1. Read Heath Chapter 2, Sections 2.1–2.3.

2. **Vector norms.** Write C or Fortran routines to carry out the specified calculations:
   
   (a) The $L_p$ vector norm for $1 \leq p$ for vectors in $\mathbb{R}^n$. A C routine should have the following declaration:

   ```c
   double vecnormp(double *x, double p, int n);
   ```

   The declaration for a Fortran routine should be analogous. The C built-in function `pow(x, y)` returns $x^y$, whereas in Fortran you do $x**y$. A naïve algorithm (not one you were asked to propose in Homework 1) will suffice.

   (b) The $L_\infty$ vector norm for vectors in $\mathbb{R}^n$. A C routine should have the following declaration:

   ```c
   double vecnorminf(double *x, int n);
   ```

   The declaration for a Fortran routine should be analogous.

   (c) Make sure that your routines do what they are supposed to. My programs almost never work the first time they are run. So write a little test program that calls `vecnormp` and `vecnorminf`, and make sure that they return the correct norms.

3. **Matrix norms.** Write C or Fortran routines to carry out the specified calculations:
   
   (a) The $L_1$ matrix norm for matrices in $\mathbb{R}^{m \times n}$:

   ```c
   double matnrm1(double **A, int m, int n);
   ```

   (b) The $L_\infty$ matrix norm for matrices in $\mathbb{R}^{m \times n}$:

   ```c
   double matnrminf(double **A, int m, int n);
   ```

   (c) Test the two routines as you did in question (1c).

4. **Matrix-vector products.** The high-level LAPACK routines are based on lower-level BLAS (Basic Linear Algebra Subprograms) routines. The BLAS routine that perform matrix-vector multiplications is `dgemv`.

   (a) *Netlib* is a collection of mathematical software, papers, and databases. It is the site where LAPACK and other useful routines can be downloaded. Go to [www.netlib.org](http://www.netlib.org) and search the Netlib repository for “dgemv.c” You should find that the routine is available in the C language. Download `dgemv.c` and take a look.

   (b) Write a test program that calls `dgemv` using a matrix-vector pair with known product, and make sure that `dgemv` returns the correct product vector. You will be compiling several source files to general an executable. An example command to do this for C is

   ```bash
   gcc -o dgemvtest testprogram.c -llapack
   ```

   and for Fortran replace `g++` with `f77`. Both commands link any necessary routines in the LAPACK library and generate an executable named `dgemvtest`.  

---

**Due: Friday, September 17, 2010**

---

---
5. In this problem, you will use a basic random number generator to carry out Monte Carlo calculations, which are the ultimate in “brute force” computing.

In your program:

- If you are programming in C, add the line `#include <stdlib.h>` to the top of your program.
- The function `drand48()` in C will return a random double number in the interval $0 \leq x < 1$ each time. The analogous function in Fortran is `rand()`. The return values of these functions approximate a uniform random variable on $[0,1)$.


(a) Write a routine to fill the elements of vector $\vec{x} \in \mathbb{R}^n$ with uniform random numbers on the range $[-1,1)$ (note that this range is different from the range provided by the built-in random number generators, so you will need to do some scaling); in C,

```c
void randvec(double *x, int n);
```

In the next step, you will combine this routine with your norm and product functions from earlier questions.

(b) Declare matrix (double array) $A \in \mathbb{R}^{5 \times 5}$:

$$
A = \begin{bmatrix}
1 & 2 & 3 & -2 & -1 \\
4 & 3 & 2 & -3 & -4 \\
-2 & -1 & 0 & 1 & 2 \\
2 & -3 & 4 & -3 & 2 \\
1 & 2 & 3 & 4 & 5
\end{bmatrix}
$$

Let $n = 5$. Generate a random vector $\vec{x} \in \mathbb{R}^n$ using the routine you wrote in (a). Use $A$ and $\vec{x}$ as the inputs to `dgemv`, and call the output $\vec{y}$. Find the ratio

$$r = \frac{\|\vec{y}\|_p}{\|\vec{x}\|_p}$$

for $p = 1, 2, \infty$.

(c) Put (b) in a loop $k = 1, 2, \ldots, M$ to generate $r_k$’s, where $M$ is a large number ($> 10^5$). Let $R_k$ be the max of $r_1, r_2, \ldots, r_k$. Keep track of $R_k$. For $p = 1$, make a plot of $k$ (logscale-horizontal) against $R_k$ and $\|A\|_1$, where you calculate the matrix norm using your function from question 3. How big does $k$ have to be for $R_k$ to be within 10% of the matrix norm? Within 1%?

(d) Repeat (c) for $p = \infty$.

(e) Repeat (c) for $p = 2$. We don’t have a function to compute $\|A\|_2$, so take $M = 500000$ and report $R_M$.

7. Turn in

(a) Print-outs of your `vecnormp`, `vecnorminf`, `matnorm1`, `matnorminf`, `randvec` (but not the test programs).

(b) Your typed answers and plots to question 6, parts (c–e).