Übungsblatt 3

Physikalisch-basierte Simulation und Animation

Dr.-Ing. Johannes S. Mueller-Roemer und Stephanie Ferreira, MSc

TECHNISCHE UNIVERSITÄT DARMSTADT

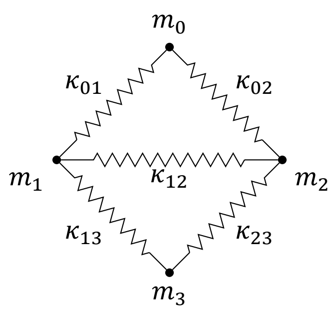
Sommersemester 2024
Übungsblatt 3

Anmerkung zur Bewertung

Durch das Lösen der unten genannten Aufgaben sind bis zu 11 Punkte erreichbar, jedoch werden je Übungsblatt maximal 10 Punkte gewertet. Durch das Bearbeiten aller Aufgaben können somit Fehler ausgeglichen werden. Bewertet werden Ergebnisse und Herleitungen.

Aufgabe 3.1: Masse-Feder-Systeme (2 Punkte)

Gegeben sei ein Masse-Feder-System mit vier Massen $m_0, \dots, m_3$ und fünf Federn κ_{01}, κ_{02}, κ_{12}, κ_{13} und κ_{23}:



Dieses System ist im Raum $\mathbb{R}^2$ eingebettet. Beschreiben Sie in Formeln das resultierende kontinuierliche Gleichungssystem sowie das mittels explizitem Euler-Verfahren zeitlich diskretisierte System.

Aufgabe 3.2: Kontinuumsmechanik (3 Punkte)

In der Kontinuumsmechanik ist die Momentankonfiguration gegeben durch eine Abbildung

$$\varphi : \Omega \subset \mathbb{R}^3 \mapsto \Omega' \subset \mathbb{R}^3, \varphi(\mathbf{x}) = \mathbf{x}' = (x_1', x_2', x_3')^T. \tag{1}$$

Bestimmen Sie das Verschiebungsfeld $\mathbf{u}(\mathbf{x}) = \mathbf{x}' - \mathbf{x}$, Deformationsgradient $\mathbf{F} = \nabla\varphi = \nabla\mathbf{u} + \mathbf{E}$, Cauchy-Green-Deformationstensor $\mathbf{C} = \mathbf{F}^T\mathbf{F}$, Green-Strain-Tensor $\varepsilon_G = \frac{1}{2}(\mathbf{C} - \mathbf{E})$ und den linearen Cauchy-Strain-Tensor $\varepsilon_C = \frac{1}{2}(\nabla\mathbf{u}^T + \nabla\mathbf{u})$ für die folgenden Momentankonfigurationen:

1. $\varphi_1(\mathbf{x}) = \mathbf{0}$
2. $\varphi_2(\mathbf{x}) = -\mathbf{x}$
3. $\varphi_3(\mathbf{x}) = \mathbf{x} + \mathbf{b}$, wobei $\mathbf{b} \in \mathbb{R}^3$ beliebig ist.

4. $\varphi_4(\mathbf{x}) = (x_1^2, x_2^2, x_3^2)^T$
5. $\varphi_5(\mathbf{x}) = (-x_2, x_3, x_1)^T$

Welche Probleme ergeben sich bei dem linearen Cauchy-Strain-Tensor? Wo könnte er dennoch potentiell Anwendungsbereiche haben?

Aufgabe 3.3: Conjugate Gradients

Eines der verbreitetsten Verfahren zur Lösung linearer Gleichungssysteme ist die Methode der konjugierten Gradienten. Es wird für eine symmetrische positiv definite Matrix $A \in \mathbb{R}^{n\times n}$ angewendet.

3.3a) Ohne Preconditioner (2 Punkte)

Implementieren Sie die Methode der konjugierten Gradienten in Python. Es soll eine Ausgabe der benötigten Iterationen und erreichten Konvergenz enthalten sein. Nutzen Sie dazu Scipy (`pip install scipy` oder per Download von numpy und scipy über `http://www.lfd.uci.edu/~gohlke/pythonlibs/` und Installation per: `pip install` *`dateiname`*). Referenz `https://docs.scipy.org/doc/scipy/reference/sparse.html`, insbesondere `coo`, `csr` und `dia`. Testen Sie das CG-Verfahren für die folgenden Matrizen:

- `offsets = numpy.array([0, -1, 1]); data = numpy.array([range(3, 36, 3), range(1, 12), range(0, 11)]); A = scipy.sparse.dia_matrix((data, offsets), shape=(11, 11))`
- `offsets = numpy.array([0, -1, 1]); data = numpy.array([range(1, 9), range(1, 9), range(0, 8)]); A = scipy.sparse.dia_matrix((data, offsets), shape=(8, 8))`
- `A = scipy.sparse.csr_matrix(scipy.linalg.toeplitz([3.0, -1] + [0] * (17 - 2)))` (Analog zu Aufgabe 3 Übungsblatt 1)

Für welche dieser Matrizen ist die Methode der konjugierten Gradienten geeignet? Mit `print(A.toarray())` können die Matrizen ausgegeben werden. Als rechte Seite der Gleichungssysteme soll `numpy.ones` verwendet werden.

3.3b) Preconditioner (4 Punkte)

Die Konvergenzgeschwindigkeit des CG-Verfahrens für ein Gleichungssystem

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{2}$$

hängt vor allem von der sogenannten Kondition

$$\kappa(\mathbf{A}) = ||\mathbf{A}||||\mathbf{A}^{-1}|| \tag{3}$$

der Matrix $\mathbf{A}$ ab. Zur Verbesserung der Konditionszahl werden daher sogenannte Vorkonditionierer (engl. Preconditioner) verwendet, welche die Matrix $\mathbf{A}$ besser ausbalancieren sollen. Beim Einsatz eines Vorkonditionierers wird statt dem ursprünglichen System, das angepasste vorkonditionierte System

$$\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b} \tag{4}$$

gelöst, welches die gleiche Lösung hat. Der Vorkonditionierer $\mathbf{M}$ ist so zu wählen, dass $\mathbf{M}^{-1}\mathbf{A}$ symmetrisch positiv definit ist und die Konditionszahl verbessert:

$$\kappa(\mathbf{M}^{-1}\mathbf{A}) < \kappa(\mathbf{A})$$

Statt das vorkonditionierten System explizit aufzustellen, ist es von Vorteil die CG-Methode entsprechend anzupassen. Dies resultiert in der „Preconditioned CG-Methode“:

$\mathbf{r}_0 \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}_0$
$\mathbf{z}_0 \leftarrow \mathbf{M}^{-1}\mathbf{r}_0$

$\mathbf{p}_0 \leftarrow \mathbf{z}_0$
$k \leftarrow 0$
loop
  $\alpha_k \leftarrow \frac{\mathbf{r}_k^\top \mathbf{z}_k}{\mathbf{p}_k^\top \mathbf{A}\mathbf{p}_k}$
  $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k \mathbf{p}_k$
  $\mathbf{r}_{k+1} \leftarrow \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{p}_k$
  if r_{k+1} is sufficiently small **then** exit loop
  end if
  $\mathbf{z}_{k+1} \leftarrow \mathbf{M}^{-1}\mathbf{r}_{k+1}$
  $\beta_k \leftarrow \frac{\mathbf{z}_{k+1}^\top \mathbf{r}_{k+1}}{\mathbf{z}_k^\top \mathbf{r}_k}$
  $\mathbf{p}_{k+1} \leftarrow \mathbf{z}_{k+1} + \beta_k \mathbf{p}_k$
  $k := k + 1$
end loop
The result is $\mathbf{x_{k+1}}$.

Es sollen folgende Vorkonditionierer betrachtet werden:
Sei $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{L}^T$ eine Zerlegung (nicht zu verwechseln mit der multiplikativen $\mathbf{LDL^T}$-Zerlegung!) von $\mathbf{A}$ mit $\mathbf{D} = \text{diag}(\mathbf{A})$ und $\mathbf{L}$ einer strikt unteren Dreiecksmatrix.

- Der SSOR-Preconditioner ist gegeben durch die Matrix $\mathbf{M} = \frac{\omega}{2-\omega}(\frac{1}{\omega}\mathbf{D} + \mathbf{L})\mathbf{D}^{-1}(\frac{1}{\omega}\mathbf{D} + \mathbf{L}^T)$
- Der Jacobi-Preconditioner ist gegeben durch die Matrix $\mathbf{M} = \mathbf{D}$ und kann als Variante von SSOR mit ω gegen 0 interpretiert werden.
- Der Gauß-Seidel-Preconditioner entspricht dem SSOR-Preconditioner mit $\omega = 1$ ist gegeben durch die Matrix $\mathbf{M} = (\mathbf{D} + \mathbf{L})\mathbf{D}^{-1}(\mathbf{D} + \mathbf{L}^T)$

Warum existiert $\mathbf{M}^{-1}$ für diese Wahlen von $\mathbf{M}$?

Implementieren Sie die Preconditioned CG-Methode mit den drei genannten Preconditionern und lösen Sie die in 3.3a) gegebenen Gleichungssysteme. Alle genannten Preconditioner können und sollen effizient ohne $\mathbf{M}$ oder $\mathbf{M}^{-1}$ explizit aufzustellen berechnet werden. Vergleichen Sie die benötigten Anzahl an Iterationen.