# Etude de la qualité de l'information contenue dans l'approximation de la hessienne inverse de l'algorithme quasi-Newton à mémoire limitée au cours de la minimisation

Didier AUROUX

Stage de 2ème année sous la direction de Fabrice Veersé et Jacques Blum

Laboratoire de Modélisation et Calcul (LMC-IMAG)
Université Joseph Fourier - Grenoble

Juin - Juillet 1999

# 1 Introduction

En météorologie et en océanographie, les prévisions reposent largement sur une bonne estimation de l'état initial à partir duquel sera calculée l'évolution future du système. La reconstitution de cet état initial peut être réalisée en utilisant les observations disponibles grâce à des techniques d'assimilation variationnelle de données. Celle-ci consiste à calculer la meilleure estimation possible de l'état initial, grâce à une minimisation explicite de la fonctionnelle qui mesure l'écart entre les observations et les états estimés du système.

La minimisation de la fonctionnelle s'effectue à l'aide de méthodes de type Quasi-Newton, qui diffèrent de la méthode de Newton par l'utilisation d'approximations successives de la hessienne inverse de la fonctionnelle au point courant, approximations moins coûteuses à mettre en oeuvre que la *vraie* hessienne. Néanmoins, de nombreux paramètres doivent être stockés afin d'approcher au mieux la hessienne. Pour des raisons de coût en mémoire, on utilise des algorithmes dits *à mémoire limitée* qui consistent à ne garder qu'une quantité bornée, relativement faible, des informations nécessaires à la mise à jour de l'approximation de la hessienne.

On utilise ici l'algorithme de minimisation inverse BFGS (Broyden-Fletcher-Goldfarb-Shanno) à mémoire limitée. Au cours de la minimisation de la fonctionnelle, on construit une approximation de la hessienne inverse. A l'optimum, celle-ci représente une information importante pour le problème d'identification considéré, car elle représente les covariances d'erreur du minimum calculé. En pratique, pour des raisons de coût de calcul, la minimisation est arrêtée avant convergence. Le problème essentiel est donc de savoir quelle est la qualité de l'approximation de la hessienne inverse fabriquée.

# 2 Hessienne et covariances d'erreur

Considérons des observations $y_o$ du système, une estimation $x_e$ de l'état initial du système, on cherche alors à minimiser la fonctionnelle suivante :

$$\begin{aligned} J(x) &= J_e(x) + J_o(x) \\ &= \frac{1}{2}(x - x_e)^T [P_e(t_0)]^{-1} (x - x_e) \\ &+ \frac{1}{2}(H(x) - y_o)^T R^{-1} (H(x) - y_o), \end{aligned} \tag{1}$$

où $H$ est un opérateur qui fournit une estimation des observations du système à l'état correspondant, les matrices $P_e(t_0)$ et $R$ sont les matrices de covariance d'erreur relatives à $(x_i - x_e)$ et $(H(x_i) - y_o)$ respectivement, $x_i$ désignant l'état réel initial du système.

Soit $x_0$ le minimum de la fonctionnelle $J$, $x_0$ est caractérisé par :

$$\begin{aligned} \nabla J(x_0) &= [P_e(t_0)]^{-1}(x_0 - x_e) + H'(x_0)^T R^{-1}(H(x_0) - y_o) \\ &= 0. \end{aligned} \tag{2}$$

La matrice hessienne correspondante est alors :

$$\begin{aligned} \nabla^2 J(x_0) &= [P_e(t_0)]^{-1} + H'(x_0)^T R^{-1} H'(x_0) \\ &+ \frac{1}{2}[H''(x_0)^T R^{-1}(H(x_0) - y_o) + (H(x_0) - y_o)^T R^{-1} H''(x_0)]. \end{aligned} \tag{3}$$

La condition $\nabla J(x_0) = 0$ peut se réécrire, en introduisant l'état réel initial du système $x_i$ :

$$[P_e(t_0)]^{-1}((x_0 - x_i) - (x_e - x_i)) + H'(x_0)^T R^{-1}(H(x_i) + H'(x_i).(x_0 - x_i) - y_o) = 0,$$

soit encore

$$\begin{aligned} ([P_e(t_0)]^{-1} + [H'(x_0)]^T R^{-1} H'(x_i))(x_0 - x_i) &= [P_e(t_0)]^{-1}(x_e - x_i) \\ &- [H'(x_0)]^T R^{-1}[H(x_i) - y_o]. \end{aligned}$$

En multipliant à droite par l'expression transposée et en prenant l'espérance mathématique, on trouve la matrice de covariances d'erreur d'analyse :

$$P_0 = E((x_0 - x_i)(x_0 - x_i)^T) = ([P_e(t_0)]^{-1} + [H'(x_0)]^T R^{-1} H'(x_i))^{-1}, \quad (4)$$

sous réserve d'avoir les hypothèses suivantes :

- $\varepsilon_o = y_o - H(x_i)$ ; $\quad E(\varepsilon_o) = 0$ ; $\quad E(\varepsilon_o.\varepsilon_o^T) = R$
- $E(x_i - x_e) = 0$ ; $\quad E((x_i - x_e)(x_i - x_e)^T) = P_e(t_0)$
- $E(\varepsilon_o(x_i - x_e)^T) = 0$

On en déduit donc avec (3) et (4) que la hessienne inverse approche la matrice de covariances d'erreur d'analyse lorsque les non-linéarités sont négligeables et que $x_0$ est dans un voisinage de $x_i$ où $H'$ peut être considérée constante : $H'(x_0) \approx H'(x_i) = H'$.

# 3 Algorithmes de type Newton, Quasi-Newton, BFGS, L-BFGS, formules de mise à jour

## 3.1 Algorithme de Newton (avec recherche linéaire)

L'algorithme de Newton consiste en la méthode suivante :

- à partir de $x_k$, on construit la direction de descente $d_k = -[\nabla^2 J(x_k)]^{-1}.\nabla J(x_k)$ qui minimise l'approximation quadratique locale $J(x_k) + p.\nabla J(x_k) + \frac{1}{2}p^T.\nabla^2 J(x_k).p$.
- On calcule alors un pas de descente $\rho_k$ le long de la direction $d_k$ par recherche linéaire (on minimise la fonctionnelle le long de la direction $d_k$) de sorte que $J(x_k + \rho_k.d_k) \leq J(x_k) + \alpha\rho_k[\nabla J(x_k)]^T.d_k$ et $|[\nabla J(x_k + \rho_k.d_k)]^T.d_k| \geq \beta|[\nabla J(x_k)]^T.d_k|$, avec $0 < \alpha < \beta < 1$ (conditions de Wolfe, [15]),
- et on pose $x_{k+1} = x_k + \rho_k.d_k$.

On réitère ce procédé jusqu'à convergence de la suite $x_k$ vers le minimum de la fonctionnelle $J$.

Cet algorithme est très efficace, mais dans notre cas, il est très coûteux à mettre en oeuvre puisqu'il nécessite à chaque étape le calcul de la hessienne inverse de la fonctionnelle au point considéré. L'algorithme de Quasi-Newton offre une alternative.

## 3.2 Algorithme quasi-Newton

L'idée de cet algorithme ([3]) est de remplacer la hessienne $H = \nabla^2 J$ (ou son inverse $W$) par une suite d'approximations symétriques définies positives, que l'on met à jour à chaque itération, pour un coût relativement faible. La mise à jour de cette approximation lors d'une étape de l'algorithme est en général une correction de rang 2 : pour passer de l'étape $k$ à l'étape $k+1$, on pose $W_{k+1} = W_k + \alpha u.u^T + \beta v.v^T$, $\alpha$ et $\beta$ étant des scalaires, $u$ et $v$ des vecteurs.

L'algorithme est donc le suivant : si on a à l'étape $k$ une approximation $W_k$ de $[\nabla^2 J(x_k)]^{-1}$, on définit la direction de descente $d_k = -W_k.\nabla J(x_k)$, on cherche un pas de descente $\rho_k$ par recherche linéaire et on pose $x_{k+1} = x_k + \rho_k.d_k$, et on calcule une approximation $W_{k+1}$ de $[\nabla^2 J(x_{k+1})]^{-1}$ avec une formule de mise à jour de rang 2 utilisant $x_k$, $x_{k+1}$, $\nabla J(x_k)$ et $\nabla J(x_{k+1})$.

Toute la difficulté consiste alors à trouver une bonne formule de mise à jour ...

## 3.3 Algorithme BFGS

L'algorithme BFGS ([1]), dû à Broyden, Fletcher, Goldfarb et Shanno, est un algorithme de type quasi-Newton où la formule de mise à jour de l'approximation de la hessienne inverse est :

$$W_{k+1} = U(W_k, s_k, y_k) = \Big(I - \frac{s_k \otimes y_k}{\langle y_k, s_k\rangle}\Big) W_k \Big(I - \frac{y_k \otimes s_k}{\langle y_k, s_k\rangle}\Big) + \frac{s_k \otimes s_k}{\langle y_k, s_k\rangle}, \tag{5}$$

avec $s_k = x_{k+1} - x_k$, $y_k = \nabla J(x_{k+1}) - \nabla J(x_k)$, et $u \otimes v : d \mapsto \langle v, d\rangle u$. En pratique, on stocke $W_0$ et les paires $(s_k, y_k)$ calculées à chaque étape.

L'inconvénient majeur de cet algorithme est le coût de stockage de toutes les paires $(s_k, y_k)$, surtout lorsque la dimension du problème est importante (de l'ordre de $10^6$).

## 3.4 Algorithme L-BFGS

Cet algorithme ([7], [9]) est une extension de l'algorithme précédent, à mémoire limitée. Concrètement, on se donne un entier $m$, de l'ordre de 5, et on ne stocke que les $m$ dernières paires $(s, y)$ calculées. La mise à jour des $W_k$ se fait alors de la manière suivante :

$$\begin{aligned} W_{k+1} &= U(W_k, s_k, y_k), \qquad 0 \le k \le m-1 \\[1em] W_k^0 &= D_k, \\ W_k^{i+1} &= U(W_k^i, s_{k-m+i}, y_{k-m+i}), \qquad (0 \le i \le m-1) \\ W_k &= W_k^m, \qquad k \ge m+1 \end{aligned} \tag{6}$$

où $D_k$ est une matrice diagonale.

### 3.5 Mise à jour de la matrice diagonale

Il existe plusieurs formules de mise à jour de la matrice diagonale $D_k$ permettant une bonne initialisation de l'algorithme L-BFGS :

$$\text{BFGS inverse : } D_+^{(i)} = D^{(i)} + \Big(\frac{1}{\langle y,s\rangle} + \frac{\langle Dy,y\rangle}{\langle y,s\rangle^2}\Big)\langle s,e_i\rangle^2 - \frac{2D^{(i)}\langle y,e_i\rangle\langle s,e_i\rangle}{\langle y,s\rangle} \tag{7}$$

$$\text{BFGS direct : } D_+^{(i)} = \left(\frac{1}{D^{(i)}} + \frac{\langle y,e_i\rangle^2}{\langle y,s\rangle} - \frac{\langle s,e_i\rangle^2}{(D^{(i)})^2\langle D^{-1}s,s\rangle}\right)^{-1} \tag{8}$$

$$\text{DFP inverse : } D_+^{(i)} = D^{(i)} + \frac{\langle s,e_i\rangle^2}{\langle y,s\rangle} - \frac{(D^{(i)}\langle y,e_i\rangle)^2}{\langle Dy,y\rangle} \tag{9}$$

$$\text{Quasi-Cauchy : } D_+^{(i)} = \frac{D^{(i)}}{(1+\mu^*\langle y,e_i\rangle^2)^2}, \tag{10}$$

où $\mu^*$ est la plus grande solution de $F(\mu) = b$, avec $F(\mu) = \sum_{i=1}^n \frac{D^{(i)}\langle y,e_i\rangle^2}{(1+\mu\langle y,e_i\rangle^2)^2}$ et $b = \langle s,y\rangle$.

### 3.6 Mise à l'échelle de la matrice diagonale

Si on fait un développement limité de $\nabla J(x_{k+1})$, on obtient :

$$\begin{aligned} & \nabla J(x_{k+1}) = \nabla J(x_k) + \nabla^2 J(x_k).(x_{k+1}-x_k) + \dots \\ (i.e.) \quad & y_k \approx \nabla^2 J(x_k).s_k, \end{aligned}$$

et donc pour que $W_k$ soit une bonne approximation de $[\nabla^2 J(x_k)]^{-1}$, on peut demander à $W_k$ de vérifier $W_k y_k = s_k$ le long de $y_k$, c'est-à-dire :

$$\langle W_k y_k, y_k\rangle = \langle y_k, s_k\rangle. \tag{11}$$

Par conséquent, on effectue avant (ou éventuellement après) la mise à jour de la diagonale une remise à l'échelle, de sorte que la nouvelle matrice diagonale ait la bonne propriété. On remplace donc $D$ par $\sigma D$ avec $\sigma = \frac{\langle y,s\rangle}{\langle Dy,y\rangle}$.

*Remarque :* Cette mise à l'échelle ne sert à rien si on utilise la formule de mise à jour quasi-Cauchy (10) puisque par construction, la nouvelle matrice diagonale vérifie déjà la condition $\langle Dy,y\rangle = \langle s,y\rangle$ ([8]).

## 4 Problème modèle

Nous nous plaçons ici dans le cas particulier d'un modèle régi par l'équation de Burgers 1D sur un cercle :

$$\frac{\partial x}{\partial t} + \frac{1}{2}\frac{\partial x^2}{\partial s} - \nu\frac{\partial^2 x}{\partial s^2} = 0 \tag{12}$$

où $s$ représente l'abscisse curviligne le long du parallèle 45°N et $\nu$ est le coefficient de diffusion (ici $\nu = 10^5 \mathrm{m}^2\mathrm{s}^{-1}$).

Le minimiseur utilisé est le programme `m1qn3` et sa version en double précision `n1qn3` ([4]), la dimension du problème est 258 mais le simulateur est utilisé en mode basse résolution ($\rightsquigarrow$ dimension 130), on minimise la fonctionnelle quadratique approchée ([2]) :

$$\tilde{J}(x) \;=\; \frac{1}{2}(x - x_e)^T [P_e(t_0)]^{-1}(x - x_e) + \frac{1}{2}(\tilde{H}(x) - y_o)^T R^{-1}(\tilde{H}(x) - y_o),$$

où $\tilde{H}$ est une approximation linéaire de $H$. Le critère de convergence lors de la minimisation est le suivant :

$$\frac{||\nabla \tilde{J}(x_k)||}{||\nabla \tilde{J}(x_0)||} < \varepsilon_g$$

où $||.||$ est la norme induite par $\langle .,. \rangle$ et $\varepsilon_g$ est de l'ordre de $1,5.10^{-8}$. Les paramètres pris en option sont les suivants :

| paramètre | description | valeur par défaut |
|---|---|---|
| `niter` | nombre maximal d'itérations lors de la minimisation | 120 |
| `mupdat` | nombre de paires $(s, y)$ stockées | 5 |
| `cupdat` | formule de mise à jour de la matrice diagonale | `dir BFGS` |
| `cupscal` | moment où effectuer la mise à l'échelle | `before` |
| `cuppair` | paire $(s, y)$ utilisée pour la mise à jour de la diagonale | `newest` |

Les valeurs possibles des paramètres sont les suivantes :

| paramètre | valeurs possibles |
|---|---|
| `cupdat` | `dir BFGS`, `inv BFGS`, `inv DFP`, `QC` |
| `cupscal` | `never`, `before`, `after` (respectivement jamais, avant ou après la mise à jour de la diagonale), `after*` (mise à l'échelle uniquement pour la minimisation, voir §4.4) |
| `cuppair` | `oldest`, `newest` (respectivement la paire la plus vieille, que l'on s'apprête à effacer, et la paire la plus récente) |

Une fois la minimisation effectuée, la méthode de l'adjoint de second ordre (voir Annexe 1, [5], [14]) permet de calculer la hessienne exacte $H_{exacte}$ et donc de comparer cette dernière avec son approximation $W^{-1}_{approchée}$ calculée lors de la minimisation, et ceci de deux manières différentes :

- soit en regardant le spectre de la différence $H_{exacte} - W^{-1}_{approchée}$,
- soit en regardant le spectre de la différence relative $Id - H_{exacte} W_{approchée}$,

les valeurs propres d'une matrice donnée $A$ étant calculées avec un algorithme de type Lanczos ([6]) ne nécessitant que la mise en oeuvre de l'opérateur $x \mapsto A.x$.

## 4.1 Convergence de BFGS

En fixant le nombre de paires stockées (`mupdat`) de façon arbitrairement grande (de sorte que l'algorithme ne soit plus à mémoire limitée), on s'assure de la convergence de l'algorithme BFGS en faisant varier le nombre d'itérations (`niter`) de 10 à 40, nombre d'itération à partir duquel le critère de convergence est satisfait (Figure 1). Toutes les figures sont dédoublées, avec à gauche le spectre de $H_{exacte} - W^{-1}_{approch\acute{e}e}$ et à droite le spectre de $Id - H_{exacte}W_{approch\acute{e}e}$. Le tableau 1 (en Annexe 2) donne la somme des valeurs absolues des valeurs propres des différences $H_{exacte} - W^{-1}_{approch\acute{e}e}$ et $Id - H_{exacte}W_{approch\acute{e}e}$ en fonction de `niter`.

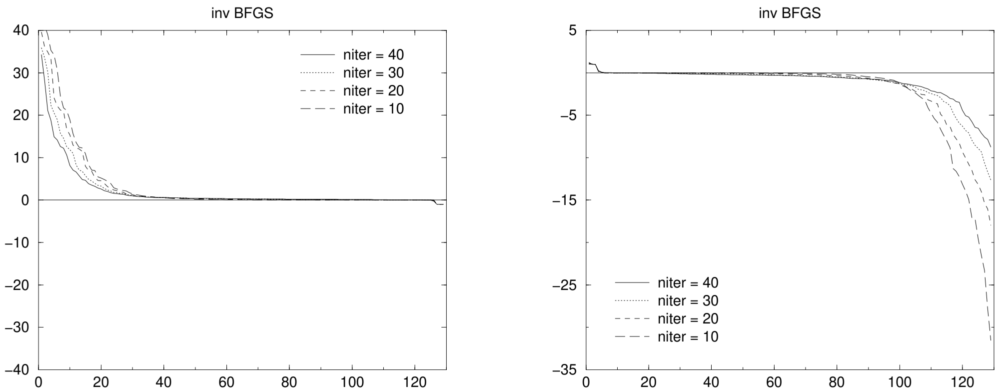


FIG. 1: Comportement de BFGS en fonction du nombre d'itérations lors de la minimisation.

Comme c'était attendu, plus le nombre d'itérations autorisé lors de la minimisation est grand, plus l'inverse de la hessienne fabriqué est proche de l'inverse exact.

## 4.2 Comparaison des diverses formules de mise à jour de la diagonale

On fixe désormais `mupdat` à 5, `cupscal` à `never` (pas de remise à l'échelle), et on fait varier `cupdat`. A chaque minimisation, on relève le nombre d'itérations/simulations nécessaires pour parvenir au minimum (*i.e.* pour satisafaire le critère de convergence)(Tableau 2), et on compare les spectres de la différence (Figure 2). La figure 3 montre la même comparaison dans le cas où `cuppair` vaut `oldest` (on utilise la paire $(s, y)$ la plus ancienne). Les tableaux 3 et 4 donnent la somme des valeurs absolues des valeurs propres de la différence $H_{exacte} - W^{-1}_{approch\acute{e}e}$ et $Id - H_{exacte}W_{approch\acute{e}e}$ dans chaque cas.

Lorsqu'on utilise la paire la plus récente, les meilleurs résultats (en terme de différence entre les deux hessiennes) sont obtenus avec `cupscal` = `dir BFGS` (et `QC` lorsqu'on regarde la différence relative des spectres). Avec la paire la plus ancienne, les meilleurs résultats sont obtenus avec `inv BFGS`, mais ils sont nettement moins bons que quand on utilise la paire la plus récente.

Au niveau du nombre d'itérations et surtout de simulations effectuées, `dir BFGS` semble être la meilleure formule de mise à jour, en utilisant la paire $(s, y)$ la plus récente.

## 4.3 Utilisation de la mise à l'échelle

La mise à l'échelle de la matrice diagonale avant (ou après) la mise à jour permet de gagner en temps de calcul. Les figures 4 et 5 montrent la comparaison des différentes

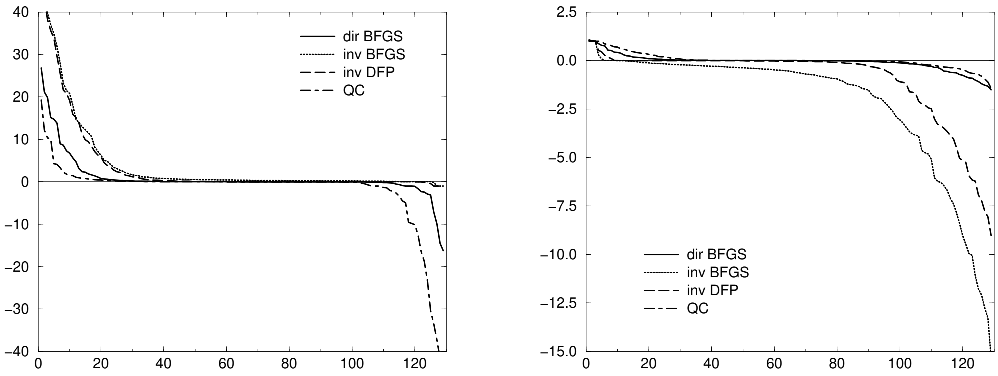


FIG. 2: Comparaison des différentes formules de mise à jour de la diagonale sans remise à l'échelle, avec la paire la plus récente.

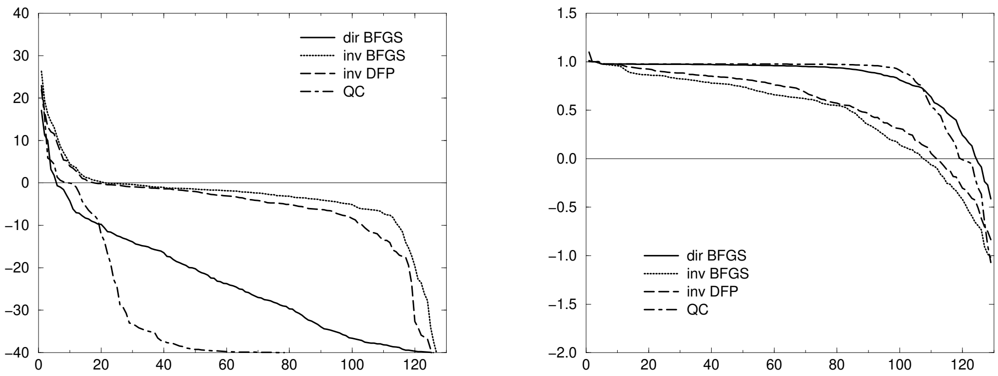


FIG. 3: Comparaison des différentes formules de mise à jour de la diagonale sans remise à l'échelle, avec la paire la plus vieille.

formules de mise à jour, avec remise à l'échelle respectivement avant ($\texttt{cupscal} = \texttt{before}$) et après ($\texttt{cupscal} = \texttt{after}$) la mise à jour.

Globalement, on obtient de bien meilleurs temps de calcul (nombres de simulations considérablement réduits dans certains cas) lorsqu'on effectue une mise à l'échelle, mais en contrepartie, l'approximation construite de la hessienne inverse est moins bonne que lorsqu'on ne fait pas de mise à l'échelle.

## 4.4 Mise à l'échelle seulement pour la minimisation

La mise à l'échelle permet de gagner en temps de calcul lors de la minimisation mais elle a tendance à produire de mauvaises approximations de la hessienne (ou de son inverse). Il semble donc judicieux d'effectuer la minimisation avec une matrice diagonale mise à l'échelle mais de mettre à jour la matrice diagonale originale. La figure 6 montre la comparaison des formules de mise à jour dans ces conditions-là.

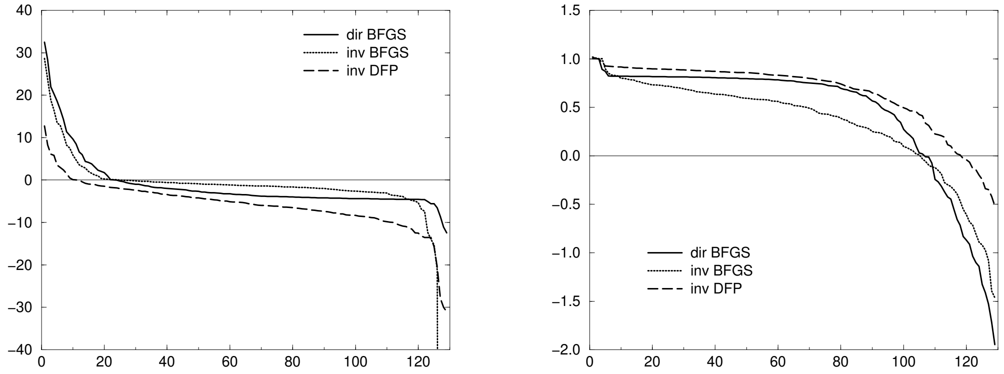


FIG. 4: Comparaison des différentes formules de mise à jour de la diagonale avec remise à l'échelle avant la mise à jour, en utilisant la paire la plus récente.

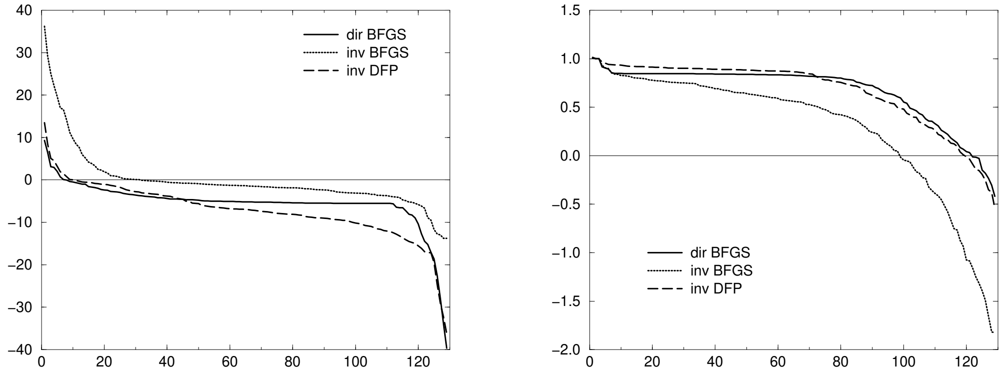


FIG. 5: Comparaison des différentes formules de mise à jour de la diagonale avec remise à l'échelle après la mise à jour, en utilisant la paire la plus récente.

Avec cette méthode là, on obtient une approximation de la hessienne inverse de l'ordre de celle obtenue lorsqu'on n'effectuait pas de mise à l'échelle, mais le nombre de simulations nécessaires pour effectuer la minimisation a été encore réduit.

Dans notre cas particulier, on a donc intérêt à utiliser la formule de mise à jour de la diagonale `dir BFGS`, en utilisant la paire $(s, y)$ la plus récente, et en effectuant une mise à l'échelle seulement pour la minimisation et pas pour la mise à jour de la diagonale.

## 4.5 Evolution en fonction de `mupdat`

On choisit donc `cupdat` = `dir BFGS`, `cupscal` = `after`* (mise à l'échelle uniquement pour la minimisation), `cuppair` = `newest`, et on fait maintenant varier `mupdat` (nombre de paires $(s, y)$ stockées). La figure 7 montre le spectre des différences ainsi obtenues pour `mupdat` variant de 20 à 2. Le tableau 5 donne la somme des valeurs absolues des valeurs propres des différences en fonction de `mupdat`.

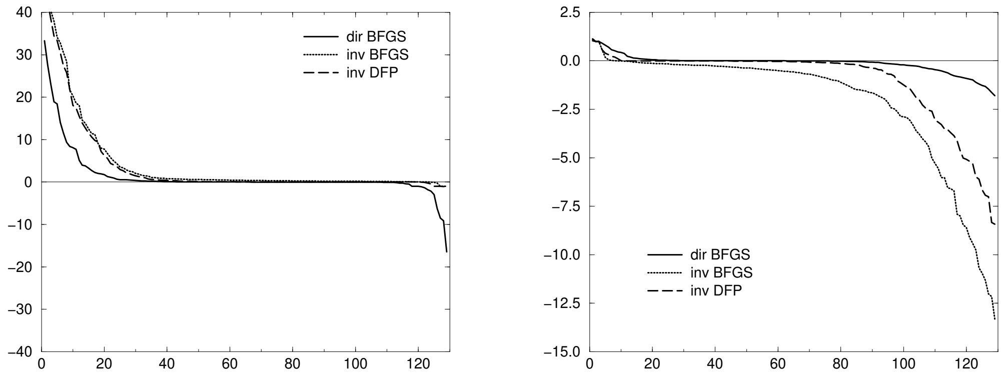


FIG. 6: Comparaison des différentes formules de mise à jour de la diagonale avec remise à l'échelle seulement pour la minimisation et pas pour la mise à jour de la diagonale.

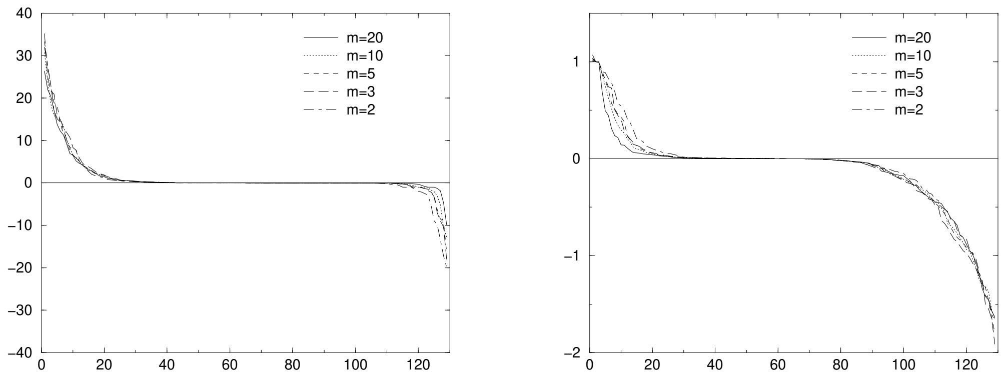


FIG. 7: Evolution des résultats obtenus dans le meilleur cas en fonction du nombre de paires $(s, y)$ stockées (`mupdat`).

Les différences sont relativement faibles, même si les meilleurs résultats sont obtenus lorsque `mupdat` est grand, ce qui est assez logique puisqu'on a alors tendance à stocker plus d'informations, ce qui permet une meilleure mise à jour.

## 4.6 Evolution en fonction de `niter`

On refixe `mupdat` à 5, et maintenant on fait varier `niter` (nombre maximum d'itérations lors de la minimisation) de 10 à 40, nombre d'itérations dont on a besoin pour mener à bien la minimisation. Les résultats apparaissent sur la figure 8 et le tableau 6.

Là encore, il est clair que plus le nombre d'itérations autorisé est grand, meilleure est l'approximation construite de la hessienne inverse.

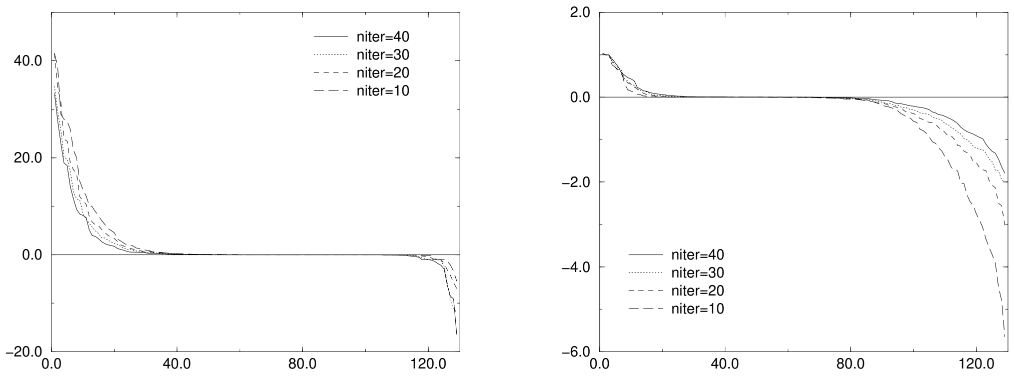


FIG. 8: Evolution des résultats obtenus dans le meilleur cas en fonction du nombre maximum d'itérations autorisé lors de la minimisation (`niter`).

# 5 Conclusion

Grâce à la mise à l'échelle de la matrice diagonale uniquement lors de la minimisation, on a obtenu de très bons résultats (en terme de précision et de rapidité de calcul). Des tests sont actuellement en cours pour vérifier la pertinence de cette technique, tout d'abord avec le même problème modèle mais utilisé en mode haute résolution, et sans faire l'approximation quadratique de la fonctionnelle, mais aussi avec plusieurs séries de cas-tests, avec des dimensions beaucoup plus grandes (1000 à 40000). Les résultats de ces tests seront rapportés dans un double article en cours d'écriture ([12], [13]).

# Références

[1] Broyden, C.G., 1969: A new double-rank minimization algorithm. *Notices American Math. Soc.*, **16,** pp. 670

[2] Courtier, P., Thépaut, J.-N. and Hollingsworth, A., 1994: A strategy for operational implementation of 4D-Var, using an incremental approach. *Quart. J. Roy. Meteor. Soc.*, **120,** 1367-1387

[3] Dennis, J.E. and Moré, J.J., 1977: Quasi-Newton methods, motivation and theory. *SIAM Rev.*, **19,** 46-89

[4] Gilbert, J.-Ch. and Lemaréchal, C., 1989: Some numerical experiments with variable storage quasi-Newton algorithms. *Math. Prog.*, **45,** 407-435

[5] Le Dimet, F.-X., Ngodock, H.E. and Luong, B., 1997: Sensitivity Analysis in Variational Data Assimilation. *J. Meteorol. Soc. Japan*, **75-1B,** 245-255

[6] Lehoucq, R.B., Sorensen, D.C. and Yang, C., 1997: ARPACK User's Guide: Solution of Large-Scale Eigenvalue Problems with Implicit Restarted Arnoldi Methods. *Available from http://www.caam.rice.edu/sofware/ARPACK/*

[7] Liu, D.C. and Nocedal, J., 1989: On the limited memory BFGS method for large scale optimization. *Math. Prog.*, **45,** 503-528

[8] Nazareth, J.L., Wolkowicz, H. and Zhu, M., 1999: The Quasi-Cauchy Relation and Diagonal Updating. To appear in *SIAM J. Optim.*

[9] Nocedal, J., 1980: Updating quasi-Newton matrices with limited-storage. *Math. Comput.*, **35,** 773-782

[10] Veersé, F., 1999a: Variable-storage quasi-Newton operators as inverse forecast/analysis error covariance matrices in variational data assimilation. *INRIA Research Report No. 3685.*

[11] Veersé, F., 1999b: Variable-storage quasi-Newton operators for modeling error covariances. To appear in *Proceedings of the Third WMO International Symposium on Assimilation of Observations in Meteorology and Oceanography, 7-11 June 1999, Quebec City, Canada*, World Meteorological Organization, Geneva.

[12] Veersé, F., Auroux, D., Gilbert, J.-Ch. and Fisher, M., 1999: Scaling and updating l-BFGS diagonal preconditioners: a case study. *In preparation.*

[13] Veersé, F., Auroux, D. and Gilbert, J.-Ch., 1999: Scaling and updating l-BFGS diagonal preconditioners: CUTE, MINPACK-2 and MODULOPT experiments. *In preparation.*

[14] Wang, Z., Navon, I.M., Le Dimet, F.-X. and Zou, X., 1992: The Second Order Adjoint Analysis: Theory and Applications. *Meteorol. Atmos. Phys.*, **50,** 3-20

[15] Wolfe, P., 1969: Convergence conditions for ascent methods. *SIAM Rev.*, **11/2,** 226-235

# Annexe 1 - Adjoint de second ordre

Considérons la fonction coût

$$\begin{aligned}\mathcal{J}(\mathbf{x}_0) &= \frac{1}{2}\left(\mathbf{x}_0-\mathbf{x}^f\right)^T\left[\mathbf{P}^f(t_0)\right]^{-1}\left(\mathbf{x}_0-\mathbf{x}^f\right)\\ &+ \frac{1}{2}\sum_{i=1}^{M}\left(H_i(\mathbf{x}(t_i))-\mathbf{y}_i^o\right)^T\mathbf{R}_i^{-1}\left(H_i(\mathbf{x}(t_i))-\mathbf{y}_i^o\right)\end{aligned}$$

où $\mathbf{x}$ est la solution du modèle

$$\left\{\begin{aligned}\frac{\partial \mathbf{x}}{\partial t} &= F(\mathbf{x})\\ \mathbf{x}(t_0) &= \mathbf{x}_0\end{aligned}\right.$$

Alors le gradient s'écrit $\nabla\mathcal{J}(\mathbf{x}_0)=\left[\mathbf{P}^f(t_0)\right]^{-1}\left(\mathbf{x}_0-\mathbf{x}^f\right)+\mathbf{p}(t_0)$ où $\mathbf{p}$ est la solution du *modèle adjoint de premier ordre*

$$\left\{\begin{aligned}-\frac{\partial \mathbf{p}}{\partial t} &= \left[\mathbf{F}'(\mathbf{x})\right]^T\mathbf{p}\\ &+ \textstyle\sum_{i=1}^{M}\mathbf{H}_i'(\mathbf{x}(t_i))^T\mathbf{R}_i^{-1}\left(H_i(\mathbf{x}(t_i))-\mathbf{y}_i^o\right)\delta_{t_i}(t)\\ \mathbf{p}(T) &= 0.\end{aligned}\right.$$

Considérons une perturbation $\delta\mathbf{x}_0$ de la condition initiale $\mathbf{x}_0$. La perturbation résultante de $\mathbf{x}$ est donnée au premier ordre par le *modèle linéaire tangent*

$$\left\{\begin{aligned}\frac{\partial \delta\mathbf{x}}{\partial t} &= \mathbf{F}'(\mathbf{x}).\delta\mathbf{x}\\ \delta\mathbf{x}(t_0) &= \delta\mathbf{x}_0.\end{aligned}\right.$$

et la pertubation correspondante $\delta\mathbf{p}$ de la variable adjointe est donnée par le *modèle adjoint de second ordre*

$$\left\{\begin{aligned}-\frac{\partial \delta\mathbf{p}}{\partial t} &= \left[\mathbf{F}'(\mathbf{x})\right]^T.\delta\mathbf{p}+\left[\mathbf{F}''(\mathbf{x}).\delta\mathbf{x}\right]^T\mathbf{p}\\ &+ \textstyle\sum_{i=1}^{M}\Big\{\left[\mathbf{H}_i'(\mathbf{x}(t_i))\right]^T\mathbf{R}_i^{-1}\mathbf{H}_i'(\mathbf{x}(t_i))\\ &\qquad+\tfrac{1}{2}\left[\mathbf{H}_i''(\mathbf{x}(t_i))\right]^T\mathbf{R}_i^{-1}\left(H_i(\mathbf{x}(t_i))-\mathbf{y}_i^o\right)\\ &\qquad+\tfrac{1}{2}\left(H_i(\mathbf{x}(t_i))-\mathbf{y}_i^o\right)^T\mathbf{R}_i^{-1}\left[\mathbf{H}_i''(\mathbf{x}(t_i))\right]\Big\}.\delta\mathbf{x}\quad\delta_{t_i}(t)\\ \delta\mathbf{p}(T) &= 0\end{aligned}\right.$$

Soit $\mathbf{p}_{\mathbf{x}_0+\delta\mathbf{x}_0}$ la solution du modèle adjoint de premier ordre après la perturbation $\delta\mathbf{x}_0$ de la condition initiale $\mathbf{x}_0$. Par définition, on a $\mathbf{p}_{\mathbf{x}_0+\delta\mathbf{x}_0}(t_0)=\mathbf{p}(t_0)+\delta\mathbf{p}(t_0)$. Or

$$\begin{aligned}\nabla\mathcal{J}(\mathbf{x}_0+\delta\mathbf{x}_0) &= \nabla\mathcal{J}(\mathbf{x}_0)+\nabla^2\mathcal{J}(\mathbf{x}_0).\delta\mathbf{x}_0+O(\|\delta\mathbf{x}_0\|^2),\\ \nabla\mathcal{J}(\mathbf{x}_0+\delta\mathbf{x}_0) &= \left[\mathbf{P}^f(t_0)\right]^{-1}(\mathbf{x}_0+\delta\mathbf{x}_0-\mathbf{x}^f)+\mathbf{p}_{\mathbf{x}_0+\delta\mathbf{x}_0}(t_0),\\ \nabla\mathcal{J}(\mathbf{x}_0) &= \left[\mathbf{P}^f(t_0)\right]^{-1}(\mathbf{x}_0-\mathbf{x}^f)+\mathbf{p}(t_0).\end{aligned}$$

On en déduit : $\nabla^2\mathcal{J}(\mathbf{x}_0).\delta\mathbf{x}_0=\left[\mathbf{P}^f(t_0)\right]^{-1}\delta\mathbf{x}_0+\delta\mathbf{p}(t_0)+O(\|\delta\mathbf{x}_0\|^2)$.

# Annexe 2

| niter | $H_{exacte} - W_{approchée}^{-1}$ | $Id - H_{exacte}W_{approchée}$ |
|---|---|---|
| 40 | 0.2649671325683594E+03 | 0.1403615417480469E+03 |
| 30 | 0.3194703674316406E+03 | 0.1751118469238281E+03 |
| 20 | 0.3929013366699219E+03 | 0.2311549224853516E+03 |
| 10 | 0.4549848937988281E+03 | 0.3213396606445312E+03 |

TAB. 1: Evolution de la somme des valeurs absolues des valeurs propres des différences entre les hessiennes inverses en fonction du nombre d'itérations lors de la minimisation (`niter`) pour BFGS.

| `cupscal` - `cuppair` | `dir BFGS` | `inv BFGS` | `inv DFP` | `QC` |
|---|---|---|---|---|
| `never` - `newest` | 40 / 52 | 42 / 84 | 42 / 84 | 120 / 130 |
| `never` - `oldest` | 77 / 78 | 69 / 70 | 68 / 69 | 96 / 103 |
| `before` - `newest` | 47 / 49 | 98 / 102 | 48 / 53 | - |
| `after` - `newest` | 47 / 49 | 51 / 53 | 50 / 52 | - |
| `after`* - `newest` | 40 / 43 | 44 / 46 | 43 / 46 | - |

TAB. 2: Nombre d'itérations/simulations nécessaires lors de la minimisation, en fonction des différentes formules de mise à jour, de la remise à l'échelle et de la paire utilisée.

| `cupscal` - `cuppair` | `dir BFGS` | `inv BFGS` | `inv DFP` | `QC` |
|---|---|---|---|---|
| `never` - `newest` | 0.241344E+03 | 0.518898E+03 | 0.455589E+03 | 0.658980E+03 |
| `never` - `oldest` | 0.316543E+04 | 0.792836E+03 | 0.103973E+04 | 0.473778E+04 |
| `before` - `newest` | 0.614110E+03 | 0.135939E+04 | 0.835252E+03 | - |
| `after` - `newest` | 0.753462E+03 | 0.533154E+03 | 0.100067E+04 | - |
| `after`* - `newest` | 0.271790E+03 | 0.542826E+03 | 0.479673E+03 | - |

TAB. 3: Somme de la valeur absolue des valeurs propres de la différence $H_{exacte} - W_{approchée}^{-1}$, en fonction des différentes formules de mise à jour, de la remise à l'échelle et de la paire utilisée.

| cupscal - cuppair | dir BFGS | inv BFGS | inv DFP | QC |
|---|---|---|---|---|
| never - newest | 0.287796E+02 | 0.284377E+03 | 0.134405E+03 | 0.287478E+02 |
| never - oldest | 0.108512E+03 | 0.786104E+02 | 0.835380E+02 | 0.111167E+03 |
| before - newest | 0.935679E+02 | 0.699889E+02 | 0.888812E+02 | - |
| after - newest | 0.888926E+02 | 0.815810E+02 | 0.900866E+02 | - |
| after* - newest | 0.334120E+02 | 0.277566E+03 | 0.137993E+03 | - |

TAB. 4: Somme de la valeur absolue des valeurs propres de la différence relative $Id - H_{exacte}W_{approchée}$, en fonction des différentes formules de mise à jour, de la remise à l'échelle et de la paire utilisée.

| mupdat | $H_{exacte} - W^{-1}_{approchée}$ | $Id - H_{exacte}W_{approchée}$ |
|---|---|---|
| 20 | 0.2101870269775391E+03 | 0.2955487251281738E+02 |
| 10 | 0.2425678405761719E+03 | 0.3203291320800781E+02 |
| 5 | 0.2717900085449219E+03 | 0.3341202545166016E+02 |
| 3 | 0.2860183715820313E+03 | 0.3513659667968750E+02 |
| 2 | 0.2945249938964844E+03 | 0.3372309494018555E+02 |

TAB. 5: Evolution de la somme des valeurs absolues des valeurs propres des différences entre les hessiennes inverses en fonction du nombre de paires stockées (`mupdat`) pour L-BFGS avec `dir BFGS`, `newest` et mise à l'échelle uniquement lors de la minimisation.

| niter | $H_{exacte} - W^{-1}_{approchée}$ | $Id - H_{exacte}W_{approchée}$ |
|---|---|---|
| 40 | 0.2717900085449219E+03 | 0.3341202545166016E+02 |
| 30 | 0.2984325561523438E+03 | 0.3938418197631836E+02 |
| 20 | 0.3348155212402344E+03 | 0.4909420776367188E+02 |
| 10 | 0.3803320922851562E+03 | 0.7779969787597656E+02 |

TAB. 6: Evolution de la somme des valeurs absolues des valeurs propres des différences entre les hessiennes inverses en fonction du nombre d'itérations lors de la minimisation (`niter`) pour L-BFGS avec `dir BFGS`, `newest` et mise à l'échelle uniquement lors de la minimisation.