# Lineare Eigenwertprobleme

Tristan Bergmann

30. Juni 2015

# Gliederung

# Gliederung

# Einleitung

Einfache Methode lineare Eigenwertprobleme zu lösen:

1. Diskretisierung -> Matrix mit Dimension $N$
2. 
3. 

# Einleitung

Einfache Methode lineare Eigenwertprobleme zu lösen:

1. Diskretisierung -> Matrix mit Dimension $N$
2. QR/QZ Algorithmus.
3. 

# Einleitung

Einfache Methode lineare Eigenwertprobleme zu lösen:

1. Diskretisierung -> Matrix mit Dimension $N$
2. QR/QZ Algorithmus.
3. Für verschiedene $N$ wiederholen. Nur die Eigenwerte auswählen, die bei verschiedenen $N$ gleich sind.

# Wo liegt das Problem?

1. QR/QZ Algorithmen haben Laufzeit $O(10N^3)$,
2. 
3. 
4. 
5. 

# Wo liegt das Problem?

1. QR/QZ Algorithmen haben Laufzeit $O(10N^3)$,
2. Ausgabe: endlich viele Eigenwerte,
3.
4.
5.

# Wo liegt das Problem?

1. QR/QZ Algorithmen haben Laufzeit $O(10N^3)$,
2. Ausgabe: endlich viele Eigenwerte,
3. Vergleichen ist anstrengend,
4. 
5. 

# Wo liegt das Problem?

1. QR/QZ Algorithmen haben Laufzeit O($10N^3$),
2. Ausgabe: endlich viele Eigenwerte,
3. Vergleichen ist anstrengend,
4. Konditionszahl der Matrizen wächst sehr schnell
5.

# Wo liegt das Problem?

1. QR/QZ Algorithmen haben Laufzeit O($10N^3$),
2. Ausgabe: endlich viele Eigenwerte,
3. Vergleichen ist anstrengend,
4. Konditionszahl der Matrizen wächst sehr schnell
5. unrealistische Werte

# Gliederung

# Definitionen

**Definition 17 (Eigenwertproblem)**

Ein Eigenwertproblem ist eine Gleichung der Form $Lu = \lambda Mu$, wobei $L$ und $M$ lineare differential oder integral Operatoren oder Matrizen sind und $\lambda \in \mathbb{R}$ der Eigenwert.

**Definition 18 (Eigenvektor)**

Ein $u$, welches die obere Gleichung für ein $\lambda$ löst, heißt Eigenvektor bzw. Eigenfunktion zu dem Eigenwert $\lambda$.

# Gliederung

# Beispiel

**Eigenwertproblem:**

$u_{xx} + \lambda u = 0; u(-1) = u(1) = 0$

Lösung:

$$u_j(x) = \begin{cases} cos(j\frac{\pi}{2}x) & \text{,wenn } j \text{ ungerade} \\ sin(j\frac{\pi}{2}x) & \text{,wenn } j \text{ gerade} \end{cases}$$

$\lambda_j = j^2\frac{\pi^2}{4}$; j=1,2,3,...

# Laufzeit

Nicht Optimierte Diskretisierung:

| N= | 25 | 36 | 49 | 64 | 81 | 100 | 100 zu 25 |
|---|---|---|---|---|---|---|---|
| Sec: | 190 | 430 | 840 | 1443 | 2414 | 3828 | 20x |

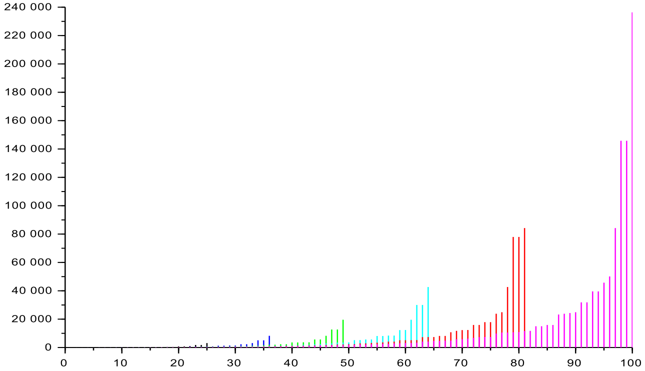
QR-Algorithmus:

| N= | 25 | 49 | 100 | 1000 | 2500 | 10000 | 10000 zu 2500 |
|---|---|---|---|---|---|---|---|
| Sec: | 0,07 | 0,07 | 0,1 | 7 | 40 | 1794 | 45x |

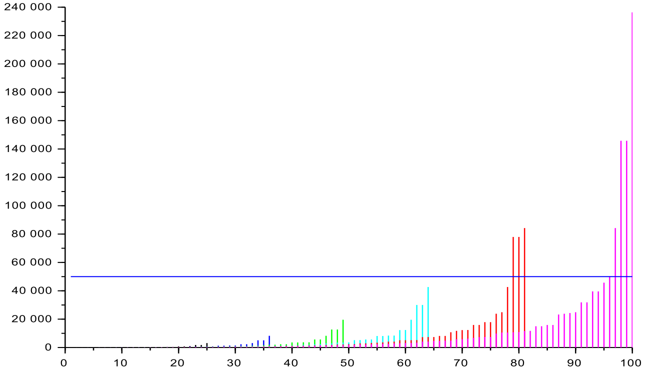
# Eigenwerte

| N= | 25 | 36 | 49 | 64 | 81 | 100 |
|---|---|---|---|---|---|---|
| 1.EW | 2.469 | 2.469 | 2.469 | 2.469 | 2.469 | 2.469 |
| 2.EW | 9.885 | 9.879 | 9.879 | 9.879 | 9.879 | 9.879 |
| 3.EW | 9.885 | 9.879 | 9.879 | 9.879 | 9.879 | 9.879 |
| 4.EW | 22.316 | 22.316 | 22.229 | 22.229 | 22.229 | 22.229 |
| 5.EW | 22.316 | 22.316 | 22.229 | 22.229 | 22.229 | 22.229 |
| 6.EW | 39.564 | 39.518 | 39.518 | 39.518 | 39.518 | 39.518 |
| 7.EW | 50.175 | 40.038 | 40.038 | 39.529 | 39.529 | 39.518 |
| 8.EW | 50.175 | 40.038 | 40.038 | 39.529 | 39.529 | 39.518 |
| 9.EW | 87.828 | 87.828 | 63.667 | 63.667 | 61.821 | 61.821 |
| 10.EW | 87.828 | 87.828 | 63.667 | 63.667 | 61.821 | 61.821 |
| N.EW | 3123 | 8254 | 19578 | 42663 | 84236 | 236374 |
| exakt | 2.467 | 9.869 | 22.206 | 39.478 | 61.685 | 88.826 |

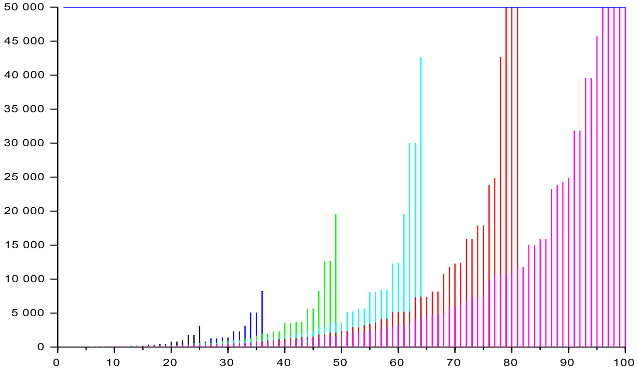
# Eigenwerte graphisch



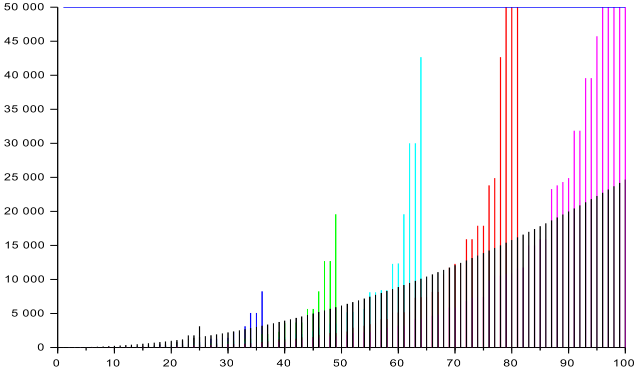
# Eigenwerte graphisch



# Eigenwerte graphisch



# Eigenwerte graphisch



# Rule-of-Thumb

## Rule-of-Thumb

Benutzt man $N + 1$ Terme in der beschränkten Spektralreihe, so sind gewöhnlich höchstens die niedrigeren $N/2$ Eigenwerte bis auf wenige Prozente genau.

# Gliederung

# Möglichkeit 1

Skalierung:

$\sigma_1 = |\lambda_1 - \lambda_2|$

$\sigma_j = \frac{1}{2}(|\lambda_j - \lambda_{j-1}| + |\lambda_{j+1} - \lambda_j|)$ , $j > 1$

Skalierte „ordinal“ Differenz:

$\delta_{j,ordinal} = |\lambda_j(N_1) - \lambda_j(N_2)|/\sigma_j$

# Möglichkeit 2

Skalierung:

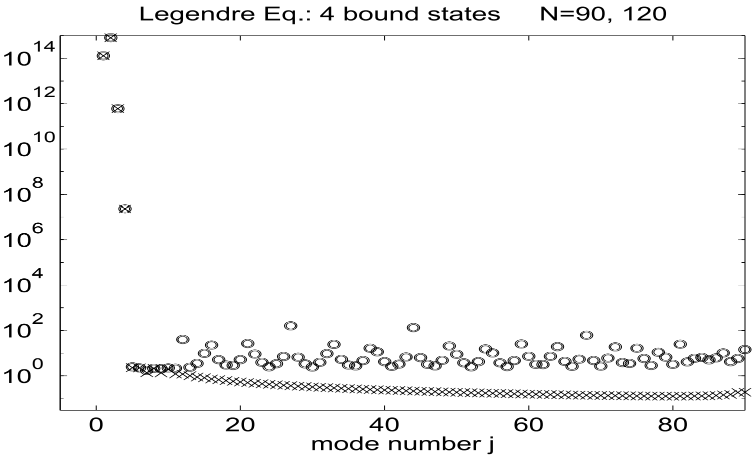
$\sigma_1 = |\lambda_1 - \lambda_2|$

$\sigma_j = \frac{1}{2}(|\lambda_j - \lambda_{j-1}| + |\lambda_{j+1} - \lambda_j|)$ , $j > 1$

Skalierte „nearest“ Differenz:

$$\delta_{j,nearest} = \min_{k \in [1, N_2]} |\lambda_j(N_1) - \lambda_k(N_2)| / \sigma_j$$

# Vergleich


Legendre Eq.: 4 bound states N=90, 120
10^14
10^12
10^10
10^8
10^6
10^4
10^2
10^0
0
20
40
60
80
mode number j

# Gliederung

# Falsche Eigenwerte

## Numerisch falsche Eigenwerte

Numerisch falsche Eigenwerte sind schlechte Approximationen von exakten Eigenwerten.

## Physikalisch falsche Eigenwerte

Physikalisch falsche Eigenwerte sind Eigenwerte, die aufgrund von fehlerhaften Randbedingungen oder Unterrepresentation der Physik im Widerspruch zum Problem stehen.

# Physialisch falsche Eigenwerte

- tauchen oft auf, wenn ein System von Gleichungen zu einer Gleichung reduziert wird,

# Physialisch falsche Eigenwerte

- tauchen oft auf, wenn ein System von Gleichungen zu einer Gleichung reduziert wird,

Hilfreich:

- Diskretisierung des Systems,
- Reduktion der Diskretisierung zu einer Gleichung

# Warum sind diese falschen Eigenwerte problematisch?

- abschnittsweise Instabilität,
- instabile Modes finden wird schwerer,
-
-

# Warum sind diese falschen Eigenwerte problematisch?

- abschnittsweise Instabilität,
- instabile Modes finden wird schwerer,
- Verschlechterung der Konditionszahl,
-

# Warum sind diese falschen Eigenwerte problematisch?

- abschnittsweise Instabilität,
- instabile Modes finden wird schwerer,
- Verschlechterung der Konditionszahl,
- Jedoch: Werte wachsen meist mit $O(N^4)$ -> einfach als schlechte Eigenwerte aufzufinden

# Gliederung

# Konditionszahl

Mit Tschebyscheff oder Legendre Basis kommt es zu ungewollten Schwierigkeiten:

-
-

# Konditionszahl

Mit Tschebyscheff oder Legendre Basis kommt es zu ungewollten Schwierigkeiten:

- Wachstum der Konditionszahl: $O(N^{2p})$,
- -> für Gleichungen der sechsten Ordnung: (p=6): $O(N^{12})$

# Konditionszahl

Mit Tschebyscheff oder Legendre Basis kommt es zu ungewollten Schwierigkeiten:

- Wachstum der Konditionszahl: $O(N^{2p})$,
- -> für Gleichungen der sechsten Ordnung: (p=6): $O(N^{12})$

Lösung: durch geeignete Wahl der Basisfunktionen: $O(N^{p})$

# Einfachste Wahl an Basisfunktionen für p=2

Randbedingungen: $u(-1) = u(+1) = 0$

Basisfunktionen:
$\phi_{2j}(x) = T_{2j}(x) - T_0(x)$
$\phi_{2j-1}(x) = T_{2j-1}(x) - T_1(x)$
Wachstum der Konditionszahl: $O(N^4)$

# Heinrichs Basis für Probleme 2. Ordnung

Randbedingungen: $u(-1) = u(+1) = 0$

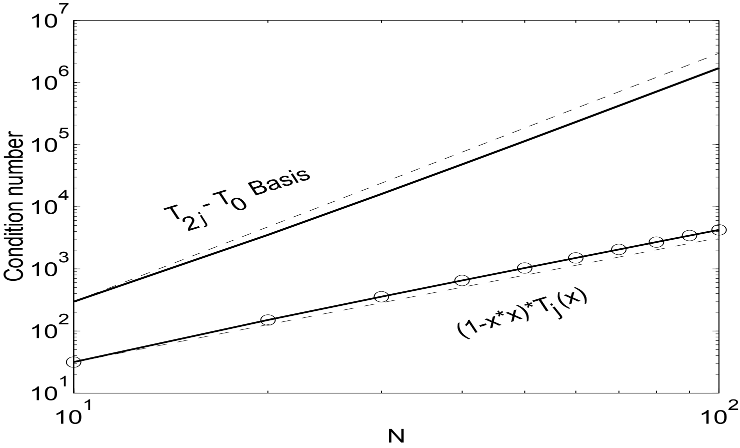
Basisfunktionen:
$\phi_j(x) = (1 - x^2)T_j(x)$
Wachstum der Konditionszahl: $O(N^4)$ -> $O(N^2)$

# Heinrichs Basis für Probleme 4. Ordnung

Randbedingungen: $u(-1) = u(+1) = u_x(-1) = u_x(+1) = 0$

Basisfunktionen:
$\phi_j(x) = (1 - x^2)^2 T_j(x)$
Wachstum der Konditionszahl: $O(N^8)$ -> $O(N^4)$

# Vergleich


10^7
10^6
10^5
10^4
10^3
10^2
10^1
Condition number
T_2j - T_0 Basis
(1-x*x)*T_j(x)
10^1
10^2
N

# Gliederung

# 1. Potenzmethode

- Idee: Wiederholtes Integrieren der Gleichung.
  -> Dominieren des größten Eigenwerts
-
-

# 1. Potenzmethode

- Idee: Wiederholtes Integrieren der Gleichung.
  -> Dominieren des größten Eigenwerts
- Problem: Dies ist sehr aufwendig.
-

# 1. Potenzmethode

- Idee: Wiederholtes Integrieren der Gleichung.
  -> Dominieren des größten Eigenwerts
- Problem: Dies ist sehr aufwendig.
- Lösung: Diskretisierung

# 1. Potenzmethode auf Matrizen

Matrixeigenwertproblem der Form: $Au = \lambda u$

1. Wahl eines beliebigen Vektors $u_0$

# 1. Potenzmethode auf Matrizen

Matrixeigenwertproblem der Form: $Au = \lambda u$

1. Wahl eines beliebigen Vektors $u_0$

   Wiederhole:

2. $u_{k+1} = Au_k$
3. $\lambda_{k+1} = ||u_{k+1}||$
4. $u_{k+1} = u_{k+1} / ||u_{k+1}||$

   bis $|\lambda_{k+1} - \lambda_k| < \epsilon$

# Vor- und Nachteile

Matrixeigenwertproblem der Form: $Au = \lambda u$

1. schneller, kostengünstiger Algorithmus
2. 
3. 
4. 

# Vor- und Nachteile

Matrixeigenwertproblem der Form: $Au = \lambda u$

1. schneller, kostengünstiger Algorithmus
2. funktioniert mit jeder Matrix $A$
3. 
4. 

# Vor- und Nachteile

Matrixeigenwertproblem der Form: $Au = \lambda u$

1. schneller, kostengünstiger Algorithmus
2. funktioniert mit jeder Matrix $A$
3. funktioniert auch für komplexe Eigenwerte mit:

$$\lambda_{k+1} = \frac{1}{N} \sum_{j=1}^{N} \frac{u_{k+1,j}}{u_{k,j}}$$

4.

# Vor- und Nachteile

Matrixeigenwertproblem der Form: $Au = \lambda u$

1. schneller, kostengünstiger Algorithmus
2. funktioniert mit jeder Matrix $A$
3. funktioniert auch für komplexe Eigenwerte mit:
$$\lambda_{k+1} = \frac{1}{N} \sum_{j=1}^{N} \frac{u_{k+1,j}}{u_{k,j}}$$
4. Berechnet nur den Eigenwert mit dem größten Betrag.

# 2. Inverse Potenzmethode

Idee:

- Definiere M= $(A - \Lambda I)^{-1}$ , $\Lambda$ gewählte konstante.
-
-

# 2. Inverse Potenzmethode

Idee:

- Definiere M= $(A - \Lambda I)^{-1}$ , $\Lambda$ gewählte konstante.
- Berechne betragsgrößten Eigenwert $\mu$,
-

# 2. Inverse Potenzmethode

Idee:

- Definiere M= $(A - \Lambda I)^{-1}$ , $\Lambda$ gewählte konstante.
- Berechne betragsgrößten Eigenwert $\mu$,
- es gilt: $\mu = \frac{1}{\lambda - \Lambda}$ , $\lambda$ Eigenwert von $A$, der $\Lambda$ am nächsten ist.

# 2. Inverse Potenzmethode

Idee:

- Definiere M= $(A - \Lambda I)^{-1}$ , $\Lambda$ gewählte konstante.
- Berechne betragsgrößten Eigenwert $\mu$,
- es gilt: $\mu = \frac{1}{\lambda - \Lambda}$ , $\lambda$ Eigenwert von $A$, der $\Lambda$ am nächsten ist.

-> durch Wahl von $\Lambda$ kann man jeden Eigenwert berechnen.

# 2. Inverse Potenzmethode

Matrixeigenwertproblem der Form: $Au = \lambda u$

1. Wahl eines beliebigen Vektors $u_0$

   Wiederhole:

2. Löse $(A - \Lambda_k I)u_{k+1} = u_k$
3. $\mu_{k+1} = \frac{1}{N} \sum_{j=1}^{N} \frac{u_{k+1,j}}{u_{k,j}}$
4. $\Lambda_{k+1} = \frac{1}{\mu_{k+1}} + \Lambda_k$
5. $u_{k+1} = u_{k+1}/||u_{k+1}||$

   bis $|\lambda_{k+1} - \lambda_k| < \epsilon$

# Vor- und Nachteile

1. durch Lösen der Matrixgleichung teurer,
2.
3.
4.

# Vor- und Nachteile

1. durch Lösen der Matrixgleichung teurer,
2. funktioniert mit jeder Matrix $A$,
3. funktioniert auch für komplexe Eigenwerte,
4. 

# Vor- und Nachteile

1. durch Lösen der Matrixgleichung teurer,
2. funktioniert mit jeder Matrix $A$,
3. funktioniert auch für komplexe Eigenwerte,
4. kann jeden Eigenwert berechnen, wenn gute Näherung vorhanden.

# Kombination der Verfahren

Beispiel:
$\alpha u_{xx} + \lambda u = 0; u(-1) = u(1) = 0$

# Kombination der Verfahren

Beispiel:
$\alpha u_{xx} + \lambda u = 0; u(-1) = u(1) = 0$

Idee: QR für geringe Anzahl $\alpha$, Potenzmethode für andere
z.B.: $\alpha \in \mathbb{N}_{<10}$ mit QR , $\alpha \in [0, 10] \backslash \mathbb{N}$ mit Potenzmethode

# häufige Fehler

# häufige Fehler

- Leichtgläubigkeit - Glauben, dass man genaue Werte hat.
-
-

# häufige Fehler

- Leichtgläubigkeit - Glauben, dass man genaue Werte hat.
- Nachlässigkeit - wichtige Eigenfunktionen fehlen
-

# häufige Fehler

- Leichtgläubigkeit - Glauben, dass man genaue Werte hat.
- Nachlässigkeit - wichtige Eigenfunktionen fehlen
- „Coding“ - Fehler im Programm

# Fazit

# Fazit

- leichteren Vergleich für verschiedene N
-
-
-

# Fazit

- leichteren Vergleich für verschiedene N
- Verkleinern der Konditionszahl verbessert Berechenbarkeit
-
-

# Fazit

- leichteren Vergleich für verschiedene N
- Verkleinern der Konditionszahl verbessert Berechenbarkeit
- Potenzmethoden ermöglichen schnellere Berechnung
- 

# Fazit

- leichteren Vergleich für verschiedene N
- Verkleinern der Konditionszahl verbessert Berechenbarkeit
- Potenzmethoden ermöglichen schnellere Berechnung
- -> Wahl größerer N möglich -> Berechnung von mehr und genaueren Eigenwerten

# Quellen

John P. Boyd: Chebyshev and Fourier Spectral Methods. Ann Arbor, Michigan, 2000

Grafiken:

- Seite 26: Figure 7.7 auf Seite 137
- Seite 41: Figure 7.9 auf Seite 143