Final Examination, Scientific Computing, May 4, 2006

1. We compute x in single and double precision, getting (printing only five digits) $\widehat{x}_s = 3.4367 \times 10^6$ and $\widehat{x}_d = 3.4543 \times 10^6$ respectively. The calculation is complicated but the only error is roundoff. The computational algorithm is stable in that it is roughly as accurate as is allowed by the condition number. The computation would be exact in exact arithmetic. How many digits of the computed $\widehat{x}_s(k)$ and $\widehat{x}_d(k)$ are likely to be correct? Assume the double precision result is more accurate than the single precision result.

2. In each case below you are asked to judge whether a code is working correctly based only on a few small tests. In each case, state whether the code is behaving quantitatively as a correct code would and give your reasoning.

   (a) The code is supposed to evaluate $\int_0^2 f(x)dx$ using a third order panel method using n equal size panels. We tried it for $f(x) = xe^{-x^2/2}$ and compared the computed result to the analytical result.

n	20	40	80
error	1.2×10^{-2}	1.8×10^{-3}	2.3×10^{-4}

   (b) The code applies Newton's method to finding the maximum of a function of n variables. We applied it to a relatively easy problem with a single nondegenerate local maximum.

iteration	7	8	9	10
error	2.5×10^{-1}	8.3×10^{-2}	2.8×10^{-2}	9.1×10^{-3}

   (c) The code, `double Ahat(int n)`, applies simple Monte Carlo to estimate $A = E[X]$ by averaging n independent samples of X. To test the code we did four runs listed below. Each run estimated the statistical error in `Ahat` using the code fragment

```
int calls = 50;
double AhatSum = 0;
double AhatSS  = 0;
double AhatVal;
for ( int call = 0; call < calls; call++ ) {
   AhatVal  = Ahat(n);
   AhatSum += AhatVal;
   AhatSS  += AhatVal*AhatVal;  }
AhatBar = AhatSum/calls;
AhatVar = ( AhatSS - calls*AhatBar*AhatBar ) / calls;
cout << "Standard deviation from n = << n << " samples, "
     << " estimated using " << calls << " calls = "
     << sqrt( AhatVar )<< endl;
```

The results were

n	10^6	4×10^6	16×10^6	64×10^6
standard dev.	3.1×10^{-3}	1.1×10^{-3}	4.0×10^{-4}	1.3×10^{-4}

3. Suppose A is a symmetric positive definite matrix. Consider the system of second order differential equations

$$\ddot{x} = -Ax \; . \tag{1}$$

The scalar case $\ddot{y} = -\omega^2 y$ has solution

$$y(t) = \cos(\omega t)y(0) + \frac{1}{\omega}\sin(\omega t)\dot{y}(0) = a\cos(\omega t) + b\sin(\omega t) \; .$$

(a) Write an expression for the general solution of (1) using eigenvalues and eigenvectors, $Au_\alpha = \lambda_\alpha u_\alpha$, that shows that solutions remain bounded as $t \to \infty$. Do not worry about how the numbers a_α and b_α are found from initial data.

(b) If we discretize (1) using the Verlet scheme:

$$(x_{k+1} - 2x_k + x_{k-1})/\Delta t^2 = -Ax_k \; ,$$

what order of accuracy would we get? As usual, x_k is the approximation to $x(t_k) = x(k\Delta t)$.

(c) The statement: "x_k is bounded as $k \to \infty$" is true: (i) for any positive definite symmetric A and positive Δt, or (ii) for any positive definite symmetric A and Δt sufficiently small (possibly depending on A), or (iii) for no $\Delta t > 0$? Decide which and explain.

4. Define $I(h) = \int_0^h f(x)dx$. We want to estimate $I(h)$ using a linear combination of the data $f(0)$ and $I(2h)$. What is the highest order of accuracy we can achieve and what is the estimator that achieves it?

5. Show that if R is an $n \times m$ matrix and $B = R^*R$. Show that $\|B\|_{l^2} = \|R\|_{l^2}^2$. (Hint: Use the singular value decomposition of R.) Is it true that $\|B\| = \|R\|^2$ in other norms? Explain why or why not, possibly by a counterexample.

6. These questions are related to perturbation theory and the condition number of the Choleski decomposition. The perturbation part comes up in Monte Carlo computation using multivariate normals. Always A is a symmetric and positive definite matrix and $LL^* = A$ is the Choleski decomposition.

(a) Suppose $A = I + \epsilon\dot{A}$, and $L = I + \epsilon\dot{L} + O(\epsilon^2)$. Find the entries of $\dot{L}$ in terms of the entries of $\dot{A}$. The operation count should be $O(n^2)$ or less.

(b) Suppose $A(\epsilon) = A + \epsilon\dot{A}$ and $L(\epsilon) = L + \epsilon\dot{L} + O(\epsilon^2)$. Find an $O(n^3)$ work algorithm to find the entries of $M = L^{-1}\dot{L}$ from the entries of L and $\dot{A}$. Hint: differentiate, then multiply by L^{-1} from one side and by L^{-*} (the inverse of L^*, which also is the transpose of L^{-1}) from the other. Finally, use this to give an $O(n^3)$ way to find the entries of $\dot{L}$.

(c) Show that $\|M\| \leq \left\|\dot{A}\right\| \left\|L^{-1}\right\|^2$, and from that find a formula for the (informal) condition number of finding L from A in terms of the condition number of A. Hint, use the result of Problem 5.

7. Consider the code

```
int    n = 10 // The power of 2 for shrinking.
double x = 1. + rng();  // rng() makes standard uniform ...
double y = 1. + rng();  //    random variables
double s = 1 / ( (double) pow(2,n) );
y = s*y;
if ( y == ( x + y ) - x ) cout << "Exact!" << endl;
```

In IEEE floating point arithmetic with all intermediate results truncated to 64 bits, is the result more likely to be "`Exact!`" when `n = 0` or `n = 10`? The purpose of `n = 10` is to make y much less than x.