1/9 2009

Problems for NumIntro
Jens Hugger

Problem JH-1.

Consider the following 4 Maple functions: **trunc**, **round**, **floor**, **ceil.**

1. Try them out on 2.6 and -2.4.
2. What is the relation between **trunc**, **floor** and **ceil.** (Note the difference for positive and negative numbers).
3. Explain in words the functions of **trunc**, **round**, **floor** and **ceil.**

Problem JH-2.

Consider the Fibonacci sequence from computer problem 1.2.1 with c=0:

(1) x[0]=1, x[1]=0, x[n+1]=x[n]+x[n-1] for n=1,2,3,…

1. Rewrite (1) as a matrix equation system A*X=B where $X=(x[2],x[3],\ldots,x[N+1])^T$
2. In Maple, first define the integer N=10.
   Now create a vector B of dimension N and a matrix A of dimension NxN.
   Assemble (load) A and B with the values from above.
   Solve for X using the Maple command **LinearAlgebra[LinearSolve](A,B);**
3. Now change N to 50 and run your program again.
   Change (if necessary) your program, so that you do not get too much intermediate output (changing ";" to ":" in appropriate places) but also so that you get the final result printed out, i.e. x[2],…,x[50].
4. Convert X to a list and insert the initial data 1 and 0 at the beginning of the list.
5. Plot the matrix A using **plots[matrixplot](A,heights=histogram,axes=boxed);**
6. Plot the result X using **plots[listplot](X);**

Problem JH-3.

1. Write a Maple procedure "MyHilbert" with input variable n, which outputs the Hilbert matrix H_n.
2. Compute **MyHilbert(5);**
3. Compute **LinearAlgebra[HilbertMatrix](5);**
4. Write a Maple procedure "RHS" with input variable n, which outputs the vector whose i'th entry is the sum of the values in the i'th row of the Hilbert matrix H_n.
5. Compute **RHS(5);**
6. Use the Maple command **LinearAlgebra[LinearSolve](MyHilbert(N),RHS(N));** and loop over values of N from 1 to 20.
7. In (6) you got a lot of output. Change your program, so that it prints nothing, except in the case where one of the components of the solution vector is not 1. In this case use the Maple command **print("Maple error",N,i);** as error message. Comment the solutions (Give credit to Maple:-)

8. Maple is perfect because it computes symbolically and perfect. Now change your program so that it computes with floats. Most likely, you only need to insert a single **evalf(…)** at one point in **MyHilbert**. Repeat (6) but only loop over N from 1 to 10. Comment on the degradation of the results.
9. Insert the command **Digits:=100;** and run the float part from (8) again. Note the improvement. (Maple is ideal for this type of test, because it is so easy to change the number of significant digits that Maple computes with).
   Note: You may reset to default precision with the command **Digits:=10;**
   Note: You can avoid seing in the output all the digits computed with by issuing the command **interface(displayprecision=10);** (10 can be replaced by nonnegative integer up to 100 or by -1 meaning "show all digits determined by Digits".
10. Redo (9) showing only 10 digits but computing with 100.

I tend to give the full command name for clarity. In practice you would normally load the relevant packages with commands like **with(LinearAlgebra); with(plots);** and so on, to be able to just use the commandnames in […].