1 Inverse d'une matrice

En algèbre, on est souvent amené à faire apparaître l'inverse A^{-1} d'une matrice A dans les formules et les raisonnements. En calcul numérique, pour des raisons de stabilité numérique, on préfère généralement résoudre un système d'équations linéaires. Par exemple, plutôt que de calculer

$$u \;=\; A^{-1}\,b + w\,,$$

on préfèrera

$$\begin{array}{rrcl} \text{résoudre} & A\,x & = & b\,, \\ & u & = & x + w\,. \end{array}$$

Toutefois, il peut arriver qu'on souhaite quand même calculer l'inverse d'une matrice A.

Question 1. En utilisant les méthodes étudiées précédemment, proposer un algorithme pour calculer l'inverse d'une matrice A.

Question 2. Donner les instructions Python correspondant à votre algorithme

2 Analyse de code Python

On rappelle le code de `kappa.py` qui permet de tracer le graphique du chapitre 5 du support de cours.

```
import math
import numpy as np
import scipy.linalg as nla
import matplotlib.pyplot as plt
from scipy.stats import ortho_group

abscissas = []
ordinates = []
m = 10
for exponent in range(5,15) :
    U = ortho_group.rvs(m)
    V = ortho_group.rvs(m)
    kappa = 10**exponent
    T = np.diag ([kappa**(-i/(m-1)) for i in range(m)])
```

```
    A = np.dot (U, np.dot (T, V))
    est_kappa = np.linalg.cond(A)
    x = np.random.rand(m)
    b = np.dot (A,x)
    xback = nla.solve (A,b)
    errrel = np.linalg.norm(x-xback)/np.linalg.norm(x)
    abscissas.append(exponent)
    ordinates.append(- math.log(errrel)/math.log(10))

a = np.polyfit(abscissas, ordinates, 1)
def f(x) :
    return a[0]*x+a[1]

plt.figure ()
plt.axes ()
plt.plot (abscissas, ordinates)
x = np.linspace(abscissas[0], abscissas[-1], 20)
plt.plot (x, f(x))
plt.show ()
```

L'appel à `polyfit` permet de calculer les deux coefficients a_0 et a_1 de la droite d'équation $y = a_0\,x + a_1$ qui passe "au plus près" des points expérimentaux stockés dans les listes *abscissas* et *ordinates*.

Ces deux coefficients peuvent se calculer par les méthodes étudiées en cours. Notons $x_i = abscissas[i]$ et $y_i = ordinates[i]$ pour $0 \leqslant i \leqslant n$.

Posons que la droite d'équation $y = a_0\,x + a_1$ passe par les points (x_i, y_i) pour $0 \leqslant i \leqslant n$.

$$\begin{aligned} a_0\,x_0 + a_1 &= y_0\,, \\ a_0\,x_1 + a_1 &= y_1\,, \\ &\vdots \\ a_0\,x_n + a_1 &= y_n\,. \end{aligned}$$

Mettons ce système sous forme matricielle :

$$\underbrace{\begin{pmatrix} x_0 & 1 \\ x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} a_0 \\ a_1 \end{pmatrix}}_{v} = \underbrace{\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}}_{b}.$$

Les coefficients a_0 et a_1 recherchés s'obtiennent en résolvant le système (ça se démontre, ce n'est pas évident)

$$A^T\,A\,v = A^T\,b\,. \tag{1}$$

La matrice A est symétrique. Elle est aussi définie positive.

Question 3. En utilisant les méthodes étudiées précédemment, proposer un algorithme pour résoudre le système (1).

Question 4. Donner les instructions Python correspondant à votre algorithme.