## 0.1 Ausgleichsrechnung

- Datenpunkte mit gewissen Streuung durch einfache Funktion annähern
- Mehr Gleichungen als unbekannte (Mehr Datenpunkte als Parameter)

### 0.1.1 Polynominterpolation

Gesucht: $P_n(x)$ welche n+1 Stützpunkte interpoliert Jeder Stützpunkt gibt lin.Gleichung für die Bestimmung der Koeffizienten.

Grad n so wählen , dass lin.Gleichungssystem gleich viele Gleichungen wie unbekannte Koeffizienten hat.

$P_n(x_0) = a_0 + a_1x_0 + a_2x_0^2 + ... + a_nx_0^n = y_0$

$P_n(x_1) = a_0 + a_1x_1 + a_2x_1^2 + ... + a_nx_1^n = y_1$
.
.
.
$P_n(x_n) = a_0 + a_1x_n + a_2x_n^2 + ... + a_nx_n^n = y_n$

$$= \left.\begin{bmatrix} 1 & x_0 & ... & x_0^n \\ . & . & & \\ . & . & & \\ . & . & & \\ 1 & x_n & ... & n_n^n \end{bmatrix} * \begin{bmatrix} a_0 \\ . \\ . \\ . \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ . \\ . \\ . \\ y_n \end{bmatrix}\right\} Vandermonde - Matrix = Schlechtkonditioniert.Abn \geq 20numerischinstabil$$

**Lagrange Interpolationsformel:**
Lagrangeform von $P_n(x)$ :

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

LagrangePolynome $= l_i(x)$:

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

**Stückweise Interpolation:**
Interpolationspolynom erster Ordnung: n = 1
Stützpunkte so wählen: $x_{j-1} und x_{j+1}$ ,somit 2 Werte: n = 1

**Grösse des Fehlers an Stelle x wenn:**
$y_i$ Funktionswerte (genügend oft stetig differenzierbare Funktion)
$|f(x) - P_n(x)| \leq \dfrac{|(x - x_0)(x - x_1)...(x - x_n)|}{(n + 1!)}$
Max der (n+1)-ten Abletung der f(x) Intervall $[x_0, x_n]$ kennen, Fehlerabschätzung nur dann möglich.

### 0.1.2 Spline-Interpolation

**Bedingungen für $S_i$:**

1. $S_i(x_i) = y_i, S_{i+1}(x + 1) = y_{i+1}, ...Interpolation$
2. $S_i(x_{i+1}) = S_{i+1}(x_{i+1}), S_{i+1}(x_{i+2}) = S_{i+2}(x_{i+2}), ...$Stetiger Übergang

3. $S'_i(x_{i+1}) = S'_{i+1}(x_{i+1}), S'_{i+1}(x_{i+2}) = S'_{i+2}, ... KeineKnicke$
4. $S''_i(x_{i+1}) = S''_{i+1}(x_{i+1}), S''_{i+1}(x_{i+2}) = S''_{i+2}, ...$ Gleiche Krümmung
5. Mindestens den Grad 3 ... Kubische Splines

3 Intervalle, 4 Stützpunkte:
$[x_0, x_1], [x_1, x_2], [x_2, x_3]$
**Ansatz:**
$S_0 = a_0 + b_0(x - x_0) + c_0(x - x_0)^2 + d_0(x - x_0)^3, x \in [x_0, x_1]$
$S_1 = a_1 + b_1(x - x_1) + c_1(x - x_1)^2 + d_1(x - x_1)^3, x \in [x_1, x_2]$
$S_2 = a_2 + b_2(x - x_2) + c_2(x - x_2)^2 + d_2(x - x_2)^3, x \in [x_2, x_3]$

$3 * 4 = 12$ Koeffizienten $\Longrightarrow$ 12 Bedingungen
**1.Interpolation der Stützpunkte:**
1.$S_0(x_0) = y_0$
2.$S_1(x_1) = y_1$
3.$S_2(x_2) = y_2$
4.$S_3(x_3) = y_3$

**2. Stetiger Übergang an Stellen $x_1$ und $x_2$:**
5.$S_0(x_1) = S_1(x_1)$
6.$S_1(x_2) = S_2(x_2)$

**3. Erste Ableitung an Übergangstellen übereinstimmen:**
7.$S'_0(x_1) = S'_1(x_1)$
8.$S'_1(x_2) = S'_2(x_2)$

**4. Zweite Ableitung an Übergangstellen übereinstimmen:**
9.$S''_0(x_1) = S''_1(x_1)$
10.$S''_1(x_2) = S''_2(x_2)$

$= 10 Bedingungen$

Die weiteren 2 Bedingungen können frei gewählt"werden.
Diese beziehen sich häufig auf Randstellen $x_0$ und $x_3$.
**Beispiele:**
Natürliche kubische Splinefunktion:
Mit moeglichen Wendepunkt im Anfangs und Endpunkt.

$$\left.\begin{aligned} S'''_0(x_0) = 0 \\ S'''_2(x_3) = 0 \end{aligned}\right\}$$

Periodische kubische Splinefunktion:
Wenn man Periode $p = x_3 - x - 0$ hat und damit $y_0$ bzw. $S_0(x_0) = S_2(x_3)$ gilt

$$\left.\begin{aligned} S'_0(x_0) = S'_2(x_3) \\ S''_0(x_0) = S''_2(x_3) \end{aligned}\right\}$$

mit not-a-knot Bedingung kubische Splinefunktion:
s.d Auch dritte Ableitung in $x - 1, x - 2$ noch stetig ist. ($x_1, x_2$ keine echten Knoten)

$$\left.\begin{aligned} S'''_0(x_1) = S'''x_1(x_1) \\ S'''_1(x_2) = S'''_2(x_2) \end{aligned}\right\}$$

**Algorithmus: natürliche kubische Splinefunktion**
$S_i = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$
Koeffizienten $a_i, b_i, c_i, d_i$ berechnen:

1. $a_i = y_i$
2. $h_i = x_{i+1} - x_i$
3. $c_0 = 0, c_n = 0$ !!! (für periodisch($s_1 = s_0$),für not a knot($d_0 = d_1 d_{n-2} = d_{n-1}$)
4. Berechnung der Koeffizienten $c_1, c_2, c_3, ..., c_{n-1}$
   - i = 1:
     - $2(h_0 + h_1)\; c_1 + h_1\; c_2 = 3\frac{y_2 - y_1}{h_1} - 3\frac{y_1 - y_0}{h_0}$
   - falls $n \geq 4$ dann für i = 2,...,n-2
     - $h_{i-1}\; c_{i-1} + 2(h_{i-1} + h_i)\; c_i + h_i\; c_{i+1} = 3\frac{y_{i+1} - y_i}{h_i} - 3\frac{y_i - y_{i-1}}{h_{i-1}}$
   - i = n - 1:
     - $2(h_{n-2} + h_i)\; c_{n-2} + 2(h_{n-2} + h_{n-1})\; c_{n-1} = 3\frac{y_n - y_{n-1}}{h_{n-1}} - 3\frac{y_{n-1} - y_{n-2}}{h_{n-2}}$
5. $b_i = \frac{y_{i+1} - y_i}{h_i} - \frac{h_i}{3}(c_{i+1} + 2c_i)$
6. $d_i = \frac{1}{3h_i}(c_{i+1} - c_i)$

Ergibt das Gleichungssystem A c = z

- A ist immer invertierbar
- Numerische Lsg.durch Gauss-Algo
- System ist gut konditioniert
- Pivotsuche nicht erforderlich

*i = 0,...,n-1*

$$A = \begin{bmatrix} \\ \\ \\ \end{bmatrix} c = \begin{bmatrix} \\ \\ \end{bmatrix} = z = \begin{bmatrix} \\ \\ \end{bmatrix}$$

### 0.1.3 Lineare Ausgleichsprobleme

Ausgleichsgerade: $f(x) = ax + b$: gesucht: $F = af_1 + bf_2 \mid a, b \in \mathbb{R}$ **Basisfunktion:** $f_1(x) = x$ und $f_2(x) = 1$

**Form Fehlerfunktional:**

$E(f)(a,b) := \sum_{i=1}^{n}(y_i - f(x_i))^2 = \sum_{i=1}^{n}(y_i - (ax_i + b))^2)$

minimal = patiellen Ableitung nach a,b müssen verschwinden

$$\begin{bmatrix} \sum_{i=1}^{n}(x_i)^2 & \sum_{i=1}^{n}(x_i) \\ \\ \sum_{i=1}^{n}(x_i) & n \end{bmatrix} * \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n}(x_iy_i) \\ \\ \sum_{i=1}^{n}(y_i) \end{bmatrix}$$

Nach a,b auflösen

**Allgemeine Definition:**
Gegeben:
n-Wertepaare: $(x_i, y_i), i = 1, ..., n$
Basisfunktionen: $(f_1, ..., f_m)[a, b]$
Ansatzfunktionen: $f := \lambda_1 f_1 + ... + \lambda_m f_m$

Lineares Ausgleichproblem mit Fehlerfunktional:
$E(f) = \|y - f(x)\|_2^2 = \sum_{i=1}^{n}(y_i - \sum_{i=1}^{m} \lambda_j * f_j * (x_i))^2 = \|y - A\lambda\|_2^2$
Fehlergleichungssystem: $A\lambda = y$

$$A=\begin{bmatrix} f_1(x_1) & f_2(x)(x_1) & ... & f_m(x_1) \\ f_1(x_2) & f_2(x_2)(x_1) & ... & f_m(x_2) \\ . & & & \\ . & & & \\ . & & & \\ f_1(x_n) & f_2(x_n)(x_n) & ... & f_m(x_n) \\ & & & \end{bmatrix} * y = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix} = \lambda \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ . \\ . \\ . \\ \lambda_n \end{bmatrix}$$

Normalgleichungen: $\dfrac{\delta E(f)(\lambda_1, ..., \lambda_m)}{\delta \lambda_j}$

Normalgleichungssystem:$A^T A\lambda = A^T y$

Lösung:$\lambda$ beinhaltet gesuchte Prameter des linearen Ausgleichproblems
Verfahren am Besten mit QR-Zerlegung

### 0.1.4 Nichtlineare Ausgleichprobleme

**Allgemeines Ausgleichproblem:**

Fehlerfunktional:
$f_p$ = Ansatzfunktion

$$E(f)(a, b) := \sum_{i=1}^{n}(y_i - f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_i))^2 = \begin{bmatrix} y_1 - f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_1) \\ y_2 - f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_2) \\ . \\ . \\ . \\ y_n - f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_n) \\ \\ \end{bmatrix} \|_2^2 =$$

$\|y - f(\lambda)\|_2^2$

$$f(\lambda) := f(\lambda_1, \lambda_2, ..., \lambda_m) := \begin{bmatrix} f_p(\lambda_1, \lambda_2, ..., \lambda_m) \\ f_p(\lambda_1, \lambda_2, ..., \lambda_m) \\ . \\ . \\ . \\ f_p(\lambda_1, \lambda_2, ..., \lambda_m) \end{bmatrix} := \begin{bmatrix} f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_1) \\ f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_2) \\ . \\ . \\ . \\ f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_n) \end{bmatrix},$$

$$y=\begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}, \lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ . \\ . \\ . \\ \lambda_n \end{bmatrix}$$

Falls Ansatzfunktionen linear in den Parametern $\implies$
Spezialfall des Ausgleichsproblems: $f(\lambda) = A\lambda \implies$ Allg.Ausgleichsproblem = Minimums einer Funktion

**Schritte:**

1. Normalgleichungen aufstellen (part.Abl.von f)
2. $\lambda_i = 0$
3. Nichtlineare Gleichungssystem lösen

### 0.1.5 Das Gauss-Newton Verfahren

Quadratmittelproblem: Problem einen Vektor x zu finden für den E(x) minimal wird.
Nichtlineare Ausgleichsprobleme: $g(\lambda) := y - f(\lambda)$
Fehlerfunktional: $E(\lambda) := \|g(\lambda)\|_2^2 = \|y - f(\lambda)\|_2^2$
Gauss-Newton-Verfahren: Lin.Ausgl.RG + Newton-Verfahren = $g(\lambda) = g(\lambda^{(0)}) + Dg(\lambda^0) * (\lambda - \lambda^0)$
= Verallg.Tangentgleichung!!

$$Dg(\lambda^0) = \begin{bmatrix} \frac{\delta g_1}{\lambda_1}(\delta\lambda^0) & \frac{\delta g_1}{\lambda_2}(\delta\lambda^0) & \dots & \frac{\delta g_1}{\lambda_m}(\delta\lambda^{(0)}) \\ \frac{\delta g_2}{\lambda_1}(\delta\lambda^0) & \frac{\delta g_2}{\lambda_2}(\delta\lambda^0) & \dots & \frac{\delta g_2}{\lambda_m}(\lambda^{(0)}) \\ . & & & \\ . & & & \\ . & & & \\ \frac{\delta g_n}{\lambda_1}(\delta\lambda^0) & \frac{\delta g_n}{\lambda_2}(\delta\lambda^0) & \dots & \frac{\delta g_n}{\lambda_m}(\delta\lambda^{(0)}) \end{bmatrix}$$

Minimum Fehlerfunktionals: $E(\lambda) = \underbrace{\|g(\lambda^{(0)})\|}_{\text{y}} + \underbrace{Dg(\lambda^0)}_{\text{-A}} * \underbrace{\lambda - \lambda^0)}_{\delta}$

Normagleichungssystem: $A^T A\delta = A^T y = Dg(\lambda^0)^T Dg(\lambda^0)\delta = Dg(\lambda^0)^T * g(\lambda^0) \implies$
QR-Zerlegung: $Dg(\lambda^0)$=QR$\geq R\delta = -Q^T g(\lambda^0)$
$\implies$ Lösung: $\Lambda = (\lambda^0) + \delta$

**Schritte:**

1. Berechne Funktion
   - $g(\lambda) := y - f(\lambda)$
   - $Dg(\lambda)$
2. k = 0,1...:
   $\delta^k$ als Lösung: $min\|g(\lambda^k) + Dg(\lambda^k) * \delta^k\|_2^2$
   $\implies Dg(\lambda^k)^T * Dg(\lambda^k)\delta^k = Dg(\lambda^k)^T * g(\lambda^k)$
   nach nach $\delta^k$ auflösen durch QR-Zerlegung.
   - $Dg(\lambda^k) = Q^k R^k$
   - $R^k\delta^k = -(Q^k)^T g(\lambda^k)$
3. Setze: $\lambda^{k+1} = \lambda^k + \delta^k$

Korrektur $\delta^k$ nur akzeptiert wenn Fehlerfunktional zur Abnahme führt: **Fehlerfunktional** $E(\lambda^{k+1}) = \|g(\lambda^{k+1})\|_2^2 < \|g(\lambda^k)\|_2^2 = E(\lambda^k)$

**Gedämpftes Gauss-Newton-Verfahren**

1. Berechne Funktion:
   - $g(\lambda) := y - f(\lambda)$
   - $Dg(\lambda)$
2. k = 0,1...:
   $\delta^k$ als Lösung: $min\|g(\lambda^k) + Dg(\lambda^k) * \delta^k\|_2^2$
   $\implies Dg(\lambda^k)^T * Dg(\lambda^k)\delta^k = Dg(\lambda^k)^T * g(\lambda^k)$
   nach nach $\delta^k$ auflösen durch QR-Zerlegung.
   - $Dg(\lambda^k) = Q^k R^k$
   - $R^k \delta^k = -(Q^k)^T g(\lambda^k)$
3. Finde das minimale p $\in$ 0,1,...,$p_{max}$
   $\underbrace{\|\lambda^{(k)} + \frac{\delta^k}{2^p}\|_2^2}_{\lambda^{(k+1)}} < \|g(\lambda^{(k)})\|_2^2$
4. Falls kein min. p gefunden:
   p=0 und weiterfahren
5. Setze: $\lambda^{(k)} + \frac{\delta^{(k)}}{2^p}$

Abbruchkriterium: $\|\frac{\delta^{(k)}}{2^p}\| < TOL$
Keine Grantie, dass die Näherung max. Abstand von TOL zum ges. Min. ist.