Laboratory 3:

Numerisches Lösen von gewöhnlichen Differentialgleichungen

Ergebnisse sind über den ILIAS-Test im Ordner

Numerische Mathematik (P-Gr. X) > Laboratory 3

bis spätestens zu folgen Terminen einzureichen:

Gruppe 1	**Gruppe 2**	**Gruppe 3**	**Gruppe 4**
Fr., 8.01.2021; 23:59:59		Fr., 15.01.2021; 23:59:59	

Bitte überprüfen Sie, ob die Ergebnisse vollständig hochgeladen wurden und der Test als bestanden gekennzeichnet ist.

Ziel:

In diesem Versuch beschäftigen wir uns mit der numerischen Lösung gewöhnlicher Differentialgleichungen erster und höherer Ordnung sowie Differentialgleichungssystemen erster Ordnung. Wir implementieren einige der gängigen numerischen Verfahren zur Lösung von gewöhnlichen Differentialgleichungen und Differentialgleichungssystemen, vergleichen den Aufwand und die Effektivität einzelner Verfahren.

Wichtig!

Zu diesem Praktikumsversuch werden Funktionsprototypen oder auch Fragmente von MATLAB-Skripten bzw. -Funktionen bereitgestellt. Machen Sie sich zur Vorbereitung zu diesem Praktikumsversuch mit der Programmierung und Arbeitsweise dieser vorgegebenen Programmteile vertraut. Das Verständnis der Funktionalität und Zweck der vorgegebenen Funktionen kann während der Präsentation ihrer Ergebnisse vom Praktikumsbetreuer abgefragt werden. Bereiten Sie alle mit **V** gekennzeichneten Aufgaben vor und bearbeiten Sie alle Fragen, um diese bei der Präsentation ihrer Ergebnisse mit dem Betreuer zu besprechen.

Die korrekte Implementierung jeder Teilaufgabe muss bei der Abgabe dem Praktikumsbetreuer vorgeführt werden. Sie müssen bei der Präsentation ihrer Implementierung und ihrer Ergebnisse nachvollziehend zeigen können, dass Sie mit ihrem Programmcode vertraut sind. Der Praktikumsbetreuer stellt zu diesem Zweck Fragen zur Implementierung der Teilaufgaben und kann mögliche Modifikationen des Programmcodes erfragen. Die Präsentation der Ergebnisse erfolgt für jeden Studierenden einzeln, eine Gruppenpräsentation ist nicht erlaubt. Nach erfolgreicher Präsentation aller Teilaufgaben erhalten Sie das Zwischentestat für das Laboratory 3 in der online Testatverwaltung.

Die Testatverwaltung finden Sie unter der URL:
https://www.testat.etechnik.fh-aachen.de/student_login.php
Login erfolgt mit ihrer FH-Kennung. Siehe hierzu auch die Hinweise auf ILIAS. Bitte kontrollieren Sie regelmäßig, am besten nach jedem Termin, dass die erbrachten Leistungen auch entsprechend im System vermerkt wurden.

Abgabe:

Das Einreichen der im Bericht genannten Graphiken und MATLAB-Skripte und-Funktionen muss bis spätestens zu dem auf der ersten Seite genannten Termin erfolgen.

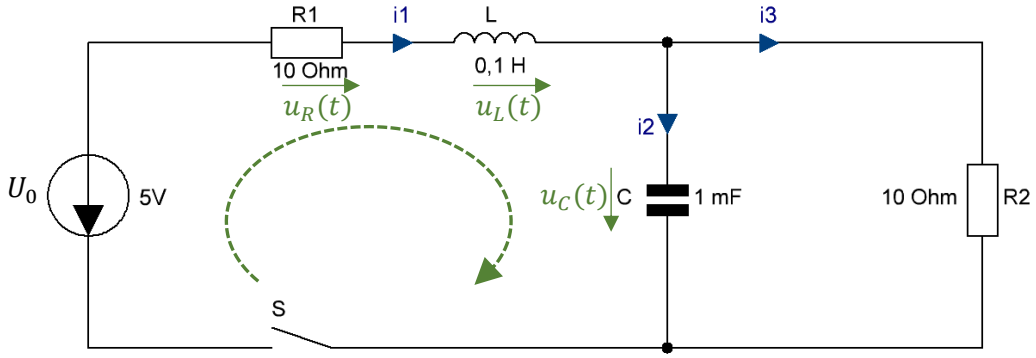
Falls entsprechend der Aufgabenstellung Ergebnisse in dem vorliegenden Dokument notiert werden müssen, muss die Praktikumsbeschreibung zur Präsentation mitgebracht und vorgelegt werden.

Matlab-Funktionen und -Skripte sind als zip-Ordner einzureichen.

Alle Matlab-Skripte müssen ordnungsgemäß kommentiert werden.

Versuch 1. Aufstellen einer Differentialgleichung

Wir betrachten die folgende Schaltung (*Quelle: A. Quarteroni, F. Saleri: „Wissenschaftliches Rechnen mit MATLAB“*):



Zum Zeitpunkt $t = 0$ betätigen wir den Schalter S und starten somit einen Ausgleichsvorgang. Als Ausgleichsvorgang bezeichnet man den Übergang der Schaltung in den stationären Zustand nach dem Schließen des Schalters, also in den Zustand, der sich für $t \to \infty$ einstellt. Der Zustand heißt stationär, da in diesem Zustand die elektrischen Größen wie Strom und Spannung konstant sind, also sich zeitabhängig nicht (mehr) ändern. Als charakteristische Größe für den Ausgleichsvorgang nehmen wir in diesem Fall den zeitlichen Verlauf der Spannung $u_C(t)$ an dem Kondensator C. Dieser Übergang wird mathematisch mithilfe einer Differentialgleichung beschrieben:

Zunächst bestimmen wir die Maschengleichung

$$-U_0 + u_{R_1}(t) + u_L(t) + u_C(t) = 0$$
$$i_1(t) \cdot R_1 + L \cdot \frac{di_1}{dt} + u_C(t) = U_0 \qquad (1)$$

Der Strom $i_1(t)$ sowie dessen erste Ableitung $\frac{di_1}{dt}$ kann durch die Knotengleichung bestimmt werden

$$i_1(t) = i_2(t) + i_3(t)$$

$$i_1(t) = C \cdot \frac{du_C}{dt} + \frac{1}{R_2} \cdot u_C(t) \qquad (2)$$

$$\frac{di_1}{dt} = C \cdot \frac{d^2u_C}{dt^2} + \frac{1}{R_2} \cdot \frac{du_C}{dt} \qquad (3)$$

(2), (3) → (1) ergibt:

$$\left(C \cdot \frac{du_C}{dt} + \frac{1}{R_2} \cdot u_C(t)\right) \cdot R_1 + L \cdot \left(C \cdot \frac{d^2u_C}{dt^2} + \frac{1}{R_2} \cdot \frac{du_C}{dt}\right) + u_C(t) = U_0$$

Nach wenigen Umformungen sieht diese Differentialgleichung wie folgt aus:

$$L \cdot C \cdot \frac{d^2u_C}{dt^2} + \left(C \cdot R_1 + \frac{L}{R_2}\right) \cdot \frac{du_C}{dt} + \left(\frac{R_1}{R_2} + 1\right) \cdot u_C(t) = U_0$$

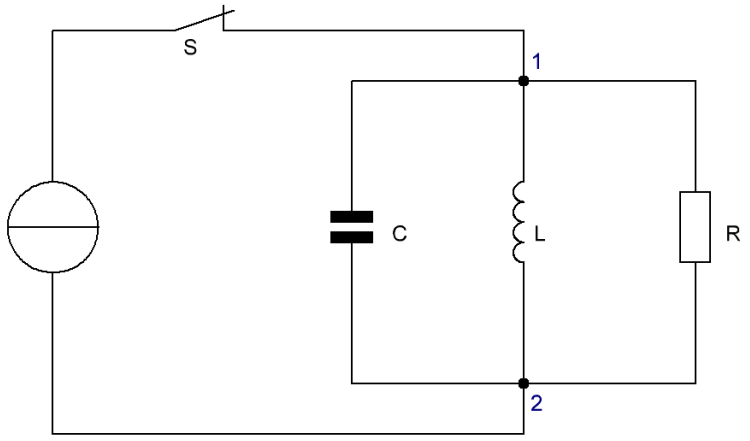
oder in expliziter Form (nach der höchsten Ableitung aufgelöst):

$$\frac{d^2u_C}{dt^2} = -\frac{\left(C \cdot R_1 + \frac{L}{R_2}\right)}{L \cdot C} \cdot \frac{du_C}{dt} - \frac{\left(\frac{R_1}{R_2} + 1\right)}{L \cdot C} \cdot u_C(t) + \frac{1}{L \cdot C} \cdot U_0 \qquad (4)$$

In Formel (4) können Vereinfachungen vorgenommen werden, da bekannt ist, dass $R_1 = R_2 = R$:

$$\frac{d^2u_C}{dt^2} = -\frac{(C \cdot R^2 + L)}{L \cdot C \cdot R} \cdot \frac{du_C}{dt} - \frac{2}{L \cdot C} \cdot u_C(t) + \frac{1}{L \cdot C} \cdot U_0$$

V 1.1 Es soll nun folgender paralleler RLC-Schaltkreis betrachtet werden. Die Stromquelle wird zum Zeitpunkt $t = t_0$ vom Schaltkreis getrennt und der sich nachfolgend einstellende Ausgleichsvorgang soll betrachtet werden. Es soll des Weiteren angenommen werden, dass zum Zeitpunkt $t = t_0 = 0$ die Spannung am Kondensator den Wert $u_C(t = t_0) = 2\,V$ besitzt und $\left.\frac{du_C}{dt}\right|_{t=t_0} = 1\frac{V}{s}$ gilt.



Stellen Sie die Differentialgleichung zur Bestimmung des zeitlichen Verlaufs der Spannung $u_C(t)$ zwischen den Knoten 1 und 2 auf:

V 1.2 Bestimmen Sie nun für die folgenden Werte der Bauelemente der RLC-Schaltung die Differentialgleichungen. Die Differentialgleichung hat die Form

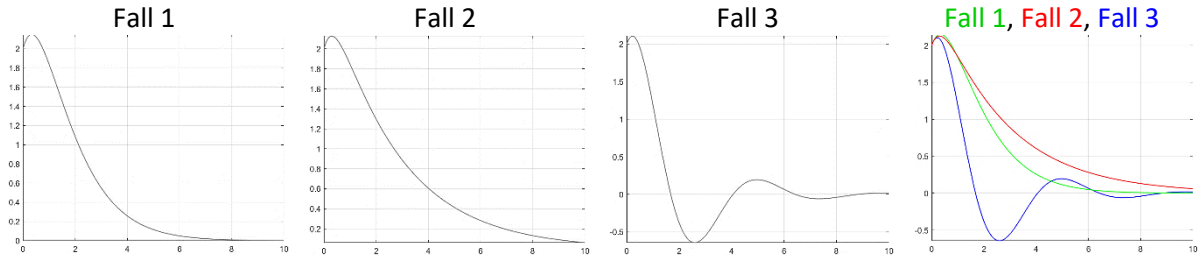
$$\frac{d^2u_C}{dt^2} = a \cdot \frac{du_C}{dt} + b \cdot u_C(t),$$

wobei die Koeffizienten a und b Zahlenwerte darstellen.

Fall 1: Aperiodischer Grenzfall	
Werte	Differentialgleichung
$R = 2{,}5\ \Omega$ $L = 5\ H$ $C = 0{,}2\ F$	
Fall 2: Kriechfall	
Werte	Differentialgleichung
$R = 1{,}667\ \Omega$ $L = 5\ H$ $C = 0{,}2\ F$	
Fall 3: Schwingfall	
Werte	Differentialgleichung
$R = 5\ \Omega$ $L = 2{,}5\ H$ $C = 0{,}2\ F$	

Anm.: Die Spannung $u(t)$ besitzt für die verschiedenen Fälle den nachfolgenden zeitlichen Verlauf. Dieser wurde hier mit Hilfe der mit Matlab-Standardfunktion `dsolve(eqn,cond)` gelöst. `dsolve` löst die Differentialgleichung `eqn`, die in symbolischer Form angegeben wird, unter Beachtung der Anfangswerte `cond`, siehe MATLAB-Hilfe. Die symbolische Eingabe der Differentialquotienten erfolgt dabei mit Hilfe der `diff`-Funktion. Das Ergebnis von `dsolve` ist eine symbolische Funktion, die mit der MATLAB-Standardfunktion `matlabFunction` in eine Funktion umgewandelt werden kann. Beispiel des Funktionsaufrufs:

```
syms u(t) t
eqn=diff(u,t,2)+1*diff(u,t)+2*u==0;
Du=diff(u,t);
cond=[u(0)==2, Du(0)==1];
f_symbolic=dsolve(eqn,cond)
f=matlabFunction(f_symbolic)
```



Diese Graphen dienen zum Vergleich des zeitlichen Verlaufs der Spannung $u_C(t)$, der später durch die numerischen Lösungen (Teil 2.3.) bestimmt wird. Schauen Sie sich Lösung der Differentialgleichung in den Variablen f_symbolic und f an.

In diesem Beispiel wurde die Matlab-Funktion `dsolve(eqn,cond)` eingesetzt, da diese die symbolische Lösung einer DGL höherer Ordnung berechnen kann, ohne diese zuvor zwingend in ein System von Differentialgleichungen erster Ordnung umzuwandeln. Zudem besteht die Möglichkeit, die Lösung in symbolischer Form darzustellen. Die im weiteren Verlauf zu diskutierenden numerischen Verfahren liefern dagegen nur numerische Näherungswerte an diskreten Punkten im zu untersuchenden Intervall. Die Verfahren sind zudem auf die Lösung von DGLn erster Ordnung ausgelegt, daher müssen DGLn höherer Ordnung zunächst in ein System von Differentialgleichungen erster Ordnung umgewandelt werden.

Versuch 2. Überführung Differentialgleichungen höherer Ordnung in ein System von Differentialgleichungen erster Ordnung.

Die im weiteren betrachteten numerischen Verfahren (Teil 3) beziehen sich auf die Lösung von Differentialgleichungen erster Ordnung. Sollen Gleichungen höherer Ordnung numerisch gelöst werden, so müssen diese zuvor stets auf ein Gleichungssystem 1. Ordnung zurückgeführt werden. Dies geschieht, indem man die Ableitungen durch neue Funktionen ersetzt.

Beispiel:
Die Differentialgleichung aus dem Teil 1 ist eine DGL zweiter Ordnung. Diese kann folgendermaßen in ein System von Differentialgleichungen erster Ordnung umgewandelt werden.

Die ursprüngliche Differentialgleichung hat folgende Darstellung:

$$\frac{d^2u_C}{dt^2} = -\frac{(C \cdot R^2 + L)}{L \cdot C \cdot R} \cdot \frac{du_C}{dt} - \frac{2}{L \cdot C} \cdot u_C(t) + \frac{1}{L \cdot C} \cdot U_0$$

Die zeitabhängige Größe (in unserem Fall $u_C(t)$), sowie deren Ableitung(en), die auf der rechten Seite dieser Gleichung vorkommen, ersetzen wir durch die neuen Variablen y_1 und y_2:

Sei	dann gilt (1.te Ableitung der linken und der rechten Seite)
$y_1 = u_C(t)$	$y_1{}' = \frac{du_C}{dt} = y_2$
$y_2 = \frac{du_C}{dt}$	$y_2' = \frac{d^2u_C}{dt^2} = -\frac{(C \cdot R^2 + L)}{L \cdot C \cdot R} \cdot \frac{du_C}{dt} - \frac{2}{L \cdot C} \cdot u_C(t) + \frac{1}{L \cdot C} \cdot U_0$

Somit erhalten wir das System von Differentialgleichungen erster Ordnung für die Variablen y_1 und y_2 in expliziter Form:

$$y_1{}' = y_2$$

$$y_2' = -\frac{(C \cdot R^2 + L)}{L \cdot C \cdot R} \cdot y_2 - \frac{2}{L \cdot C} \cdot y_1 + \frac{1}{L \cdot C} \cdot e$$

Für das Lösen des Differentialgleichungssystems müssen die Anfangswerte für $u_C(t)$ und $\frac{du_C}{dt}$ zu einem bestimmten Zeitpunkt (in unserem Fall $t_0 = 0$) bekannt sein oder nach einem bestimmten Kriterien auswählt werden.

Als Lösung dieses Differentialgleichungssystems erhält man einen Vektor $y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$, wobei $y_1 = u_C(t)$ die gesuchte Lösung der ursprünglichen Differentialgleichung zweiter Ordnung ist.

V 2.1 Bilden Sie aus der in V 1.1 bestimmten Differentialgleichung zweiter Ordnung ein Differentialgleichungssystem erster Ordnung nach dem hier dargestellten Prinzip:

V2.2 Bestimmen Sie die Differentialgleichungssysteme für alle 3 Fälle unter Berücksichtigung der bekannten Werte der RLC-Elemente:

Fall 1: Aperiodischer Grenzfall	
Werte	Differentialgleichungssystem erster Ordnung
$R = 2{,}5\ \Omega$ $L = 5\ H$ $C = 0{,}2\ F$	
Fall 2: Kriechfall	
Werte	Differentialgleichungssystem erster Ordnung
$R = 1{,}667\ \Omega$ $L = 5\ H$ $C = 0{,}2\ F$	
Fall 3: Schwingfall	
Werte	Differentialgleichungssystem erster Ordnung
$R = 5\ \Omega$ $L = 2{,}5\ H$ $C = 0{,}2\ F$	

Versuch 3. Numerische Verfahren zur Lösung der Differentialgleichungen

3.1 Euler-Verfahren.

V 3.1.1 Implementieren Sie die Matlab-Funktion mit folgendem Prototyp

```
function [x,y] = euler(ableitung, startvektor_y , xstart, xend, schritte)
% Input:
%       ableitung              rechte Seite der DGL bzw. des DGL-Systems als cell-array
                               (siehe auch Aufruf der Funktion f_symb)
%       startvektor_y          Vektor mit Anfangswert(en) der gesuchten Funktion
%       xstart                 Erste Stützstelle
%       xend                   Letzte Stützstelle
%       schritte               Anzahl der Schritte
%
% Output:
%       x                      Stützstellen im Intervall [xstart;xend]
%       y                      Werte der gesuchten Funktion
```

die sowohl die Differentialgleichungen als auch Differentialgleichungssysteme erster Ordnung nach dem Euler-Verfahren mit der Rekursionsformel für die Näherungswerte

$$Y_{i+1} = Y_i + h \cdot f(x_i, Y_i)$$

der Lösung $y(x)$ der Differentialgleichung der Form

$$y'(x) = f\big(x, y(x)\big)$$

berechnen kann. Siehe hierzu auch die Vorlesungsunterlagen. Die Matlab-Funktion `euler()` soll für jeden Schritt i die Werte der Stützstelle x_i und der Funktion y_i an dieser Stelle ausgeben. Hier ist ein Beispiel möglicher Ausgabe:

Testausgabe: für aperiodischer Grenzfall

```
…
======== Schritt 2 ========
x  = 0.2
y1 = 2.16
y2 = 0.27
======== Schritt 3 ========
x  = 0.3
y1 = 2.187
y2 = 0
…
```

Im Laufe des Verfahrens ist es notwendig, den Wert der Ableitung an einem Stützpunkt (x_i, Y_i) zu bestimmen. Die Ableitung entspricht der rechten Seite der jeweiligen DGL 1. Ordnung. Zur Bestimmung des Wertes der Ableitung kann die folgende, vorgegebene Matlab-Funktion (vollständige Funktion im ILIAS-Ordner) nützlich sein:

```
function [f] = f_symb(ableitung, x_w, y_w)
% Auswertung der symbolisch angegebener Funktionstupel an der Stelle (x_w, y_w)
% Beispiel des Funktionsaufrufs:
% [f] = f_symb({'@(x,y) y(2))'; '@(x,y) (y(1)*y(2)-2*x^3)'}, 0, [0 1])

% Input:
%       ableitung              rechte Seite der DGL bzw. des DGLS
%       x_w                    Stützstelle, an der die Ableitung berechnet wird
%       y_w                    Wert der Funktion an x_w
%
% Output:
%       f                      Wert(e) der Ableitung(en)
```

V 3.1.2 Testen Sie die Matlab-Funktion `euler()` mit allen Differentialgleichungssystemen aus V2.2. Für alle Tests wählen Sie als Anfangsbedingung $u(0) = 2$, $\frac{du}{dt}\Big|_{t=0} = 1$. Für die restlichen Parameter nehmen Sie die folgenden Werte

```
xstart     = 0,
xend       = 10,
schritte   = 100.
```

Das Ergebnis des Tests stellen Sie als Graphen des Verlaufs der gesuchten Größe $u(t)$ dar. Plotten Sie ihre Ergebnisse für die drei Differentialgleichungssysteme jeweils zusammen mit dem Ergebnis über `dsolve`() in einer MATLAB-Figure.

Anmerkung:
Unter Umständen könnte es hilfreich sein, dass Sie ihre Funktion `euler()` zunächst an einer DGL erster Ordnung testen, bevor Sie diese auf die Differentialgleichungssysteme anwenden. Sie können dafür z.B. die DGL für das Abklingverhalten an einem Kondensator aus der Vorlesung verwenden:

$$\frac{du_c(t)}{dt} = -\frac{1}{RC} \cdot u_c(t)$$

mit

$$u_c(t = 0) = 5V.$$

Einzureichende Daten:
Bitte reichen Sie über den ILIAS-Test

- Ihre MATLAB-Funktion `euler()`,
- ein Skript zum Aufrufen der MATLAB-Funktion für die Differentialgleichungssysteme aus V2.2
- und die drei MATLAB-Figure mit der Gegenüberstellung der Ergebnisse.

Speichern Sie die Daten in einem komprimierten (ZIP-)Ordner mit dem Namen Versuch 3p1p2

3.2 Heun-Verfahren.

V3.2.1 Implementieren Sie die Matlab-Funktion mit folgendem Prototyp

```
function [x,y] = heun(ableitung, startvektor_y , xstart, xend, schritte)
% Input:
%       ableitung               rechte Seite der DGL bzw. des DGLS
%       startvektor_y           Vektor mit Anfangswert(en) der gesuchten Funktion
%       xstart                  Erste Stützstelle
%       xend                    Letzte Stützstelle
%       schritte                Anzahl der Schritte
%
% Output:
%       x                       Stützstellen im Intervall [xstart;xend]
%       y                       Werte der gesuchten Funktion
```

die sowohl die Differentialgleichungen als auch Differentialgleichungssysteme erster Ordnung nach dem Heun-Verfahren lösen kann. Das Prinzip des Heun-Verfahrens entnehmen Sie bitte den Vorlesungsunterlagen:

$$\tilde{y}_{i+1}^{Heun} = \tilde{y}_i^{Heun} + \frac{h}{2} \cdot \left(f\left(x_i, \tilde{y}_i^{Heun}\right) + f\left(x_{i+1}, \tilde{y}_{i+1}^{Euler}\right) \right)$$

mit

$$\tilde{y}_{i+1}^{Euler} = \tilde{y}_i^{Heun} + \mathrm{h} \cdot f\left(x_\mathrm{i}, \tilde{y}_i^{Heun}\right)$$

Die `heun()`-Funktion soll für jeden Schritt i die Werte der Stützstelle x_i und der Funktion y_i ausgeben, die Form der Ausgabe kann von V3.1.1 übernommen werden.

V3.2.2 Testen Sie die Matlab-Funktion `heun()` mit allen Differentialgleichungssystemen aus V2.2. Für alle Tests wählen Sie als Anfangsbedingung $u(0) = 2$, $\frac{du}{dt}_{(t=0)} = 1$. Für die restlichen Parameter nehmen Sie die folgenden Werte

```
xstart      = 0,
xend        = 10,
schritte    = 50.
```

Das Ergebnis des Tests stellen Sie als Graphen des Verlaufs der gesuchten Größe $u(t)$ dar. Plotten Sie ihre Ergebnisse für die drei Differentialgleichungssysteme jeweils zusammen mit dem Ergebnis über `dsolve()` in einer MATLAB-Figure.

Einzureichende Daten:

Bitte reichen Sie über den ILIAS-Test

- Ihre MATLAB-Funktion `heun()`,
- ein Skript zum Aufrufen der MATLAB-Funktion für die Differentialgleichungssysteme aus V2.2
- und die drei MATLAB-Figures mit der Gegenüberstellung der Ergebnisse.

Speichern Sie die Daten in einem komprimierten (ZIP-)Ordner mit dem Namen Versuch 3p2p2

3.3 Runge-Kutta-Verfahren

Die hier betrachteten Verfahren zur numerischen Lösung einer Differentialgleichung können verallgemeinert werden, indem das Runge-Kutta-Verfahren eingeführt wird. Denn das Euler-Verfahren bildet das Runge-Kutta-Verfahren erster Ordnung ab, das Heun-Verfahren stellt das Runge-Kutta-Verfahren zweiter Ordnung dar. Hier soll das klassische und oft genutzte Runge-Kutta-Verfahren vierter Ordnung implementiert werden. Laut diesem wird der nächste Näherungswert Y_{i+1} wie folgt bestimmt:

Man nimmt das Intervall von $(x_i;\ y_i)$ bis $(x_{i+1};\ y_{i+1})$ und berechnet an vier Stellen des Richtungsfeldes die Steigungen k_1, k_2, k_3 und k_4:

$$k_1 = h \cdot f(x_i, Y_i)$$

$$k_2 = h \cdot f\left(x_i + \frac{1}{2} \cdot h, Y_i + \frac{k_1}{2}\right)$$

$$k_3 = h \cdot f\left(x_i + \frac{1}{2} \cdot h, Y_i + \frac{k_2}{2}\right)$$

$$k_4 = h \cdot f(x_i + h, Y_i + k_3)$$

Somit lässt sich die nächste Annäherung zum y-Wert wie folgt berechnen:

$$Y_{i+1} = Y_i + \frac{1}{6} \cdot (k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4)$$

V3.3.1 Implementieren Sie die Matlab-Funktion mit folgendem Prototyp

```
function [x,y] = runge_kutta(ableitung, startvektor_y , xstart, xend, schritte)
% Input:
%       ableitung               rechte Seite der DGL bzw. des DGLS
%       startvektor_y           Vektor mit Anfangswert(en) der gesuchten Funktion
%       xstart                  Erste Stützstelle
%       xend                    Letzte Stützstelle
%       schritte                Anzahl der Schritte
%
% Output:
%       x                       Stützstellen im Intervall [xstart;xend]
%       y                       Werte der gesuchten Funktion
```

V3.3.2 Testen Sie die Matlab-Funktion `runge_kutta()` mit allen Differentialgleichungssystemen aus V2.2. Für alle Tests wählen Sie als Anfangsbedingung $u(0) = 2$, $\frac{du}{dt}_{(t=0)} = 1$. Für die restlichen Parameter nehmen Sie die folgenden Werte

```
xstart     = 0,
xend       = 10,
schritte   = 40.
```

Das Ergebnis des Tests stellen Sie als Graphen des Verlaufs der gesuchten Größe $u(t)$ dar. Plotten Sie ihre Ergebnisse für die drei Differentialgleichungssysteme jeweils zusammen mit dem Ergebnis über `dsolve()` in einer MATLAB-Figure.

Einzureichende Daten:

Bitte reichen Sie über den ILIAS-Test

- Ihre MATLAB-Funktion `runge_kutta()`,

- ein Skript zum Aufrufen der MATLAB-Funktion für die Differentialgleichungssysteme aus V2.2
- und die drei MATLAB-Figures mit der Gegenüberstellung der Ergebnisse.

Speichern Sie die Daten in einem komprimierten (ZIP-)Ordner mit dem Namen Versuch 3p3p2

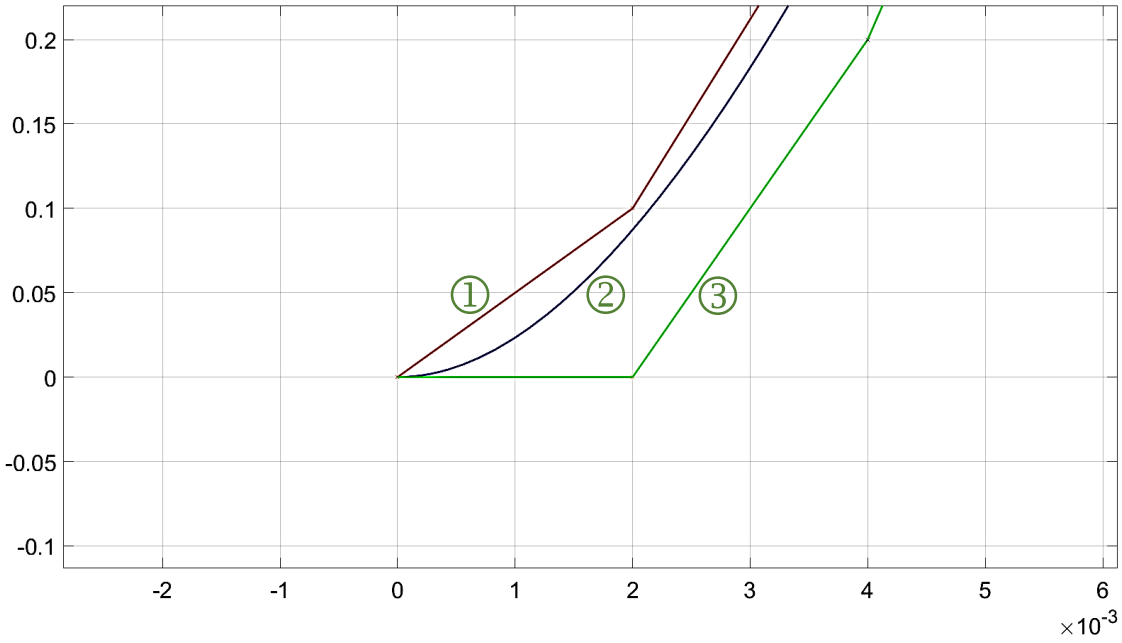
Versuch 4. Vergleich der Effektivität der einzelnen numerischen Lösungsverfahren.

V4.1 Testen Sie alle drei Verfahren für die Eingabeparameter

```
xstart    = 0,
xend      = 30,
schritte  = 40.
```

Was beobachten Sie bei den Kurvenverläufen der einzelnen Verfahren? Was ist ihre Schlussfolgerung bzgl. der Genauigkeit der Verfahren?

V4.2 Es sei eine Differentialgleichung nach Euler- und Heun-Verfahren gelöst und die Ergebnisse graphisch dargestellt. Hier ist ein Ausschnitt aus diesem Graph gegeben:



Die Kurve ② beschreibt die exakte Lösung. Welche der beiden anderen Kurven stellt die Lösung nach Euler-, welche nach Heun-Verfahren dar? Begründen Sie Ihre Antwort:

Welche Schrittweite wurde hier ausgewählt?

Welche Auswirkungen in Sicht der Genauigkeit der Lösung und des Aufwands der Berechnung kann die Vergrößerung und die Verkleinerung der Schrittweite haben?