Tehnici de Optimizare
Laborator 2
– Probleme fără constrângeri –

1 Metodele Gradient şi Gradient Stohastic

Metoda Gradient reprezintă un algoritm de ordin I care, pornind dintr-un punct iniţial ales x^0, generează un şir de iteraţii (vectori) $x^1, x^2, \cdots$ pe baza direcţiei gradientului funcţiei obiectiv. Vom presupune că f are gradient $L-$continuu Lipschitz.

Metoda Gradient(x^0, ϵ)
Iniţializează $k = 0$.
Cât timp $\|\nabla f(x^k)\| \leq \epsilon$:
1. Calculează $\nabla f(x^k)$
2. Actualizează: $x^{k+1} = x^k - \alpha_k \nabla f(x^k)$
3. $k := k + 1$.

La linia 1, presupunem că un oracol de ordin I returnează gradientul ∇f evaluat în punctul curent x^k. Un aspect important al metodei este alegerea pasului $\alpha_k > 0$ dintre următoarele opţiuni:

- $\alpha_k = \alpha \in \left(0, \frac{2}{L}\right)$
- $\alpha_k = \arg\min_{\alpha \geq 0} f(x^k - \alpha \nabla f(x^k))$
- Alege $c > 0$, ajustează pasul α_k astfel încât să aibă loc relaţia de descreştere:
$$f(x^k - \alpha_k \nabla f(x^k)) \leq f(x^k) - c\alpha_k \|\nabla f(x^k)\|^2. \tag{1}$$
Procedura de ajustare presupune alegerea $\rho \in (0, 1]$ şi actualizarea:
  - (i) Alegem $c, \rho \in (0, 1), \alpha_{k,0} > 0$
  - (ii) Cât timp $\alpha_{k,t}$ nu satisface (1) iterăm: $\alpha_{k,t+1} := \rho\alpha_{k,t}; \quad t := t + 1;$

În multe aplicaţii precum *regresie liniară*, se urmăreşte minimizarea unui cost (reziduu) fără a cunoaşte distribuţia datelor, e.g.
$$\min_x \mathbb{E}[(a_\xi x - b_\xi)^2].$$
Observăm următoarea structură a costului:
$$f(x) = \mathbb{E}[F(x, \xi)] = \int F(x, \xi) dP(\xi),$$
unde $F(x, \xi)$ sunt funcţii componente (accesibile), dar distribuţia $P(\xi)$ este necunoscută. In acest context, lipsa întregii informaţii legate de funcţia obiectiv $\{f(x), \nabla f(x), \cdots\}$ este modelată ca o perturbare a valorilor $\{f(x), \nabla f(x)\}$ într-un punct dat de zgomot aleator. De aceea, aproximăm f şi $\nabla f(x)$ eşantionând $\{\xi_1, \cdots, \xi_N\}$ dintr-o distribuţie uniformă şi calculând $\frac{1}{N} \sum_{i=1}^{N} F(x, \xi_i)$ şi $\frac{1}{N} \sum_{i=1}^{N} \nabla F(x, \xi_i)$.

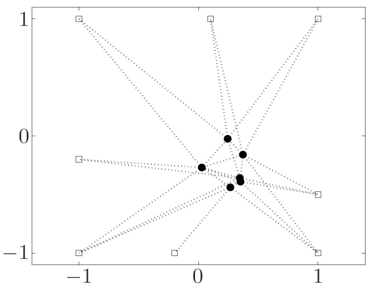

Figure 1: Exemplu plasare depozite

Metoda Gradient Stohastic(x^0, ϵ)
Iniţializează $k = 0$.
Cât timp $k < T$:
1. Alege aleator (uniform) $\{\xi_1, \cdots, \xi_N\} \subset \{1, \cdots, m\}$ şi calculează $\{\nabla f(x^k; \xi_1), \cdots, \nabla f(x^k; \xi_N)\}$
2. Actualizează: $g(x^k) = \frac{1}{N} \sum\limits_{i=1}^{N} \nabla f(x^k; \xi_i)$
2. Actualizează: $x^{k+1} = x^k - \alpha_k g(x^k)$
3. $k := k + 1$.

1.1 Probleme propuse

1. Consideraţi problema pătratică:

$$\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Ax - b\|^2 \ (=: f(x))$$

a) Generaţi netrivial (A, b) astfel încât $\kappa = \frac{L}{\sigma} > 10^6$ şi $V^* > 10^3$ (verificaţi!)

b) Pentru datele de la punctul a), implementaţi Metoda Gradient pentru fiecare opţiune a pasului α_k din secţiunea precedentă.

c) Implementaţi Metoda Gradient Stohastic prezentată la curs. Calculaţi limitele admise ale pasului α_k pentru care metoda converge la optim. Explicaţi aceste limite.

d) Scrieţi un algoritm iterativ: $x^{k+1} = x^k - \alpha_k s^k$ cu proprietăţile:

(*i*) s^k este ales aleator;

(*ii*) la fiecare iteraţie distanţa euclidiană dintre direcţia s^k şi $\nabla f(x^k)$ este în limitele $[0.01, 0.1]$;

(*iii*) algoritmul este convergent.

Explicaţi alegerile.

2 Aplicaţie: plasare şi localizare

O companie are un set de m sedii fixe care trebuiesc aprovizionate periodic (puncte în R^2 sau R^3). Construim un set de depozite care trebuie plasate (legate prin muchii la primul set de puncte fixe) astfel încât o măsură a lungimii de interconectare este minimă (de exemplu, în fig. 1 pătrăţelele sunt sediile şi punctele negre reprezintă poziţiile unde sunt plasate depozitele).

Considerând distanţa dintre două noduri $\|x_i - x_j\|_2$, atunci problema de plasare se formulează:

$$\min_{x \in \mathbb{R}^n} \sum_{i<j} \omega_{ij} h(\|x_i - x_j\|), \tag{2}$$

unde x_i reprezintă un punct (sediu sau depozit). Unele x_i sunt constante (sedii), iar cele rămase sunt variabilele de optimizare (poziţiile depozitelor). Funcţia h este măsura lungimii de interconectare. Dimensiunea n este 2 sau 3.
Cerinţe:

a) Alegeţi $h(z) = z^2$. Aplicaţi metodele Gradient şi Gradient Stohastic (scrise pentru secţiunea precedentă) pentru rezolvarea problemei (2).

b) Încercaţi aceeaşi aplicare pentru $h(z) = z$. Determinaţi o situaţie în care aceste metode nu funcţionează (argumentaţi).

c) Alegeţi $h(z) = z^p$. Din punct de vedere geometric, ce se întâmplă cu soluţia (poziţiile optime ale depozitelor) când p creşte? Dar când $p < 1$?

2.1 Ghid Python

Importare pachete necesare:

```
import numpy as np
from numpy import linalg
from numpy.linalg import norm
from numpy.linalg import eigvals
import matplotlib.pyplot as plt
```

Exemplu generare date:

```
n = 50
m = 70
A = np.random.randn(m,n)
b = np.random.randn(m,)
```

Exemplu calcul expresie gradient:

```
gradient = lambda x: np.matmul(A.T, np.matmul(A, x) - b)
```

Exemplu grafic curbe de convergenţă:

```
iterations_SG = range(100)
plt.xlabel("Iteratii")
plt.ylabel("f(x) - f*")
line = plt.semilogy(iterations_SG, f_traj_SG1, "b")
line = plt.semilogy(iterations_SG, f_traj_SG5, "r")
plt.legend(["SG1","SG2"])
plt.show()
```