Optimisation : méthodes numériques pour l'optimisation différentiable

Jérôme MALICK

1ère moitié de la partie 2 du cours d'optimisation

ENSIMAG 2A – version automne 2009, mise à jour en 2012

Objectifs de cette partie

Présentation des méthodes numériques pour résoudre

problèmes d'optimisation différentiables et sans contraintes

Situation "simple" mais suffisamment riche pour :

- introduire les idées générales de l'optimisation numérique
- les différents composants des algorithmes
- mêmes idées et les mêmes briques de base en :
  - optimisation avec contraintes
  - optimisation non-différentiable
- présenter des algorithmes efficaces utilisés en pratique
- focus sur les problèmes où l'objectif est une "boite noire"

Plan de la présentation

Minimiser une fonction

Problème :

Étant donné $f : \mathbb{R}^n \longrightarrow \mathbb{R}$

trouver $x^* \in \mathbb{R}^n$ tel que

$$f(x^*) \leqslant f(x) \quad \text{pour tout } x \in \mathbb{R}^n \tag{I}$$

ce qu'on écrit

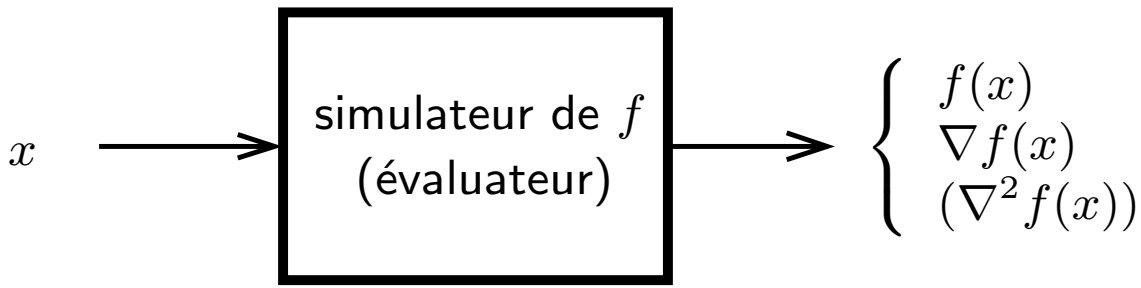
$$\min_{x \in \mathbb{R}^n} f(x) \qquad \text{ou} \qquad x^* = \operatorname*{argmin}_{x \in \mathbb{R}^n} f(x) \tag{P}$$

Remarques et limitations :

- dimension finie ($\longrightarrow$ algorithmes)
- existence d'un minimum x^*
- f est différentiable - de classe $C^{2,1}$ ($= C^2$ avec hessien loc. lipschitz)
- f peut être non convexe :
  - minimum global $\longrightarrow$ (I)
  - minimum local $\longrightarrow$ (I) valide que pour x proche de x^*

La fonction à minimiser

- Parfois : expression explicite (ex : apprentissage stats, chimie)
- Souvent : on ne dispose que d'une information ponctuelle
- On dispose d'un simulateur de la fonction (fourni par l'utilisateur)



- Exemple en météo : simulateur = une boîte qui calcule la variable d'état p_x = résolution de l'EDP (E)
- Conséquence 1 : critère pragmatique de la qualité d'un algorithme

nombre d'appels au simulateur

- Conséquence 2 : soit x, trouver y tel que $f(y) < f(x)$ non trivial

"Calculer" le minimum

- De temps en temps : résoudre le problème (P) explicitement
  = expression explicite de x^*
- Conditions d'optimalité :

$$\begin{array}{rcl} x^* \text{ minimum (local ou global)} & \Longrightarrow & \nabla f(x^*) = 0 \quad \text{et} \quad \nabla^2 f(x^*) \succcurlyeq 0 \\ x \text{ minimum local} & \Longleftarrow & \nabla f(x) = 0 \quad \text{et} \quad \nabla^2 f(x) \succ 0 \end{array}$$

- Situation usuelle : algorithme pour résoudre approximativement (P)
- Algorithme génère une suite $(x_k)_{k\in\mathbb{N}}$ dont on peut exiger :
  - $x_k \to \bar{x}$ minimum (local)
  - $x_k \to \bar{x}$ avec $\nabla f(\bar{x}) = 0$
  - $\liminf \|\nabla f(x_k)\| = 0$

  on dit qu'il y a convergence de l'algorithme, que la suite est minimisante
- Critère d'arrêt pratique : $\boxed{\|\nabla f(x_k)\| \leqslant \varepsilon}$

$$\|\nabla f(x_k)\| \leqslant \varepsilon \|\nabla f(x_0)\|$$

Qualité de la convergence

Un algorithme d'optimisation est aussi jugé sur les propriétés (théoriques) de $(x_k)_k$, en particulier propriété de convergence

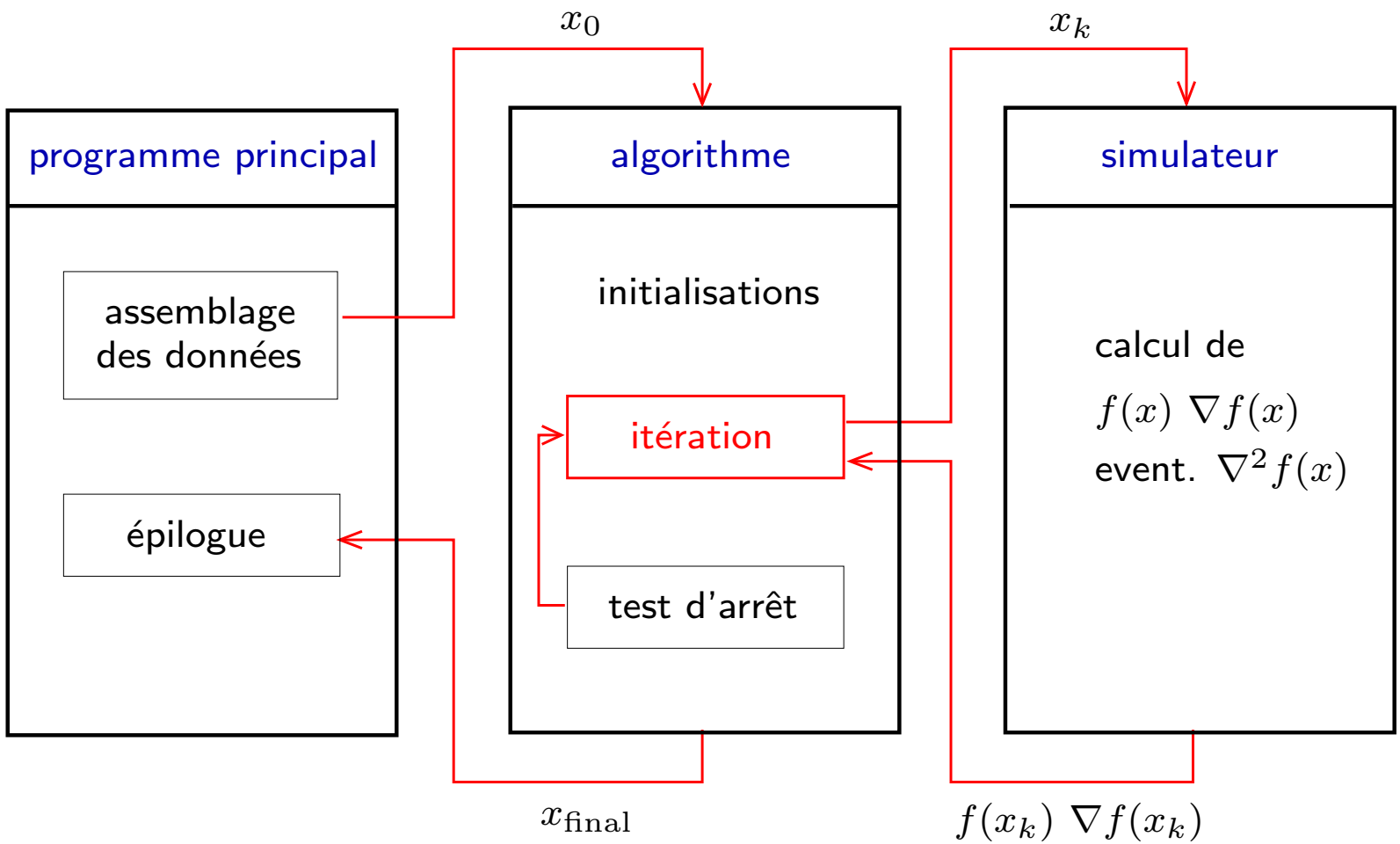
- Convergence : $\liminf \|\nabla f(x_k)\| = 0$
- Convergence globale : pour tout point initial $x_0 \in \mathbb{R}^n$
- Convergence locale : $x_k \to \bar{x}$ à quelle vitesse ?
  On compare les algorithmes (théorie) en regardant

$$q_k := \frac{\|x_{k+1} - \bar{x}\|}{\|x_k - \bar{x}\|}$$

Terminologie :

$q_k \to 1$	sous-linéaire
$q_k \leqslant r < 1$	linéaire
$q_k \to 0$	super-linéaire
$q_k \simeq \|x_k - \bar{x}\|$	quadratique

Structure d'un algorithme d'optimisation



Itération d'un algorithme d'optimisation

Une itération d'un algorithme d'optimisation consiste en

$$\boxed{\text{estimation} + \text{correction}}$$

L'itération k (à l'itéré courant x_k) se décompose en

- **Estimation :** construire un modèle local de la fonction et calculer $\widetilde{x}_k$
- **Correction :** en jouant sur un paramètre t, trouver un "bon" $\widetilde{x}_k(t)$ en dialoguant avec le simulateur de la fonction $\longrightarrow$ globalisation
- **Objectif :** pour obtenir $x_{k+1} = \widetilde{x}_k(t)$ satisfaisant en particulier

$$\boxed{f(x_{k+1}) < f(x_k)}$$

$\longrightarrow$ privilège des problèmes d'optimisation...

Remarque : annulation vs minimisation

- Résoudre les équations

$$g(x) = 0 \qquad \text{avec} \quad g\colon \mathbb{R}^n \to \mathbb{R}^n$$

  permet de résoudre $\min f(x)$! Considérer $g(x) := \nabla f(x)$

- Minimiser les fonctions permet aussi de résoudre $g(x) = 0$
  Considérer

$$\min \quad f(x) := \|g(x)\|^2$$

- Certes en théorie... mais en pratique ?
  L'optimisation est plus agréable car on a l'objectif

$$\boxed{f(x_{k+1}) < f(x_k)}$$

  $\longrightarrow$ globaliser les algorithmes

- Morale : pour l'estimation c'est pareil, mais la clé est la correction

Estimation : principe

Idée : on ne sait pas résoudre le problème

$$\boxed{\min_{x\in\mathbb{R}^n} \quad f(x)} \qquad \text{(P)}$$

on va le remplacer par un problème plus facile

À l'itéré courant x_k :

- on considère un modèle $\widetilde{f}_k$ de f
- on résout le problème simplifié

$$\boxed{\min_{d\in\mathbb{R}^n} \quad \widetilde{f}_k(x_k+d)} \qquad \text{(P}_k\text{)}$$

- on espère que : si $\widetilde{f}_k$ est un "bon" modèle de f, alors l'estimation

$$\widetilde{x}_k = x_k + d_k \qquad \text{avec} \qquad d_k := \operatorname*{argmin}_{d\in\mathbb{R}^n} \widetilde{f}_k(x_k+d)$$

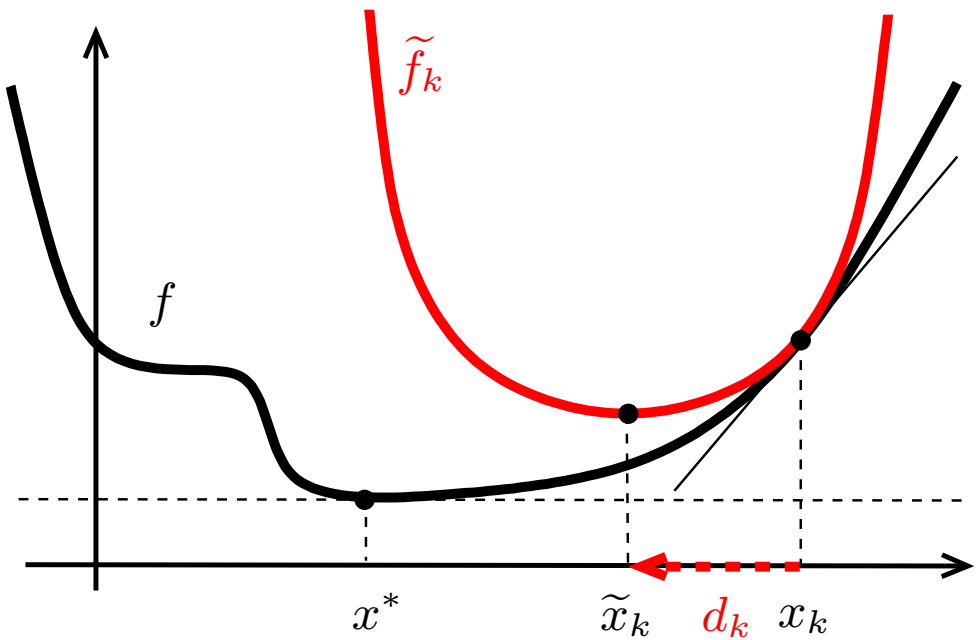
sera une bonne approximation de la solution

Exemple : modèle de Newton

Modèle local du 2e ordre

$$\widetilde{f}_k(x_k + d) := f(x_k) + \nabla f(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 f(x_k)\, d$$

Taylor-Young : $f(x_k + d) = \widetilde{f}_k(x_k + d) + \mathrm{o}(\|d\|^2)$



Modèle quadratique

- Choix de $\widetilde{f}_k$ en général : dépend des propriétés de f !
  $\longrightarrow$ équilibre entre : précision $\perp$ simplicité

- Choix de $\widetilde{f}_k$ ici : f différentiable $\longrightarrow$ modèle quadratique

$$\boxed{\widetilde{f}_k(x_k + d) := f(x_k) + \nabla f(x_k)^\top d + \frac{1}{2}\, d^\top M_k\, d}$$

  $\widetilde{f}_k$ coïncide avec f en x_k au 1er ordre, et 2e ordre si $M_k = \nabla^2 f(x_k)$

- Choix de M_k (selon disponibilité de $\nabla^2 f(x_k)$)

$M_k = \nabla^2 f(x_k)$	Newton
$M_k = \mathrm{Id}$	gradient
$M_k = \mathrm{Id} + \text{rang } 2$	gradient conjugué
$M_k \simeq \nabla^2 f(x_k)$	(quasi-, Gauss-) Newton

  $\longrightarrow$ préférence pour des méthodes du 2nd ordre (X-Newton)

Méthode du gradient

Modèle local avec $M_k = \mathrm{Id}$ constante

$$\widetilde{f}_k(x_k + d) = f(x_k) + \nabla f(x_k)^\top d + \frac{1}{2}\|d\|^2$$

Le problème de l'estimation devient

$$\min_{d \in \mathbb{R}^n} \quad \nabla f(x_k)^\top d + \frac{1}{2}\|d\|^2$$

Alors d_k est caractérisé par l'équation en d

$$\nabla f(x_k) + d = 0$$

L'itération du gradient est alors

$$x_{k+1} = (\widetilde{x}_k =)\, x_k - \nabla f(x_k)$$

Problème de myopie et de scaling $\longrightarrow$ mauvaise méthode (cf. TP1)

Méthode de Newton

Modèle local au 2e ordre

$$\widetilde{f}_k(x_k + d) = f(x_k) + \nabla f(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 f(x_k)\, d$$

Le problème de l'estimation devient

$$\min_{d \in \mathbb{R}^n} \quad \nabla f(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 f(x_k)\, d$$

Si $\nabla^2 f(x_k) \succcurlyeq 0$, alors d_k est caractérisé par l'équation en d

$$\nabla f(x_k) + \nabla^2 f(x_k)\, d = 0$$

Si de plus $\nabla^2 f(x_k) \succ 0$

$$d_k = -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

L'estimation "de Newton" est alors

$$\widetilde{x}_k = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

Newton : avantage et inconvénients

Itération de Newton $x_{k+1} = (\widetilde{x}_k =)\ x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$

Théorème (Convergence locale de Newton)

Si f est de classe $C^{2,1}$ et x_0 proche d'un minimum x^* vérifiant $\nabla^2 f(x^*) \succ 0$, la suite est bien définie et la convergence est quadratique

$$\|x_{k+1} - x^*\| \leqslant \kappa \|x_k - x^*\|^2$$

Mais : Newton diverge aussi rapidement $\longrightarrow$ besoin de globalisation

Exemple : pour $f(x) := \sqrt{1 + x^2}$ (convexe, C^∞)

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} = -{x_k}^3$$

Inconvénients :

1. (Pouvoir) calculer les dérivées secondes
2. Résoudre un système linéaire $n \times n$ (stocker une matrice $n \times n$)
3. Exige $\nabla^2 f(x) \succ 0$

Quasi-Newton

$\longrightarrow$ régler tous ces problèmes en même temps (cf. TP3)

Idée : pendant les itérations successives k

en même temps que $x_k \longrightarrow \bar{x}$

construire $M_k \longrightarrow \nabla^2 f(\bar{x})$

encore mieux $W_k \longrightarrow [\nabla^2 f(\bar{x})]^{-1}$

en n'utilisant que l'information du 1er ordre $\nabla f(x_\ell)$ pour $\ell = 0, \ldots, k$

Schéma : Initialisation de W_0, puis boucle sur k

- Calcul de $d_k = -W_k \nabla f(x_k)$
- Estimation $\widetilde{x}_k = x_k + d_k$
- Correction $x_{k+1} = x_k + t_k d_k$ (recherche linéaire de Wolfe)
- Mise à jour de W_k (??)

Quasi-Newton : mise à jour de W_k

Contraintes : étant donné W_k, on exige de W_{k+1}

- symétrique
- définie positive ($\Longrightarrow$ $d_k = -W_k \nabla f(x_k)$ direction de descente)
- proche de W_k (stabilité) $\longrightarrow$ plusieurs méthodes de quasi-Newton
- approchant $[\nabla^2 f]^{-1}$ $\longrightarrow$ équation de quasi-Newton

Équation : avec x_k et x_{k+1}, on pose

$$s_k := x_{k+1} - x_k \qquad y_k := \nabla f(x_{k+1}) - \nabla f(x_k)$$

Intégration de la dérivée (fonction $t \mapsto \nabla f(x_k + t s_k)$)

$$y_k = \left[\int_0^1 \nabla^2 f(x_k + t\, s_k) \right] s_k$$

Équation de quasi-Newton

$$y_k = M_{k+1} s_k \qquad s_k = W_{k+1} y_k$$

Quasi-Newton : BFGS

Il existe plusieurs mises à jour : si la stabilité correspond à minimiser la distance, alors : BFGS (C. Broyden, R. Fletcher, D. Goldfarb, D. Shanno)

$$W_{k+1} := W_k - \frac{s_k y_k^\top W_k + W_k y_k s_k^\top}{y_k^\top s_k} + \left[1 + \frac{y_k^\top W_k y_k}{y_k^\top s_k}\right] \frac{s_k s_k^\top}{y_k^\top s_k}$$

(et formules similaires pour M_k)

Propriétés :

- On montre que : W_{k+1} symétrique, vérifie l'équation
- $W_{k+1} \succ 0 \iff y_k^\top s_k > 0$ ($\Longleftarrow$ recherche linéaire)
- $W_{k+1} - W_k$ est de rang 2 (petit)
- mieux vaut commencer avec un bon W_0 (défaut : $W_0 = \alpha \mathrm{Id}$)
- BFGS commence comme le gradient puis incorpore du 2nd ordre...
- BFGS a de bons résultats de convergence (cf après)

  $\longrightarrow$ convergence superlinéaire $\quad \|x_{k+1} - x^*\| \leqslant \mathrm{o}(\|x_k - x^*\|)$

Estimation : conclusion

- L'estimation utilise un modèle quadratique

$$\widetilde{f}_k(x_k + d) \;=\; f(x_k) + \nabla f(x_k)^\top d + \frac{1}{2}\, d^\top M_k\, d$$

  et donne un $\widetilde{x}_k$ - qui demande à être stabilisé

- Choix de M_k : règle grossière
  - si on a facilement $\nabla^2 f(x_k)$, on prend $M_k = \nabla^2 f(x_k)$ (Newton)
  - si on a uniquement $\nabla f(x_k)$, on prend $M_k =$ BFGS (quasi-Newton)

  $\longrightarrow$ convergence locale rapide

- Pas si simple !
  La construction de M_k peut être affinée selon la structure de f...

- Exemple (important) : problèmes de moindres carrés

Exemple : moindres carrés

Somme de carrés $\min_{x \in \mathbb{R}^n} \quad f(x) := \frac{1}{2} \sum_{j=1}^{m} {f_j}^2(x)$

Gradient $\nabla f(x) = \sum_{j=1}^{m} f_j(x) \nabla f_j(x)$

Hessien $\nabla^2 f(x) = \underbrace{\sum_{j=1}^{m} \nabla f_j(x) \nabla f_j(x)^\top}_{G(x)} + \sum_{j=1}^{m} f_j(x) \nabla^2 f_j(x)$

En x_k, si on n'a que les $\nabla f_j(x_k)$ (et pas les $\nabla^2 f_j(x_k)$), on peut

$$M_k = G(x_k) \simeq \nabla^2 f(x_k)$$

$\longrightarrow$ Méthode de Gauss-Newton (à stabiliser par régions de confiance)

L'approximation est "bonne" si

- on est proche du minimum i.e. $f_j(x) \simeq 0$ pour tout j
- les f_j sont peu non-linéaires i.e. $\nabla^2 f_j(x) \simeq 0$ pour tout j

Correction : principe

- Questions
  - $\widetilde{x}_k = \operatorname{argmin} \widetilde{f}_k$ existe ?
  - a-t-on $f(\widetilde{x}_k) < f(x_k)$?
- Le problème est que le modèle $\widetilde{f}_k$
  - modèle simplifié donc imparfait $\longrightarrow$ besoin de stabiliser
  - valable localement $\longrightarrow$ besoin de globaliser
- Solution : observer le vrai comportement de la fonction f
- Comment ? (puisqu'on ne dispose que d'une info ponctuelle)
  - essais et errreurs
  - dialogue intense avec le simulateur
  - réglage d'un paramètre réel t pour avoir : $f(\widetilde{x}_k(t)) < f(x_k)$
- 2 techniques
  1. recherche linéaire (cf. TP2)
  2. régions de confiance

Bon itéré sur une demi-droite ?

- Données : $x_k \in \mathbb{R}^n$ et $d_k = \widetilde{x}_k - x_k \in \mathbb{R}^n$ (après estimation)
- On cherche un "bon" itéré suivant parmi : $\boxed{\widetilde{x}_k(t) = x_k - t\, d_k}$
- Besoin d'avoir une bonne direction : direction de descente

$$\nabla f(x_k)^\top d_k < 0 \quad \Longrightarrow \quad (\exists\, t :)\ f(x_k + t\, d_k) < f(x_k)$$

- 1ère idée : $\min\limits_{t \geqslant 0}\ q(t) := f(x_k + t\, d_k)$

  $\longrightarrow$ difficile et inutile (si d_k est mauvaise)
- Objectif : avoir un "bon" t, pour pas cher = peu d'appels
- Principe : à l'aide d'un test 3 sorties :

  (a) t satisfaisant **(b)** t est trop grand **(c)** t est trop petit

  on fait une succession d'essais

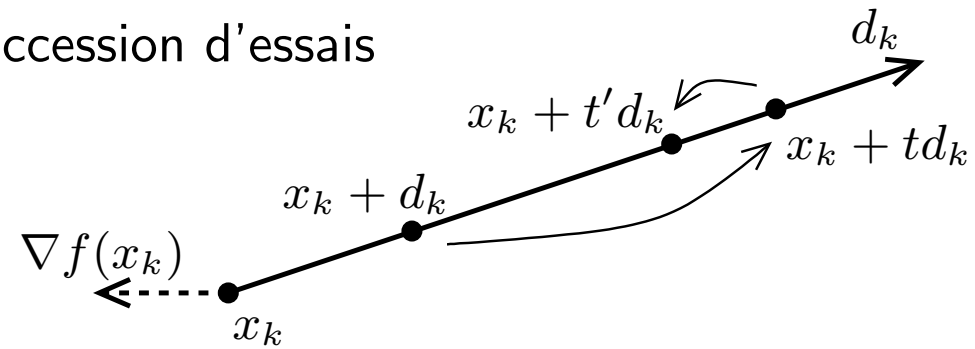


Schéma du dialogue avec le simulateur

Avec le test à 3 sorties, on teste des $t \in [t_G, t_D]$ (un intervalle de confiance que l'on réduit successivement)

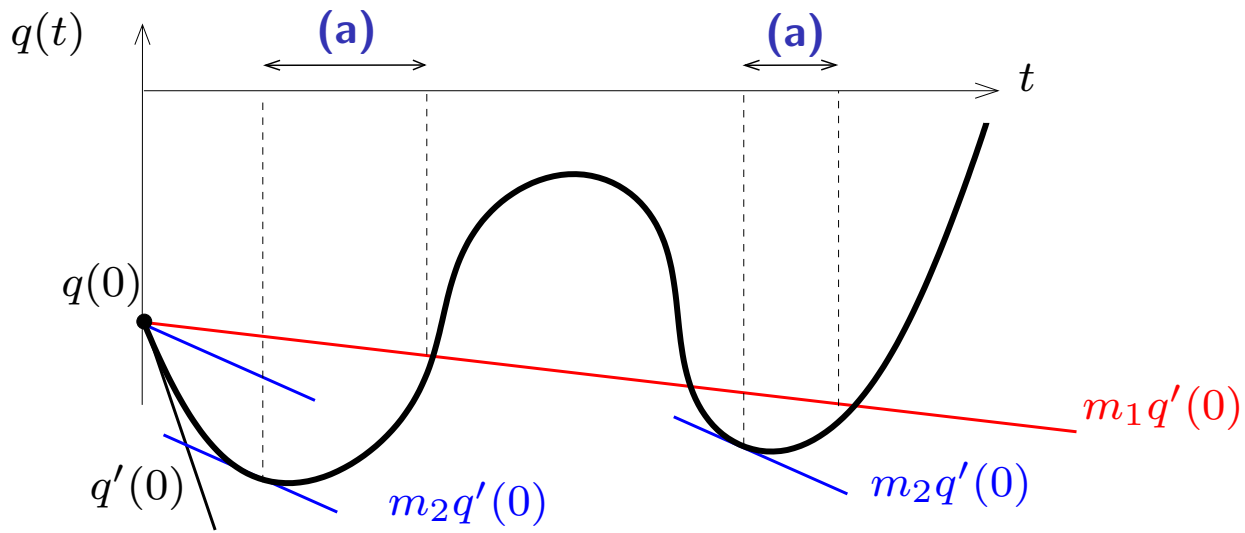
Processus :

1. Initialisation : $t_G = 0$, $t_D = +\infty$ et $t = 1$
2. Test de t :
   - si **(a)** : OK c'est fini
   - si **(b)** : on pose $t_D = t$ et on passe en 3
   - si **(c)** : on pose $t_G = t$ et on passe en 3
3. Choix d'un nouveau t :
   - Extrapolation : si $t_D = +\infty$, on choisit $t > t_G$ (ex : $t = 10t_G$)
   - Interpolation : sinon on choisit $t \in]t_G, t_D[$ (ex : $t = (t_G + t_D)/2$)

   et on boucle en 2

Test de Wolfe

Avec 2 constantes $0 < m_1 < m_2 < 1$



(b) t est trop grand si $q(t) > q(0) + m_1 t\, q'(0)$

(c) t est trop petit si $q(t) \leqslant q(0) + m_1 t\, q'(0)$ et $q'(t) < m_2\, q'(0)$

(a) t est satisfaisant si $q(t) \leqslant q(0) + m_1 t\, q'(0)$ et $q'(t) \geqslant m_2\, q'(0)$

Coût d'un test = un appel du simulateur (en $\widetilde{x}_k(t) = x_k + t d_k$)

$$q(t) = f(\widetilde{x}_k(t)) \qquad q'(t) = \nabla f(\widetilde{x}_k(t))^\top d_k$$

Convergence

Théorème (Consistence de la recherche linéaire de Wolfe)

Soit f de classe C^1 et d_k une direction de descente en x_k. Si q est minorée, alors la recherche linéaire s'arrête en nombre fini d'itérations.

Théorème (BFGS + Wolfe)

Soit f de classe C^2 (minorée sur $\mathbb{R}^n$). Supposons que la suite $(x_k)_k$ générée par BFGS stabilisé par la recherche linéaire de Wolfe est bornée.

Convergence globale : si f convexe, alors tout point d'accumulation $\bar{x}$ vérifie

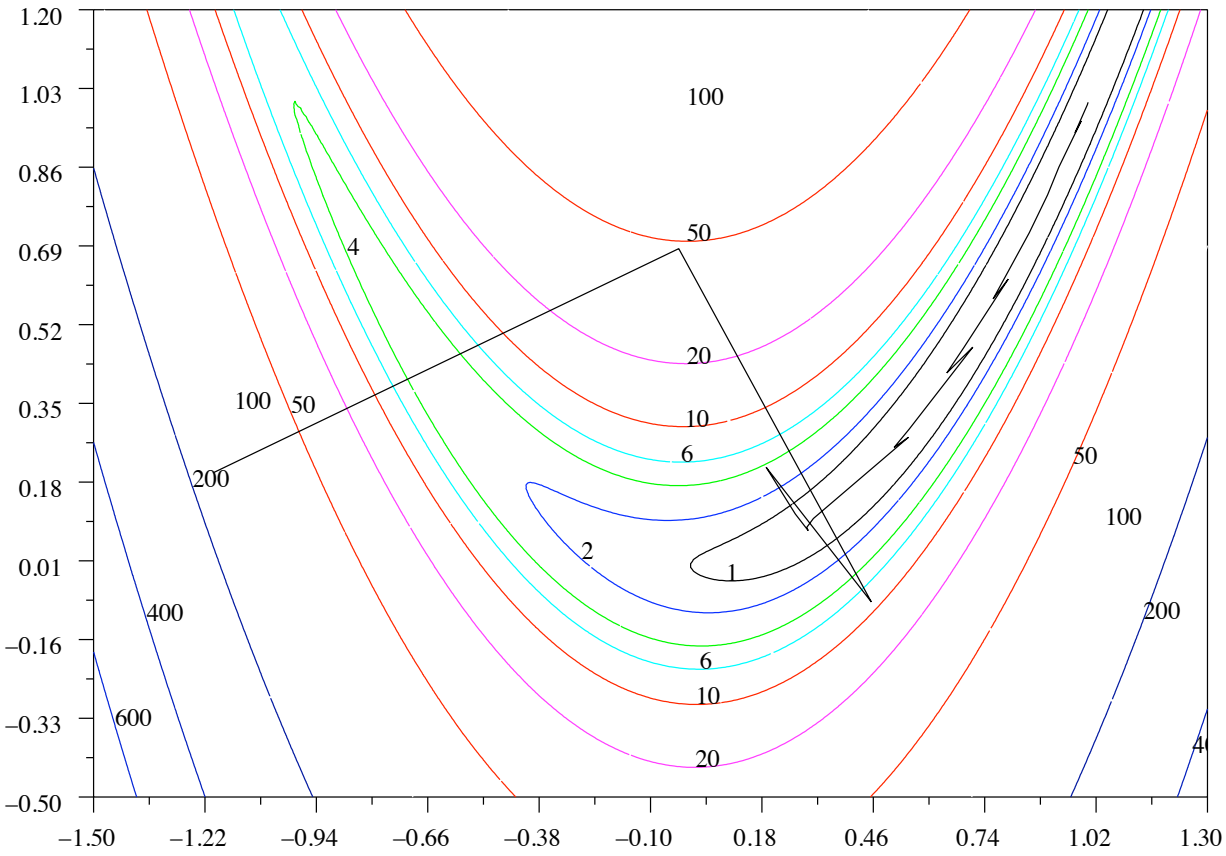
$$\nabla f(\bar{x}) = 0$$

Convergence locale : si de plus $f \in C^{2,1}$ et $\nabla^2 f(\bar{x}) \succ 0$, alors la convergence est super-linéaire

$$\|x_{k+1} - \bar{x}\| = o(\|x_k - \bar{x}\|)$$

Illustration sur un exemple classique

Fonction de Rosenbrock : $f(x,y) := (1-x)^2 + 100(y-x^2)^2$



Correction par région de confiance

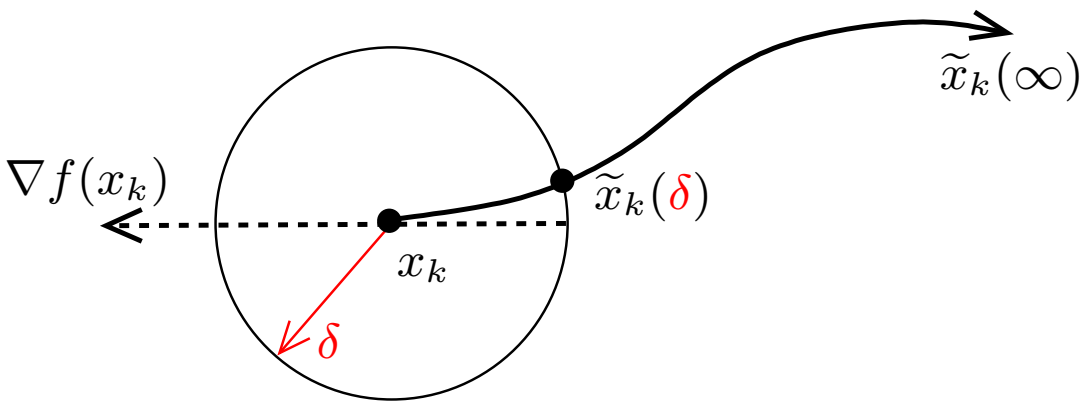
Globalisation par recherche linéaire

- bonne pour BFGS + Wolfe : simple et robuste
- inadaptée pour Newton si $\nabla^2 f(x) \not\succ 0$
- Naturelle quand d_k est normalisée (1er ordre)
  Pourquoi une demi-droite ? (2nd ordre)

Idée : incorporer la globalisation dans le modèle

$$\widetilde{x}_k(\delta) = x_k + d_k(\delta) \qquad \text{avec} \qquad d_k(\delta) := \underset{\|d\| \leqslant \delta}{\operatorname{argmin}} \ \widetilde{f}_k(x_k + d)$$

Région de confiance $\simeq$ recherche "curviligne"



Gestion du rayon

On cherche l'itéré suivant dans une région de confiance de rayon δ

$$\left\{ \begin{array}{c} \min \quad \widetilde{f}_k(x_k + d) \\ \|d\| \leqslant \delta \end{array} \right. \qquad (\mathrm{RC}_\delta)$$

La question est donc de trouver un "bon" rayon $\delta > 0$

Confronter $\widetilde{x}_k(\delta)$ (qui vient du modèle) à la vraie fonction f

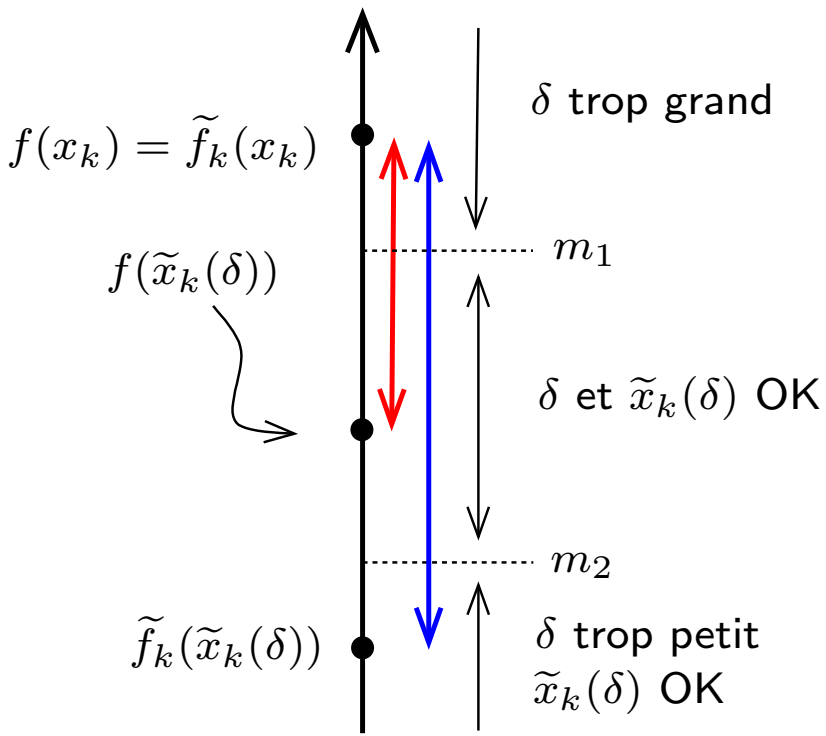
Essais/erreurs : augmenter/diminuer la taille δ de la région

Schéma similaire à la recherche linéaire

- calcul de $\widetilde{x}_k(\delta)$ solution de (RC_δ)
- test si $\widetilde{x}_k(\delta)$ est bon (avec $f(\widetilde{x}_k(\delta))$, $\nabla f(\widetilde{x}_k(\delta))$,...)
- accepter $\widetilde{x}_k(\delta)$ - ou corriger δ et recommencer
- réinitialiser δ après itération

Test du rayon δ

2 constantes $0 < m_1 < m_2 < 1$



- on espère

$$\begin{aligned} \Delta_{\mathrm{e}} &:= \widetilde{f}_k(x_k) - \widetilde{f}_k(\widetilde{x}_k(\delta)) \\ &= f(x_k) - \widetilde{f}_k(\widetilde{x}_k(\delta)) \end{aligned}$$

- on observe

$$\Delta_{\mathrm{o}} := f(x_k) - f(\widetilde{x}_k(\delta))$$

- on exige $\Delta_{\mathrm{o}} \geqslant m_1 \Delta_{\mathrm{e}}$
- mais $\Delta_{\mathrm{o}} \geqslant m_2 \Delta_{\mathrm{e}}$ révèle :

δ trop petit

Résolution de (RC_δ)

Essentiel de pouvoir résoudre efficacement

$$\left\{ \begin{array}{ll} \min & \widetilde{f}_k(x_k + d) \\ & \|d\| \leqslant \delta \end{array} \right. \qquad (\mathrm{RC}_\delta)$$

pour tester les δ

Miracle : si $\widetilde{f}_k$ modèle quadratique (avec M_k), alors on peut résoudre (RC_δ) par dualité (cf suite du cours + exam 07). En bref, la solution est

$$d(\mu^*) := -(M_k + \mu^* \mathrm{Id})^{-1} \nabla f(x_k)$$

où la variable duale optimale $\mu^* \geqslant 0$ (et $\mu > \lambda_{\min}(M_k)$) est solution de

$$\|d(\mu)\| = \delta \qquad (1/\|d(\mu)\| = 1/\delta)$$

$\longrightarrow$ résolution par une méthode de Newton en une dimension !

Convergence

Théorème (Convergence globale et vitesse locale)

Soit f de classe $C^{2,1}$ sur $\mathbb{R}^n$. Si $\widetilde{f}_k$ est le modèle de Newton et si la suite $(x_k)_k$ générée par la méthode de régions de confiance est bornée, alors tout point d'accumulation $\bar{x}$ vérifie

$$\nabla f(\bar{x}) = 0 \qquad \text{et} \qquad \nabla^2 f(\bar{x}) \succcurlyeq 0$$

De plus, la convergence vers $\bar{x}$ est quadratique

$$\|x_{k+1} - \bar{x}\| \leqslant \kappa \|x_k - \bar{x}\|^2$$

très bonnes propriétés !

$\longrightarrow$ avec méthode d'estimation avec 2e ordre

$\longrightarrow$ plus évoluée, plus technique que recherche linéaire

Correction : conclusion

recherche linéaire $\perp$ régions de confiance

$\longrightarrow$ diffèrent dans l'ordre entre direction et distance pour aller à x_{k+1}

- méthodes de RL : direction d_k - puis distance t_k
- méthodes de RC : distance maximum δ_k - puis direction et pas x_k

recherche linéaire $\simeq$ régions de confiance

$\longrightarrow$ basées sur mêmes principes :

- trouver un "bon" paramètre réel t ou δ
- (pas besoin de trouver le "meilleur" !)
- dialogue intense avec le simulateur (essais/erreurs)
- autres méthodes : Levenberg-Marquardt – méthodes proximales

Météo : modélisation

- Variable : $p(z,t)$ état de l'atmosphère au point $z \in \mathbb{R}^3$ au temps $t \in [0,7]$ (pression, vitesse du vent, humidité...)
- Évolution : Navier-Stokes ou approximation - avec un opérateur Φ

$$\frac{\partial p}{\partial t}(z,t) = \Phi(p(z,t)) \qquad \text{(E)}$$

- Résolution : il "suffit" d'intégrer l'équation à partir des
  - conditions initiales (?)
  - conditions aux limites (pb non considéré ici)

  $\longrightarrow$ solution unique
- Question : estimer les conditions initiales $p(\cdot,0)$?
  $\longrightarrow$ problème d'optimisation
- Information : observations pendant le jour précédent $\omega_i := p(z_i,t_i)$ (on note : $\omega := \{\omega_i\}_{i\in I}$)

Météo : optimisation

⟶ Problème de commande optimale (variable $p\colon \mathbb{R}^4 \to \mathbb{R}^d$)

$$\begin{cases} \min \quad \|p-\omega\|^2 \\ \quad p \text{ satisfait } (\mathrm{E}) \end{cases}$$

p sur $[-1,0]$ est la variable d'état, (E) est l'équation d'état
problème dur avec contraintes !

⟶ Problème d'optimisation sans contraintes

- État de l'atmosphère à $t=-1$: on note $x(z) := p(z,-1)$
  (x est la variable de commande ou contrôle)
- Point clé : à x donné, la résolution de (E) fournit $p = p_x$ sur $[-1,0]$
- L'optimisation se réécrit

$$\min_x \quad f(x) := \|p_x - \omega\|^2$$

- Le calcul de $f(x)$ demande d'intégrer (E) sur $[-1,0]$

Météo : prévision

$\longrightarrow$ Schéma de résolution du problème de prévision (après discrétisation)

1. résoudre le problème d'optimisation sans contrainte
2. avec une succession d'intégrations de (E) entre $[-1, 0]$
3. le meilleur contrôle $\bar{x}$ (et l'état associé $p_{\bar{x}}$) donne la meilleure condition initiale $p_{\bar{x}}(\cdot, 0)$
4. avec laquelle on résout (E) sur $[0, 7]$ pour obtenir la prévision

$\longrightarrow$ Alors, concrètement ?

- Méthode proposée par Ph. Courtier (Ponts) et F.-X. Ledimet (LJK) dans les années 80
- En particulier : Centre Européen de Météorologie (UK)
- Pbs avec $10^6 - 10^7$ variables (après discrétisation)
- Partie optimisation : algorithme de quasi-Newton avec mémoire limitée de J.-Ch. Gilbert (INRIA) et C. Lemaréchal (INRIA, LJK)

Problématique

- Les modèles quadratiques manipulent M_k, matrices de taille n symétriques ($n(n+1)/2$ éléments)
- Si $n \sim 10^6$ (ex : météo), difficultés en pratique à stocker M_k en mémoire puis y accéder :

$$\boxed{\text{temps de calcul} \ll \text{temps accès mémoire (swapping)}}$$

- 2 méthodes
  1. Newton inexact
  2. quasi-Newton à mémoire limitée
- En pratique : QN n'a besoin que des $\nabla f(x_k)$. Pour Newton inexact, besoin d'une fonction qui calcule le produit matrice×vecteur

$$d \in \mathbb{R}^n \rightarrow \text{calculs} \rightarrow M_k d \in \mathbb{R}^n$$

Newton inexact : principe

- Le pas de Newton de base = résoudre le système

$$\nabla^2 f(x_k)\, d = -\nabla f(x_k) \qquad \text{(N)}$$

- **Idée 1 :** ne pas résoudre le système (N) exactement
  Pourquoi ? Ça ne sert à rien d'avoir un $\widetilde{x}_k$ précis si on est loin de x^* !
  On arrête quand

$$\|\nabla^2 f(x_k)\, d + \nabla f(x_k)\| \leqslant \eta_k \|\nabla f(x_k)\|$$

- **Idée 2 :** résoudre (N) par gradient conjugué
  - diminue le résidu $r_k := \nabla^2 f(x_k)\, d + \nabla f(x_k)$
  - utilise uniquement les produits matrice×vecteur

- En bref : une itération de Newton inexact

calcul (approx.) $[\nabla^2 f(x_k)]^{-1}\nabla f(x_k)$ sans stocker $\nabla^2 f(x_k)$

Newton inexact : convergence

Théorème (Convergence locale quadratique)

Soit f de classe $C^{2,1}$. Si x_0 est proche d'un minimum x^* tel que $\nabla^2 f(x^*) \succ 0$, alors la suite x_k générée par Newton inexact converge vers x^*. Si on choisit

$$\eta_k = \min\{1/2, \|\nabla f(x_k)\|\}$$

la convergence est quadratique

$$\|x_{k+1} - x^*\| \leqslant \kappa \|x_k - x^*\|^2$$

$\longrightarrow$ Newton inexact garde les bonnes propriétés locales de Newton

$\longrightarrow$ Globalisation par

- recherche linéaire (si GC voit une courbure négative, $d_k = -\nabla f(x_k)$)
- régions de confiance (idem + contrainte $\|d\| \leqslant \delta$)

Quasi-Newton : grande taille

- Si $n \sim 10^6$, stocker W_k en mémoire puis y accéder.
- W_k comporte $n(n+1)/2$ termes...
  ... mais ne dépend que de $2k$ vecteurs (les y_ℓ et s_ℓ) !

$$W_{\ell+1} \;=\; \mathrm{BFGS}(W_\ell, s_\ell, y_\ell)$$

- Si l'itération $k << n$, on peut tirer parti de cette information
- Étant donnés
  - W_0
  - l'expression $W_{\ell+1} = \mathrm{BFGS}(W_\ell, s_\ell, y_\ell)$
  - le gradient $\nabla f(x_k)$

  on montre qu'on a une moulinette

calculer $W_k \nabla f(x_k)$ sans stocker W_k

Quasi-Newton à mémoire limitée

En pratique : l'utilisateur donne m le nombre maximal de couples (s_ℓ, y_ℓ) qu'on accepte de stocker

Pour les itérations $k = 1, ..., m$: OK

Pour l'itération $k > m$:

- Données
  - x_k
  - W_k^0 diagonale (ex le plus simple : $W_k^0 = \alpha \mathrm{Id}$)
  - m couples $(s_{k-m}, y_{k-m}), \ldots, (s_{k-1}, y_{k-1})$
- Calculs
  - estimation : calcul $d_k = W_k \nabla f(x_k)$ par la moulinette
  - correction : calcul $x_{k+1} = x_k + t_k d_k$ par Wolfe
- Mise à jour
  - on jette la plus vieille paire (s_{k-m}, y_{k-m})
  - on calcule la nouvelle (s_k, y_k)
  - on définit W_{k+1}^0 (hot restart)

Illustration de l'impact de m

Exemple issu d'un pb réel ("écoulement transsonique")

Taille $n = 403 \longrightarrow$ on peut faire varier la mémoire limitée $m \in \{0, \ldots, n\}$

m	nb simuls	
0	$+\infty$	gradient
1	795	grad. conj.
2	554	
3	510	
4	492	
5	423	
10	421	
20	354	
30	343	best choice ?
60	344	
403	343	pur BFGS

Limiter la mémoire pour les problèmes de grandes tailles

Newton inexact :

- Meilleure compréhension théorique de Newton inexact
- Utilise produit matrice hessienne $\times$ vecteur
- Utilisée pour résoudre des problèmes de calibration

Quasi-Newton mémoire limitée :

- Gestion de la mémoire ?? (règle $m = 10, 20, 30$?)
- Pas de résultat théorique de convergence
- Utilise uniquement $\nabla f(x)$
- Méthode pragmatique, très efficace en pratique
- Utilisée tous les jours en météorologie...

La fin de cette histoire

L'optimisation différentiable sans contraintes nous apprend que :

- Les algorithmes utilisent une information locale - mais on les globalise
- Les méthodes efficaces exigent d'utiliser de l'information du 2nd ordre

Mêmes idées et les mêmes briques de base en :

- Optimisation avec contraintes
- Optimisation non-différentiable

En pratique :

- Méthodes très utilisées dans les applications ! (pénalisation)
- Choix d'une méthode : dépend de la structure de la fonction - mais souvent plusieurs méthodes sont disponibles
- Critères : taille des problèmes, contraintes pratiques et surtout temps de calcul de $f(x)$, $\nabla f(x)$ et $\nabla^2 f(x)$