MODIA **Février 2021**

Durée : 1h.

Les notes de cours sont autorisées.

La plus grande partie de cet examen reprend exactement le contenu du TP. Comme vous n'avez pas eu le temps de répondre à toutes les questions, nous allons essayer de bien interpréter les résultats dans cet examen. Répondez de façon concise, mais précise !

Nous rappelons d'abord le cadre de travail. Soit $X \sim \mathcal{N}(0, \mathrm{Id})$ un vecteur aléatoire de $\mathbb{R}^d$, pour $d \in \mathbb{N}$. Pour un certain vecteur $\theta \in \mathbb{R}^d$, on construit une variable aléatoire $Y \in \mathbb{R}$ définie par $Y = \langle \theta, X \rangle + B$ où $B \sim \mathcal{N}(0, \sigma^2)$ est une variable aléatoire gaussienne indépendante de X. L'objectif ici est d'apprendre le vecteur θ inconnu à partir de $n \in \mathbb{N}$ observations $(x_i, y_i)_{1 \leq i \leq n}$ tirées indépendamment.

Pour ce faire, on peut simplement résoudre le problème de minimisation du risque empirique suivant :

$$\inf_{w \in \mathbb{R}^d} E_n(w) = \frac{1}{2n} \sum_{i=1}^{n} (\langle w, x_i \rangle - y_i)^2 \tag{1}$$

On notera w_n^* n'importe quel minimiseur du problème ci-dessus. On rappelle aussi la définition du risque moyen :

$$E(w) = \frac{1}{2} \int \int (\langle w - \theta, x \rangle + b)^2 dP_X(x) dP_B(b) \tag{2}$$

On note w^* le minimiseur de E et on rappelle que sous les hypothèses précédentes $w^* = \theta$.

Conformément au TP, les lignes suivantes ont été déjà codées.

```python
import numpy as np
import matplotlib.pyplot as plt

# Generateur de donnees (X,Y)
def generate_data(d,n,theta,sigma) :
    X = np.random.randn(d,n)
    Y = np.dot(X.transpose(),theta) + sigma*np.random.randn(n)
    return X, Y

# Risque moyen
def E(w,theta,sigma) :
    dif = w-theta
    return np.sum(dif**2)/(2.*n) + sigma**2/2.

# Risque empirique
def En(w,X,Y) :
    n = Y.shape[0]
    dif = np.dot(X.transpose(),w) - Y
    return 0.5/n*np.sum(dif**2)

# Gradient du risque empirique
```

```python
def grad_En(w,X,Y) :
    n = Y.shape[0]
    dif = np.dot(X.transpose(),w) - Y
    return np.dot(X, dif)/n

# Gradient stochastique
def grad_sto_En(w,X,Y,n_batch) :
    n = Y.shape[0]
    I = np.random.randint(0,n,n_batch)
    dif = np.dot(X[:,I].transpose(),w) - Y[I]
    return 1/n*np.dot(X[:,I], dif)/n_batch
```

Question 1.

Le but de cette question est d'interpréter un code, de comprendre ce qu'il fait et d'interpréter les résultats.

```python
d = 5
sigma = 1
theta = np.ones(d)
n_rep = 100

Erreur = []
list_n = []
for i in range(7):
    n = 10**i
    E_dif = 0
    for j in range(n_rep):
        Xn, Yn = generate_data(d,int(n),theta,sigma)
        h = theta
        hn = np.linalg.solve(np.dot(Xn,Xn.transpose()), np.dot(Xn,Yn))
        E_dif += E(hn,theta,sigma) - E(h,theta,sigma)

    E_dif/=n_rep
    Erreur.append(E_dif)
    list_n.append(n)

print(Erreur)
plt.figure(1)
plt.loglog(list_n,np.abs(Erreur))
plt.show()
```

1. Que mesure E_dif ?
2. Dans quel intervalle varie n ?
3. Que représente h_n ?
4. Quel est le rôle de la boucle sur j et de la variable n_rep ?
5. Quel est l'objectif de ce code ?
6. La sortie du programme est la figure 1. La courbe semble être une droite, à part le premier point qui semble différent. Pouvez-vous expliquer pourquoi ?
7. Pouvez-vous estimer le coefficient directeur de la droite ?
8. Interprétez sa signification en relation avec le cours.

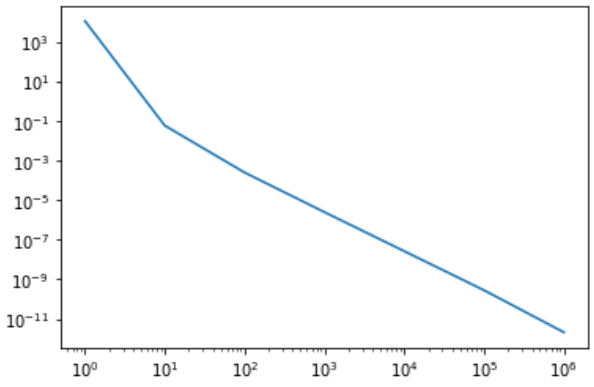

FIGURE 1 – La sortie du programme de la question 1.

Question 2. Là encore, nous allons essayer d'interpréter les résultats du TP.

```
1  sigma = 0.1
2  d = 5
3  n = 20
4  n_batch = 1
5  theta = np.ones(d)
6  Xn, Yn = generate_data(d,int(n),theta,sigma)
7
8  n_epoch = 100
9  n_batch = 1
10 L=1./n*np.linalg.norm(Xn.dot(Xn.transpose()))
11
12 CF_gd = np.zeros(n_epoch)
13 E_gd = np.zeros(n_epoch)
14 alpha = 1/L
15 w = np.zeros(d)
16 for i in range(n_epoch) :
17     CF_gd[i] = En(w,Xn,Yn)
18     E_gd[i] = E(w,theta,sigma)
19     grad = grad_En(w,Xn,Yn)
20     w = w - alpha*grad
21 w_gd = w
22
23 plt.figure(1)
24 plt.semilogy(CF_gd-np.min(CF_gd))
25 plt.show()
```

1. Que fait le code ci-dessus ?
2. Pourquoi retire-t'on np.min(CF_gd) à la ligne 24 ?
3. Le résultat de ce code est indiqué sur la figure 2. Est-ce que la méthode implémentée converge ? Avec une bonne précision ? Pouvez-vous expliquer pourquoi ?
4. Le résultat en sortie est le vecteur $[1.01, 0.95, 1.04, 1.04, 0.99]$, ce qui est différent de θ. Pouvez-vous expliquer pourquoi ?

Question 3.

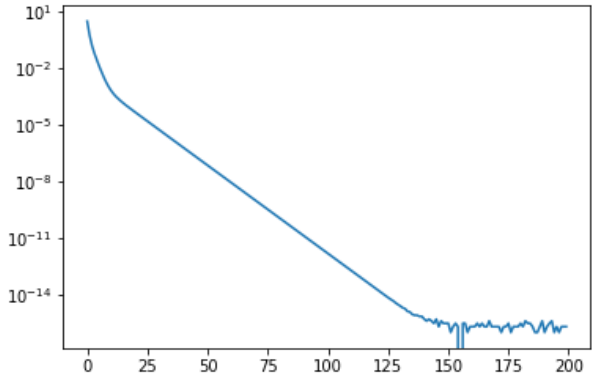

FIGURE 2 – Sortie de la question 2.

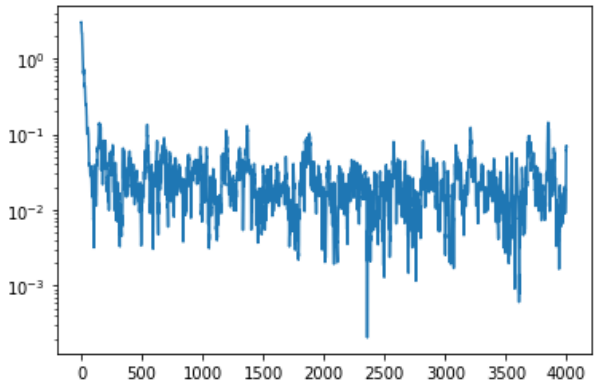

FIGURE 3 – Sortie de la question 3.

```
nit_gs = int(n_epoch*n/n_batch)
alpha = 1/L
CF_gs = np.zeros(nit_gs)
E_gs = np.zeros(nit_gs)
w = np.zeros(d)
for i in range(nit_gs) :
    CF_gs[i] = En(w,Xn,Yn)
    E_gs[i] = E(w,theta,sigma)
    grad = grad_sto_En(w,Xn,Yn,n_batch)
    w = w - alpha*grad
w_gs = w

plt.figure(2)
plt.semilogy(CF_gs - np.min(CF_gd))
plt.show()
```

1. Que fait le code ci-dessus ?
2. Le résultat de ce code est indiqué sur la figure 3. Est-ce que la méthode implémentée converge ? Avec une bonne précision ? Pouvez-vous expliquer pourquoi ?
3. Pouvez-vous proposer des stratégies pour améliorer la convergence ?
4. Finalement, la figure 4, montre le résultat des lignes suivantes.

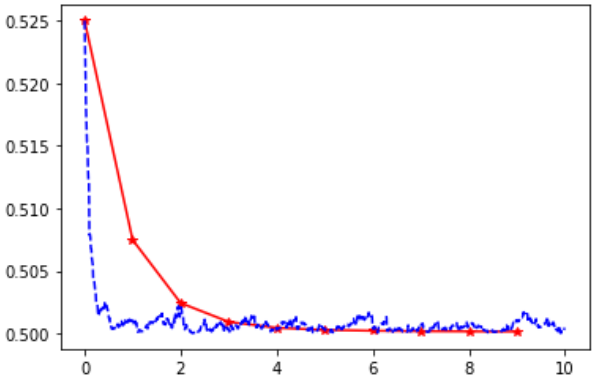

FIGURE 4 – Sortie de la question 4.

```
1
2
3
4
5
```

```
plt.figure(3)
plt.plot(range(n_epoch),E_gd,'r*-')
plt.plot(np.linspace(0,n_epoch,nit_gs),E_gs,'b-')
plt.show()
```

Pouvez-vous comparer les avantages et inconvénients de chaque méthode ?