HW 2, due Wednesday Sept. 2

1 Empirical error estimate, integration: (based on 6.2.5 in text, page 132)

1. The goal of this problem is to analyze the error dependence as a function of N for three different numerical integration algorithms: trapezoid rule, Simpson's rule, and Gauss-Legendre quadrature. You are to "experimentally" determine the optimum N for these methods and see if the approximation and roundoff errors vary with N as expected.

2.
   - *PH4433:* Do the following integral. Assume that you know the exact answer which you may use to calculate the exact error.
   $$\int_0^1 \cos^2(x)dx$$
   - *PH6433:* Do the following integral. Assume that you do *not* know the exact value of the integral and plot $|(A_N - A(_{2N}))/A_{2N}|$ as described in the text and class instead of the exact error. Note that for some integration rules the number of points N must be an odd number. In this case replace $2N$ with $2N+1$.
   $$\int_0^1 e^{-x^2}dx$$

3. Start with the sample program `integ.f95`. You can treat the Gaussian quadrature part as a black box for now. There are some problems with the program as written on the CDROM, use my version instead from the course web page.

4. Plot the error of each method as a function of N on a log-log plot (like Fig. 6.3 in the text). Questions:
   (a) What is the optimum N for each method? What is best possible error of each method (number of decimal places)? How do these values compare with the expressions we derived in class for trapezoid and Simpson's?
   (b) Can you see regions of N where approximation or roundoff error dominate? Determine the slopes in these regions by fitting your data, and show these fits on your plots.

5. Repeat everything in single precision.

Some hints:

- You will need to increase N a lot in the sample code. This may take several minutes to run. Compiling with *optimization* for speed will help:

```
gfortran -O2 integ.95
```

- In some cases the error may underflow to `0.0000`. This can't be plotted on a log scale and these points should be ignored when fitting the data. You may want to avoid printing the result when it is zero or close to machine precision. You can also ignore these points when fitting by using "regions" in xmgrace.