Math 410 - Introduction to Numerical Methods
Project I- Fall 2009

Prof. Vianey Villamizar

Due date: Wednesday October 14, 2009 (in class)

1 Minimum cost design. Multidimensional Newton application

Consider a gas pipeline transmission system, where the compressor stations are placed L miles apart. Assume that the total annual cost of the transmission system and its operation is

$$C(D, P_{out}, L, r) = 7D^2 P_{out} + 3e4D + \frac{6e6}{L} + \frac{770e6}{L}(r^{0.2} - 1), \tag{1.1}$$

where

$D =$ pipe diameter [in].

$P_{in} =$ compressor inlet pressure [psia].

$P_{out} =$ compressor outlet pressure [psia].

$L =$ length between stations [miles].

$r =$ compression ratio $= P_{out}/P_{in}$.

Furthermore, assume that the flow rate is

$$Q = 3.4\Big[\frac{(P_{out}^2 - P_{in}^2)D^5}{fL}\Big]^{1/2}, \tag{1.2}$$

with $f =$ friction factor $= 0.008D^{-1/3}$. Let the flow rate $Q = 100e6$ SCF/day, and use expression (1.2) to eliminate P_{out} from (1.1).

The problem consists of using the multi-dimensional Newton's method to find values for the parameters involved that produce minimum-cost design. Try an initial guess within the following ranges: $P_{in} \in [30, 90]$, $L \in [5, 15]$ and $D \in [50, 150]$. Run experiments using Newton method for values of the tolerance $TOL = 10^{-n}$, for $n = 2, 4, 6, 8, 10$. For EACH NEWTON ITERATION report the following:

1. Values of the parameters involved that produce minimum-cost design.
2. Approximations of the minimum cost.
3. Error between consecutive approximations of the 3-D parameter vector (P_{in}, L, D) using both Euclidean norm and Max norm.
4. Check if quadratic convergence is verified. Explain why or why not.

   ALSO DO THE FOLLOWING ONCE:

5. Graph a function from the values of the parameter D (pipe diameter) obtained at each iteration until convergence is reached.

6. Graph the function being minimized fixing the final values of two of the parameters that produce the minimum cost. Let the remainder parameter take values in the neighborhood of its minimum value. You should draw three graphs. Do the values obtained for the parameters actually correspond to a minimum of the "cost" function?

2 Interpolation

1. Create a subroutine for Neville's Method. $[Q] = SubNeville(p, N, x, y)$, where
   p: Intermediate point where the interpolation polynomial will be evaluated.
   N: Degree of the Polynomial.
   x: Vector of dimension $N + 1$ containing x-axis component of data points.
   y: Vector of dimension $N + 1$ containing y-axis component of data points.
   Q: Output matrix $(N + 1 \times N + 1)$ containing Neville's Table.

2. Validate your algorithm by applying it to problems 5(b)(c) of Sect 3.1 of the book. You should have an output containing the following information:
   a) Vector x and point p.
   b) Matrix Q.
   c) Degree of the polynomial, approximated value at p, error of the approximation by comparing against the exact value (problems 9(b)(c)).

3. Construct and implement an improved algorithm that is capable to automatically add interpolation points (initially provided by you) to obtain a better approximation. Keep using your subroutine $[Q] = SubNeville(p, N, x, y)$.

4. Create a subroutine for Newton Divided Differences: $[P, DD] = SDivDiff(p, N, x, y)$, where
   p: Intermediate point where the interpolation polynomial will be evaluated.
   N: Degree of the Polynomial.
   x: Vector of dimension $N + 1$ containing x-axis component of data points.
   y: Vector of dimension $N + 1$ containing y-axis component of data points.
   DD: It is an output matrix $(N + 1 \times N + 1)$ containing Divided Difference's Table.
   P: It is the value that the interpolation polynomial reaches at $x = p$.

5. Validate your algorithm by applying it to Problems 5(b)(c) of Sect 3.1 of the book. You should have an output containing the following information:
   a) Vector x and point p
   b) Matrix DD
   c) Degree of the polynomial, approximated value at p, error of the approximation by comparing against the exact value (problems 9(b)(c)).

6. Construct and implement an improved algorithm that is capable to automatically add interpolation points (initially provided by you) to obtain a better approximation. Keep using your subroutine $[P, DD] = SDivDiff(p, N, x, y)$.

7. Use either your Neville or Newton DD algorithms and 21 equally spaced nodes on the interval $[-5, 5]$ to find the interpolating polynomial $P_{20}(x)$ of degree 20 for the function $f(x) = 1/(x^2 + 1)$. Then, make graphs for both the function $f(x)$ and the polynomial $P_{20}(x)$. Include the two graphs in the same plot to compare. To construct the graph make a partition of 500 points for the interval $[-5, 5]$.

8. Do the Same as required in problem (5), but now use Chebyshev nodes; $x_i = 5 * \cos(i * pi/20)$, for $i = 1, \ldots 20$.

3 Muller's Method

Apply Muller's method to find all the roots of the following polynomial

$$x^8 - 36x^7 + 546x^6 - 4536x^5 + 22449x^4 - 67284x^3 + 118124x^2 - 109584x + 40320$$

Then, change x^7 coefficient to -37 and find all the roots. Observe how a minor perturbation in the coefficients can cause massive changes in the roots. In each case report your three initial guess points and the final approximation after convergence. Why Newton's method is not a good choice to find the roots after perturbation?