Serie 2

Debugging und LR-Zerlegung mit Pivotisierung

Aufgabe 2.1 (2 Punkte)**:** In dieser Aufgabe beschäftigen wir uns mit dem in MATLAB eingebauten Debugger. Lies dir die Beilage sorgfältig durch und benutze den MATLAB-Debugger, um das Skript `bug_script.m` von der Webseite zu debuggen. Das Skript sollte die Matrix

$$\mathbf{C} = \begin{bmatrix} -1 & 1 & -1 & 17 & -1 & 3 & 16 \\ 3 & 53 & -1 & 3 & 1 & 41 & 9 \\ -1 & -1 & -1 & 17 & -1 & 5 & 4 \\ -1 & -1 & -1 & 17 & -1 & -1 & 1 \end{bmatrix}$$

erzeugen. Welche Fehler hast du gefunden?

Aufgabe 2.2 (1 + 3 + 0 + 0 Punkte)**:** Wir versuchen nun, ein etwas schwierigeres Programm zu debuggen. Dazu benötigen wir ein wenig Theorie:

Gesucht ist eine Funktion $u(x)\colon [0,1] \to \mathbb{R}$, für welche

$$\begin{cases} -u''(x) = f(x), & 0 < x < 1 \\ \quad u(0) = u(1) = 0, & \text{(Randbedingungen)} \end{cases} \tag{$*$}$$

für eine gegebene Funktion $f(x)$ gilt. Das Problem $(*)$ ist eine sogenannte *Randwertaufgabe* (RWA), in diesem Fall das *eindimensionale Poisson-Problem.*

Um eine Approximation der Lösung zu erhalten, gehen wir folgendermassen vor: Sei $h = \frac{1}{n}$ für ein $n \in \mathbb{N}$. Wir betrachten die Stützstellen $x_i = ih$ für $i = 0, 1, \ldots, n$. Mit u_i bezeichnen wir die Approximation von u an der Stelle x_i. (Es gilt also $u_i \approx u(x_i)$). Nun ersetzen wir die zweite Ableitung $u''(x)$ durch den Differenzenquotienten $D_h^2 u(x)$ (siehe Beilage) und betrachten die Gleichung nur an den Punkten x_i. Wir erhalten die Gleichungen

$$-\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} = f(x_i), \quad i = 1, 2, \ldots, n-1.$$

Es fehlen noch die Gleichungen an den Punkten x_0 und x_n. Diese folgen aus den Randbedingungen:

$$u_0 = u(0) = 0 \text{ und } u_n = u(1) = 0.$$

Zusammengefasst erhalten wir also das Gleichungssystem

$$\begin{array}{ccccccccccl} u_0 & & & & & & & = & 0 & & (0) \\ -u_0 & + & 2u_1 & - & u_2 & & & = & h^2 f(x_1) & & (1) \\ & & -u_1 & + & 2u_2 & - & u_3 & = & h^2 f(x_2) & & (2) \\ & & & \ddots & & \ddots & & \ddots & \vdots & & \\ & & & & -u_{n-2} & + & 2u_{n-1} & - \; u_n = & h^2 f(x_{n-1}) & & (n-1) \\ & & & & & & u_n & = & 0 & & (n) \end{array}$$

Eine Lösung dieses Gleichungssystems ist eine Approximation der Lösung von $(*)$.

a) Schreibe (auf Papier) diese Gleichungen als lineares Gleichungssystem der Form

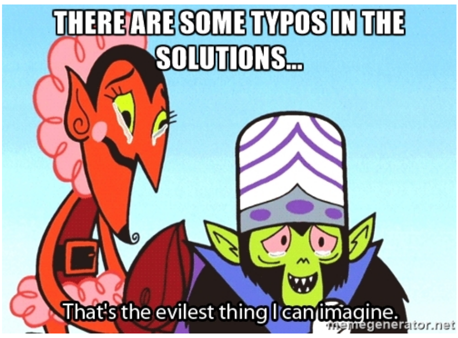
$$\mathbf{A}\mathbf{u} = \mathbf{f}.$$

Beachte, dass $\mathbf{u}$ und $\mathbf{f}$ Vektoren der Länge $n+1$ sind! Wie sieht die $(n+1) \times (n+1)$-Matrix $\mathbf{A}$ aus? Bestimme auch die Bandbreite der Matrix $\mathbf{A}$.

b) Lade von der Website das Skript `main.m` und die Funktion `RWA_func.m` herunter.

Das Skript `main.m` definiert $f(x) = \pi^2 \sin(\pi x)$, ruft die Funktion `RWA_func.m` auf, die das obige Gleichungssystem für f lösen sollte, und plottet die exakte sowie die approximierte Lösung in ein Bild.

Die Funktion `[x, u] = RWA_func(n, f_handle)` in `RWA_func.m` nimmt eine Zahl `n` und ein function-handle `f_handle` als Input. Damit sollte die Funktion das dazugehörige Gleichungssystem (siehe oben) aufstellen und dessen Lösung im Vektor `u` speichern. Zusätzlich sollte sie die Stützstellen x_i im Vektor `x` ausgeben. Sollte.



Allerdings enthält das Programm `RWA_func.m` einige Fehler. Debugge es mit dem MATLAB-Debugger. Gib `doc spdiags` im Command Window ein, um zu sehen, wie der Befehl `spdiags` funktioniert. Während des Debuggens ist ausserdem der Befehl `full(A)`, der alle Einträge der sparse-Matrix $\mathbf{A}$ anzeigt, hilfreich.

Führe das Skript `main.m` aus, um zu sehen, ob die richtige Lösung herauskommt. Dafür vergleiche die exakte und die approximierte Lösung. Wenn beide Kurven an den Stützstellen x_i übereinstimmen, sollte das Programm `RWA_func.m` korrekt funktionieren.

c) *(freiwillig)* Die exakte Lösung von $(*)$ ist in unserem Fall $u(x) = \sin(\pi x)$. Rechne dies nach. Wenn du das Programm erfolgreich debugged hast, erweitere `main.m` so, dass es die Lösung für $n = 2^k$ mit $k = 2, 3, \ldots, 9$ berechnet. Zeichne den maximalen Fehler

$$e(h) = \max_{i=0,\ldots,n} |u(x_i) - u_i|$$

bezüglich h mit dem Befehl `loglog` auf einer loglog-Skala. Bestimme die Ordnung p des Verfahrens und zeichne in den Plot eine entsprechende Vergleichsgerade, das heisst, finde ein $p \in \mathbb{N}$ und ein $c \in \mathbb{R}$, so dass $e(h) \approx ch^p$ gilt. Ist das der Fall, schreiben wir auch $e(h) = O(h^p)$.

d) *(freiwillig)* Berechne die numerische Lösung von $(*)$ für

$$f(x) = \begin{cases} -1 & \text{für } 0.5 < x < 0.75, \\ 0 & \text{sonst,} \end{cases}$$

und $h = 2^{-8}$. Zeichne nun die Funktion u, sowie ihre erste und zweite Ableitung. Benutze dazu die Funktion `differentiate` von der Website. Erkläre deine Ergebnisse.

Aufgabe 2.3 (3 + 1 Punkte)**:** Letzte Woche haben wir gesehen, dass die LR-Zerlegung mit dem Gauss-Algorithmus nicht funktioniert, wenn die zu zerlegende Matrix an bestimmten Stellen Nulleinträge hat. Dieses Problem lässt sich beheben, wenn der Spaltenpivotstrategie gefolgt wird: Wir speichern alle Einträge der LR-Zerlegung in der Matrix $\mathbf{A}$ selbst ab. Sind wir in der k-ten Spalte von $\mathbf{A}$ angekommen, schauen wir, in welcher der Zeilen von k bis n der betragsmässig grösste Eintrag dieser k-ten Spalte steht, und vertauschen diese Zeile mit der k-ten Zeile. Die Vertauschung wird in einer Permutation p eingetragen. Danach fahren wir mit dem ursprünglichen Algorithmus der LR-Zerlegung fort. Der Algorithmus für die LR-Zerlegung mit Spaltenpivotstrategie ist dann (Pseudocode):

Setze $p = (1, \dots, n)$
for $k = 1$ to $n-1$ **do**
  Bestimme m so, dass $|a_{m,k}| = \max_{k \le i \le n} |a_{i,k}|$
  Vertausche die Einträge $p(m)$ und $p(k)$
  Vertausche die m-te und die k-te Zeile von $\mathbf{A}$
  for $i = k+1$ to n **do**
    Setze $a_{i,k} = \frac{a_{i,k}}{a_{k,k}}$
    Setze $a_{i,(k+1,\dots,n)} = a_{i,(k+1,\dots,n)} - a_{i,k} a_{k,(k+1,\dots,n)}$
  end for
end for
Setze $\mathbf{L} = \mathbf{E}$ + untere Dreieckshälfte von $\mathbf{A}$
Setze $\mathbf{R}$ = Hauptdiagonale von $\mathbf{A}$ + obere Dreieckshälfte von $\mathbf{A}$
Setze $\mathbf{P}$ = die zu p gehörende Permutationsmatrix

Hinweis: Um m zu erhalten, kannst du den Befehl `max` mit zwei Outputargumenten aufrufen: `[x_max, ind_max] = max(x)`. Ausserdem kannst du `tril` und `triu` benutzen, um die untere bzw. obere Dreieckshälfte von $\mathbf{A}$ zu bekommen. Für mehr Informationen gib `doc tril` bzw. `doc triu` im Command Window ein. Für das Aufstellen der Permutationsmatrix $\mathbf{P}$ kannst du deine Funktion `per_mat` von letzter Woche verwenden.

Mit der Spaltenpivotstrategie gilt nicht mehr $\mathbf{A} = \mathbf{LR}$, sondern

$$\mathbf{A} = \mathbf{PLR} \quad \iff \quad \mathbf{P}^\top \mathbf{A} = \mathbf{LR}$$

aufgrund der Orthogonalität $\mathbf{P}^\top \mathbf{P} = \mathbf{E}$ (siehe Beilage 1). Ausserdem lösen wir nicht mehr das Gleichungssystem $\mathbf{Ax} = \mathbf{b}$, sondern

$$\mathbf{P}^\top \mathbf{A} \mathbf{x} = \mathbf{P}^\top \mathbf{b}.$$

In der Vorwärtssubstitution müssen wir daher $\mathbf{b}$ mit $\mathbf{P}^\top \mathbf{b}$ ersetzen.

a) Schreibe nun eine neue MATLAB-Funktion `function x = solve_lr_pivot(A, b)`, um die Lösung des linearen Gleichungssystems

$$\mathbf{Ax} = \mathbf{b}$$

mit Hilfe der pivotisierten LR-Zerlegung von $\mathbf{A}$ zu berechnen. Du kannst dabei den Pseudocode zur Vorwärtssubstitution aus der Beilage 1 zu Hilfe nehmen und die Vorwärts- und Rückwärtssubstitution von Blatt 1 verwenden, wenn du deren Anpassungen an Bandmatrizen entfernst.

b) Benutze `solve_lr_pivot`, um die Systeme

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix} \quad \text{und} \quad \begin{pmatrix} 2 & 1 & 3 & 5 \\ 4 & 2 & 1 & 6 \\ 3 & 7 & 2 & 1 \\ 1 & 3 & 5 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 49 \\ 55 \\ 37 \\ 53 \end{pmatrix}$$

zu lösen. Teste deine Programme durch Berechnung des Residuums $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$. Die Einträge des Residuumvektors $\mathbf{r}$ müssen kleiner als 10^{-10} sein.

Allgemeine Informationen befinden sich auf der Website.
Zuletzt editiert am 8. März 2024.