# HM2 Summary (beliebig lang)

23. Februar, 2024; rev. 12. Juni 2024

Linda Riesen (rieselin)

## 1 Vorlesung 01

Numerik nichtlinearer Gleichungssysteme

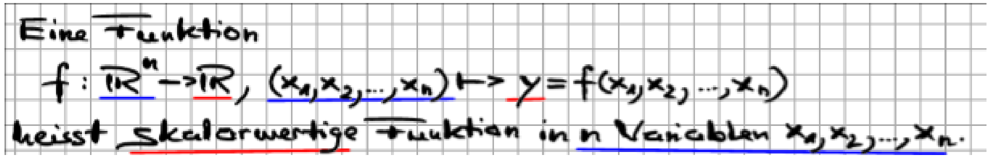



Abbildung 1: Definition Skalarwertige Funktion

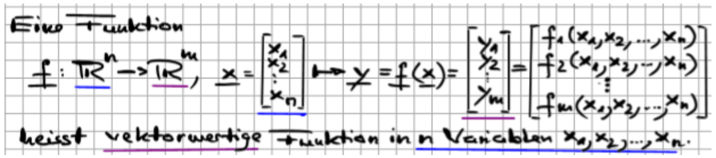



Abbildung 2: Definition Vektorwertige Funktion

## 2 Vorlesung 02

### 2.1 Ableitungen von Funktionen mit mehreren Variablen

Partielle Ableitung: Alle anderen Variablen werden als Konstanten betrachtet und dann nach xi Abgeleitet, Graphisch ist dies die Steigung des Graphen von f in Richtung xi

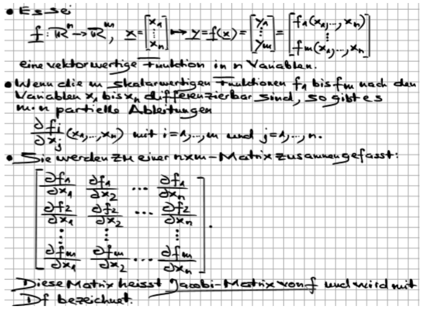


Abbildung 3: Jacobi Matrix

### 2.2 Linearisierung von Funktionen mit mehreren Variablen

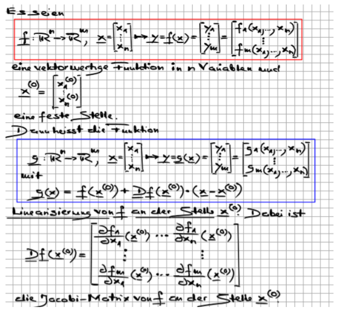
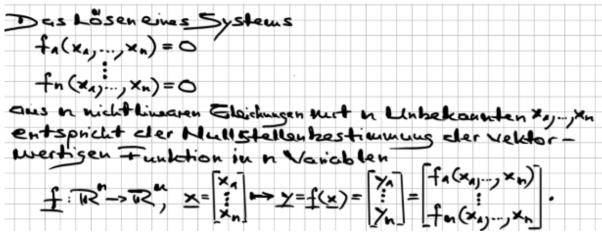


Abbildung 4: Definition Linearisierung mit skalarwertigen Funktionen

Graphisch: Graph g (Linearisierung von f an Stelle x0) ist n-dimensionale Tangentialebene an f

# 3 Vorlesung 03

## 3.1 Nichtlineare Gleichungssysteme als Nullstellenprobleme von Funktionen mit mehreren Variablen



Das Lösen eines Systems

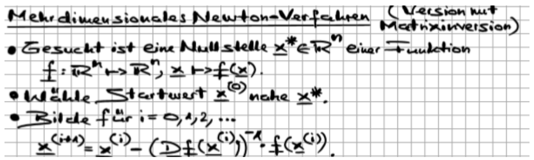
$$f_1(x_1, \dots, x_n) = 0$$
$$\vdots$$
$$f_n(x_1, \dots, x_n) = 0$$

aus $n$ nichtlinearen Gleichungen mit $n$ Unbekannten $x_1, \dots, x_n$ entspricht der Nullstellenbestimmung der vektorwertigen Funktion in $n$ Variablen

$$\underline{f}: \mathbb{R}^n \to \mathbb{R}^n, \quad \underline{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \mapsto \underline{y} = \underline{f}(\underline{x}) = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} f_1(x_1, \dots, x_n) \\ \vdots \\ f_n(x_1, \dots, x_n) \end{bmatrix}.$$

Abbildung 5: Nichtlineare Gleichungssysteme als Nullstellenprobleme v Funktionen mit mehreren Variablen

## 3.2 Newton-Verfahren zur Nullstellenbestimmung von Funktionen mit mehreren Variablen

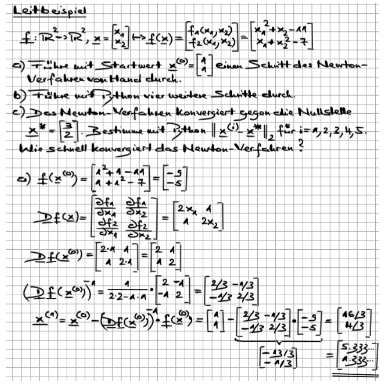


Mehrdimensionales Newton-Verfahren (Version mit Matrixinversion)

- Gesucht ist eine Nullstelle $\underline{x}^* \in \mathbb{R}^n$ einer Funktion $\underline{f}: \mathbb{R}^n \mapsto \mathbb{R}^n, \ \underline{x} \mapsto \underline{f}(\underline{x})$.
- Wähle Startwert $\underline{x}^{(0)}$ nahe $\underline{x}^*$.
- Bilde für $i = 0, 1, 2, \dots$

$$\underline{x}^{(i+1)} = \underline{x}^{(i)} - (\underline{D}\underline{f}(\underline{x}^{(i)}))^{-1} \cdot \underline{f}(\underline{x}^{(i)}).$$

Abbildung 6: Mehrdimensionales Newton Verfahren



Leitbeispiel

$$\underline{f}: \mathbb{R}^2 \to \mathbb{R}^2, \quad \underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \mapsto \underline{f}(\underline{x}) = \begin{bmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_1^2 + x_2 - 11 \\ x_1 + x_2^2 - 7 \end{bmatrix}$$

a) Führe mit Startwert $\underline{x}^{(0)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ einen Schritt des Newton-Verfahren von Hand durch.

b) Führe mit Python vier weitere Schritte durch.

c) Das Newton-Verfahren konvergiert gegen die Nullstelle $\underline{x}^* = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$. Bestimme mit Python $\|\underline{x}^{(i)} - \underline{x}^*\|_2$ für $i = 1, 2, 3, 4, 5$. Wie schnell konvergiert das Newton-Verfahren?

a) $$\underline{f}(\underline{x}^{(0)}) = \begin{bmatrix} 1^2 + 1 - 11 \\ 1 + 1^2 - 7 \end{bmatrix} = \begin{bmatrix} -9 \\ -5 \end{bmatrix}$$

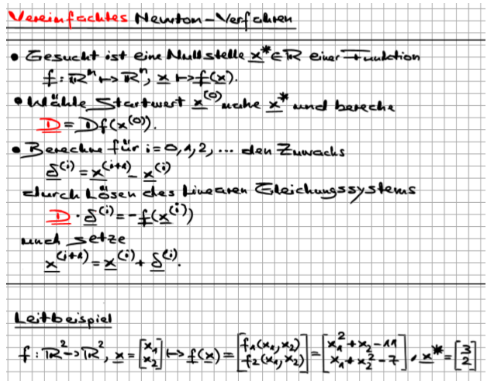
$$\underline{D}\underline{f}(\underline{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 & 1 \\ 1 & 2x_2 \end{bmatrix}$$

$$\underline{D}\underline{f}(\underline{x}^{(0)}) = \begin{bmatrix} 2 \cdot 1 & 1 \\ 1 & 2 \cdot 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$(\underline{D}\underline{f}(\underline{x}^{(0)}))^{-1} = \frac{1}{2 \cdot 2 - 1 \cdot 1} \cdot \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} = \begin{bmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{bmatrix}$$

$$\underline{x}^{(1)} = \underline{x}^{(0)} - (\underline{D}\underline{f}(\underline{x}^{(0)}))^{-1} \cdot \underline{f}(\underline{x}^{(0)}) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \underbrace{\begin{bmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{bmatrix} \cdot \begin{bmatrix} -9 \\ -5 \end{bmatrix}}_{\begin{bmatrix} -13/3 \\ -1/3 \end{bmatrix}} = \begin{bmatrix} 16/3 \\ 4/3 \end{bmatrix} = \begin{bmatrix} 5.333\dots \\ 1.333\dots \end{bmatrix}$$

Abbildung 7: Leitbeispiel Newtonverfahren



Vereinfachtes Newton-Verfahren

- Gesucht ist eine Nullstelle $\underline{x}^* \in \mathbb{R}$ einer Funktion $\underline{f}: \mathbb{R}^n \mapsto \mathbb{R}^n, \ \underline{x} \mapsto \underline{f}(\underline{x})$.
- Wähle Startwert $\underline{x}^{(0)}$ nahe $\underline{x}^*$ und berechne $\underline{D} = \underline{D}f(x^{(0)})$.
- Berechne für $i = 0, 1, 2, \dots$ den Zuwachs

$$\underline{\delta}^{(i)} = \underline{x}^{(i+1)} - \underline{x}^{(i)}$$

durch Lösen des linearen Gleichungssystems

$$\underline{D} \cdot \underline{\delta}^{(i)} = -\underline{f}(\underline{x}^{(i)})$$

und setze

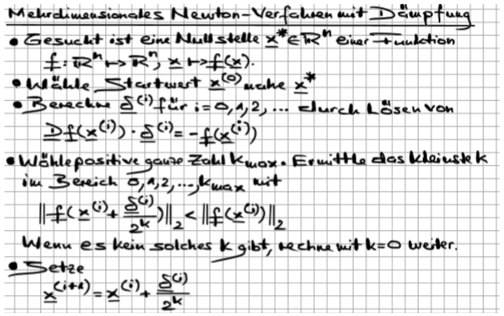
$$\underline{x}^{(i+1)} = \underline{x}^{(i)} + \underline{\delta}^{(i)}.$$

Leitbeispiel

$$f: \mathbb{R}^2 \to \mathbb{R}^2, \quad \underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \mapsto \underline{f}(\underline{x}) = \begin{bmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_1^2 + x_2 - 11 \\ x_1 + x_2^2 - 7 \end{bmatrix}, \quad \underline{x}^* = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

Abbildung 8: Vereinfachtes NewtonVerfahren

### 3.3 Newton Verfahren mit Dämpfung



Mehrdimensionales Newton-Verfahren mit Dämpfung

- Gesucht ist eine Nullstelle $\underline{x}^* \in \mathbb{R}^n$ einer Funktion $\underline{f}: \mathbb{R}^n \mapsto \mathbb{R}^n, \underline{x} \mapsto \underline{f}(\underline{x})$.
- Wähle Startwert $\underline{x}^{(0)}$ nahe $\underline{x}^*$
- Berechne $\underline{\delta}^{(i)}$ für $i = 0, 1, 2, \ldots$ durch Lösen von $D\underline{f}(\underline{x}^{(i)}) \cdot \underline{\delta}^{(i)} = -\underline{f}(\underline{x}^{(i)})$
- Wähle positive ganze Zahl $k_{max}$. Ermittle das kleinste $k$ im Bereich $0, 1, 2, \ldots, k_{max}$ mit $\|\underline{f}(\underline{x}^{(i)} + \frac{\underline{\delta}^{(i)}}{2^k})\|_2 < \|\underline{f}(\underline{x}^{(i)})\|_2$

  Wenn es kein solches $k$ gibt, rechne mit $k=0$ weiter.
- Setze $\underline{x}^{(i+1)} = \underline{x}^{(i)} + \frac{\underline{\delta}^{(i)}}{2^k}$

Abbildung 9: Mehrdimensionales Newtonverfahren Mit Dämpfung

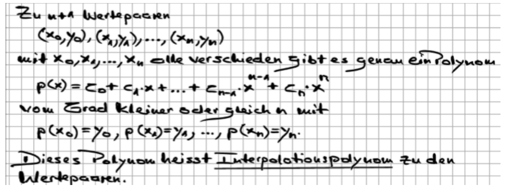
# 4 Vorlesung 04

### 4.1 Ausgleichsrechnung Problemstellung

Gesucht: Funktion deren Graph die gegebenen Wertepaare (/Stützpunkte) möglichst gut approximiert
Mögliche Lösungen

- Interpolation (Graph geht durch alle Stützpunkte)
  - Stückweise lineare Funktion (welche durch alle Stützpunkte geht)
  - Polynomfunktion (welche durch alle Stützpunkte geht)
- Regression (Graph nähert alle Stützpunkte möglichst gut an)
  - Lineare Funktion (welche Stützpunkte möglichst gut approximiert)
  - Sinusfunktion (welche Stützpunkte möglichst gut approximiert)

### 4.2 Ausgleichsrechnung Polynominterpolation



Zu $n+1$ Wertepaaren $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ mit $x_0, x_1, \ldots, x_n$ alle verschieden gibt es genau ein Polynom

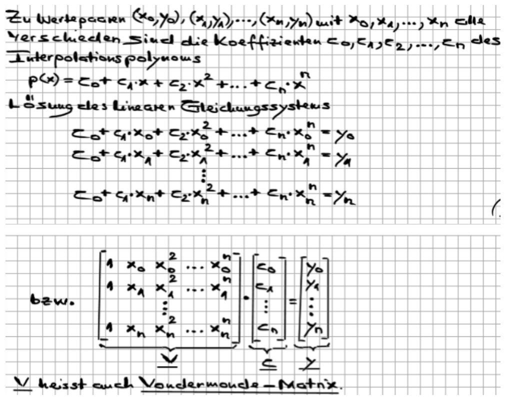
$p(x) = c_0 + c_1 \cdot x + \ldots + c_{n-1} \cdot x^{n-1} + c_n \cdot x^n$

vom Grad kleiner oder gleich $n$ mit

$p(x_0) = y_0, p(x_1) = y_1, \ldots, p(x_n) = y_n$.

Dieses Polynom heisst Interpolationspolynom zu den Wertepaaren.

Abbildung 10: Polynominterpolation



Zu Wertepaaren $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ mit $x_0, x_1, \ldots, x_n$ alle verschieden sind die Koeffizienten $c_0, c_1, c_2, \ldots, c_n$ des Interpolationspolynoms

$p(x) = c_0 + c_1 \cdot x + c_2 \cdot x^2 + \ldots + c_n \cdot x^n$

Lösung des linearen Gleichungssystems

$$\begin{aligned} c_0 + c_1 \cdot x_0 + c_2 \cdot x_0^2 + \ldots + c_n \cdot x_0^n &= y_0 \\ c_0 + c_1 \cdot x_1 + c_2 \cdot x_1^2 + \ldots + c_n \cdot x_1^n &= y_1 \\ &\vdots \\ c_0 + c_1 \cdot x_n + c_2 \cdot x_n^2 + \ldots + c_n \cdot x_n^n &= y_n \end{aligned}$$

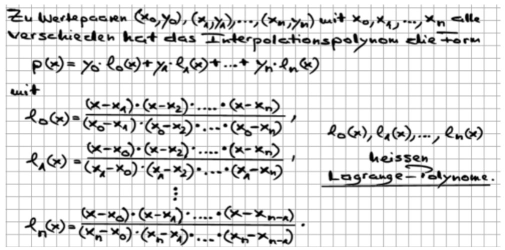
bzw.

$$\underbrace{\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ & & \vdots & & \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{bmatrix}}_{\underline{V}} \cdot \underbrace{\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix}}_{\underline{c}} = \underbrace{\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}}_{\underline{y}}$$

$\underline{V}$ heisst auch Vandermonde-Matrix.

Abbildung 11: Gleichungssystem Methode, Vandermonde Matrix

Die Vandermonde Matrix ist i.d.Regel sehr schlecht konditioniert daher ist Berechnung von Interpolationspolynom durch lösen des linearen Gleichungssystems stark fehlerbehaftet.



Zu Wertepaaren $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ mit $x_0, x_1, \ldots, x_n$ alle verschieden hat das Interpolationspolynom die Form

$p(x) = y_0 \cdot \ell_0(x) + y_1 \cdot \ell_1(x) + \ldots + y_n \cdot \ell_n(x)$

mit

$$\ell_0(x) = \frac{(x-x_1)\cdot(x-x_2)\cdot\ldots\cdot(x-x_n)}{(x_0-x_1)\cdot(x_0-x_2)\cdot\ldots\cdot(x_0-x_n)},$$

$$\ell_1(x) = \frac{(x-x_0)\cdot(x-x_2)\cdot\ldots\cdot(x-x_n)}{(x_1-x_0)\cdot(x_1-x_2)\cdot\ldots\cdot(x_1-x_n)},$$

$$\vdots$$

$$\ell_n(x) = \frac{(x-x_0)\cdot(x-x_1)\cdot\ldots\cdot(x-x_{n-1})}{(x_n-x_0)\cdot(x_n-x_1)\cdot\ldots\cdot(x_n-x_{n-1})}.$$

$\ell_0(x), \ell_1(x), \ldots, \ell_n(x)$ heissen Lagrange-Polynome.

Abbildung 12: Lagrange Formel, Lagrange Polynome

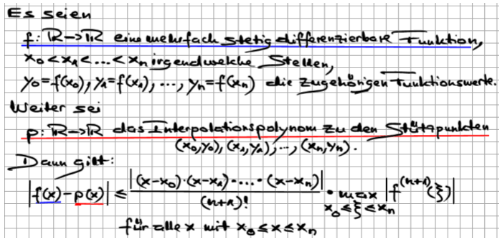



Abbildung 13: Fehlerabschätzung Polynominterpolation

# 5 Vorlesung 05

## 5.1 Splineinterpolation

Stückweise Interpolation der Daten durch 4 Polynome von Grad 3 mit jeweils gleicher Steigung und gleicher Krümmung an Übergängen
Natürliche Kubische Splineinterpolation

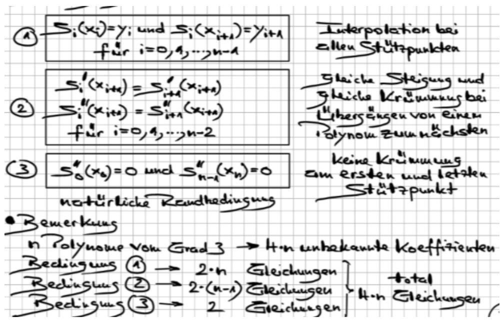



Abbildung 14: Bedingungen Spline Interpolation

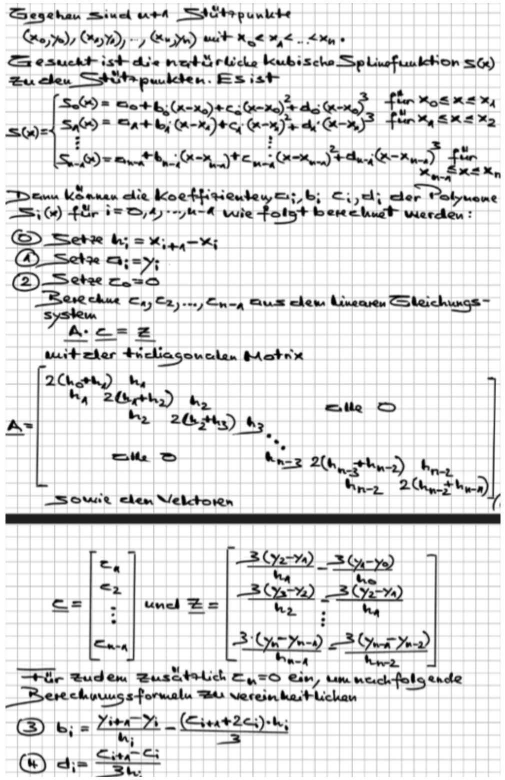



Abbildung 15: Lösung Spline Interpolation

## 5.2 Grundlagen der Regression

### 5.2.1 Problemstellung:

Gegeben: Wertepaare/Datenpunkte, Menge/Klasse von Funktionen (f)
Gesucht: Funktion welche Wertepaare bestmöglich approximiert sodass: $E(f) = (y_1 - f(x_1))^2 + ... + (y_n - f(x_n))^2$

### 5.2.2 Begriffe

- E = Fehlerfunktional

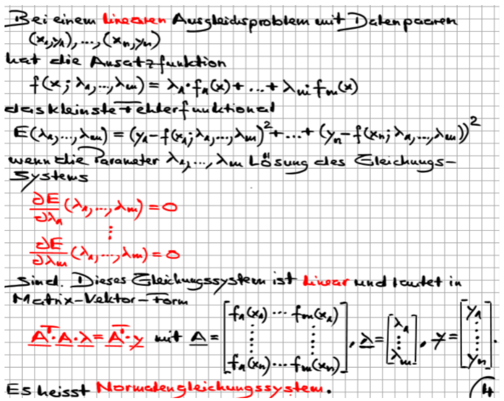
- Zugelassene Funktionen (f): Ansatzfunktionen
- Funktion f für welche E am kleinsten: Ausgleichs- / Regressionsfunktion, besitzt das kleinste Fehlerquadrat

### 5.2.3 Ausgleichsfunktion

Normalerweise geht Ausgleichsfunktion nicht durch alle Datenpunkte (Fehlerfunktional ist positiv)
Im Spezialfall geht Ausgleichsfunktion durch alle Datenpunkte: Interpolationsfunktion (Fehlerfunktional = 0)

**lineare / nicht lineare Ausgleichsprobleme** linear (ist Linearkombination): Ansatzfunktion von der Form: $f(x) = \lambda_1 * f_1(x) + ... + \lambda_m * f_m(x)$ mit festen Basisfunktionen f1(x)... fm(x) aus $R$ mit variableln Parametern $\lambda$
nicht linear (ist keine Linearkombination): keine Linearkombination von Basisfunktionen

# 6 Vorlesung 06



Bei einem linearen Ausgleichsproblem mit Datenpaaren $(x_1,y_1),\dots,(x_n,y_n)$
hat die Ansatzfunktion
$f(x;\lambda_1,\dots,\lambda_m) = \lambda_1 \cdot f_1(x) + \dots + \lambda_m f_m(x)$
das kleinste Fehlerfunktional
$E(\lambda_1,\dots,\lambda_m) = (y_1 - f(x_1;\lambda_1,\dots,\lambda_m))^2 + \dots + (y_n - f(x_n;\lambda_1,\dots,\lambda_m))^2$
wenn die Parameter $\lambda_1,\dots,\lambda_m$ Lösung des Gleichungs-Systems

$\frac{\partial E}{\partial \lambda_1}(\lambda_1,\dots,\lambda_m) = 0$
$\vdots$
$\frac{\partial E}{\partial \lambda_m}(\lambda_1,\dots,\lambda_m) = 0$

sind. Dieses Gleichungssystem ist linear und lautet in Matrix-Vektor-Form

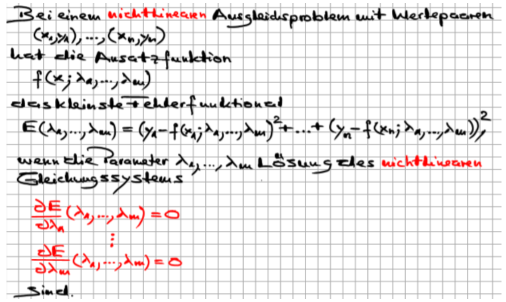
$$\underline{A}^T \cdot \underline{A} \cdot \underline{\lambda} = \underline{A}^T \cdot \underline{y} \quad \text{mit } \underline{A} = \begin{bmatrix} f_1(x_1) & \cdots & f_m(x_1) \\ \vdots & & \vdots \\ f_1(x_n) & \cdots & f_m(x_n) \end{bmatrix}, \underline{\lambda} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix}, \underline{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

Es heisst Normalengleichungssystem.

(4)

Abbildung 16: Lösungsverfahren Lineare AUsgleichsprobleme

# 7 Vorlesung 08



Bei einem nichtlinearen Ausgleichsproblem mit Wertepaaren $(x_1,y_1),\dots,(x_n,y_n)$
hat die Ansatzfunktion
$f(x;\lambda_1,\dots,\lambda_m)$
das kleinste Fehlerfunktional
$E(\lambda_1,\dots,\lambda_m) = (y_1 - f(x_1;\lambda_1,\dots,\lambda_m))^2 + \dots + (y_n - f(x_n;\lambda_1,\dots,\lambda_m))^2$,
wenn die Parameter $\lambda_1,\dots,\lambda_m$ Lösung des nichtlinearen Gleichungssystems

$\frac{\partial E}{\partial \lambda_1}(\lambda_1,\dots,\lambda_m) = 0$
$\vdots$
$\frac{\partial E}{\partial \lambda_m}(\lambda_1,\dots,\lambda_m) = 0$

sind.

Abbildung 17: Nichtlineares Ausgleichsproblem

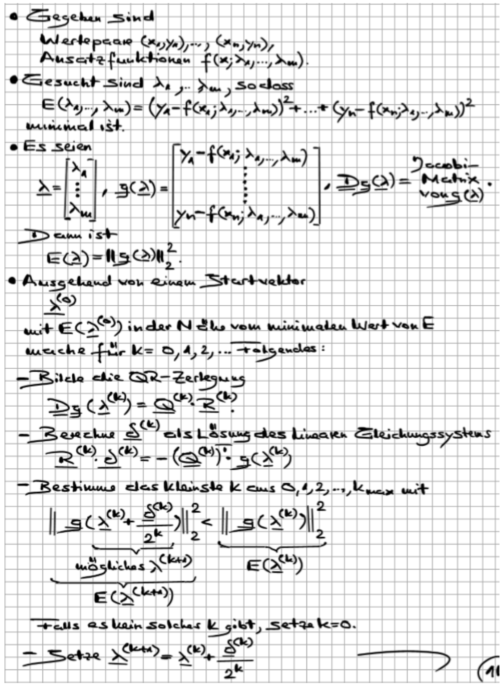
- Gegeben sind
  Wertepaare $(x_1,y_1),\dots,(x_n,y_n)$,
  Ansatzfunktionen $f(x;\lambda_1,\dots,\lambda_m)$.
- Gesucht sind $\lambda_1,\dots,\lambda_m$, sodass
  $E(\lambda_1,\dots,\lambda_m) = (y_1 - f(x_1;\lambda_1,\dots,\lambda_m))^2 + \dots + (y_n - f(x_n;\lambda_1,\dots,\lambda_m))^2$
  minimal ist.
- Es seien

$$\underline{\lambda} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix}, \underline{g}(\underline{\lambda}) = \begin{bmatrix} y_1 - f(x_1;\lambda_1,\dots,\lambda_m) \\ \vdots \\ y_n - f(x_n;\lambda_1,\dots,\lambda_m) \end{bmatrix}, \underline{Dg}(\underline{\lambda}) = \text{Jacobi-Matrix von } \underline{g}(\underline{\lambda}).$$

Dann ist
$E(\underline{\lambda}) = \|\underline{g}(\underline{\lambda})\|_2^2$.

- Ausgehend von einem Startvektor $\underline{\lambda}^{(0)}$
  mit $E(\underline{\lambda}^{(0)})$ in der Nähe vom minimalen Wert von E
  mache für k = 0, 1, 2, ... Folgendes:
  - Berechne $\underline{\delta}^{(k)}$ als Lösung des linearen Gleichungssystems
    $\underline{Dg}(\underline{\lambda}^{(k)})^T \cdot \underline{Dg}(\underline{\lambda}^{(k)}) \cdot \underline{\delta}^{(k)} = -\underline{Dg}(\underline{\lambda}^{(k)})^T \underline{g}(\underline{\lambda}^{(k)})$
  - Setze
    $\underline{\lambda}^{(k+1)} = \underline{\lambda}^{(k)} + \underline{\delta}^{(k)}$

Abbildung 18: Gauss Newton Verfahren ohne Dämpfung



- Gegeben sind
  Wertepaare $(x_1, y_1), \dots, (x_n, y_n)$,
  Ansatzfunktionen $f(x; \lambda_1, \dots, \lambda_m)$.
- Gesucht sind $\lambda_1, \dots, \lambda_m$, sodass
  $E(\lambda_1, \dots, \lambda_m) = (y_1 - f(x_1; \lambda_1, \dots, \lambda_m))^2 + \dots + (y_n - f(x_n; \lambda_1, \dots, \lambda_m))^2$
  minimal ist.
- Es seien
  $\underline{\lambda} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix}$, $\underline{g}(\underline{\lambda}) = \begin{bmatrix} y_1 - f(x_1; \lambda_1, \dots, \lambda_m) \\ \vdots \\ y_n - f(x_n; \lambda_1, \dots, \lambda_m) \end{bmatrix}$, $\underline{\underline{D}}\underline{g}(\underline{\lambda}) =$ Jacobi-Matrix von $\underline{g}(\lambda)$.
  Dann ist
  $E(\underline{\lambda}) = \|\underline{g}(\underline{\lambda})\|_2^2$.
- Ausgehend von einem Startvektor
  $\underline{\lambda}^{(0)}$
  mit $E(\underline{\lambda}^{(0)})$ in der Nähe vom minimalen Wert von E
  mache für $k = 0, 1, 2, \dots$ Folgendes:
  - Bilde die QR-Zerlegung
    $\underline{\underline{D}}\underline{g}(\underline{\lambda}^{(k)}) = \underline{\underline{Q}}^{(k)} \cdot \underline{\underline{R}}^{(k)}$.
  - Berechne $\underline{\delta}^{(k)}$ als Lösung des linearen Gleichungssystems
    $\underline{\underline{R}}^{(k)} \cdot \underline{\delta}^{(k)} = -(\underline{\underline{Q}}^{(k)})^T \cdot \underline{g}(\underline{\lambda}^{(k)})$
  - Bestimme das kleinste k aus $0, 1, 2, \dots, k_{max}$ mit
    $\underbrace{\|\underline{g}(\underbrace{\underline{\lambda}^{(k)} + \frac{\underline{\delta}^{(k)}}{2^k}}_{\text{mögliches } \lambda^{(k+1)}})\|_2^2}_{E(\underline{\lambda}^{(k+1)})} < \underbrace{\|\underline{g}(\underline{\lambda}^{(k)})\|_2^2}_{E(\underline{\lambda}^{(k)})}$
    Falls es kein solches k gibt, setze k=0.
  - Setze $\underline{\lambda}^{(k+1)} = \underline{\lambda}^{(k)} + \frac{\underline{\delta}^{(k)}}{2^k}$

Abbildung 19: Gauss Newton Verfahren mit Dämpfung

# 8 Vorlesung 09

Ziel: Möglichst genauer Näherungswert des bestimmten Integrals (oft nicht algebraisch berechenbar) (einer mindest. stetigen Funktion)

- **Rechteckregel:** Approximation von f(x) durch **konstante** Funktion
- **Trapezregel:** Approximation von f(x) durch **lineare** Funktion
- **Simpsonregel:** Approximation von f(x) durch **quadratische** Funktion

## 8.1 Rechteckregel und Trapezregel

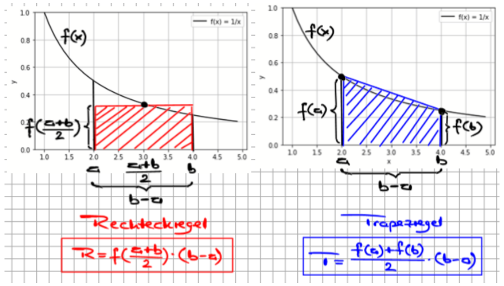



Abbildung 20: Rechteck- und Trapezregel

### 8.1.1 Summierte Rechteck- und Trapezregel

Summierte Rechteckregel: $R(h) = h * \sum_{i=0}^{n-1} f(x_i + \frac{h}{2})$ mit $[h = \frac{b-a}{n}, x_i = a + i * h]$
Summierte Trapezregel: $T(h) = h * (\frac{f(x_0+f(x_n}{2} + \sum_{i=0}^{n-1} f(x_i))$ mit $[h = \frac{b-a}{n}, x_i = a + i * h]$

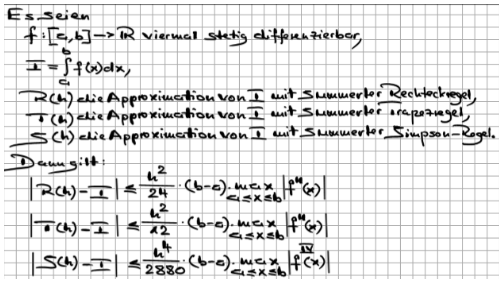
## 8.2 Simpsonregel

Simpsonregel: $S = \frac{b-a}{3} * (\frac{f(a)}{2} + 2 * f(\frac{a+b}{2}) + \frac{f(b)}{2})$

### 8.2.1 Summierte Simpsonregel

Summierte Simpsonregel: $S(h) = \frac{h}{3}(\frac{f(x_0)+f(x_n)}{2} + \sum_{i=0}^{n-1} f(x_i) + 2 * \sum_{i=0}^{n-1} f(x_i + \frac{h}{2}))$ mit $[h = \frac{b-a}{n}, x_i = a + i * h]$

### 8.3 Fehlerabschätzung



Es seien

$f : [a,b] \rightarrow \mathbb{R}$ viermal stetig differenzierbar,

$I = \int_a^b f(x)dx$,

$R(h)$ die Approximation von $I$ mit summierter Rechteckregel,

$T(h)$ die Approximation von $I$ mit summierter Trapezregel,

$S(h)$ die Approximation von $I$ mit summierter Simpson-Regel.

Dann gilt:

$$|R(h) - I| \leq \frac{h^2}{24} \cdot (b-a) \cdot \max_{a \leq x \leq b} |f''(x)|$$

$$|T(h) - I| \leq \frac{h^2}{12} \cdot (b-a) \cdot \max_{a \leq x \leq b} |f''(x)|$$

$$|S(h) - I| \leq \frac{h^4}{2880} \cdot (b-a) \cdot \max_{a \leq x \leq b} |f^{IV}(x)|$$

Abbildung 21: Fehlerabschätzung

## 9 Vorlesung 10

### 9.1 Gauss-Formel

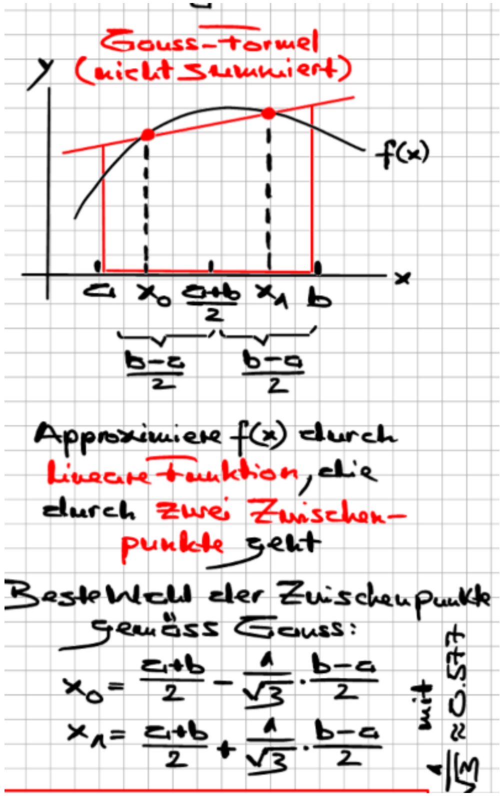



Abbildung 22: Gauss Formel

$$G = \frac{f(\frac{a+b}{2} - \frac{1}{\sqrt{3}} * \frac{a-b}{2}) + f(\frac{a+b}{2} + \frac{1}{\sqrt{3}} * \frac{a-b}{2})}{2} * (b-a)$$

```
# Implementation
def sum_gauss(f, a, b, h):
    n = int((b - a)/h)
    G = 0
    alpha = 1/3**0.5
    for i in range(0, n):
        xm = a + i*h + h/2 # mittlerer x-Wert des i-ten Teilintervalls
        G = G + f(xm - alpha*h/2) + f(xm + alpha*h/2)
    G = G/2*h
    return G
```

Abbildung 23: Summierte Gauss-Formel

```
# Fehler fuer h = 1, 0.1, 0.01,
kmax = 5
for k in range(0, kmax+1):
    h = 10.**(-k)
    G = sum_gauss(f, a, b, h)
    print(h, G, np.abs(G - I))
```

Abbildung 24: Fehler Gauss-Formel

## 9.2 Romberg-Extrapolation

Es seien

$I = \int_a f(x)dx$,

$T(h)$ die Approximation von $I$ mit Summierter Trapezregel.

Dann heisst

$$E(h) = \frac{4 \cdot T(h/2) - T(h)}{3}$$

einstufige Romberg-Extrapolation.

Abbildung 25: Romberg-Extrapolation

- Extrapolationsschema mit drei Stufen

Stufe 0

$T_{00} = T(h/2^0)$

$T_{10} = T(h/2^1)$

$T_{20} = T(h/2^2)$

$T_{30} = T(h/2^3)$

Stufe 1

$$T_{01} = \frac{4^1 \cdot T_{10} - T_{00}}{4^1 - 1}$$

$$T_{11} = \frac{4^1 \cdot T_{20} - T_{10}}{4^1 - 1}$$

$$T_{21} = \frac{4^1 \cdot T_{30} - T_{20}}{4^1 - 1}$$

Stufe 2

$$T_{02} = \frac{4^2 \cdot T_{11} - T_{01}}{4^2 - 1}$$

$$T_{12} = \frac{4^2 \cdot T_{21} - T_{11}}{4^2 - 1}$$

Stufe 3

$$\boxed{T_{03} = \frac{4^3 \cdot T_{12} - T_{02}}{4^3 - 1}}$$

Abbildung 26: Mehrstufige Romberg-Extrapolation

# 10 Vorlesung 11

## 10.1 Definitionen DGL

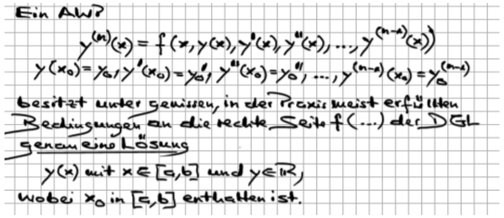
- Eine gewöhnliche Differentialgleichung (DGL) ist eine Gleichung für eine gesuchte Funktion
  $y(x)$ mit $x \in [a,b]$ und $y \in \mathbb{R}$,
  wobei in der Gleichung $y(x)$, mindestens eine ihrer Ableitungen $y'(x), y''(x), y'''(x), \dots$ und allenfalls $x$ auftreten:
  $$F(x, y(x), y'(x), y''(x), y'''(x), \dots) = 0$$
- Eine DGL hat Ordnung $n$ mit $n \in \mathbb{N}$, wenn die höchste auftretende Ableitung $y^{(n)}(x)$ ist:
  $$F(x, y(x), y'(x), y''(x), \dots, y^{(n)}(x)) = 0$$
- Eine DGL $n$-ter Ordnung heisst explizit, wenn die Gleichung nach $y^{(n)}(x)$ aufgelöst ist:
  $$y^{(n)}(x) = f(x, y(x), y'(x), y''(x), \dots, y^{(n-1)}(x))$$

Eine nicht explizite DGL heisst implizit.

Abbildung 27: Definiton Begriffe DGL

### 10.1.1 AWP (Anfangswertproblem)

AWP = DGL n-ter Ordnung und Vorgabe Funktionswert von y(x) sowie Werte der Ableitungen an Stelle x0


Ein AWP

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

$$y(x_0) = y_0, y'(x_0) = y_0', y''(x_0) = y_0'', \ldots, y^{(n-1)}(x_0) = y_0^{(n-1)}$$

besitzt unter gewissen, in der Praxis meist erfüllten Bedingungen an die rechte Seite $f(\ldots)$ der DGL genau eine Lösung

$y(x)$ mit $x \in [a,b]$ und $y \in \mathbb{R}$,

wobei $x_0$ in $[a,b]$ enthalten ist.

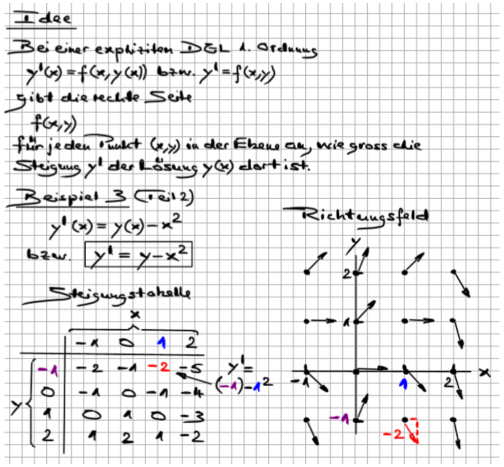
Abbildung 28: Existenz und Eindeutigkeit der Lösung eines AWP

**Existenz u Eindeutigkeit**

## 10.2 Anwendungen

$\ddot{s}(t) = -\frac{k}{m} * s(t)$ [mit k = Federkonstante, Ort s(t)]

## 10.3 Richtungsfeld von expliziter DGL 1. Ordnung


Idee

Bei einer expliziten DGL 1. Ordnung

$y'(x) = f(x, y(x))$ bzw. $y' = f(x,y)$

gibt die rechte Seite

$f(x,y)$

für jeden Punkt $(x,y)$ in der Ebene an, wie gross die Steigung $y'$ der Lösung $y(x)$ dort ist.

Beispiel 3 (Teil 2)

$y'(x) = y(x) - x^2$

bzw. $\boxed{y' = y - x^2}$

Steigungstabelle

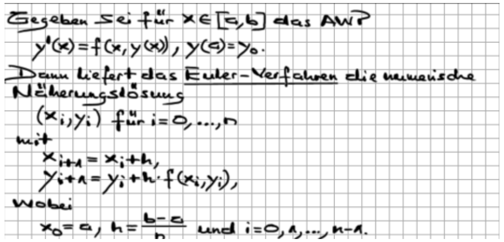
| $y$ \ $x$ | -1 | 0 | 1 | 2 |
|---|---|---|---|---|
| -1 | -2 | -1 | -2 | -5 |
| 0 | -1 | 0 | -1 | -4 |
| 1 | 0 | 1 | 0 | -3 |
| 2 | 1 | 2 | 1 | -2 |

$y' = (-1) - 1^2$



Abbildung 29: Richtungsfeld

## 10.4 Euler-Verfahren


Gegeben sei für $x \in [a,b]$ das AWP

$y'(x) = f(x, y(x))$, $y(a) = y_0$.

Dann liefert das Euler-Verfahren die numerische Näherungslösung

$(x_i, y_i)$ für $i = 0, \ldots, n$

mit

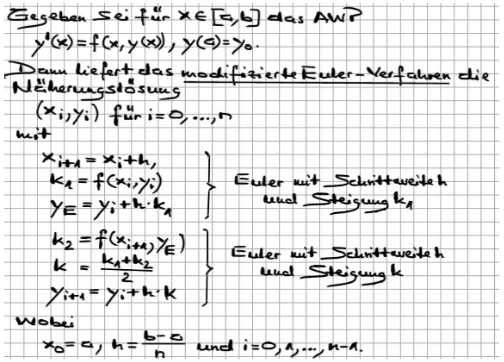
$x_{i+1} = x_i + h$,

$y_{i+1} = y_i + h \cdot f(x_i, y_i)$,

wobei

$x_0 = a$, $h = \frac{b-a}{n}$ und $i = 0, 1, \ldots, n-1$.

Abbildung 30: Euler-Verfahren


Gegeben sei für $x \in [a,b]$ das AWP

$y'(x) = f(x, y(x))$, $y(a) = y_0$.

Dann liefert das modifizierte Euler-Verfahren die Näherungslösung

$(x_i, y_i)$ für $i = 0, \ldots, n$

mit

$x_{i+1} = x_i + h$,
$k_1 = f(x_i, y_i)$
$y_E = y_i + h \cdot k_1$

Euler mit Schrittweite h und Steigung $k_1$

$k_2 = f(x_{i+1}, y_E)$
$k = \frac{k_1 + k_2}{2}$
$y_{i+1} = y_i + h \cdot k$

Euler mit Schrittweite h und Steigung k

wobei

$x_0 = a$, $h = \frac{b-a}{n}$ und $i = 0, 1, \ldots, n-1$.

Abbildung 31: Modifiziertes Euler-Verfahren

# 11 Vorlesung 12

## 11.1 Mittelpunktverfahren

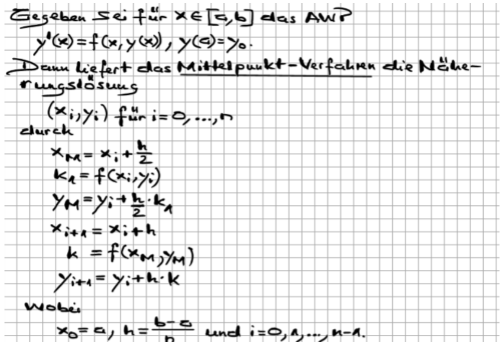



Abbildung 32: Mittelpunkt Verfahren Algorithmus

## 11.2 Rundungsfehler vs Diskretierungsfehler

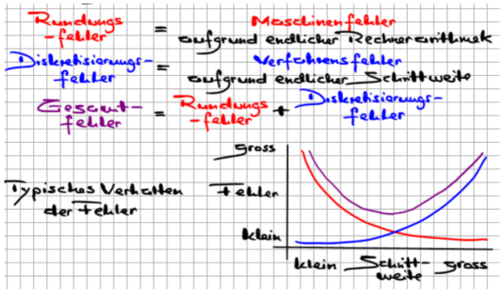



Abbildung 33: Rundungs vs Diskretisierungsfehler

## 11.3 Lokaler + Globaler Fehler, Konvergenzordnung

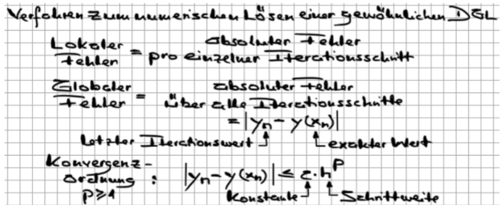



Abbildung 34: Lokaler Fehler, Globaler Fehler, Konvergenzordnung

## 11.4 Runge-Kutta-Verfahren (Runge Kutta Verfahren)

### 11.4.1 Kassisches vierstufiges Runge-Kutta-Verfahren

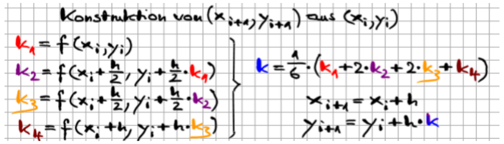



Abbildung 35: Klassisches Vierstufiges Runge Kutta Verfahren

### 11.4.2 Allgemeines vierstufiges Runge-Kutta-Verfahren

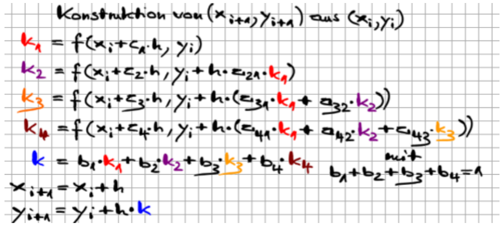



Abbildung 36: Allgemeines Vierstufiges Runge-Kutta

### 11.4.3 Zweistufige u Dreistufige Runge-Kutta-Verfahren

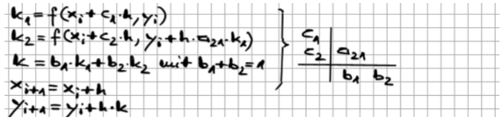



Abbildung 37: Zweistufiges Runge-Kutta 2

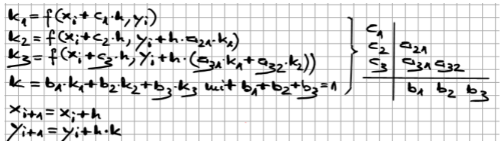



Abbildung 38: Dreistufiges Runge-Kutta 3

# 12 Vorlesung 13

Numerisches Lösen von gewöhnlichen Differnetialgleichungen

## 12.1 Systeme von expliziten Differenzialgleichungen 1. Ordnung

Gesucht m Funktionen $y_1(x), y_2(x)...y_m(x)$ welche alle m DGL und alle m AnfangsWerte erfüllen (bzw. numerische Approximation davon)

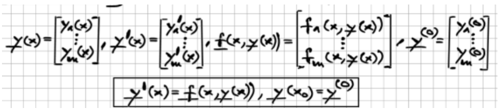



Abbildung 39: Vektorisierung

Vorher genannte Verfahren können durch Vektorisierung angewendet werden.

### 12.1.1 Beispiel - Anwendung

- Bilanzgleichung: $S(t) + I(t) + R(t) = N$ [S = suspectible (noch nicht erkrankt), I = infected, R = removed (genesen/verstorben)]
  - $S'(t) = -a * S(t) * I(t)$
  - $I'(t) = a * S(t) * I(t) - b * I(t)$

## 12.2 Einzelne explizite Differentzialgleichungen höherer Ordnung

Umwandlung DGL höherer Ordnung in System von mehreren DGL erster Ordnung

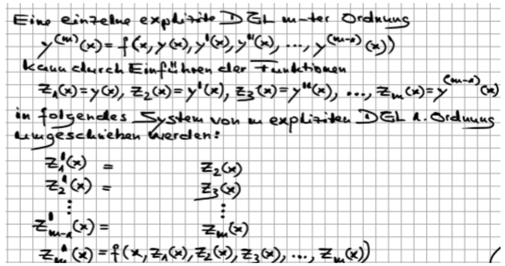



Abbildung 40: Explizite DGL m-ter Ordnung