Norges teknisk–naturvitenskapelige universitet
Department of Mathematical Sciences

Semester project TMA4215

Deadline October 3rd 2010

You can work in groups of maximum 3 persons. Your reports should not exceed 5 pages (reduce the size of the figures to stay within the limits of pages). We appreciate if you try to write well documented Matlab code.

We consider minimization problems of the type

$$\min_{\mathbf{x}\in\mathbf{R}^n} g(\mathbf{x}), \quad g(\mathbf{x}) := -\mathbf{b}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T H\mathbf{x} + \frac{1}{6}\mathbf{x}^T C(\mathbf{x})\mathbf{x},$$

here $\mathbf{b} \in \mathbf{R}^n$, H is a $n \times n$ symmetric and positive definite matrix[1] and $C(\mathbf{x})$ is a diagonal matrix with diagonal entries $c_i x_i$, $i = 1, \ldots, n$. Here c_i are the components of a vector $\mathbf{c} \in \mathbf{R}^n$ and x_i are the components of $\mathbf{x}$. The vectors $\mathbf{b}$ and $\mathbf{c}$ and the matrix H are the data of our problem.

To solve the problem we will combine the *steepest descent* method and the *Newton method*.

You can generate the data H, $\mathbf{b}$ and $\mathbf{c}$ as you please, just make sure H is positive definite. We expect different groups to use different data.

a) It is an advantage to be able to generate the data by a procedure which works for different values of n. Report the data you have been using and the procedure generating it.

b) Find the gradient and the Hessian of g. Implement g, its gradient and its Hessian in three Matlab functions. The answer to this question consists of the three formulae and the printout of the three matlab codes.

c) Case $n = 2$. In this case $g(\mathbf{x})$ is a surface in $\mathbf{R}^3$, and can be plotted in Matlab using the command `surf`. Understand how to use this command, use the "edit plot" option (under tools in the plot window). Rotate the surface, and learn to use the zoom tool. The results you obtain here will be part of answer **d)**.

   Try to identify potential candidates for minima of g. Learn how to use the option `shading`. This might turn useful in the next tasks of the project.

d) Implement now the steepest descent method (see the note on nonlinear equations). Consider first the case where α is a small positive constant value $\alpha = 0.1$ or $\alpha = 0.01$ (try various different values). Try to implement

[1] A matrix A is positive definite if $\forall \mathbf{u} \in \mathbf{R}^n, \quad \mathbf{u}^T A\mathbf{u} > 0$ when $\mathbf{u} \neq 0$.

the method so that it works for any value of n. See how the method converges.

Recalling that any local minimum $\mathbf{x}^*$ is such that $\nabla g(\mathbf{x}^*) = 0$, implement a stopping criterion by monitoring $\frac{\|\nabla g(\mathbf{x}^k)\|}{\|\nabla g(\mathbf{x}^0)\|}$ (relative residual) where $\mathbf{x}^0$ is the initial guess. Stop the iteration whenever this relative residual is less than a given tolerance or a maximum number of iterations is reached.

In the report you should plot the results of the iterative method: show how the relative residual decreases with the number of iterations (number of iterations on the x-axis and relative residual on the y-axis); show how the values of g decrease through the iteration. Use `semilogy`. Use also `plot3` in Matlab to plot the results of the iteration on the surface for the case $n = 2$ (by using the `shading` option of `surf` it is possible to obtain a better picture).

e) Show that the *steepest descent* with constant step-size α is equivalent to applying the forward Euler method to solve a certain differential equation. Find this equation.

f) Implement a Matlab function computing the optimal step length α^* for the steepest descent method. This is a univariate optimization problem and it amounts at solving a simple quadratic equation for α. Implement a strategy to pick the correct value of α (among the two possible solutions). Plot the results given by the steepest descent using the optimal α and compare them with the method with constant α. As an answer to this question give some of the details of derivation of the formula for the optimal α and the corresponding Matlab function. Include this iteration in the picture of the $n = 2$ case made with `surf`, to compare the behavior.

g) Implement finally the Newton method to solve $\nabla g(\mathbf{x}^*) = 0$. You shall use the steepest descent method to produce the initial guess of the Newton method. Make many experiments with different values of the tolerance and of the maximum number of iterations for the steepest descent and the Newton method. This should help you to come up with an optimal strategy for solving the problem in the quickest way possible. Measure CPU times using the commands `tic` and `toc` or `cputime` in Matlab. Include this iteration in the picture of the $n = 2$ case made with `surf`, to compare the behavior. Make a table where you compare the three methods ((1) steepest descent with constant α, (2) steepest descent with optimal α, (3) steepest descent with optimal α combined with Newton method), showing the values of the norm of the residual per iteration and the CPU times.