Höhere Mathematik

Jil Zerndt, Lucien Perret
January 2025

Rechnerarithmetik

Maschinenzahlen Eine maschinendarstellbare Zahl zur Basis B ist ein Element der Menge:

$$M = \{x \in \mathbb{R} \mid x = \pm 0.m_1 m_2 m_3 \dots m_n \cdot B^{\pm e_1 e_2 \dots e_l}\} \cup \{0\}$$

- $m_1 \neq 0$ (Normalisierungsbedingung)
- $m_i, e_i \in \{0, 1, \dots, B-1\}$ für $i \neq 0$
- $B \in \mathbb{N}, B > 1$ (Basis)

Zahlenwert $\hat{\omega}$ $= \sum_{i=1}^{n} m_i B^{\hat{e}-i}$, mit $\hat{e} = \sum_{i=1}^{l} e_i B^{l-i}$

B = Basis, n = Mantissenlänge, l = Exponentenlänge
m_i = Mantissenstelle, e_i = Exponentenstelle

Werteberechnung einer Maschinenzahl

1. Normalisierung überprüfen: $m_1 \neq 0$ (für $x \neq 0$)
   - Sonst: Mantisse verschieben und Exponent anpassen
2. Exponent berechnen: $\hat{e} = \sum_{i=1}^{l} e_i B^{l-i}$
   - Von links nach rechts: Stelle · Basis hochgestellt zur Position
3. Wert berechnen: $\hat{\omega} = \sum_{i=1}^{n} m_i B^{\hat{e}-i}$
   - Mantissenstellen · Basis hochgestellt zu (Exponent - Position)
4. Vorzeichen berücksichtigen

Werteberechnung Berechnung einer vierstelligen Zahl zur Basis 4:

$$\underbrace{0.3211}_{n=4} \cdot \underbrace{4^{12}}_{l=2}$$

Exponent: $\hat{e} = 1 \cdot 4^1 + 2 \cdot 4^0 = 6$
Wert: $\hat{\omega} = 3 \cdot 4^5 + 2 \cdot 4^4 + 1 \cdot 4^3 + 1 \cdot 4^2 = 3664$

IEEE-754 Standard definiert zwei wichtige Gleitpunktformate:

Single Precision (32 Bit)	**Double Precision** (64 Bit)
Vorzeichen(V): 1 Bit	Vorzeichen(V): 1 Bit
Exponent(E): 8 Bit (Bias 127)	Exponent(E): 11 Bit (Bias 1023)
Mantisse(M):	Mantisse(M):
23 Bit + 1 hidden bit	52 Bit + 1 hidden bit

IEEE-754 Details

- **Overflow**: Zahlen $\notin [-x_{max}, x_{max}]$ führen zum Abbruch mit `inf`
- **Underflow**: Zahlen in $[-x_{min}, x_{min}]$ werden zu 0 gerundet

Darstellungsbereich Für jedes Gleitpunktsystem existieren:

- Grösste darstellbare Zahl: $x_{\max} = (1 - B^{-n}) \cdot B^{e_{\max}}$
- Kleinste darstellbare positive Zahl: $x_{\min} = B^{e_{\min}-1}$

Maschinengenauigkeit analysieren

1. Anzahl Maschinenzahlen bestimmen: $2 \cdot (B-1) \cdot B^{n-1} \cdot (B^l - 1)$
2. Darstellungsbereich bestimmen: x_{max}, x_{min}
3. Maschinengenauigkeit berechnen: $eps = \frac{B}{2} B^{-n}$

Maschinenzahlen analysieren Gegeben: 15-stellige Gleitpunktzahlen mit 5-stelligem Exponenten im Dualsystem.

1. Basis $B = 2$, $n = 15$, $l = 5$
2. Anzahl verschiedener Zahlen:
   - Pro Stelle: $B - 1 = 1$ mögliche Ziffern
   - Mantisse: $(B-1)B^{n-1} = 2^{14}$ Kombinationen
   - Exponent: $B^l = 2^5 = 32$ Kombinationen
   - Mit Vorzeichen: $2 \cdot 2^{14} \cdot 31 = 1\,015\,808$ Zahlen
3. Maschinengenauigkeit: $eps = \frac{2}{2} 2^{-15} = 2^{-15} \approx 3.052 \cdot 10^{-5}$

→ kleineres eps bedeutet höhere Genauigkeit

Approximations- und Rundungsfehler

Fehlerarten Sei $\tilde{x}$ eine Näherung des exakten Wertes x:

Absoluter Fehler: $|\tilde{x} - x|$

Relativer Fehler: $\left|\frac{\tilde{x}-x}{x}\right|$ bzw. $\frac{|\tilde{x}-x|}{|x|}$ für $x \neq 0$

Maschinengenauigkeit eps ist die kleinste positive Zahl, für die gilt:

Allgemein: $\text{eps} := \frac{B}{2} \cdot B^{-n}$ **Dezimal:** $\text{eps}_{10} := 5 \cdot 10^{-n}$

Sie begrenzt den maximalen relativen Rundungsfehler: $\left|\frac{rd(x)-x}{x}\right| \leq \text{eps}$

Rundungseigenschaften Für alle $x \in \mathbb{R}$ mit $|x| \geq x_{\min}$ gilt:

Absoluter Fehler: $|rd(x) - x| \leq \frac{B}{2} \cdot B^{e-n-1}$

Relativer Fehler: $\left|\frac{rd(x)-x}{x}\right| \leq \text{eps}$

Fehlerfortpflanzung

Konditionierung Die Konditionszahl K beschreibt die relative Fehlervergrösserung bei Funktionsauswertungen:

$$K := \frac{|f'(x)| \cdot |x|}{|f(x)|}$$

- $K \leq 1$: gut konditioniert
- $K > 1$: schlecht konditioniert
- $K \gg 1$: sehr schlecht konditioniert

Fehlerfortpflanzung Für f (differenzierbar) gilt näherungsweise:

Absoluter Fehler: $|f(\tilde{x}) - f(x)| \approx |f'(x)| \cdot |\tilde{x} - x|$

Relativer Fehler: $\frac{|f(\tilde{x}) - f(x)|}{|f(x)|} \approx K \cdot \frac{|\tilde{x} - x|}{|x|}$

Analyse der Fehlerfortpflanzung einer Funktion

1. Berechnen Sie $f'(x)$ und die Konditionszahl K
2. Schätzen Sie den absoluten und den relativen Fehler ab
3. Beurteilen Sie die Konditionierung anhand von K

Konditionierung berechnen Für $f(x) = \sqrt{1+x^2}$ und $x_0 = 10^{-8}$:

1. $f'(x) = \frac{x}{\sqrt{1+x^2}}$, $K = \frac{|x \cdot x|}{|\sqrt{1+x^2} \cdot (1+x^2)|} = \frac{x^2}{(1+x^2)^{3/2}}$
2. Für $x_0 = 10^{-8}$:
   - $K(10^{-8}) \approx 10^{-16}$ (gut konditioniert)
   - Relativer Fehler wird um Faktor 10^{-16} verkleinert

Fehleranalyse Beispiel: Fehleranalyse von $f(x) = \sin(x)$

1. $f'(x) = \cos(x)$, $K = \frac{|x \cos(x)|}{|\sin(x)|}$
2. Konditionierung:
   - Für $x \to 0$: $K \to 1$ (gut konditioniert)
   - Für $x \to \pi$: $K \to \infty$ (schlecht konditioniert)
   - Für $x = 0$: $\lim_{x \to 0} K = 1$ (gut konditioniert)
3. Der absolute Fehler wird nicht vergrössert, da $|\cos(x)| \leq 1$

Praktische Fehlerquellen der Numerik

Kritische Operationen häufigste Fehlerquellen:

- Auslöschung bei Subtraktion ähnlich großer Zahlen
- Überlauf (overflow) bei zu großen Zahlen
- Unterlauf (underflow) bei zu kleinen Zahlen
- Verlust signifikanter Stellen durch Rundung

Vermeidung von Auslöschung

1. Identifizieren Sie Subtraktionen ähnlich großer Zahlen
2. Suchen Sie nach algebraischen Umformungen
3. Prüfen Sie alternative Berechnungswege
4. Verwenden Sie Taylorentwicklungen für kleine Werte

Beispiele für bessere Formeln:

- $\sqrt{1+x^2} - 1 \to \frac{x^2}{\sqrt{1+x^2}+1}$
- $1 - \cos(x) \to 2\sin^2(x/2)$
- $\ln(1+x) \to x - \frac{x^2}{2}$ für kleine x

Numerische Lösung von Nullstellenproblemen

Nullstellensatz von Bolzano Sei $f : [a, b] \to \mathbb{R}$ stetig. Falls

$$f(a) \cdot f(b) < 0$$

dann existiert mindestens eine Nullstelle $\xi \in (a, b)$.

Nullstellenproblem systematisch lösen

1. Existenz prüfen:
   - Intervall $[a, b]$ identifizieren
   - Vorzeichenwechsel prüfen: $f(a) \cdot f(b) < 0$
   - Stetigkeit von f sicherstellen
2. Verfahren auswählen:
   - Fixpunktiteration: einfache Umformung $x = F(x)$ möglich
   - Newton: $f'(x)$ leicht berechenbar
   - Sekantenverfahren: $f'(x)$ schwer berechenbar
3. Konvergenz sicherstellen:
   - Fixpunktiteration: $|F'(x)| < 1$ im relevanten Bereich
   - Newton: $|\frac{f(x)f''(x)}{[f'(x)]^2}| < 1$ im relevanten Bereich
   - Geeigneten Startwert wählen:
     - Fixpunkt: x_0 nahe Fixpunkt
     - Newton: $f'(x_0) \neq 0$
     - Sekanten: Zwei Startwerte x_0 und x_1: $f(x_0) \neq f(x_1)$
4. Abbruchkriterien festlegen:
   - Funktionswert: $|f(x_n)| < \epsilon_1$
   - Iterationsschritte: $|x_{n+1} - x_n| < \epsilon_2$
   - Maximale Iterationszahl

Verfahrensauswahl Finden Sie die positive Nullstelle von $f(x) = x^3 - 2x - 5$

Vorgehen:

1. Existenz:
   - $f(2) = -1 < 0$ und $f(3) = 16 > 0$
   - $\Rightarrow$ Nullstelle in $[2, 3]$
2. Verfahrenswahl:
   - $f'(x) = 3x^2 - 2$ leicht berechenbar
   - $\Rightarrow$ Newton-Verfahren geeignet
3. Konvergenzcheck:
   - $f'(x) > 0$ für $x > 0.82$
   - $f''(x) = 6x$ monoton
   - $\Rightarrow$ Newton-Verfahren konvergiert

NSP: Nullstellenproblem, NS: Nullstelle

Typische Prüfungsaufgaben lösen

1. Theorieaufgaben:
   - Konvergenzordnungen vergleichen
   - Konvergenzbeweise durchführen
   - Fehlerabschätzungen herleiten
2. Praktische Aufgaben:
   - Existenz von Nullstellen nachweisen
   - Geeignetes Verfahren auswählen
   - 2-3 Iterationsschritte durchführen
   - Konvergenzgeschwindigkeit vergleichen
3. Implementierungsaufgaben:
   - Verfahren in Python implementieren
   - Abbruchkriterien einbauen
   - Konvergenzverhalten visualisieren

Implementation von Nullstellenverfahren

1. Grundstruktur:
   - Funktion $f(x)$ definieren
   - Ableitung $f'(x)$ falls nötig
   - Abbruchkriterien festlegen
   - Iterations-Schleife implementieren
2. Abbruchkriterien:

```python
def newton(f, df, x0, eps=1e-6, maxiter=100):
    x = x0
    for i in range(maxiter):
        fx = f(x)
        if abs(fx) < eps:  # Funktionswert klein genug
            return x
        dfx = df(x)
        if abs(dfx) < eps:  # Division durch Null vermeiden
            raise ValueError("Ableitung nahe Null")
        x_new = x - fx/dfx
        if abs(x_new - x) < eps:  # Aenderung klein genug
            return x_new
        x = x_new
    raise RuntimeError("Keine Konvergenz")
```

3. Fehlerbehandlung beachten:
   - Division durch Null bei $f'(x) = 0$
   - Keine Konvergenz
   - Über-/Unterlauf bei großen Zahlen

Fixpunktiteration

Fixpunktgleichung ist eine Gleichung der Form: $F(x) = x$
Die Lösungen $\bar{x}$, für die $F(\bar{x}) = \bar{x}$ erfüllt ist, heissen Fixpunkte.

Grundprinzip der Fixpunktiteration sei $F : [a, b] \to \mathbb{R}$ mit $x_0 \in [a, b]$

$$\text{Die rekursive Folge } x_{n+1} \equiv F(x_n), \quad n = 0, 1, 2, \ldots$$

heisst Fixpunktiteration von F zum Startwert x_0.

Fixpunktiteration anwenden

1. Umformung vorbereiten:
   - $f(x) = 0$ in $x = F(x)$ umformen
   - Verschiedene Umformungen testen
   - Form mit kleinstem $|F'(x)|$ wählen
2. Konvergenznachweis:
   - Intervall $[a, b]$ bestimmen mit $F([a, b]) \subseteq [a, b]$
   - $\alpha = \max_{x\in[a,b]} |F'(x)|$ berechnen
   - Prüfen ob $\alpha < 1$
3. Fehlerabschätzung:
   - A-priori: $|x_n - \bar{x}| \leq \frac{\alpha^n}{1-\alpha}|x_1 - x_0|$
   - A-posteriori: $|x_n - \bar{x}| \leq \frac{\alpha}{1-\alpha}|x_n - x_{n-1}|$
4. Iterationszahl bestimmen:

$$n \geq \frac{\ln(\frac{\text{tol}(1-\alpha)}{|x_1-x_0|})}{\ln \alpha}$$

Fixpunktiteration Bestimmen Sie $\sqrt{3}$ mittels Fixpunktiteration.

Lösung:

1. Umformung: $x^2 = 3 \Rightarrow x = \frac{x^2+3}{2x} =: F(x)$
2. Konvergenznachweis für $[1, 2]$:
   - $F'(x) = \frac{x^2-3}{2x^2}$
   - $|F'(x)| \leq \alpha = 0.25 < 1$ für $x \in [1, 2]$
   - $F([1, 2]) \subseteq [1, 2]$
3. Für Genauigkeit 10^{-6}:
   - $|x_1 - x_0| = |1.5 - 2| = 0.5$
   - $n \geq \frac{\ln(10^{-6}\cdot 0.75/0.5)}{\ln 0.25} \approx 12$

Konvergenzverhalten

Sei $F : [a, b] \to \mathbb{R}$ mit stetiger Ableitung F' und $\bar{x} \in [a, b]$ ein Fixpunkt von F. Dann gilt für die Fixpunktiteration $x_{n+1} = F(x_n)$:

Anziehender Fixpunkt: $|F'(\bar{x})| < 1$

x_n konvergiert gegen $\bar{x}$, falls x_0 nahe genug bei $\bar{x}$

Abstossender Fixpunkt: $|F'(\bar{x})| > 1$

x_n konvergiert für keinen Startwert $x_0 \neq \bar{x}$

Banachscher Fixpunktsatz $F : [a, b] \to [a, b]$ und $\exists$ Konstante α:

- $0 < \alpha < 1$ (Lipschitz-Konstante)
- $|F(x) - F(y)| \leq \alpha|x - y|$ für alle $x, y \in [a, b]$

Dann gilt:

- F hat genau einen Fixpunkt $\bar{x}$ in $[a, b]$
- Die Fixpunktiteration konvergiert gegen $\bar{x}$ für alle $x_0 \in [a, b]$

Fehlerabschätzungen:

a-priori: $|x_n - \bar{x}| \leq \frac{\alpha^n}{1-\alpha} \cdot |x_1 - x_0|$

a-posteriori: $|x_n - \bar{x}| \leq \frac{\alpha}{1-\alpha} \cdot |x_n - x_{n-1}|$

Konvergenznachweis für Fixpunktiteration

1. Bringe die Gleichung in Fixpunktform: $f(x) = 0 \Rightarrow x = F(x)$
2. Prüfe, ob F das Intervall $[a, b]$ in sich abbildet:
   - Wähle geeignetes Intervall ($[a, b]$ $F(a) \geq a$ und $F(b) \leq b$)
3. Bestimme die Lipschitz-Konstante α: $\to$ Berechne $F'(x)$
   - Finde $\alpha = \max_{x\in[a,b]} |F'(x)|$ und prüfe $\alpha < 1$
4. Berechnen Sie die nötigen Iterationen für Genauigkeit tol: $n \geq \frac{\ln(\frac{tol\cdot(1-\alpha)}{|x_1-x_0|})}{\ln \alpha}$

Fixpunktiteration Nullstellen von $f(x) = e^x - x - 2$
Umformung in Fixpunktform: $x = \ln(x + 2)$, also $F(x) = \ln(x + 2)$

1. $F'(x) = \frac{1}{x+2}$ monoton fallend
2. Für $I = [1, 2]$: $F(1) = 1.099 > 1$, $F(2) = 1.386 < 2$
3. $\alpha = \max_{x\in[1,2]} |\frac{1}{x+2}| = \frac{1}{3} < 1$
4. Konvergenz für Startwerte in $[1, 2]$ gesichert
5. Für Genauigkeit 10^{-6} benötigt: $n \geq 12$ Iterationen

Newton-Verfahren

Grundprinzip Newton-Verfahren
Approximation der NS durch sukzessive Tangentenberechnung:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Konvergiert, wenn für alle x im relevanten Intervall gilt:

$$\left|\frac{f(x) \cdot f''(x)}{[f'(x)]^2}\right| < 1$$

Newton-Verfahren anwenden

1. Funktion $f(x)$ und Ableitung $f'(x)$ aufstellen
2. Geeigneten Startwert x_0 nahe der Nullstelle wählen
   - Prüfen, ob $f'(x_0) \neq 0$
3. Iterieren bis zur gewünschten Genauigkeit: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$
4. Abbruchkriterien prüfen:
   - Funktionswert: $|f(x_n)| < \epsilon_1$
   - Änderung aufeinanderfolgenden Werte: $|x_{n+1} - x_n| < \epsilon_2$
   - Maximale Iterationszahl nicht überschritten

Newton-Verfahren Nullstellen von $f(x) = x^2 - 2$
Ableitung: $f'(x) = 2x$, Startwert $x_0 = 1$

1. $x_1 = 1 - \frac{1^2-2}{2\cdot 1} = 1.5$
2. $x_2 = 1.5 - \frac{1.5^2-2}{2\cdot 1.5} = 1.4167$
3. $x_3 = 1.4167 - \frac{1.4167^2-2}{2\cdot 1.4167} = 1.4142$

$\to$ Konvergenz gegen $\sqrt{2}$ nach wenigen Schritten

Newton-Verfahren anwenden

1. Vorbereitung:
   - $f'(x)$ bestimmen
   - Startwert x_0 nahe der Nullstelle wählen
   - $f'(x_0) \neq 0$ prüfen
2. Iteration durchführen:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

3. Konvergenz prüfen:
   - $|\frac{f(x)f''(x)}{[f'(x)]^2}| < 1$ im relevanten Bereich
   - Quadratische Konvergenz erwarten
4. Fehlerabschätzung:
   - $|x_n - \bar{x}| < \epsilon$ falls
   - $f(x_n - \epsilon) \cdot f(x_n + \epsilon) < 0$

Newton vs Sekanten Bestimmen Sie $\sqrt{2}$ mit beiden Verfahren.

Newton-Verfahren:

$f(x) = x^2 - 2$

- $f'(x) = 2x$
- $x_0 = 1.5$
- $x_1 = 1.5 - \frac{1.5^2-2}{2\cdot 1.5} = 1.4167$
- $x_2 = 1.4167 - \frac{1.4167^2-2}{2\cdot 1.4167} = 1.4142$

Sekantenverfahren:

- $x_0 = 1$, $x_1 = 2$
- $x_2 = x_1 - \frac{x_1-x_0}{f(x_1)-f(x_0)} f(x_1) = 1.5$
- $x_3 = 1.5 - \frac{1.5-2}{1.5^2-2} 1.5 = 1.4545$
- $x_4 = 1.4545 - \frac{1.4545-1.5}{1.4545^2-2} 1.4545 = 1.4143$

Vergleich:

- Newton: Schnellere Konvergenz (quadratisch)
- Sekanten: Keine Ableitungsberechnung nötig
- Beide erreichen 10^{-4} Genauigkeit in 4-5 Schritten

Vereinfachtes Newton-Verfahren

Alternative Variante mit konstanter Ableitung: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_0)}$

Konvergiert langsamer, aber benötigt weniger Rechenaufwand.

Sekantenverfahren

Alternative zum Newton-Verfahren ohne Ableitungsberechnung. Verwendet zwei Punkte $(x_{n-1}, f(x_{n-1}))$ und $(x_n, f(x_n))$:

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \cdot f(x_n)$$

Benötigt zwei Startwerte x_0 und x_1.

Sekantenverfahren Nullstellen von $f(x) = x^2 - 2$

Startwerte $x_0 = 1$ und $x_1 = 2$

1. $x_2 = 1 - \frac{1-2}{1^2-2} \cdot 1 = 1.5$
2. $x_3 = 1.5 - \frac{1.5-1}{1.5^2-2} \cdot 1.5 = 1.4545$
3. $x_4 = 1.4545 - \frac{1.4545-1.5}{1.4545^2-2} \cdot 1.4545 = 1.4143$

→ Konvergenz gegen $\sqrt{2}$ nach wenigen Schritten

Newton für Nichtlineare Systeme Bestimmen Sie die Nullstelle des Systems: $f_1(x, y) = x^2 + y^2 - 1 = 0$ $f_2(x, y) = y - x^2 = 0$

Lösung:

1. Jacobi-Matrix aufstellen:

$$J = \begin{pmatrix} 2x & 2y \\ -2x & 1 \end{pmatrix}$$

2. Newton-Iteration:

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} x_k \\ y_k \end{pmatrix} - J^{-1}(x_k, y_k) \begin{pmatrix} f_1(x_k, y_k) \\ f_2(x_k, y_k) \end{pmatrix}$$

3. Mit Startwert $(0.5, 0.25)$ erste Iteration durchführen

Implementierung von Nullstellenverfahren

1. Grundstruktur:
   - Funktion $f(x)$ definieren
   - Ableitung $f'(x)$ falls nötig
   - Abbruchkriterien festlegen
   - Iterations-Schleife implementieren
2. Abbruchkriterien:

```python
def newton(f, df, x0, eps=1e-6, maxiter=100):
    x = x0
    for i in range(maxiter):
        fx = f(x)
        if abs(fx) < eps:  # Funktionswert klein genug
            return x
        dfx = df(x)
        if abs(dfx) < eps:  # Division durch Null vermeiden
            raise ValueError("Ableitung nahe Null")
        x_new = x - fx/dfx
        if abs(x_new - x) < eps:  # Aenderung klein genug
            return x_new
        x = x_new
    raise RuntimeError("Keine Konvergenz")
```

3. Fehlerbehandlung beachten:
   - Division durch Null bei $f'(x) = 0$
   - Keine Konvergenz
   - Über-/Unterlauf bei großen Zahlen

Fehleranalyse der Verfahren Vergleich der Fehlerkonvergenz für $f(x) = e^x - x - 2$:

Theoretisch:

- Newton: $|e_{n+1}| \leq C|e_n|^2$ mit $e_n = x_n - \xi$
- Sekanten: $|e_{n+1}| \leq C|e_n|^{1.618}$
- Fixpunkt: $|e_{n+1}| \leq \alpha|e_n|$ mit $\alpha < 1$

Praktisch:

Mit $x_0 = 1$:

n	$\lvert x_n - \xi\rvert_{Newton}$	$\lvert x_n - \xi\rvert_{Sekanten}$	$\lvert x_n - \xi\rvert_{Fixpunkt}$
1	1.0e-1	2.3e-1	3.1e-1
2	5.2e-3	4.5e-2	9.6e-2
3	1.4e-5	3.8e-3	3.0e-2

Vergleichende Fehleranalyse

1. Theoretische Analyse:
   - Konvergenzordnung bestimmen
   - Fehlerabschätzungen herleiten
   - Konvergenzbereich identifizieren
2. Praktische Analyse:
   - Iterationsschritte zählen
   - Relative Fehler berechnen
   - Konvergenzgeschwindigkeit visualisieren
3. Vergleichskriterien:
   - Rechenaufwand pro Iteration
   - Anzahl benötigter Iterationen
   - Robustheit/Zuverlässigkeit
   - Konvergenzbereich

Fehleranalyse der Verfahren Vergleich der Fehlerkonvergenz für $f(x) = e^x - x - 2$:

Theoretisch:

- Newton: $|e_{n+1}| \leq C|e_n|^2$ mit $e_n = x_n - \xi$
- Sekanten: $|e_{n+1}| \leq C|e_n|^{1.618}$
- Fixpunkt: $|e_{n+1}| \leq \alpha|e_n|$ mit $\alpha < 1$

Praktisch:

Mit $x_0 = 1$:

n	$\lvert x_n - \xi\rvert_{Newton}$	$\lvert x_n - \xi\rvert_{Sekanten}$	$\lvert x_n - \xi\rvert_{Fixpunkt}$
1	1.0e-1	2.3e-1	3.1e-1
2	5.2e-3	4.5e-2	9.6e-2
3	1.4e-5	3.8e-3	3.0e-2

Fehlerabschätzung

Fehlerabschätzung für Nullstellen

So schätzen Sie den Fehler einer Näherungslösung ab:

1. Sei x_n der aktuelle Näherungswert
2. Wähle Toleranz $\epsilon > 0$
3. Prüfe Vorzeichenwechsel: $f(x_n - \epsilon) \cdot f(x_n + \epsilon) < 0$
4. Falls ja: Nullstelle liegt in $(x_n - \epsilon, x_n + \epsilon)$
5. Damit gilt: $|x_n - \xi| < \epsilon$

Praktische Fehlerabschätzung Fehlerbestimmung bei $f(x) = x^2 - 2$

1. Näherungswert: $x_3 = 1.4142157$
2. Mit $\epsilon = 10^{-5}$:
3. $f(x_3 - \epsilon) = 1.4142057^2 - 2 < 0$
4. $f(x_3 + \epsilon) = 1.4142257^2 - 2 > 0$

Also: $|x_3 - \sqrt{2}| < 10^{-5}$

→ Nullstelle liegt in $(1.4142057, 1.4142257)$

Konvergenzverhalten

Konvergenzordnung Sei (x_n) eine gegen $\bar{x}$ konvergierende Folge. Die Konvergenzordnung $q \geq 1$ ist definiert durch:

$$|x_{n+1} - \bar{x}| \leq c \cdot |x_n - \bar{x}|^q$$

wo $c > 0$ eine Konstante. Für $q = 1$ muss zusätzl. $c < 1$ gelten.

Konvergenzordnungen der Verfahren Konvergenzgeschwindigkeiten

Newton-Verfahren: Quadratische Konvergenz: $q = 2$

Vereinfachtes Newton: Lineare Konvergenz: $q = 1$

Sekantenverfahren: Superlineare Konvergenz: $q = \frac{1+\sqrt{5}}{2} \approx 1.618$

Konvergenzgeschwindigkeit Vergleich der Verfahren:
Startwert $x_0 = 1$, Funktion $f(x) = x^2 - 2$, Ziel: $\sqrt{2}$

n	Newton	Vereinfacht	Sekanten
1	1.5000000	1.5000000	1.5000000
2	1.4166667	1.4500000	1.4545455
3	1.4142157	1.4250000	1.4142857
4	1.4142136	1.4125000	1.4142136

LGS und Matrizen

Matrizen

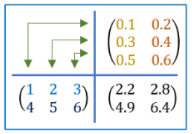
Matrix Tabelle mit m Zeilen und n Spalten: $m \times n$-Matrix A
a_{ij}: Element in der i-ten Zeile und j-ten Spalte

Addition und Subtraktion
- $A + B = C$
- $c_{ij} = a_{ij} + b_{ij}$

Skalarmultiplikation
- $k \cdot A = B$
- $b_{ij} = k \cdot a_{ij}$

Rechenregeln für die Addition und skalare Multiplikation von Matrizen
Kommutativ-, Assoziativ- und Distributiv-Gesetz gelten für Matrix-Addition

Matrixmultiplikation $A^{m\times n}$, $B^{n\times k}$
Bedingung: A n Spalten, B n Zeilen.
Resultat: C hat m Zeilen und k Spalten.
- $A \cdot B = C$
- $c_{ij} = a_{i1} \cdot b_{1j} + a_{i2} \cdot b_{2j} + \ldots + a_{in} \cdot b_{nj}$
- $A \cdot B \neq B \cdot A$



Rechenregeln für die Multiplikation von Matrizen Assoziativ, Distributiv, nicht Kommutativ!

Transponierte Matrix $A^{m\times n} \to (A^T)^{n\times m}$
- A^T: Spalten und Zeilen vertauscht
- $(A^T)_{ij} = A_{ji}$ und $(A \cdot B)^T = B^T \cdot A^T$

$$\begin{pmatrix} Z_1 \rightarrow \\ Z_2 \rightarrow \\ Z_3 \rightarrow \end{pmatrix}^T = \left(\boxed{Z_1 \downarrow} \; \boxed{Z_2 \downarrow} \; \boxed{Z_3 \downarrow} \right)$$

Spezielle Matrizen
- **Symmetrische Matrix**: $A^T = A$
- **Einheitsmatrix/Identitätsmatrix**: E bzw. I
  mit $e_{ij} = 1$ für $i = j$ und $e_{ij} = 0$ für $i \neq j$
- **Diagonalmatrix**: $a_{ij} = 0$ für $i \neq j$
- **Dreiecksmatrix**: $a_{ij} = 0$ für $i > j$ (obere Dreiecksmatrix)
  oder $i < j$ (untere Dreiecksmatrix)

Lineare Gleichungssysteme (LGS)

Lineares Gleichungssystem (LGS) Ein *lineares Gleichungssystem* ist eine Sammlung von Gleichungen, die linear in den Unbekannten sind. Ein LGS kann in Matrixform $A \cdot \vec{x} = \vec{b}$ dargestellt werden.
A: Koeffizientenmatrix
$\vec{x}$: Vektor der Unbekannten
$\vec{b}$: Vektor der Konstanten

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \cdots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

Rang einer Matrix $rg(A)$ = Anzahl Zeilen - Anzahl Nullzeilen
$\Rightarrow$ Anzahl linear unabhängiger Zeilen- oder Spaltenvektoren

Zeilenstufenform (Gauss)
- Alle Nullen stehen unterhalb der Diagonalen, Nullzeilen zuunterst
- Die erste Zahl $\neq 0$ in jeder Zeile ist eine führende Eins
- Führende Einsen, die weiter unten stehen $\rightarrow$ stehen weiter rechts

Reduzierte Zeilenstufenform: (Gauss-Jordan)
Alle Zahlen links und rechts der führenden Einsen sind Nullen.

Gauss-Jordan-Verfahren
1. bestimme linkeste Spalte mit Elementen $\neq 0$ (Pivot-Spalte)
2. oberste Zahl in Pivot-Spalte $= 0$
   $\rightarrow$ vertausche Zeilen so dass $a_{11} \neq 0$
3. teile erste Zeile durch $a_{11} \rightarrow$ so erhalten wir führende Eins
4. Nullen unterhalb führender Eins erzeugen (Zeilenperationen)

nächste Schritte: ohne bereits bearbeitete Zeilen Schritte 1-4 wiederholen, bis Matrix Zeilenstufenform hat

Zeilenperationen erlaubt bei LGS (z.B. Gauss-Verfahren)
- Vertauschen von Zeilen
- Multiplikation einer Zeile mit einem Skalar
- Addition eines Vielfachen einer Zeile zu einer anderen

Lösbarkeit von linearen Gleichungssystemen
- Lösbar: $rg(A) = rg(A|b)$
- genau eine Lösung: $rg(A) = n$
- unendlich viele Lösungen: $rg(A) < n$

Parameterdarstellung bei unendlich vielen Lösungen
Führende Unbekannte: Spalte mit führender Eins
Freie Unbekannte: Spalten ohne führende Eins

$$\begin{array}{cccc|c} x_1 & x_2 & x_3 & x_4 & \\ \hline 1 & -2 & 0 & 3 & 5 \\ 0 & 0 & 1 & 1 & 3 \end{array}$$

Auflösung nach der führenden Unbekannten:
- $1x_1 - 2x_2 + 0x_3 + 3x_4 = 5 \quad x_2 = \lambda \rightarrow x_1 = 5 + 2 \cdot \lambda - 3 \cdot \mu$
- $0x_1 + 0x_2 + 1x_3 + 1x_4 = 3 \quad x_4 = \mu \rightarrow x_3 = 3 - \mu$

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 5+2\lambda-3\mu \\ \lambda \\ 3-\mu \\ \mu \end{pmatrix} = \begin{pmatrix} 5 \\ 0 \\ 3 \\ 0 \end{pmatrix} + \lambda \begin{pmatrix} 2 \\ 1 \\ 0 \\ 0 \end{pmatrix} + \mu \begin{pmatrix} -3 \\ 0 \\ -1 \\ 1 \end{pmatrix}$$

Homogenes LGS $\vec{b} = \vec{0} \rightarrow A \cdot \vec{x} = \vec{0} \rightarrow rg(A) = rg(A \mid \vec{b})$
nur zwei Möglichkeiten:
- eine Lösung $x_1 = x_2 = \cdots = x_n = 0$, die sog. *triviale Lösung.*
- unendlich viele Lösungen

Koeffizientenmatrix, Determinante, Lösbarkeit des LGS
Für $n \times n$-Matrix A sind folgende Aussagen äquivalent:
- $\det(A) \neq 0$
- $rg(A) = n$
- A ist invertierbar
- Spalten von A sind linear unabhängig.
- Zeilen von A sind linear unabhängig.
- LGS $A \cdot \vec{x} = \vec{0}$ hat eindeutige Lösung $x = A^{-1} \cdot 0 = 0$

Quadratische Matrizen

Umformen bestimme die Matrix X: $A \cdot X + B = 2 \cdot X$
$\Rightarrow A \cdot X = 2 \cdot X - B \Rightarrow A \cdot X - 2 \cdot X = -B \Rightarrow (A - 2 \cdot E) \cdot X = -B$
$\Rightarrow (A - 2 \cdot E) \cdot (A - 2 \cdot E)^{-1} \cdot X = (A - 2 \cdot E)^{-1} \cdot -B$
$\Rightarrow X = (A - 2 \cdot E)^{-1} \cdot -B$

Inverse

Inverse einer quadratischen Matrix A A^{-1}
A^{-1} existiert, wenn $rg(A) = n$. A^{-1} ist eindeutig bestimmt.

Eine Matrix heisst *invertierbar / regulär*, wenn sie eine Inverse hat. Andernfalls heisst sie *singulär*

Eigenschaften invertierbarer Matrizen
- $A \cdot A^{-1} = A^{-1} \cdot A = E$
- $(A^{-1})^{-1} = A$
- $(A \cdot B)^{-1} = B^{-1} \cdot A^{-1}$ Die Reihenfolge ist relevant!
  A und B invertierbar $\Rightarrow$ AB invertierbar
- $(A^T)^{-1} = (A^{-1})^T$ A invertierbar $\Rightarrow$ A^T invertierbar

Inverse einer 2×2-Matrix $A = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ mit $det(A) = ad - bc$

$$A^{-1} = \frac{1}{\det(A)} \cdot \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

NUR Invertierbar falls $ad - bc \neq 0$

Inverse berechnen einer quadratischen Matrix $A^{n\times n}$

$$A \cdot A^{-1} = E \rightarrow \left(A|E\right) \rightsquigarrow \text{Zeilenoperationen} \rightsquigarrow \left(E|A^{-1}\right)$$

$$\underbrace{\begin{pmatrix} 4 & -1 & 0 \\ 0 & 2 & 1 \\ 3 & -5 & -2 \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{pmatrix}}_{A^{-1}} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{E}$$

$$\rightarrow \left(\begin{array}{ccc|ccc} 4 & -1 & 0 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 & 1 & 0 \\ 3 & -5 & -2 & 0 & 0 & 1 \end{array}\right)$$

Zeilenstufenform (linke Seite)

$$\rightsquigarrow \left(\begin{array}{ccc|ccc} 1 & -1/4 & 0 & 1/4 & 0 & 0 \\ 0 & 1 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 1 & -6 & 17 & 8 \end{array}\right)$$

Reduzierte Zeilenstufenform (linke Seite)

$$\rightsquigarrow \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & -2 & -1 \\ 0 & 1 & 0 & 3 & -8 & -4 \\ 0 & 0 & 1 & -6 & 17 & 8 \end{array}\right) \Rightarrow A^{-1} = \begin{pmatrix} 1 & -2 & -1 \\ 3 & -8 & -4 \\ -6 & 17 & 8 \end{pmatrix}$$

LGS mit Inverse lösen $A \cdot \vec{x} = \vec{b}$

$$A^{-1} \cdot A \cdot \vec{x} = A^{-1} \cdot \vec{b} \rightarrow \vec{x} = A^{-1} \cdot \vec{b}$$

Beispiel:

$$\underbrace{\begin{pmatrix} 1/2 & 0 \\ 0 & 1/3 \end{pmatrix}}_{A^{-1}} \cdot \underbrace{\begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} x \\ y \end{pmatrix}}_{\vec{x}} = \underbrace{\begin{pmatrix} 1/2 & 0 \\ 0 & 1/3 \end{pmatrix}}_{A^{-1}} \cdot \underbrace{\begin{pmatrix} 4 \\ 5 \end{pmatrix}}_{\vec{b}}$$

Numerische Lösung linearer Gleichungssysteme

Linear System of Equations Given $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$, find $x \in \mathbb{R}^n$ such that:

$$Ax = b \quad \text{or} \quad \sum_{j=1}^{n} a_{ij}x_j = b_i, \quad i = 1, \dots, n$$

Systematisches Vorgehen bei LGS

1. Eigenschaften der Matrix analysieren:
   - Diagonaldominanz prüfen
   - Konditionszahl berechnen oder abschätzen
   - Symmetrie erkennen
2. Verfahren auswählen:
   - Direkte Verfahren: für kleinere Systeme
   - Iterative Verfahren: für große, dünnbesetzte Systeme
   - Spezialverfahren: für symmetrische/bandförmige Matrizen
3. Implementation planen:
   - Pivotisierung bei Gauss berücksichtigen
   - Speicherbedarf beachten
   - Abbruchkriterien festlegen

Systematische Verfahrensauswahl für LGS

1. Eigenschaften der Matrix analysieren:
   - Diagonaldominanz prüfen
   - Konditionszahl berechnen oder abschätzen
   - Symmetrie erkennen
2. Verfahren auswählen:
   - Direkte Verfahren: für kleinere Systeme
   - Iterative Verfahren: für große, dünnbesetzte Systeme
   - Spezialverfahren: für symmetrische/bandförmige Matrizen

Verfahrensauswahl Gegeben ist das LGS $Ax = b$ mit:

$$A = \begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Analyse:
- Matrix ist symmetrisch
- Streng diagonaldominant
- Dünnbesetzt (tridiagonal)

Verfahrenswahl:
- Gauss: möglich wegen kleiner Dimension
- Gauss-Seidel: konvergiert wegen Diagonaldominanz
- LR-Zerlegung: effizient wegen Bandstruktur

Method Selection Given system $Ax = b$ with:

$$A = \begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Analysis:
- Matrix is symmetric
- Strictly diagonally dominant
- Sparse (tridiagonal)

Method Choice:
- Gauss: feasible due to small size
- Gauss-Seidel: converges due to diagonal dominance
- LR: efficient for band structure

Verfahrensauswahl Gegeben ist das LGS $Ax = b$ mit:

$$A = \begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Analyse:
- Matrix ist symmetrisch
- Streng diagonaldominant
- Dünnbesetzt (tridiagonal)

Verfahrenswahl:
- Gauss: möglich wegen kleiner Dimension
- Gauss-Seidel: konvergiert wegen Diagonaldominanz
- LR-Zerlegung: effizient wegen Bandstruktur

Permutationsmatrix P ist eine Matrix, die aus der Einheitsmatrix durch Zeilenvertauschungen entsteht.

Für die Vertauschung der i-ten und j-ten Zeile hat P_k die **Form**:
- $p_{ii} = p_{jj} = 0$
- $p_{ij} = p_{ji} = 1$
- Sonst gleich wie in E_n

Wichtige Eigenschaften:
- $P^{-1} = P^T = P$
- Mehrere Vertauschungen: $P = P_l \cdot \ldots \cdot P_1$

Zeilenvertauschung für Matrix A mit Permutationsmatrix P_1:

$$\underbrace{\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}}_{P_1} = \begin{pmatrix} 7 & 8 & 9 \\ 4 & 5 & 6 \\ 1 & 2 & 3 \end{pmatrix}$$

$\Rightarrow A \cdot P_1$ bewirkt die Vertauschung von Zeile 1 und 3

Pivotisierung

Spaltenpivotisierung
Strategie zur numerischen Stabilisierung des Gauss-Algorithmus durch Auswahl des betragsmäßig größten Elements als Pivotelement. Vor jedem Eliminationsschritt in Spalte i:
- Suche k mit $|a_{ki}| = \max\{|a_{ji}| \mid j = i, \dots, n\}$
- Falls $a_{ki} \neq 0$: Vertausche Zeilen i und k
- Falls $a_{ki} = 0$: Matrix ist singulär

Gauss-Algorithmus mit Pivotisierung

1. Elimination (Vorwärts):
- Für $i = 1, \dots, n-1$:
  - Finde $k \geq i$ mit $|a_{ki}| = \max\{|a_{ji}| \mid j = i, \dots, n\}$
  - Falls $a_{ki} = 0$: Stop (Matrix singulär)
  - Vertausche Zeilen i und k
  - Für $j = i+1, \dots, n$:
    - $z_j := z_j - \frac{a_{ji}}{a_{ii}} z_i$

2. Rückwärtseinsetzen: $x_i = \frac{b_i - \sum_{j=i+1}^{n} a_{ij}x_j}{a_{ii}}, \quad i = n, n-1, \dots, 1$

Gauss mit Pivotisierung $A = \begin{pmatrix} 0 & 1 & 1 \\ 2 & 4 & -2 \\ 0 & 3 & 15 \end{pmatrix}, \quad b = \begin{pmatrix} 4 \\ 2 \\ 36 \end{pmatrix}$

Eliminationsschritte:

$$\left(\begin{array}{ccc|c} 2 & 4 & -2 & 2 \\ 0 & 3 & 15 & 36 \\ 0 & 1 & 1 & 4 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 2 & 4 & -2 & 2 \\ 0 & 3 & 15 & 36 \\ 0 & 0 & -2 & -8 \end{array}\right)$$

Rückwärtseinsetzen:

$$x_3 = \frac{-8}{-2} = 4$$
$$x_2 = \frac{36 - 15(4)}{3} = 1$$
$$x_1 = \frac{2 - 4(4) + 2}{2} = -6$$

Vorteile der Permutationsmatrix
- Exakte Nachverfolgung aller Zeilenvertauschungen
- Einfache Rückführung auf ursprüngliche Reihenfolge durch P^{-1}
- Kompakte Darstellung mehrerer Vertauschungen
- Numerisch stabile Implementierung der Pivotisierung

Zeilenvertauschungen verfolgen
1. Initialisiere $P = I_n$
2. Für jede Vertauschung von Zeile i und j:
   - Erstelle P_k durch Vertauschen von Zeilen i, j in I_n
   - Aktualisiere $P = P_k \cdot P$
   - Wende Vertauschung auf Matrix an: $A := P_k A$
3. Bei der LR-Zerlegung mit Pivotisierung:
   - $PA = LR$
   - Löse $Ly = Pb$ und $Rx = y$

Gauss-Algorithmus mit Pivotisierung
1. Vorbereitung:
   - Erweiterte Matrix $(A|b)$ aufstellen
   - Pivotisierungsstrategie wählen
2. Elimination:
   - Für jede Spalte i:
   - Pivotelement in Spalte i bestimmen
   - Zeilenvertauschung falls nötig
   - Nullen unterhalb erzeugen
3. Rückwärtseinsetzen:

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} a_{ij}x_j}{a_{ii}}, \quad i = n, n-1, \dots, 1$$

4. Kontrolle:
   - Residuum $\|Ax - b\|$ berechnen
   - Pivotisierungsschritte protokollieren

Gauss mit Pivotisierung Lösen Sie $Ax = b$ mit:

$$A = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & -1 \\ 4 & -2 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix}$$

Lösung:

1. Erste Spalte: Pivot $a_{31} = 4 \to$ Z1 $\leftrightarrow$ Z3

$$\left(\begin{array}{ccc|c} 4 & -2 & 1 & 0 \\ 2 & 4 & -1 & 2 \\ 1 & 2 & 1 & 1 \end{array}\right)$$

2. Eliminationsschritte:

$$\left(\begin{array}{ccc|c} 4 & -2 & 1 & 0 \\ 0 & 5 & -1.5 & 2 \\ 0 & 2.5 & 0.75 & 1 \end{array}\right)$$

$$\left(\begin{array}{ccc|c} 4 & -2 & 1 & 0 \\ 0 & 5 & -1.5 & 2 \\ 0 & 0 & 1.5 & 0.2 \end{array}\right)$$

3. Rückwärtseinsetzen:

$$x_3 = 0.2/1.5 = \frac{2}{15}$$
$$x_2 = (2 + 1.5 \cdot \frac{2}{15})/5 = 0.5$$
$$x_1 = (0 + 2 \cdot 0.5 - 1 \cdot \frac{2}{15})/4 = 0.2$$

Matrix-Zerlegungen

Dreieckszerlegung Eine Matrix $A \in \mathbb{R}^{n \times n}$ kann zerlegt werden in:

Untere Dreiecksmatrix L:	**Obere Dreiecksmatrix R:**
$l_{ij} = 0$ für $j > i$	$r_{ij} = 0$ für $i > j$
Diagonale normiert ($l_{ii} = 1$)	Diagonalelemente $\neq 0$

LR-Zerlegung

LR-Zerlegung
Jede reguläre Matrix A, für die der Gauss-Algorithmus ohne Zeilenvertauschungen durchführbar ist, lässt sich zerlegen in: $A = LR$ wobei L eine normierte untere und R eine obere Dreiecksmatrix ist.

Berechnung der LR-Zerlegung
So berechnen Sie die LR-Zerlegung:
1. Führen Sie Gauss-Elimination durch
2. R ist die resultierende obere Dreiecksmatrix
3. Die Eliminationsfaktoren $-\frac{a_{ji}}{a_{ii}}$ bilden L
4. Lösen Sie dann nacheinander:
   - $Ly = b$ (Vorwärtseinsetzen)
   - $Rx = y$ (Rückwärtseinsetzen)

LR-Zerlegung $A = \begin{pmatrix} -1 & 1 & 1 \\ 1 & -3 & -2 \\ 5 & 1 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 5 \\ 3 \end{pmatrix}$

Schritt 1: Erste Spalte

Max. Element in 1. Spalte: $|a_{31}| = 5$, also Z1 und Z3 tauschen:

$$P_1 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}, \quad A^{(1)} = \begin{pmatrix} 5 & 1 & 4 \\ 1 & -3 & -2 \\ -1 & 1 & 1 \end{pmatrix}$$

Berechne Eliminationsfaktoren: $l_{21} = \frac{1}{5}, \quad l_{31} = -\frac{1}{5}$

Nach Elimination: $A^{(2)} = \begin{pmatrix} 5 & 1 & 4 \\ 0 & -3.2 & -2.8 \\ 0 & 1.2 & 1.8 \end{pmatrix}$

Schritt 2: Zweite Spalte

Max. Element in 2. Spalte unter Diagonale: $|-3.2| > |1.2|$, keine Vertauschung nötig.

Berechne Eliminationsfaktor: $l_{32} = \frac{1.2}{-3.2} = -\frac{3}{8}$

Nach Elimination: $R = \begin{pmatrix} 5 & 1 & 4 \\ 0 & -3.2 & -2.8 \\ 0 & 0 & 0.75 \end{pmatrix}$

Endergebnis

Die LR-Zerlegung mit $PA = LR$ ist:

$$P = P_1 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}, L = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{5} & 1 & 0 \\ -\frac{1}{5} & -\frac{3}{8} & 1 \end{pmatrix}, R = \begin{pmatrix} 5 & 1 & 4 \\ 0 & -3.2 & -2.8 \\ 0 & 0 & 0.75 \end{pmatrix}$$

Lösung des Systems

1. $Pb = \begin{pmatrix} 3 \\ 5 \\ 0 \end{pmatrix}$
2. Löse $Ly = Pb$ durch Vorwärtseinsetzen: $y = \begin{pmatrix} 3 \\ 4.4 \\ 2.25 \end{pmatrix}$
3. Löse $Rx = y$ durch Rückwärtseinsetzen: $x = \begin{pmatrix} -1 \\ -4 \\ 3 \end{pmatrix}$

Probe

$$Ax = \begin{pmatrix} -1 & 1 & 1 \\ 1 & -3 & -2 \\ 5 & 1 & 4 \end{pmatrix} \begin{pmatrix} -1 \\ -4 \\ 3 \end{pmatrix} = \begin{pmatrix} 0 \\ 5 \\ 3 \end{pmatrix} = b$$

LR-Zerlegung durchführen
1. Zerlegung bestimmen:
   - Gauss-Schritte durchführen
   - Eliminationsfaktoren in L speichern
   - Resultierende Dreiecksmatrix ist R
2. System lösen:
   - Vorwärtseinsetzen: $Ly = b$
   - Rückwärtseinsetzen: $Rx = y$
3. Bei Pivotisierung:
   - Permutationsmatrix P erstellen
   - $PA = LR$ speichern
   - $Ly = Pb$ lösen

LR-Zerlegung Bestimmen Sie die LR-Zerlegung von:

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 4 & -2 & -1 \\ -2 & 3 & -1 \end{pmatrix}$$

Lösung:

1. Eliminationsfaktoren:
   - $l_{21} = 4/2 = 2$
   - $l_{31} = -2/2 = -1$
   - $l_{32} = 1/(-4) = -0.25$
2. Zerlegung:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & -0.25 & 1 \end{pmatrix}$$

$$R = \begin{pmatrix} 2 & 1 & 1 \\ 0 & -4 & -3 \\ 0 & 0 & -0.25 \end{pmatrix}$$

QR-Zerlegung

QR-Zerlegung
Eine orthogonale Matrix $Q \in \mathbb{R}^{n\times n}$ erfüllt: $Q^TQ = QQ^T = I_n$

Die QR-Zerlegung einer Matrix A ist: $A = QR$

wobei Q orthogonal und R eine obere Dreiecksmatrix ist.

Householder-Transformation
Eine Householder-Matrix hat die Form: $H = I_n - 2uu^T$
mit $u \in \mathbb{R}^n$, $\|u\| = 1$. Es gilt:
- H ist orthogonal ($H^T = H^{-1}$) und symmetrisch ($H^T = H$)
- $H^2 = I_n$

QR-Zerlegung mit Householder
1. Initialisierung: $R := A$, $Q := I_n$
2. Für $i = 1, \dots, n-1$:
   - Bilde Vektor v_i aus i-ter Spalte von R ab Position i
   - $w_i := v_i + \text{sign}(v_{i1})\|v_i\|e_1$
   - $u_i := w_i/\|w_i\|$
   - $H_i := I_{n-i+1} - 2u_iu_i^T$
   - Erweitere H_i zu Q_i durch I_{i-1} links oben
   - $R := Q_iR$ und $Q := QQ_i^T$

QR-Zerlegung mit Householder $A = \begin{pmatrix} 2 & 5 & -1 \\ -1 & -4 & 2 \\ 0 & 2 & 1 \end{pmatrix}$

Schritt 1: Erste Spalte
Erste Spalte a_1 und Einheitsvektor e_1: $a_1 = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}$, $e_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$

Householder-Vektor für erste Spalte:

1. Berechne Norm: $|a_1| = \sqrt{2^2 + (-1)^2 + 0^2} = \sqrt{5}$
2. Bestimme Vorzeichen: $\text{sign}(a_{11}) = \text{sign}(2) = 1$
   - Wähle positives Vorzeichen, da erstes Element positiv
   - Dies maximiert die erste Komponente von v_1
   - Verhindert Auslöschung bei der Subtraktion
3. $v_1 = a_1 + \text{sign}(a_{11})|a_1|e_1 = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix} + \sqrt{5}\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 2+\sqrt{5} \\ -1 \\ 0 \end{pmatrix}$
4. Normiere v_1: $|v_1| = \sqrt{(2+\sqrt{5})^2 + 1} \Rightarrow u_1 = \frac{v_1}{|v_1|} = \begin{pmatrix} 0.91 \\ -0.41 \\ 0 \end{pmatrix}$

Householder-Matrix berechnen: $H_1 = I - 2u_1u_1^T = \begin{pmatrix} -0.67 & -0.75 & 0 \\ -0.75 & 0.67 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

A nach 1. Transformation: $A^{(1)} = H_1A = \begin{pmatrix} -\sqrt{5} & -6.71 & 0.45 \\ 0 & -0.89 & 1.79 \\ 0 & 2.00 & 1.00 \end{pmatrix}$

Schritt 2: Zweite Spalte
Untermatrix für zweite Transformation: $A_2 = \begin{pmatrix} -0.89 & 1.79 \\ 2.00 & 1.00 \end{pmatrix}$

Householder-Vektor für zweite Spalte:

1. $|a_2| = \sqrt{(-0.89)^2 + 2^2} = 2.19$
2. $\text{sign}(a_{22}) = \text{sign}(-0.89) = -1$ (da erstes Element negativ)
3. $v_2 = \begin{pmatrix} -0.89 \\ 2.00 \end{pmatrix} - 2.19\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -3.09 \\ 2.00 \end{pmatrix}$
4. $u_2 = \frac{v_2}{|v_2|} = \begin{pmatrix} -0.84 \\ 0.54 \end{pmatrix}$

Erweiterte Householder-Matrix: $Q_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -0.41 & -0.91 \\ 0 & -0.91 & 0.41 \end{pmatrix}$

nach 2. Transformation: $R = Q_2A^{(1)} = \begin{pmatrix} -\sqrt{5} & -6.71 & 0.45 \\ 0 & -2.19 & 1.34 \\ 0 & 0 & -1.79 \end{pmatrix}$

Endergebnis
Die QR-Zerlegung $A = QR$ ist:

$Q = H_1^TQ_2^T = \begin{pmatrix} -0.89 & -0.45 & 0 \\ 0.45 & -0.89 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $R = \begin{pmatrix} -\sqrt{5} & -6.71 & 0.45 \\ 0 & -2.19 & 1.34 \\ 0 & 0 & -1.79 \end{pmatrix}$

Probe
1. $QR = A$ (bis auf Rundungsfehler)
2. $Q^TQ = QQ^T = I$ (Orthogonalität)
3. R ist obere Dreiecksmatrix

Wichtige Beobachtungen
- Vorzeichenwahl bei v_k ist entscheidend für numerische Stabilität
- Ein falsches Vorzeichen kann zu Auslöschung führen
- Betrag der Diagonalelemente in R = Norm transformierter Spalten
- Q ist orthogonal: Spaltenvektoren sind orthonormal

Fehleranalyse

Matrix- und Vektornormen
Eine Vektornorm $\|\cdot\|$ erfüllt für alle $x, y \in \mathbb{R}^n, \lambda \in \mathbb{R}$:
- $\|x\| \geq 0$ und $\|x\| = 0 \Leftrightarrow x = 0$
- $\|\lambda x\| = |\lambda| \cdot \|x\|$
- $\|x + y\| \leq \|x\| + \|y\|$ (Dreiecksungleichung)

Wichtige Normen
1-Norm: $\|x\|_1 = \sum_{i=1}^n |x_i|, \|A\|_1 = \max_j \sum_{i=1}^n |a_{ij}|$

2-Norm: $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}, \|A\|_2 = \sqrt{\rho(A^TA)}$

∞-Norm: $\|x\|_\infty = \max_i |x_i|, \|A\|_\infty = \max_i \sum_{j=1}^n |a_{ij}|$

Fehlerabschätzung für LGS
Sei $\|\cdot\|$ eine Norm, $A \in \mathbb{R}^{n\times n}$ regulär und $Ax = b$, $A\tilde{x} = \tilde{b}$

Absoluter Fehler:
$\|x - \tilde{x}\| \leq \|A^{-1}\| \cdot \|b - \tilde{b}\|$

Relativer Fehler:
$\frac{\|x-\tilde{x}\|}{\|x\|} \leq \text{cond}(A) \cdot \frac{\|b-\tilde{b}\|}{\|b\|}$

Mit der Konditionszahl $\text{cond}(A) = \|A\| \cdot \|A^{-1}\|$

Konditionierung
Die Konditionszahl beschreibt die numerische Stabilität eines LGS:
- $\text{cond}(A) \approx 1$: gut konditioniert
- $\text{cond}(A) \gg 1$: schlecht konditioniert
- $\text{cond}(A) \to \infty$: singulär

Konditionierung $A = \begin{pmatrix} 1 & 1 \\ 1 & 1.01 \end{pmatrix}, b = \begin{pmatrix} 2 \\ 2.01 \end{pmatrix}$
Konditionszahl: $\text{cond}(A) = \|A\| \cdot \|A^{-1}\| \approx 400$

Fehlerabschätzung
Absoluter Fehler: $\|x - \tilde{x}\| \leq 400 \cdot 0.01 = 4$

Relativer Fehler: $\frac{\|x-\tilde{x}\|}{\|x\|} \leq 400 \cdot \frac{0.01}{2} = 2$

Vergleich Lösungsverfahren $A = \begin{pmatrix} 1 & 2 & 0 \\ 2 & 1 & 2 \\ 0 & 2 & 1 \end{pmatrix}$, $b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$
- Matrix ist symmetrisch und nicht streng diagonaldominant
- $\text{cond}_\infty(A) \approx 12.5$

Verfahren	Iterationen	Residuum	Zeit
LR mit Pivot	1	$2.2 \cdot 10^{-16}$	1.0
QR	1	$2.2 \cdot 10^{-16}$	2.3
Jacobi	12	$1.0 \cdot 10^{-6}$	1.8
Gauss-Seidel	7	$1.0 \cdot 10^{-6}$	1.4

- Direkte Verfahren erreichen höhere Genauigkeit
- Iterative Verfahren brauchen mehrere Schritte

Iterative Verfahren

Zerlegung der Systemmatrix
A zerlegt in: $A = L + D + R$
- L: streng untere Dreiecksmatrix
- D: Diagonalmatrix
- R: streng obere Dreiecksmatrix

Jacobi-Verfahren
Gesamtschrittverfahren

Iteration: $x^{(k+1)} = -D^{-1}(L + R)x^{(k)} + D^{-1}b$

Komponentenweise: $x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1, j\neq i}^n a_{ij}x_j^{(k)}\right)$

Gauss-Seidel-Verfahren
Einzelschrittverfahren

Iteration: $x^{(k+1)} = -(D + L)^{-1}Rx^{(k)} + (D + L)^{-1}b$

Komponentenweise:
$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)}\right)$

Implementierung von Jacobi- und Gauss-Seidel-Verfahren

Vorbereitungsphase
- Matrix zerlegen in $A = L + D + R$
- Diagonaldominanz prüfen: $|a_{ii}| > \sum_{j\neq i} |a_{ij}|$ für alle i
- Sinnvolle Startwerte wählen (z.B. $x^{(0)} = 0$ oder $x^{(0)} = b$)
- Toleranz ϵ und max. Iterationszahl n_{max} festlegen

Verfahren durchführen
- **Jacobi**: Komponentenweise parallel berechnen
- **Gauss-Seidel**: Komponentenweise sequentiell berechnen

Konvergenzprüfung
- Absolute Änderung: $\|x^{(k+1)} - x^{(k)}\| < \epsilon$
- Relatives Residuum: $\frac{\|Ax^{(k)}-b\|}{\|b\|} < \epsilon$
- Maximale Iterationszahl: $k < n_{max}$

A-priori Fehlerabschätzung
- Spektralradius ρ der Iterationsmatrix bestimmen
- Benötigte Iterationen n für Genauigkeit ϵ:
  $n \geq \frac{\ln(\epsilon(1-\rho)/\|x^{(1)}-x^{(0)}\|)}{\ln(\rho)}$

Iterative Verfahren implementieren
1. Matrix zerlegen:
   - $A = L + D + R$ für Jacobi
   - $A = (L + D) + R$ für Gauss-Seidel
2. Konvergenz prüfen:
   - Diagonaldominanz
   - Spektralradius der Iterationsmatrix
3. Iteration durchführen:
   - Jacobi: $x^{(k+1)} = -D^{-1}(L + R)x^{(k)} + D^{-1}b$
   - Gauss-Seidel: $x^{(k+1)} = -(D + L)^{-1}Rx^{(k)} + (D + L)^{-1}b$
4. Abbruchkriterien:
   - Residuum: $\|Ax^{(k)} - b\| < \epsilon$
   - Änderung: $\|x^{(k+1)} - x^{(k)}\| < \epsilon$
   - Maximale Iterationen

Implementation of Iterative Methods

1. Preparation:
   - Split matrix $A = L + D + R$
   - Check diagonal dominance
   - Choose sensible starting values
   - Set tolerance ϵ and max iterations
2. Execute Method:
   - Jacobi: Compute components in parallel
   - Gauss-Seidel: Compute sequentially
3. Check Convergence:
   - Absolute change: $\|x^{(k+1)} - x^{(k)}\| < \epsilon$
   - Relative residual: $\frac{\|Ax^{(k)}-b\|}{\|b\|} < \epsilon$
   - Maximum iterations: $k < k_{max}$

Jacobi vs Gauss-Seidel Gegeben sei das System:

$$\begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix} x = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Vergleich nach 4 Iterationen:

k	Jacobi	Gauss-Seidel
1	0.250	0.250
2	0.281	0.297
3	0.295	0.299
4	0.298	0.300

Beobachtungen:

- Gauss-Seidel konvergiert schneller
- Beide Verfahren konvergieren monoton
- Konvergenz gegen $x_1 = 0.3$

Konvergenzkriterien Ein iteratives Verfahren konvergiert, wenn:

1. Die Matrix A diagonaldominant ist:
   $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$ für alle i
2. Der Spektralradius der Iterationsmatrix kleiner 1 ist:
   $\rho(B) < 1$ mit B als jeweilige Iterationsmatrix

Konvergenzverhalten $\begin{pmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$

Die Matrix ist diagonaldominant: $|a_{ii}| = 4 > 1 = \sum_{j \neq i} |a_{ij}|$

k	Residuum		Rel. Fehler	
	Jacobi	G-S	Jacobi	G-S
0	3.74	3.74	-	-
1	0.94	0.47	0.935	0.468
2	0.23	0.06	0.246	0.125
3	0.06	0.01	0.065	0.017
4	0.01	0.001	0.016	0.002

Beobachtungen:

- Gauss-Seidel konvergiert etwa doppelt so schnell wie Jacobi
- Das Residuum fällt linear (geometrische Folge)
- Die Konvergenz ist gleichmäßig (keine Oszillationen)

Aufwandsabschätzung

Rechenaufwand Anzahl benötigter Gleitkommaoperationen für n×n Systeme:

- Gauss-Elimination: $\frac{2}{3}n^3 + \frac{3}{2}n^2 - \frac{13}{6}n$
- LR-Zerlegung: $\frac{2}{3}n^3 + \frac{7}{2}n^2 - \frac{13}{6}n$
- QR-Zerlegung: $\frac{5}{3}n^3 + 4n^2 + \frac{7}{3}n - 7$
- Cramer'sche Regel: $n(n+1)! - 1$ (nicht praktikabel)

Aufwand für n×n Systeme abschätzen

1. Hauptterm identifizieren:
   - Kubische Terme dominieren für große n
   - Niedrigere Terme vernachlässigbar
2. Ordnung bestimmen:
   - Gauss/LR: $\mathcal{O}(n^3)$
   - QR: $\mathcal{O}(n^3)$ aber höherer Vorfaktor
   - Iterative Verfahren: $\mathcal{O}(n^2)$ pro Iteration
3. Speicherbedarf berücksichtigen:
   - Direkte Verfahren: $\mathcal{O}(n^2)$
   - Iterative Verfahren: $\mathcal{O}(n)$ für dünnbesetzte Matrizen

Rechenzeit-Vergleich System mit $n = 1000$ auf einem 100 GFLOPS Rechner:

Verfahren	Operationen	Zeit
Gauss	$0.67 \cdot 10^9$	6.7 ms
LR	$0.67 \cdot 10^9$	6.7 ms
QR	$1.67 \cdot 10^9$	16.7 ms
Cramer	$> 10^{2567}$	>Jahre

Spezialfälle

Bandmatrizen Matrizen mit $a_{ij} = 0$ für $|i - j| > m$ (Bandbreite $2m + 1$)

- Speicheraufwand: $\mathcal{O}(mn)$ statt $\mathcal{O}(n^2)$
- Rechenaufwand: $\mathcal{O}(mn^2)$ statt $\mathcal{O}(n^3)$
- Häufig bei FEM/Differentialgleichungen

Bandmatrizen effizient lösen

1. Speicherformat wählen:
   - Nur Diagonalen speichern
   - Komprimierte Speicherung
   - Spezielle Datenstrukturen
2. Algorithmen anpassen:
   - Nullen außerhalb des Bandes ignorieren
   - Pivotisierung nur innerhalb Band
   - Bandstruktur erhalten
3. Effizienz prüfen:
   - Bandbreite vs. Systemgröße
   - Direktes vs. iteratives Verfahren
   - Speicherverbrauch kontrollieren

Tridiagonales System Bandmatrix mit $m = 1$ (Bandbreite 3):

$$\begin{pmatrix} b_1 & c_1 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 \\ 0 & a_3 & b_3 & c_3 \\ 0 & 0 & a_4 & b_4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{pmatrix}$$

Effizienter Thomas-Algorithmus:

1. Vorwärtselimination:

$$c'_1 = c_1/b_1, \quad d'_1 = d_1/b_1$$
$$b'_i = b_i - a_i c'_{i-1}$$
$$c'_i = c_i/b'_i$$
$$d'_i = (d_i - a_i d'_{i-1})/b'_i$$

2. Rückwärtssubstitution:

$$x_n = d'_n$$
$$x_i = d'_i - c'_i x_{i+1}$$

Komplexität:

- Speicher: $\mathcal{O}(n)$ statt $\mathcal{O}(n^2)$
- Rechenzeit: $\mathcal{O}(n)$ statt $\mathcal{O}(n^3)$

Komplexe Zahlen

Fundamentalsatz der Algebra

Eine algebraische Gleichung n-ten Grades mit komplexen Koeffizienten:

$$a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0 = 0$$

besitzt in $\mathbb{C}$ genau n Lösungen (mit Vielfachheiten gezählt).

Komplexe Zahlen

Die Menge der komplexen Zahlen $\mathbb{C}$ erweitert die reellen Zahlen $\mathbb{R}$ durch Einführung der imaginären Einheit i mit der Eigenschaft:

$$i^2 = -1$$

Eine komplexe Zahl z ist ein geordnetes Paar (x, y) mit $x, y \in \mathbb{R}$:

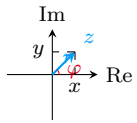
$$z = x + iy$$

Die Menge aller komplexen Zahlen ist definiert als:

$$\mathbb{C} = \{z \mid z = x + iy \text{ mit } x, y \in \mathbb{R}\}$$

Bestandteile komplexer Zahlen

Realteil: $\mathrm{Re}(z) = x$
Imaginärteil: $\mathrm{Im}(z) = y$
Betrag: $|z| = \sqrt{x^2 + y^2} = \sqrt{z \cdot z^*}$
Konjugation: $\overline{z} = x - iy$



Darstellungsformen

- Normalform: $z = x + iy$
- Trigonometrische Form: $z = r(\cos\varphi + i\sin\varphi)$
- Exponentialform: $z = re^{i\varphi}$

Umrechnung zwischen Darstellungsformen komplexer Zahlen

Von Normalform in trigonometrische Form/Exponentialform

1. Berechne Betrag $r = \sqrt{x^2 + y^2}$
2. Berechne Winkel mit einer der Formeln:
   - $\varphi = \arctan(\frac{y}{x})$ falls $x > 0$
   - $\varphi = \arctan(\frac{y}{x}) + \pi$ falls $x < 0$
   - $\varphi = \frac{\pi}{2}$ falls $x = 0, y > 0$
   - $\varphi = -\frac{\pi}{2}$ falls $x = 0, y < 0$
   - φ unbestimmt falls $x = y = 0$
3. Trigonometrische Form: $z = r(\cos\varphi + i\sin\varphi)$
4. Exponentialform: $z = re^{i\varphi}$

Von trigonometrischer Form in Normalform

1. Realteil: $x = r\cos\varphi$
2. Imaginärteil: $y = r\sin\varphi$
3. Normalform: $z = x + iy$

Von Exponentialform in Normalform/trigonometrische Form

1. Trigonometrische Form durch Euler-Formel:
   $re^{i\varphi} = r(\cos\varphi + i\sin\varphi)$
2. Dann wie oben in Normalform umrechnen

Wichtige Hinweise:

- Achten Sie auf das korrekte Quadranten beim Winkel
- Winkelfunktionen im Bogenmaß verwenden
- Bei Umrechnung in Normalform Euler-Formel nutzen
- Vorzeichen bei Exponentialform beachten

Komplexe Zahlen umrechnen

1. Normalform $\leftrightarrow$ Polarform:
   - Betrag: $r = \sqrt{x^2 + y^2}$
   - Winkel: $\varphi = \arctan(\frac{y}{x})$ (Quadrant beachten!)
   - Normalform: $z = x + iy$
   - Polarform: $z = r(\cos\varphi + i\sin\varphi) = re^{i\varphi}$
2. Rechenoperationen:
   - Addition: $(x_1 + iy_1) + (x_2 + iy_2) = (x_1 + x_2) + i(y_1 + y_2)$
   - Multiplikation: $r_1 r_2 e^{i(\varphi_1+\varphi_2)}$
   - Division: $\frac{r_1}{r_2}e^{i(\varphi_1-\varphi_2)}$
   - n-te Potenz: $r^n e^{in\varphi}$

Komplexe Operationen Gegeben $z_1 = 1 + i$ und $z_2 = 2 - i$:

Umrechnung in Polarform:

- $z_1 : r_1 = \sqrt{2}$, $\varphi_1 = \frac{\pi}{4}$
- $z_2 : r_2 = \sqrt{5}$, $\varphi_2 = -\arctan(\frac{1}{2})$

Berechnungen:

- $z_1 \cdot z_2 = (2 - i)(1 + i) = (2 + 1) + i(2 - 1) = 3 + i$
- $z_1^3 = (\sqrt{2})^3(\cos(\frac{3\pi}{4}) + i\sin(\frac{3\pi}{4}))$

Rechenoperationen mit komplexen Zahlen

Für $z_1 = x_1 + iy_1$ und $z_2 = x_2 + iy_2$ gilt:

Addition: $z_1 + z_2 = (x_1 + x_2) + i(y_1 + y_2)$

Subtraktion: $z_1 - z_2 = (x_1 - x_2) + i(y_1 - y_2)$

Multiplikation:
$$z_1 \cdot z_2 = (x_1x_2 - y_1y_2) + i(x_1y_2 + x_2y_1)$$
$$= r_1r_2e^{i(\varphi_1+\varphi_2)} \text{ (in Exponentialform)}$$

Division:
$$\frac{z_1}{z_2} = \frac{z_1 \cdot z_2^*}{z_2 \cdot z_2^*} = \frac{(x_1x_2 + y_1y_2) + i(y_1x_2 - x_1y_2)}{x_2^2 + y_2^2}$$
$$= \frac{r_1}{r_2}e^{i(\varphi_1-\varphi_2)} \text{ (in Exponentialform)}$$

Potenzen und Wurzeln

Für eine komplexe Zahl in Exponentialform $z = re^{i\varphi}$ gilt:

- n-te Potenz: $z^n = r^n e^{in\varphi} = r^n(\cos(n\varphi) + i\sin(n\varphi))$
- n-te Wurzel: $z_k = \sqrt[n]{r}e^{i\frac{\varphi+2\pi k}{n}}$, $k = 0, 1, \ldots, n-1$

Eigenwerte und Eigenvektoren

Eigenwerte und Eigenvektoren

Für eine Matrix $A \in \mathbb{R}^{n\times n}$ heißt $\lambda \in \mathbb{C}$ Eigenwert von A, wenn es einen Vektor $x \in \mathbb{C}^n \backslash \{0\}$ gibt mit:

$$Ax = \lambda x$$

Der Vektor x heißt dann Eigenvektor zum Eigenwert λ.

Bestimmung von Eigenwerten

Ein Skalar λ ist genau dann Eigenwert von A, wenn gilt:

$$\det(A - \lambda I_n) = 0$$

Diese Gleichung heißt charakteristische Gleichung. Das zugehörige Polynom

$$p(\lambda) = \det(A - \lambda I_n)$$

ist das charakteristische Polynom von A.

Eigenschaften von Eigenwerten

Für eine Matrix $A \in \mathbb{R}^{n\times n}$ gilt:

$$\det(A) = \prod_{i=1}^{n} \lambda_i \text{ (Produkt der Eigenwerte)}$$

$$\operatorname{tr}(A) = \sum_{i=1}^{n} \lambda_i \text{ (Summe der Eigenwerte)}$$

- Bei Dreiecksmatrix sind die Diagonalelemente die Eigenwerte
- Ist λ Eigenwert von A, so ist $\frac{1}{\lambda}$ Eigenwert von A^{-1}

Vielfachheiten

Für einen Eigenwert λ unterscheidet man:

- Algebraische Vielfachheit:
  Vielfachheit als Nullstelle des charakteristischen Polynoms
- Geometrische Vielfachheit:
  Dimension des Eigenraums $= n - \operatorname{rg}(A - \lambda I_n)$

Die geometrische Vielfachheit ist stets kleiner oder gleich der algebraischen Vielfachheit.

Bestimmung von Eigenwerten und Eigenvektoren

Vorbereitung

- Matrix $A \in \mathbb{R}^{n\times n}$ aufschreiben
- Charakteristische Matrix $(A - \lambda I)$ aufstellen

Eigenwerte bestimmen

1. Charakteristisches Polynom aufstellen:
   - Bei 2×2 Matrizen direkt: $\det(A - \lambda I)$
   - Bei 3×3 Matrizen: Entwicklung nach einer Zeile/Spalte
   - Bei größeren Matrizen: Spezielle Eigenschaften nutzen (z.B. Dreiecksform, Symmetrie)
2. Polynom vereinfachen und auf Nullform bringen:
   - Ausmultiplizieren
   - Zusammenfassen nach Potenzen von λ
   - Form: $p(\lambda) = (-1)^n\lambda^n + a_{n-1}\lambda^{n-1} + \cdots + a_1\lambda + a_0$
3. Nullstellen bestimmen:
   - Bei quadratischer Gleichung: Mitternachtsformel
   - Bei Grad 3: Substitution oder Cardanische Formeln
   - Bei höherem Grad: Numerische Verfahren

Eigenvektoren bestimmen

1. Für jeden Eigenwert λ_i:
   - Matrix $(A - \lambda_i I)$ aufstellen
   - Homogenes LGS $(A - \lambda_i I)x = 0$ lösen
   - Lösungsvektor normieren falls gewünscht
2. Bei mehrfachen Eigenwerten:
   - Basis des Eigenraums bestimmen
   - Linear unabhängige Eigenvektoren finden

Kontrolle

- Für jeden Eigenvektor x_i prüfen: $Ax_i = \lambda_i x_i$
- Bei 2×2 Matrix: $\lambda_1 + \lambda_2 = \operatorname{tr}(A)$ und $\lambda_1 \cdot \lambda_2 = \det(A)$
- Bei 3×3 Matrix zusätzlich: $\sum \lambda_i = \operatorname{tr}(A)$ und $\prod \lambda_i = \det(A)$
- Bei reellen Matrizen: Komplexe Eigenwerte treten in konjugierten Paaren auf

Spezialfälle beachten

- Bei Dreiecksmatrizen: Eigenwerte sind die Diagonalelemente
- Bei symmetrischen Matrizen: Alle Eigenwerte sind reell
- Bei orthogonalen Matrizen: $|\lambda_i| = 1$ für alle Eigenwerte
- Bei nilpotenten Matrizen: Alle Eigenwerte sind 0

Eigenwertberechnung Gegeben ist die Matrix $A = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix}$

1. Charakteristisches Polynom aufstellen:

$$\det(A - \lambda I) = \begin{vmatrix} 2-\lambda & 1 & 0 \\ 1 & 2-\lambda & 1 \\ 0 & 1 & 2-\lambda \end{vmatrix}$$

2. Entwicklung nach 1. Zeile:

$$p(\lambda) = (2 - \lambda)\begin{vmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{vmatrix} - 1\begin{vmatrix} 1 & 1 \\ 1 & 2-\lambda \end{vmatrix}$$

3. Ausrechnen:

$$p(\lambda) = (2 - \lambda)((2 - \lambda)^2 - 1) - ((2 - \lambda) - 1) = -\lambda^3 + 6\lambda^2 - 11\lambda + 6$$

4. Nullstellen bestimmen: $\lambda_1 = 1, \lambda_2 = 2, \lambda_3 = 3$
5. Eigenvektoren bestimmen für $\lambda_1 = 1$:

$(A - I)x = 0$ führt zu $x_1 = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$

Eigenwerte bestimmen

1. Charakteristisches Polynom aufstellen:
   - $p(\lambda) = \det(A - \lambda I)$ berechnen
   - Auf Standardform bringen
2. Nullstellen bestimmen:
   - Quadratische Formel für $n = 2$
   - Cardano-Formel für $n = 3$
   - Numerische Verfahren für $n > 3$
3. Vielfachheiten bestimmen:
   - Algebraische Vielfachheit: Nullstellenordnung
   - Geometrische Vielfachheit: $n - \text{rang}(A - \lambda I)$

Charakteristisches Polynom Bestimmen Sie die Eigenwerte von:

$$A = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}$$

Lösung:

1. $p(\lambda) = \det(A - \lambda I)$:

$$\begin{vmatrix} 2-\lambda & -1 & 0 \\ -1 & 2-\lambda & -1 \\ 0 & -1 & 2-\lambda \end{vmatrix}$$

2. Determinante entwickeln:

$$p(\lambda) = (2-\lambda)^3 - 2(2-\lambda) = -\lambda^3 + 6\lambda^2 - 11\lambda + 6$$

3. Nullstellen:

$$\lambda_1 = 1, \lambda_2 = 2, \lambda_3 = 3$$

Eigenvektoren bestimmen

1. Für jeden Eigenwert λ:
   - $(A - \lambda I)x = 0$ aufstellen
   - Homogenes LGS lösen
   - Lösungsvektor normieren
2. Bei mehrfachen Eigenwerten:
   - Geometrische Vielfachheit bestimmen
   - Basis des Eigenraums finden
3. Kontrolle:
   - $Ax = \lambda x$ überprüfen
   - Orthogonalität bei symmetrischen Matrizen
   - Linear unabhängig?

Eigenvektoren Bestimmen Sie die Eigenvektoren zum Eigenwert $\lambda = 2$ der Matrix:

$$A = \begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 2 \end{pmatrix}$$

Lösung:

1. $(A - 2I)x = 0$:

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

2. Homogenes System lösen:
   - $x_2 = 0$ (aus 1. Zeile)
   - x_1, x_3 frei wählbar
3. Basis des Eigenraums:

$$v_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, v_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Numerische Berechnung von Eigenwerten

Ähnliche Matrizen

Zwei Matrizen $A, B \in \mathbb{R}^{n \times n}$ heißen ähnlich, wenn es eine reguläre Matrix T gibt mit:

$$B = T^{-1}AT$$

Eine Matrix A heißt diagonalisierbar, wenn sie ähnlich zu einer Diagonalmatrix D ist:

$$D = T^{-1}AT$$

Eigenschaften ähnlicher Matrizen

Für ähnliche Matrizen A und $B = T^{-1}AT$ gilt:

1. A und B haben dieselben Eigenwerte mit gleichen algebraischen Vielfachheiten
2. Ist x Eigenvektor von B zum Eigenwert λ, so ist Tx Eigenvektor von A zum Eigenwert λ
3. Bei Diagonalisierbarkeit:
   - Die Diagonalelemente von D sind die Eigenwerte von A
   - Die Spalten von T sind die Eigenvektoren von A

Spektralradius Der Spektralradius einer Matrix A ist definiert als:

$$\rho(A) = \max\{|\lambda| \mid \lambda \text{ ist Eigenwert von } A\}$$

Er gibt den Betrag des betragsmäßig größten Eigenwerts an.

Von-Mises-Iteration

Von-Mises-Iteration (Vektoriteration)

Für eine diagonalisierbare Matrix A mit Eigenwerten $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$ konvergiert die Folge:

$$v^{(k+1)} = \frac{Av^{(k)}}{\|Av^{(k)}\|_2}, \quad \lambda^{(k+1)} = \frac{(v^{(k)})^T Av^{(k)}}{(v^{(k)})^T v^{(k)}}$$

gegen einen Eigenvektor v zum betragsmäßig größten Eigenwert λ_1.

Von-Mises-Iteration / Vektoriteration

Algorithmus

1. Startvektor $v^{(0)}$ wählen:
   - Zufälligen Vektor oder $(1, \ldots, 1)^T$ wählen
   - Auf Länge 1 normieren: $\|v^{(0)}\|_2 = 1$
2. Für $k = 0, 1, 2, \ldots$ bis zur Konvergenz:
   - Iterationsvektor berechnen: $w^{(k)} = Av^{(k)}$
   - Normieren: $v^{(k+1)} = \frac{w^{(k)}}{\|w^{(k)}\|_2}$
   - Eigenwertapproximation (Rayleigh-Quotient):

$$\lambda^{(k+1)} = \frac{(v^{(k)})^T Av^{(k)}}{(v^{(k)})^T v^{(k)}}$$

3. Abbruchkriterien prüfen:
   - Änderung des Eigenvektors: $\|v^{(k+1)} - v^{(k)}\| < \varepsilon$
   - Änderung des Eigenwertes: $|\lambda^{(k+1)} - \lambda^{(k)}| < \varepsilon$
   - Maximale Iterationszahl erreicht

Verifikation

- Prüfen ob $Av^{(k)} \approx \lambda^{(k)} v^{(k)}$
- Residuum berechnen: $\|Av^{(k)} - \lambda^{(k)} v^{(k)}\|$
- Orthogonalität zu anderen Eigenvektoren prüfen

Von-Mises-Iteration Gegeben sei die Matrix $A = \begin{pmatrix} 4 & -1 & 1 \\ -1 & 3 & -2 \\ 1 & -2 & 3 \end{pmatrix}$

Mit Startvektor $v^{(0)} = \frac{1}{\sqrt{3}}(1, 1, 1)^T$:

1. Erste Iteration:
   - $w^{(0)} = Av^{(0)} = \frac{1}{\sqrt{3}}(4, 0, 2)^T$
   - $v^{(1)} = \frac{w^{(0)}}{\|w^{(0)}\|} = \frac{1}{\sqrt{20}}(4, 0, 2)^T$
   - $\lambda^{(1)} = (v^{(0)})^T Av^{(0)} = 3.33$
2. Zweite Iteration:
   - $w^{(1)} = Av^{(1)} = \frac{1}{\sqrt{20}}(18, -2, 8)^T$
   - $v^{(2)} = \frac{w^{(1)}}{\|w^{(1)}\|} = \frac{1}{\sqrt{388}}(18, -2, 8)^T$
   - $\lambda^{(2)} = 5.12$

   Konvergenz gegen $\lambda_1 \approx 5.17$ und $v = (0.89, -0.10, 0.39)^T$

Vektoriteration durchführen

1. Voraussetzungen prüfen:
   - Matrix diagonalisierbar
   - $|\lambda_1| > |\lambda_2|$
2. Iteration:
   - $w^{(k)} = Av^{(k)}$
   - $v^{(k+1)} = \frac{w^{(k)}}{\|w^{(k)}\|}$
   - $\lambda^{(k+1)} = \frac{(v^{(k)})^T Av^{(k)}}{(v^{(k)})^T v^{(k)}}$
3. Konvergenz:
   - $v^{(k)} \to$ Eigenvektor zu $|\lambda_1|$
   - $\lambda^{(k)} \to |\lambda_1|$

Von-Mises-Iteration Bestimmen Sie den betragsmäßig größten Eigenwert von:

$$A = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}$$

Lösung:

1. Start mit $v^{(0)} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$
2. Erste Iteration:
   - $w^{(0)} = \begin{pmatrix} 4 \\ 4 \end{pmatrix}$
   - $v^{(1)} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$
   - $\lambda^{(1)} = 4$
3. Ergebnis:
   - Eigenvektor bereits gefunden
   - Eigenwert $\lambda = 4$ ist korrekt

QR-Verfahren

QR-Verfahren

Das QR-Verfahren transformiert die Matrix A iterativ in eine obere Dreiecksmatrix, deren Diagonalelemente die Eigenwerte sind:

1. Initialisierung: $A_0 := A$, $P_0 := I_n$
2. Für $i = 0, 1, 2, \ldots$:
   - QR-Zerlegung: $A_i = Q_i R_i$
   - Neue Matrix: $A_{i+1} = R_i Q_i$
   - Update: $P_{i+1} = P_i Q_i$

QR-Verfahren

Voraussetzungen

- Matrix $A \in \mathbb{R}^{n \times n}$
- Eigenwerte sollten verschiedene Beträge haben für gute Konvergenz

Algorithmus

1. Initialisierung:
   - $A_0 := A$
   - $Q_0 := I_n$
2. Für $k = 0, 1, 2, \ldots$ bis zur Konvergenz:
   - QR-Zerlegung von A_k berechnen: $A_k = Q_k R_k$
   - Neue Matrix berechnen: $A_{k+1} = R_k Q_k$
   - Transformationsmatrix aktualisieren: $P_{k+1} = P_k Q_k$
3. Abbruchkriterien prüfen:
   - Subdiagonalelemente nahe Null: $|a_{i+1,i}| < \varepsilon$
   - Änderung der Diagonalelemente klein
   - Maximale Iterationszahl erreicht

Auswertung

- Eigenwerte: Diagonalelemente von A_k
- Eigenvektoren: Spalten der Matrix P_k
- Bei 2×2-Blöcken: Komplexe Eigenwertpaare

QR-Verfahren Gegeben sei die Matrix $A = \begin{pmatrix} 1 & 2 & 0 \\ 2 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$

1. Erste Iteration:
   - QR-Zerlegung: $Q_1 = \begin{pmatrix} 0.45 & 0.89 & 0 \\ 0.89 & -0.45 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $R_1 = \begin{pmatrix} 2.24 & 2.24 & 0.45 \\ 0 & -1 & 0.89 \\ 0 & 0 & 1 \end{pmatrix}$
   - $A_1 = R_1 Q_1 = \begin{pmatrix} 2.24 & 0.45 & 0.45 \\ 0.45 & 0.38 & 0.89 \\ 0.45 & 0.89 & 1 \end{pmatrix}$
2. Nach Konvergenz: $A_k \approx \begin{pmatrix} 3 & * & * \\ 0 & 0 & * \\ 0 & 0 & 0 \end{pmatrix}$

   Eigenwerte sind also $\lambda_1 = 3, \lambda_2 = 0, \lambda_3 = 0$

QR-Algorithmus anwenden

1. Initialisierung:
   - $A_0 := A$
   - $Q_0 := I_n$
2. Iteration:
   - QR-Zerlegung: $A_k = Q_k R_k$
   - Neue Matrix: $A_{k+1} = R_k Q_k$
   - Update: $P_{k+1} = P_k Q_k$
3. Abbruch wenn:
   - Subdiagonalelemente klein
   - Diagonalelemente konvergieren
   - Maximale Iterationen erreicht

QR-Iteration Führen Sie eine QR-Iteration durch für:

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

Lösung:

1. QR-Zerlegung von A:

$$Q_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}, R_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} \sqrt{2} & 1 \\ 0 & -1 \end{pmatrix}$$

2. Neue Matrix:

$$A_1 = R_1 Q_1 = \begin{pmatrix} 1.5 & 0.5 \\ 0.5 & -0.5 \end{pmatrix}$$

3. Konvergenz nach mehreren Iterationen gegen:

$$A_\infty \approx \begin{pmatrix} \phi & 0 \\ 0 & -\phi^{-1} \end{pmatrix}$$

mit $\phi = \frac{1+\sqrt{5}}{2}$

Numerische Aspekte

1. Wahl des Startpunkts:
   - Von-Mises: zufälliger normierter Vektor
   - Inverse Iteration: Näherung für μ wichtig
   - QR: Matrix vorher auf Hessenberg-Form
2. Konvergenzprüfung:
   - Residuum $\|Ax^{(k)} - \lambda^{(k)} x^{(k)}\|$
   - Änderung in aufeinanderfolgenden Iterationen
   - Subdiagonalelemente bei QR
3. Spezialfälle:
   - Mehrfache Eigenwerte
   - Komplexe Eigenwerte/vektoren
   - Schlecht konditionierte Matrizen

Eigenwerte und Eigenvektoren

Eigenwerte und Eigenvektoren

Für eine Matrix $A \in \mathbb{R}^{n \times n}$ ist $\lambda \in \mathbb{C}$ ein Eigenwert und $x \in \mathbb{C}^n \setminus \{0\}$ ein zugehöriger Eigenvektor, wenn gilt:

$$Ax = \lambda x$$

Spektralradius

Der Spektralradius $\rho(A)$ einer Matrix A ist der betragsmäßig größte Eigenwert:

$$\rho(A) = \max\{|\lambda| : \lambda \text{ ist Eigenwert von } A\}$$

Der Spektralradius spielt eine wichtige Rolle bei der Konvergenz iterativer Verfahren.

Bestimmung von Eigenwerten und Eigenvektoren

1. Eigenwerte bestimmen:
   - Charakteristisches Polynom aufstellen: $p(\lambda) = \det(A - \lambda I)$
   - Nullstellen von $p(\lambda)$ finden
2. Für jeden Eigenwert λ:
   - Löse $(A - \lambda I)x = 0$
   - Bestimme Basisvektoren des Eigenraums
   - Normiere die Eigenvektoren falls gewünscht
3. Bei QR-Verfahren:
   - QR-Zerlegung iterativ durchführen
   - Diagonalelemente konvergieren gegen Eigenwerte
   - Produkt der Q-Matrizen ergibt Eigenvektoren

Eigenwerte und Eigenvektoren Bestimmen Sie Eigenwerte und -vektoren von:

$$A = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$$

Lösung:

1. Charakteristisches Polynom:

$$\det(A - \lambda I) = \begin{vmatrix} 2-\lambda & -1 \\ -1 & 2-\lambda \end{vmatrix} = (2-\lambda)^2 - 1 = 0$$

2. Eigenwerte: $\lambda_1 = 3$, $\lambda_2 = 1$
3. Eigenvektoren für $\lambda_1 = 3$:

$$(A - 3I)x = \begin{pmatrix} -1 & -1 \\ -1 & -1 \end{pmatrix} x = 0 \Rightarrow x_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

4. Eigenvektoren für $\lambda_2 = 1$:

$$(A - I)x = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} x = 0 \Rightarrow x_2 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Numerische Eigenwertberechnung mit QR-Verfahren

1. Vorbereitung:
   - Matrix $A_0 := A$ setzen
   - Maximale Iterationszahl und Toleranz festlegen
2. QR-Iteration:
   - QR-Zerlegung: $A_k = Q_k R_k$
   - Neue Matrix: $A_{k+1} = R_k Q_k$
   - Eigenvektormatrix: $V_{k+1} = V_k Q_k$
3. Konvergenzprüfung:
   - Nebendiagonalelemente nahe Null?
   - Änderung der Diagonalelemente klein genug?
   - Maximale Iterationen erreicht?

QR-Verfahren Bestimmen Sie die Eigenwerte der Matrix mit dem QR-Verfahren:

$$A = \begin{pmatrix} 1 & -1 \\ 4 & 3 \end{pmatrix}$$

Erste Iteration:

1. QR-Zerlegung von A_0:

$$Q_1 = \begin{pmatrix} 0.2425 & -0.9701 \\ 0.9701 & 0.2425 \end{pmatrix}, R_1 = \begin{pmatrix} 4.1231 & 2.6656 \\ 0 & -0.3656 \end{pmatrix}$$

2. Neue Matrix $A_1 = R_1 Q_1$:

$$A_1 = \begin{pmatrix} 3.8 & -0.9 \\ 0.9 & 0.2 \end{pmatrix}$$

Nach weiteren Iterationen konvergiert A_k gegen eine obere Dreiecksmatrix, deren Diagonalelemente die Eigenwerte sind: $\lambda_1 \approx 4$, $\lambda_2 \approx 0$.

Inverse Iteration

Inverse Iteration Die inverse Iteration berechnet einen Eigenvektor zu einem bekannten oder geschätzten Eigenwert μ durch:

$$v^{(k+1)} = \frac{(A - \mu I)^{-1} v^{(k)}}{\|(A - \mu I)^{-1} v^{(k)}\|_2}$$

Konvergiert typischerweise gegen den Eigenvektor zum betragsmäßig kleinsten Eigenwert $\lambda_i - \mu$.

Inverse Iteration anwenden

1. Vorbereitung:
   - Näherungswert μ für Eigenwert wählen
   - Zufälligen Startvektor $v^{(0)}$ normieren
   - LR-Zerlegung von $(A - \mu I)$ berechnen
2. Iteration durchführen:
   - LR-System $(A - \mu I)w^{(k)} = v^{(k)}$ lösen
   - Neuen Vektor normieren: $v^{(k+1)} = \frac{w^{(k)}}{\|w^{(k)}\|_2}$
   - Rayleigh-Quotient berechnen für Eigenwert
3. Abbruch wenn:
   - Residuum $\|(A - \lambda^{(k)} I)v^{(k)}\| < \epsilon$
   - Maximale Iterationszahl erreicht

Inverse Iteration Bestimmen Sie einen Eigenvektor zum Eigenwert $\lambda \approx 2$ der Matrix:

$$A = \begin{pmatrix} 2.1 & -0.1 & 0.1 \\ -0.1 & 2.0 & 0.2 \\ 0.1 & 0.2 & 1.9 \end{pmatrix}$$

Lösung:

1. $\mu = 2.0$ als Näherung wählen
2. Startvektor $v^{(0)} = \frac{1}{\sqrt{3}}(1, 1, 1)^T$
3. Erste Iteration:
   - $(A - 2I)w^{(0)} = v^{(0)}$ lösen
   - $v^{(1)} = \frac{w^{(0)}}{\|w^{(0)}\|} \approx (0.61, 0.63, 0.48)^T$
   - $\lambda^{(1)} \approx 2.01$

Vergleich der Eigenwertverfahren

1. Von-Mises Iteration:
   - Findet betragsmäßig größten Eigenwert
   - Einfach zu implementieren
   - Langsame lineare Konvergenz
2. Inverse Iteration:
   - Braucht Näherung für Eigenwert
   - Schnelle Konvergenz
   - LR-Zerlegung pro Schritt nötig
3. QR-Verfahren:
   - Berechnet alle Eigenwerte
   - Kubischer Aufwand pro Iteration
   - Globale und stabile Konvergenz

Numerischer Vergleich Matrix $A = \begin{pmatrix} 4 & -1 \\ 1 & 3 \end{pmatrix}$ mit $\lambda_1 = 5, \lambda_2 = 2$

Verfahren	Iterationen	Genauigkeit	Zeit
Von-Mises	23	10^{-8}	1.0
Inverse Iteration	6	10^{-8}	1.5
QR	8	10^{-12}	2.3

Beobachtungen:

- Von-Mises braucht viele Iterationen
- Inverse Iteration konvergiert schnell
- QR liefert höchste Genauigkeit

Typische Prüfungsaufgaben

1. Theorieaufgaben:
   - Eigenschaften von Eigenwerten beweisen
   - Konvergenzverhalten analysieren
   - Spezialfälle untersuchen
2. Rechenaufgaben:
   - Charakteristisches Polynom aufstellen
   - Eigenwerte/vektoren bestimmen
   - 2-3 Iterationsschritte durchführen
3. Implementierungsaufgaben:
   - Verfahren in Python implementieren
   - Konvergenzverhalten visualisieren
   - Verfahren vergleichen

Spektralradius und Anwendungen

Spektralradius Der Spektralradius $\rho(A)$ einer Matrix $A \in \mathbb{R}^{n \times n}$ ist definiert als der größte Absolutbetrag ihrer Eigenwerte:

$$\rho(A) = \max\{|\lambda| \mid \lambda \text{ ist Eigenwert von } A\}$$

Bedeutung des Spektralradius Der Spektralradius ist wichtig für:

- Konvergenz von Iterationsverfahren
- Stabilität dynamischer Systeme
- Abschätzung von Matrixnormen
- Konvergenz von Potenzreihen mit Matrizen

Konvergenzsatz Für eine Matrix $A \in \mathbb{R}^{n \times n}$ sind äquivalent:

- $\rho(A) < 1$
- $\lim_{k \to \infty} A^k = 0$
- Die Neumannsche Reihe $\sum_{k=0}^{\infty} A^k$ konvergiert
- $(I - A)$ ist invertierbar mit $(I - A)^{-1} = \sum_{k=0}^{\infty} A^k$

Spektralradius bestimmen und anwenden

1. Berechnung:
   - Eigenwerte λ_i bestimmen
   - Maximum der Absolutbeträge bilden
   - Bei großen Matrizen: numerische Verfahren
2. Konvergenzanalyse:
   - Bei Iterationsverfahren: $\rho(M) < 1$ prüfen
   - Bei Matrixpotenzen: $\rho(A) < 1$ prüfen
   - Konvergenzgeschwindigkeit $\approx |\rho(A)|^k$
3. Abschätzungen:
   - $\rho(A) \leq \|A\|$ für jede Matrixnorm
   - $\rho(AB) = \rho(BA)$ für beliebige Matrizen
   - $\rho(A^k) = [\rho(A)]^k$ für $k \in \mathbb{N}$

Spektralradius und Konvergenz Untersuchen Sie die Konvergenz des Jacobi-Verfahrens für:

$$A = \begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix}$$

Lösung:

1. Zerlegung $A = D + L + R$:

$$D = \begin{pmatrix} 4 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{pmatrix}, L + R = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

2. Jacobi-Matrix $M = -D^{-1}(L + R)$:

$$M = \begin{pmatrix} 0 & 1/4 & 0 \\ 1/4 & 0 & 1/4 \\ 0 & 1/4 & 0 \end{pmatrix}$$

3. Eigenwerte von M: $\lambda_1 = 0.5$, $\lambda_2 = 0$, $\lambda_3 = -0.5$
4. Spektralradius: $\rho(M) = 0.5 < 1$
5. Schlussfolgerung:
   - Jacobi-Verfahren konvergiert
   - Fehler reduziert sich pro Iteration etwa um Faktor 0.5
   - Konvergenzrate ist linear

Anwendungen des Spektralradius

1. Iterative Verfahren:
   - Jacobi: $\rho(-D^{-1}(L + R)) < 1$
   - Gauss-Seidel: $\rho(-(D + L)^{-1}R) < 1$
   - SOR: Optimaler Parameter ω bestimmen
2. Matrixreihen:
   - Konvergenz von $\sum_{k=0}^{\infty} A^k$
   - Existenz von $(I - A)^{-1}$
   - Abschätzung der Reihensumme
3. Stabilitätsanalyse:
   - Diskrete dynamische Systeme
   - Numerische Integration
   - Differenzenverfahren

Matrixreihe Untersuchen Sie die Konvergenz der Reihe $\sum_{k=0}^{\infty} A^k$ für:

$$A = \begin{pmatrix} 0 & 1/2 \\ 1/2 & 0 \end{pmatrix}$$

Lösung:

1. Eigenwerte bestimmen:

$$\det(A - \lambda I) = \lambda^2 - \frac{1}{4} = 0 \Rightarrow \lambda_{1,2} = \pm\frac{1}{2}$$

2. Spektralradius:

$$\rho(A) = \max\{|-\frac{1}{2}|, |\frac{1}{2}|\} = \frac{1}{2} < 1$$

3. Schlussfolgerungen:
   - Reihe konvergiert
   - $(I - A)$ ist invertierbar
   - Summe ist $(I - A)^{-1}$

Prüfungstipps

Allgemeine Hinweise

- Prüfungszeit: 120 Minuten für 6 Aufgaben → ca. 20 min pro Aufgabe
- Alle Aufgaben gleich gewichtet (10 Punkte)
- Lösungsweg muss vollständig und nachvollziehbar sein
- Zwischenschritte sind wichtig - auch bei falschen Endergebnissen gibt es Punkte
- Python-Code muss lauffähig sein und kommentiert werden

Was ist immer dabei?

1. Rechnerarithmetik/Konditionierung:
   - Maschinengenauigkeit berechnen
   - Darstellungsbereich bestimmen
   - Konditionszahl analysieren
   - Fehlerfortpflanzung abschätzen
2. Nullstellenverfahren:
   - Newton oder Fixpunktiteration
   - Konvergenznachweis (Banach)
   - A-priori/a-posteriori Abschätzungen
   - 2-3 Iterationsschritte von Hand
3. Lineare Gleichungssysteme:
   - Direkte Verfahren (Gauss, LR)
   - Iterative Verfahren (Jacobi, Gauss-Seidel)
   - Konvergenzbetrachtungen
   - Praktische Anwendungen
4. Eigenwerte:
   - Charakteristisches Polynom
   - Eigenvektoren berechnen
   - QR-Verfahren
   - Von-Mises Iteration

Typische Fallstricke

- Bei Konditionierung:
  - Vorzeichen bei Fehlerabschätzungen beachten
  - Grenzwertbetrachtungen durchführen
  - Auf Sonderfälle achten (z.B. $x \to 0$)
- Bei Nullstellenproblemen:
  - Konvergenzradius beachten
  - Startwert sinnvoll wählen
  - Abbruchkriterien definieren
- Bei LGS:
  - Pivotisierung nicht vergessen
  - Zeilenvertauschungen dokumentieren
  - Diagonaldominanz prüfen
- Bei Eigenwerten:
  - Vielfachheiten unterscheiden
  - Auf komplexe Eigenwerte achten
  - QR-Schritte sauber durchführen

Effiziente Prüfungsstrategie

1. Erste Durchsicht:
   - Alle Aufgaben überfliegen
   - Schwierigkeitsgrad einschätzen
   - Zeitplan erstellen
2. Bei jeder Aufgabe:
   - Methode identifizieren
   - Zwischenschritte planen
   - Ergebnisse verifizieren
3. Zeit einteilen:
   - Einfache Aufgaben zuerst
   - Zeit für Kontrolle einplanen
   - Nicht zu lange an einer Aufgabe festbeissen
4. Python-Code:
   - Grundgerüst schnell erstellen
   - Gut kommentieren
   - Ausgabe klar kennzeichnen

Musterlösung strukturieren Für eine typische Aufgabe:

1. Aufgabenstellung analysieren:
   - Welche Methode ist gefragt?
   - Was sind die gegebenen Grössen?
   - Was ist das Ziel?
2. Lösungsweg skizzieren:
   - Formeln aufschreiben
   - Zwischenschritte planen
   - Benötigte Berechnungen identifizieren
3. Berechnung durchführen:
   - Schrittweise vorgehen
   - Zwischenergebnisse notieren
   - Einheiten mitführen
4. Ergebnis überprüfen:
   - Plausibilitätskontrolle
   - Dimensionskontrolle
   - Eventuell Probe durchführen