Application de la méthode RRE aux problèmes de Navier-Stokes

Sébastien DUMINIL

L.M.P.A
Université du Littoral - Calais

en collaboration avec H. Sadok et D. Silvester
SMAI 2011 Guidel - 27 Mai 2011

Plan

Considérons un système d'équations linéaire ou non linéaire :

$$F(x) = 0; F : \mathbf{R}^N \to \mathbf{R}^N$$

Notons s la solution de cette équation. On construit une suite (s_n) telle que $\lim_{n\to\infty} s_n = s$.

$$s_{n+1} = G(s_n), n = 0, 1, \cdots$$

où $G(x) = x - F(x)$.

But de l'extrapolation

Trouver une suite (t_n) qui converge vers s plus rapidement que (s_n).

On pose :

- $\Delta s_j = s_{j+1} - s_j,\ j = 0, 1, \cdots, k$
- $\Delta^2 s_j = \Delta s_{j+1} - \Delta s_j,\ n = 0, 1, \cdots, j$
- $\Delta^i S_k = \left[\Delta^i s_0, \cdots, \Delta^i s_{k-1}\right]$

La méthode RRE (Reduced Rank extrapolation) génère une suite t_k définie par :

$$t_k = \sum_{j=0}^{k} \gamma_j^{(k)} s_j, \tag{1}$$

où

$$\sum_{j=0}^{k} \gamma_j^{(k)} = 1 \text{ and } \sum_{j=0}^{k} \eta_{ij}\gamma_j^{(k)} = 0, \quad i = 0, \cdots, k-1 \tag{2}$$

avec η_{ij} définie par

$$\eta_{ij} = (\Delta^2 s_i, \Delta s_j)$$

t_k peut être exprimé comme le quotient de deux déterminants :

$$t_k = \frac{\begin{vmatrix} s_0 & s_1 & \cdots & s_k \\ \eta_{0,0} & \eta_{0,1} & \cdots & \eta_{0,k} \\ \vdots & \vdots & \vdots & \vdots \\ \eta_{k-1,0} & \eta_{k-1,1} & \cdots & \eta_{k-1,k} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \cdots & 1 \\ \eta_{0,0} & \eta_{0,1} & \cdots & \eta_{0,k} \\ \vdots & \vdots & \vdots & \vdots \\ \eta_{k-1,0} & \eta_{k-1,1} & \cdots & \eta_{k-1,k} \end{vmatrix}}$$

En utilisant la formule de Schur, on peut exprimer t_k sous la forme suivante :

méthode RRE

$$t_k = s_k - \Delta S_k \Delta^2 S_k^+ \Delta s_k, \tag{3}$$

où $\Delta^2 S_k^+$ est le pseudo-inverse de $\Delta^2 S_k$ définie par

$$\Delta^2 S_k^+ = (\Delta^2 S_k^T \Delta^2 S_k)^{-1} \Delta^2 S_k^T.$$

On remarque que t_k existe et est unique si et seulement si $\det(\Delta^2 S_k^T \Delta^2 S_k) \neq 0$.

Si on considère une suite définie par
$s_n = s + \lambda_1^n v_1 + \lambda_2^n v_2 + \cdots + \lambda_k^n v_k + \cdots + \lambda_m^n v_m$, où
$0 \leq |\lambda_m| \leq \cdots \leq |\lambda_1|$ et $|\lambda_{k+1}| < |\lambda_k|$ alors,

$$t_k = s + O((\lambda_{k+1})^n)$$

Implémentation de RRE

Soit P la matrice de taille $k \times k$ définie par :

$$P = \begin{pmatrix} 1 & & \\ \vdots & \ddots & \\ 1 & \cdots & 1 \end{pmatrix}, \text{ alors } P^{-1} = \begin{pmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{pmatrix}$$

On a

$$t_k = s_k - \Delta S_k P P^{-1} \Delta^2 S_k^+ \Delta s_k. \qquad (4)$$

De plus, par construction :

$$s_k = s_0 + \Delta S_k \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}.$$

Posons $\gamma^{(k)} = P^{-1}\Delta^2 S_k^+ \Delta s_k$, dans (4), on obtient une formule simple :

$$t_k = s_0 + \Delta S_k \alpha^{(k)}, \quad k = 0, 1, \ldots \quad \text{with } \alpha^{(k)} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} - P\gamma^{(k)}. \tag{5}$$

Supposons que ΔS_k soit de rang maximal c'est à dire $\text{rang}(\Delta S_k) = k + 1$. On peut appliquer la décomposition QR de la matrice ΔS_k : $\Delta S_k = Q_k R_k$, où $Q_k \in \mathbf{R}^{N\times(k+1)}$ est unitaire et $R_k \in \mathbf{R}^{(k+1)\times(k+1)}$ est une matrice triangulaire supérieure avec les éléments diagonaux strictement positifs.

Algorithme de la méthode RRE

1. Step 0.
   - Input : Les vecteurs $s_0, s_1, \cdots, s_{k+1}$
2. Step 1.
   - Calculer $\Delta s_i = s_{i+1} - s_i, \quad i = 0, 1, \cdots, k$.
   - Poser $\Delta S_k = \left[\, \Delta s_0 \,|\, \Delta s_1 \,|\cdots|\, \Delta s_k \,\right]$,
   - Calculer la décomposition QR de ΔS_k, et nommer $\Delta S_k = Q_k R_k$.
3. Step 2.
   - Résoudre le système linéaire : $\quad R_k^T R_k d = e$;
   - $d = [d_0, d_1, \cdots, d_k]^T$, $e = [1, 1, \cdots, 1]^T$.
   - Poser $\lambda = \sum_{i=0}^{k} d_i$.
   - Poser $\gamma_i = (1/\lambda)\, d_i$ pour $i = 0, \cdots, k$.
4. Step 3.
   - Calculer $\alpha = [\alpha_0, \alpha_1, \cdots, \alpha_{k-1}]^T$ via (5)
   - Calculer t_k via $t_k = s_0 + Q_{k-1}(R_{k-1}\alpha)$.

Méthode RRE redémarrée

1. Step 0.
   - Input : set $k = 0$, Choisir un entier m et un vecteur s_0.
2. Step 1.
   - Construire $s_{j+1} = G(s_j), \quad j = 0, \cdots, m$.
3. Step 2.
   - Calculer t_m en utilisant l'algorithme précédent.
4. Step 3.
   - Si t_m est satisfaisant, stop.
   - Sinon, poser $s_0 = t_m$, $k = k + 1$ et retour à Step 2.

Equations de Navier-Stokes

Objectif : Résoudre ce système

$$\begin{cases} u.\nabla u - \nu \nabla^2 u + \nabla p = f \text{ in } \Omega \\ \nabla . u = 0 \text{ in } \partial\Omega \end{cases}$$

où ν est une constante donnée, avec les conditions aux bords suivantes :

$$u = w \text{ on } \partial\Omega_D,\ \nu \frac{\partial u}{\partial n} - \vec{n} p = 0 \text{ on } \partial\Omega_N$$

Formulation faible

Trouver $u \in H^1_E$ et $p \in L_2(\Omega)$ tels que :

$$\nu \int_\Omega \nabla u : \nabla v + \int_\Omega (u \cdot \nabla u) \cdot v - \int_\Omega p(\nabla \cdot v) = \int_\Omega f \cdot v \quad \forall v \in H^1_{E_0} \quad (6)$$

$$\int_\Omega q(\nabla \cdot u) = 0 \quad \forall q \in L_2(\Omega). \quad (7)$$

Nous rappelons deux procédures classiques de linéarisation :

- La méthode de Picard
- La méthode de Newton

Résidu non linéaire à l'itération k

$$R_k(v) = \int_\Omega f \cdot v - c(u_k; u_k, v) - \nu \int_\Omega \nabla u_k : \nabla v + \int_\Omega p_k (\nabla \cdot v), \quad (8)$$

$$r_k(q) = - \int_\Omega q(\nabla \cdot u_k) \quad (9)$$

pour tout $v \in H^1_{E_0}$ et $q \in L_2(\Omega)$.

c est un terme de convection. C'est une forme trilinéaire $c : H^1_{E_0} \times H^1_{E_0} \times H^1_{E_0} \rightarrow \mathbf{R}$ définit comme suit :

$$c(z; u, v) := \int_\Omega (z \cdot \nabla u) \cdot v. \quad (10)$$

Posons $u = u_k + \delta u_k$ et $p = p_k + \delta p_k$, on voit que $\delta u_k \in H^1_{E_0}, \delta p_k \in L_2(\Omega)$ satisfont :

$$\begin{aligned} D(u_k, \delta u_k, v) + \nu \int_\Omega \nabla \delta u_k : \nabla v - \int_\Omega \delta p_k (\nabla \cdot v) &= R_k(v) \\ \int_\Omega q(\nabla \cdot \delta u_k) &= r_k(q) \end{aligned} \tag{11}$$

pour tout $v \in H^1_{E_0}, q \in L_2(\Omega)$, où $D(u_k, \delta u_k, v)$ représente la différence des termes non linéaires :

$$D(u_k, \delta u_k, v) = c(\delta u_k; \delta u_k, v) + c(\delta u_k; u_k, v) + c(u_k; \delta u_k, v). \tag{12}$$

Méthode de Newton

Pour tout $v \in H^1_{E_0}$ et $q \in L_2(\Omega)$, trouver $\delta u_k \in H^1_{E_0}$ et $\delta p_k \in L_2(\Omega)$ satisfaisant

$$c(u_k; \delta u_k, v) + c(\delta u_k; u_k, v) + \nu \int_\Omega \nabla \delta u_k : \nabla v - \int_\Omega \delta p_k (\nabla \cdot v) = R_k(v)$$

$$\int_\Omega q(\nabla \cdot \delta u_k) = r_k(q).$$

Méthode de Picard

Pour tout $v \in H^1_{E_0}$ et $q \in L_2(\Omega)$, trouver $\delta u_k \in H^1_{E_0}$ et $\delta p_k \in L_2(\Omega)$ satisfaisant

$$\begin{aligned} c(u_k; \delta u_k, v) + \nu \int_\Omega \nabla \delta u_k : \nabla v - \int_\Omega \delta p_k (\nabla \cdot v) &= R_k(v) \\ \int_\Omega q(\nabla \cdot \delta u_k) &= r_k(q). \end{aligned}$$

Dans les deux cas, on pose $u_{k+1} = u_k + \delta u_k$, $p_{k+1} = p_k + \delta p_k$ pour définir l'itéré suivant.

Formulation faible discrète

Soient $X_0^h \subset H^1_{E_0}$ et $M^h \subset L_2(\Omega)$ des espaces de dimensions finies. La version discrète de (6)–(7) est : trouver $u_h \in X_E^h$ et $p \in M^h$ tels que

$$\nu \int_\Omega \nabla u_h : \nabla v_h + \int_\Omega (u_h \cdot \nabla u_h).v_h - \int_\Omega p_h(\nabla \cdot v_h) = \int_\Omega f \cdot v_h \quad \forall v_h \in X_0^h, \tag{13}$$

$$\int_\Omega q_h(\nabla \cdot u_h) = 0 \quad \forall q_h \in M^h. \tag{14}$$

Méthode de Newton : Trouver $\delta u_h \in X_0^h$ et $\delta p_h \in M^h$ satisfaisant

$$c(u_h; \delta u_h, v_h) + c(\delta u_h; u_h, v_h) + \nu \int_\Omega \nabla \delta u_h : \nabla v_h - \int_\Omega \delta p_h (\nabla \cdot v_h) = R_k(v_h)$$

$$\int_\Omega q_h (\nabla \cdot \delta u_h) = r_k(q_h).$$

Méthode de Picard : Trouver $\delta u_h \in X_0^h$ et $\delta p_h \in M^h$ satisfaisant

$$\begin{aligned} c(u_h; \delta u_h, v_h) + \nu \int_\Omega \nabla \delta u_h : \nabla v_h - \int_\Omega \delta p_h (\nabla \cdot v_h) &= R_k(v_h), \\ \int_\Omega q_h (\nabla \cdot \delta u_h) &= r_k(q_h), \end{aligned}$$

pour tout $u_h \in X_0^h$ et $q_h \in M^h$.

Pour se ramener à un problème d'algèbre linéaire, on introduit des fonctions de bases $\{\phi_j\}$ et $\{\psi_k\}$ définis par :

$$u_h = \sum_{j=1}^{n_u} u_j\phi_j + \sum_{j=n_u+1}^{n_u+n_\delta} u_j\phi_j, \qquad \delta u_h = \sum_{j=1}^{n_u} \Delta u_j\phi_j. \tag{15}$$

$$p_h = \sum_{k=1}^{n_p} p_k\psi_k, \qquad \delta p_h = \sum_{k=1}^{n_p} \Delta p_k\psi_k. \tag{16}$$

Picard :

$$\begin{pmatrix} \nu A + N & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \Delta u \\ \Delta p \end{pmatrix} = \begin{pmatrix} R \\ r \end{pmatrix} \tag{17}$$

Newton :

$$\begin{pmatrix} \nu A + N + W & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \Delta u \\ \Delta p \end{pmatrix} = \begin{pmatrix} R \\ r \end{pmatrix} \tag{18}$$

où

- $A_{ij} = (\nabla \phi_i, \nabla \phi_j)$ diffusion
- $B_{ij} = -(\nabla \phi_j, \psi_i)$ divergence
- $N_{ij} = (u_h . \nabla \phi_i, \phi_j)$ convection
- $W_{ij} = (\phi_i . \nabla u_h, \phi_j)$ Newton

Algorithme de Newton

1. Step 1.
   - Input : the vector $x_0 = (u_0, p_0)^T$,
   - compute the vector $(R_0, r_0)^T$ and set $k = 0$.
2. Step 2.
   - Retrieve A, B. Compute N_k,W_k R_k and r_k.
   - Solve the system (18) to find the vector $\delta x_k = (\delta u_k, \delta p_k)^T$.
   - Set $x_{k+1} = x_k + \delta x_k$.
   - Compute the vector $(R_{k+1}, r_{k+1})^T$.
3. Step 3.
   - If $\left\|(R_{k+1}, r_{k+1})^T\right\| < tol$, stop.
   - Otherwise set $k = k + 1$ and go to Step 2.

Algorithme de Picard

1. Step 1.
   - Input : the vector $x_0 = (u_0, p_0)^T$,
   - compute the vector $(R_0, r_0)^T$ and set $k = 0$.
2. Step 2.
   - Retrieve A, B. Compute N_k, R_k and r_k.
   - Solve the system (17) to find the vector $\delta x_k = (\delta u_k, \delta p_k)^T$.
   - Set $x_{k+1} = x_k + \delta x_k$.
   - Compute the vector $(R_{k+1}, r_{k+1})^T$.
3. Step 3.
   - If $\left\|(R_{k+1}, r_{k+1})^T\right\| < tol$, stop.
   - Otherwise set $k = k + 1$ and go to Step 2.

Résultats Numériques

- Ecoulement d'un fluide dans un conduit contenant un obstacle
- Méthode de Newton a une convergence locale
- $\Rightarrow$ quelques itérations de Picard avant d'appliquer Newton
- On applique la méthode RRE aux itérations de Picard.
- Pour les résultats numériques, on utilise le package IFISS (Incompressible Flow Iterative Solution Software).

Caractéristiques :

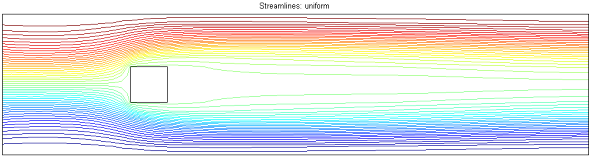
- On pose la longueur du conduit égale à 8
- On suppose un écoulement de Poiseuille à l'entrée du conduit

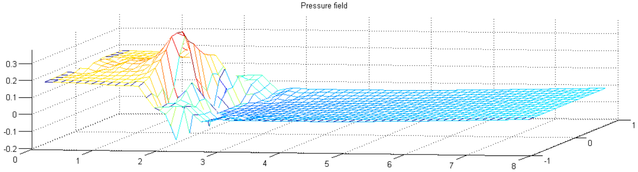
$$x = 0;\ -1 \leq y \leq 1$$

- Vitesse nulle sur le bord du conduit
- On suppose des conditions de Neumann à la sortie du conduit $\Rightarrow p = 0$

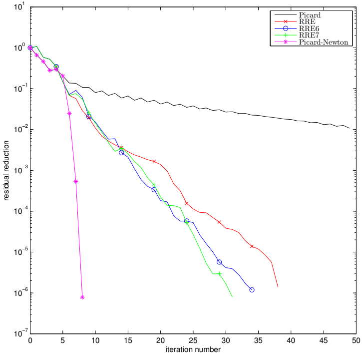
$$x = 8;\ -1 \leq y \leq 1$$

Solution pour $\nu = 1/600$

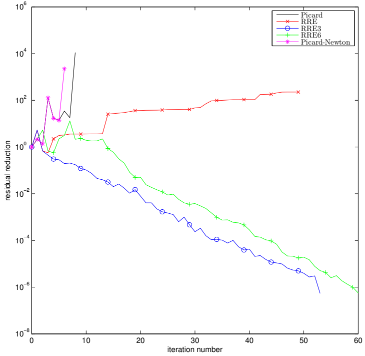




Convergence nonlinéaire pour $\nu = 1/600$



Convergence nonlinéaire pour $\nu = 1/800$



Merci de votre attention

Caractéristiques :

- On pose la longueur du conduit égale à 40.
- On suppose un écoulement de Poiseuille à l'entrée du conduit

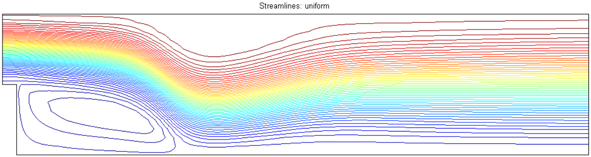
$$x = 0;\ -1 \leq y \leq 1$$

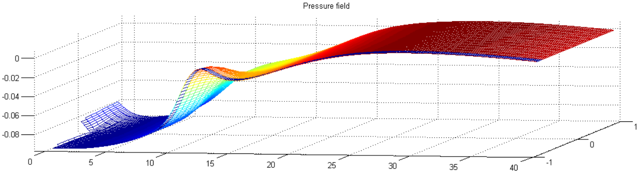
- Vitesse nulle sur le bord du conduit
- On suppose des conditions de Neumann à la sortie du conduit $\Rightarrow p = 0$

$$x = 40;\ -1 \leq y \leq 1$$

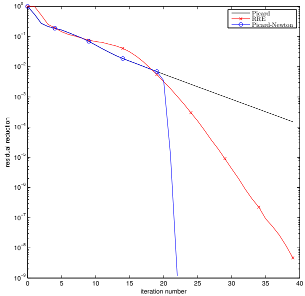
On fait varier le terme de viscosité et le type de discrétization

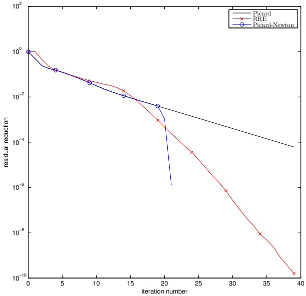
Solution pour $\nu = 1/600$



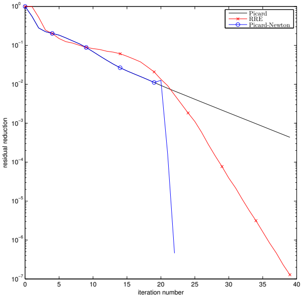


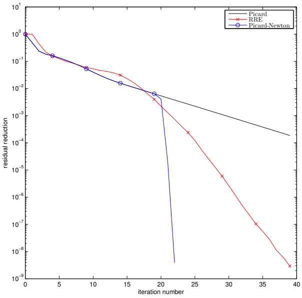
Convergence nonlinéaire pour $\nu = 3/800$: Q_2–P_{-1}(gauche) et Q_1–P_0(droite)





Convergence nonlinéaire pour $\nu = 1/300$: Q_2–P_{-1}(gauche) et Q_1–P_0(droite)





Convergence nonlinéaire pour $\nu = 1/600$: Q_2–P_{-1}(gauche) et Q_1–P_0(droite)