Présentation TP Optimisation

Minimisation sans contrainte

Lucas Rosas et Jeanne Marque

16 janvier 2026

Introduction : Objectifs du TP

Problématique

Comment minimiser une fonction $f : \mathbb{R}^n \to \mathbb{R}$ de manière efficace et robuste ?

Objectif du TP

Implémenter et comparer des algorithmes :

- Gradient à pas fixe.
- Newton.
- Recherche linéaire (backtracking, Wolfe).

Méthodes implémentées

Gradient à pas fixe

- Direction : $-\nabla f(x)$.
- Limite : Sensible au choix du pas.

Newton

- Direction : $H^{-1}\nabla f(x)$.
- Avantages : Convergence quadratique.

Recherche linéaire

- **Backtracking** : Divise le pas par 2 si f n'améliore pas.
- **Wolfe** : Équilibre décroissance et pente.

Le "cas" Rosenbrock : Pourquoi est-ce difficile ?

$$f(x, y) = 100(y - x^2)^2 + (1 - x)^2$$

- **Géométrie :** Une "vallée" étroite et courbée.
- **Mauvais conditionnement :**
  - Ratio des valeurs propres $\kappa(H) \approx 2500$.
  - Déséquilibre massif entre la courbure des parois et celle du fond.
- **Minimum en** $(1, 1)$

Descente de Gradient à Pas Fixe (`ls_constant`)

Méthode : La plus basique des méthodes de descente de gradient, utilisant un pas fixe s pour avancer dans la direction opposée au gradient.

Sensibilité au choix du pas :

- **Pas trop grands** (0.325, 0.25, 0.125, 0.05) :
  - L'algorithme **diverge rapidement** (8 à 9 itérations).
- **Pas petit** (10^{-3}) :
  - Convergence **très lente**.
  - Nécessite plus de 20 000 itérations.

Descente de Gradient avec Recherche Linéaire (Line Search)

Objectif : Stabiliser et accélérer la convergence grâce à un pas adaptatif.

Méthodes :

- **Backtracking (`ls_backtracking`)** :
  - Stable, converge même quand le pas fixe diverge.
  - Exemple : 19 083 itérations, coût final 1.251×10^{-8}
- **Partial Backtrack (`ls_partial_backtrack`)** :
  - Plus rapide que le backtracking.
  - Exemple : 5 513 itérations.
- **Conditions de Wolfe (`ls_wolfe`)** :
  - $f(x_k + sd_k) \leq f(x_k) + \epsilon_1 s(\nabla f(x_k)^T d_k)$
  - $\nabla f(x_k + sd_k)^T d_k \geq \epsilon_2(\nabla f(x_k)^T d_k)$
  - Comportement un peu "erratique mais efficace".
  - Exemple : 8 631 itérations.

Méthode de Newton

Principe : Utilise la **matrice Hessienne** pour calculer une direction de descente plus précise.

Exemples :

- **Newton à pas fixe (1.0)** :
  - Point de départ : $(-1, 1)$
  - Convergence ultra-rapide en 6 itérations
  - Coût quasi nul : 3.478×10^{-23}
- **Newton loin du minimum** :
  - Point de départ : $(8, 2)$
  - Pas fixe de 1 $\rightarrow$ divergence (`NaN`)
  - Avec line search de Wolfe $\rightarrow$ convergence en 244 itérations
  - Variante `ls_wolfe_step_is_one` $\rightarrow$ convergence en 61 itérations

Performance de l'algorithme Newton + Wolfe modifié

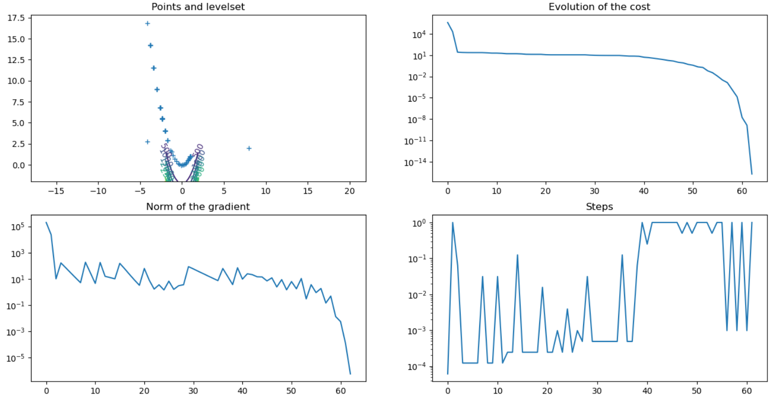

Figure – Algorithme de Newton-Wolfe avec pas initial de 1

Analyse Comparative sur Rosenbrock

Algorithme	Stabilité	Itérations	Comportement
Gradient (Pas Fixe)	Critique	> 1000	Très Lent / Divergence
Gradient + Wolfe	Robuste	≈ 200	Lent (problème de zigzag)
Newton (Pas Fixe)	Nul	NaN	Diverge hors du bassin local
Newton + Wolfe	**Optimale**	**10 - 20**	**Convergence Quadratique**

Table – Tests effectués depuis le point éloigné $x_0 = [8, 2]$

Le rôle de la Hessienne

Anticipe la courbure, évitant les zigzags inefficaces du gradient. Mais le coût de son calcul est important !

L'apport de Wolfe

"Globalise" Newton : sécurise la descente quand l'approximation quadratique est fausse.

Visualisation des itérations

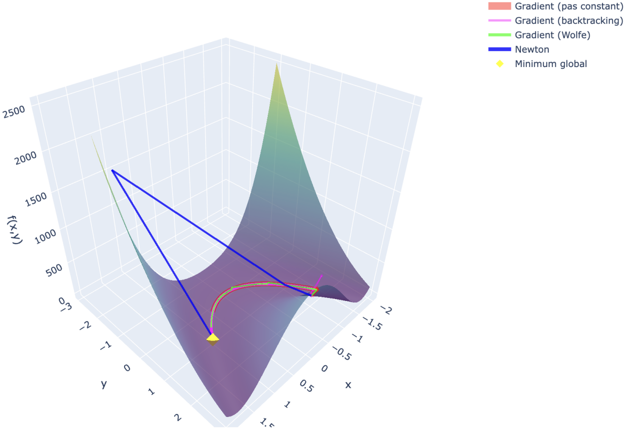

Figure – "Chemins" empruntés par les méthodes