CP.2.3.3, Sauer3

Let A be the n-by-n matrix with entries

$$a_{ij} = |i - j| + 1.$$

Define $x = (1, \ldots, 1)^T$ and $b = Ax$. For $n = 100$, 200, 300, 400, and 500, use the python program from Computer Problem 2.1.1 or Numpy's `numpy.linalg.solve` command to compute `xc`, the double precision computed solution. Calculate the infinity norm of the forward error for each solution. Find the five error magnification factors of the problems $Ax = b$, and compare with the corresponding condition numbers.

Hint: Note that this $a_{ij} = |i - j| + 1$ formula is the same with 0-base indexing (Python) or 1-base indexing (Sauer and Matlab). Here is a code snippet to generate A with $n = 5$.

```
n = 5
A = np.zeros( [ n, n ], dtype=float )
for i in range(0,n):
  for j in range(0,n):
      A[i,j] = abs( i - j ) + 1.
print(A)
```

```
[[1. 2. 3. 4. 5.]
 [2. 1. 2. 3. 4.]
 [3. 2. 1. 2. 3.]
 [4. 3. 2. 1. 2.]
 [5. 4. 3. 2. 1.]]
```


CP.2.3.3, Sauer3, solution, Langou

Colab: `https://colab.research.google.com/drive/1qPbZqHhLwn8EOHQODdnI_NWOv9-bSkLp`

Few comments:

- We do feel that this matrices are somewhat big and our `lu_no_pivoting` code is not at all optimized for performance. So running on colab does take some time. The code using `numpy.linalg.solve` is significantly faster. I added `%%time` to time, and so it took about a second to run the code with `numpy.linalg.solve` while it took about a minute to run the code with `lu_no_pivoting`.

- We see that the backward error is growing more with `lu_no_pivoting` than `numpy.linalg.solve`. Pivoting is useful to keep backward error low.

- We observe that the relation

  $$\texttt{error\_magnification\_factor} = \texttt{forward\_error} \;/\; \texttt{backward\_error} \approx \texttt{condition\_number}$$

  holds true.

```
%%time

for n in range(100,600,100):
  A = np.zeros( [ n, n ] )
  for i in range(0,n):
    for j in range(0,n):
      A[i,j] = abs( i - j ) + 1.
  x = np.ones( [ n, 1 ] )
  b = A @ x

  xc = np.linalg.solve( A, b )

  relative_forward_error = np.linalg.norm( x - xc, np.inf )\
     / np.linalg.norm( x, np.inf )
  relative_backward_error = np.linalg.norm( b - A@xc, np.inf )\
    / np.linalg.norm( b, np.inf )
  error_magnification_factor = relative_forward_error\
    / relative_backward_error
  print( "n = ",f"{n:4d}" )
  print( "relative forward error                      = ",\
    f"{relative_forward_error:8.2e}" )
  print( "relative backward error                     = ",\
    f"{relative_backward_error:8.2e}" )
  print( "error magnification factor                  = ",\
   f"{error_magnification_factor:8.2e}" )
  print( "kappa( A ) = || A ||_oo * || A^{-1} ||_oo   = ",\
   f"{np.linalg.norm(A,np.infty)\
    * np.linalg.norm(np.linalg.inv(A),np.infty):8.2e}")
  print("\n")
```

```
n =   100
relative forward error                      =  2.87e-12
relative backward error                     =  5.40e-16
error magnification factor                  =  5.31e+03
kappa( A ) = || A ||_oo * || A^{-1} ||_oo   =  1.01e+04


n =   200
relative forward error                      =  2.01e-11
relative backward error                     =  9.05e-16
error magnification factor                  =  2.22e+04
kappa( A ) = || A ||_oo * || A^{-1} ||_oo   =  4.02e+04


n =   300
relative forward error                      =  4.93e-11
relative backward error                     =  1.29e-15
error magnification factor                  =  3.82e+04
kappa( A ) = || A ||_oo * || A^{-1} ||_oo   =  9.03e+04


n =   400
```

```
relative forward error                              =  1.02e-10
relative backward error                             =  1.27e-15
error magnification factor                          =  8.04e+04
kappa( A ) = || A ||_oo * || A^{-1} ||_oo           =  1.60e+05


n =   500
relative forward error                              =  2.09e-10
relative backward error                             =  1.74e-15
error magnification factor                          =  1.20e+05
kappa( A ) = || A ||_oo * || A^{-1} ||_oo           =  2.51e+05

CPU times: user 495 ms, sys: 311 ms, total: 805 ms
Wall time: 441 ms
```

```
%%time

for n in range(100,600,100):
  A = np.zeros( [ n, n ], dtype=float )
  for i in range(0,n):
    for j in range(0,n):
      A[i,j] = abs( i - j ) + 1.
  x = np.ones( [ n, 1 ], dtype=float )
  b = A @ x

  L, U = lu_no_pivoting( A )
  y = forward__substitution( L, b )
  xc = backward_substitution( U, y )

  relative_forward_error = np.linalg.norm( x - xc, np.inf )\
     / np.linalg.norm( x, np.inf )
  relative_backward_error = np.linalg.norm( b - A@xc, np.inf )\
    / np.linalg.norm( b, np.inf )
  error_magnification_factor = relative_forward_error\
    / relative_backward_error
  print( "n = ",f"{n:4d}" )
  print( "relative forward error                              = ",\
    f"{relative_forward_error:8.2e}" )
  print( "relative backward error                             = ",\
    f"{relative_backward_error:8.2e}" )
  print( "error magnification factor                          = ",\
   f"{error_magnification_factor:8.2e}" )
  print( "kappa( A ) = || A ||_oo * || A^{-1} ||_oo           = ",\
   f"{np.linalg.norm(A,np.infty)\
    * np.linalg.norm(np.linalg.inv(A),np.infty):8.2e}")
  print("\n")
```

```
n =   100
relative forward error                              =  5.29e-11
relative backward error                             =  4.23e-14
error magnification factor                          =  1.25e+03
kappa( A ) = || A ||_oo * || A^{-1} ||_oo           =  1.01e+04
```

```
n =   200
relative forward error                         =  5.78e-10
relative backward error                        =  9.19e-14
error magnification factor                     =  6.29e+03
kappa( A ) = || A ||_oo * || A^{-1} ||_oo      =  4.02e+04


n =   300
relative forward error                         =  3.03e-09
relative backward error                        =  3.49e-13
error magnification factor                     =  8.67e+03
kappa( A ) = || A ||_oo * || A^{-1} ||_oo      =  9.03e+04


n =   400
relative forward error                         =  4.48e-09
relative backward error                        =  6.40e-13
error magnification factor                     =  7.00e+03
kappa( A ) = || A ||_oo * || A^{-1} ||_oo      =  1.60e+05


n =   500
relative forward error                         =  9.63e-09
relative backward error                        =  2.01e-13
error magnification factor                     =  4.80e+04
kappa( A ) = || A ||_oo * || A^{-1} ||_oo      =  2.51e+05

CPU times: user 1min 7s, sys: 880 ms, total: 1min 7s
Wall time: 1min 7s
```