Lab 9—Solving Complicated Systems and Understanding Errors

Objective: Computers excel at solving complex systems, but the results must sometimes be carefully interpreted.

MATLAB Commands:

`hilb(n)`	Makes the nxn Hilbert matrix.
`inv(A)`	Finds the inverse of matrix A.
`format rat`	Displays all numbers as ratios of fairly simple integers, even when this is only an approximation.
`format long`	Displays all numbers as 10-digit floating point numbers, even when this is only an approximation.
`x1=linspace(a,b,n)`	Generates n points in the interval $[a,b]$. Used for plotting.
`y1=2.1*x1+4.3`	One example of how to define a line.
`plot(x1,y1)`	Plots the line we just defined as `x1` versus `y1`.
`plot(x1,y1,x2,y2)`	Use this syntax to plot `x1` versus `y1` and `x2` versus `y2` on the same set of axes.

Here we will study a linear system $\mathbf{Ax}=\mathbf{b}$ where A is a fairly large matrix, say bigger than the 3x3 matrices we typically work with. As a convenient example, let's choose it to be a Hilbert matrix. A Hilbert matrix is one whose ij^{th} entry is given by $a_{ij}=\dfrac{1}{i+j}$. Let's look at a few in order to get the idea. MATLAB can generate Hilbert matrices automatically (which is one reason why they were chosen for this example—we won't have to type in the 400 entries of a typical 20x20 matrix in the problem below). First set the format to rational:

```
format rat
hilb(3)
hilb(4)
A=hilb(5)
```

Got the idea?

Now consider the system $\mathbf{Ax}=\mathbf{b}$ where A is `hilb(5)` and $x=(1,1,1,1,1)$. Use MATLAB to find **b**. Now we change formats:

```
format long
```

This induces small round-off errors as the rational number entries in A and **b** are converted to floating point, so A and **b** are subtlely changed by this process.

Because of these changes, one might expect that $x = (1,1,1,1,1)$ is not quite a solution of $A\mathbf{x} = \mathbf{b}$ anymore. Let's check:

```
xNew=inv(A)*b
xDifference=x-xNew
```

There is a small error, but it's not bad.

---

1. Repeat the above experiment, but use the 20x20 Hilbert matrix `hilb(20)`. Dont' forget to first reset the format to `rat`. For **x**, use the column vector each of whose twenty entries is a 1. Find **b**. Then set the format to `long` to modify A and **b** by inducing round-off errors in them, and find the **x** that solves the new system $A\mathbf{x} = \mathbf{b}$. How large is the error in **x** that is induced by the round-off errors we created in A and **b**?

   From this we see that for delicate problems it will be important not to assume our results are insensitive to numerical error. We can minimize such errors by avoiding unnecessary operations, such as format changes.

2. This exercise will help us to understand geomatrically why small round-off errors can sometimes lead to large effects. You can solve it by hand or with MATLAB.

   Consider a system of two linear equations, where the "slope" coefficients appearing in the two equations are nearly equal and are not precisely known. For simplicity, we will assume all other quantities in the equations are precisely known. Such a system is

   $$x_2 = -(0.31 \pm 0.01)x_1 + 1.5$$
   $$x_3 = -(0.31 \pm 0.01)x_1 + 1.0$$

   Solve this system when the coefficient of x_1 appearing in the first equation take value -0.32 and the coefficient in the second takes value -0.30. Then solve the system if the situation were reversed (so the coefficient of x_1 in the first equation is -0.30 and the coefficient in the second equation is -0.32). For each solution, give a sketch representing the solution as the intersection of the two lines described by these equations.