Ecole Polytechnique Fédérale de Lausanne
Analyse numérique (GM) / MATH-251(e)

Alexandre Caboussat
Printemps 2025

Corrigé 6

Exercice 1

1.a) On peut écrire le problème sous la forme $\vec{F}(\vec{x}) = \vec{0}$ où $\vec{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ et

$$\vec{F}(\vec{x}) = \begin{pmatrix} F_1(x_1, x_2, \ldots, x_N) \\ F_2(x_1, x_2, \ldots, x_N) \end{pmatrix} = \begin{pmatrix} (1 + (x_1 - x_2)^2)2x_1 - x_2 - 1 \\ -x_1 + (1 + (x_2 - x_1)^2)2x_2 - 1 \end{pmatrix} = 0.$$

1.b) Soit $\vec{y} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \in \mathbb{R}^2$. On a

$$D\vec{F}(\vec{y}) = \begin{pmatrix} 2 + 2(y_1 - y_2)(3y_1 - y_2) & -1 - 4y_1(y_1 - y_2) \\ -1 - 4y_2(y_2 - y_1) & 2 + 2(y_2 - y_1)(3y_2 - y_1) \end{pmatrix}.$$

1.c) Le premier pas de la méthode de Newton s'écrit: étant donné $\vec{x}_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ trouve $\vec{x}^1$ tel que

$$D\vec{F}(\vec{x}^0)(\vec{x}^0 - \vec{x}^1) = \vec{F}(\vec{x}^0).$$

En pratique, on résoud le système linéaire $D\vec{F}(\vec{x}^0)\vec{y} = \vec{F}(\vec{x}^0)$ où $\vec{y} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ et on pose $\vec{x}^1 = \vec{x}^0 - \vec{y}$. En effectuant le calcul avec $\vec{x}^0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, on obtient:

$$\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} -1 \\ -1 \end{pmatrix},$$

soit

$$\vec{y} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} -1 \\ -1 \end{pmatrix},$$

et donc

$$\vec{x}^1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Pour trouver $\vec{x}^2$, on résoud le système linéaire $D\vec{F}(\vec{x}^1)\vec{y} = \vec{F}(\vec{x}^1)$ et on pose $\vec{x}^2 = \vec{x}^1 - \vec{y}$. Puisque $\vec{F}(\vec{x}^1) = 0$, alors $\vec{y} = \vec{0}$, et donc

$$\vec{x}^2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \vec{x}^1,$$

qui est bien une solution du problème.

Exercice 2

2.a) En écrivant le système linéaire sous la forme $\vec{F}(\vec{x}) = \vec{0}$, où $\vec{F} : \mathbb{R}^N \to \mathbb{R}^N$ est une fonction vectorielle, on obtient :

$$\vec{F}(\vec{x}) \;=\; \begin{pmatrix} F_1(x_1, x_2, \ldots, x_N) \\ F_2(x_1, x_2, \ldots, x_N) \\ \vdots \\ F_N(x_1, x_2, \ldots, x_N) \end{pmatrix},$$

les fonctions F_i étant définies par :

$$F_i(x_1, x_2, \ldots, x_N) = -x_{i-1} + 2x_i - x_{i+1} + e^{x_i} \qquad i = 1, 2, \ldots, N.$$

2.b) La matrice jacobienne associée au système non-linéaire s'écrit :

$$D\vec{F}(\vec{x}) = \begin{pmatrix} \dfrac{\partial F_1(\vec{x})}{\partial x_1} & \dfrac{\partial F_1(\vec{x})}{\partial x_2} & \cdots & \dfrac{\partial F_1(\vec{x})}{\partial x_N} \\ \dfrac{\partial F_2(\vec{x})}{\partial x_1} & \dfrac{\partial F_2(\vec{x})}{\partial x_2} & \cdots & \dfrac{\partial F_2(\vec{x})}{\partial x_N} \\ \vdots & \vdots & & \vdots \\ \dfrac{\partial F_N(\vec{x})}{\partial x_1} & \dfrac{\partial F_N(\vec{x})}{\partial x_2} & \cdots & \dfrac{\partial F_N(\vec{x})}{\partial x_N} \end{pmatrix}$$

$$= \begin{pmatrix} 2+e^{x_1} & -1 & & & & \\ -1 & 2+e^{x_2} & -1 & & & 0 \\ & -1 & 2+e^{x_3} & -1 & & \\ & & \ddots & \ddots & \ddots & \\ 0 & & & \ddots & \ddots & -1 \\ & & & & -1 & 2+e^{x_N} \end{pmatrix}.$$

2.c) Pour que l'algorithme soit toujours bien défini, il est nécessaire qu'à chaque pas le système linéaire $A\vec{y} = \vec{b}$ avec $A = D\vec{F}(\vec{x}^n)$, $\vec{b} = \vec{F}(\vec{x}^n)$ puisse être résolu. Ceci est possible si la matrice A est symétrique définie positive donc régulière. Pour $N = 2$ la matrice jacobienne s'écrit :

$$A \;=\; D\vec{F}(\vec{x}) \;=\; \begin{pmatrix} 2+e^{x_1} & -1 \\ -1 & 2+e^{x_2} \end{pmatrix}.$$

On a clairement $A = A^T$ et, pour tout $\vec{y} = (y_1, y_2) \in \mathbb{R}^2$,

$$\vec{y}^T A\vec{y} = (1+e^{x_1})y_1^2 + (1+e^{x_2})y_2^2 + (y_1 - y_2)^2 \geqslant 0.$$

Donc $\vec{y}^T A\vec{y} = 0$ si et seulement si $\vec{y} = 0$.

2.d) Les lignes de l'algorithme, une fois complétées, sont les suivantes :

```
function [x] = newtsys(N, max_iter = 1000)
%
% Methode de Newton : Etant donne x^n, trouver x^{n+1}
% tel que DF(x^n)(x^n - x^{n+1}) = F(x^n)
```

```
% En pratique on construit A=DF(x^n), b=F(x^n)
% on resoud Ay=b et on pose x^{n+1}=x^n-y
%
% parametres
% ------------------------------------------------------
%  N         : nombre d inconnues du systeme non lineaire
%  x         : N-vecteur, contient x^n puis x^{n+1}

    x = ones(N,1);

    stop = 1;
    iter = 0;
    while stop>1e-10 && iter < max_iter
        iter = iter + 1;

        A = jacobian(x);
        b = func(x);

        y = A \ b;
        x = x - y;

        stop = norm(y) / norm(x);
        fprintf('iter=%i, stop = %e \n',iter,stop)
    end
end
```

```
function [df] = jacobian(x)
    N = length(x);
    df = zeros(N,N);

    df(1,1) = 2 + exp(x(1));
    df(1,2) = -1;
    df(N,N) = 2 + exp(x(N));
    df(N,N-1) = -1;

    for i = 2 : N-1
        df(i,i) = 2 + exp(x(i));
        df(i, i-1) = -1;
        df(i, i+1) = -1;
    end

end
```

```
function [FF] = func(x)
    N = length(x);
    FF = zeros(N,1);

    FF(1) = 2*x(1) + exp(x(1)) - x(2);
    FF(N) = 2*x(N) + exp(x(N)) - x(N-1);
    for i = 2 : N-1
        FF(i) = 2*x(i) + exp(x(i)) - x(i-1) - x(i+1);
    end

end
```

2.e) Dans la méthode de Newton-corde, la matrice jacobienne est évaluée une unique fois lors de l'initialisation.

```
function [x] = newtcorde(N, max_iter = 1000)
%
```

```
% Methode de Newton-Corde : Etant donne x^n, trouver x^{n+1}
% tel que DF(x^0)(x^n - x^{n+1}) = F(x^n)
% En pratique on construit A=DF(x^0), b=F(x^n)
% on resoud Ay=b et on pose x^{n+1}=x^n-y
%
% parametres
% ------------------------------------------------------
%  N        : nombre d inconnues du systeme non lineaire
%  x        : N-vecteur, contient x^n puis x^{n+1}

    x = ones(N,1);

    stop = 1;
    iter = 0;
    A = jacobian(x);

    while stop>1e-10 && iter < max_iter
        iter = iter + 1;

        b = func(x);

        y = A \ b;
        x = x - y;

        stop = norm(y) / norm(x);
        fprintf('iter=%i, stop = %e \n',iter,stop)
    end
end
```

2.f) La vitesse de convergence peut-être obtenue en regardant la discrépance. C'est également cette grandeur que l'on peut utiliser comme critère d'arrêt pour les algorithmes de résolution, elle est définie par :

$$\frac{\left\|x^{n+1} - x^n\right\|}{\|x^{n+1}\|}.$$

Le tableau ci-dessous présente les résultats obtenus pour $N = 5$ points et $n = 5$ itérations de la méthode de Newton et de la méthode de Newton-corde. L'algorithme de Newton converge plus rapidement que celui de Newton-corde; la précision atteinte après 5 itérations étant beaucoup plus grande.

	Méthode de Newton					
n	x_1^n	x_2^n	x_3^n	x_4^n	x_5^n	$\frac{\|\mathbf{x}^{n+1}-\mathbf{x}^n\|}{\|\mathbf{x}^{n+1}\|}$
0	1.0000	1.0000	1.0000	1.0000	1.0000	
1	0.0000	0.0000	0.0000	0.0000	0.0000	5.0706e+31
2	-0.6111	-0.8333	-0.8888	-0.8333	-0.6111	1.0000e+00
3	-0.8185	-1.2070	-1.3232	-1.2070	-0.8185	9.2272e-02
4	-0.8418	-1.2529	-1.3787	-1.2529	-0.8418	1.2969e-03
5	-0.8421	-1.2535	-1.3793	-1.2535	-0.8421	**1.8307e-07**

	Méthode de Newton-corde					
n	y_1^n	y_2^n	y_3^n	y_4^n	y_5^n	$\frac{\|\|\mathbf{y}^{n+1}-\mathbf{y}^n\|\|}{\|\|\mathbf{y}^{n+1}\|\|}$
0	1.0000	1.0000	1.0000	1.0000	1.0000	
1	0.0000	0.0000	0.0000	0.0000	0.0000	5.0706e+31
2	-0.2858	-0.3487	-0.3597	-0.3487	-0.2858	1.0000e+00
3	-0.4450	-0.5714	-0.5973	-0.5714	-0.4450	1.4669e-01
4	-0.5464	-0.7277	-0.7692	-0.7277	-0.5464	4.4029e-02
5	-0.616	-0.8426	-0.8985	-0.8426	-0.6161	**1.7693e-02**

Exercice 3

3.a) Le calcul donne, écrit par composante:

$$\frac{\partial}{\partial y_k}\mathcal{L}(\vec{y}) = \left(A\vec{y} - \vec{b}\right)_k + (y_k)^3 \quad k = 1,\ \ldots,\ N.$$

3.b) Puisque $\vec{x}$ est le minimum de $\mathcal{L}$, sa dérivée doit être nulle, donc satisfaire $\vec{grad}\mathcal{L}(\vec{x}) = \vec{0}$, ce qui correspond à :

$$A\vec{x} - \vec{b} + \begin{pmatrix} (x_1)^3 \\ \vdots \\ (x_N)^3 \end{pmatrix} = \vec{0}$$

3.c) Étant donné $\vec{x}^0$, $n = 0,\ 1,\ 2,\ \ldots$, la méthode de Newton s'écrit, pour trouver $\vec{x}^{n+1}$ à partir de $\vec{x}^n$:

$$DF(\vec{x}^n)\left(\vec{x}^{n+1} - \vec{x}^n\right) = -\vec{F}(\vec{x}^n) \tag{1}$$

On a explicitement les composantes de la matrice jacobienne, avec $i, j = 1,\ \ldots,\ N$:

$$(DF(\vec{y}))_{ij} = \frac{\partial F_i}{\partial y_j}(\vec{y}) = \frac{\partial}{\partial y_j}\left((A\vec{y} - \vec{b})_i + (y_i)^3\right) = \frac{\partial}{\partial y_j}\left(\sum_{k=1}^{N} A_{ik}y_k - b_i + (y_i)^3\right) = A_{ij} + 3(y_j)^2\delta_{ij},$$

où on a noté

$$\delta_{ij} = \begin{cases} 1 \text{ si } i = j, \\ 0 \text{ si } i \neq j \end{cases}.$$

La symétrie est évidente. Reste à prouver la positivité. Pour tout $\vec{y},\ \vec{z} \in \mathbb{R}^N$ on a:

$$\vec{z}^T(DF(\vec{y}))\vec{z} = \vec{z}^T A\vec{z} + \vec{z}^T \begin{pmatrix} 3(y_1)^2 & & (0) \\ & \ddots & \\ (0) & & 3(y_N)^2 \end{pmatrix} \vec{z} = \vec{z}^T A\vec{z} + 3\sum_{k}^{N}(y_k)^2(z_k)^2 \geqslant 0.$$

On a aussi que $\vec{z}^T(DF(\vec{y}))\vec{z} = 0 \implies \vec{z}^T A\vec{z} = 0 \implies \vec{z} = \vec{0}$. Donc la matrice jacobienne $DF(\vec{y})$ est bien symétrique définie positive pour tout $\vec{y} \in \mathbb{R}^N$.