OEFENINGEN NUMERIEKE WISKUNDE

OEFENZITTING 14 (PC): ITERATIEVE METHODEN VOOR HET OPLOSSEN VAN STELSELS

Voor het uitwerken van de opgaven moet je de `.m`-bestanden gebruiken die je kan vinden op Toledo

1. Lineaire stelsels

Stel $A = L + D + U$ met $\forall i : a_{ii} \neq 0$. Hierbij is L de onderdriehoeksmatrix, D de diagonaalmatrix en U de bovendriehoeksmatrix. De iteratieformule

$$Mx^{(k+1)} = Nx^{(k)} + B$$

voor de opsplitsingen van $A = M - N$ bepaald de iteratieve methodes van

Jacobi: $M = D$, $N = -(L + U)$;

Gauss-Seidel: $M = L + D$, $N = -U$.

Probleem 1. (Jacobi vs Gauss-Seidel) Gebruik de methodes van Jacobi en Gauss-Seidel, geïmplementeerd in resp. `jacobi.m` en `gs.m`, om het volgende stelsel op te lossen. Gebruik als startwaarden $x_1 = x_2 = x_3 = 1$.

$$\begin{cases} 4x_1 - x_2 = 1 \\ -x_1 + 4x_2 - x_3 = 0 \\ -x_2 + 4x_3 = 1 \end{cases}$$

Ga na welke methode het snelst convergeert en vergelijk dit met de spectraalradius van de iteratiematrix. Verklaring van de parameters van de Matlab-routines:

`a`	de matrix van het stelsel
`b`	het rechterlid van het stelsel
`x0`	een vector met de startwaarden
`xe`	de exacte oplossing, indien je een meer betrouwbaar beeld van de fout wil dan zou volgen uit de residus (Verklaar!)
`n`	het aantal iteratiestappen

Ga in de code zelf na wat de twee outputparameters betekenen.

Probleem 2. (Gevoeligheid voor initiële condities) Doe dertig Jacobi-iteraties om het volgende stelsel op te lossen:

$$\begin{cases} 3x_1 - 2x_2 + 2x_3 = 3 \\ -2x_1 + 3x_2 - 2x_3 = -1 \\ 2x_1 - 2x_2 + 3x_3 = 3 \end{cases}$$

Neem als startwaarden $\begin{bmatrix} 3 & 2 & 0 \end{bmatrix}^T$ en $\begin{bmatrix} 3.0001 & 2 & 0 \end{bmatrix}^T$.

Verklaar wat er gebeurt.

(Hint: Stelt men $\varepsilon^{(k)} = x^{(k)} - x^*$, de k-de iteratiefout, dan ziet men eenvoudig in dat $\varepsilon^{(k)} = M^{-1}N\varepsilon^{(k-1)}$. Gebruik de eigenstructuur van de iteratiematrix om het gedrag van de fouten te verklaren.)

2. Niet-lineaire stelsels

Probleem 3. (Newton-Raphson methode) Beschouw het stelsel $\mathbf{f}(\mathbf{x}) = \mathbf{0}$:

$$\begin{cases} f_1(x_1, x_2) = x_1^2 + x_2^2 - 1 = 0, \\ f_2(x_1, x_2) = \sin(x_1 \cdot \frac{\pi}{2}) + x_2^3 = 0. \end{cases}$$

(1) Schrijf een Matlab-functie met als invoer een vector `x` en als uitvoer de waarde van de functie $\mathbf{f}(\mathbf{x})$ in dit punt `x`. De hoofding zou zijn: `function y = F(x)`.
(2) Schrijf een Matlab-functie met als invoer een vector `x` en als uitvoer de waarde van de Jacobiaan `JF` van de functie $\mathbf{f}(\mathbf{x})$ berekend in het punt `x`. De hoofding zou zijn: `function y = JF(x)`.
(3) Schrijf een Matlab-functie met als invoer
- `F`: functie die het stelsel bepaalt (afkomstig van de stap (1))
- `JF`: functie die de Jacobiaan bepaalt (afkomstig van de stap (2))
- `x0`: startwaarde
- `tol`: tolerantie om te stoppen (norm van de correctie $\|\Delta\mathbf{x}\|$)
- `nmax`: max aantal iteraties

en als uitvoer
- `x`: oplossing
- `R`: residu berekend in het punt `x`
- `n`: feitelijk aantal iteraties

De hoofding zou zijn: `function [x, R, n] = newtonsys(F, JF, x0, tol, nmax)`.
(4) Gebruik de startvector `x0 = [1; 1]`, zet de tolerantie `tol = 1e-5`, `nmax = 10` en los het stelsel $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ op m.b.v. de functie `newtonsys.m`.

Probleem 4. (Complexe nulpunten - I) Implementeer de methode van Newton-Raphson om de complexe nulpunten van onderstaande vergelijkingen te vinden. Beschouw een complex getal $z = x + iy$ als twee variabelen en genereer een stelsel.

$$2z^2 + z + 1 = 0, \tag{1}$$
$$z - e^z = 0. \tag{2}$$

Probleem 5*. (Complexe nulpunten - II) Wanneer de complexe functie $f : \mathbb{C} \to \mathbb{C}$ analytisch is, zal de iteratieformule

$$z^{(k+1)} = z^{(k)} - \frac{f(z^{(k)})}{f'(z^{(k)})}, \quad z^{(0)} \in \mathbb{C} \tag{3}$$

dezelfde eigenschappen behouden als de Newton-Raphson formule voor reële nulpunten. Formule (3) kan direct geïmplementeerd worden omdat Matlab complex rekent. Zoek de complexe wortels van de functies uit de vorige opgaven met behulp van deze methode. Merk wel dat je daarbij alleen naar een complex nulpunt kan convergeren indien de startwaarde complex is. Bewijs m.b.v. de Cauchy-Riemann voorwaarden dat beide methoden in feite identiek zijn.