Zusammenfassung WR

Samuel Brinkmann (Matr. 624568)

22. Januar 2023

Inhaltsverzeichnis

Kapitel 1

Vorlesung I

1.1 Einführung

Def.: Lösen von mathematisch formulierten Problemstellungen die physikalische Sachverhalte beschreiben mit dem Computer.

$$\text{Wissenschaftlicher Rechnen} \Longleftrightarrow \text{Rechnergestützte (Ingenieur-))Wissenschaft}$$
$$\text{Entwurf von Verfahren} \Longleftrightarrow \text{Verwendung dieser Verfahren}$$

Workflow:
physikalisches Problem
$\Longrightarrow$ mathematisches Modell
$\Longrightarrow$ numerisches Modell
$\Longrightarrow$ Lösungsmethode, Code, Rechner (Fokus in WR)

Problemstellungen:

1 Agentenbasierte Simulation

2 (Nicht-)Lineare Optimierung

3 Reduktion von Sensordaten

Beispiel Anwendungen:

1 Wetter und Klima

2 Computertomografie

3 Simulationen von Pandemien

1.2 Modellierung

Def.: Ein Modell ist ein vereinfachtes Abbild der Realität.
Phasen: (Beispiel Pandemie Simulation)

1 Abgrenzung: Irrelevantes ausschließen
(Lieblingsfarbe, Schuhgröße)

2 Reduktion: Irrelevantes ausschließen
(Kontaktmatrix statt positionsgetreue Nachverfolgung)

3 Dekomposition: Bildung einzelner Segmente
(Populationsgruppen nach Region, Alter, Impfstatus, ...)

4 Aggregation: Zusammenfassung von Segmenten
(nur jeden 1000. Menschen simulieren)

5 Abstraktion: Bildung von Klassen
(S (susceptible), I (infected), R (recovered))

1.3 mathematisches Modell zu numerischem Modell

Mathematisches Modell: System von PDEs
$\longrightarrow$ Gleichungen geben nur Kopplung der Variablen an und keine tatsächlichen Werte
Lösung: Ableitungen werden durch den Differenzenquotienten approximiert

$$\frac{\partial f}{\partial x}\Big|_{x=x_0} \longrightarrow \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

Diskretisierung: Konstruktion eines Gitternetzes, wobei jede Variable an ihre benachbarten Gitterpunkte gekoppelt ist. So wird aus dem System von PDEs ein System algebraischer Gleichungen (endlich viele).

Kapitel 2

Vorlesung II

2.1 Finite Differenzen in 2D

FD-Approximation der Wärmeleitung in 2D

Wärmeleitungsgleichung:

$$\Delta u = \frac{\partial u}{\partial t} \text{ mit } \Delta = \sum_{i=1}^{\#dim} \frac{\partial^2}{\partial x_i^2}$$

Randbedingungen:

a linker Rand besitzt konstanten Wert u_0

b rechter Rand besitzt konstanten Wert u_1

Approximation durch 2D-Gitter $(x_j, t_n)_{j=1,\dots,N}$ mit $x_0 = 0$ und $x_N = 1$:

$$\begin{aligned}
u_0^n &= u_0 \ \forall n \quad \textit{(Randbedingung)} \\
u_N^n &= u_1 \ \forall n \quad \textit{(Randbedingung)} \\
u_j^0 &= f(x_j) \text{ für } 1 < j < N \quad \textit{(Anfangswerte)} \\
\frac{\partial u}{\partial t} &\approx \frac{u_j^{n+1} - u_j^n}{\Delta t} \\
\frac{\partial^2 u}{\partial x^2} &\approx \frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{\Delta x^2} \quad \textit{(explizit)} \\
\frac{\partial^2 u}{\partial x^2} &\approx \frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{\Delta x^2} \quad \textit{(implizit)}
\end{aligned}$$

In PDE einsetzen und nach u_j^{n+1} umstellen!

5-Punkte-Stern für 2D

1 Randpunkte: Randbedingung

2 Innere Punkte:
a) Randfern: 5-Punkte-Stern
b) Randnah: Anpassung durch Einfügen der Randbedingung für Randknoten

3 LGS: $A_h u_h = q_h$ mit $q_h = -\hat{A}_h g + f$ (g Randwerte, f 5-Punkte-Stern)
$(\hat{A}_h)_{ij} = -\frac{1}{h^2}$, falls Knoten i randnah und j ein Nachbar mit 5-Punkte-Stern ist
$(\hat{A}_h)_{ij} = 0$, sonst
$f_{ij} = \frac{1}{h^2}(-u_{i,j-1} - u_{i-1,j} + 4u_{i,j} - u_{i+1,j} - u_{i,j+1})$

2.2 LinA Wiederholung

Laplace-Matrix für Graphen: L = Gradmatrix − Adjazenzmatrix
geometrische Bedeutung LGS: Schnittmenge von $\#dim$ Hyperebenen
Matrixnorm: $\|A\| = \max_{x \neq 0} \frac{\|Ax\|}{\|x\|}$
Kondition einer Matrix: $\text{cond}(A) = \|A\|\|A^{-1}\|$
Kondition einer sym. Matrix: $\text{cond}(A) = \frac{|\lambda_{max}|}{|\lambda_{min}|}$

Kapitel 3

Vorlesung III

3.1 Speicher

Zeitliche Lokalität: Adressbereiche, auf die zugegriffen wird, werden auch in naher Zukunft mit hoher Wahrscheinlichkeit wieder benutzt.
Beispiel: int sum

```
sum = 0
for i in range(1,n):
    sum += 1
```

Räumliche Lokalität: Nach einem Zugriff auf einen Adressbereich erfolgt mit hoher Wahrscheinlichkeit der nächste Zugriff auf eine Adresse in unmittelbarer Nachbarschaft.
Beispiel: list A

```
A = [...]
for i in range(len(A)):
    print(A[i])
```

ACHTUNG: Aufpassen, ob Arrays in *column-major order* oder *row-major order* im Speicher liegen (räumliche Lokalität).

3.2 Dichte Vektor-Matrix-Operationen

Skalarprodukt $a^{\mathrm{T}} \cdot b$: $\mathcal{O}(n)$
Skalarprodukt $a \cdot b^{\mathrm{T}}$: $\mathcal{O}(n)$
Matrix-Vektor-Produkt $A \cdot b$: $\mathcal{O}(n^2)$
Matrix-Matrix-Produkt $A \cdot B$: $\mathcal{O}(n^3)$ (Cache-Effizient durch Blockbasierte Abarbeitung)

3.3 Dünn besetzte Matrizen

CSR - compressed sparse row
3 Arrays der Datenstruktur:

- IR: Größe N+1, Index des Zeilenstarts in anderen Arrays
- JC: Größe nnz, Spaltenindex des Eintrags
- NUM: Größe nnz, Wert des Eintrags

Alterantiven: CSC, CSR mit Diagonalverschiebung (Diagonale in separatem Array)
Grundoperationen:
Zeilenindizierung $A[i]$: $\mathcal{O}(1)$
SpMV (sparse matrix vector product) $A \cdot b$: $\mathcal{O}(nnz)$
SpGEMM $A \cdot B$: $\mathcal{O}(nnz(A) + flops) \longrightarrow$ Sparse Accumulator (SPA)

Kapitel 4

Vorlesung IV

4.1 direkte Lösungsverfahren

Voraussetzung: A invertierbar, d.h. $\det(A) \neq 0$. (Sonst keine exakte Lösung)
Bespiele:

- LR-Zerlegung (allg. Matrizen)
- Cholesky-Zerlegung (spd. Matrizen)
- QR-Zerlegung (allg. Matrizen)

Vorteil: exakte Lösung nach Faktorisierung schnell
Nachteil: kubische Laufzeit, dichtere Zwischenergebnisse

4.2 Cholesky-Zerlegung

Voraussetzung: $A \in \mathbb{R}^{n\times n}$ ist symmetrisch und positiv definit.
Zerlegung: $A = LL^{\mathrm{T}}$ mit $L = (\ell_{ij}) \in \mathbb{R}^{n\times n}$ untere Dreiecksmatrix.
Verfahren:

$$\ell_{kk} = \sqrt{a_{kk} - \sum_{j=1}^{k-1} \ell_{kj}^2}$$
$$\ell_{ik} = \frac{1}{\ell_{kk}} \left(a_{ik} - \sum_{j=1}^{k-1} \ell_{ij} \cdot \ell_{kj} \right)$$

Laufzeit: $\mathcal{O}(n^3) = \mathcal{O}(\#Multiplikation + \#Division + \#Wurzeln)$

4.3 Iterative Lösungsverfahren

Raten einer Anfangslösung x_0 und dann iterative Verbessung dieser.
Mit M als Approximation von A^{-1}.

$$x^{(t+1)} = x^{(t)} - M\left(Ax^{(t)} - b\right)$$

Jacobi-Verfahren (Splitting-Verfahren)

Sei $A \in \mathbb{R}^{n\times n}$, $D = \operatorname{diag}(a_{11}, \ldots, a_{nn})$ und $Ax = b$ das zu lösende LGS. Dann gilt für die i-te $(i = 1, \ldots, n)$ Komponente der iterierten Lösung des nächsten Schrittes

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

Das Verfahren ist gut parallelisierbar, da es komponentenweise vorgeht und auf $x^{(t+1)}$ nur schreibend und $x^{(t)}$ lesend zugegriffen wird.

Kapitel 5

Vorlesung V

5.1 Konjugierte Gradienten (CG)

Voraussetzung: Matrix A ist symmetrisch und positiv definit.
Residuum: $r := b - Ax^{(t)}$
Verfahren: