Math 365 / Comp 365: Homework 3

Please bring a stapled hard copy of your answers to class

Problem 1

In class, we showed how to construct a sequence of unit lower-triangular matrices $\{L_k\}_{k=1,2,\ldots,n-1}$ of the form

$$L_k = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -\ell_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -\ell_{n,k} & & 1 \end{pmatrix}, \text{ where } \ell_{j,k} = \frac{A_{j,k}}{A_{k,k}}, \quad \text{for } k \leq j \leq n,$$

such that

$$L_{n-1} L_{n-2} \ldots L_2 L_1 A = U,$$

with U being an upper-triangular matrix. We now want to show in two steps that

$$L := L_1^{-1} L_2^{-1} \ldots L_{n-2}^{-1} L_{n-1}^{-1}$$

is a unit lower-triangular matrix.

a. Show that

$$L_k^{-1} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & \ell_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & \ell_{n,k} & & 1 \end{pmatrix}.$$

Hint: Start by showing that $L_k = I - \ell_k e_k^T$, where e_k is an $n \times 1$ vector with a 1 in the k^{th} entry, and zeros elsewhere, and

$$\ell_k = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \ell_{k+1,k} \\ \vdots \\ \ell_{n,k} \end{pmatrix}.$$

We know that if we multiply the vector ℓ_k by the vector e_k^T we get the $n \times n$ 0 matrix except for ℓ_k in the $k_t h$ column. Also, we can conclude that $I - \ell_k e_k^T = L_k$, since:

$$L_k = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -\ell_{k+1,k} & 1 & \\ & & \ddots & & \ddots \\ & & -\ell_{n,k} & & 1 \end{pmatrix}, \text{ where } \ell_{j,k} = \frac{A_{j,k}}{A_{k,k}}, \quad \text{for } k \le j \le n,$$

Further, since we know that $L_k^{-1} L_k = I$, we can conclude that:

$$L_k^{-1} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & \ell_{k+1,k} & 1 & \\ & & \ddots & & \ddots \\ & & \ell_{n,k} & & 1 \end{pmatrix}.$$

As this is the matrix that results from performing row operations on L_k augmented with I to get L_{-1}.

b. Now show that

$$L := L_1^{-1} L_2^{-1} \ldots L_{n-2}^{-1} L_{n-1}^{-1} = \begin{pmatrix} 1 & & & & \\ \ell_{21} & 1 & & & \\ \ell_{31} & \ell_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{n,n-1} & 1 \end{pmatrix}$$

Because each L is a lower triangular matrix, when we perform matrix multiplication on $L_k^{-1} L_{k+1}^{-1}$ we wind up with the matrix:

$$L^{-1}_{k*(k+1)} = \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & \ell_{k+1,k} & 1 & & \\ & & \ddots & \ell_{k+1+1,k+1} & \ddots & \\ & & \ell_{n,k} & \ell_{n,k+1} & & 1 \end{pmatrix}.$$

This shows that multiplying the lower triangular L_k^{-1} and L_{k+1}^{-1} matricies does not change the values in either the k_{th} or the $k+1$ columns, proving that the resulting matrix will not change its column values after multiplication. This implies that for all $L = L_1^{-1} L_2^{-1} \ldots L_{n-2}^{-1} L_{n-1}^{-1}$ the matrix resulting from multiplying out every component matrix is:

$$L = L_1^{-1} L_2^{-1} \ldots L_{n-2}^{-1} L_{n-1}^{-1} = \begin{pmatrix} 1 & & & & \\ \ell_{21} & 1 & & & \\ \ell_{31} & \ell_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{n,n-1} & 1 \end{pmatrix}$$

Because the individual column vectors do not change as a result of the multiplication for any two lower triangular matricies. This proves what we set out to prove.

Problem 2

This was Exercise 2 on Activity A6.

This problem is from Section 2.5, page 131 of Linear Algebra and Its Applications, by David Lay, the textbook many of you used for MATH 236.

An important concern in the study of heat transfer is to determine the steady-state temperature distribution of a thin plate when the temperature around the boundary is known. Assume the plate shown in the figure above represents a cross section of a metal beam, with negligible heat flow in the direction perpendicular to the plate. Let the variables $x_1, x_2, \ldots, x_8$ denote the temperatures at nodes 1 through 8 in the picture. In steady state, the temperature at a node is approximately equal to the average of the four nearest nodes (to the left, above, right, below).

a. The solution to the approximate steady-state heat flow problem for this plate can be written as a system of linear equations $Ax = b$, where $x = [x_1, x_2, \ldots, x_8]$ is the vector of temperatures at nodes 1 through 8. Find the 8×8 matrix A and the vector b. Hint: A should be a banded matrix with many zeros in the top right and bottom left parts of A.

Matrix A can be constructed as:

```
A = rbind(c(4, -1, -1, 0, 0, 0, 0, 0), c(-1, 4, 0, -1, 0, 0, 0, 0), c(-1, 0, 4, -1, -
1, 0, 0, 0), c(0, -1, -1, 4, 0, -1, 0, 0), c(0, 0, -1, 0, 4, -1, -1, 0), c(0, 0, 0, -
1, -1, 4, 0, -1), c(0, 0, 0, 0, -1, 0, 4, -1), c(0,0,0,0,0,-1,-1,4))
print(A)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## [1,]    4   -1   -1    0    0    0    0    0
## [2,]   -1    4    0   -1    0    0    0    0
## [3,]   -1    0    4   -1   -1    0    0    0
## [4,]    0   -1   -1    4    0   -1    0    0
## [5,]    0    0   -1    0    4   -1   -1    0
## [6,]    0    0    0   -1   -1    4    0   -1
## [7,]    0    0    0    0   -1    0    4   -1
## [8,]    0    0    0    0    0   -1   -1    4
```

While the vector B we are solving for is:

```
b = c(5, 15, 0, 10, 0, 10, 20, 30)
print(b)
```

```
## [1]  5 15  0 10  0 10 20 30
```

b. Use your function from Exercise 1 (on the activity) to perform an LU factorization of A. Do you notice anything special about the structures of L and U?

Here is my LU Function:

```
myLU = function(A) {
  n = nrow(A)
  U = A
  L = diag(n)
  for(k in 1:(n-1)) {
    for(j in (k+1):n){
      L[j,k] = U[j,k]/U[k,k]
      U[j,k:n] = U[j,k:n]-L[j,k]*U[k,k:n]
    }
  }
  print(nrow(L))
  return(list(L=L,U=U))
}
```

```
aLU = myLU(A)
```

```
## [1] 8
```

```
aL = aLU$L
aU = aLU$U
#print(aL)
#print(nrow(aL))
```

c. Once you have an LU factorization for a matrix A, you need to do the two-step procedure to complete

the back substitution to solve $Ax = b$. Here is code for that:

```
mySolve=function(L,U,b,tol=1e-10){
  n=nrow(L)

  # First solve Ly=b
  y = rep(0,n)         # pre-allocate a vector y with 0s in it.
  if(abs(L[1,1])<tol) stop('There is a zero on the diagonal of L')
  y[1] = b[1]/L[1,1]  # Fill in the 1st value of y
  for (j in 2:n ) {
    if(abs(L[j,j])<tol) stop('There is a zero on the diagonal of L')
    y[j] = (b[j] - L[j,1:(j-1)]%*%y[1:(j-1)])/L[j,j]
  }

  # Then solve Ux=y
  x = rep(0,n)         # pre-allocate a vector x with 0s in it.
  if(abs(U[n,n])<tol) stop('There is a zero on the diagonal of U')
  x[n] = y[n]/U[n,n]  # Fill in the nth value of x
  for (j in (n-1):1 ) {
    if(abs(U[j,j])<tol) stop('There is a zero on the diagonal of U')
    x[j] = (y[j] - U[j,(j+1):n]%*%x[(j+1):n])/U[j,j]
  }
  return(x)
}
```

Make sure you understand what the code is doing, and then use it to find the steady-state temperatures at nodes 1 through 8.

```
xvec = mySolve(aL, aU, b)
print(xvec)
```

```
## [1]  3.956938  6.588517  4.239234  7.397129  5.602871  8.760766  9.411483
## [8] 12.043062
```

d. The temperature on the right-hand side of the plate was measured incorrectly. It is actually 30 degrees. Find the new steady-state temperatures. Hint: you should not do another LU factorization!

To do this all we need to do is backsolve for a new b vector, b2. Where b2 is:

```
b2 = c(5, 15, 0, 10, 0, 10, 0, 10, 30, 40)
```

So the new x2vec should be:

```
x2vec = mySolve(aL, aU, b2)
print(x2vec)
```

```
## [1] 3.593301 6.224880 3.148325 6.306220 2.693780 5.851675 1.775120 4.406699
```

e. Use the R function `solve(A)` to compute A^{-1}. Note that A^{-1} is a **dense** matrix (without many zeros). When A is large, L and U can be stored in much less space than A^{-1}. This fact is another reason for preferring the LU factorization of A to A^{-1} itself.

```
Ainv = solve(A)
print(Ainv)
```

```
##              [,1]        [,2]       [,3]       [,4]       [,5]       [,6]
## [1,] 0.295264808 0.086553374 0.09450586 0.05094869 0.03180993 0.02273552
## [2,] 0.086553374 0.295264808 0.05094869 0.09450586 0.02273552 0.03180993
## [3,] 0.094505857 0.050948688 0.32707474 0.10928890 0.10450421 0.05913216
## [4,] 0.050948688 0.094505857 0.10928890 0.32707474 0.05913216 0.10450421
## [5,] 0.031809932 0.022735522 0.10450421 0.05913216 0.32707474 0.10928890
## [6,] 0.022735522 0.031809932 0.05913216 0.10450421 0.10928890 0.32707474
## [7,] 0.009998350 0.008183468 0.03180993 0.02273552 0.09450586 0.05094869
## [8,] 0.008183468 0.009998350 0.02273552 0.03180993 0.05094869 0.09450586
##              [,7]        [,8]
## [1,] 0.009998350 0.008183468
## [2,] 0.008183468 0.009998350
## [3,] 0.031809932 0.022735522
## [4,] 0.022735522 0.031809932
## [5,] 0.094505857 0.050948688
## [6,] 0.050948688 0.094505857
## [7,] 0.295264808 0.086553374
## [8,] 0.086553374 0.295264808
```

Problem 3

This problem is taken from the in-class portion of an old midterm.

You are trying to solve a linear system of four equations:

$$\begin{bmatrix} & & & \\ & A & & \\ & & & \\ & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ -2 \\ 4 \\ 2.25 \end{bmatrix},$$

but unfortunately A is a black box and you cannot see the contents. Oh my! Luckily, from the physics of the problem, you do have three pieces of information about the system and the solution:

- $||x||_\infty = 2$
- The condition number of A, using the ∞-norm, is 32
- $$\begin{bmatrix} & & & \\ & A & & \\ & & & \\ & & & \end{bmatrix} \begin{bmatrix} 2.0750 \\ 0.1625 \\ 0.4500 \\ -0.1875 \end{bmatrix} = \begin{bmatrix} 3 \\ -2 \\ 4 \\ 2.1875 \end{bmatrix}$$

Is this information enough to determine the solution x to $Ax = b$? If yes, find x. If no, say as much as you can about x.

NOTE: All norms used to calculate this problem are infinity norms.

This is not enough information to actually solve for x. However, with this information we can say a lot about the composition of our vector x.

First, we know that the following equation holds true:

$$RFE \leq cond(A) * RBE$$

And we know the RBE is (using infinity norms):

$$RBE = \frac{||b - Ax_a||}{||b||} = \frac{||\begin{pmatrix} 3 \\ -2 \\ 4 \\ 2.25 \end{pmatrix} - \begin{pmatrix} 3 \\ -2 \\ 4 \\ 2.25 \end{pmatrix}||}{4} = \frac{0.0625}{4} = 0.015625$$

Further, we know that $||x||$ using the infinitny norm is 2. And we know that the RFE is (using infinity norms):

$$RFE = \frac{||x - x_a||}{||x||} = \frac{||x - \begin{pmatrix} 2.0750 \\ 0.1625 \\ 0.4500 \\ 0.1875 \end{pmatrix}||}{2}$$

We also know know that $cond(A) = 32$. So if we substitute these three values back into our original equation we get:

$$\frac{||x - \begin{pmatrix} 2.0750 \\ 0.1625 \\ 0.4500 \\ 0.1875 \end{pmatrix}||}{2} \leq 32 * 0.015625$$

Which simplifies to:

$$||x - \begin{pmatrix} 2.0750 \\ 0.1625 \\ 0.4500 \\ 0.1875 \end{pmatrix}|| \leq 1$$

This implies that the largest value of $x - x_a \leq 1$ (using infinity norm), which implies that the largest value in x_a must be within 1 away from 2, the largest value of x. From this we can conclude that $x_1 = 2$, because it is the only point in the vector x_a that is within 1 unit away from 2. Further, we can conclude that every x_i value

must be within 1 unit away from its corresponding x_{ai} value.

Thus we can conclude that the vector x is:

$$x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \; where \; x_1 = 2, \; -0.8375 \leq x_2 \leq 1.1625, \; -0.55 \leq x_3 \leq 1.45, \; -0.8125 \leq x_4 \leq 1.1875$$

Problem 4

A real $n \times n$ matrix Q is *orthonormal* if

$$Q^T Q = QQ^T = I; \; i.e., Q^{-1} = Q^T.$$

Note: sometimes you will see these matrices just called *orthogonal matrices*. The analagous matrices that contain complex entries ($Q^* Q = QQ^* = I$, where the * is a conjugate transpose) are called *unitary*.

Show that $||Q||_2 = ||Q^{-1}||_2 = 1$, and therefore the 2-norm condition number of any orthonormal matrix Q is $\kappa_2(Q) = ||Q||_2 ||Q^{-1}||_2 = 1$.

We know that $||Q||_2 = \sqrt{max\{eigenvalue(QQ^T)\}}$ and that $Q^T = Q^{-1} \rightarrow QQ^{-1} = I$.
$\rightarrow ||Q||_2 = \sqrt{max\{eigenvalue(I)\}} = 1$. And since $Q^{-1} = Q^T$ then $||Q^{-1}|| = ||Q^T||$.
$\rightarrow ||Q^T||_2 = \sqrt{max\{eigenvalue(QQ^T)\}} = 1$. $||Q||_2 = ||Q^{-1}||_2 = 1 \square$.

Problem 5

This problem has 3 parts.

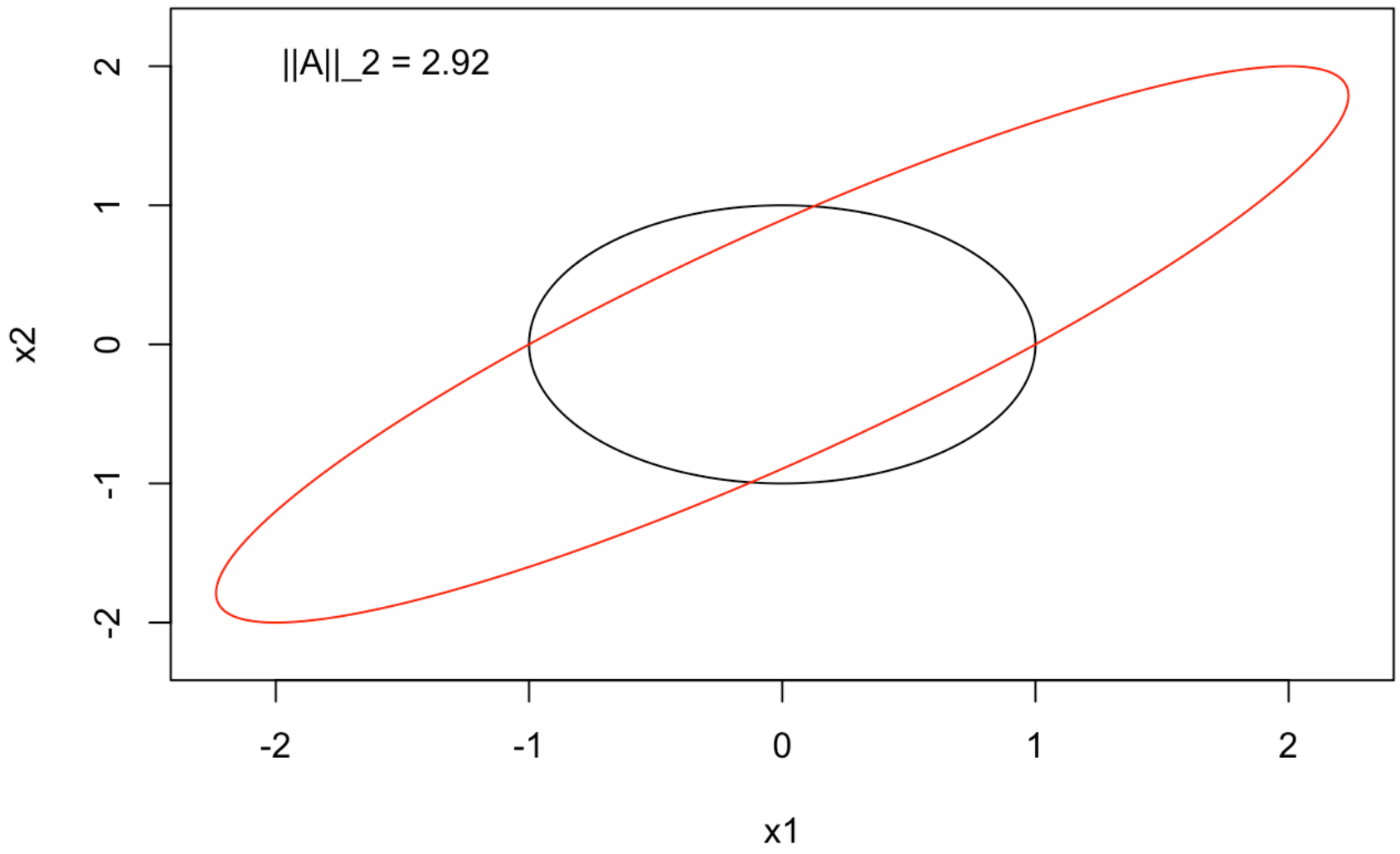
a. Modify my `UnitCircleMap` function, which shows the image of the unit ball under a linear mapping A and computes the matrix norm of A. We want to let the user choose the norm to be 1, 2, or ∞ ("I"). The code is below, and there are a few places where you need to insert a few lines.

```
UnitCircleMap = function(A,p=2) {

  if (p==2) {
    t = seq(0,2*pi,len=1000)
    x = cos(t)
    y = sin(t)
    nn=norm(A,type="2")
    ni=as.character(p)
  }
  else if (p==1){
    # Insert some code here
    t1 = seq(-1, 0, len=250)
    x1 = t1
    y1 = -1 - t1
    x2 = x1
    y2 = -1*y1
```

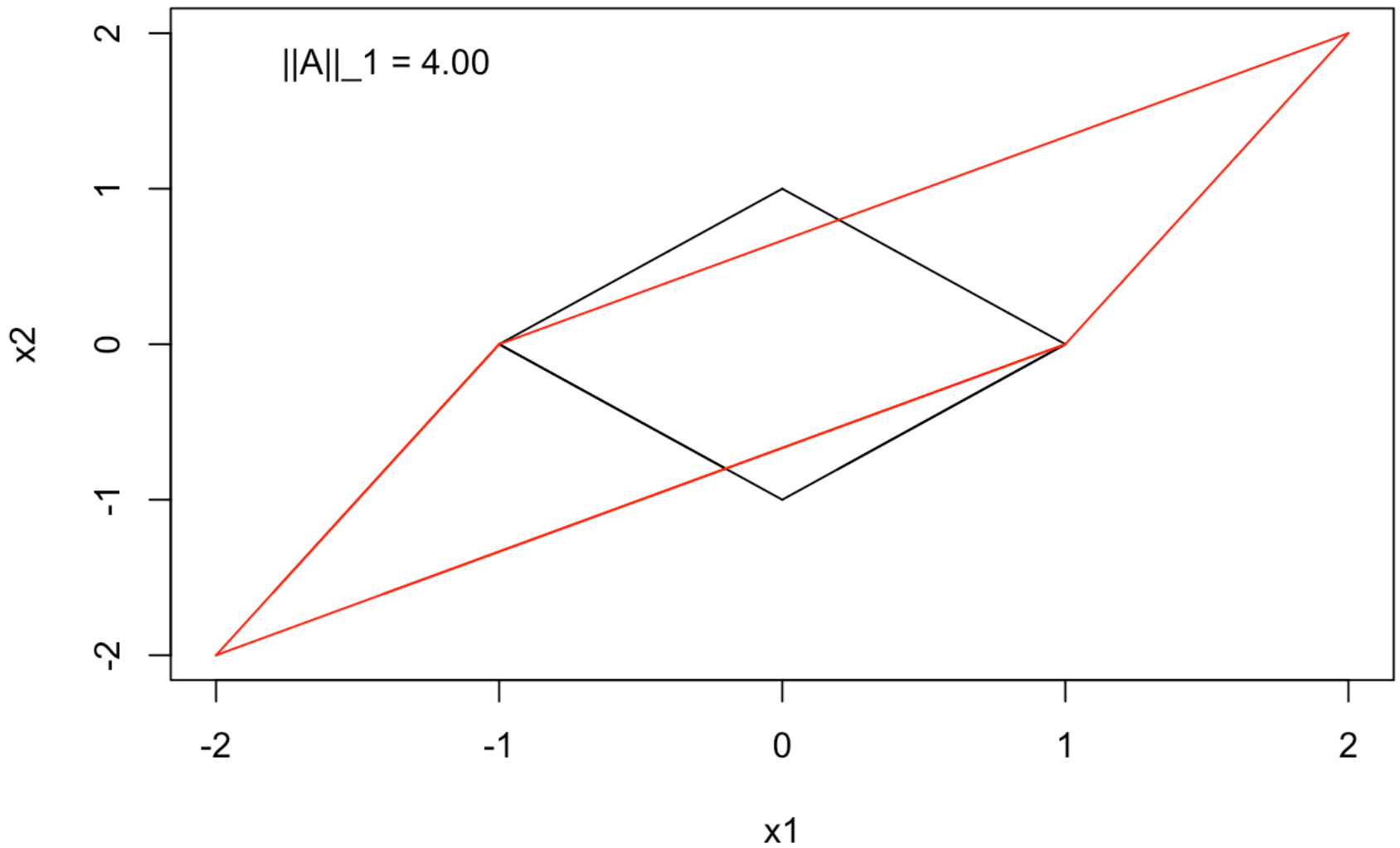
```
  t2 = seq(0, 1, len=250)
  x3 = t2
  y3 = 1 - t2
  x4 = x3
  y4 = -1*y3
  x = c(x1, x2, x3, x4)
  y = c(y1, y2, y3, y4)
  nn=norm(A,type="1")
  ni=as.character(p)
}
else if (p=="I"){
  # Insert some code here
  t1 = seq(-1, 1, len=250)
  x1 = t1
  y1 = rep(1, 250)
  t2 = seq(1, -1, len=250)
  x2 = rep(1, 250)
  y2 = t2
  x3 = t2
  y3 = rep(-1, 250)
  x4 = y3
  y4 = x1
  x = c(x1, x2, x3, x4)
  y = c(y1, y2, y3, y4)
  nn=norm(A,type="I")
  ni="Inf"
  }

  pts = A %*% t(cbind(x,y))

  newx = pts[1,]
  newy = pts[2,]

  M = max(c(newx,newy,1.5))
  m = min(c(newx,newy,-1.5))

  plot(x,y,type='l',col='black',xlim=c(m,M),ylim=c(m,M),xlab="x1",ylab="x2")
  lines(newx,newy,col='red')

  normSize=sprintf("||A||_%s = %1.2f",ni,nn)
  text(-.7*M,.9*M,normSize)
}
```

Here is an example of how the function should be called, and the output it should display. In this case, $A = \begin{bmatrix} 1 & 2 \\ 0 & 2 \end{bmatrix}$. Once you've filled in the missing lines of the function, you can also test your code on this A with the 1 norm and ∞-norm.

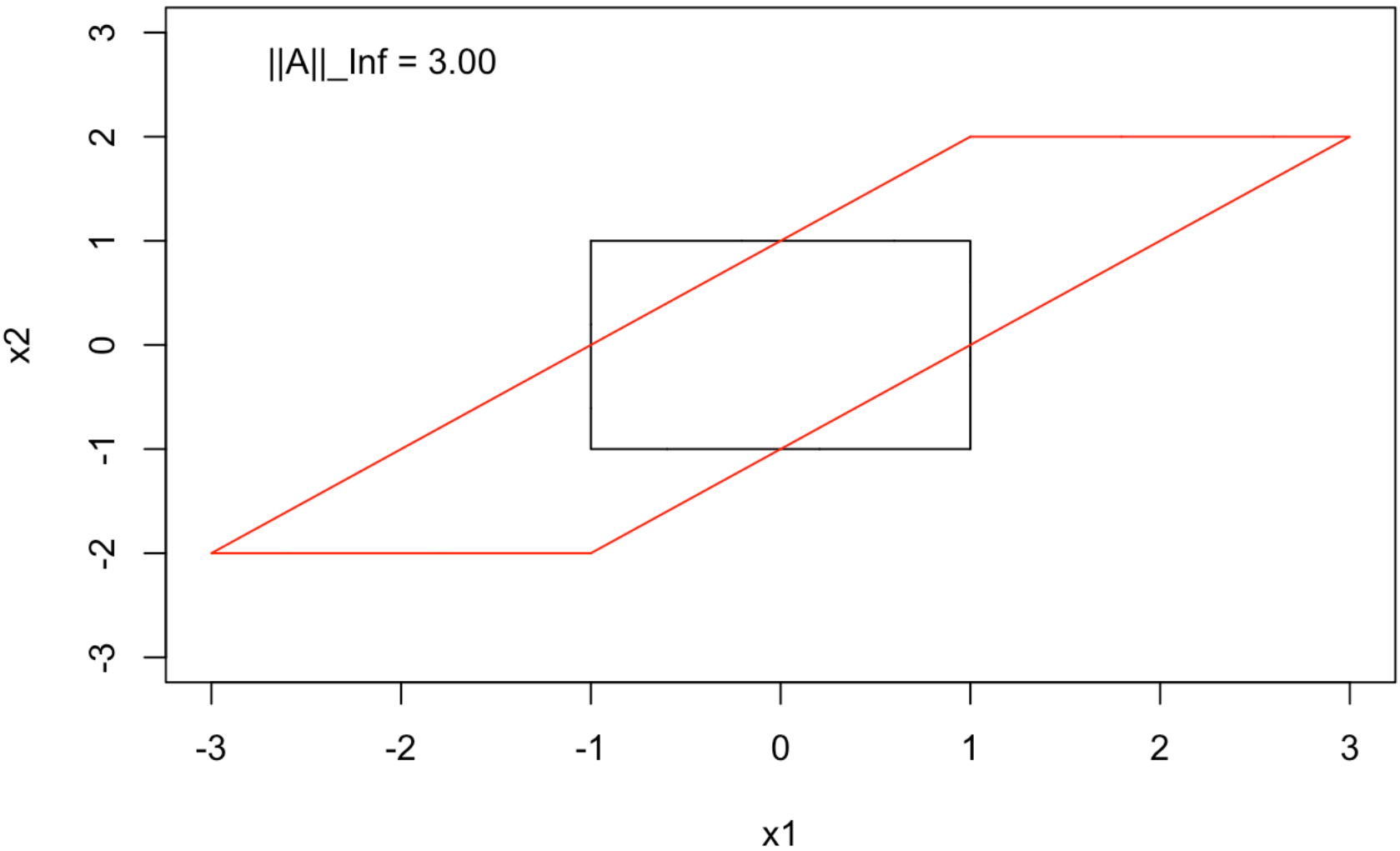
```
A=cbind(c(1,0),c(2,2))
UnitCircleMap(A,p=2)
```




```
UnitCircleMap(A, p=1)
```

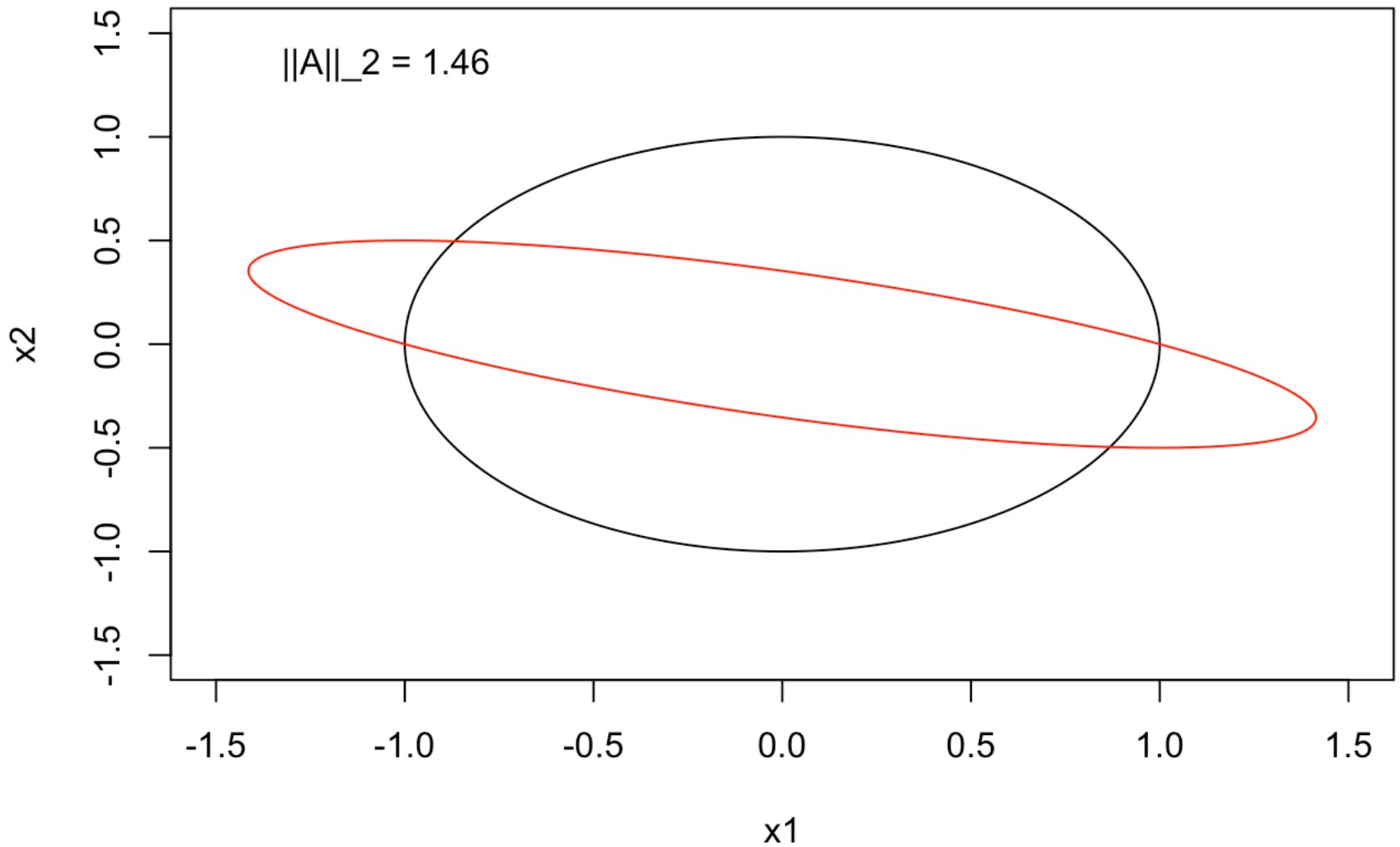



```
UnitCircleMap(A, p="I")
```

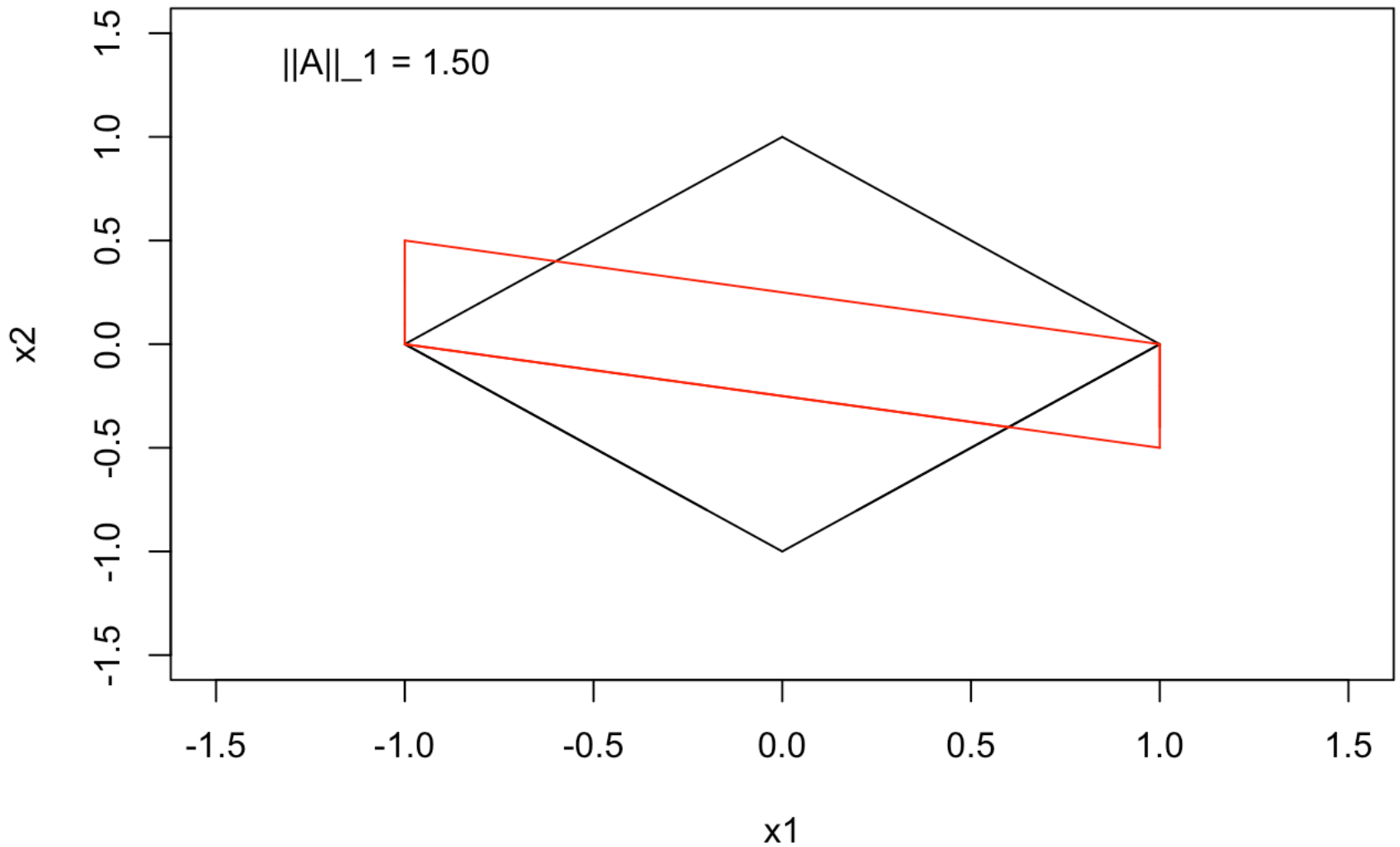


b. Compute the condition number of the matrix A above (using the $p = 1, 2, \infty$-norms). Visually explain why the condition number for each p is the value you found.

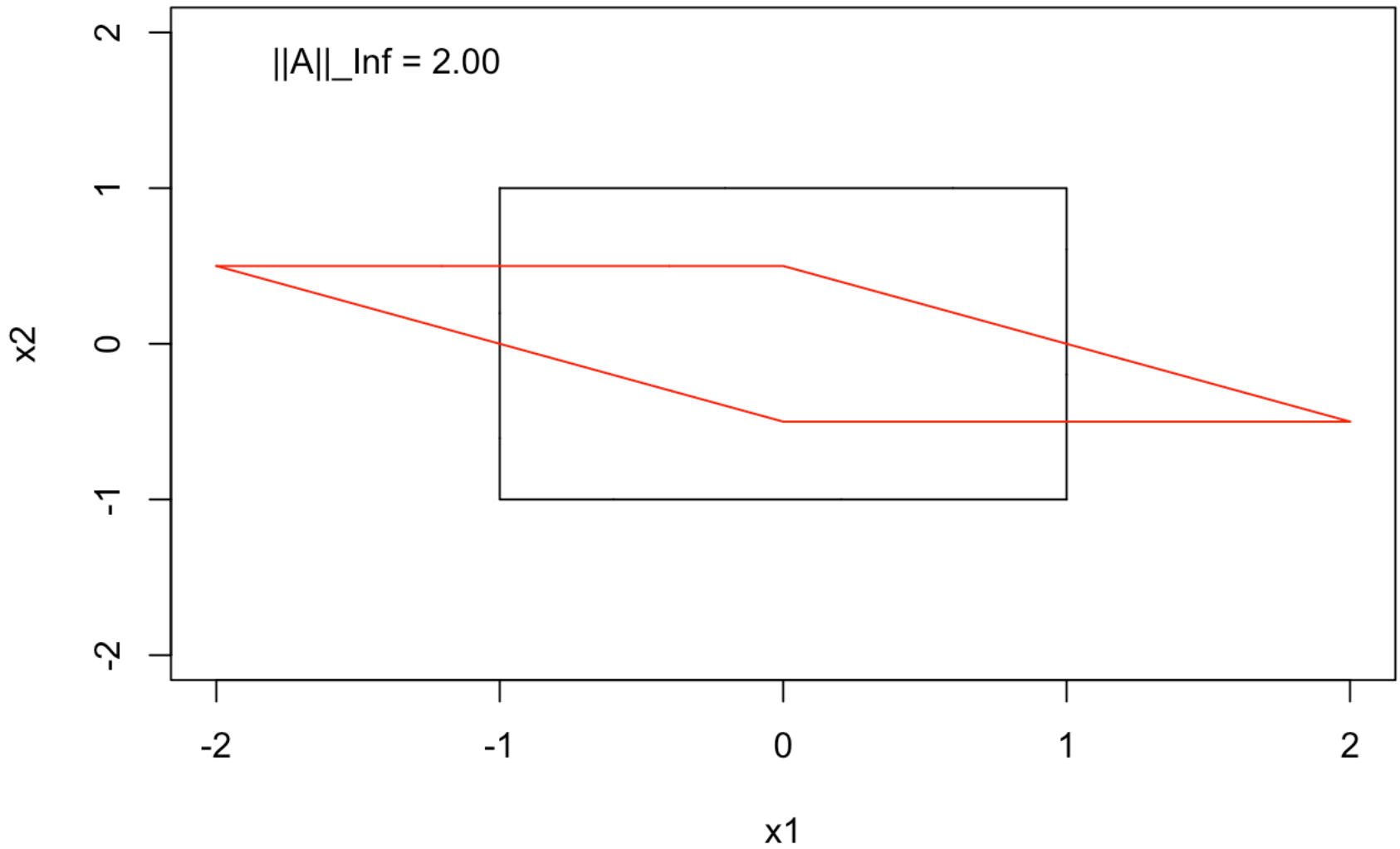
```
Cond = function(A,p=2) {
  if (p == 2) {  # by default use the
    s = svd(A)$d
    s = s[s>0]
    return(max(s)/min(s))
  }
  if (p == 1) {  # use the 1 norm
    Ainv = solve(A)
    return(max(colSums(abs(A)))*max(colSums(abs(Ainv))))
  }
  if (p == 'I') {   # use the infinity norm
    Ainv = solve(A)
    return(max(rowSums(abs(A)))*max(rowSums(abs(Ainv))))
  }
}

Ainv = solve(A)
UnitCircleMap(Ainv,p=2)
```




```
UnitCircleMap(Ainv, p=1)
```




```
UnitCircleMap(Ainv, p="I")
```




```
c1 = Cond(A, p=1)
c2 = Cond(A, p=2)
cinf = Cond(A, p="I")

print(c1)
```

```
## [1] 6
```

```
print(c2)
```

```
## [1] 4.265564
```

```
print(cinf)
```

```
## [1] 6
```

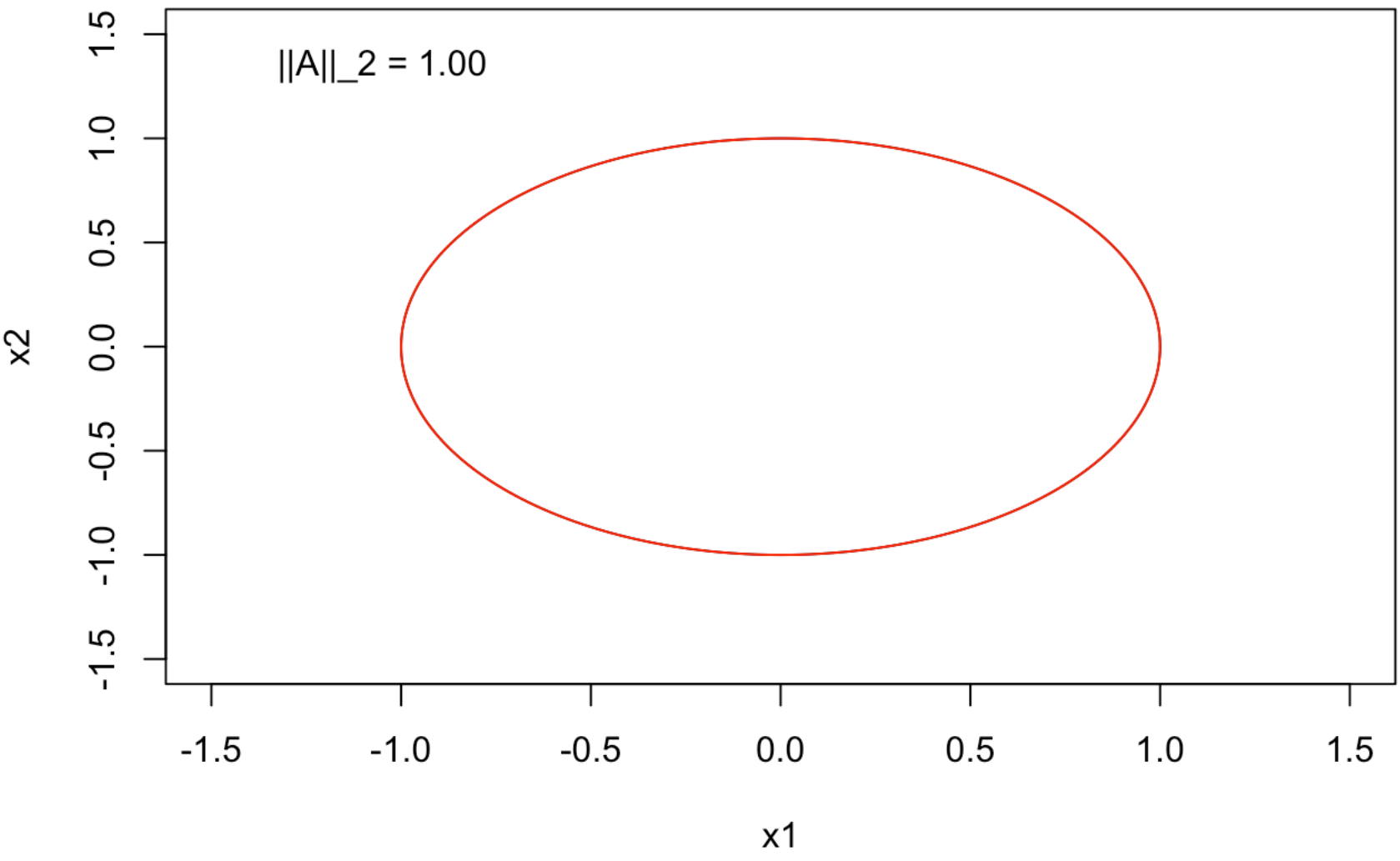
From the above visualizations we can see that the magnitudes of A^{-1} explain the condition number for the one, two, and infinity norms. For the infinity norm, we see that the maximum transformation is 4 for A and 1.5 for A^{-1}, which when multiplied is equal to the condition number of 6. When looking at both the two norm and

infinity norm, we see the same result. The maximum transformations for A and A^{-1}, the magnitude, multiply to be equal to the condition number.

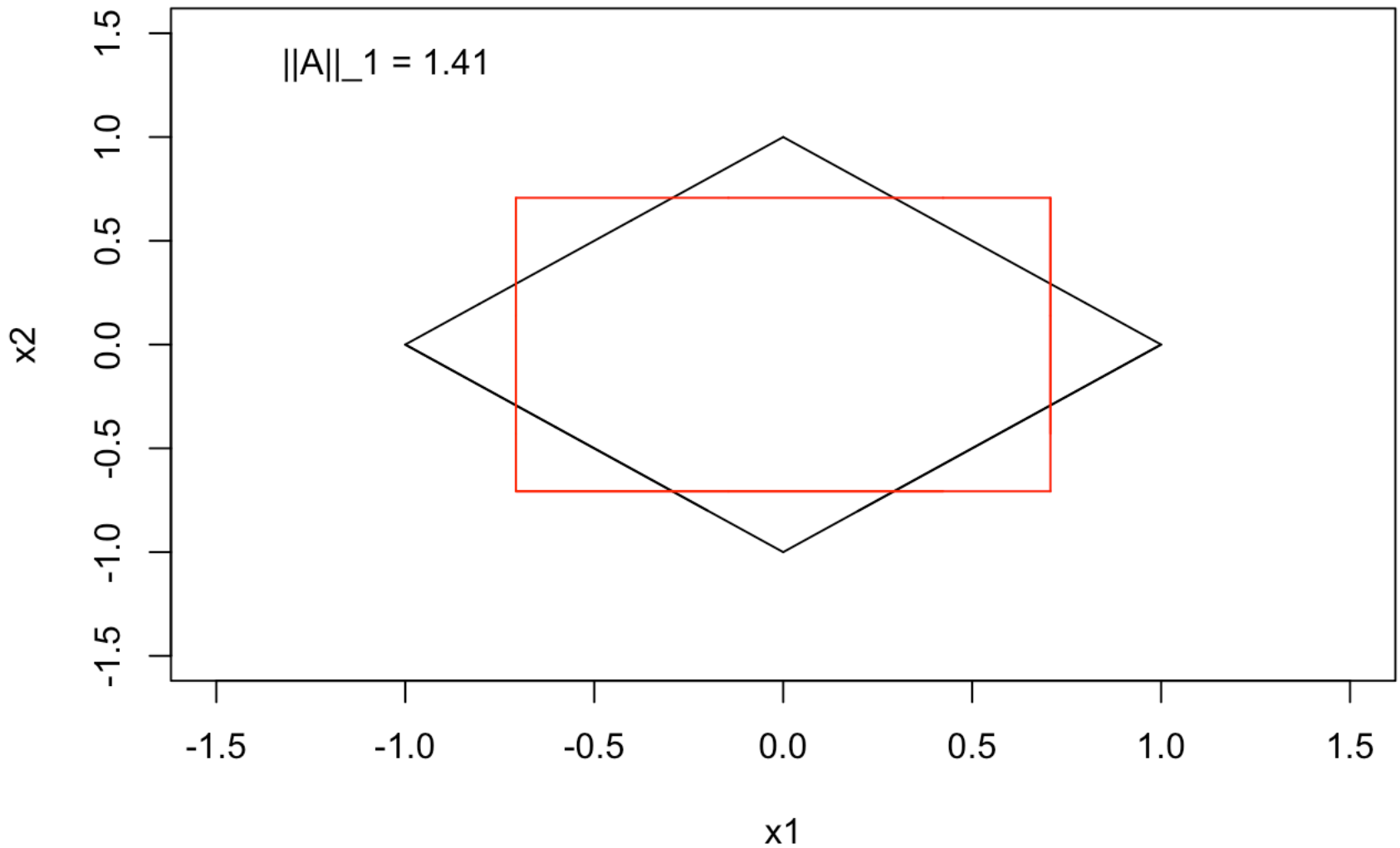
c. Use your code to determine whether $||Q||_1 = ||Q||_\infty = ||Q||_2 = 1$ for all orthonormal matrices Q.

Hint: Try $Q = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$, which is a rotation matrix that rotates vectors counter-clockwise by $\frac{\pi}{4}$.

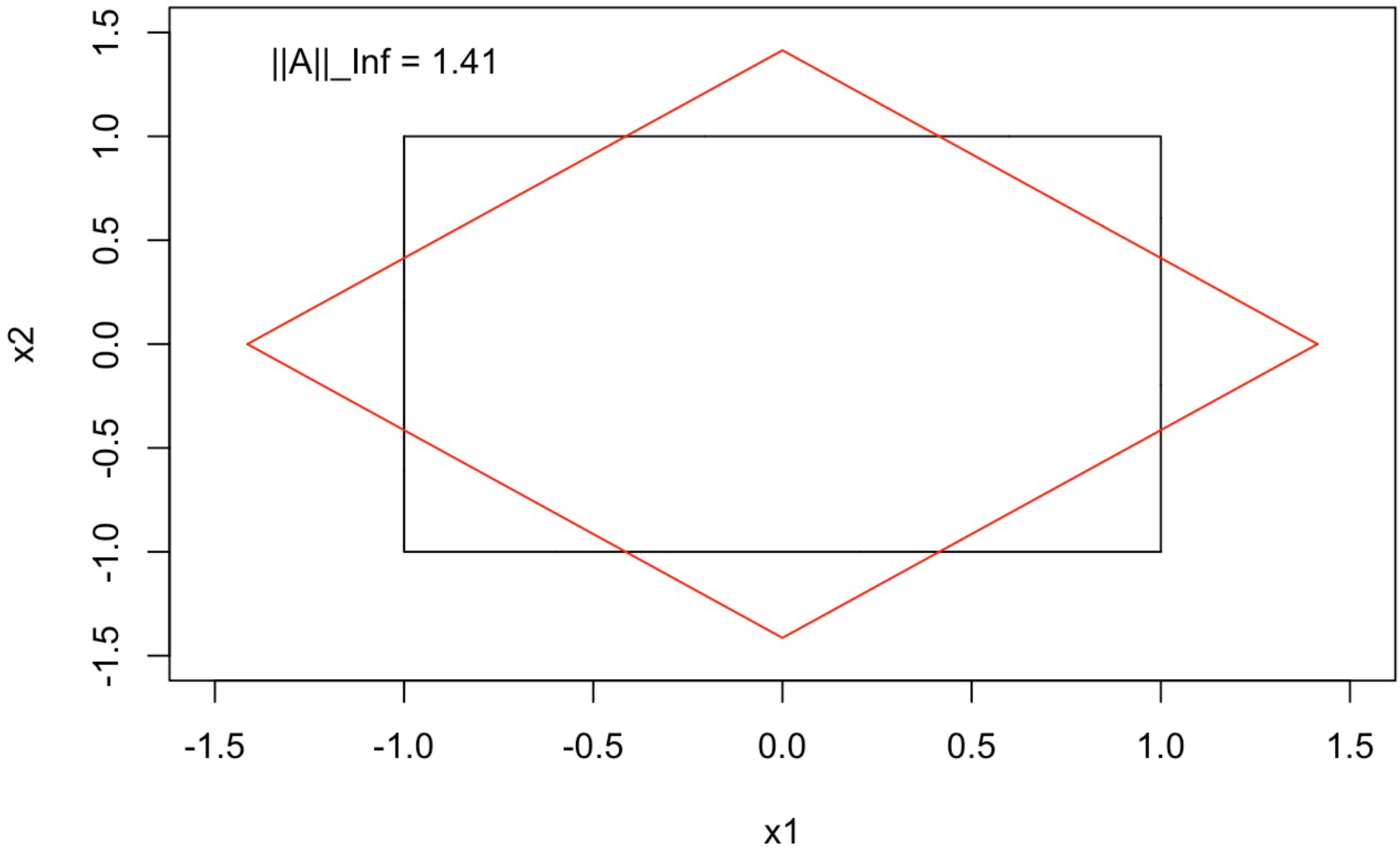
```
Q = cbind(c(1/sqrt(2),1/sqrt(2)), c(-1/sqrt(2), 1/sqrt(2)))
UnitCircleMap(Q,p=2)
```




```
UnitCircleMap(Q, p=1)
```




```
UnitCircleMap(Q, p="I")
```



From the above images it is clear that for any orthonormal matrix Q, the statement $||Q||_1 = ||Q||_\infty = ||Q||_2 = 1$ is false because while $||Q||_2 = 1$, $||Q||_2 \neq ||Q||_\infty$ and $||Q||_2 \neq ||Q||_1$. Further, neither $||Q||_1$ nor $||Q||_\infty$ equal 1. This proves that the statement $||Q||_1 = ||Q||_\infty = ||Q||_2 = 1$ is not true for all orthonormal matricies Q.

Problem 6

This was Exercise 3 on Activity A7.

Prove that

$$||A||_\infty = \max_{1 \leq i \leq n} \left\{ \sum_{j=1}^{n} |A_{ij}| \right\} = \text{maximum absolute row sum.}$$

For the left hand side:

$$||A||_\infty = \max_{x \neq 0} \frac{||Ax||_\infty}{||x||_\infty}$$

Where $||Ax||_\infty$ is the vector containing the linear combination of the rows of A.

Now we can conclude that:

$$||A||_\infty \leq \max_{x \neq 0} ||b||_\infty$$

Where b represents the vector resulting from normalizing $||Ax||_\infty$ by $||x||_\infty$. This implies that $||A||_\infty$ must be less than or equal to the maximum value $||b||_\infty$, where the vector b contains the row sums of A. This means that $||A||_\infty$ must be less than or equal to the maximum absolute row sum of A.

For the right hand side:

If we substitute out $\frac{||Ax||_\infty}{||x||_\infty}$ again for A we can say that:

$$\frac{||Ax||_\infty}{||x||_\infty} \geq \max_{1 \leq i \leq n} \left\{ \sum_{j=1}^{n} |A_{ij}| \right\}$$

However, if we look at the summation on the right side of the equation we see that it sums absolute value for every j_{th} column in A for every i_{th} row in A. From these row sums, it finds the maximum sum that occurs at some arbitrary row k, and returns that. We then get that:

$$\frac{||Ax||_\infty}{||x||_\infty} \geq k$$

Where k is the maximum absolute row sum of A. However, on the LHS we proved that:

$$||A||_\infty \leq \max_{x \neq 0} ||b||_\infty$$

Where

$$b$$

is the vector containing the maximum absolute row sums of A. Thus we have proved that $||A||_\infty$ is both less than or equal to the absoulte row sum of A and greater than or equal to the absolute row sum of A. Thus we can conclude that:

$$||A||_\infty = \max_{1 \leq i \leq n} \left\{ \sum_{j=1}^{n} |A_{ij}| \right\} = \text{maximum absolute row sum.}$$

Is a true statement.

Problem 7

Solve the following system by finding the $PA = LU$ factorization and then carrying out the two-step back substitution (all by hand):

$$A = \begin{pmatrix} -1 & 0 & 1 \\ 3 & 1 & 1 \\ 2 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \\ 5 \end{pmatrix}.$$

For the first step of the factorization:

$$A1 = \begin{pmatrix} 3 & 1 & 1 \\ 0 & \frac{1}{3} & \frac{4}{3} \\ 0 & \frac{-2}{3} & \frac{1}{3} \end{pmatrix}$$

$$P1 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$L1 = \begin{pmatrix} 1 & 0 & 0 \\ \frac{-1}{3} & 1 & 0 \\ \frac{2}{3} & 0 & 1 \end{pmatrix}$$

And after one more step we get the final PA = LU Factorization of:

$$P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$A = \begin{pmatrix} -1 & 0 & 1 \\ 3 & 1 & 1 \\ 2 & 0 & 1 \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & 0 & 0 \\ \frac{2}{3} & 1 & 0 \\ \frac{-1}{3} & \frac{-1}{2} & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} 3 & 1 & 1 \\ 0 & \frac{-2}{3} & \frac{1}{3} \\ 0 & 0 & \frac{3}{2} \end{pmatrix}$$

Then to solve the back substitution of $Ly = b$ we get that:

$$Ly = \begin{pmatrix} 1 & 0 & 0 \\ \frac{2}{3} & 1 & 0 \\ \frac{-1}{3} & \frac{-1}{2} & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \\ 5 \end{pmatrix}.$$

Or that

$$Ly = \begin{pmatrix} 2 \\ \frac{11}{3} \\ \frac{15}{2} \end{pmatrix}.$$

And then to solve

$$Ux = b$$

:

$$Ux = \begin{pmatrix} 3 & 1 & 1 \\ 0 & \frac{-2}{3} & \frac{1}{3} \\ 0 & 0 & \frac{3}{2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2 \\ \frac{11}{3} \\ \frac{15}{2} \end{pmatrix}.$$

Or that

$$Ly = \begin{pmatrix} 0 \\ -3 \\ 5 \end{pmatrix}.$$