Methodische und Praktische Grundlagen der Informatik 4

Aufgabenblatt 1

mpgi4@cg.tu-berlin.de

WiSe 2013/2014

Allgemeine Hinweise:

- Die Aufgaben sollen in Gruppen bestehend aus zwei bis drei Personen bearbeitet werden. Ausnahmen müssen mit dem jeweiligen Tutor abgesprochen werden.
- Bitte reichen Sie Ihre Lösungen in Form einer ZIP Datei bis **Donnerstag, den 20.11.2014, um 12:00 Uhr** auf der ISIS Webseite der Vorlesung ein. Tragen Sie in jede abgegebene Datei Name und Email-Adresse aller Gruppenmitglieder ein.
- Wenn eine Aufgabe die Abgabe einer Grafik verlangt, dann muss ein vollständig funktionsfähiges Programm in der Lösung enthalten sein, welches bei der Ausführungen die Grafik erstellt.
- Verwenden Sie den vorgegebenen Skelettcode, um die Aufgaben umzusetzen.

Aufgabe 1: Gaußsche Eliminationsmethode mit Pivoting (3 Punkte)

Implementieren Sie das Gaußsche Eliminationsverfahren inklusive Rückwärtseinsetzen sowohl mit als auch ohne Pivoting in Python. Eingabe der Funktionen sind ein NumPy Array und ein Vektor. Für den Fall, dass es keine eindeutige Lösung gibt, soll ein Fehler ausgegeben werden. Die von Ihnen zu implementierenden Funktionen sollen folgende Signaturen haben:

```
def gaussianElimination(M, b, use_pivoting = True):
    """
    Gaussian Elimination with Pivoting.

    Inputs:
    M :: matrix representing linear system (numpy.array)
    b :: vector, representing right hand side (numpy.array)
    use_pivoting :: flag if pivoting should be used or no flag if pivoting should not be used

    Return:
    M :: input matrix in row echelon form (numpy.array)
    b :: input vector with pivoting permutations applied (numpy.array)
    """

def backSubstitution(M, b):
     """
    Back substitution for the solution of a linear system in row echelon form.

    Inputs:
    M :: matrix in row echelon representing linear system (numpy.array)
    b :: vector, representing right hand side (numpy.array)

    Returns:
    x :: solution of linear system (numpy.array)
    """
```

Testen Sie Ihre Implementierung an $Ax = b$ für

$$A = \begin{pmatrix} 11 & 44 & 1 \\ 0.1 & 0.4 & 3 \\ 0 & 1 & -1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

und vergleichen Sie Ihre Ergebnisse mit der exakten Lösung $x = (-1732/329, 438/329, 109/329)^T$.

Aufgabe 2: LR-Zerlegung (3 Punkte)

a) Implementieren Sie die LR-Zerlegung einer Matrix mit Pivoting in Python. Ihre Funktion soll folgende Signatur besitzen:

```
def LUDecomposition(M):
    """
    LU decomposition of matrix M.

    Inputs:
    M :: matrix to be decomposed (numpy.array)

    Return:
    P :: Permutation matrix of pivoting transformations (numpy.array)
    L :: L matrix of LU decomposition, lower triangular (numpy.array)
    U :: U matrix of LU decomposition, upper triangular (numpy.array)
    """
```

b) Implementieren Sie eine Funktion, welche für eine gegebene LR-Zerlegung $A = LR$ das zugehörige Gleichungssystem $Ax = b$ für eine konkrete rechte Seite b löst.

```
def solveLU( P, L, U, b):
    """
      LU decomposition of matrix M.

      Inputs:
      P :: Permutation matrix of pivoting transformations (numpy.array)
      L :: lower triangular matrix of LU (numpy.array)
      U :: upper triangular matrix of LU (numpy.array)
      b :: right hand side of linear system (numpy.array)

      Return:
      x :: solution of linear system represented by L,U,b (numpy.array)
    """
```

Aufgabe 3: Lineare Ausgleichsrechnung (3 Punkte)

Bei der Computertomographie soll die Dichteverteilung innerhalb eines Objektes bestimmen werden, ohne das Objekt zu zerschneiden. Das Objekt wird dazu mit einer Vielzahl von Strahlen $\{S_k\}_{k\in K}$ durchleuchtet (siehe Abbildung 1) und für jeden Strahl S_k wird der Intensitätsverlust bestimmt, welcher beim Durchdringen des Objektes auftritt. Aus den Intensitätsverlusten lässt sich dann die Dichteverteilung im Objekt rekonstruieren.

Grundlage der Rekonstruktion ist die folgende Gleichung, welche Dichte $\varrho(x)$ und Intensitätsverlust $I_0^{(k)}/I_1^{(k)}$ in Zusammenhang setzt:

$$\log\left(\frac{I_0^{(k)}}{I_1^{(k)}}\right) = \int_{S_k} \varrho(s)\,\mathrm{d}s. \tag{1}$$

Die rechte Seite der Gleichung bestimmt den Intensitätsverlust entlang des Strahles S_k durch das Integral über die Dichte, welche proportional zur Absorption ist. Die linke Seite ist der logarithmierte Intensitätsverlust zwischen Anfang und Ende eines Strahls. Die Intensitäten $I_0^{(k)}$ und $I_1^{(k)}$ sind experimentell bestimmbare Größen, was die Voraussetzung für die Rekonstruktion der Dichte $\varrho(x)$ im Objekt ist. Gleichung 1 bekommt eine besonders einfache Form, wenn die Dichte innerhalb des Objektes stückweise konstant ist (wie z.B. in Abbildung 2(a)). Ein Strahl S_k lässt sich dann in mehrere Segmente $\gamma_1 \dots \gamma_n$ mit Länge $|\gamma_i|$ unterteilen, über welchen die Dichte $\varrho(x)$ einen konstanten Wert ϱ_i hat. Für Gleichung 1 gilt dann

$$\log\left(\frac{I_0^{(k)}}{I_1^{(k)}}\right) = \int_{S_k} \varrho(s)\,\mathrm{d}s = \sum_{i=1}^{n} \int_{\gamma_i} \varrho(s)\,\mathrm{d}s = \sum_{i=1}^{n} |\gamma_i|\,\varrho_i. \tag{2}$$

Diese Vereinfachung wird bei der Computertomographie ausgenutzt. Um die Dichteverteilung innerhalb des Objektes näherungsweise zu bestimmen, wird ein Raster über das Objekt gelegt und für jeden Quadranten des Rasters angenommen, dass die Dichte darin konstant ist. Es bezeichne Q_{ij} den Quadranten

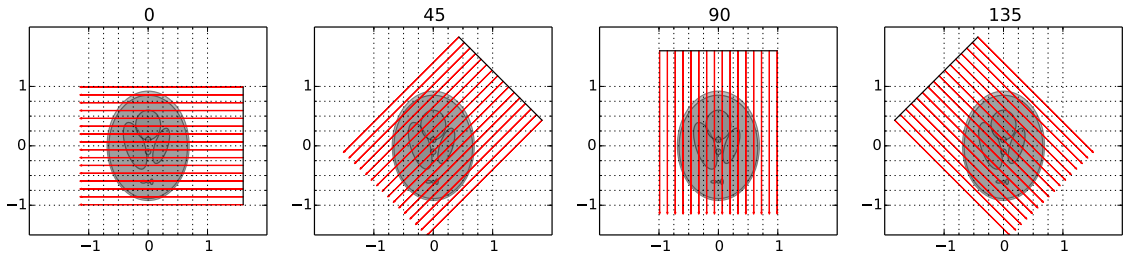


Abbildung 1: Bei der Computertomographie werden aus verschiedenen Winkeln eine Reihe paralleler Röntgenstrahlen ausgesendet und der Intensitätsverlust der Strahlen gemessen, welcher beim Durchdringen des Objektes auftritt.

in der i-ten Zeile und j-ten Spalte eines $n \times n$-Rasters, ϱ_{ij} die Dichte im Quadranten Q_{ij} und $L_{ij}^{(k)}$ die Länge des Schnittes des Strahls S_k mit dem Quadranten Q_{ij}. Aus Gleichung 2 erhalten wir dann:

$$V_k = \log\left(\frac{I_0}{I_1}\right) = \sum_{i,j=1}^{n} \int_{Q_{ij} \cap S_k} \varrho_{ij} \, \mathrm{d}s = \sum_{i,j} L_{ij}^{(k)} \varrho_{ij}. \tag{3}$$

Gleichung 3 ist ein lineares Gleichungssystem mit einer Gleichung für jeden Strahl und den Dichten der Quadranten als Unbekannten. In Matrixnotation, $Ax = b$, lässt sich dieses Gleichungssystem folgendermaßen schreiben:

$$\underbrace{\begin{pmatrix} L_{11}^{(1)} & L_{12}^{(1)} & \cdots & L_{1n}^{(1)} & L_{21}^{(1)} & L_{22}^{(1)} & \cdots & L_{nn}^{(1)} \\ L_{11}^{(2)} & L_{12}^{(2)} & \cdots & L_{1n}^{(2)} & L_{21}^{(2)} & L_{22}^{(2)} & \cdots & L_{nn}^{(2)} \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ L_{11}^{(K)} & L_{12}^{(K)} & \cdots & L_{1n}^{(K)} & L_{21}^{(K)} & L_{22}^{(K)} & \cdots & L_{nn}^{(K)} \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} \varrho_{11} \\ \varrho_{12} \\ \vdots \\ \varrho_{1n} \\ \varrho_{21} \\ \varrho_{22} \\ \vdots \\ \varrho_{nn} \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} V_1 \\ V_2 \\ \\ \vdots \\ \\ V_K \end{pmatrix}}_{b}, \tag{4}$$

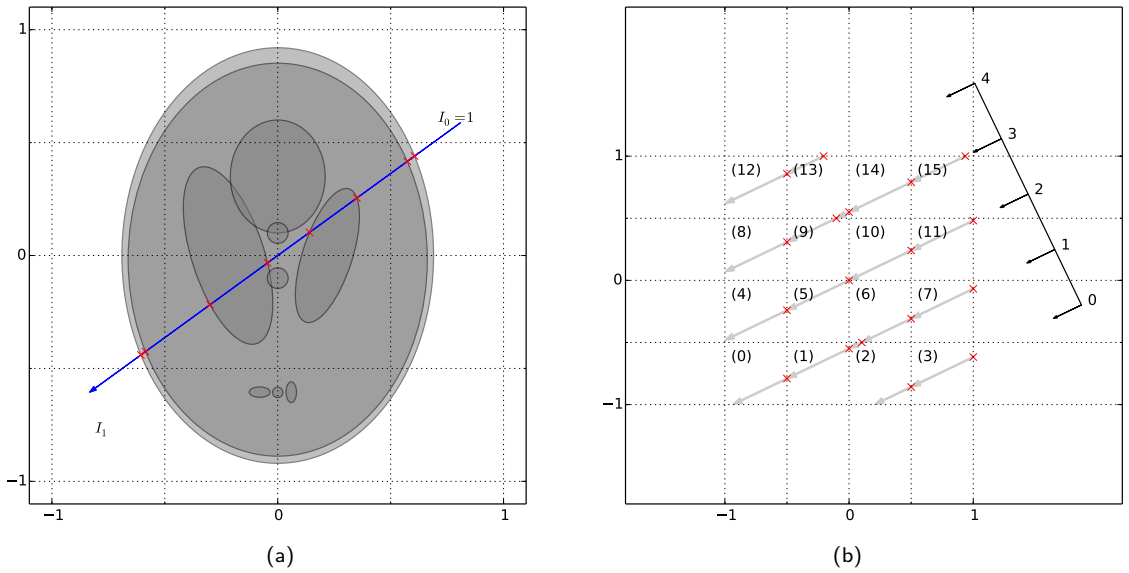
mit $A \in \mathbb{R}^{K \times n^2}$, $x \in \mathbb{R}^{n^2}$ und $b \in \mathbb{R}^K$. In der Regel sind sinnvolle Ergebnisse nur zu erwarten, wenn die Anzahl der Gleichungen K (d.h. die Anzahl der Strahlen) größer ist als die Anzahl n^2 der Quadranten des Rasters. Es handelt sich dann um ein überbestimmtes Gleichungssystem, welches mittels Methoden der Ausgleichsrechnung gelöst werden muss.

a) Bei der Computertomographie werden nicht beliebige Strahlen verwendet, sondern eine Reihe von Aufnahmen mit parallelen Strahlen aus verschiedenen Winkeln gemacht (Abbildung 1). Eine Matrix, welche in jeder Spalte die für eine Aufnahme ermittelten Intensitäten $I_1^{(k)}$ enthält, heißt *Sinogramm* (Abbildung 3).

   Implementieren Sie die Berechnung eines Sinnogramms in Python. Die Eingabe ist die Anzahl der Strahlen pro Aufnahme und die Anzahl der Aufnahmen. Plazieren Sie die Aufnahmen ($\varphi \in [0, 180°)$) uniform entlang des oberen Halbkreises. Verwenden Sie dazu die in `tomograph.py` bereitgestellte Funktion `trace(r,d)` (Abbildung 2(a)). Als Eingangsintensität wird dabei $I_0 = 1$ angenommen. Erstellen Sie Sinogramme mit der Auflösung 64×64 und 128×128. Visualisieren Sie diese mit der Funktion `imshow` von `matplotlib` und reichen Sie die Visualisierungen in Form einer PNG Datei ein.

b) Implementieren Sie eine Python Funktion, welche die in Gleichung 4 definierte Matrix A für eine gegebene Anzahl von Strahlen K und ein Raster der Größe $n \times n$ berechnet. Verwenden Sie dazu die in `tomograph.py` bereitgestellte Funktion `grid_intersect(n,r,d)` (Abbildung 2(b)), um die Längen der Strahlen in einem Quadranten zu bestimmen.

c) Bestimmen Sie die Dichten für die in (a) berechneten Sinogramme durch Lösen des Ausgleichsproblems $A^T A x = A^T b$. Nutzen Sie hierfür Ihre Implementierung des Gaußschen Eliminationsverfahrens aus Aufgabe 1. Visualisieren Sie die ermittelten Dichten mit der Funktion `imshow` von `matplotlib` und reichen Sie die Visualisierungen in Form einer PNG Datei ein.



(a) (b)

Abbildung 2: (a) Die Funktion `trace(r,d)` berechnet für einen Strahl die Intensität, welche der Strahl nach dem Durchdringen des Testobjektes hat. (b) Die Funktion `grid_intersect(n,r,d)` berechnet den Schnitt eines Rasters mit einer Menge von parallelen Strahlen.

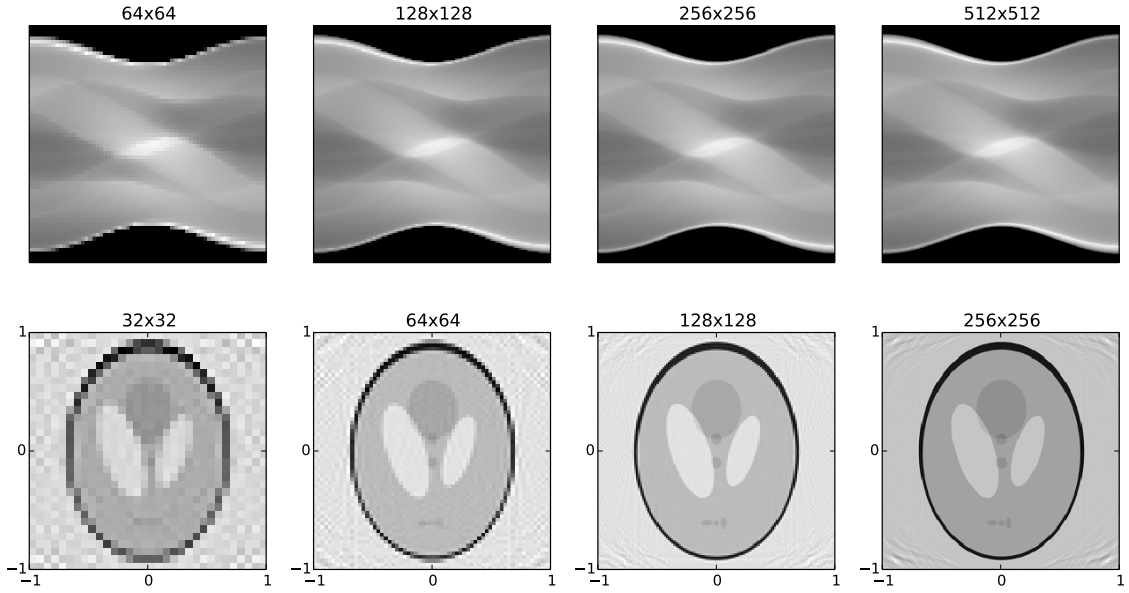


Abbildung 3: Die obere Zeile zeigt Sinogramme welche mit der `trace` Funktion aus `tomography.py` erstellt wurden. Die unter Zeile zeigt die rekonstruierten Dichten.