Numerical Analysis

Math 370 Spring 2009 MWF 11:30am - 12:25pm Fowler 110

 http://faculty.oxy.edu/ron/math/370/09/

Worksheet 11

SUMMARY Solving Systems of Non-Linear Equations, i.e. $\vec{f}(\vec{x}) = 0$
READING Recktenwald, Sec 8.5, pp. 427–445; Mathews & Fink, 167–185

Introduction
We have spent the last Unit learning techniques of solving the equation $f(x) = 0$ numerically. That is, we have been solving non-linear equations in one-variable. Of course, most interesting problems have more than one variable involved. In this next Unit we will learn how to solve systems involving many variables, in the form of non-linear or linear equations.

EXAMPLE

Consider

$$\begin{aligned} y &= \alpha x + \beta \\ y &= x^2 + \sigma x + \tau \end{aligned}$$

This nonlinear system consists of the equations for a line and a parabola, respectively. Our problem is to find the coordinates of the point of intersection for these two curves, for any line and parabola in this form.

1. What are the parameters in this system? What are the variables? What's the difference between these kind of mathematical objects?

2. Can you write this system in the form $A\vec{x} = \vec{b}$ where A is a 2x2 matrix and $\vec{x}$ is a 2x1 vector of variables and $\vec{b}$ is a 2x1 vector of constants?

3. How is this version of $A\vec{x} = \vec{b}$ different from the linear systems you solved in Math 212/214?

Note in this case we could think of this system as vector root-finding problem, i.e.

$$\vec{f}(\vec{x}) = A\vec{x} - \vec{b} = \vec{0}$$

Similar to the solution technique in solving $f(x) = 0$ we need to find numerical algorithms which generate a sequence of vectors $\{\vec{x}_n\}$ which have as their limit the value of $\vec{x}$ which makes $\vec{f} = 0$, i.e. solves the systems of non-linear equations.
The two most common iterative methods for solving these kinds of systems are called **Succesive Substitution**, and, **Newton's Method (for Systems)**.

Generic Algorithm for Iterative Solution of Nonlinear Systems

(Input initial guess for solution)
1. LET $x = x^{(0)}$
(Begin Iterating ...)
2. FOR k = 0, 1, 2, ...
(Evaluate the vector function to see how close to the solution we are)
3. $f^{(k)} = f(x^{(k)}) = A(x^{(k)})x^{(k)} - b(x^{(k)})$
(Convergence criterion)
4. IF $||f^{(k)}||$ is ''small enough'', STOP
(Calculate how to modify the current guess: **Will be different for each method**)
5. $\Delta x^{(k)} = ...$
(Produce a new guess from the old guess)
6. $x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$
7. END FOR
(end iteration)
8. END PROGRAM

Successive Substitution (Picard Iteration for Vector Functions)
The modify step (LINE 5) in the generic algorithm for iterative solution of nonlinear system becomes
SOLVE $A^{(k)}\Delta x^{(k)} = -f^{(k)}$
Note, that one can combine the modify (LINE 5) and update (LINE 6) steps to produce one step to find your next guess:
SOLVE $A^{(k)}x^{(k+1)} = b^{(k)}$

Newton's Method for Vector Functions
The modify step (LINE 5) in the generic algorithm for iterative solution of nonlinear system becomes
SOLVE $J^{(k)}\Delta x^{(k)} = -f^{(k)}$
where J is the Jacobian of the non-linear system. The Jacobian matrix of a system of non-linear equations is given by

$$\begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \cdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$$

So, in practice the "Update" line of Newton's Method becomes $\Delta x^{(k)} = -J^{-1}(\vec{x}_{(k)})\vec{f}(\vec{x}^{(k)})$
Norms
Note that the meaning of $||\vec{x}||$ is the norm or magnitude of $\vec{x}$. It is a real number. What is $||[-1, 2, 0, 4]||$?

Exercise

Consider the system

$$\begin{aligned} y &= 1.4x - 0.6 \\ y &= x^2 - 1.6x - 4.6 \end{aligned}$$

EXAMPLE

0. Write the system given above as $\vec{f}(\vec{x}) = \vec{0}$ and find its Jacobian matrix.

We know the system has two solutions : (-1,-2) and (4,5). Depending on the initial guess, numerical algorithms will converge to one or the other solution.

Exercise

1. Use the quadratic formula to confirm the two solutions to the systems are indeed (-1,-2) and (4,5).

EXAMPLE

2. Use Cramer's Rule or some other method to write the system in the form $\underline{x}^{(k+1)} = \underline{G}(\underline{x}^{(k)})$. [This should remind you of Picard Iteration's $x_{k+1} = G(x_k)$]

$$x = g_1(x,y) \qquad (1)$$

$$y = g_2(x,y) \qquad (2)$$

Solutions should be: $g_1(x,y) = \dfrac{4}{x-3}, \qquad g_2(x,y) = \dfrac{7.4 - 0.6x}{x-3}$

There's a reasonably obvious way to improve the successive substitution method $\underline{x}^{(k+1)} = \underline{G}(\underline{x}^k)$. (**HINT: are we using all the information we have as soon as we have it?**) This improvement is called **Seidel Iteration**.

3. Write down your improved iterative step in the space below.

Three Different Iterative Methods To Solve This System: Successive Substitution, Seidel Iteration and Newton's Method

Note, that we are trying to obtain the $(k+1)^{th}$ approximation of $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ given we currently have the k^{th} approximation. So far we have three different possible iterative scheme that tell us how to do that.

The first iterative method is the Succesive Substitution Scheme and it looks like:

$$\begin{aligned} x_1^{(k+1)} &= \frac{4}{x_1^{(k)} - 3} \\ x_2^{(k+1)} &= \frac{7.4 - 0.6x_1^{(k)}}{x_1^{(k)} - 3} \end{aligned}$$

The second iterative method involved applying the Seidel Enchancement to this, which produces the Seidel Iteration Scheme:

$$\begin{aligned} x_1^{(k+1)} &= \frac{4}{x_1^{(k)} - 3} \\ x_2^{(k+1)} &= \frac{7.4 - 0.6x_1^{(k+1)}}{x_1^{(k+1)} - 3} \end{aligned}$$

The third iterative method involves Newton's Method, which is a bit more complicated. In this case

$\underline{x}^{(k+1)} = \underline{x}^{(k)} + \Delta\underline{x}^{(k)}$ where $\Delta\vec{x}^{(k)} = -J^{-1}(\vec{x}^{(k)})\vec{f}(\vec{x}^{(k)})$

For our specific problem $\vec{f} = \begin{bmatrix} 1.4x_1 - 0.6 - x_2 \\ x_1^2 - 1.6x_1 - 4.6 - x_2 \end{bmatrix}$ and $J = \begin{bmatrix} 1.4 & -1 \\ 2x - 1.6 & -1 \end{bmatrix}$ and thus $J^{-1} = \frac{1}{2x_1 - 3} \begin{bmatrix} -1 & 1 \\ 1.6 - 2x & 1.4 \end{bmatrix}$ so that

$$\begin{aligned} x_1^{(k+1)} &= x_1 + \frac{3x_1 + 4 - x_1^2}{2x_1 - 3} \\ x_2^{(k+1)} &= x_2 + \frac{(1.6 - 2x_1)(1.4x_1 - 0.6 - x_2) - 1.4(x_1^2 - 1.6x_1 - 4.6 - x_2)}{2x_1 - 3} \end{aligned}$$

(NOTE: the superscript (k) has been supressed on the right hand side of the iterative steps in Newton's Method. It should appear over every x_1 and x_2. I also didn't simply the right-hand side so that you could see that it comes from the multiplication of the matrrix J and $-\vec{f}$.)

GroupWork

4. Let's do 2 iterations by hand of each method using $\vec{x}^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ as our initial vector.

Newton's Method Iterative Step

$$\begin{aligned} J(\underline{x}^{(k)})\Delta\underline{x} &= -\underline{f}(\underline{x}^{(k)}) \\ \underline{x}^{(k+1)} &= \underline{x}^{(k)} + \Delta\underline{x}^{(k)} \end{aligned}$$

Successive Substitution Iterative Step

$$\begin{aligned} A(\underline{x}^{(k)})\Delta\underline{x} &= -\underline{f}(\underline{x}^{(k)}) \\ \underline{x}^{(k+1)} &= \underline{x}^{(k)} + \Delta\underline{x}^{(k)} \end{aligned}$$

which can also be represented as solving $A(\underline{x}^{(k)})\underline{x}^{(k+1)} = b(\underline{x}^{(k)})$ for $\underline{x}^{(k+1)}$. Both of these expressions should be equivalent to using $\underline{x}^{(k+1)} = \underline{G}(\underline{x}^{(k)})$ from above.

Seidel Iterative Step

$$\begin{aligned} x_1^{(k+1)} &= G_1(x^{(k)}) \\ x_2^{(k+1)} &= G_2(x_1^{(k+1)}, x_2^{(k)}, \ldots, x_n^{(k)}) \\ &\vdots \\ x_n^{(k+1)} &= G_N(x_1^{(k+1)}, x_2^{(k+1)}, x_2^{(k+1)}, \ldots, x_{n-1}^{(k+1)}, x_n^{(k)}) \end{aligned}$$

QUESTION

Do you see how the methods (Newton's, Successive Substitution, and Seidel Iteration) are similar and different? **List the differences and similarities below.**

Implementation

We will use MATLAB programs `demossub` and `demonewtonsys` and `linepara` in `S:\Math Courses\Math 370\Spring2009\nmm\rootfind` to confirm your Group's calculations by hand.

What's the difference between running `demossub` and `demonewtonsys` and running `seidel('linepara',[0,0])` and `succsub('linepara',[0,0])`?