TP No 1

L'objectif de ce TP est d'implémenter et de tester les algorithmes du gradient à pas constant et à pas optimal (exact ou approché). On utilisera pour cela le langage de programmation PYTHON.

Pour chacune des fonctions demandées ci-dessous, on réalisera des tests en utilisant la fonction de Rosenbrock (que l'on cherche à minimiser) :

$$f_p(X) = (X_1 - 1)^2 + p(X_1^2 - X_2)^2.$$

Pour mettre au point les diverses méthodes d'optimisation, on pourra commencer avec $p = 10$, $X_0 = (-1, 1)$ et un nombre d'itérations maximum de 100. Lorsque ce sera fait, on pourra étudier l'influence du paramètre p sur le comportement des algorithmes.

Pour chacune des méthodes testées, on fera figurer sur un même graphe l'ensemble des points des suites obtenues, ainsi que les courbes d'isovaleurs de la fonction de Rosenbrock (fonctions **contour** et **meshgrid** en prenant garde de prendre un nombre suffisant d'isovaleurs).

On tracera également **sur un même graphe**, pour comparer les méthodes d'optimisation entre elles, l'évolution de la norme de la différence entre la solution exacte (connue ici) et les points obtenus au cours des itérations. On pourra éventuellement utiliser une échelle logarithmique (fonctions **loglog** ou **semilogy**).

1. Écrire trois fonctions :
   (a) la première évalue la fonction de Rosenbrock en un point X donné. On peut imaginer une entête de fonction du genre
   ```
   def Rosenbrock(X,param):
       p = param[0]
       x1 = X[0]
       x2 = X[1]
       ...
   ```
   (b) la seconde évalue le gradient de la fonction de Rosenbrock en un point X donné ;
   (c) la troisième trace les courbes d'isovaleurs de la fonction de Rosenbrock sur un pavé $[a, b] \times [c, d]$.
2. Écrire une fonction `GPF` réalisant l'algorithme du **gradient à pas fixe**.
   On pourra utiliser pour paramètres d'entrée de cette fonction :
   — nom de la fonction donnant le gradient en un point X de la fonction à minimiser,
   — valeur du pas constant ρ le long du gradient,
   — valeur initiale X_0,
   — nombre maximum d'itérations,
   — la liste $param$ des paramètres de la fonction à minimiser.
   Pour paramètre de sortie :
   — la liste des points successifs calculés par l'algorithme.
   Indication : pour la fonction de Rosenbrock, on peut commencer par $\rho = 0.01$.
3. Intéressons nous maintenant à l'algorithme du gradient avec une **recherche linéaire inexacte de type Armijo (backtracking)**.
   (a) Écrire d'abord une fonction `RLA` pour la recherche linéaire inexacte de type Armijo.
   Paramètres d'entrée :

— nom de la fonction donnant la valeur en un point X de la fonction à minimiser,
— nom de la fonction donnant le gradient en un point X de la fonction appelée en premier argument,
— valeur du point courant X,
— la direction de descente d considérée,
— valeur ρ_0 pour initier la recherche linéaire,
— la liste $param$ des paramètres de la fonction à minimiser,
— la liste $paramRL$ des paramètres de la recherche linéaire.

Paramètres de sortie : la valeur ρ du pas tel que

$$f(X+\rho d) \leq f(X) + c_1 \rho \nabla f(X)^\top d.$$

(b) Écrire maintenant une fonction `GRL` réalisant l'algorithme du gradient avec une recherche linéaire.

Paramètres d'entrée :

— nom de la fonction donnant la valeur en un point X de la fonction à minimiser,
— nom de la fonction donnant le gradient en un point X de la fonction appelée en premier argument,
— nom de la recherche linéaire choisie,
— valeur ρ_0 pour initier la recherche linéaire,
— valeur initiale X_0,
— nombre maximum d'itérations,
— la liste $param$ des paramètres de la fonction à minimiser,
— la liste $paramRL$ des paramètres de la recherche linéaire.

Paramètres de sortie :

— la liste des points successifs calculés par l'algorithme,
— (éventuellement) la liste des valeurs optimales successives du paramètre ρ obtenues à chaque itération de l'algorithme.

Indication : pour la fonction de Rosenbrock, on peut commencer par $\rho_0 = 0.1$.

4. Que se passe-t-il si on utilise plutôt une **recherche linéaire inexacte de type Wolfe**.

(a) Écrire d'abord une fonction `RLW` pour la recherche linéaire inexacte de type Wolfe. On rappelle qu'il s'agit de chercher la valeur ρ du pas tel que

$$f(X+\rho d) \leq f(X) + c_1 \rho \nabla f(X)^\top d \quad \text{et} \quad \nabla f(X+\rho d)^\top d \geq c_2 \nabla f(X)^\top d.$$

(b) Puis comparer le comportement de `GRL` lorsqu'il est appelé avec `RLA` ou `RLW`.

5. Écrire une fonction `GPE` réalisant l'algorithme du gradient à **pas optimal exact**. Sachant que la fonction à minimiser est un polynôme, il faut donc déterminer, pour le calcul du pas optimal exact, le polynôme dérivé le long de la direction de descente, calculer ses racines réelles (fonction **roots**) et prendre la plus petite racine positive.

Paramètres d'entrée :

— nom de la fonction donnant le gradient en un point X d'une fonction générique,
— valeur initiale X_0,
— nombre maximum d'itérations,
— la liste $param$ des paramètres (du gradient) de la fonction à minimiser.

Paramètres de sortie :

— la liste des points successifs calculés par l'algorithme,
— (éventuellement) la liste des valeurs optimales successives du paramètre ρ obtenues à chaque itération de l'algorithme.