Nichtlineare Gleichungssysteme

Funktionen mit mehreren Variablen

auch multivariat gennant, hier nur skalarwertige Funktionen (keine Komplexen Zahlen). Definition:

Explizite Darstellung Funktionsgleichung ist nach einer variablen aufgelöst.

$$y = f(x_1, x_2, ..., x_n)$$

Implizite Darstellung nicht nach einer Variablen aufgelöst(nur n-1 unabhängige Variablen)

$$F(x_1, x_2, ..., x_n) = 0$$

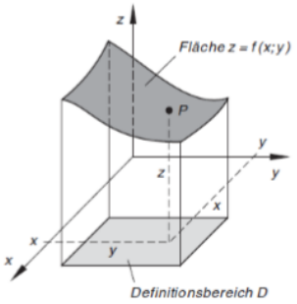
für vektorwertige funktionen

$$\vec{f}(x_1, ..., x_n) = \begin{pmatrix} y_1 = f_1(x_1, ..., x_n) \\ \dots \\ y_m = f_m(x_1, ..., x_n) \end{pmatrix}$$

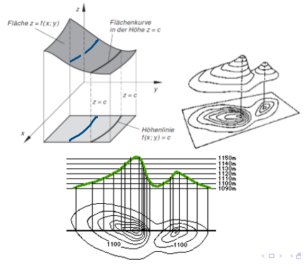
Graphische Darstellungsformen

Funktionen mit 2 Variablen können 3D dargestellt werden.
Interpretieren als $z = f(x, y)$

Fläche Punkte $(x, y, f(x, y))$

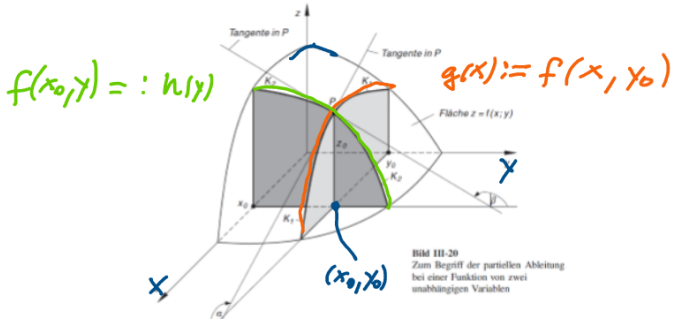


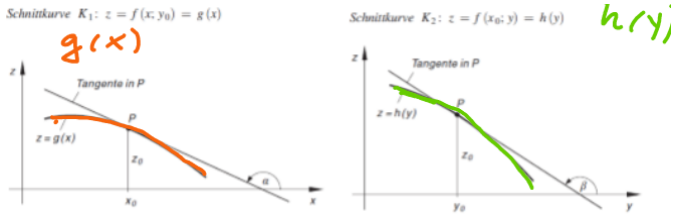
Schnittkurve bei konstanter Höhe z, auch Höhen-/Contour-Plot



Partielle Ableitungen

Nur eine der Variablen wird abgeleitet, der Rest als Konstante behandelt. Visuell entspricht dies der Steigung an einer Flächentangente.





Partielle Ableitungen 1. Ordnung

Beispiel nach x abgeleitet(normale Ableitungsregeln, andere Variablen als Konstanten betrachten):

$$\frac{\partial f}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x, y) - f(x, y)}{\Delta x}$$

$$f(x, y) = 3xy^3 + 10x^2y + 5y + 3y * sin(5xy)$$
$$\frac{\partial f}{\partial x} = 3 * 1 * y^3 + 10 * 2x + 0 + 3y * cos(5xy) * 5 * 1 * y$$

Linearisierung

Repetition Tangentengleichung

Dient als Annäherung für eindimensionale $f(x)$ in der Nähe von x_0 (Linearisierung): $g(x) = f(x_0) + f'(x_0)(x - x_0)$

Jacobi-Matrix

Sozusagen wie Tangentengleichung aber für mehrere Variablen

$$\vec{y} = \vec{f}(\vec{x}) = \begin{pmatrix} y_1 = f_1(x_1, ..., x_n) \\ \dots \\ y_m = f_m(x_1, ..., x_n) \end{pmatrix}$$

Die Jacobi-Matrix enthält sämtliche Partielle Ableitungen 1. Ordnung von $\vec{f}$.
Auf jeder Spalte bleibt die funktion f_j die gleiche und in den Zeilen $x_i \rightarrow \frac{\partial f_j}{\partial x_i}$

$$D\vec{f}(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) & \frac{\partial f_1}{\partial x_2}(x) & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1}(x) & \frac{\partial f_2}{\partial x_2}(x) & \dots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1}(x) & \frac{\partial f_m}{\partial x_2}(x) & \dots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

verallgemeinerte Tangentengleichung

$$\vec{g}(\vec{x}) = \vec{f}(\vec{x}^{(0)}) + D\vec{f}(\vec{x}^{(0)}) * (\vec{x} - \vec{x}^{(0)})$$

ist eine lineare Funktion und für $\vec{x}$ in der Umgebung von $\vec{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})$ gilt $\vec{f}(\vec{x}) \approx \vec{g}(\vec{x})$

Df entspricht der obigen Funktion zur Erzeugung einer Jacobi-Matrix.

Hochgestellte Zahlen in Klammern $(x^{(n)})$ stehen wie zuvor für eine Variable nach n Iterationsschritten.

Tangentialebene

- Für den speziellen Fall $f : \mathbb{R}^2 \longrightarrow \mathbb{R}$ mit $y = f(x_1, x_2)$ und $\boldsymbol{x}^{(0)} = (x_1^{(0)}, x_2^{(0)})^T \in \mathbb{R}^2$ ist die Jacobi-Matrix nur ein Zeilenvektor mit zwei Elementen, nämlich

$$\boldsymbol{Df}(\boldsymbol{x}^{(0)}) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(x_1^{(0)}, x_2^{(0)}) & \frac{\partial f}{\partial x_2}(x_1^{(0)}, x_2^{(0)}) \end{pmatrix}.$$

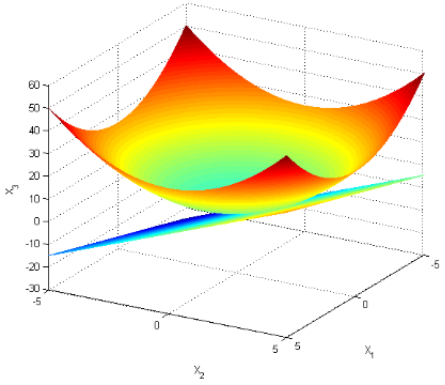
  Dann liefert die Linearisierung

$$\begin{aligned} g(x_1, x_2) &= f(x_1^{(0)}, x_2^{(0)}) + \begin{pmatrix} \frac{\partial f}{\partial x_1}(x_1^{(0)}, x_2^{(0)}) & \frac{\partial f}{\partial x_2}(x_1^{(0)}, x_2^{(0)}) \end{pmatrix} \cdot \begin{pmatrix} x_1 - x_1^{(0)} \\ x_2 - x_2^{(0)} \end{pmatrix} \\ &= f(x_1^{(0)}, x_2^{(0)}) + \frac{\partial f}{\partial x_1}(x_1^{(0)}, x_2^{(0)}) \cdot \begin{pmatrix} x_1 - x_1^{(0)} \end{pmatrix} + \frac{\partial f}{\partial x_2}(x_1^{(0)}, x_2^{(0)}) \cdot \begin{pmatrix} x_2 - x_2^{(0)} \end{pmatrix} \end{aligned}$$

  die Gleichung der **Tangentialebene**.

- Sie enthält sämtliche im Flächenpunkt $P = (x_1^{(0)}, x_2^{(0)}, f(x_1^{(0)}, x_2^{(0)}))$ an die Bildfläche von $y = f(x_1, x_2)$ angelegten Tangenten.

Graphische Darstellung der Fläche $x_3 = f(x_1, x_2) = x_1^2 + x_2^2$ sowie Tangentialebene durch den Flächenpunkt $(x_1^{(0)} = 1, x_2^{(0)} = 2, f(x^{(0)}) = 5)$



Nullstellenbestimmung für nichtlineare Systeme

- Gegeben: $n \in \mathbb{N}$ und eine Funktion $\vec{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$
- Gesucht: $\vec{x} \in \mathbb{R}^n \quad \text{mit} \quad \vec{f}(\vec{x}) = 0$

$$\vec{f}(x_1, ..., x_n) = \begin{pmatrix} f_1(x_1, ..., x_n) \\ \vdots \\ f_n(x_1, ..., x_n) \end{pmatrix} = \vec{0}$$

Newton-Verfahren für Systeme

Herleitung

Repetition 1-Dimensional: (nur für $f : \mathbb{R} \rightarrow \mathbb{R}$)

Aus der Linearisierung der Funktion f mittels der Tangente g an der Selle x_n

$$f(x) \approx g(x) = f(x_n) + f'(x_n)(x - x_n)$$

folgte die Iteration

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad (n = 0, 1, 2, 3, ...).$$

Mit der Jacobi-Matrix $Df(x)$ kann das analog für Vektor-wertige Funktionen $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ angewendet werden.

$$\vec{x}^{(n+1)} = \vec{x}^{(n)} - (Df(\vec{x}^{(n)}))^{-1} * \vec{f}(\vec{x}^{(n)})$$

Das Inverse der Jacobi-Matrix wird aber nie berechnet sondern die obige Gleichung via Substitution als lineares Gleichungsystem aufgefasst.

$$\boldsymbol{\delta}^{(n)} := -\left(\boldsymbol{Df}(\boldsymbol{x}^{(n)})\right)^{-1} \cdot \boldsymbol{f}(\boldsymbol{x}^{(n)})$$

als lineares Gleichungssystem auffasst gemäss

$$\boldsymbol{Df}(\boldsymbol{x}^{(n)})\boldsymbol{\delta}^{(n)} = -\boldsymbol{f}(\boldsymbol{x}^{(n)})$$

und so δ^n bestimmen und anschliessend

$$\boldsymbol{x}^{(n+1)} := \boldsymbol{x}^{(n)} + \boldsymbol{\delta}^{(n)}$$

Quadratisch-konvergentes Newton-Verfahren für Systeme

Gesucht: Nullstellen von $\vec{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ mit Startvektor $\vec{x}^{(0)}$ Nahe der Nullstelle.
es kann passieren, dass das Newton-Verfahren Statt einer Nullstelle ein lokales Minimum $x_{min}! = 0$ findet, in diesem Fall ist $Df(x_{min})$ aber immer nicht regulär.

für $n = 0, 1, ...$:

1. Berechne $\delta^{(n)}$ als Lösung des linearen Gleichungsystems

$$D\vec{f}(\vec{x}^{(n)})\delta^{(n)} = -\vec{f}(\vec{x}^{(n)})$$

2. Setze

$$\vec{x}^{(n+1)} = \vec{x}^{(n)} + \delta^{(n)}$$

mögliche Abbruchkriterien

- $n \geq n_{max}, \; n_{max} \in \mathbb{N}$
- $||x^{(n+1)} - x^{(n)}|| \leq \epsilon \qquad \leftrightarrow \qquad ||\delta^{(n)}|| \leq \epsilon$
- $||x^{(n+1)} - x^{(n)}|| \leq \epsilon * ||x^{(n+1)}||$
- $||\vec{f}(x^{(n+1)})|| \leq \epsilon$

Konvergenz

Das Newton-Verfahren konvergiert quadratisch für nahe genug an einer Nullstelle $\vec{x}$ liegende Startvektoren, wenn $D\vec{f}(\vec{x})$ regulär und $\vec{f}$ dreimal stetig differenzierbar ist.

Vereinfachtes Newton-Verfahren für Systeme

Gesucht: Nullstellen von $\vec{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ mit Startvektor $\vec{x}^{(0)}$ Nahe der Nullstelle.

Der Unterschied zum Quadratisch-konvergenten Newton-Verfahren liegt darin, dass nur einmal die Jacobi-Matrix für den Startvektor berechnet werden muss (rot) → weniger Aufwand aber konvergiert langsamer (linear).

für $n = 0, 1, ...$:

1. Berechne $\delta^{(n)}$ als Lösung des linearen Gleichungsystems

$$D\vec{f}(\vec{x}^{(0)})\delta^{(n)} = -\vec{f}(\vec{x}^{(n)})$$

2. Setze

$$\vec{x}^{(n+1)} = \vec{x}^{(n)} + \delta^{(n)}$$

Gedämpftes Newton-Verfahren

Die Idee ist bei jedem Iterationsschritt zu überprüfen ob es sich um eine Verbesserung handelt und falls nicht, es stattdessen mit einem gedämpften Schritt zu versuchen.

Die gewählte Dämpfungsgrösse k_{max} ist stark vom Problem abhängig. $k_{max} = 4$ ist ein vernünftiger Standard Wert.

für $n = 0, 1, ...$:

1. Berechne $\delta^{(n)}$ als Lösung des linearen Gleichungsystems

$$D\vec{f}(\vec{x}^{(n)})\delta^{(n)} = -\vec{f}(\vec{x}^{(n)})$$

2. Finde das minimale $k \in 0, 1, ..., k_{max}$ mit

$$||\vec{f}(x^{(n)}) + \frac{\delta^{(n)}}{2^k}||_2 < ||\vec{f}(x^{(n)})||_2$$

3. Setze (falls kein minimales $k \rightarrow k = 0$)

$$\vec{x}^{(n+1)} = \vec{x}^{(n)} + \frac{\delta^{(n)}}{2^k}$$

Ausgleichsrechnung / Interpolation

Zur Auswertung Datenpunkte mit Streuung durch eine Funktion annähern.

Interpolation eine Funktion die exakt durch die Messpunkte geht. Geeignet falls:
- wenig Datenpunkte
- (fast) keine Messfehler

Ausgleichsrechnung eine Funktion die summiert die kleinste Abweichung von den Messpunkten hat. Zwischen den Messpunkten oftmals stabiler als Interpolation
- typischerweise viele Datenpunkte
- fehlerbehaftet

Interpolation

- Gegeben: $n+1$ Wertepaare (x_i, y_i), $i = 0, ..., n$ mit $x_i \neq x_j \mid i \neq j$
- Gesucht: stetige Funktion $g(x)$ mit $g(x_i) = y_i \; \forall i = 0, 1, ..., n$
  (was zwischen diesen Punkten für Resultate herauskommen kann stark variieren)

Polynominterpolation

Zu den $n+1$ Stützpunkten ist ein Polynom $P_n(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n$ vom Grad n gesucht.

Weil das Polynom vom Grad n ist lässt sich zusammen mit den Stützpunkten ein lineares Gleichungsystem dazu aufstellen.

$$\begin{aligned} a_0 + a_1 x_0 + a_2 x_0^2 + ... + a_n x_0^n &= y_0 \\ a_0 + a_1 x_1 + a_2 x_1^2 + ... + a_n x_1^n &= y_1 \\ &\vdots \\ a_0 + a_1 x_n + a_2 x_n^2 + ... + a_n x_n^n &= y_n \end{aligned}$$

bzw.

$$\begin{pmatrix} 1 & x_0 & \cdots & x_0^n \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}$$

Lagrange Interpolationsformel

Durch $n+1$ Stützpunkte mit verschiedenen Stützstellen gibt es genau EIN Polynom $P_n(x)$ vom Grade $\leq n$ welches alle Stützpunkte interpoliert.

Lagrangeform für $P_n(x)$:

$$P_n(x) = \sum_{i=0}^{n} l_i(x) y_i$$

Die Lagrangepolynome vom Grad n ($l_i(x)$) sind definiert durch:

$$l_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \qquad i = 0, 1, ..., n$$

Beispiel:

t	08.00	10.00	12.00	14.00	Uhr
T	11.2	13.4	15.3	19.5	°C

Wir haben $n+1 = 4$ Stützpunkte, also ist $n = 3$ und das Interpolationspolynom hat die Form

$$P_n(x) = \sum_{i=0}^{3} l_i(x) y_i = 11.2 \cdot l_0(x) + 13.4 \cdot l_1(x) + 15.3 \cdot l_2(x) + 19.5 \cdot l_3(x).$$

Die Lagrangepolynome berechnen sich zu

$$\begin{aligned} l_0(x) &= \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} = \frac{(x-10)(x-12)(x-14)}{(-2)(-4)(-6)} = -\frac{1}{48}x^3 + \frac{3}{4}x^2 - \frac{107}{12}x + 35 \\ l_1(x) &= \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} = \frac{(x-8)(x-12)(x-14)}{(2)(-2)(-4)} = +\frac{1}{16}x^3 - \frac{17}{8}x^2 + \frac{47}{2}x - 84 \\ l_2(x) &= \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} = \frac{(x-8)(x-10)(x-14)}{(4)(2)(-2)} = -\frac{1}{16}x^3 + 2x^2 - \frac{83}{4}x + 70 \\ l_3(x) &= \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} = \frac{(x-8)(x-10)(x-12)}{(6)(4)(2)} = +\frac{1}{48}x^3 - \frac{5}{8}x^2 + \frac{37}{6}x - 20 \end{aligned}$$

Fehlerabschätzung

Rein theoretisch weil man die tatsächliche Funktion f kennen müsste.
Gegeben $y_i = f(x_i)$ und f genügend oft stetig differenzierbar:

$$|f(x) - P_n(x)| \leq \frac{|(x-x_0)(x-x_1)...(x-x_n)|}{(n+1)!} * \max_{x_0 \leq \xi \leq x_n} |f^{(n+1)}(\xi)|$$

Spline-Interpolation

Kontext

Die Annhäherung durch ein einziges Polynom ist zwischen den Messpunkten oft hoch instabil. Als Alternative kann stattdessen stückweise interpoliert werden.