Takehome Exam: MATH 471
Due: Tuesday, November 18, 2008.

1. Show that Newton's method applied to the problem of computing the cube root of a number a, i.e. $x^3 - a = 0$, has the form

$$x_{k+1} = \frac{1}{3}(2x_k + ax_k^{-2}).$$

Hand in psuedo-code (hand written), as well as an m-file that implements this algorithm with a stopping tolerance of 10^{-10} for the value of $x^3 - a$. The first line of your m-file should look something like `function[x]=cuberoot(a)`. Hand in a listing of your code.

2. Derive the third order Taylor series method for numerically solving

$$\frac{dx}{dt} = t\cos(x) - x\sin(t), \quad x(0) = 1.$$

Hand in psuedo-code, as well as an implementation of the method which makes use of `taylor3method.m`. Hand in a plot of the numerical solution on [0,3] using the step size $h = 0.01$.

3. (a) Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be symmetric, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$, and $c \in \mathbb{R}$. Show that if $J(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{x}^T\mathbf{b} + c$, then $\nabla J(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$ and $\nabla^2 J(\mathbf{x}) = \mathbf{A}$, where "$\nabla$" and "$\nabla^2$" denote the gradient and Hessian operators, respectively. Note that $[\nabla J(\mathbf{x})]_i = \partial J(\mathbf{x})/\partial x_i$ and $[\nabla^2 J(\mathbf{x})]_{ij} = \partial^2 J(\mathbf{x})/\partial x_i \partial x_j$, where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. *Show all of your work and don't skip steps!*

   (b) Prove that if $\mathbf{A} \in \mathbb{R}^{m\times n}$ has linearly independent columns, $\mathbf{A}^T\mathbf{A}$ is positive definite.

   (c) The following statement is true:

   Theorem: Suppose $J : \mathbb{R}^n \to \mathbb{R}$ such that $\nabla^2 J(\mathbf{x})$ is positive definite for all $\mathbf{x}$, then $\mathbf{x}^* = \arg\min_{\mathbf{x}} J(\mathbf{x})$ if and only if $\nabla J(\mathbf{x}^*) = \mathbf{0}$. Moreoever, $\mathbf{x}^*$ is unique.

   Use this result, together with those from parts (a) and (b), to prove that if $\mathbf{A}$ has linearly independent columns, $J(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$ has a unique minimizer satisfying $\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}$. **Hint:** First show that $J(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} - \mathbf{x}^T\mathbf{b} - \mathbf{b}^T\mathbf{b}$. Then use (a) and (b).

4. Let

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

   (a) Compute the **QR**-factorization of $\mathbf{A}$ using the algorithm on page 281 of the text. You must write out all of your steps on paper, but you may use MATLAB for individual computations.

(b) Show that if $\mathbf{A} = \mathbf{QR}$, then $\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|^2$ implies $\mathbf{Rx}^* = \mathbf{Q}^T\mathbf{b}$ (**Hint:** use 3(c)). Use this to compute the least squares solution for $\mathbf{A}$ and $\mathbf{b}$ given above. Again, write out your steps, and be sure to exploit the structure of the $\mathbf{Q}$ and $\mathbf{R}$ matrices wherever appropriate.

(c) Use the **QR**-factorization from (a) to compute the SVD of $\mathbf{A}$ as follows: compute the SVD of $\mathbf{R}$ to obtain $\mathbf{R} = \mathbf{U}'\mathbf{SV}$, then the SVD of $\mathbf{A}$ is given by $\mathbf{A} = \mathbf{USV}$, where $\mathbf{U} = \mathbf{QU}'$. Verify your result by comparing with your HW #4 SVD of $\mathbf{A}$ or with MATLAB computation.

5. (a) Hand in psuedo-code as well as an m-file that implements the *steepest descent method* for the problem of minimizing $J(\mathbf{x}) = \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|^2$ when $\mathbf{A}$ has linearly independent columns. Use MATLAB implementation of your code to compute the solution when $\mathbf{A}$ and $\mathbf{b}$ are given in problem 2. Hand in a listing of your m-file. The first line should look something like `function[x]=SteepDesc(A,b,x0,maxiters)`.

(b) Apply the conjugate gradient algorithm to the same problem using `cg.m` from the web site and compare your results. Which algorithm converges more rapidly? Explain why.