### Zahldarstellung, Fließkommazahlen, Rundung

* Überlauf: Zahl hat mehr Stellen als gespeicherte Bit 🡪 Überlauf fällt weg (z. B. bei 8Bit: modulu 256)

***Fließkommazahlen:***

* Umwandlung IEEE:
  + Vorzeichen: 0 = positiv, 1 = negativ
  + Charakterisitk c: *c = 127 + e* 🡪 e ist Exponent für Mantisse, bei Überlauf c = 255
  + Mantisse m: Zahl als 1.x 🡪 Nachkommastellen x sind Mantisse
  + 32bit: |c| = 8bit, |m| = 23bit 64bit: |c| = 11bit, |m| = 52bit
* Maschinenzahl: *c = e m = Zahl als 0.x*
* Rundung IEEE: wenn 1. Überlaufbit 1 🡪 letztes Bit = 1, sonst 0
* Rechnen mit Fließkommazahlen:
  + Addition: Exponenten angleichen > addieren > renormalisieren > runden
  + Multiplikation: Exponent bzw. Charakteristik wird geändert

***Abschätzungen – Rundung:***

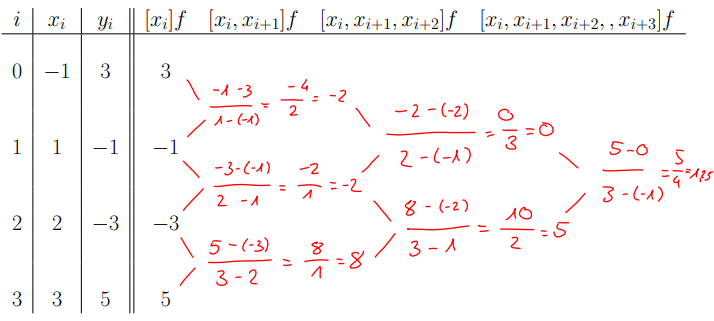
* relativer Fehler: 🡪 d = absoluter Fehler, x = Wert zur Relation

### Kondition

* absolute Kondition: 🡪 Ableitung
* relative Kondition:
  + bei mehreren mit part. Ableitungen:
  + lineare Analyse: δf(k) ≤ ≤ δx 🡪 relative Kondition \* δk ≤ relativer Fehler
    - Abschätzung: δk angeben 🡪 mithilfe linearer Analyse ausrechnen
  + relative Fehlerabschätzung mit Kondition: (wenn angegeben, siehe oben)
* partielle Ableitung: Ableitung auf eine Variable
  + Schreibweise: 🡪 partielle Ableitung auf x
  + Gradient: 🡪 Spaltenvektor
* Instabilität: Auslöschungseffekt bei Subtraktion annähernd gleicher Zahlen (S. 27)

### Interpolation

* Lagrange Polynome: 🡪 k ≠ j, j durchlaufen 0 – n
  + Interpolationspolynom in Lagrange-Form:

* Newton Polynome: 🡪 0 ≤ j ≤ n, für j = 0 🡪 ω = 1
  + Interpolationspolynom in Newton-Form:

...

* Aitken-Neville-Algorithmus:

### Lineare Gleichungssysteme

* LR-Zerlegung: Matrix A wird in L und R zerlegt (L \* R = A)
  + A mit Gauß auf Dreiecksmatrix = R 🡪 benötigte Schritte in L (z.B. II – 2 \* I 🡪 L[2,1] = 2)
  + L \* z = b, R \* x = z 🡪 x gesucht
* Norm & Kondition:
  + Norm einer Matrix:
    - 1-Norm: Maximum 1-Normen der Spalten von A
      * Summe jeder Spalte ausrechnen (**Beträge**) 🡪 danach Maximum aller Spaltensummen
    - ∞-Norm: Maximum Summen der Zeilen von A
      * Summe jeder Zeile ausrechnen (**Beträge**) 🡪 danach Maximum aller Zeilensummen
* Kondition einer Matrix:
  + absolute Kondition:
  + relative Kondition: 🡪 nur möglich, wenn Matrix invertierbar

### Ausgleichsrechnung / Regressionsrechnung

* Aufstellen einer Gleichung: Form
  + abhängige Variablen: 🡪 Ergebnis b (Vektor)
  + Konstanten: 🡪 Matrix A ()
  + Parameter: 🡪 Parameter x () 🡪 #Zeilen = #Spalten von A
  + Unbekannte: 🡪 Konstante γ, in Matrix A = 1er-Spalte

oder (mit Konstante)

* + - wenn rank(A) ≠ |x| 🡪 Parameter x nicht eindeutig
    - rank(A) ≠ rank([Ab]) 🡪 Gleichungssystem nicht lösbar
* kleinste-Quadrate-Lösung: 🡪 ai \* x 🡪 Formel y = ax + b ...
  + bilde Summenformel 🡪 nicht ausmultiplizieren
  + partielle Ableitung auf alle jeweiligen Variablen 🡪 setze partielle Ableitungen = 0
  + einsetzen in Gauss-Verfahren 🡪 Ergebnis ist x

### Splines

* kubischer Spline:
  + Funktion:
  + Bedingungen:
  + Randbedingungen:
    - natürliche Randbedingung: S. 59

### Nichtlineare Gleichungen

* Bisektion:
  + Halbierung eines Intervalls mit VZ-Übergang 🡪 Annäherung an Nullstelle
* Newton-Verfahren: S.60
  + Sekanten-Verfahren:

### MATLAB

* Matrix:
  + Aufruf: name(zeile, spalte)
  + Deklarierung: name = zeros(zeilen,spalten) name=[werte;werte] (Zeilenwerte)
  + transponieren: name = [...]‘ bzw. name‘
  + Zugriff ab 1!!
* Schleife: for i = anfang:[schrittweite:]:ende end
* if-Abfrage: if i == x
* Funktion: function x = name(var1,var2) end
* Variablen: name = wert
* Gleichung lösen: x = R \ z (Matrix / Vektor)
* Operatoren:
  + Ungleich: ~=
  + Skalarmult.: .\*
* Maximum: max(op1, op2)
* Zeilen/Spalten tauschen: matrix([z1 z2], :) = matrix([z2 z1], :)