**LABORATORIO # 5**

[Introduction](http://www.fim.utp.ac.pa/Members/fernando.castillo/metodos-numericos/laboratorio-parte-2/Laboratorio%20#5#INTRODUCTION)  
[Vector Norms](http://www.fim.utp.ac.pa/Members/fernando.castillo/metodos-numericos/laboratorio-parte-2/Laboratorio%20#5#VECTORNORMS)  
Matrix Norms  
[Quiz](http://www.fim.utp.ac.pa/Members/fernando.castillo/metodos-numericos/laboratorio-parte-2/Laboratorio%20#5#QUIZ)  
[Compatible Matrix Norms](http://www.fim.utp.ac.pa/Members/fernando.castillo/metodos-numericos/laboratorio-parte-2/Laboratorio%20#5#COMPATIBLEMATRIX)  
[Types of Errors](http://www.fim.utp.ac.pa/Members/fernando.castillo/metodos-numericos/laboratorio-parte-2/Laboratorio%20#5#TYPES)  
[The Condition Number](http://www.fim.utp.ac.pa/Members/fernando.castillo/metodos-numericos/laboratorio-parte-2/Laboratorio%20#5#CONDITION)  
[ASSIGNMENT](http://www.fim.utp.ac.pa/Members/fernando.castillo/metodos-numericos/laboratorio-parte-2/Laboratorio%20#5#ASSIGNMENT)  
[Quiz Answers](http://www.fim.utp.ac.pa/Members/fernando.castillo/metodos-numericos/laboratorio-parte-2/Laboratorio%20#5#ANSWERS)

**Introduction**

The objects we work with in linear systems are vectors and matrices. In order to make statements about the size of these objects, and the errors we make in solutions, we want to be able to describe the "sizes" of vectors and matrices, which we do by using norms.

We then need to consider whether we can bound the size of the product of a matrix and vector, given that we know the "size" of the two factors. In order for this to happen, we will need to use matrix and vector norms that are compatible. These kinds of bounds will become very important in error analysis.

We will then consider the notions of forward error and backward error in a linear algebra computation.

From the definitions of norms and errors, we can now define the condition number of a matrix, which will give us an objective way of measuring how "bad" the Hilbert matrix is, and how many digits of accuracy we can expect when solving a particular linear system.

**Vector Norms**

A vector norm assigns a size to a vector, in such a way that scalar multiples do what we expect, and the triangle inequality is satisfied. There are three common vector norms:

\* the L1 vector norm

||x||1 = sum ( 1 <= i <= N ) |xi|.

\* the L2 (or "Euclidean") vector norm;

||x||2 = sqrt ( sum ( 1 <= i <= N ) xi2 )

\* the L Infinity vector norm;

||x||inf = max ( 1 <= i <= N ) |xi|.

To compute the norm of a vector x in MATLAB:

\* ||x||1 = norm(x,1);  
\* ||x||2 = norm(x,2) = norm(x);  
\* ||x||inf = norm(x,inf) ("inf" is a special MATLAB name for infinity)

Exercise: For the following vectors:

x1 = [ 1; 2; 3 ]  
x2 = [ 1; 0; 0 ]  
x3 = [ 1; 1; 1 ]

compute the vector norms, using the appropriate MATLAB commands.

L1 L2 L Infinity

x1 ---------- ---------- ----------  
x2 ---------- ---------- ----------  
x3 ---------- ---------- ----------

**Matrix Norms**

A matrix norm assigns a size to a matrix, again, in such a way that scalar multiples do what we expect, and the triangle inequality is satisfied. However, what's more important is that we want to be able to mix matrix and vector norms in various computations. So we are going to be very interested in whether a matrix norm is compatible with a particular vector norm, that is, when it is safe to say:

||A\*x|| <= ||A|| \* ||x||

There are five common matrix norms:

\* the L1 or "max column sum" matrix norm:

||A||1 = max ( 1 <= j <= N ) sum ( 1 <= i <= N ) |Ai,j|;

\* the L2 matrix norm:

||A||2 = max ( 1 <= i <= N ) ( sqrt ( lambdai ) ),

where lambdai is an (always real) eigenvalue of ATA; or

||A||2 = max ( 1 <= i <= N ) ( mui),

where mui is a singular value of A;

\* the L Infinity or "max row sum" matrix norm:

||A||inf = max ( 1 <= i <= N ) sum ( 1 <= j <= N ) |Ai,j|;

\* the Frobenius matrix norm:

||A||F = sqrt sum ( 1 <= i <= N ) ( 1 <= j <= N ) Ai,j2;

\* the spectral matrix norm;

||A||spec = max ( 1 <= i <= N ) |lambdai|,

(only defined for a square matrix), where lambdai is a (possibly complex) eigenvalue of A.

To compute the norm of a matrix x in MATLAB:

\* ||A||1 = norm(A,1);  
\* ||A||2 = norm(A,2) = norm(A);  
\* ||A||inf = norm(A,inf);  
\* ||A||F = norm(A,'fro')  
\* ||A||spec = (do the quiz)

**Quiz**

It's worth while to try to try to express these norms in one-line functions yourself, using the fact that:

\* max(A) returns a row vector of the maximum of each column of A;  
\* max(v) returns a single value;  
\* sum(A) returns a row vector of the sum of elements of each column of A;  
\* sum(v) returns a single value;  
\* eig(A) returns a column vector of the eigenvalues of A;  
\* svd(A) returns a column vector of the singular values of A;  
\* diag(A) returns a column vector of the diagonal elements of A;  
\* trace(A) returns the sum of the diagonal elements of A;

QUIZ: Express the five matrix norms using simple MATLAB commands:

\* ||A||1 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;  
\* ||A||2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;  
\* ||A||inf = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;  
\* ||A||F = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;  
\* ||A||spec = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

Exercise: For the matrix A:

4 1 1  
0 2 2  
1 0 4

compute, by hand or by MATLAB:

||A||1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
||A||2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
||A||inf \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
||A||F \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
||A||spec \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Answers at the end of the lab.)

**Compatible Matrix Norms**

One way to define a matrix norm is to do so in terms of a particular vector norm. We use the formula:

||A|| = supremum ||A\*x|| / ||x||

where the supremum is taken over nonzero vectors x. A matrix norm defined in this way is said to be vector-bound to the given vector norm.

The most interesting and useful property a matrix norm can have is when we can use it to bound certain expressions involving a matrix-vector product. We want to be able to say the following:

||A\*x|| <= ||A|| \* ||x||

but this expression is not true for an arbitrary matrix norm and vector norm. It must be the case that the two are compatible.

If a matrix norm is vector-bound to a particular vector norm, then the two norms are guaranteed to be compatible. Thus, for any vector norm, there is always at least one matrix norm that we can use. But that vector-bound matrix norm is not always the only choice. In particular, the L2 matrix norm is actually difficult to compute, but there is a simple alternative.

Note that:

\* The L1, L2 and L Infinity matrix norms can be shown to be vector-bound to the corresponding vector norms and hence are guaranteed to be compatible with them;  
\* The Frobenius matrix norm is not vector-bound to the L2 vector norm, but is compatible with it; the Frobenius norm is much easier to compute than the L2 matrix norm.  
\* The spectral matrix norm is not vector-bound to any vector norm, but it "almost" is. This norm is useful because we often want to think about the behavior of a matrix as being determined by its largest eigenvalue, and it almost is. But there is no vector norm for which it is always true that

||A\*x|| <= ||A||spec \* ||x||

Exercise: Consider each of the following column vectors:

x1 = [ 1, 2, 3 ]'  
x2 = [ 1, 0, 0 ]'  
x3 = [ 1, 1, 1 ]'

For our given matrix A, verify that the compatibility condition holds by comparing the values of ||A|| that you computed in the previous exercise with the ratios of ||A\*x||/||x||. For the Frobenius matrix norm, you can copy the vector ratios from the L2 case. What must be true about the numbers in each row?

Matrix Vector ||A|| ||A\*x1||/||x1|| ||A\*x2||/||x2| |||A\*x3||/||x3||  
norm norm  
L1 L1 \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_   
L2 L2 \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_   
F L2 \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_   
Linf Linf \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

**Types of Errors**

A natural assumption to make is that the term "error" refers always to the difference between the computed and exact "answers". We are going to have to discuss several kinds of error, so let's refer to this first error as solution error or forward error. Suppose we want to solve a linear system of the form A\*x=b, (with exact solution x) and we computed x1. We define the solution error as

SE = || x1 - x ||

Sometimes we don't care whether we are close to the exact solution. What we want is something whose behavior is acceptably close to that of the exact solution. In this case, we are interested in the residual error or backward error, which is defined by

RE = || A \* x1 - b || = || b1 - b ||

where, for convenience, we have defined the variable b1 to equal A\*x1. Another way of looking at the residual error is to see that it's telling us the difference between the right hand side that would "work" for x1 versus the right hand side we were trying to handle.

If we think of the right hand side as being a target, and our solution procedure as determining how we should aim an arrow so that we hit this target, then

\* The forward error is telling us how badly we aimed our arrow;  
\* The backward error is telling us how much we would have to move the target in order for our badly aimed arrow to be a bullseye.

There are problems for which the forward error is huge and the backward error tiny, and all the other possible combinations also occur.

The norms of the matrix and its inverse exert some limits on the relationship between the forward and backward errors. Assuming we have compatible norms:

|| x1 - x || = || A-1 \* A \* ( x1 - x ) || <= || A-1|| \* || b1 - b ||

and

|| A \* x1 - b || = || A \* x1 - A \* x || <= || A || \* || x1 - x ||  
RE <= || A || \* SE

or

SE <= || A-1|| \* RE  
RE <= || A || \* SE

Quiz #2: For all questions, assume that A is an orthogonal matrix...

\* What is ||A||2?\_\_\_\_\_\_\_\_\_\_  
\* What is ||A-1||2?\_\_\_\_\_\_\_\_\_\_  
\* If we have a linear system A\*x=b, with A and b given, express ||x||2 in terms of quantities we know?\_\_\_\_\_\_\_\_\_\_  
\* If the entries of b are roughly of magnitude 1, then we can assume that the absolute error in each entry of b is bounded by eps, our friend the unit roundoff. What is a bound on the L2 norm of the entire right hand side error?\_\_\_\_\_\_\_\_\_\_  
\* Since we actually will be solving the linear system A\*x1=b1, where b1 is the rounded value, what is the bound on the L2 norm of the solution error?\_\_\_\_\_\_\_\_\_\_

Often, it's useful to consider the size of an error relative to the true quantity. Thus, if the true solution is x and we computed x1=x+dx, the relative solution error is defined as

RSE = || x1 - x || / || x ||  
= || dx || / || x ||

Given the computed solution x1, we know that it satifies the equation A\*x1=b1. If we write b1=b+db, then we can define the relative residual error as:

RRE = || b1 - b || / || b ||  
= || db || / || b ||

These quantities depend on the vector norm used, and cannot be defined, in cases where the divisor is zero.

Exercise: - consider the Frank matrix of size 5, and let x=[0,0,0,0,10] and b=[10,10,10,10,10]. For the vector x1=[1,0,0,0,10], and using the L1 matrix and vector norms, determine:

||A|| : \_\_\_\_\_\_\_\_\_\_\_\_\_  
||A-1|| : \_\_\_\_\_\_\_\_\_\_\_\_\_  
||x|| : \_\_\_\_\_\_\_\_\_\_\_\_\_  
||x1|| : \_\_\_\_\_\_\_\_\_\_\_\_\_  
||SE|| : \_\_\_\_\_\_\_\_\_\_\_\_\_  
RSE : \_\_\_\_\_\_\_\_\_\_\_\_\_  
||RE|| : \_\_\_\_\_\_\_\_\_\_\_\_\_  
RRE : \_\_\_\_\_\_\_\_\_\_\_\_\_

**Condition Numbers**

Given a square matrix A, the L2 condition number k2(A) is defined as:

k2(A) = ||A||2 \* ||A-1||2

if the inverse of A exists. If the inverse does not exist, then we say that the condition number is infinite. Similar definitions apply for k1(A) and kinf(A).

MATLAB finds it convenient to report rcond(A), the reciprocal condition number, which is

\* rcond(A) = 1 / k1(A) if A is nonsingular.  
\* rcond(A)= 0.0 if A is singular.

So as a matrix "goes singular", rcond(A) goes to zero in a way similar to the determinant.

We won't worry about the fact that the condition number is somewhat expensive to compute, since it requires evaluating the inverse matrix. Instead, we'll concentrate on what it's good for. We already know that it's supposed to give us some idea of how singular the matrix is. But its real role is in error estimation for the linear system problem.

We suppose that we are really interested in solving the linear system

A \* x = b

but that the right hand side we give to the computer has a small error or "perturbation" in it. We might denote this perturbed right hand side as b+db. We can then assume that our solution will be "slightly" perturbed, so that we are justified in writing the system as

A \* (x + dx) = b + db

The question is, if db is really small, can we expect that dx is small? Can we actually guarantee such a limit?

If we are content to look at the relative errors, and if the norm used to define k(A) is compatible with the vector norm used, it is fairly easy to show that:

||dx|| / ||x|| <= k(A) \* ||db|| / ||b||

You can see that we would like the condition number to be as small as possible. (What is the smallest possible value of the condition number?). In particular, since we have about 14 digits of accuracy in MATLAB, if a matrix has a condition number of 10^14, or rcond(A) of 10^(-14), then an error in the last significant digit of any element of the right hand side makes all the digits of the solution potentially wrong.

Exercise - it doesn't matter too much which matrix norm you use to get a condition number. The values should be close in a predictable way. Verify this statement by computing the condition number of the Frank matrix in several norms, for several matrix sizes:

Matrix Size 2 4 8 16  
k1(A) \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_  
k2(A) \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_  
kinf(A) \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

**Assignment**

To see how the condition number can warn you about loss of accuracy, let's try solving the problem A\*x=b, for x=ones(n,1), and with A set to the Hilbert matrix. We know that the solution error gets bad very quickly. Let's simply use MATLAB's estimate rcond(A) for (the inverse of) the L1 condition number, and assume that the relative error in b is eps, the machine precision. If that's true, then we expect that the relative solution error to be bounded by eps \* 1/rcond(A).

Matrix Size rcond(A) eps/rcond(A) ||dx||/||x||  
2 \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_  
4 \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_  
8 \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_  
16 \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

Do your results seem reasonable? Fill out the table and mail it to me.

**Quiz Answers**

\* ||A||1 = max ( sum ( abs ( A ) ) );  
\* ||A||2 = max ( sqrt ( eig ( A' \* A ) ) );  
||A||2 = max ( svd ( A ) );  
\* ||A||inf = max ( sum ( abs ( A' ) ) );  
\* ||A||F = sqrt ( sum ( diag ( A' \* A ) ) );  
||A||F = sqrt ( trace ( A' \* A ) );  
||A||F = sqrt ( sum ( sum ( A .^ 2 ) ) );  
\* ||A||spec = max ( abs ( eig ( A ) ) ).

**Quiz #2**:

\* ||A||2=1  
\* ||A-1||2-1?  
\* ||b|| = ||A\*x||<=||A|| \* ||x|| = ||x||b;  
||x|| = ||A-1 \* b|| <= ||A-1|| \* ||b||;  
||b|| = ||x||  
\* A\*x1=b1  
A\*(x+dx)=(b+db)  
A\*dx=db  
dx = A-1\*db so ||dx|| = ||A-1\*db|| <= ||A-1||\*||db|| = ||db||  
Similarly, ||db|| <= ||dx||,  
so ||dx|| = ||db|| = n \* eps