homework2

November 13, 2021

1 Homework Set 2

Please **submit this Jupyter notebook through Canvas** no later than **Mon Nov. 15, 9:00**.

Homework is in **groups of two**, and you are expected to hand in original work. Work that is copied from another group will not be accepted.

2 Exercise 0

Write down the names + student ID of the people in your group.

Florian Tiggeloven 11872802 Jacco van Wijk 11282479

Write your answer, using LaTeX, in this box.

Run the following cell to import the necessary packages.

[1]:
```
import numpy as np
import scipy.linalg as la
import matplotlib.pyplot as plt
import random
```

3 Exercise 1

The goal of this problem is to show that apparently harmless looking systems of linear equations may be very difficult to solve. Some functions that may be useful are `numpy.triu`, `numpy.tril`, `numpy.eye`, `random.randrange`. ## (a) Generate an $n \times n$ matrix B with random integer elements in the range $b_{ij} \in [-10, 10]$. Choose for instance $n = 20$.

[2]:
```
n = 20
b_min = -10
b_max = 10
B = np.random.randint(b_max - b_min, size=(n,n)) + b_min

print('\n'.join([''.join(['{:4}'.format(item) for item in row])
      for row in B]))
```

```
  -4  -7   7  -2  -3 -10   2   5  -4   4   1  -4   1   8  -4   0   8  -4  -2   2
   8   2  -6  -5   2  -6   8  -2  -4   3  -7  -3   3   2   5  -9  -8   8  -5  -5
   2  -1  -8 -10   7   9  -5   5 -10   6   2  -3   0  -6  -8   5  -2   7   0  -8
   9   8   0   7   2   8   9  -7   9  -1  -7  -1  -2 -10   3  -1  -8  -9   6  -6
  -5   3  -5   2   4   7   2  -3   1   7   8 -10   0   2   9   4   3   5  -7   6
  -7   9  -9  -6   4   8 -10 -10  -3   3  -9  -9  -9   0  -1   4  -4   0   2  -9
   1   6   8   2   0   2   5   3   5  -7   8 -10 -10   7 -10   2   0  -4   8  -9
   0  -2   9   9   4   1   2  -4   1  -4   7   3   3   3  -5   2  -1  -5  -3  -3
   3  -5   2   8  -7   1   7   6   2 -10   9  -3   3   0   7  -8  -1   9   3  -6
  -1   7  -7  -1  -6   7   5   2   8  -1 -10  -2   5   2  -6  -4   7 -10   0  -3
  -8  -6  -4   3   6  -7   0  -6  -7   7   0  -7   3  -7  -4   1  -8   5  -8  -1
   4  -6   3  -7   8   8 -10   8  -8   0   5   5   6  -5 -10  -6   5  -6   1  -6
   0   5   2  -7   2  -4   4  -6  -8   8  -1   7   3   3   3  -1  -3   1  -2  -3
  -6  -1   3   9  -3  -5  -7  -3  -5  -2   4   1   2   8  -8   1  -6   0   8   3
  -2   1   5   3   3  -1   6  -7  -3   4  -5   2  -5  -3   3  -4   8  -6 -10  -6
  -6  -3  -1  -8  -2  -7  -9   9  -8   5  -4   4   6   6  -9 -10  -6  -4  -8   8
   3  -9   8   2  -6   4   1   7   3  -4   8 -10   2  -8   8   3   8  -6  -2  -6
   5  -1   6  -5   9  -7   0   5   7 -10  -5   6  -9   8   6 -10   9   9   3   8
 -10   0   6  -1   9  -5   8   1   7   1  -3   1  -1  -6   6  -2   3  -5  -5  -1
   7  -1   4   4   6   9   9   2   4  -6   9   2  -5  -5  -6  -4  -3   0  -4   0
```

3.1 (b)

Remove the diagonal of B, save the upper triangular part in U and the lower triangular part in L, and put ones on the diagonals $l_{ii} = u_{ii} = 1$.

[3]:
```python
# Remove diagonal
np.fill_diagonal(B, 0)

# Create U and L
U = np.triu(B) + np.eye(n)
L = np.tril(B) + np.eye(n)

# Print U and L
print("U = ")
print('\n'.join([''.join(['{:4}'.format(round(item)) for item in row])
      for row in U]), end="\n\n")
print("L = ")
print('\n'.join([''.join(['{:4}'.format(round(item)) for item in row])
      for row in L]))
```

```
U = 
   1  -7   7  -2  -3 -10   2   5  -4   4   1  -4   1   8  -4   0   8  -4  -2   2
   0   1  -6  -5   2  -6   8  -2  -4   3  -7  -3   3   2   5  -9  -8   8  -5  -5
   0   0   1 -10   7   9  -5   5 -10   6   2  -3   0  -6  -8   5  -2   7   0  -8
   0   0   0   1   2   8   9  -7   9  -1  -7  -1  -2 -10   3  -1  -8  -9   6  -6
   0   0   0   0   1   7   2  -3   1   7   8 -10   0   2   9   4   3   5  -7   6
   0   0   0   0   0   1 -10 -10  -3   3  -9  -9  -9   0  -1   4  -4   0   2  -9
   0   0   0   0   0   0   1   3   5  -7   8 -10 -10   7 -10   2   0  -4   8  -9
   0   0   0   0   0   0   0   1   1  -4   7   3   3   3  -5   2  -1  -5  -3  -3
   0   0   0   0   0   0   0   0   1 -10   9  -3   3   0   7  -8  -1   9   3  -6
   0   0   0   0   0   0   0   0   0   1 -10  -2   5   2  -6  -4   7 -10   0  -3
   0   0   0   0   0   0   0   0   0   0   1  -7   3  -7  -4   1  -8   5  -8  -1
   0   0   0   0   0   0   0   0   0   0   0   1   6  -5 -10  -6   5  -6   1  -6
   0   0   0   0   0   0   0   0   0   0   0   0   1   3   3  -1  -3   1  -2  -3
   0   0   0   0   0   0   0   0   0   0   0   0   0   1  -8   1  -6   0   8   3
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1  -4   8  -6 -10  -6
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1  -6  -4  -8   8
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1  -6  -2  -6
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   3   8
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1  -1
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1

L = 
   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   8   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   2  -1   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   9   8   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  -5   3  -5   2   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  -7   9  -9  -6   4   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   1   6   8   2   0   2   1   0   0   0   0   0   0   0   0   0   0   0   0   0
   0  -2   9   9   4   1   2   1   0   0   0   0   0   0   0   0   0   0   0   0
   3  -5   2   8  -7   1   7   6   1   0   0   0   0   0   0   0   0   0   0   0
  -1   7  -7  -1  -6   7   5   2   8   1   0   0   0   0   0   0   0   0   0   0
  -8  -6  -4   3   6  -7   0  -6  -7   7   1   0   0   0   0   0   0   0   0   0
   4  -6   3  -7   8   8 -10   8  -8   0   5   1   0   0   0   0   0   0   0   0
   0   5   2  -7   2  -4   4  -6  -8   8  -1   7   1   0   0   0   0   0   0   0
  -6  -1   3   9  -3  -5  -7  -3  -5  -2   4   1   2   1   0   0   0   0   0   0
  -2   1   5   3   3  -1   6  -7  -3   4  -5   2  -5  -3   1   0   0   0   0   0
  -6  -3  -1  -8  -2  -7  -9   9  -8   5  -4   4   6   6  -9   1   0   0   0   0
   3  -9   8   2  -6   4   1   7   3  -4   8 -10   2  -8   8   3   1   0   0   0
   5  -1   6  -5   9  -7   0   5   7 -10  -5   6  -9   8   6 -10   9   1   0   0
 -10   0   6  -1   9  -5   8   1   7   1  -3   1  -1  -6   6  -2   3  -5   1   0
   7  -1   4   4   6   9   9   2   4  -6   9   2  -5  -5  -6  -4  -3   0  -4   1
```

3.2 (c)

Compute $A = L \cdot U$. What is the value of $\det(A)$ and why? Compute the determinant using the appropriate python command and confirm your prediction. In case that you have doubts about the result, compute separately $\det(L)$ and $\det(U)$.

Write your answer, using LaTeX, in this box.

[11]:
```python
# Compute A = L * U
A = L.dot(U)

# Print A
print('A = ')
print('\n'.join([''.join(['{:4}'.format(round(item)) for item in row])
      for row in A]))

print(np.linalg.det(A))
print(np.linalg.det(L), np.linalg.det(U), np.linalg.det(L) * np.linalg.det(U))
```

```
A =
   1  -7   7  -2  -3 -10   2   5  -4   4   1  -4   1   8  -4   0   8  -4  -2   2
   8 -55  50 -21 -22 -86  24  38 -36  35   1 -35  11  66 -27  -9  56 -24 -21  11
   2 -15  21  -9  -1  -5  -9  17 -14  11  11  -8  -1   8 -21  14  22  -9   1   1
   9 -55  15 -57  -9-130  91  22 -59  59 -54 -61  31  78   7 -73   0  19 -52 -28
  -5  38 -58  47  -9  10  59 -73  77 -36 -42  14   0 -22  90 -50 -67  -4   0   9
  -7  58-112  53 -32 -84  47 -78  29 -18 -23 -15  23  84 162-100 -54 111 -93  64
   1  -1 -21-110  69  44   9   2 -91  67 -49 -76 -13 -41 -44  -6 -80  78  -8-131
   0  -2  21 -71  81 194  20 -29  11  52  15 -96 -50-123 -45  80 -67 -27  51-122
   3 -26  53   7   4  34  11  17  92-128  28  -5 -86 -25-186  41 -36-194 159-171
  -1  14 -56  36 -40-138   3 -82  45-167 -52 -68 -56  89  21-147 -91  22  75-176
  -8  50 -24  89  -4 139  65 -23 137  45 -58  44  30 -81  39  44  68-139  -3 165
   4 -34  67 -15  -9  31-192 -30-151 221   2 -38  49   2 -53 199  87   7-227 135
   0   5 -28 -52  12 -58  15  89 -83 120 -93 -34  63  73-167 -52 144 -57 -50  11
  -6  41 -33  -4  52 139  83 -41  48  55-133 137  74-252  34   8-152 -25  -7  37
  -2  15 -15 -48  52 103  28   4   7  60 -60 -77 -74 -32 -50   8  84 -65  72 -74
  -6  39 -25  29 -13 -16 -46  85 -51  46 -19 290 200  34-122  47  18 -64  10 239
   3 -30  83 -39  27  74-139  47 -41 -68 116 -43 -98 -62 -27 148 -33 -12-151-141
   5 -36  47 -70  24  26  15 140 -79   0 387  -2  73  51 -41 -12 215 123  27 -55
 -10  70 -64 -41  79 204  17  35  43 -84 247-100 -18 -20  86 -35  93  79 -62-212
   7 -50  59 -45  19  55 -47 -50   2  -3 152-327-153 -34 -81 116 -86 134  63-169
-1583925.5251070834
0.9999994016560421 1.0 0.9999994016560421 1.0
```

We notice that the determinants calculated between A, U and L are vastly different. ?

3.3 (d)

Choose now an exact solution, for instance $x_e =$ `numpy.ones(n)`, and compute the corresponding right hand side $b = Ax_e$.

[5]:
```
# Create x_e and compute A * x_e
x_e = np.ones(n)
b = A.dot(x_e)

# Print b
print("b = ")
print('\n'.join([''.join(['{:4}'.format(round(item)) for item in [row]])
      for row in b]))
```

```
b =
  -1
 -36
  16
```

```
-247
 -22
  15
-401
-107
-408
-769
 575
  55
-142
  55
 -64
 675
-386
 838
 307
-438
```

3.4 (e)

Solve $Ax = b$ using `scipy.linalg.lu_factor` and `scipy.linalg.lu_solve` and compare the solution with the exact x_e.

[6]:
```
# Compute L and U using lu_factor and compute x with lu_solve
lu, piv = la.lu_factor(A)
x = la.lu_solve((lu, piv), b)

print("x = ")
print('\n'.join([''.join(['{:4}'.format(round(item,4)) for item in [row]])
      for row in x]), end="\n\n")

print("x_e = ")
print('\n'.join([''.join(['{:4}'.format(round(item,4)) for item in [row]])
      for row in x_e]), end="\n\n")

print("x_e - x = ")
print('\n'.join([''.join(['{:4}'.format(round(item,4)) for item in [row]])
      for row in x_e - x]))
```

```
x =
2607.9894
441.2425
68.8178
4.9975
-3.8695
1.692
1.06
1.0161
```

```
0.9748
0.9973
0.9997
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0

x_e =
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0
 1.0

x_e - x =
-2606.9894
-440.2425
-67.8178
-3.9975
4.8695
-0.692
-0.06
-0.0161
0.0252
0.0027
0.0003
 0.0
```

```
-0.0
 0.0
 0.0
 0.0
 0.0
 0.0
 0.0
-0.0
```

The difference $x_e - x$ illustrates the difference between every element of the vector. The lower elements are very similar, with differences only being small. However, the higher up the vector, the higher the differences, with the highest difference reaching 2606. This is extremely significant.

3.5 (f)

Explain the bad results by computing the condition number of A.

```
[7]: condition = [la.norm(A, ord=x) * la.norm(la.inv(A), ord=x) for x in [1,2,np.
      ↪inf]]
     print(condition)
```

```
[1.6525902506636385e+21, 6.19445786603689e+20, 1.4030186593593564e+21]
```

All three methods of computing the condition of A were computed. All three values are very high, which, according to the definition of the condition number corresponds to high errors in the output. This result can also be observed in the differences found in 1d and 1e.

4 Exercise 2

(N.B. this is a theory exercise.) Suppose we write a $(p+q)\times(p+q)$ matrix M in block form where A, B, C, D are respectively $p\times p, p\times q, q\times p$ and $q\times q$ matrices

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}.$$

4.1 (a)

Verify that

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} I_p & 0 \\ CA^{-1} & I_q \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & D - CA^{-1}B \end{bmatrix} \begin{bmatrix} I_p & A^{-1}B \\ 0 & I_q \end{bmatrix}.$$

$$\begin{bmatrix} I_p & 0 \\ CA^{-1} & I_q \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & D - CA^{-1}B \end{bmatrix} \begin{bmatrix} I_p & A^{-1}B \\ 0 & I_q \end{bmatrix} = \begin{bmatrix} I_pA + 0 \cdot 0 & I_p \cdot 0 + 0 \cdot (D - CA^{-1}B) \\ CA^{-1}A + I_q \cdot 0 & CA^{-1} \cdot 0 + I_q \cdot (D - CA^{-1}B) \end{bmatrix} \begin{bmatrix} I_p & A^{-1}B \\ 0 & I_q \end{bmatrix} \tag{1}$$

$$= \begin{bmatrix} A & 0 \\ C & D - CA^{-1}B \end{bmatrix} \begin{bmatrix} I_p & A^{-1}B \\ 0 & I_q \end{bmatrix} \tag{2}$$

$$= \begin{bmatrix} A \cdot I_p + 0 \cdot 0 & AA^{-1}B + 0 \cdot I_q \\ C \cdot I_p + 0 \cdot (D - CA^{-1}B) & CA^{-1}B + I_q(D - CA^{-1}B) \end{bmatrix} \tag{3}$$

$$= \begin{bmatrix} A & B \\ C & CA^{-1}B + D - CA^{-1}B \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{4}$$

4.2 (b)

Describe how a system $Mx = b$, with x and b in $\mathbb{R}^{p+q}$, can be solved by applying matrix-vector products with C and B and solves with A and $(D - CA^{-1}B)$.

Write your answer, using LaTeX, in this box.

Define M_1, M_2 and M_3 respectively as

$$\begin{bmatrix} I_p & 0 \\ CA^{-1} & I_q \end{bmatrix}, \begin{bmatrix} A & 0 \\ 0 & D - CA^{-1}B \end{bmatrix}, \begin{bmatrix} I_p & A^{-1}B \\ 0 & I_q \end{bmatrix}.$$

Then for solving $Mx = b$ it is known that $x = M^{-1}b = (M_1M_2M_3)^{-1}b = M_3^{-1}M_2^{-1}M_1^{-1}b$. For the inverses of block matrices, it is equal to the matrix of the inverses of its blocks, i.e.:

$$M_1^{-1} = \begin{bmatrix} I_p & 0 \\ -CA^{-1} & I_q \end{bmatrix} M_2^{-1} = \begin{bmatrix} A^{-1} & 0 \\ 0 & (D - CA^{-1}B)^{-1} \end{bmatrix} M_3^{-1} = \begin{bmatrix} I_p & -A^{-1}B \\ 0 & I_q \end{bmatrix}.$$

This means that:

$$x = \begin{bmatrix} I_p & -A^{-1}B \\ 0 & I_q \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & (D - CA^{-1}B)^{-1} \end{bmatrix} \begin{bmatrix} I_p & 0 \\ -CA^{-1} & I_q \end{bmatrix} b = \begin{bmatrix} I_p & -A^{-1}B \\ 0 & I_q \end{bmatrix} \Bigg[\begin{matrix} A^{-1}I_p & \\ (D - CA^{-1}B)^{-1}(-CA^{-1}) & (D - C \end{matrix}$$

Also, b can be defined as a block vector with length p called b_p and length q called b_q, then this creates a formula that can be solved if you want to solve for x: (Willen we dit doen?)

$$x = \begin{bmatrix} A^{-1} & A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1} \\ -(D - CA^{-1}B)^{-1}CA^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix} \begin{bmatrix} b_p \\ b_q \end{bmatrix} = \Bigg[\begin{matrix} A^{-1}b_p - A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1}b \\ -(D - CA^{-1}B)^{-1}CA^{-1}b_p + (D - CA^{-1}B \end{matrix}$$

4.3 (c)

What is the cost, to highest order, of LU-factorizing A and of computing and LU-factorizing $D - CA^{-1}B$?

Write your answer, using LaTeX, in this box.

$A = O(P^3)$

$D - CA^{-1}B = O(p^2q)$ or $O(p^3)$

H e l p

Remark: Although in this case no savings were obtained, the decomposition above is very useful for solving linear systems with many zero coefficients, in other words where M is a sparse matrix. After applying a permutation of the indices such a matrix is written in the above form, where q is as small as possible and A is blockdiagonal, i.e. $A = \begin{bmatrix} E & O \\ O & F \end{bmatrix}$. This blockdiagonal form then causes big savings in computational cost. Moreover, the procedure can be applied recursively to E and F.