Numerische Simulation von Differentialgleichungen

Wie kann man die Wärmeleitungsgleichung numerisch simulieren?

Interdisziplinäre Arbeit
Gemäss den kantonalen Vorgaben des Kantons Graubünden
und den GKD-spezifischen Ausführungen

vorgelegt von

Jan Davids

Sagogn / Schweiz

am Gymnasium Kloster Disentis

genehmigt auf Antrag von Frau

Sarah Huber

Disentis / 12.04.2022

1.Überarbeitung (nach der Abgabe) am 19.10.2022

Abstract

Viele Abläufe in der Natur können durch Differentialgleichungen beschrieben werden, so ist es auch nicht verwunderlich, dass die Differentialgleichung eine Notwendigkeit in allen Naturwissenschaften darstellt. Es stellt sich jedoch die Frage nach der Methode, bzw. wie die Lösungen von Differentialgleichungen numerisch angenähert werden können. Deshalb werden in dieser Arbeit zwei Verfahren betrachtet die Differentialgleichungen numerisch lösen können. Diese beiden Verfahren werden verglichen und anschliessend miteinander und mit der exakten Lösung einer Funktion verglichen. Zudem werden die mathematischen Hintergründe zu den Verfahren angeschaut und das erste Verfahren hergeleitet.

Inhaltsverzeichnis

1 Vorwort

Da ich mich schon seit meiner Primarschulzeit brennend für Mathematik interessiere, fiel mir die Wahl des Faches für die interdisziplinäre Arbeit nicht schwer. Auf Vorschlag meiner Lehrperson in Mathematik fiel mir dann die Wahl nach dem Thema der Arbeit abermals nicht schwer. Den Vorschlag habe ich später nicht bereut, da es sich wirklich um ein sehr spannendes und herausforderndes Thema handel.
Ich bedanke mich zudem ganz herzlichst bei unserer Mathematiklehrerin Sarah Huber, die mir bei Fragen immer hilfsbereit weitergeholfen hat und keine Mühen gescheut hat mir das Wissen zu diesem Thema beizubringen.

2 Einleitung

In dieser interdisziplinären Arbeit (kurz IDA), wollen wir uns mit der Lösung von Differentialgleichungen beschäftigen. Da vieles in der Natur durch Differentialgleichungen beschrieben werden kann, sind Differentialgleichungen in den Naturwissenschaften von sehr grosser Bedeutung. Als Beispiel wollen wir die Wärmeleitungsgleichung aus der Physik durch numerische Verfahren lösen. Wir werden uns weniger mit dem physikalischen Hintergrund befassen als mit den mathematischen Methoden zur Lösung. Für die numerischen Verfahren nehmen wir zwei eher "simple"Verfahren: Das explizite Euler-Verfahren und das klassische Runge-Kutta-Verfahren. Zudem werden wir diese zwei Verfahren auf Differentialgleichungen mit exakter Lösung anwenden und später die Fehlerentwicklung analysieren. Bevor jedoch die Verfahren angeschaut werden, müssen im Vorfeld noch einige wichtige Grundbegriffe geklärt werden, die notwendig für das anschliessende Verständnis sind.

3 Grundbegriffe

3.1 Differentialquotient/Ableitung

Der Differentialquotient oder auch Ableitung genannt gibt die Punktsteigung einer Funktion wieder. Bei einer linearen Funktion ($f(x) = ax + b$) entspricht die Ableitung gerade der Variablen a und wird bei der linearen Funktion wie folgt berechnet:

$$\text{Gegeben: } f(x) = ax + b \Rightarrow f'(x) = \frac{f(a) - f(b)}{a - b}$$

Will man jedoch die Ableitung von, beispielsweise, einer quadratischen Funktion bestimmen, so muss man eine allgemeine Form für die Ableitung finden. Für diese allgemeine Form wird folgender Grenzwert (Limes) benötigt, der Differentialquotient heisst:

$$f'(x) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}$$

3.2 Partielle Ableitung

Die partielle Ableitung ist die Ableitung einer Funktion mit mehreren veränderlichen Variablen nach einer von diesen Variablen, wobei die restlichen Variabeln wie Konstanten behandelt werden[8]. Um dies besser zu veranschaulichen lösen wir zuerst ein Beispiel. Gegeben sei eine Funktion im dreidimensionalen Raum: $z = 5x^3 + 3y^5$. Leitet man nun diese Funktion nach der x-Variablen ab, so ergibt sich eine Ableitung die mit den folgenden zwei Notationen aufgestellt werden kann:

$$\partial_x z = 15x^2 \qquad \frac{\partial z}{\partial x} = 15x^2$$

Bei der partiellen Ableitung nach der x-Variablen, werden alle anderen Variablen wie konstanten behandelt und auch nach diesen Regeln abgeleitet. Wir können jedoch auch das oben aufgeführte Beispiel nach der y-Variablen ableiten:

$$\partial_y z = 15y^4 \qquad \frac{\partial z}{\partial y} = 15y^4$$

3.3 Differentialgleichung

Durch die Definitionen vom Differentialquotienten können wir nun den Begriff der Differentialgleichung definieren. Wobei wir zwischen gewöhnlichen und partiellen Differentialgleichungen unterscheiden. In Worten, ist eine gewöhnliche Differentialgleichung eine Gleichung in der mindestens eine Ableitung einer unbekannten Funktion vorkommt[6]. Dabei ist es wichtig, dass es sich um eine Funktion mit nur einer veränderlichen x handelt, wobei die Funktion mit $f(x)$, $g(x)$, $u(x)$, ... angegeben werden kann.

Beispiele für gewöhnliche Differentialgleichungen:

$f'(x) = x^2, \quad f'(x) = f(x), \quad y'(x) + 2 \cdot y(x) = x^2 + e^x,$

$f''(x) + f'(x) = sin(x), \quad \dfrac{f''(x) + f(x)}{x} = 5$

Wir betrachten nun partielle Differentialgleichungen. Diese unterscheiden sich von den gewöhnlichen Differentialgleichungen dadurch, das sie Ableitungen nach mehreren Variablen besitzten[3]. Die Form für eine Differentialgleichungen, die von x und y abhängt lautet:

$$F\left(x, y, u(x,y), \frac{\partial u(x,y)}{\partial x}, \frac{\partial u(x,y)}{\partial y}, ..., \frac{\partial^2 u(x,y)}{\partial x \partial y}, ...\right) = 0$$

Wobei F eine beliebige Funktion ist.

Beispiel für eine partielle Differentialgleichung:

$\dfrac{\partial u(x,t)}{\partial t} + c \cdot \dfrac{\partial u(x,t)}{\partial x} = 0$ (lineare Transportgleichung in einer Raumdimension)

4 Wärmeleitungsgleichung

Wie in der Einleitung bereits erklärt, ist unser Ziel, die Wärmeleitungsgleichung numerisch zu simulieren, wobei wir jedoch bei dem physikalischen Teil nicht zu sehr ins Detail gehen werden, da es uns vor allem um die numerische Lösung geht. Wir betrachten nun die Bedingungen der Wärmeleitungsgleichung[10]:

$\partial_t u - \partial^2_{xx} u = f$

$u(t,0) = sin\left(10.0t - \dfrac{\pi}{2} + 1\right)$

$u(t,1) = 0$

$u(0,x) = u_0(x)$ im Intervall $(0,1)$

Um diese Bedingungen besser zu verstehen, versuchen wir sie in Worte zu übersetzen.

1. Bedingung: Die zweite Ableitung in x-Richtung subtrahiert von der ersten Ableitung in Richtung der Zeit ergibt eine stetige Funktion f.

2. Bedingung: An der Stelle $x = 0$ entspricht der y-Wert der Gleichung $sin\left(10.0t - \dfrac{\pi}{2} + 1\right)$, wobei t die Variable der Zeit ist. Diese Gleichung bewirkt ein sich wiederholendes Auf- und Absteigen des y-Wertes zwischen 0 und 1.

3. Bedingung: An der Stelle $x = 1$ entspricht der y-Wert unabhängig von der Zeit t immer 0.

Die zweite und dritte Bedingung werden auch Randbedingungen genannt.

4. Bedingung: Zum Zeitpunkt $t = 0$ besitzten alle x-Werte im Intervall von 0 bis 1 den y-Wert $u_0(x)$, ausser bei 0 und 1 selbst.

5 Diskretisierung des Intervalls $[0, 1]$

Da wir die Wärmeleitungsgleichung nicht exakt berechnen können, müssen wir eine Annäherung finden, deshalb unterteilen wir das Intervall von 0 bis 1 in n verschiedene Gitterpunkte $(x_1, ..., x_n)$. Wobei der Abstand zwischen diesen Stützstellen $h = \frac{1}{n}$ ist. An diesen Stellen versuchen wir die y-Werte möglichst genau zu bestimmen und anschliessend legen wir einen Graphen durch diese Punkte. Dadurch erhalten wir eine Funktion, welche sich mit der Zeit verändert und die Wärmeleitungsgleichung beliebig genau annähern kann, je nach der Anzahl Gitterpunkte. Je mehr Gitterpunkte, desto genauer die Näherung.

6 Finite-Differenzen-Methode

Die Idee der finiten-Differenzen-Methode ist es, die Ortsableitung an endlich vielen Stellen (finiten Stellen) mit dem gleichen Abstand zueinander zu approximieren. Diese Punkte haben wir bereits im vorherigen Abschnitt "Diskretisierung des Intervall $[0, 1]$"definiert.
Deshalb ist nun zu zeigen, dass die Ableitungen in der Wärmeleitungsgleichung den folgenden Differenzenquotienten entsprechen:

$$\partial_x u_i \approx \frac{u_{i+1} - u_i}{h}$$

$$\partial_x u_i \approx \frac{u_i - u_{i-1}}{h}$$

$$\partial^2_{xx} u_i \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}$$

Um uns diese Bedingungen und die anschliessende Herleitung etwas besser vorzustellen, betrachten wir die unten aufgeführte Funktion mit mehreren y-Werten, welche in x-Richtung alle die gleiche Distanz besitzen.

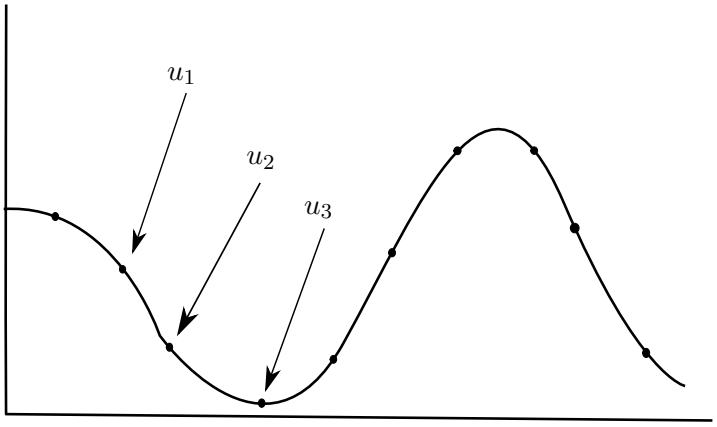

Abbildung 1: beliebige stetige Funktion und Punkte mit äquidistanten x-Werten

Um eine Ableitung anzunähern können wir uns den Differenzenquotienen zu nutze machen, grob gesagt ist der Differenzenquotient der Differentialquotient ohne den Grenzwert:

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

Der Differenzenquotient gibt die durchschnittliche Steigung zwischen den Punkten x und $x + h$ an. Je kleiner der Abstand h gewählt wird, desto genauer kann die Punktsteigung an den stellen x bzw. $x + h$ approximiert werden.

Wir unterscheiden zwischen drei Ableitungsformen: Der Vorwärtsableitung, der Rückwärtsableitung und der zentralen Ableitung. Wobei wir nur die ersten zwei Ableitungsformen benötigen und die letzte lediglich zur Vollständigkeit erwähnt wurde.
Wollen wir nun die Vorwärtsableitung für den Punkt u_3 berechnen, so gehen wir "vorwärts"auf der x-Achse und betrachten den nächsten Punkt. Das wäre also der Punkt u_4. Wir setzen diese nun in den Differenzenquotienten ein:

$$u_3' \approx \frac{u_4 - u_3}{h}$$

Wir können nun aber auch den Differenzenquotient für allgemeine u_i notieren:

$$u_i' \approx \frac{u_{i+1} - u_i}{h}$$

Wenn wir nun, wie mit der Vorwärtsableitung, die Rückwärtsableitung für den Punkt u_3 berechnen wollen, so müssen wir auf der x-Achse "rückwärts "gehen und den Punkt u_2 betrachten. Setzen wir diese Punkte nun wieder ein, so erhalten wir die folgende Gleichung:

$$u_3' = \frac{u_3 - u_2}{h}$$

Auch diesen Ausdruck können wir wieder für allgemeine u_i definieren:

$$u_i' \approx \frac{u_i - u_{i-1}}{h}$$

Weil wir den Ausdruck u_i' auch als $\partial_x u_i$ notieren können, haben wir die ersten zwei Bedingungen hiermit schon hergeleitet.

Um die letzte Bedingung $\partial_{xx}^2 u_i \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}$ nun noch herzuleiten benötigen wir zum Verständnis den Differenzialquotienten.

Leiten wir eine Funktion ein Mal ab, so erhalten wir den folgenden Differenzialquotienten:

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Nun ist aber ∂_{xx}^2 die zweite Ableitung in x-Richtung. Deshalb leiten wir die Funktion zwei Mal ab und erhalten folgenden Ausdruck:

$$f''(x) = \lim_{h \to 0} \frac{f'(x+h) - f'(x)}{h}$$

Wenn wir wiederum die Approximation dieses Ausdrucks auf den Punkt u_3 anwenden und die zuvor hergeleiteten Vorwärts- und Rückwärtsableitung zu Hilfe nehmen, so erhalten wir:

$$u_3'' \approx \frac{\frac{u_4 - u_3}{h} - \frac{u_3 - u_2}{h}}{h} = \frac{u_4 - 2u_3 - u_2}{h^2}$$

Auch hier können wir die Approximation wieder für allgemeine u_i definieren:

$$u_i'' \approx \frac{\frac{u_{i+1} - u_i}{h} - \frac{u_i - u_{i-1}}{h}}{h} = \frac{u_{i+1} - 2u_i - u_{i-1}}{h^2}$$

Zum Schluss gilt auch in diesem Fall: $u_i'' = \partial_{xx}^2 u_i$

Somit haben wir gezeigt, das die oben aufgeführten Bedingungen gelten[1]. ■

7 Matrixform

Wir haben zuvor gezeigt, das die folgeneden drei Bedingungen gelten:

$$\partial_x u_i \approx \frac{u_{i+1} - u_i}{h}$$

$$\partial_x u_i \approx \frac{u_i - u_{i-1}}{h}$$

$$\partial^2_{xx} \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}$$

Letztere können wir in eine Art Tabelle eintragen, die wie folgt aussieht:

$$i = 1 \ : \quad \frac{1}{h^2}(u_2 - 2u_1 + u_0) = \frac{1}{h^2}(u_0 - 2u_1 + u_2)$$

$$i = 2 \ : \quad \frac{1}{h^2}(u_3 - 2u_2 + u_1) = \frac{1}{h^2}(u_1 - 2u_2 + u_3)$$

$$i = 3 \ : \quad \frac{1}{h^2}(u_4 - 2u_3 + u_2) = \frac{1}{h^2}(u_2 - 2u_3 + u_4)$$

.

.

.

$$i = n \ : \quad \frac{1}{h^2}(u_{n+1} - 2u_n + u_{n-1}) = \frac{1}{h^2}(u_{n-1} - 2u_n + u_{n+1})$$

Diese Ausdrücke für die verschiedenen i können wir nun in Matrixform bringen.
Wegen der Randwertbedingung $u(t,0) = sin\left(10.0t - \frac{\pi}{2} + 1\right)$ muss die erste Zeile der Matrix mit Nullen gefüllt werden.

Dadurch erhalten wir eine Matrix die so aussieht[7]:

$$\Rightarrow \quad \frac{1}{h^2} \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 1 & -2 & 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & 0 & 1 & -2 \end{pmatrix} \cdot \begin{pmatrix} u_0 \\ \cdot \\ \cdot \\ \cdot \\ u_n \end{pmatrix} = Au$$

Zur Erinnerung, diesen Ausdruck können wir für die Stelle $\partial^2_{xx} u_i$ einsetzen.

8 Das Explizite Euler-Verfahren

Das Eulerverfahren oder auch Einschrittverfahren ist das einfachste Verfahren um Anfangswertprobleme zu lösen. Das Vorgehen geht folgendermassen: Ein Anfangspunkt ist gegeben und man versucht den nächsten Stützpunkt mithilfe einer Approximation der Steigung anzunähern, von diesem approximierten Punkt wiederholt man den Prozess und erhält eine Annäherung an die exakte Lösung. Für das Euler-Verfahren sollte man möglichst kleine h (Abstand zwischen den beiden Punkten) wählen[5].

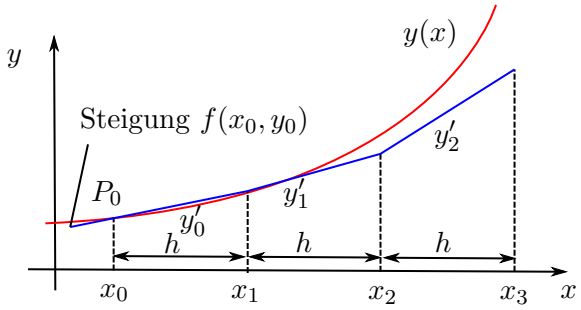

Abbildung 2: geometrische Interpretation von Euler-Verfahren[11]

8.1 Herleitung

Das explizite Eulerverfahren besitzt die Form $y_{n+1} = y_n + hf(t_n, y(n))$. In diese Gleichung wird Punkt für Punkt eingesetzt, und so später eine Approximation erstellt. Bevor wir aber fortfahren, leiten wir erst mal das explizite Euler-Verfahren her.

Es sei ein Anfangswertproblem gegeben[4]:

$$y'(t) = f(t, y(t)) \quad y(t_0) = y_0$$

Es gilt zudem: f ist stetig und $y_0 \in \mathbb{R}$

Wir betrachten nun den Differenzialquotienten:

$$y'(t) = \lim_{h \to 0} \frac{y(t+h) - y(t)}{h}$$

Da wir wiederum die exakte Ableitung nicht numerisch auswerten können, approximieren wir sie mit dem Differenzenquotienten:

$$y'(t) \approx \frac{y(t+h) - y(t)}{h}$$

Wir approximieren also den Differenzialquotienten $y'(t) = f(t, y(t))$ durch die Differenzengleichung

$$\frac{y(t+h) - y(t)}{h} = f(t, y(t))$$

Wenn wir nun die Gleichung nach $y(t + h)$ umformen, so erhalten wir die folgende Gleichung:

$$y(t + h) = y(t) + hf(t, y(t))$$

Dies ist nun genau ein Schritt der expliziten Euler-Methode. Durch die Einfürhung der Notation $t_{n+1} = t_n + h$ und $y_n = y(t_n)$ ergibt sich:

$$y_{n+1} = y_n + hf(t_n, y_n)$$

Somit haben wir das explizite Eulerverfahren hergeleitet. Nun gilt es dieses Verfahren noch auf unser Problem anzuwenden. Dafür formen wir die Gleichung der Wärmeleitungsgleichung nach $\partial_t u$ um:

$\partial_t u - \partial_{xx}^2 u = f \quad | + \partial_{xx}^2 u$

$\partial_t u = f + \partial_{xx}^2 u$

Wir können nun $\partial_{xx}^2 u$ durch Au ersetzen:

$\partial_t u = f + Au$

Ändern wir nun das Anfangswertproblem von

$y'(t) = f(t, y(t))$ zu $\partial_t u = f + Au$

So ergibt sich eine Gleichung die wie folgt aussieht:

$u(t_{n+1}) = u(t_n) + h \cdot (f + Au)$

Wobei $u(t_{n+1})$, $u(t_n)$ Vektoren sind und h der Zeitschritt ist. Wichtig ist jedoch, dass der Faktor vor der Matrix $\frac{1}{h^2}$ nicht die gleichen h wie die des Zeitschritts beinhaltet, sondern die Abstände zwischen den Gitterpunkten auf der x-Achse sind. Das heisst wir haben zwei verschiedene h.

8.2 Implementierung vom expliziten Euler-Verfahren in Octave

Für den Programmcode verwenden wir Octave bzw. Matlab. Diese Programmierumgebungen eignen sich besonders gut, um numerische Probleme in der Mathematik zu lösen. Für unser numerisches Problem schreiben wir folgenden Programmcode:

```
n=10;

  f=zeros(n);
  A=zeros(n,n);
    for i=1:n
      for j=1:n
       if i == j
       A(i,j)=-2;
       endif
       if j-1==i
       A(i, j)=1;
       endif
       if j+1==i
       A(i, j)=1;
       endif
       if i==1;
      A(1, j)=0;
       endif
    endfor
  endfor

%Hier wird die Matrix erstellt, sie entspricht der
Variblen A

x=linspace(0, 1, n)'
% x ist ein Vektor mit n Komponenten, die alle im gleichen
   Abstand zueinander sind
h1=1/(n+1);
%hier wird die Schrittweite zwischen den Gitterpunkten
   deifiniert, die als h1 bezeichnet wird
h2=1/n^3;
%nun muss noch der zeitliche Abstand h2 definiert werden


y=[0, 0]';
y(n)=0;
```

```
%es wird ein Vektor y definiert mit n Komponenten, die alle
   0 sind
b=0-h2;

for i=0:h2:1
  b=b+h2;
  %nun wird so lange h2 zu 0 addiert bis die Addition 1
     erreicht hat
  y(1)=sin(10*b-pi/2)+1;
  %Randwertbedingungen werden definiert

  y=y+h2*((1/(h1*h1))*A*y)
  %dies ist die zuvor hergeleitete Gleichung, wobei wir f=0
     setzten
  plot(x, y, 'r');
  axis([0 1 0 2])
  grid on
  pause(0.001)
  %zum Schluss werden die Werte in einem Koordinatensystem
     als Graph bildlich dargestellt

endfor
```

Bemerkung: Der Zeitschritt $h2$ muss kleiner als $h1$ sein, sonst konvergiert die Iteration nicht.
Wenn wir nun dieses Programm starten, so erhalten wir einen Graphen, der sich auf und ab bewegt und die Wärmeleitung darstellt, wenn links eine Wärmequelle ist:

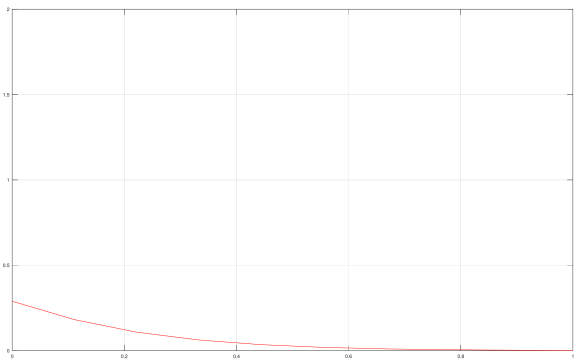

Abbildung 3: Foto aus der Ausgabe des Programms von oben

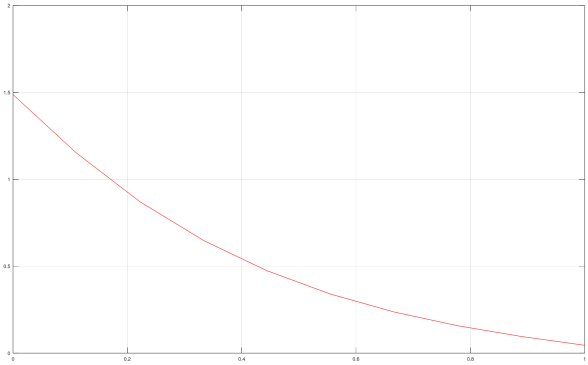

Abbildung 4: Graph steigt an

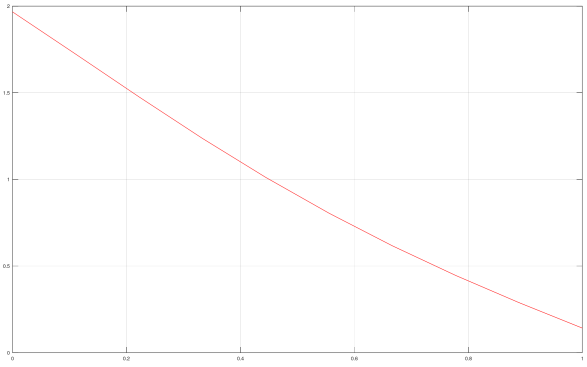

Abbildung 5: sobald der Punkt (0, 1) erreicht wird sinkt der Graph wieder

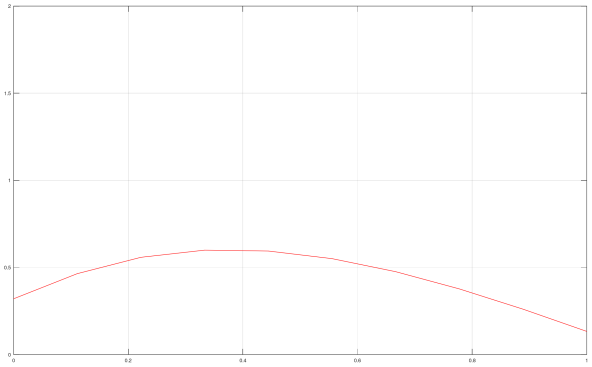

Abbildung 6: Graph sinkt wieder auf null und steigt danach wieder zudem breitet sich Wärme nach rechts aus

Hiermit haben wir die Lösung der Wärmeleitungsgleichung angenähert. Wir müssen nun noch verstehen, was die Masseinheiten auf den Achsen sind, bzw. den Graphen interpretieren. Da es sich um die Wärmeleitungsgleichung handelt, handelt es sich bei der x-Achse um die Entfernung von der Wärmequelle und bei der y-Achse um die Temperatur. Der Graph sagt nun aus, dass sich beispielsweise ein Metallstab an der Stelle, an der sich die Wärmequelle befindet stärker erhitzt als an einer entfernteren Stelle. Da die Wärmequelle in regelmässigen Abständen aktiviert und wieder deaktiviert wird, entsteht eine Art "Wellenform".
Wir können nun noch vergleichen, wie gross die Abweichung dieses Verfahrens ist, wenn wir ein Anfangswertproblem erstellen, das eine exakte Lösung hat. Wir nehmen hier als Beispiel das Anfangswertproblem:

$$u_0 = 1, \quad u_n = e^{t \cdot n}, \quad (x - t^2) \cdot e^{tx} = f, \quad \partial_t u - \partial_{xx}^2 u = f$$

Dieses Anfangswertproblem hat die exakte Lösung $u = e^{tx}$. Wobei $\partial_t u = x \cdot e^{tx}$ und $\partial_{xx}^2 u = \partial_x (t \cdot e^{tx}) = t^2 \cdot e^{tx}$. Wenn wir dieses Anfangswertproblem nun wieder ins Euler-Verfahren einsetzen, so erhalten wir diese Gleichung:

$$u(t_n + 1) = u(t_n) + h \cdot ((x - t^2) \cdot e^{tx} + Au)$$

Wir implementieren nun wiederum diese Gleichung in Octave:

```
n=0;
c=[0];
d=[0];
k=30;
t=0;

for a=0:1:k;
%Hier wird von null bis 30 raufgezaehlt, wobei sich der Wert
%von a pro Durchgang um 1 erhoeht
t=t+1;
n=n+10;
%Hier wird die Anzahl Gitterpunkte definiert
%(Gitterpunkte=n)
a
n
%Ausgabe dieser beiden Werten
f=zeros(n);
A=zeros(n,n);
for i=1:n
  for j=1:n
    if i == j
    A(i,j)=-2;
    endif
    if j-1==i
    A(i, j)=1;
    endif
    if j+1==i
    A(i, j)=1;
    endif
    if i==1;
    A(1, j)=0;
    endif
    if i==n;
    A(n, j)=0;
    endif
  endfor
endfor
%Wie beim vorherigen Code wird hier wieder die benoetigte
   Matrix erstellt


x=linspace(0, 1, n)';
```

```
h1=1/(n+1);

h2=1/n^3;
%analog des vorherigen Codes

y=ones(n,1);
%Da zu Beginn alle y-Werte gleich 1 sind muss ein Vektor mit
    lauter Einsen erstellt werden
b=0-h2;

for i=0:h2:1
  b=b+h2;
  y(1)=1;
  y(n)=exp(x(n)*b);
  %Randbedingungen

  y=y+h2*((x-b^2).*exp(x.*b)+(1/(h1*h1))*A*y);
  %Implememntierung des Euler-Verfahrens


  f=exp(x.*b);
  %exakte Loesung der Funktion wird berechnet


endfor

%hier wird der Fehler fuer verschiedene n berechnet und mit
   einem Graphen ausgegeben
c(a+1)=max(g=abs(f-y))
%maximaler Fehler pro n
d(a+1)=n
plot(d, c, 'r');
axis([0 k*10 0 0.07])
grid on
pause(0.001)
%Code um Graphen auszugeben
endfor
```

Wenn wir nun dieses Programm über eine längere Zeit laufen lassen, so erhalten wir den folgenden Graphen:

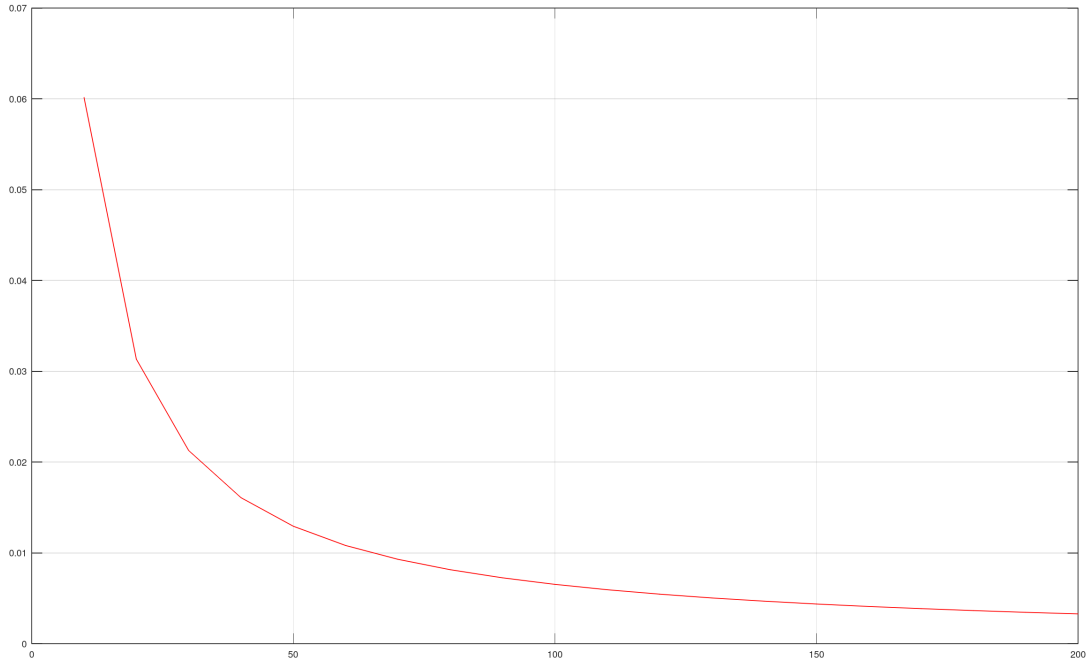

Abbildung 7: Abnahme des Fehlers für grössere n

Wir sehen nun eine Kurve, die für immer grössere x-Werte kleinere y-Werte annimmt. Die x-Achse stellt in unserem Fall die Anzahl Gitterpunkte n dar und die y-Achse den Fehler. Wir sehen nun, das der Fehler stark abnimmt, je mehr Gitterpunkte wir verwenden.

9 Runge-Kutta-Verfahren

Das Runge-Kutta-Verfahren ist nicht ein einzelnes Verfahren, sondern eher eine Familie von Verfahren. So gibt es beispielsweise das Runge-Kutta-Verfahren der Ordnung 2, das Heun-Verfahren, Runge-Kutta-Verfahren der Ordnung 4 und noch viele weitere.
Das bekannteste Verfahren ist das letztere, das Runge-Kutta-Verfahren der Ordnung 4 oder auch klassisches Runge-Kutta-Verfahren genannt. Wir wollen nun zum Abschluss die Wärmeleitungsgleichung mit diesem Verfahren lösen und anschliessend, wie beim Eulerverfahren, mit einer exakten Lösung vergleichen.

Bevor wir jedoch das Verfahren in Octave implementieren, müssen wir zuerst wissen, wie das klassische Runge-Kutta-Verfahren überhaupt aussieht[9]:

Für eine Differentialgleichung der Form $y'(t) = f(t, y(t)),\ y(t_0) = y_0$ ist die Näherungslösung:

$$y(t_1) \approx y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

mit

$$k_1 = h \cdot f(t_i, y_i)$$

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

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

$$k_4 = h \cdot f\left(t_i + h, y_i + k_3\right)$$

Um das Runge-Kutta-Verfahren nachzuvollziehen ist eine geometrische Interpretation angebracht[2]:

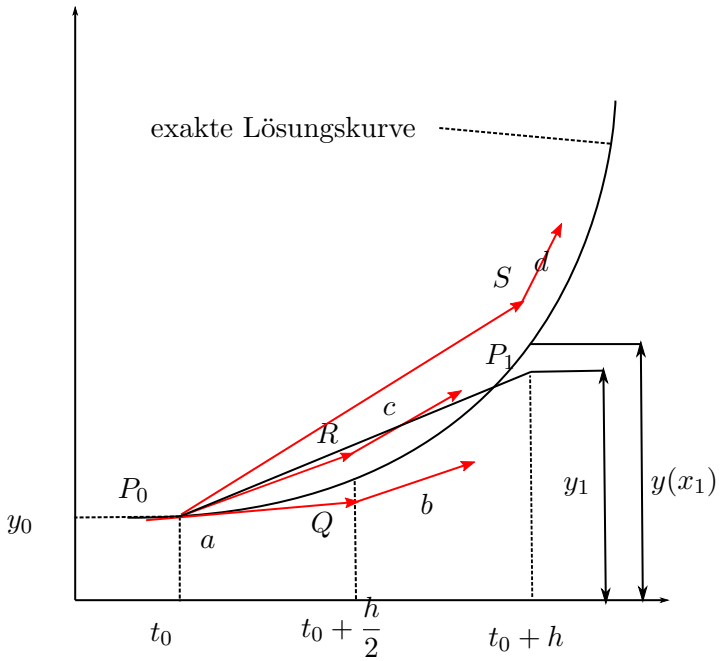

Abbildung 8: geometrische Interpretation von Runge-Kutta-Verfahren

1. Steigungswert: $m_1 = f(x_0, y_0)$

Dieser Wert ist die Steigung der Tangenten an die exakte Lösungskurve im Punkt P_0 (Linie a), weil $y'(t) = f(t, y(t))$ ist.

2. Steigungswert: $m_2 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{k1}{2}\right)$

Wir gehen von P_0 in Richtung der Tangente bis zur Mitte des Intervalls und erreichen dort den Punkt Q mit den Koordinaten:

$$x_Q = x_0 + \frac{h}{2}, \quad y_Q = y_0 + \frac{1}{2}h \cdot f(x_0, y_0) = y_0 + \frac{k_1}{2}$$

Die Lösungskurve durch Q besitzt an dieser Stelle die Steigung(b):

$$m_2 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{k_1}{2}\right)$$

3. Steigungswert $m_3 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{k2}{2}\right)$

Wir gehen nun von P_0 aus geradlinig mit der Steigung m_2 bis zur Intervallmitte und gelangen so zum Punkt R mit den Koordinaten:

$$x_R = x_0 + \frac{h}{2}, \quad y_R = y_0 + \frac{1}{2}h \cdot f\left(x_0 + \frac{h}{2}, y_0 + \frac{k_1}{2}\right) = y_0 + \frac{k2}{2}$$

Die Lösungskurve durch R bestitzt an dieser Stelle die Steigung(c):

$$m_3 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{k2}{2}\right)$$

Steigungswert $m_4 = f(x_0 + h, y_0 + k_3)$

Wir gehen nun von P_0 geradlinig mit der Steigung m_3 bis zum Intervallende. Man erreicht somit den Punkt S mit den Koordinaten

$$x_S = x_0 + h, \quad y_S = y_0 + h \cdot f\left(x_0 + \frac{h}{2}, y_0 + \frac{k_2}{2}\right) = y_0 + k_3$$

Die Lösungkurve durch S besitzt dann an dieser Stelle die Steigung(d):

$m_4 = f(x_0 + h, y_0 + k_3)$.

Die Steigung der Geraden, die vom Anfangspunkt P_0 zum Endpunkt P_1 führt, wird dann aus den vier Steigungswerten wie folgt berechnet:

$$m = \frac{m_1 + 2m_2 + 2m_3 + m_4}{6}$$

Dieser Wert ist eine mittlere Steigung der Lösungkurve im betrachteten Intervall:

$$x_0 \leq x \leq x_1 = [x_0, x_0 + h]$$

Nun implementieren wir das Runge-Kutta-Verfahren in Octave:

```
n=10;

f=zeros(n);
A=zeros(n,n);
for i=1:n
for j=1:n
if i == j
A(i,j)=-2;
endif
if j-1==i
A(i, j)=1;
endif
if j+1==i
A(i, j)=1;
endif
if i==1;
A(1, j)=0;
endif
endfor
endfor



x=linspace(0, 1, n)';

h1=1/(n+1);

h2=1/1000;

y=[0, 0]';
```

```
y(n)=0;
b=0-h2;
%bis hier hin analog zum Code vom Eulerverfahren

for i=0:h2:1
b=b+h2;
y(1)=sin(10*b-pi/2)+1;

k=h2*((1/(h1*h1))*A*y);
l=h2*((1/(h1*h1))*A*(y+k/2));
m=h2*((1/(h1*h1))*A*(y+l/2));
n=h2*((1/(h1*h1))*A*(y+m));
%Implementierung des Runge-Kutta-Verfahrens
o=1/6*(k+2*l+2*m+n);
%mittlere Steigung
y=y+o;
%wird nun zu y0 addiert


plot(x, y, 'r');
axis([0 1 0 2])
grid on
pause(0.001)

endfor
```

Wir wollen nun das Runge-Kutta-Verfahren mit der exakten Lösung und gleichzeitig mit dem Eulerverfahren vergleichen, um ein Bild zu haben, wie sich der Fehler verkleinert, bzw. verändert hat. Dazu bedienen wir uns wieder am gleichen Beispiel wie beim Vergleich zwischen der exakten Lösung und dem Eulerverfahren aus Kapitel 8.2.

Dazu erstellen wir ein Programm, das alle drei Berechnungen durchrechnet und anschliessend diese mitteinander vergleicht:

```
n=0;
c=[0];
d=[0];
e=[0];
i=[0];
k=30;
t=0;
%hier werden Vektoren und Variablen definiert,
%die spaeter verwendet werden
for a=0:1:k;
t=t+1;
n=n+10;
a
n
%analog wie zum ersten Programm zur Fehlerermittlung

f=zeros(n);
A=zeros(n,n);
for i=1:n
for j=1:n
if i == j
A(i,j)=-2;
endif
if j-1==i
A(i, j)=1;
endif
if j+1==i
A(i, j)=1;
endif
if i==1;
A(1, j)=0;
endif
if i==n;
A(n, j)=0;
endif
endfor
endfor
%Matrix


x=linspace(0, 1, n)';
```

```
h1=1/(n+1);

h2=1/n^3;

z=ones(n, 1);
y=ones(n,1);
b=0-h2;
%alles unterhalb der Matrix ist gleich wie
%beim ersten Vergleich
for i=0:h2:1
b=b+h2;
y(1)=1;
y(n)=exp(x(n)*b);
z(1)=1;
z(n)=exp(x(n)*b);

k1=((x-b^2).*exp(x.*b)+(1/(h1*h1))*A*y);
k2=((x-(b+h2/2)^2).*exp(x.*(b+h2/2))+(1/(h1*h1))*A*(y+h2/2*
   k1));
k3=((x-(b+h2/2)^2).*exp(x.*(b+h2/2))+(1/(h1*h1))*A*(y+h2/2*
   k2));
k4=((x-(b+h2)^2).*exp(x.*(b+h2))+(1/(h1*h1))*A*(y+k3*h2));
%Implementierung des Runge-Kutta-Verfahren
y=y+h2*(k1+2*k2+2*k3+k4)/6;

z=z+h2*((x-b^2).*exp(x.*b)+(1/(h1*h1))*A*z);
%Implementierung des Eulerverfahrens
f=exp(x.*b);
%Implementierung der exakten Funktion


endfor

c(a+1)=max(g=abs(f-y))
%Ermittlung des Fehlers zwischen Runge-Kutta
%und realer Loesung
d(a+1)=n
e(a+1)=max(j=abs(f-z))
%Ermittlung des Fehlers zwischen Eulerverfahren
%und realer Loesung
plot(d, c, 'r')
hold on
plot(d, e, 'b')
```

```
axis([0 t*10 0 0.07])
grid on
pause(0.001)
%hier werden anschliessend zwei Graphen ausgegeben
endfor
```

Wenn wir das Programm einmal kurz und einmal über einen längeren Zeitraum laufen lassen (wir sprechen von mehreren Stunden Berechnung als längere Dauer) so erhalten wir folgende Ausgabe:

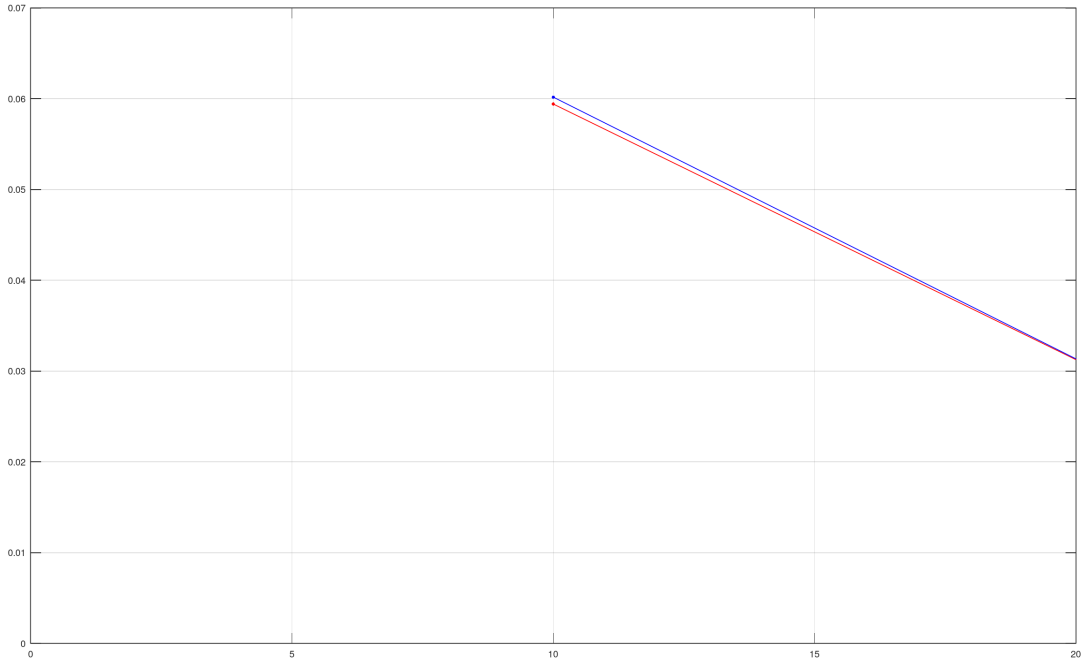

Abbildung 9: roter Graph Runge-Kutta-Verfahren, blauer Graph Euler-Verfahren

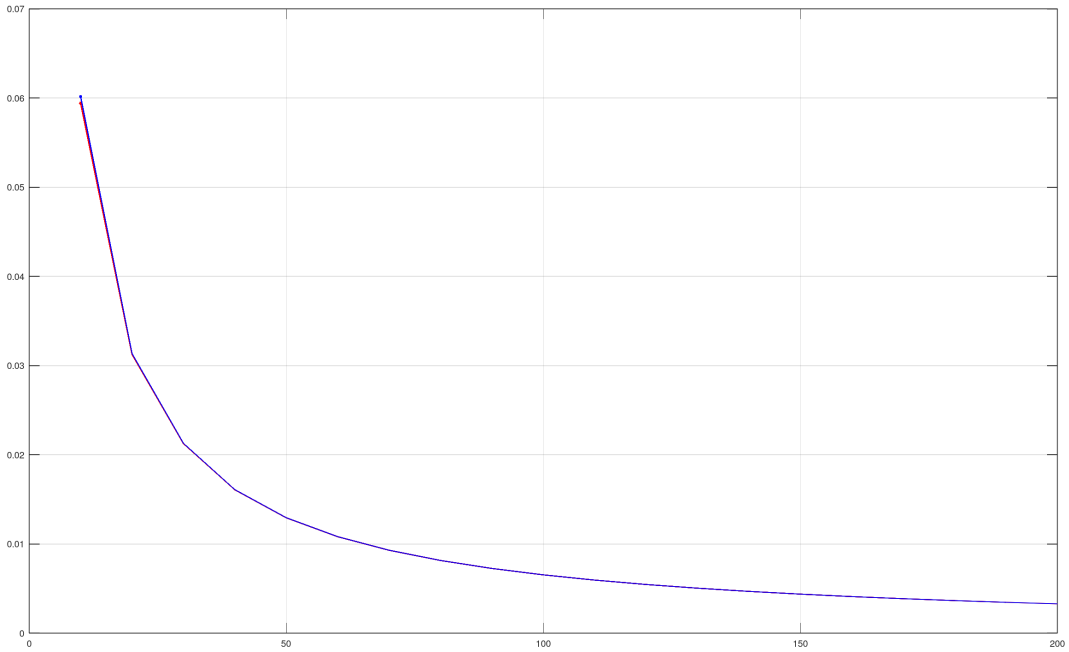

Abbildung 10: Fehlerentwicklung über längere Zeit

Wie beim vorherigen Vergleich, entspicht die x-Achse der Anzahl Stützpunkte n und die y-Achse der Grösse des Fehlers. Wir sehen, das Runge-Kutta-Verfahren ist ein wenig genauer als das Eulerverfahren, wenn auch nicht in grossem Masse.

10 Fazit

Wir haben nun festgestellt, dass man Differentialgleichungen mithilfe von numerischen Verfahren annähern kann, auch wenn diese nicht explizit lösbar sind. Von diesen numerischen Verfahren gibt es zahlreiche, wobei die vorher angeschauten Verfahren zu den eher "einfacheren"und "ungenaueren "gehören. Es lässt sich auch beobachten, dass die Verfahren immer komplexer und umfangreicher werden, je genauer man die Lösung einer Differentialgleichung annähern will. Deshalb ist auch das Runge-Kutta-Verfahren ein wenig "schwieriger "und exakter als das Eulerverfahren. Bei der Wahl der Verfahren kommt es auch darauf an, für welche Zwecke man die Annäherung braucht, das heisst, muss die Näherung relativ exakt sein oder darf es gröbere Abweichungen haben? Obwohl das Euler-Verfahren nicht sehr präzise ist gibt es trotzdem eine sehr gute Näherung mit einem maximalen Fehler von 0.005 in unserem Beispiel.

Literatur

[1] *Methode der finiten Differenzen*
https://www.math.tugraz.at/ ganster/lv_mathematik_2_bau_ss_2016/09_finite_differenzen.pdf

[2] Papula Lothar: *Mathematik für Ingenieure und Naturwissenschaftler ein Lehr- und Arbeitsbuch für das Grundstudium*

[3] Peter Jossen: *Partielle Differentialgleichungen, Zusammenfassung der Vorlesung*

[4] *The Explicid Euler Method*
http://www.maths.lth.se/na/courses/FMN050/media/material/part14.pdf

[5] Wikipedia: Explizites Euler-Verfahren

[6] Wikipedia: Differentialgleichung

[7] Wikipedia: *Finite-Differenzen-Methode*

[8] Wikipedia: *Partielle Ableitung*

[9] Wikipedia: *Runge-Kutta-Verfahren 4. Ordnung*

[10] Wikipedia: *Wärmeleitungsgleichung*

[11] item: *Euler-Verfahren*
https://glossar.item24.com/glossarindex/artikel/item/euler-verfahren.html

Einverständinserklärung

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit ohne fremde Hilfe und ohne Verwendung anderer als der angegebenen Hilfsmittel verfasst habe,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss der gängigen wissenschaftlichen Zitierregeln korrekt zitiert habe.

Ort / Datum Unterschrift: