

PHYS 410/555 Computational Physics: Solution of Non Linear Equations
(a.k.a. Root Finding) (Reference *Numerical Recipes*, 9.0, 9.1, 9.4)

- We will consider two cases

1. $f(x) = 0$ “1-dimensional”
2. $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ “ d -dimensional”
 $\mathbf{x} \equiv [x_1, x_2, \dots, x_d]$
 $\mathbf{f} \equiv [f_1(x_1, x_2, \dots, x_d), \dots, f_d(x_1, x_2, \dots, x_d)]$

1. Solving Nonlinear Equations in One Variable

- We have briefly discussed bisection (binary search), will consider one other technique: *Newton’s method* (Newton-Raphson method).

Preliminaries

- We want to find one or more roots of

$$f(x) = 0 \tag{1}$$

We first note that *any* nonlinear equation in one unknown can be cast in this canonical form.

- *Definition:* Given a canonical equation, $f(x) = 0$, the *residual* of the equation for a given x -value is simply the function f evaluated at that value.
- *Iterative technique:* Assume $f(x) = 0$ has a root at $x = x^*$; then consider sequence of estimates (iterates) of x^* , $x^{(n)}$

$$x^{(0)} \rightarrow x^{(1)} \rightarrow x^{(2)} \rightarrow \dots \rightarrow x^{(n)} \rightarrow x^{(n+1)} \rightarrow \dots \rightarrow x^*$$

- Associated with the $x^{(n)}$ are the corresponding residuals

$$\begin{array}{cccccccc} r^{(0)} & \rightarrow & r^{(1)} & \rightarrow & r^{(2)} & \rightarrow & \dots & \rightarrow & r^{(n)} & \rightarrow & r^{(n+1)} & \rightarrow & \dots & \rightarrow & 0 \\ \parallel & & \parallel & & \parallel & & & & \parallel & & \parallel & & & & \parallel \\ f(x^{(0)}) & & f(x^{(1)}) & & f(x^{(2)}) & & & & f(x^{(n)}) & & f(x^{(n+1)}) & & & & f(x^*) \end{array}$$

Locating a root \equiv Driving the residual to 0

- *Convergence:* When we use an iterative technique, we have to decide *when* to stop the iteration. For root finding case, it is natural to stop when

$$|\delta x^{(n)}| \equiv |x^{(n+1)} - x^{(n)}| \leq \epsilon \quad (2)$$

where ϵ is a prescribed convergence criterion.

- A better idea is to use a “relativized” δx

$$\frac{|\delta x^{(n)}|}{|x^{(n+1)}|} \leq \epsilon \quad (3)$$

but we should “switch over” to “absolute” form (2) if $|x^{(n+1)}|$ becomes “too small” (examples in on-line code).

- *Motivation:* Small numbers often arise from “unstable processes” (numerically sensitive), e.g. $f(x+h) - f(x)$ as $h \rightarrow 0$, or from “zero crossings” in periodic solutions etc.—in such cases may not be possible and/or sensible to achieve stringent relative convergence criterion

Newton’s method

- Requires “good” initial guess, $x^{(0)}$; “good” depends on specific nonlinear equation being solved
- Refer to Numerical Recipes for more discussion; we will assume that we have a good $x^{(0)}$, and will discuss one general technique for determining good initial estimate later.
- First consider a rather circuitous way of solving the “trivial” equation

$$ax = b \quad \longrightarrow \quad f(x) = ax - b = 0 \quad (4)$$

Clearly, $f(x) = 0$ has the root

$$x^* = \frac{b}{a} \quad (5)$$

- Consider, instead, starting with some initial guess, $x^{(0)}$, with residual

$$r^{(0)} \equiv f(x^{(0)}) \equiv ax^{(0)} - b \quad (6)$$

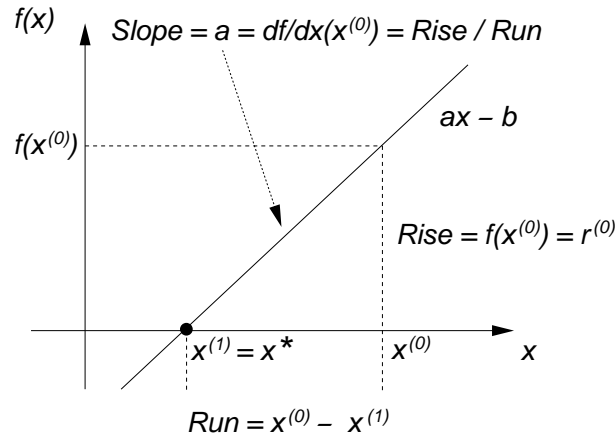
Then we can compute an improved estimate, $x^{(1)}$, which is actually the solution, x^* , via

$$x^{(1)} = x^{(0)} - \delta x^{(0)} = x^{(0)} - \frac{r^{(0)}}{f'(x^{(0)})} = x^{(0)} - \frac{f(x^{(0)})}{f'(x^{(0)})} \quad (7)$$

“Proof”:

$$x^{(1)} = x^{(0)} - \frac{r^{(0)}}{a} = x^{(0)} - \frac{ax^{(0)} - b}{a} = \frac{b}{a} \quad (8)$$

- Graphically, we have



- *Summary*

$$x^{(1)} = x^{(0)} - \delta x^{(0)} \quad (9)$$

where $\delta x^{(0)}$ satisfies

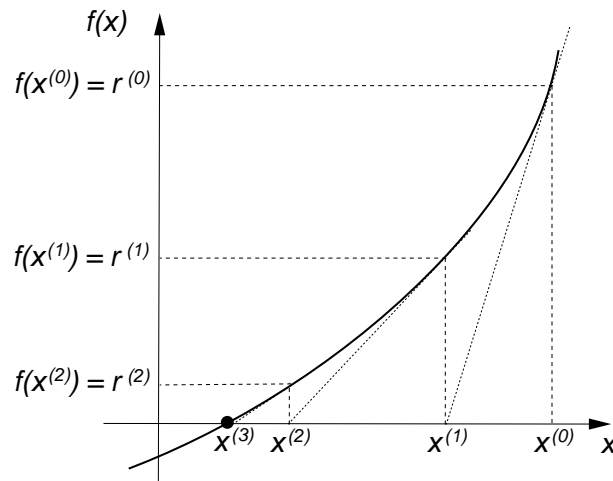
$$f'(x^{(0)})\delta x^{(0)} = f(x^{(0)}) \quad (10)$$

or

$$f'(x^{(0)})\delta x^{(0)} = r^{(0)} \quad (11)$$

- Equations (9-10) immediately generalize to non-linear $f(x)$ and, in fact, are precisely Newton's method.

- For a general nonlinear function, $f(x)$, we have, graphically



- *Newton's method for $f(x) = 0$* : Starting from some initial guess $x^{(0)}$, generate iterates $x^{(n+1)}$ via

$$x^{(n+1)} = x^{(n)} - \delta x^{(n)} \quad (12)$$

$$f'(x^{(n)})\delta x^{(n)} = r^{(n)} \equiv f(x^{(n)}) \quad (13)$$

or more compactly

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})} \quad (14)$$

- *Convergence*: When Newton's method converges, it does so *rapidly*; expect number of significant digits (accurate digits) in $x^{(n)}$ to roughly *double* at each iteration (quadratic convergence)

- *Example: “Square Roots”*

$$f(x) = x^2 - a = 0 \quad \longrightarrow \quad x^* = \sqrt{a} \quad (15)$$

Application of (14) yields

$$\begin{aligned} x^{(n+1)} &= x^{(n)} - \frac{x^{(n)2} - a}{2x^{(n)}} \\ &= \frac{2x^{(n)2} - (x^{(n)2} - a)}{2x^{(n)}} \\ &= \frac{x^{(n)2} + a}{2x^{(n)}} \end{aligned}$$

which we can write as

$$x^{(n+1)} = \frac{1}{2} \left(x^{(n)} + \frac{a}{x^{(n)}} \right) \quad (16)$$

- Try it manually, compute $\sqrt{2} = 1.414\ 2135\ 6237$ using 12-digit arithmetic (hand calculator)

Iterate	Sig. Figs
$x^{(0)} = 1.5$	1
$x^{(1)} = \frac{1}{2} (1.5 + 2.0/1.5) = 1.41\mathbf{6}\ 6666\ 6667$	3
$x^{(2)} = \frac{1}{2} (1.416\cdots + 2.0/1.416\cdots) = 1.414\ 21\mathbf{5}6\ 8628$	6
$x^{(3)} = \frac{1}{2} (1.4142\cdots + 2.0/1.4142\cdots) = 1.414\ 2135\ 62\mathbf{3}8$	11

Note the quadratic convergence of the method, as advertised.

Alternate Derivation of Newton’s Method (Taylor series)

- Again, let x^* be a root of $f(x) = 0$, then

$$0 = f(x^*) = f(x^{(n)}) + (x^* - x^{(n)})f'(x^{(n)}) + O((x^* - x^{(n)})^2) \quad (17)$$

Neglecting the higher order terms, we have

$$0 \approx f(x^{(n)}) + (x^* - x^{(n)})f'(x^{(n)}) \quad (18)$$

Now, treating the last expression as an equation, and replacing $x^{(n)}$ with the new iterate, $x^{(n+1)}$, we obtain

$$0 = f(x^{(n)}) + (x^{(n+1)} - x^{(n)})f'(x^{(n)}) \quad (19)$$

or

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})} \quad (20)$$

as previously.

2. Newton's Method for Systems of Equations

- We now want to solve

$$\mathbf{f}(\mathbf{x}) = \mathbf{0} \quad (21)$$

where

$$\mathbf{x} = (x_1, x_2, \dots, x_d) \quad (22)$$

$$\mathbf{f} = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_d(\mathbf{x})) \quad (23)$$

- *Example* ($d = 2$):

$$\sin(xy) = \frac{1}{2} \quad (24)$$

$$y^2 = 6x + 2 \quad (25)$$

In terms of our canonical notation, we have

$$\mathbf{x} \equiv (x, y) \quad (26)$$

$$\mathbf{f} \equiv (f_1(\mathbf{x}), f_2(\mathbf{x})) \quad (27)$$

$$f_1(\mathbf{x}) = f_1(x, y) = \sin(xy) - \frac{1}{2} \quad (28)$$

$$f_2(\mathbf{x}) = f_2(x, y) = y^2 - 6x - 2 \quad (29)$$

- The method is again iterative, we start with some initial guess, $\mathbf{x}^{(0)}$, then generate iterates

$$\mathbf{x}^{(0)} \rightarrow \mathbf{x}^{(1)} \rightarrow \mathbf{x}^{(2)} \rightarrow \dots \rightarrow \mathbf{x}^{(n)} \rightarrow \mathbf{x}^{(n+1)} \rightarrow \dots \rightarrow \mathbf{x}^*$$

where \mathbf{x}^* is a solution of (21)

- *Note:* The task of determining a good initial estimate $\mathbf{x}^{(0)}$ in the d -dimensional case is even more complicated than it is for the case of a single equation—again we will *assume* that $\mathbf{x}^{(0)}$ is a good initial guess, and that $\mathbf{f}(\mathbf{x})$ is sufficiently well-behaved that Newton’s method will provide a solution (i.e. *will* converge).
- As we did with the scalar (1-d) case, with any estimate, $\mathbf{x}^{(n)}$, we associate the *residual vector*, $\mathbf{r}^{(n)}$, defined by

$$\mathbf{r}^{(n)} \equiv \mathbf{f}(\mathbf{x}^{(n)}) \quad (30)$$

- The analogue of $f'(x)$ in this case is the *Jacobian matrix*, \mathbf{J} , of first derivatives. Specifically, \mathbf{J} has elements J_{ij} given by

$$J_{ij} = \frac{\partial f_i}{\partial x_j} \quad (31)$$

- For our current example we have

$$\begin{aligned} f_1(x, y) &= \sin(xy) - \frac{1}{2} \\ f_2(x, y) &= y^2 - 6x - 2 \end{aligned}$$

$$\mathbf{J} = \begin{bmatrix} \partial f_1/\partial x & \partial f_1/\partial y \\ \partial f_2/\partial x & \partial f_2/\partial y \end{bmatrix} = \begin{bmatrix} y \cos(xy) & x \cos(xy) \\ -6 & 2y \end{bmatrix}$$

- We can now derive the multi-dimensional Newton iteration, by considering a multivariate Taylor series expansion, paralleling what we did in the 1-d case:

$$\mathbf{0} = \mathbf{f}(\mathbf{x}^*) = \mathbf{f}(\mathbf{x}^{(n)}) + \mathbf{J}[\mathbf{x}^{(n)}] \cdot (\mathbf{x}^* - \mathbf{x}^{(n)}) + O((\mathbf{x}^* - \mathbf{x}^{(n)})^2) \quad (32)$$

where the notation $\mathbf{J}[\mathbf{x}^{(n)}]$ means we evaluate the Jacobian matrix at $\mathbf{x} = \mathbf{x}^{(n)}$.

Dropping higher order terms, and replacing \mathbf{x}^* with $\mathbf{x}^{(n+1)}$, we have

$$\mathbf{0} = \mathbf{f}(\mathbf{x}^{(n)}) + \mathbf{J}[\mathbf{x}^{(n)}](\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}) \quad (33)$$

Defining $\delta\mathbf{x}^{(n)}$ via

$$\delta \mathbf{x}^{(n)} \equiv -(\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}) \quad (34)$$

the d -dimensional Newton iteration is given by

$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \delta \mathbf{x}^{(n)} \quad (35)$$

where the update vector, $\delta \mathbf{x}^{(n)}$, satisfies the $d \times d$ *linear system*

$$\mathbf{J}[\mathbf{x}^{(n)}] \delta \mathbf{x}^{(n)} = \mathbf{f}(\mathbf{x}^{(n)}) \quad (36)$$

- Again note that the Jacobian matrix, $\mathbf{J}[\mathbf{x}^{(n)}]$, has elements

$$J_{ij}[\mathbf{x}^{(n)}] = \left. \frac{\partial f_i}{\partial x_j} \right|_{\mathbf{x}=\mathbf{x}^{(n)}} \quad (37)$$

- At each step of the Newton iteration, the linear system (36) can, of course, be solved using an appropriate linear solver (e.g. general, tridiagonal, or banded).

General Structure of a Multidimensional Newton Solver

x: Solution vector
res: Residual vector
J: Jacobian matrix
dx: Update vector

```

x = x(0)
do while ||dx||2 > ε
  do i = 1 , neq
    res(i) = fi(x)
    do j = 1 , neq
      J(i,j) = [∂fi/∂xj](x)
    end do
  end do
  dx = solve(J dx = res)
  x = x - dx
end do

```


Finite Difference Example: Non-Linear BVP

- Consider the nonlinear two-point boundary value problem

$$u(x)_{xx} + (uu_x)^2 + \sin(u) = F(x) \quad (38)$$

which is to be solved on the interval

$$0 \leq x \leq 1 \quad (39)$$

with the boundary conditions

$$u(0) = u(1) = 0 \quad (40)$$

- As we did for the case of the linear BVP, we will approximately solve this equation using $O(h^2)$ finite difference techniques. As usual we introduce a uniform finite difference mesh:

$$x_j \equiv (j-1)h \quad j = 1, 2, \dots, N \quad h \equiv (N-1)^{-1} \quad (41)$$

- Then, using the standard $O(h^2)$ approximations to the first and second derivatives

$$u_x(x_j) = \frac{u_{j+1} - u_{j-1}}{2h} + O(h^2) \quad (42)$$

$$u_{xx}(x_j) = \frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} + O(h^2) \quad (43)$$

the discretized version of (38-40) is

$$\frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} + (u_j)^2 \left[\frac{u_{j+1} - u_{j-1}}{2h} \right]^2 + \sin(u_j) - F_j = 0; \quad j = 2 \dots N-1 \quad (44)$$

$$u_1 = 0 \quad (45)$$

$$u_N = 0 \quad (46)$$

Note that we have cast the discrete equations in the canonical form $\mathbf{f}(\mathbf{u}) = \mathbf{0}$

- In order to apply Newton's method to the algebraic equations (45-46), we must compute the Jacobian matrix elements of the system.
- We first observe that due to the "nearest-neighbor" couplings of the unknowns u_j via the approximations (42-43), the Jacobian matrix is *tridiagonal* in this case.

- For the *interior* grid points, $j = 2 \dots N$, corresponding to rows $2 \dots N$ of the matrix, we have the following non-zero Jacobian elements:

$$J_{j,j} = -\frac{2}{h^2} + 2u_j \left[\frac{u_{j+1} - u_{j-1}}{2h} \right]^2 + \cos(u_j) \quad (47)$$

$$J_{j,j-1} = \frac{1}{h^2} - (u_j)^2 \frac{u_{j+1} - u_{j-1}}{2h^2} \quad (48)$$

$$J_{j,j+1} = \frac{1}{h^2} + (u_j)^2 \frac{u_{j+1} - u_{j-1}}{2h^2} \quad (49)$$

- For the *boundary* points, $j = 1$ and $j = N$, corresponding to the first and last row, respectively, of \mathbf{J} , we have

$$J_{1,1} = 1 \quad (50)$$

$$J_{1,2} = 0 \quad (51)$$

and

$$J_{N,N} = 1 \quad (52)$$

$$J_{N,N-1} = 0 \quad (53)$$

- Note that these last expressions correspond to the “trivial” equations

$$f_1 = u_1 = 0 \quad (54)$$

$$f_N = u_N = 0 \quad (55)$$

which have associated residuals

$$r_1^{(n)} = u_1^{(n)} \quad (56)$$

$$r_N^{(n)} = u_N^{(n)} \quad (57)$$

- Observe that if we initialize $u_1^{(0)} = 0$ and $u_N^{(0)} = 0$, then we will *automatically* have $\delta u_1^{(n)} = \delta u_N^{(n)} = 0$, which will yield $u_1^{(n)} = 0$ and $u_N^{(n)} = 0$ as desired.
- This is an example of the general procedure we have seen previously for imposing Dirichlet conditions; namely the conditions are implemented as “trivial” (linear) equations (but it is, of course, absolutely crucial to implement them *properly* in this fashion!)
- *Testing procedure:* We adopt the same technique used for the linear BVP case—we *specify* $u(x)$, then compute the function $F(x)$ that is required to satisfy (38); $F(x)$ is then supplied as input to the code, and we ensure that as $h \rightarrow 0$ we observe second order convergence of the computed finite difference solution $\hat{u}(x)$ to the continuum solution $u(x)$.

- *Example:* Taking

$$u(x) = \sin(4\pi x) \equiv \sin(\omega x) \quad (58)$$

then

$$\begin{aligned} F(x) &= u_{xx} + (uu_x)^2 + \sin(u) \\ &= -\omega^2 \sin(\omega x) + \omega^2 \sin^2(\omega x) \cos^2(\omega x) + \sin(\sin(\omega x)) \end{aligned} \quad (59)$$

- We note that due to the nonlinearity of the system, we will actually find *multiple* solutions, depending on how we initialize the Newton iteration; this is illustrated with the on-line code `nlbvp1d`.

3. Determining Good Initial Guesses: Continuation

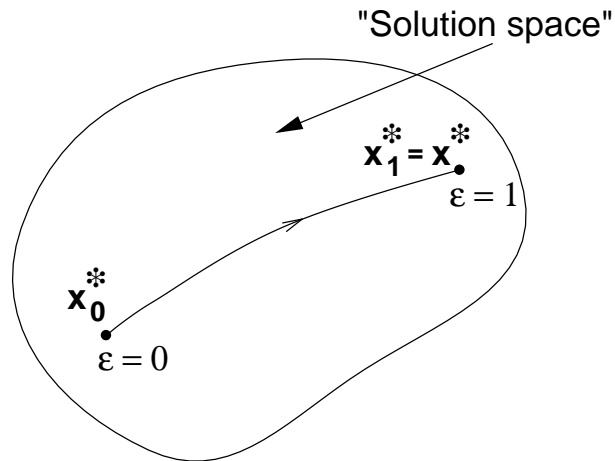
- It is often the case that we will want to solve nonlinear equations of the form

$$\mathbf{N}(\mathbf{x}; \bar{\mathbf{p}}) = 0 \quad (60)$$

where we have adopted the notation $\mathbf{N}(\dots)$ to emphasize that we *are* dealing with a nonlinear system. Here $\mathbf{x} = (x_1, x_2 \dots x_d)$ is, as previously, a vector of unknowns, with $\mathbf{x} = \mathbf{x}^*$ a solution of (60).

- The quantity $\bar{\mathbf{p}}$ in (60) is another vector, of length m , which enumerates any additional parameters (generally adjustable) that enter into the problem; these could include: coupling constants, rate constants, “perturbation” amplitudes etc.
- The nonlinearity of any particular system of the form (60) may make it *very* difficult to compute \mathbf{x}^* without a good initial estimate $\mathbf{x}^{(0)}$; in such cases, the technique of *continuation* often provides the means to generate such an estimate.
- *Continuation:* The basic idea underlying continuation is to “sneak up” on the solution by introducing an additional parameter, ϵ (the continuation parameter), so that by *continuously* varying ϵ from 0 to 1 (by convention), we vary from:
 1. A problem that we know how to solve, or for which we already have a solution.
 to
 2. The problem of interest.

- Schematically we can sketch the following picture:



- Note that we thus consider a *family* of problems

$$\mathbf{N}_\epsilon(\mathbf{x}; \bar{\mathbf{p}}) = 0 \quad (61)$$

with corresponding solutions

$$\mathbf{x}_\epsilon = \mathbf{x}_\epsilon^* \quad (62)$$

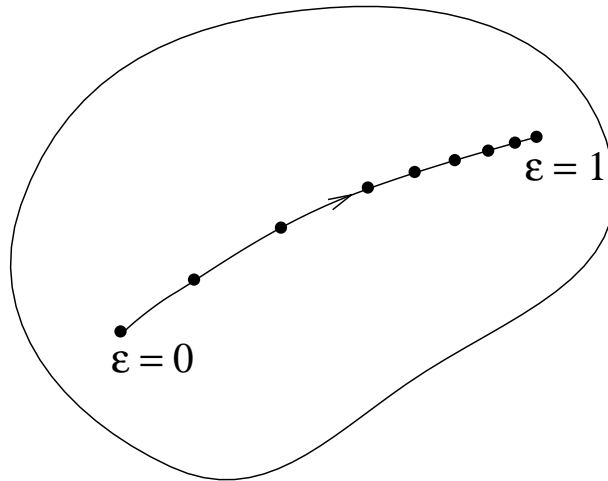
- The efficacy of continuation generally depends on two crucial points:
 1. $\mathbf{N}_0(\mathbf{x}; \bar{\mathbf{p}})$ has a known or easily calculable root at \mathbf{x}_0^* .
 2. Can often choose $\Delta\epsilon$ judiciously (i.e. sufficiently small) so that

$$\mathbf{x}_{\epsilon-\Delta\epsilon}^*$$

is a “good enough” initial estimate for

$$\mathbf{N}_\epsilon(\mathbf{x}; \bar{\mathbf{p}}) = 0$$

- Again, schematically, we have



where we note that we may have to adjust (adapt) $\Delta\epsilon$ as the continuation proceeds.

Continuation: Summary and Comments

- Solve sequence of problems with $\epsilon = 0, \epsilon_2, \epsilon_3 \dots 1$ using previous solution as initial estimate for each $\epsilon \neq 0$.
- Will generally have to tailor idea on a case-by-case basis.
- Can often identify ϵ with one of the p_i (intrinsic problem parameters) *per se*.
- The first problem, $\mathbf{N}_0(\mathbf{x}, \bar{\mathbf{p}}) = 0$, can frequently be chosen to be *linear*, and therefore “easy” to solve, modulo sensitivity/poor conditioning.
- For time-dependent problems, *time evolution* often provides “natural” continuation; $\epsilon \rightarrow t$, and we can use $\mathbf{x}^*(t - \Delta t)$ as the initial estimate $\mathbf{x}^{(0)}(t)$.