantem Querschnitt ist an einem Ende reibungsfrei gelagert. Er wird aus der vertikalen Lage um den Winkel  $\varphi_0$  ausgelenkt und ohne Anfangsgeschwindigkeit freigegeben. Die freie Schwingung wird durch das Anfangswertproblem

$$\ddot{\varphi} = -\frac{3g}{2l} \sin \varphi ; \quad \varphi(t=0) = \varphi_0 ; \quad \dot{\varphi}(t=0) = 0$$



beschrieben (die unabhängige Variable  $t$  taucht in der Differentialgleichung gar nicht auf, dies ist sehr häufig bei Bewegungs-Differentialgleichungen). Durch Einführen einer zusätzlichen abhängigen Variablen  $\omega$  wird aus der Differentialgleichung 2. Ordnung ein Differentialgleichungssystem 1. Ordnung:

$$\begin{aligned} \dot{\omega} &= -\frac{3g}{2l} \sin \varphi & ; & \quad \omega(t=0) = 0 & ; \\ \dot{\varphi} &= \omega & ; & \quad \varphi(t=0) = \varphi_0 & . \end{aligned}$$

Dieses Anfangswertproblem kann zum Beispiel mit dem folgenden EULER-CAUCHY-Formelsatz berechnet werden (zum Problem passend ist die Zeit  $t$  die unabhängige Variable, an Stelle der Schrittweite  $h$  wird  $\Delta t$  geschrieben und für  $y_1$  und  $y_2$  werden  $\omega$  bzw.  $\varphi$  verwendet):

$$\begin{aligned} \omega_{i+1} &= \omega_i + \Delta t \dot{\omega}_i , \\ \varphi_{i+1} &= \varphi_i + \Delta t \dot{\varphi}_i , \\ t_{i+1} &= t_i + \Delta t . \end{aligned}$$

Die Programmierung ist im Vergleich zu einem Problem mit nur einer Differentialgleichung nicht nennenswert aufwendiger (siehe Listing auf der folgenden Seite).

Die Frage, wie man die Ergebnisse bei Problemen kontrollieren kann, für die keine exakten Vergleichslösungen vorliegen (ist der Regelfall, denn sonst bräuchte man ja nicht numerisch zu integrieren), ist ebenso schwierig zu beantworten wie die Frage nach einer geeigneten Schrittweite. An diesem Beispiel sollen einige Möglichkeiten aufgezeigt werden:

- ◆ Man sollte mehrere Rechnungen mit unterschiedlichen Schrittweiten durchführen. Wenn sich die Ergebnisse am Ende des Integrationsintervalls bei halbierter Schrittweite nicht wesentlich ändern, kann man der Rechnung vertrauen.
- ◆ Für sehr kleine Ausschläge kann die Differentialgleichung linearisiert und exakt gelöst werden. Für die Dauer einer vollen Schwingung erhält man die Formel

$$T = \frac{2\pi}{\sqrt{\frac{3g}{2l}}} = 2\pi \sqrt{\frac{2l}{3g}} .$$

Man sollte eine Testrechnung mit einer kleinen Anfangsauslenkung durchführen und die Ergebnisse vergleichen.

```

C      ***** EULER-CAUCHY (Pendelschwingung) *****
PROGRAM ECPNDL
C      =====
double precision  t0 , phi0 , omega0 , dt ,
+                t , phi , omega , phip , omegap
integer          n , i
print * , char (27) , '[2J'
print * , 'Berechnung einer Pendelschwingung nach EULER/CAUCHY'
print * , '=====
print *

100 print '(a$)' , ' Anfangsbedingung:      t0      = '
read  (* , * , ERR = 100) t0
200 print '(a$)' , '                        phi0     = '
read  (* , * , ERR = 200) phi0
250 print '(a$)' , '                        omega0    = '
read  (* , * , ERR = 250) omega0
300 print '(a$)' , ' Schrittweite:          Delta-t = '
read  (* , * , ERR = 300) dt
400 print '(a$)' , ' Integrationsschritte:  n       = '
read  (* , * , ERR = 400) n
print *
print * , '                        t                phi' //
+
print * , '                        omega'
print * , t0 , phi0 , omega0

t      = t0
phi    = phi0
omega  = omega0
do i = 1 , n
C      **** EULER-CAUCHY-Schritt: *****
phi    = phi  + dt * phip (t , phi , omega)
omega  = omega + dt * omegap (t , phi , omega)
t      = t    + dt
C      *****
print * , t , phi , omega
enddo
END

C      Physikalisches Pendel (duenner Stab):
DOUBLE PRECISION FUNCTION  phip (t , phi , omega)
double precision          t , phi , omega
  phip = omega
return
END

DOUBLE PRECISION FUNCTION  omegap (t , phi , omega)
double precision faktor , t , phi , omega
parameter (faktor = 1.5d0*9.81d0/0.5d0)
  omegap = - faktor * sin (phi)
return
END

```

- ◆ Da das Schwingungsproblem ohne Berücksichtigung von Bewegungswiderständen (Reibung, Luftwiderstand) behandelt wird, muß das Pendel bei beliebiger Anfangsauslenkung nach jeder vollen Schwingung wieder die Anfangslage erreichen (sehr gutes Indiz für eine "gesunde" Rechnung).
- ◆ Auch für beliebig große Ausschläge läßt sich die Abhängigkeit der Winkelgeschwindigkeit  $\omega$  von der Winkelkoordinate  $\varphi$  nach dem Energiesatz exakt berechnen.

Speziell liefert diese Betrachtung für die Winkelgeschwindigkeit beim Durchgang durch die tiefste Lage des Pendels ( $\varphi = 0$ ) aus

$$m g \frac{l}{2} (1 - \cos \varphi_0) = \frac{1}{2} \left( \frac{1}{3} m l^2 \right) \omega_{max}^2$$

die maximale Winkelgeschwindigkeit

$$\omega_{max} = \sqrt{\frac{3g}{l} (1 - \cos \varphi_0)} ,$$

die mit dem numerisch berechneten Wert verglichen werden kann.

Natürlich kann man nicht bei jeder Aufgabe so viele Testmöglichkeiten finden, man sollte jedoch immer bestrebt sein, die Ergebnisse besonders mit "physikalischen und technischen Überlegungen" zu verifizieren.

## 5.3 Verbesserte Integrationsformeln

### 5.3.1 Prädiktor-Korrektor-Verfahren, das Verfahren von HEUN

Eine Verbesserung der Näherung für das Integral in

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} f(x, y) dx$$

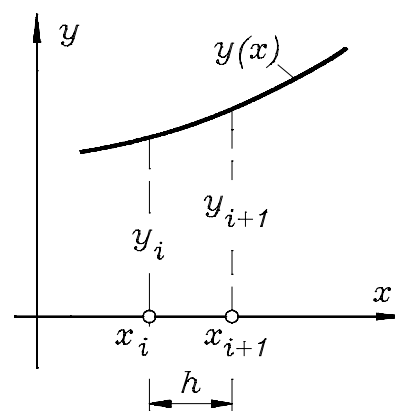
(vgl. Abschnitt 5.1) kann nur durch das Einbeziehen weiterer Punkte des Integrationsintervalls  $h$  erreicht werden. Wenn der Integrand nicht nur durch einen Funktionswert (am linken Rand des Intervalls wie beim Verfahren von EULER-CAUCHY) ersetzt wird, sondern z. B. auch der Funktionswert am rechten Rand  $f(x_{i+1}, y_{i+1})$  in die Näherung einbezogen wird, kann der Integrand als linear veränderliche Größe angenähert werden ("Rechteck"-Näherung wird zur deutlich besseren "Trapez"-Näherung). Mit

$$\int_{x_i}^{x_{i+1}} f(x, y) dx \approx \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1})}{2} (x_{i+1} - x_i) = (y'_i + y'_{i+1}) \frac{h}{2}$$

gelangt man zur wesentlich besseren Integrationsformel

$$y_{i+1} = y_i + (y'_i + y'_{i+1}) \frac{h}{2} ,$$

die allerdings nicht ohne Vorleistung anwendbar ist, denn auf der rechten Seite geht über  $y'_{i+1} = f(x_{i+1}, y_{i+1})$  der Wert  $y_{i+1}$  ein, der mit dieser Formel erst ermittelt werden soll. Man berechnet deshalb einen vorläufigen Näherungswert (**Prädiktor**) nach der EULER-CAUCHY-



Formel, der dann eine (gegebenenfalls mehrfache) Verbesserung nach der verbesserten Formel erfährt (*Korrektor*-Schritte). Dies ist das

### Verfahren von HEUN:

$$\begin{array}{rcl}
 \text{Prädiktor:} & p_{i+1} = y_i + y'_i h & ; \quad x_{i+1} = x_i + h \\
 & \downarrow & \\
 p_{i+1} = y_{i+1} & \Rightarrow & y'_{i+1} = f(x_{i+1}, p_{i+1}) \\
 \uparrow & & \downarrow \\
 \leftarrow \leftarrow \leftarrow \leftarrow & y_{i+1} = y_i + \left( y'_i + y'_{i+1} \right) \frac{h}{2} & \quad (\text{Korrektor})
 \end{array}$$

Das Verfahren kann mit einer festen Anzahl von Korrektorschritten arbeiten oder aber einen Integrationsschritt erst dann beenden, wenn sich  $y_{i+1}$  nicht mehr ändert. Wie das EULER-CAUCHY-Verfahren kann auch das Verfahren von HEUN auf ein System von Differentialgleichungen angewendet werden, indem für jede Differentialgleichung der angegebene Algorithmus ausgeführt wird.

Die Programmierung ist nicht wesentlich aufwendiger als beim Verfahren von EULER-CAUCHY. Für das im Abschnitt 5.2 behandelte Pendelproblem (2 Differentialgleichungen) werden nachfolgend nur die Programmzeilen aufgelistet (Schleife über die  $n$  Integrations-schritte), die sich gegenüber dem Programm **ECPNDL** im vorigen Abschnitt ändern. In jedem Integrationsschritt werden nach dem Prädiktorschritt 3 Korrektorschritte ausgeführt:

```

double precision  phipi , ompi , prphi , prom
...
do i = 1 , n
  phipi = phip (t , phi , omega) ! ... vorab, weil im Integra-
  ompi  = omegap (t , phi , omega) !   tionsschritt konstant
C      **** Praediktor-Schritt: *****
  prphi = phi  + dt * phipi      ! ... nach EULER-CAUCHY
  prom  = omega + dt * ompi
C      **** Drei Korrektor-Schritte: *****
  t     = t     + dt
  do j = 1 , 3
    prphi = phi  + (phipi + phip (t , prphi , prom)) * dt / 2
    prom  = omega + (ompi  + omegap (t , prphi , prom)) * dt / 2
  enddo
C      *****
  phi   = prphi                    ! ... die neuen Werte
  omega = prom
  print * , t , phi , omega
enddo

```

Wenn man mit dem auf diese Weise modifizierten Programm die gleichen Testrechnungen ausführt, die mit dem Programm nach EULER-CAUCHY durchgeführt wurden, stellt man fest, daß schon bei wesentlich groberer Schrittweite die Genauigkeit erreicht wird, die nach EULER-CAUCHY eine sehr feine Intervalleinteilung erfordert.

### 5.3.2 RUNGE-KUTTA-Verfahren, Realisierung in CAMMPUS

Mit den Verfahren von EULER-CAUCHY und HEUN wurden zwei einfache Vertreter einer kaum zu überblickenden Anzahl von Integrationsverfahren vorgestellt, an denen aber die typischen Probleme sichtbar werden, die der Anwender beachten muß. Die verschiedenen Verfahren unterscheiden sich im wesentlichen in der Anzahl der Funktionswertberechnungen für den Integranden in einem Integrationsschritt (beeinflusst die Qualität der Näherung des Integrals), in der Strategie der Berechnung von  $y_{i+1}$  (feste oder variable Anzahl von Operationen) und in der Festlegung der Schrittweiten  $h$  für die Integrationsschritte (feste oder variable Schrittweite).

Auf dieses weite Feld kann hier nicht weiter eingegangen werden. Nachfolgend werden nur noch die Formeln des Verfahrens angegeben, das im Programm MCALCU des Programmsystems CAMMPUS realisiert ist. Es ist ein Vertreter der **RUNGE-KUTTA-Algorithmen**, die aus Funktionswerten für den Integranden in

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} f(x, y) dx$$

(vgl. Abschnitte 5.1 und 5.3.1) an verschiedenen Punkten des Integrationsintervalls  $h$  den Wert für  $y_{i+1}$  am rechten Intervallrand so ermitteln, daß die Genauigkeit der Berücksichtigung möglichst vieler Glieder der Taylorreihen-Entwicklung der Lösung entspricht. Programmiert wurde ein Formelsatz, der für einen Integrationsschritt vier Funktionswerte des Integranden berechnet, das

#### **RUNGE-KUTTA-Verfahren 4. Ordnung:**

$$y_{i+1} = y_i + \frac{h}{6} (k_1 + 2k_2 + 2k_3 + k_4) \quad , \quad x_{i+1} = x_i + h$$

mit

$$k_1 = f(x_i, y_i) \quad ,$$

$$k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2} k_1\right) \quad ,$$

$$k_3 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2} k_2\right) \quad ,$$

$$k_4 = f(x_i + h, y_i + h k_3) \quad .$$

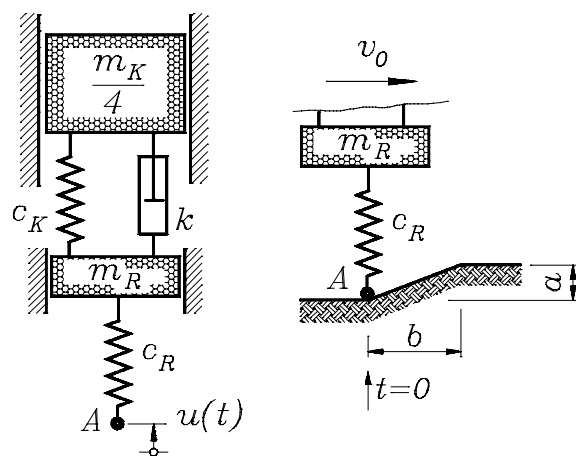
Bei einem Verfahren 4. Ordnung (Übereinstimmung mit den ersten 5 Gliedern der Taylorreihen-Entwicklung) entsteht in jedem Integrationsschritt ein Fehler in der Größenordnung  $h^5$ , der bei genügend kleiner Schrittweite sehr klein ist, andererseits reagiert das Verfahren bei zu großer Schrittweite sehr empfindlich. Gerade für die Runge-Kutta-Algorithmen gibt es eine recht ausgefeilte Theorie zur geeigneten Schrittweitenwahl (und damit auch der Schrittweitenänderung während der Rechnung), die allerdings den gravierenden Mangel hat, daß sie sichere Werte nur dann liefert, wenn die Lösung bekannt ist.

Bei der numerischen Integration eines Anfangswertproblems ist die Wahl einer geeigneten Schrittweite  $h$  ebenso wichtig wie schwierig.

Dem Praktiker kann deshalb als effektives Verfahren zur Beurteilung der Qualität einer Rechnung nur empfohlen werden, eine zusätzliche Kontrollrechnung mit halber Schrittweite (und doppelter Anzahl von Integrationschritten) auszuführen. Die Übereinstimmung beider Lösungen (bei einer vertretbaren Toleranz) ist das sicherste Kriterium für die Bestätigung der Schrittweitenwahl.

Im Anhang B von "Dankert/Dankert: Technische Mechanik, computerunterstützt" findet man vier komplett vorgeführte Berechnungen mit unterschiedlichem Schwierigkeitsgrad (einfachstes Beispiel ist dort das im Abschnitt 5.2 bereits behandelte Pendel), wobei ausführlich die Arbeit mit dem CAMMPUS-Programm MCALCU auch unter dem Aspekt einer geeigneten Schrittweitenwahl beschrieben wird. Deshalb wird hier nur ein spezielles Beispiel behandelt.

**Beispiel:** Für die Analyse der Vertikalschwingungen eines Rades infolge der Bodenunebenheiten und der Übertragung der Schwingungen auf die Karosse dient das skizzierte Berechnungsmodell: Zwischen dem Rad (Masse  $m_R$ ) und der Karosserie befinden sich eine Feder und ein Dämpfungsglied (Stoßdämpfer mit geschwindigkeitsproportionaler Dämpfung), auf denen näherungsweise ein Viertel der Karosseriemasse  $m_K$  lastet. Die Elastizität der Bereifung wird durch die Federzahl  $c_R$  erfaßt. Dem Punkt A wird die Vertikalbewegung  $u(t)$  aufgezwungen.



Es sollen die Vertikalschwingungen des Rades und der Karosserie ermittelt werden (Analyse der Bewegungen für  $t = 0 \dots 4 \text{ s}$ ), wenn der Punkt A zum Zeitpunkt  $t = 0$  eine Aufwärtsbewegung auf einer geneigten Linie (rechte Skizze) beginnt und nach Erreichen der Höhe  $a$  sich horizontal weiter bewegt. Es soll angenommen werden, daß bis zum Erreichen der Höhe  $a$  eine horizontale Strecke  $b$  mit einer konstanten Geschwindigkeit  $v_0$  zurückgelegt wird.

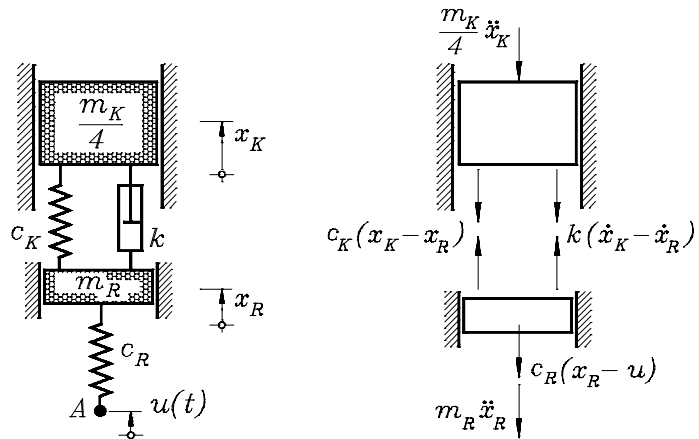
Geg.:  $m_K = 1000 \text{ kg}$  ;  $c_K = 300 \text{ N/cm}$  ;  $k = 750 \text{ kg/s}$  ;  $v_0 = 50 \text{ km/h}$  ;  
 $m_R = 25 \text{ kg}$  ;  $c_R = 900 \text{ N/cm}$  ;  $a = 20 \text{ cm}$  ;  $b = 40 \text{ cm}$  .

Die horizontale Strecke  $b$  wird von dem sich mit der konstanten Geschwindigkeit  $v_0$  bewegendem Punkt A in der Zeit  $\Delta t = b/v_0$  zurückgelegt, die Aufwärtsbewegung erfolgt dabei nach dem Weg-Zeit-Gesetz

$$u(t) = \frac{a}{b} v_0 t \quad \text{für} \quad t \leq \Delta t .$$

Diese Vertikalbewegung wird dem unteren Punkt der unteren Feder aufgezwungen, für  $t > \Delta t$  gilt  $u(t) = a$ .

Die Bewegungskordinaten für das Rad  $x_R$  bzw. die Karosse  $x_K$  (Skizze) werden so festgelegt, daß sie in der statischen Ruhelage des Systems beide den Wert Null haben. Dann stehen die Eigengewichtskräfte mit den Feder-Vorspannungen während des gesamten Bewegungsvorgangs im Gleichgewicht und können bei der Formulierung der Gleichgewichtsbedingungen weggelassen werden (vgl. z. B.: "Dankert/Dankert: Technische Mechanik, computerunterstützt", Beispiel 3 auf Seite 584).



Nach dem Prinzip von d'Alembert liefern die Gleichgewichtsbedingungen der in der rechten Skizze eingezeichneten Kräfte die Bewegungs-Differentialgleichungen:

$$\begin{aligned} \frac{m_K}{4} \ddot{x}_K + c_K(x_K - x_R) + k(\dot{x}_K - \dot{x}_R) &= 0, \\ m_R \ddot{x}_R - c_K(x_K - x_R) - k(\dot{x}_K - \dot{x}_R) + c_R(x_R - u) &= 0. \end{aligned}$$

Als zusätzliche Variablen werden die Geschwindigkeiten  $v_R$  bzw.  $v_K$  eingeführt, und man erhält ein Anfangswertproblem mit vier Differentialgleichungen 1. Ordnung:

$$\begin{aligned} \dot{x}_K &= v_K, & x_K(t=0) &= 0, \\ \dot{v}_K &= -[c_K(x_K - x_R) + k(\dot{x}_K - \dot{x}_R)] / (m_K/4), & v_K(t=0) &= 0, \\ \dot{x}_R &= v_R, & x_R(t=0) &= 0, \\ \dot{v}_R &= [c_K(x_K - x_R) - c_R(x_R - u) + k(\dot{x}_K - \dot{x}_R)] / m_R, & v_R(t=0) &= 0. \end{aligned}$$

Der Bildschirm-Schnappschuß zeigt die Definition des Anfangswertproblems: Als unabhängige Variable wurde **T** eingestellt, auch die übrigen Bezeichnungen wurden der Aufgabenstellung angepaßt, die Anfangsbedingungen wurden auch bereits eingegeben.

Die voreingestellte Anzahl von Integrationsschritten (**500**) wurde bestätigt, eine Kontrollrechnung mit feinerer Schrittweite zeigte, daß 500 Schritte für den Integrationsbereich ausreichend sind.

```

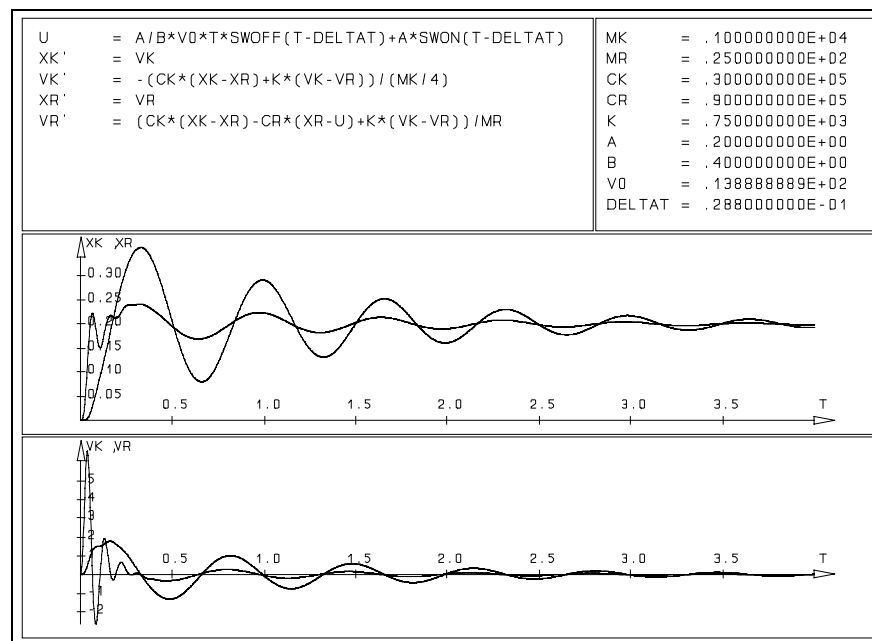
Definition eines Differentialgleichungssystems 1. Ordnung
=====
Anzahl der Differentialgleichungen:      NDGL =    4
Unabhaengige Variable:                  t
Intervall
von T-anf =  0.000000000
bis T-end =  4.000000000                Delta-T =  0.008000000
Anzahl der Integrationsschritte:        NSTEPS =   500
Abhaengige Variablen
(maximal 5 Zeichen):                    Anfangswerte
1 . Variable:  xk                        XK(T-anf) =  0.000000000
2 . Variable:  vk                        VK(T-anf) =  0.000000000
3 . Variable:  xr                        XR(T-anf) =  0.000000000
4 . Variable:  vr                        VR(T-anf) =  0.000000000

Alle Eingaben richtig? (J/N)
    
```

Nach Bestätigung dieser Werte werden die Differentialgleichungen abgefragt. Da in die Differentialgleichung für die Geschwindigkeit des Rades die Funktion  $u(t)$  eingeht, die während der Aufwärtsbewegung des Punktes  $A$  zeitabhängig und danach konstant ist, empfiehlt sich die Annahme des Angebots, "noch eine Funktion vorab zu definieren". Mit den speziellen CAMMPUS-Funktionen *swon* und *swoff* (vgl. Abschnitt 1.2.: "Funktionen, die nur bereichsweise aufschreibbar sind") kann  $u(t)$  folgendermaßen definiert werden:

$$u(t) = \frac{a}{b} v_0 t \text{swoff}(t - \Delta t) + a \text{swon}(t - \Delta t) .$$

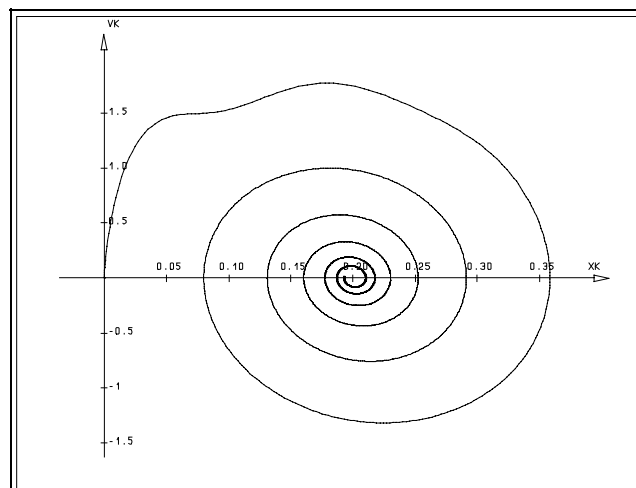
Die nachfolgende HPGL-Ausgabe des Programms zeigt links oben die Funktion  $u(t)$  und die Differentialgleichungen. Im Fenster rechts oben sind die Parameter der Aufgabenstellung zu sehen (man beachte, daß einige umgerechnet wurden, so daß alle Dimensionen nur **kg, m** und **s** enthalten, die Ergebnisse erhält man dann auch mit diesen Dimensionen). In den beiden Graphik-Fenstern sind die Koordinaten beider Massen bzw. deren Geschwindigkeiten (unteres Fenster) dargestellt.



Die plötzliche Aufwärtsbewegung des Rades wird zwar an die Karosse wesentlich verzögert weitergegeben, die dann allerdings mit einem längeren Ausschwingvorgang nach der Art älterer amerikanischer "Straßenkreuzer" reagiert. Durch Veränderung der Feder- und Dämpfungsparameter kann diese Reaktion verändert werden.

Die nebenstehende graphische Darstellung des Geschwindigkeit-Weg-Gesetzes für die Karosse zeigt den typischen spiralförmigen Verlauf gedämpfter Schwingungen.

Die getroffene Aussage über das Schwingungsverhalten wird bestätigt: Nur langsam nähert sich die Spirale ihrer Ruhelage  $v_K = 0$  bei  $x_K = 0,2 \text{ m}$  (Höhe der überfahrenen Schwelle)



### 5.3.3 RUNGE-KUTTA-NYSTRÖM-Verfahren im Programm MOTION

Da Bewegungen wegen der Beschleunigungsglieder in der Regel durch Differentialgleichungen 2. Ordnung beschrieben werden, muß bei der Anwendung der im Abschnitt 5.3.2 beschriebenen RUNGE-KUTTA-Formeln stets eine zusätzliche Variable (Geschwindigkeit oder Winkelgeschwindigkeit) eingeführt werden.

Es ist möglich, aus dem zweimal aufzuschreibenden RUNGE-KUTTA-Formelsatz für die beiden Differentialgleichungen 1. Ordnung, die durch Einführen einer neuen Variablen aus einer Differentialgleichung 2. Ordnung hervorgehen, einen Formelsatz zu bilden, der dann direkt auf Differentialgleichungen 2. Ordnung angewendet werden kann. Die sich auf diesem Weg ergebenden Formeln werden hier im Hinblick auf die Verwendung zur Berechnung von Bewegungen gleich mit der unabhängigen Variablen  $t$  aufgeschrieben. Für ein **Anfangswertproblem 2. Ordnung**

$$\ddot{y} = f(t, y, \dot{y}) \quad , \quad y(t = t_0) = y_0 \quad , \quad \dot{y}(t = t_0) = \dot{y}_0$$

lautet die Berechnungsvorschrift für einen Integrationsschritt nach dem sogenannten

#### **RUNGE-KUTTA-NYSTRÖM-Verfahren 4. Ordnung:**

$$y_{i+1} = y_i + \dot{y}_i \Delta t + \frac{\Delta t}{3} (k_1 + k_2 + k_3) \quad , \quad t_{i+1} = t_i + \Delta t \quad ,$$

$$\dot{y}_{i+1} = \dot{y}_i + \frac{1}{3} (k_1 + 2k_2 + 2k_3 + k_4)$$

mit  $k_1 = \frac{\Delta t}{2} f(t_i, y_i, \dot{y}_i) \quad ,$

$$k_2 = \frac{\Delta t}{2} f\left(t_i + \frac{\Delta t}{2}, y_i + \frac{\Delta t}{2} \dot{y}_i + \frac{\Delta t}{4} k_1, \dot{y}_i + k_1\right) \quad ,$$

$$k_3 = \frac{\Delta t}{2} f\left(t_i + \frac{\Delta t}{2}, y_i + \frac{\Delta t}{2} \dot{y}_i + \frac{\Delta t}{4} k_1, \dot{y}_i + k_2\right) \quad ,$$

$$k_4 = \frac{\Delta t}{2} f(t_i + \Delta t, y_i + \Delta t \dot{y}_i + \Delta t k_3, \dot{y}_i + 2k_3) \quad .$$

- ◆ Das RUNGE-KUTTA-NYSTRÖM-Verfahren ist nur eine Vereinfachung für den Anwender, der Differentialgleichungen 2. Ordnung nicht mehr in zwei Differentialgleichungen 1. Ordnung zerlegen muß. Hinsichtlich der Genauigkeit und der Aussagen zur Schrittweitenwahl gelten die im Abschnitt 5.3.2 für das RUNGE-KUTTA-Verfahren gemachten Aussagen.

Das RUNGE-KUTTA-NYSTRÖM-Verfahren ist die Basis des Programms **MOTION**, das inzwischen etwas in die Jahre gekommen ist, aber immer noch häufig angewendet wird. Es ist auf die Lösung von Bewegungs-Differentialgleichungen spezialisiert. Folgende Vor- bzw. Nachteile der Benutzung von MOTION im Vergleich mit der Benutzung des CAMMPUS-

Programms MCALCU sollten vor der Entscheidung für eines der beiden Programme beachtet werden:

- ◆ MOTION ist kein komplettes lauffähiges Programm. Der Benutzer muß in jedem Fall ein FORTRAN-Unterprogramm ergänzen. In einer

**subroutine f ( t , y , yst , y2st )**

muß die Berechnungsvorschrift für die zweiten Ableitungen der Bewegungskoodinaten programmiert werden. Die Zeit  $t$ , die Werte der Bewegungskoodinaten (im eindimensionalen Feld  $y$ ) und deren Ableitungen (Geschwindigkeiten im eindimensionalen Feld  $yst$ ) werden von MOTION bereitgestellt, abzuliefern sind die Beschleunigungen (auf dem eindimensionalen Feld  $y2st$ ). MOTION unterstützt den Benutzer beim Schreiben des FORTRAN-Unterprogramms: Für einfache Differentialgleichungen wird das Unterprogramm von MOTION automatisch erzeugt, für komplizierte Probleme kann ein Skelett-File angefordert werden.

Für die Benutzung von MOTION ist also ein FORTRAN-Compiler erforderlich, MCALCU aus CAMMPUS ist dagegen ein lauffähiges Programm.

- ◆ Ein mit MOTION erzeugtes Programm arbeitet wesentlich schneller als das Programm MCALCU aus CAMMPUS, das in jedem Schritt die als Strings definierten Differentialgleichungen mühsam mit einem mathematischen Parser auswerten muß. Für komplizierte Probleme ist also in jedem Fall MOTION zu bevorzugen.
- ◆ MOTION bietet die Möglichkeit, die vielen Besonderheiten zu berücksichtigen, die bei der Formulierung praxisnaher Ingenieur-Probleme entstehen, z. B.:
  - Änderung von Problem-Parametern während des Bewegungsvorgangs zu einem nicht vorhersagbaren Zeitpunkt (z. B.: Erreichen einer Bremsstrecke, so daß sich plötzlich der Gleitreibungskoeffizient ändert),
  - Änderung des Bewegungsgesetzes (es gilt eine andere Differentialgleichung, z. B.: Masse löst sich von einer Unterlage, Masse schlägt an einen Puffer an, Übergang von geradliniger zu krummliniger Bewegung, ...),
  - Antriebs- oder Bremskräfte werden temporär zugeschaltet bzw. abgeschaltet,
  - Bewegung-Differentialgleichungen, die in den Beschleunigungsgliedern gekoppelt sind, können durch mehrfachen Aufruf des Gauß-Algorithmus in jedem Integrationsschritt entkoppelt werden, ohne daß der Benutzer mit diesem Problem befaßt wird,
- ◆ Probleme können parametrisiert werden, so daß nach dem Erzeugen eines lauffähigen Programms mit MOTION Variantenrechnungen möglich sind,
- ◆ Ergebnisse können von MOTION auf den Bildschirm (Wertetabellen und graphische Darstellungen) ausgegeben und in Tabellen auf Files gespeichert werden, letztere können über ein Interface-Programm an das File-Format des CAMMPUS-Programms MCALCU angepaßt und in diesem weiterverarbeitet werden.

Für MOTION existiert eine gesonderte Dokumentation mit zahlreichen komplett vorgeführten Beispielen, die im WWW ("World Wide Web") unter URL

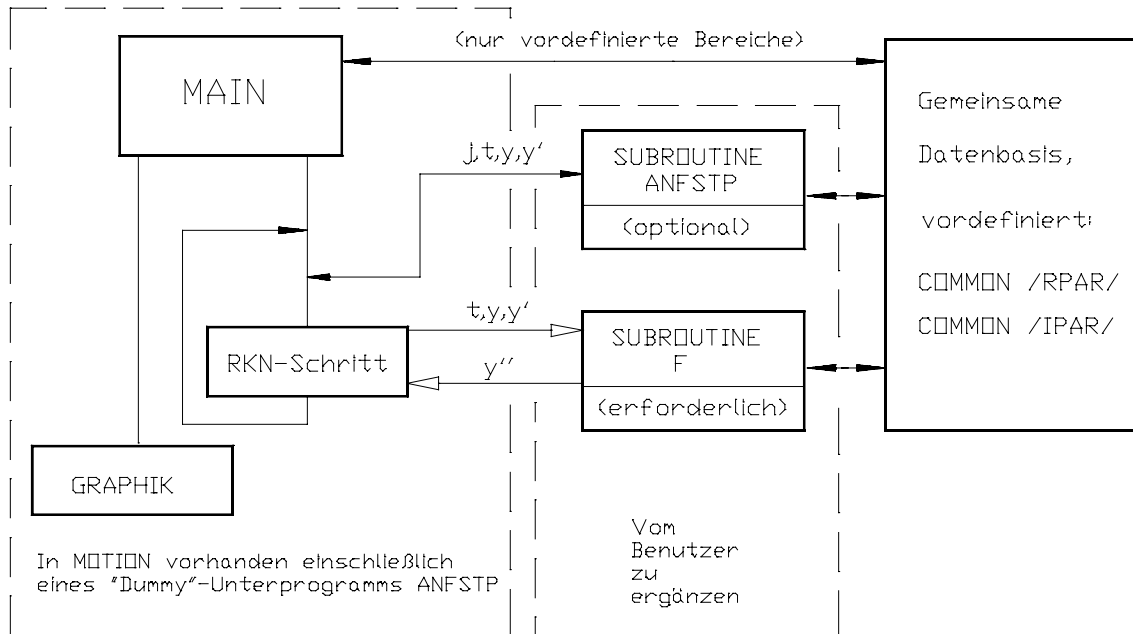
**<http://www.fh-hamburg.de/rzbt/dnksoft/motion>**

zu finden ist. MOTION kann für Ausbildungszwecke frei kopiert werden. Im Novell-Netz des Rechenzentrums Berliner Tor der FH Hamburg kann es mit

### RUN MOTION

gestartet werden.

Die nachfolgende Skizze zeigt das Zusammenwirken der Programmbausteine von MOTION:



Zusammenwirken der Programmbausteine im Programm **MOTION**

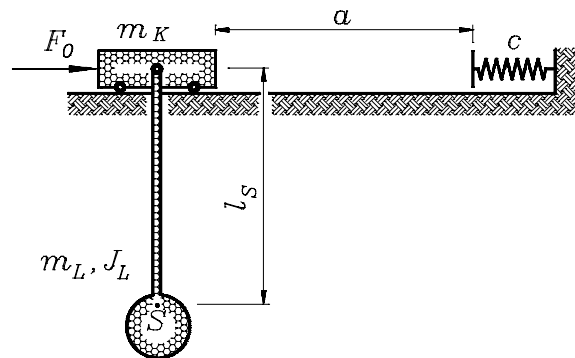
In jedem RUNGE-KUTTA-NYSTRÖM-Schritt wird (mehrfach) das vom Benutzer bereitzustellende Unterprogramm F aufgerufen. Ein Teil der Datenbasis ist in MOTION zur Mitbenutzung im Unterprogramm F vorgesehen, so daß der Benutzer die Individualität seiner Aufgabe durch das von ihm geschriebene Unterprogramm und die in der gemeinsamen Datenbasis untergebrachten Daten in MOTION einbringt.

Eine Besonderheit ist das Unterprogramm **ANFSTP**, für das MOTION eine Dummy-Routine bereitstellt, so daß das Schreiben eines individuellen Unterprogramms ANFSTP optional ist. Es wird von MOTION einmal **vor** jedem RUNGE-KUTTA-NYSTRÖM-Schritt aufgerufen und gestattet damit den Benutzereingriff in den Integrationsprozeß. Die wesentlichen Zustandsdaten (Nummer des Integrationsschritts, Zeit, Bewegungskordinaten und Geschwindigkeiten vor dem nachfolgenden Integrationsschritt) werden von MOTION an dieses Unterprogramm vermittelt, und der Benutzer kann hier steuernd in den weiteren Programmablauf eingreifen.

Die oben aufgezählten "Besonderheiten, die bei praxisnahen Ingenieur-Problemen auftauchen" können fast alle über entsprechende Aktionen des Unterprogramms ANFSTP erfaßt und berücksichtigt werden (siehe nachfolgendes Beispiel).

Hinweis: Das Beispiel ist wohl nur im Zusammenhang mit der MOTION-Dokumentation oder beim Besuch der Vorlesung, in der es behandelt wird, komplett zu verstehen.

**Beispiel:** Eine Laufkatze (Masse  $m_K$ ) trägt eine Last (Masse einschließlich Anhängervorrichtung:  $m_L$ , Massenträgheitsmoment bezüglich des Schwerpunktes  $S$ :  $J_L$ ). In der skizzierten Ruhelage beginnt für eine kurze Zeit  $\Delta t$  die konstante Antriebskraft  $F_0$  zu wirken, die danach wieder abgeschaltet wird. Nach dem Zurücklegen der Strecke  $a$  stößt die Laufkatze auf einen elastischen Puffer (Federzahl  $c$ ).



Man ermittle die Weg-Zeit-Gesetze und die Geschwindigkeits-Zeit-Gesetze für die Laufkatze und die Last für die ersten 10 s der Bewegung.

Geg.:  $m_K = 100 \text{ kg}$  ;  $J_L = 400 \text{ kg m}^2$  ;  $l_S = 4 \text{ m}$  ;  $F_0 = 2000 \text{ N}$  ;  
 $m_L = 500 \text{ kg}$  ;  $c = 200000 \text{ N/m}$  ;  $\Delta t = 1 \text{ s}$  ;  $a = 5 \text{ m}$  .

Mit den Bewegungskoodinaten  $x$  und  $\varphi$  entsprechend nachfolgender Skizze gelten folgende Bewegungs-Differentialgleichungen:

$$(m_K + m_L) \ddot{x} + (m_L l_S \cos \varphi) \ddot{\varphi} = m_L l_S \dot{\varphi}^2 \sin \varphi - c_t(x - a) + F_t ,$$

$$(m_L l_S \cos \varphi) \ddot{x} + (m_L l_S^2 + J_L) \ddot{\varphi} = -m_L g l_S \sin \varphi ,$$

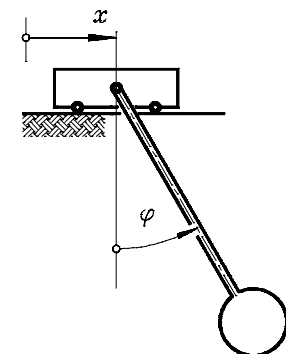
wobei  $c_t$  und  $F_t$  jeweils nur temporär von Null verschieden sind:

$$F_t = F_0 \quad \text{für} \quad t < \Delta t \quad ,$$

$$F_t = 0 \quad \text{für} \quad t \geq \Delta t \quad ,$$

$$c_t = 0 \quad \text{für} \quad x < a \quad ,$$

$$c_t = c \quad \text{für} \quad x \geq a \quad .$$



Dem Programm MOTION werden folgende Unterprogramme im File F.FOR angeboten:

```

C *****
C *   Beispiel: Laufkatze mit pendelnder Last                               *
C *                                                                                   *
C *   J. Dankert                                                                                   *
C *****
C
C   SUBROUTINE      F ( T , Y , YST , Y2ST )
C   IMPLICIT DOUBLE PRECISION ( A-H , O-Z )
C   DIMENSION      Y ( * ) , YST ( * ) , Y2ST ( * )
C
C   DOUBLE PRECISION MK , ML , LS , JL
C   COMMON /RPAR/   MK , ML , LS , JL , DELTAT ,
C   *              A , G , F0 , C , FT , CT
C
C   Die 9 Real-Parameter ML , MK , LS , JL , DELTAT , A , G , F0 , C
C   sind die in der Aufgabenstellung gegebenen Werte (werden in ANFSTP
C   eingelesen), FT und CT sind die tatsaechlich wirkende Kraft (F0 bzw. 0)
C   bzw. die gerade aktuelle Federsteifigkeit (C bzw. 0), die nicht einge-
C   geben werden muessen, weil sie in ANFSTP gesetzt werden.
C
C   Koeffizienten der Beschleunigungsglieder:
C   A11      = MK + ML
C   A12      = ML * LS * COS ( Y ( 2 ) )
    
```

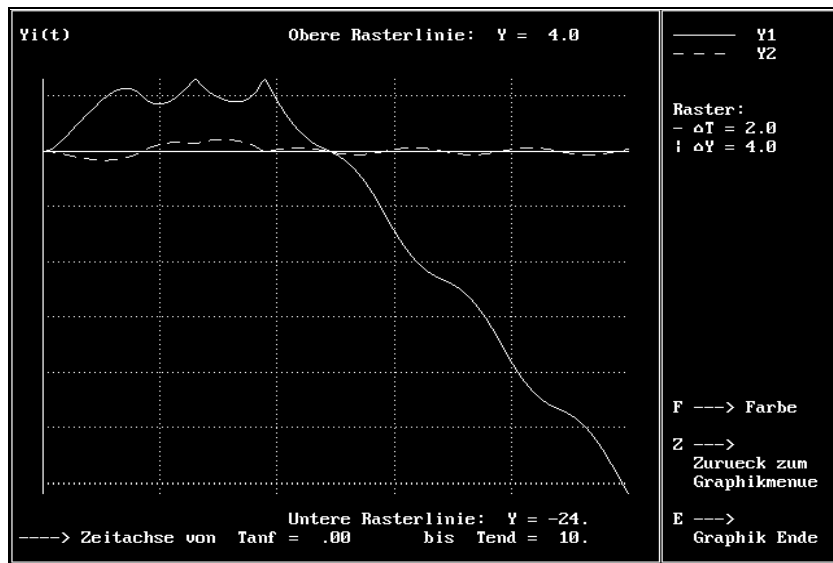
```

      A22      = ML * LS**2 + JL
C   "Rechte Seiten" der Differentialgleichungen:
      B1      = ML * LS * YST(2)**2 * SIN(Y(2)) + FT - CT * (Y(1) - A)
      B2      = - ML * G * LS * SIN(Y(2))
C   Berechnung der Beschleunigungen nach "Cramerscher Regel":
      DETA    = A11 * A22 - A12**2
      Y2ST(1) = (B1 * A22 - B2 * A12) / DETA
      Y2ST(2) = (A11 * B2 - A12 * B1) / DETA
C
      RETURN
      END

      SUBROUTINE ANFSTP (J , T , Y , YST)
      IMPLICIT DOUBLE PRECISION (A-H , O-Z)
      DIMENSION      Y(*) , YST(*)
      DOUBLE PRECISION MK , ML , LS , JL
      COMMON /RPAR/   MK , ML , LS , JL , DELTAT ,
*                   A , G , F0 , C , FT , CT
C
      IF (J .EQ. 1) THEN
C   ... vor dem ersten RKN-Schritt Parameter einlesen:
      PRINT * , CHAR (27) , '[2J'
      PRINT * , ' Laufkatze mit pendelnder Last'
      PRINT * , ' ====='
      PRINT *
10     PRINT '(A$)' , ' Masse der Laufkatze:           MK [kg]      = '
      READ (* , * , ERR = 10) MK
20     PRINT '(A$)' , ' Masse der Last:             ML [kg]      = '
      READ (* , * , ERR = 20) ML
30     PRINT '(A$)' , ' Länge:                       LS [m]       = '
      READ (* , * , ERR = 30) LS
40     PRINT '(A$)' , ' Massenträgheitsmoment          JL [kg m^2] = '
      READ (* , * , ERR = 40) JL
50     PRINT '(A$)' , ' Wirkungszeit der Kraft: DELTA-T [s]   = '
      READ (* , * , ERR = 50) DELTAT
60     PRINT '(A$)' , ' Abstand Laufkatze-Feder:      A [m]       = '
      READ (* , * , ERR = 60) A
70     PRINT '(A$)' , ' Antriebskraft:                F0 [N]      = '
      READ (* , * , ERR = 70) F0
80     PRINT '(A$)' , ' Federsteifigkeit:            C [N/m]     = '
      READ (* , * , ERR = 80) C
      G = 9.81
      PRINT *
      PRINT * , ' Rechnung startet ...'
      PRINT *
      ENDIF
C
      IF (MOD (J , 1000) .EQ. 0)
*         PRINT * , J , '. Runge-Kutta-Nyström-Schritt ...'
C
      IF (T .LT. DELTAT) THEN
C   ... wirkt die Antriebskraft:
      FT = F0
      ELSE
      FT = 0.D0
      ENDIF
C
      IF (Y(1) .GE. A) THEN
C   ... besteht Kontakt zwischen Masse MK und der Feder:
      CT = C
      ELSE
      CT = 0.D0
      ENDIF
C
      RETURN
      END

```

Der nebenstehende Bildschirm-Schnappschuß zeigt die graphische Ausgabe der Bewegungskordinaten (Programm MOTION, Y1 - Laufkatze, Y2 - Winkelkoordinate der Last). Interessant ist, daß es eine Bewegungsumkehr der Laufkatze vor dem Anschlag an die Feder gibt, nach nochmaliger Richtungsänderung stößt die Laufkatze dann **zweimal** an die Feder an, bevor sie sich auf den Rückweg begibt.



Mit dem Konvertierungsprogramm MOT2CALC wurden die von MOTION erzeugten Ergebnisse für das CAMMPUS-Programm MCALCU aufbereitet, um dort nach dem Einlesen (als "Funktionen vom File") noch folgende Auswertungen anzuschließen:

Die kinetische Energie des Gesamtsystems

$$T_{kin} = \frac{1}{2} m_K \dot{x}^2 + \frac{1}{2} m_L v_S^2 + \frac{1}{2} J_L \dot{\varphi}^2$$

mit der Geschwindigkeit des Schwerpunktes der angehängten Last  $v_S$  entsprechend

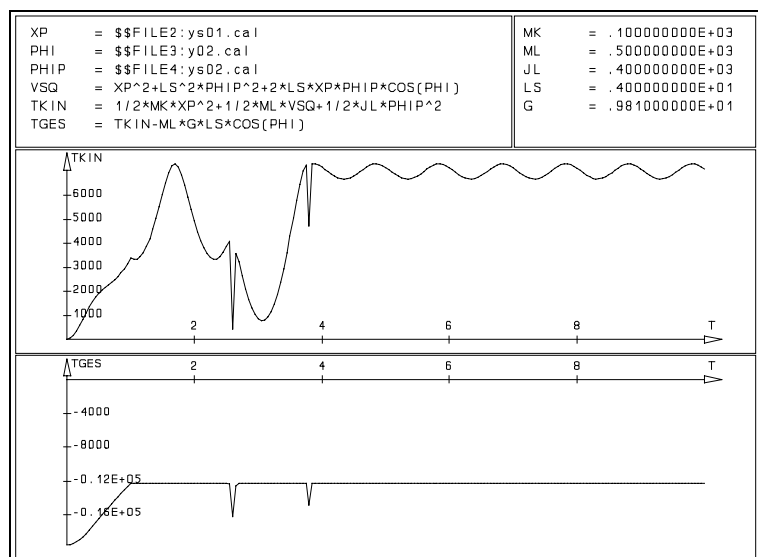
$$v_S^2 = \dot{x}^2 + l_S^2 \dot{\varphi}^2 + 2 l_S \dot{x} \dot{\varphi} \cos \varphi$$

und die Gesamtenergie im bewegten System

$$T_{ges} = T_{kin} - m_L g l_S \cos \varphi$$

(Nullpotential im Schwerpunkt der Laufkatze) sollen für das Intervall  $0 \leq t \leq 10 \text{ s}$  graphisch dargestellt werden.

Die Skizze (HPGL-Ausgabe von MCALCU) zeigt die sehr schön zu interpretierenden Verläufe, z. B.: Die Gesamtenergie in Laufkatze und Last ist nach dem Abschalten der Antriebskraft konstant, wenn nicht gerade an die Feder beim Anstoßen Energie abgegeben wird, die danach von der Feder zurückgegeben wird.



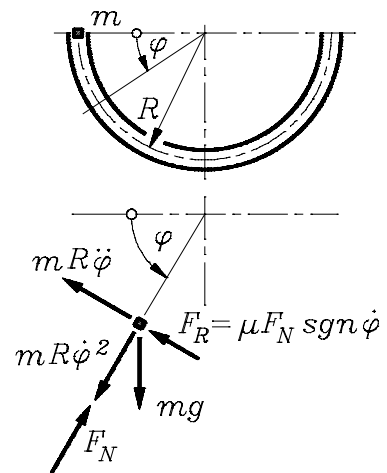
**Aufgabe 5.1:** Ein Massenpunkt  $m$  wird wie skizziert in eine halbkreisförmige Rinne gelegt und ohne Anfangsgeschwindigkeit freigegeben. Das Bewegungsgesetz  $\varphi(t)$  kann aus der Differentialgleichung

$$R \ddot{\varphi} + \mu (R \dot{\varphi}^2 + g \sin \varphi) \operatorname{sgn} \dot{\varphi} - g \cos \varphi = 0$$

ermittelt werden.

Gegeben:  $R = 1 \text{ m}$ ,  $m = 1 \text{ kg}$ ,  $\mu$ .

Gesucht: Graphische Darstellung des Lagewinkels  $\varphi(t)$ , der Winkelgeschwindigkeit des Massenpunktes und der Normalkraft  $F_N(t)$  für die Fälle  $\mu = 0$  und  $\mu = 0,05$ .



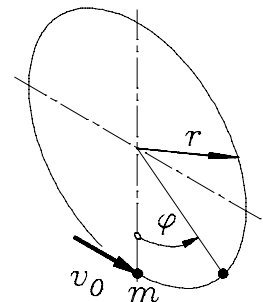
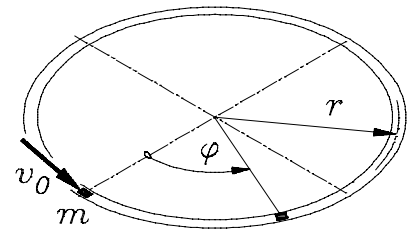
**Aufgabe 5.2:** Ein Massenpunkt  $m$  wird mit der Anfangsgeschwindigkeit  $v_0$  in eine Kreisbahn mit dem Radius  $r$  geschossen.

Gegeben:  $m = 1 \text{ kg}$ ;  $r = 1 \text{ m}$ ;  
 $v_0 = 4 \text{ m/s}$ ;  $\mu = 0,2$ .

a) Die Kreisbahn liegt als Rinne in der Horizontalebene. Reibung ist am Boden der Rinne und am Außenrand zu berücksichtigen.

Man ermittle die graphischen Darstellungen der Funktionen  $\varphi(t)$  und  $\dot{\varphi}(t)$  sowie den Winkel  $\varphi_{end}$  und die Zeit  $t_{end}$  am Ende der Bewegung.

b) Die Kreisbahn liegt in der Vertikalebene. Unter Berücksichtigung von Gleitreibung sind die graphischen Darstellungen der Funktionen  $\varphi(t)$ ,  $\dot{\varphi}(t)$  und der Normalkraft  $F_N(t)$  zu ermitteln.



**Aufgabe 5.3:** Der skizzierte Körper mit konstanter Dicke besteht aus einer Halbkreis- und einer Rechteckscheibe. Er wird aus der vertikalen Lage um einen Winkel  $\varphi_0$  ausgelenkt.

Gegeben:  $R$ ,  $b$ ,  $h$ ,  $g$ ,  $\rho$ ,  $\varphi_0$ .

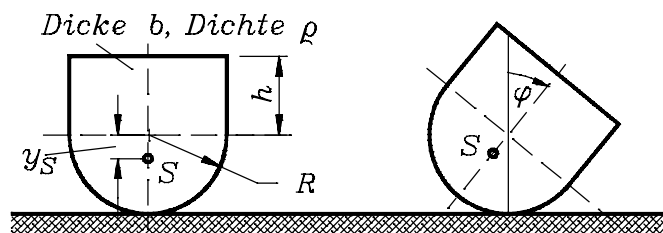
Man ermittle

a) die (Schwerpunktkoordinate)  $y_S$ , das Massenträgheitsmoment  $J_S$  der Scheibe bezüglich des Schwerpunkts und die Gesamtmasse  $m$ ,

b) das Bewegungsgesetz der Scheibe  $\varphi(t)$ , das der Differentialgleichung

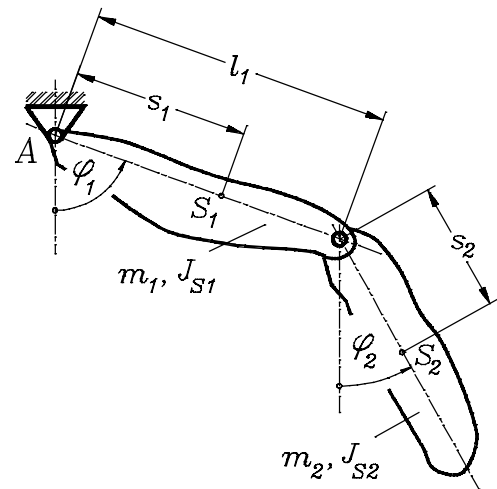
$$\left[ J_S + m (y_S^2 + R^2 - 2 R y_S \cos \varphi) \right] \ddot{\varphi} + m y_S (R \dot{\varphi}^2 + g) \sin \varphi = 0$$

genügen muß, für unterschiedliche Abmessungsverhältnisse  $h/R$  und eine beliebig gewählte Anfangsauslenkung.



**Aufgabe 5.4:** Ein Doppelpendel wird definiert durch die beiden Pendelmassen  $m_1$  und  $m_2$ , die auf die jeweiligen Schwerpunkte bezogenen Massenträgheitsmomente  $J_{S1}$  und  $J_{S2}$ , die Schwerpunktabstände von den Drehpunkten  $s_1$  und  $s_2$  und den Abstand  $l_1$  der beiden Drehpunkte voneinander.

Die freie Schwingung dieses Systems mit zwei Freiheitsgraden wird durch das folgende Differentialgleichungssystem beschrieben (vgl. "Dankert/-Dankert: Technische Mechanik, computerunterstützt", Seiten 634 und 706):

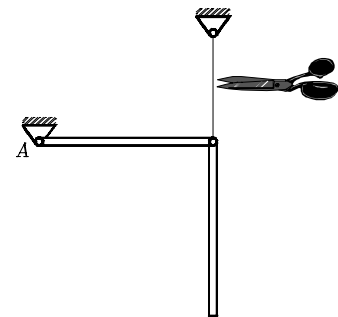


$$\begin{aligned} \left[ \left( \frac{s_1}{l_1} \right)^2 + \frac{J_{S1}}{m_1 l_1^2} + \frac{m_2}{m_1} \right] \ddot{\varphi}_1 + \left[ \frac{m_2 s_2}{m_1 l_1} \cos(\varphi_1 - \varphi_2) \right] \ddot{\varphi}_2 \\ = - \frac{m_2 s_2}{m_1 l_1} \dot{\varphi}_2^2 \sin(\varphi_1 - \varphi_2) - \left( \frac{s_1}{l_1} + \frac{m_2}{m_1} \right) \frac{g}{l_1} \sin \varphi_1 \\ \left[ \frac{m_2 s_2}{m_1 l_1} \cos(\varphi_1 - \varphi_2) \right] \ddot{\varphi}_1 + \left[ \frac{m_2}{m_1} \left( \frac{s_2}{l_1} \right)^2 + \frac{J_{S2}}{m_1 l_1^2} \right] \ddot{\varphi}_2 \\ = \frac{m_2 s_2}{m_1 l_1} \dot{\varphi}_1^2 \sin(\varphi_1 - \varphi_2) - \frac{m_2 s_2}{m_1 l_1} \frac{g}{l_1} \sin \varphi_2 \end{aligned}$$

a) Für das Programm MOTION ist ein Unterprogramm zu schreiben, das dieses Differentialgleichungssystem beschreibt (die o. g. Größen sollen als Problemparameter erst zur Ausführungszeit eingegeben werden).

b) Für den nebenstehend skizzierten Spezialfall (zwei schlanke Stäbe gleicher Masse und gleicher Länge) sind die Funktionen  $\varphi_1(t)$  und  $\varphi_2(t)$  im Intervall  $0 \leq t \leq 10 \text{ s}$  zu ermitteln. Dabei sollen (wie skizziert) die Anfangsbedingungen

$$\begin{aligned} \varphi_1(t=0) &= \pi/2 & ; & & \varphi_2(t=0) &= 0 & ; \\ \dot{\varphi}_1(t=0) &= 0 & ; & & \dot{\varphi}_2(t=0) &= 0 \end{aligned}$$



verwendet werden. In mehreren Programmläufen ist die Anzahl von Zeitschritten zu ermitteln, mit denen die Ergebnisse im betrachteten Intervall ausreichend genau werden. Es sind folgende Zahlenwerte zu verwenden:

$$\begin{aligned} m_2/m_1 &= 1 & ; & & J_{S1}/(m_1 l_1^2) &= 1/12 & ; & & s_1/l_1 &= 1/2 & ; \\ g/l_1 &= 9,81 \text{ s}^{-2} & ; & & J_{S2}/(m_1 l_1^2) &= 1/12 & ; & & s_2/l_1 &= 1/2 \end{aligned}$$