Notice fonctionnelle détaillée de gpac

Gradient à Pas Adaptatif avec Corrections

Groupe Problèmes Inverses
Laboratoire des Signaux et Systèmes (CNRS – Supélec – UPS)
Supélec, Plateau de Moulon, 91192 Gif–sur–Yvette Cedex, France

L'algorithme décrit ci-après est un algorithme de descente du premier ordre à pas adaptatif, particulièrement adapté à la minimisation de critères multivariés fonctions d'un grand nombre de variables[1]. Il utilise les caractéristiques du premier ordre (gradient) et n'utilise pas d'information du second ordre (hessien ou approximations du hessien). Diverses directions de descente sont proposées (gradient simple, gradient conjugué, corrections de Vignes et de la bissectrice) et différentes techniques d'adaptation du pas de descente sont disponibles (dichotomie et interpolations). Il bénéficie de l'expérience du Groupe Problèmes Inverses (Laboratoire des Signaux et Systèmes, CNRS – Supélec – UPS) dans l'utilisation de tels algorithmes pour la résolution de problèmes inverses.

1 Principes généraux

Le problème général est celui de l'optimisation numérique. Il s'agit de trouver $\widehat{\boldsymbol{x}}$ minimisant un critère à valeur réelle $f(\boldsymbol{x})$:

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} f(\boldsymbol{x}).$$

La littérature sur le sujet est extrêmement vaste et on pourra consulter des ouvrages de référence [1, 2, 3]. L'outil dont il est question ici est particulièrement adapté aux problèmes de « *très grande taille*[1] ». Il s'agit d'un algorithme de descente du premier ordre à pas adaptatif. A partir d'une solution initiale $\boldsymbol{x}_0$, il procède de manière itérative en utilisant (à l'itération k) la récurrence générique suivante :

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \mu_k \, \boldsymbol{d}_k,$$

où $\boldsymbol{d}_k$ est la direction de descente et μ_k le pas de descente. Il existe de nombreuses variantes d'un tel algorithme se distinguant par :

- le choix de la direction de descente, décrit au paragraphe 1.1 ;
- l'adaptation du pas de descente décrite au paragraphe 1.2.

1.1 Choix de la direction de descente

Le choix le plus simple consiste à descendre dans la direction du gradient :

$$\boldsymbol{d}_k = \boldsymbol{g}_k \, ,$$

où $\boldsymbol{g}_k$ le gradient de f en $\boldsymbol{x}_k$. Ce choix est bien souvent efficace, cependant il peut s'avérer inefficace notamment lorsque les iso-critères forment des vallées très étroites (la direction propre associée à la plus petite valeur propre du hessien). L'algorithme oscille entre les deux bords de la vallée et la convergence se trouve ralentie (voir l'exemple de la Fig. 3). Pour pallier cet inconvénient, différentes corrections de la direction de descente ont été proposées.

[1] Le nombre N de variables peut être de l'ordre de $N = 100\,000$ ou même $N = 1\,000\,000$ et au delà dans les problèmes visés notamment en traitement d'image, en reconstruction 3D ou pour le traitement de séquences d'images.

– Les corrections de Vignes et de la bissectrice ont la même philosophie. Lorsque la dernière direction de descente $\boldsymbol{d}_{k-1}$ et le gradient au point courant $\boldsymbol{g}_k$ forment un angle obtus (*e.g.*, > 150 degrés) l'algorithme génère une direction de descente qui en est la bissectrice (correction de la bissectrice) ou la demi-somme (correction de Vignes). La direction obtenue est alors orientée dans la direction de la vallée plutôt que perpendiculairement à celle-ci (voir la Fig. 1). On a donc :

$$\boldsymbol{d}_k^{\text{B}} = \text{bissec}\,(\boldsymbol{d}_{k-1}, \boldsymbol{g}_k) = \frac{1}{2}\,|\boldsymbol{d}_k|\left(\frac{\boldsymbol{d}_{k-1}}{|\boldsymbol{d}_{k-1}|} + \frac{\boldsymbol{g}_k}{|\boldsymbol{g}_k|}\right)\,,$$

pour la bissectrice tandis que la correction de Vignes est :

$$\boldsymbol{d}_k^{\text{V}} = \frac{1}{2}\,(\boldsymbol{d}_{k-1} + \boldsymbol{g}_k)\,.$$

Il est à noter que la correction de la bissectrice fournit une direction qui est bien une direction de descente (ce qui n'est pas assuré par la correction de Vignes).

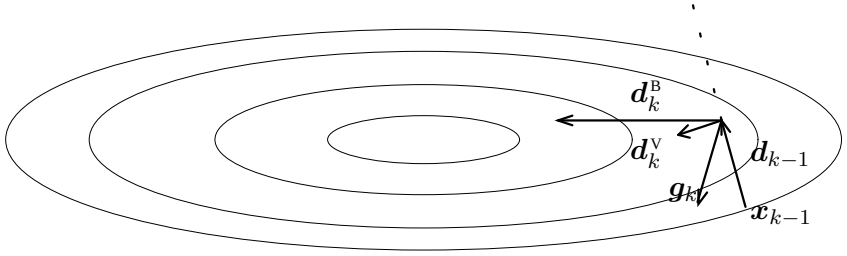


FIG. 1 – Différents choix pour la direction de descente

– Le principe du gradient conjugué est différent d'une simple correction. Son but est de construire une suite de directions de descente conjuguées, c'est-à-dire orthogonales dans la métrique définie par le produit scalaire $\langle u, v\rangle = u^{\text{t}} A\, v$ où A est le hessien de f. Dans le cas d'un critère purement quadratique la convergence exacte est obtenue en N itérations (N étant le nombre de variables). L'avantage par rapport à un algorithme du second ordre est qu'il n'est nullement besoin de stocker ou de calculer le hessien.

Il existe de plus des formules de récurrence pour générer les directions de descente conjuguées. Nous utilisons ici les formules de Polak & Ribière qui, dans le cas non quadratique, ont prouvé leur supériorité sur les formules initiales de Fletcher & Reeves :

$$\boldsymbol{d}_{k+1} = \gamma_k\,\boldsymbol{d}_k + \boldsymbol{g}_k\,, \quad \text{avec} \quad \gamma_k = \frac{(\boldsymbol{g}_{k+1} - \boldsymbol{g}_k)^{\text{t}}\boldsymbol{g}_{k+1}}{\boldsymbol{g}_k^{\text{t}}\boldsymbol{g}_k}\,.$$

Cependant, comme la minimisation monodimensionnelle est approchée, les directions successivement générées ne sont pas exactement conjuguées : on parle de pseudo-conjugaison. D'autre part, pour éviter des cumuls d'erreur sur ces directions, il est possible de réinitialiser périodiquement les directions conjuguées.

1.2 Choix du pas

Une fois la direction de descente déterminée, il s'agit de minimiser le critère dans cette direction : c'est la phase de minimisation monodimensionnelle. De façon théorique, le pas μ_k peut-être défini par :

$$\mu_k = \arg\min_{\mu} f(\boldsymbol{x}_k - \mu\,\boldsymbol{d}_k)\,.$$

Le nouveau problème est alors un problème de minimisation monodimensionnel et plusieurs options sont possibles [1, Ch. 3]. La première option disponible dans le logiciel repose sur une technique de recherche par *dichotomie*. Deux autres options reposant sur des approximations polynomiales du critère sont également disponibles. Elles sont détaillées ci-après et illustrées par la Fig. 2.

1.2.1 Trois stratégies

Dichotomie — Partant du point courant $\boldsymbol{x}_k^0$, la dichotomie repose sur la construction d'un nouveau point dans la direction $\boldsymbol{d}_k$ à partir de la dernière valeur connue μ_1 du pas :

$$\boldsymbol{x}_k^1 = \boldsymbol{x}_k^0 - \mu_1 \boldsymbol{d}_k \, .$$

Si $f(\boldsymbol{x}_k^1) < f(\boldsymbol{x}_k^0)$, on augmente μ (*e.g.*, $\mu_2 = 2.5\,\mu_1$) sinon on le réduit (*e.g.*, $\mu_2 = 0.5\,\mu_1$)[2]. On reconstruit alors un nouveau point et le processus est itéré. L'arrêt a lieu lorsque le pas devient trop faible ($\mu < \mu_\varepsilon$) ou lorsque le critère recommence à augmenter.

Interpolation quadratique — Dans ce cas on utilise le premier point $\boldsymbol{x}_k^0$ pour lequel on connaît le gradient et la valeur du critère et un deuxième point $\boldsymbol{x}_k^1 = \boldsymbol{x}_k^0 - \mu_0 \boldsymbol{d}_k$, généré à partir de la dernière valeur connue μ_0 de μ, où l'on ne connaît que la valeur du critère $f(\boldsymbol{x}_k^1)$. On identifie alors la fonction $g(\mu) = f(\boldsymbol{x}_k^0 - \mu \boldsymbol{d}_k)$ à une parabole $g(\mu) = a\mu^2 + b\mu + c$ dont le minimum fournit $\mu_1 = -b/2a$ et donc $\boldsymbol{x}_k^2 = \boldsymbol{x}_k^0 - \mu_1 \boldsymbol{d}_k$.

Interpolation cubique — Dans ce cas, on a un point où l'on connaît le gradient et la valeur du critère et deux points où l'on ne connaît que le critère (après une interpolation quadratique par exemple). On identifie la cubique $g(\mu) = a\mu^3 + b\mu^2 + c\mu + d$ dont le minimum (s'il existe) est solution de l'équation $3a\mu_2^2 + 2b\mu_2 + c = 0$. On obtient ainsi un nouveau point et la valeur du critère correspondante. S'il n'y a pas de minimum, on ne conserve que les résultats de l'interpolation quadratique.

1.2.2 Interpolation hybride

En pratique le logiciel effectue automatiquement l'adaptation du pas en combinant les deux approximations et la technique de dichotomie. Plus précisément, il propose une interpolation quadratique puis cubique et deux cas peuvent se présenter :

- si une des valeurs de μ obtenues après interpolations fournit un point meilleur on le garde,
- sinon on utilise la forme dichotomique pour effectuer la minimisation.

Par ailleurs, il peut arriver que l'interpolation conduise à des résultats instables (en particulier si les points sont trop proches) ou des valeurs inacceptables (les deux interpolations augmentent la valeur du critère). Dans ce cas le logiciel commute automatiquement sur une technique de dichotomie.

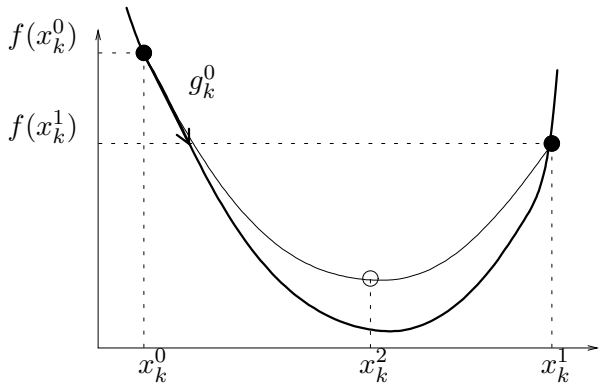
Remarque 1 – *Cette étape est importante pour deux raisons. Premièrement, elle permet d'exploiter au mieux le calcul du gradient (qui peut être coûteux). Deuxièmement, une bonne minimisation monodimensionnelle permet d'assurer des directions conjuguées de bonne qualité.*

Remarque 2 – *En revanche, dans des cas où le calcul du gradient n'est pas très coûteux, il peut être plus efficace de consacrer moins de temps à la minimisation de ligne (qui est alors moins précise) et d'itérer plus fréquemment le calcul des directions pseudo-conjuguées.*

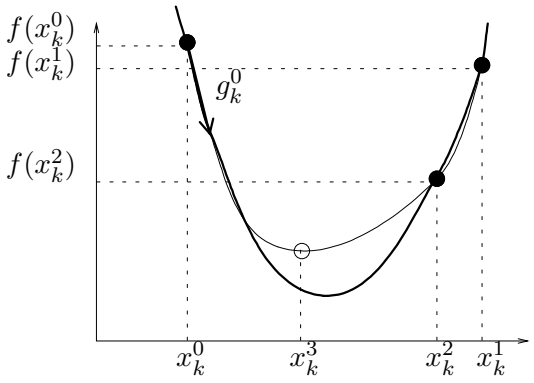
1.3 Conditions d'arrêt

Plusieurs conditions d'arrêt sont possibles, nous en avons retenu deux qui doivent être vérifiées simultanément pour que l'algorithme s'arrête. La première porte sur la norme $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|$ et plusieurs possibilités sont offertes : norme quadratique (éventuellement divisée par N) et norme infinie. La deuxième condition porte sur la décroissance du critère $f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k-1})$. Les deux quantités sont évaluées après minimisation de ligne (et non pendant) et l'algorithme se termine si elles passent en dessous d'un certain seuil. Enfin, il est également possible de fixer un nombre maximum d'itérations (nombre de calculs du gradient).

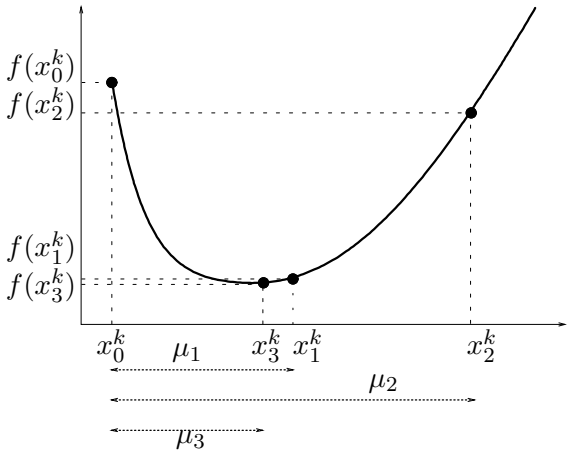
[2] Les valeurs 2.5 et 0.5 sont indicatives. Il est cependant recommandé de ne pas prendre deux valeurs dont le produit vaut 1 afi n d'éviter les oscillations.



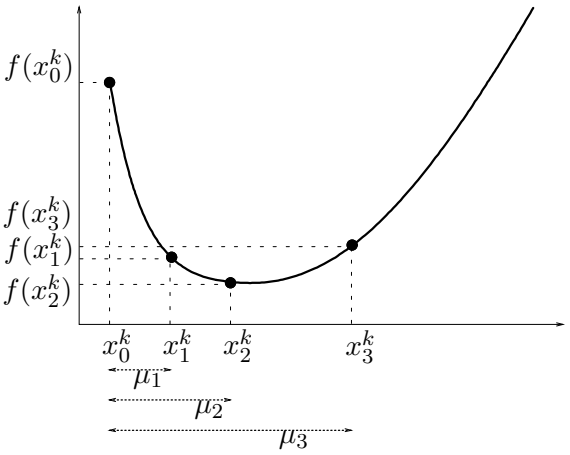
Interpolation quadratique



Interpolation cubique



Dichotomie : exemple 1



Dichotomie : exemple 2

FIG. 2 – Illustration des différents types de minimisation monodimensionnelle.

2 Codage *Matlab* et syntaxe d'appel

2.1 Codage

Le codage est réalisé à l'aide du logiciel de calcul *Matlab*, qui fonctionne sur diverses plates-formes (Linux, Unix, DOS, OS2, Mac). Le code contient environ 500 lignes réparties en trois fonctions *Matlab*.

1. `gpac` : c'est la fonction principale qui initialise l'algorithme et pilote les itérations jusqu'à ce que le test d'arrêt soit vérifié. Elle fait appel aux fonctions de calcul de direction et du pas.
2. `ChoixDir` définit la nouvelle direction de descente avec les éventuelles corrections ;
3. `Min1D` calcule le pas dans la direction choisie.

Le logiciel est suffisamment souple pour travailler avec des fonctions de la variable réelle ou complexe et qui peuvent être agencées sous forme de vecteur, de matrice ou de tableau multidimensionnels.

2.2 Syntaxe d'appel

La fonction possède une aide en ligne accessible par la commande `help gpac`.

Dans un souci de compatibilité et de transparence, les syntaxes d'appel de la fonction `gpac` sont fortement inspirées des syntaxes d'appel des fonctions de l'*Optimization Toolbox* de *Matlab*. La forme d'appel standard est la suivante :

```
X = gpac('fun',X0,Options,'grad')
```

Le logiciel fait alors appel à deux fonctions `fun.m` et `grad.m` qui renvoient respectivement les valeurs `f` du critère et `g` de son gradient en `x` : `f=fun(x)` et `g=grad(x)`. Sous cette forme, l'algorithme est initialisé en `X0` et le minimiseur fourni est `X`. Une seconde syntaxe :

```
X = gpac('fun',X0,Options,'grad',p1,p2,...)
```

très similaire à la précédente permet de passer des paramètres en plus de `x` aux fonctions `fun.m` et `grad.m`. Leur syntaxe est alors `f=fun(x,p1,p2,...)` et `g=grad(x,p1,p2,...)`. En plus de la variable optimale, deux autres sorties sont possibles :

```
[X,Options,Histo] = gpac('fun',X0,Options,'grad',p1,p2,...)
```

La variable de sortie `Options` contient en particulier le nombre d'itérations, le nombre de calculs du critère et sa valeur finale. La variable `Histo` donne accès à des informations plus détaillées concernant l'historique de la minimisation. `Histo` est une matrice possédant une ligne par itération et successivement dans les colonnes : le critère, les valeurs des deux quantités testées pour l'arrêt de l'algorithme et le temps *cpu.* Par exemple, la commande : `semilogy(Histo( :,1),Histo( :,4))` affiche les valeurs du critère en fonction du temps *cpu.* La commande : `loglog(Histo( :,1),1 :Histo(11))` affiche les valeurs du critère en fonction du nombre d'itérations.

Par ailleurs, au fur et à mesure des itérations, le logiciel peut afficher des informations sur son déroulement. L'affichage standard prend la forme suivante.

```
 Iter  NbreF      F         Pas      ErrF      ErrX   Min1D Adapte  Strat
  10    31   7.9453e+03  2.425e-02 5.81e-03  2.23e+00 Cubic   <-   PseudoC
  20    61   7.8644e+03  2.478e-02 1.09e-04  3.51e-01 Cubic   ->   PseudoC
  30    91   7.8636e+03  2.720e-02 1.22e-06  3.34e-02 Cubic   ->   PseudoC
```

Dans cet extrait, on retrouve de gauche à droite : le nombre d'itérations (nombre d'évaluations du gradient), le nombre d'évaluations de la fonction, la valeur du critère, la valeur du pas, les 2 grandeurs servant au test d'arrêt (sur f et sur x), le type de minimisation monodimensionnel, la technique d'adaptation du pas (réduction `<-` ou augmentation `->`) et le type de direction de descente (`PseudoC` pour pseudo conjuguée, `Vignes` pour la correction de Vignes, `Bissec` pour la bissectrice et `Grad` pour le gradient).

2.3 Vecteur d'options

Il est possible de modifier la plupart des paramètres de l'algorithme grâce au vecteur `Options` décrit dans le tableau de la page suivante.

Références

[1] J. Nocedal et S. J. Wright, *Numerical Optimization*, Series in Operations Research. Springer Verlag, New York, 2000.

[2] D. P. Bertsekas, *Nonlinear programming*, Athena Scientific, Belmont, MA, USA, 2nd edition, 1999.

[3] E. Walter et L. Pronzato, *Identification de modèles paramétriques à partir de données expérimentales*, Masson, Paris, 1994.

- `Options(1)` contrôle l'affi chage au cours de l'exécution du programme.
- `Options(2)` contrôle la précision sur $\boldsymbol{x}$, c'est le seuil sur $\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|$.
- `Options(3)` contrôle la précision sur $f(\boldsymbol{x})$, c'est le seuil sur $f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k-1})$.
- `Options(4)` permet de réinitialiser la correction, c-à-d relance l'algorithme du point courant.
  - `Options(4)=-1` ré-initialisation toutes les $N/12 + 3$ itérations.
  - `Options(4)=0` ne réinitialise jamais (*défaut*).
  - `Options(4)>0` ré-initialisation toutes les `Options(4)` itérations.
- `Options(5)` contrôle la stratégie de choix de la direction de descente.
  - `Options(5)=0` gradient simple à pas adaptatif.
  - `Options(5)=1` gradient à pas adaptatif avec correction de Vignes.
  - `Options(5)=2` gradient à pas adaptatif avec correction bissectrice.
  - `Options(5)=3` gradient pseudo conjugué de Polak-Ribieres (*défaut*).
- `Options(6)` norme utilisée pour le test d'arrêt.
  - `Options(6)=0` norme infi nie.
  - `Options(6)=1` norme quadratique (*défaut*).
  - `Options(6)=2` norme quadratique / nombre de variables.
- `Options(7)` méthode de minimisation en ligne.
  - `Options(7)=0` interpolation hybride, *i.e.*, quadratique/cubique/dichotomie (*défaut*).
  - `Options(7)=1` dichotomie.
- `Options(8)` valeur $f(\widehat{\boldsymbol{x}})$ (paramètre de sortie uniquement).
- `Options(9)` inutilisée.
- `Options(10)` nombre d'évaluations du gradient (paramètre de sortie uniquement).
- `Options(11)` nombre d'évaluations de la fonction (paramètre de sortie uniquement).
- `Options(12)` inutilisée.
- `Options(13)` inutilisée.
- `Options(14)` nombre maximum d'itérations.
- `Options(15)` valeur minimale du pas.
- `Options(16)` inutilisée.
- `Options(17)` inutilisée.
- `Options(18)` pas initial.
- `Options(19)` commande la sauvegarde de la solution courante toutes les `Options(19)` itérations.
- `Options(20)` la sauvegarde est réalisé dans le fi chier `gpac-sav` numéroté par `Options(20)`.
- `Options(21)` inutilisé.
- `Options(22)` permet d'imposer la positivité.

TAB. 1 – Table des options du logiciel.

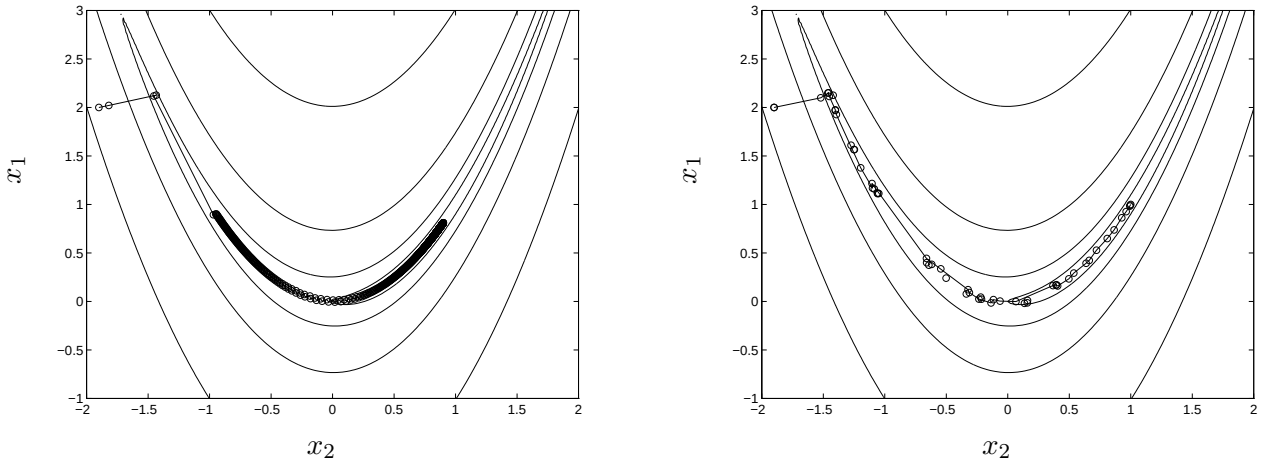


FIG. 3 – Illustration du déroulement de l'algorithme : à gauche stratégie du gradient simple et à droite stratégie de gradient conjugué (options par défaut).