

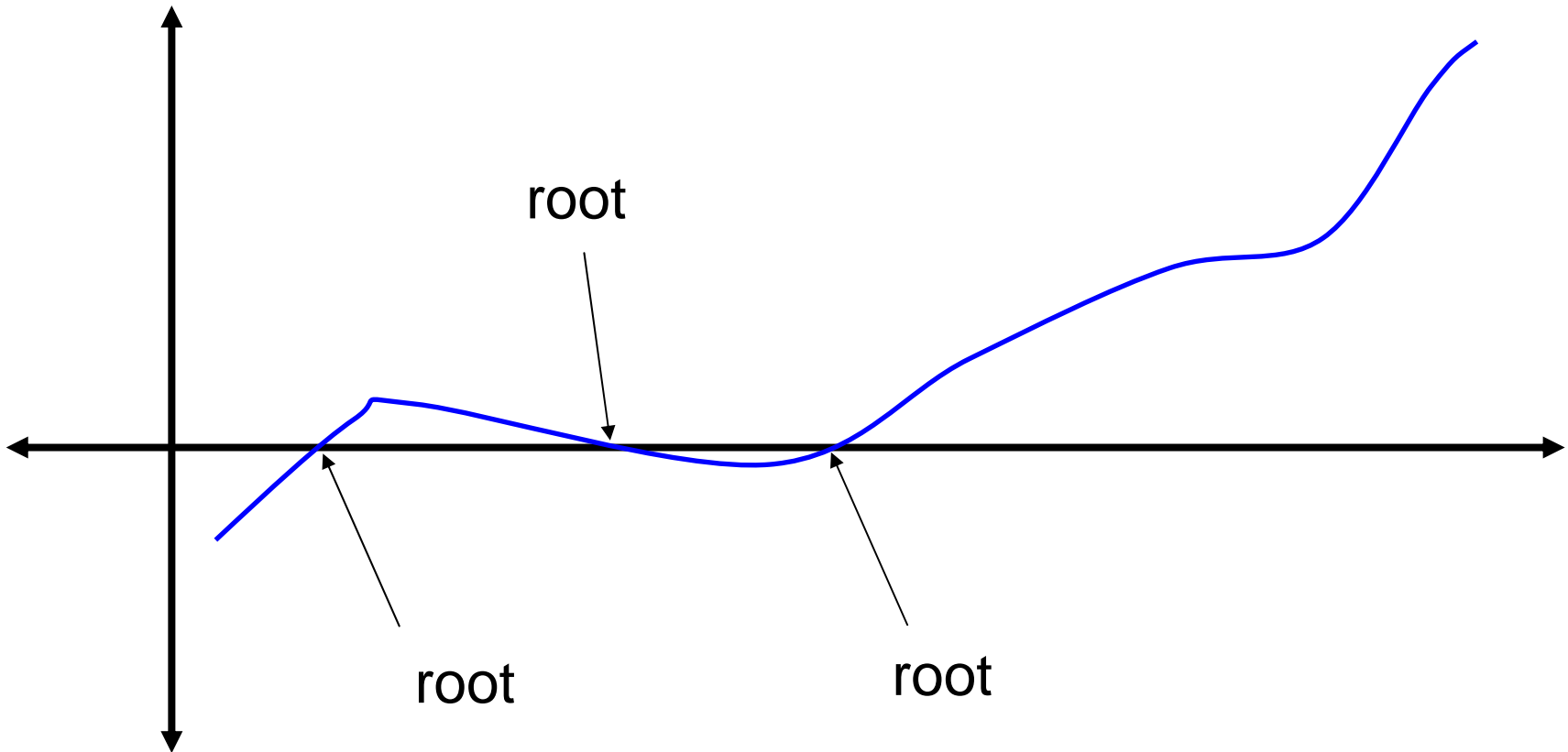
Goals for This Lecture:

- Learn the Newton-Raphson method for finding real roots of real functions
- Learn the Bisection method for finding real roots of a real function
- Look at efficient implementations of both methods in F95

Roots of a function

- We define a root of a function, F , to be a value x such that $F(x)=0$
- In general we can only find roots analytically for the simplest functions
- Example: quadratic, cubic, and quartic polynomials
- Otherwise roots must be found numerically

First Rule of Root Finding: Plot the Function whenever possible



The Newton-Raphson Method (Part I)

- Let x_r be a root of $F(x)$
- Expand $F(x)$ in a Taylor series for x near x_r

$$F(x_r) \approx F(x) + F'(x)(x_r - x)$$

- But recall that $F(x_r) = 0$ by definition
- Therefore we have

$$x_r \approx x - \frac{F(x)}{F'(x)}$$

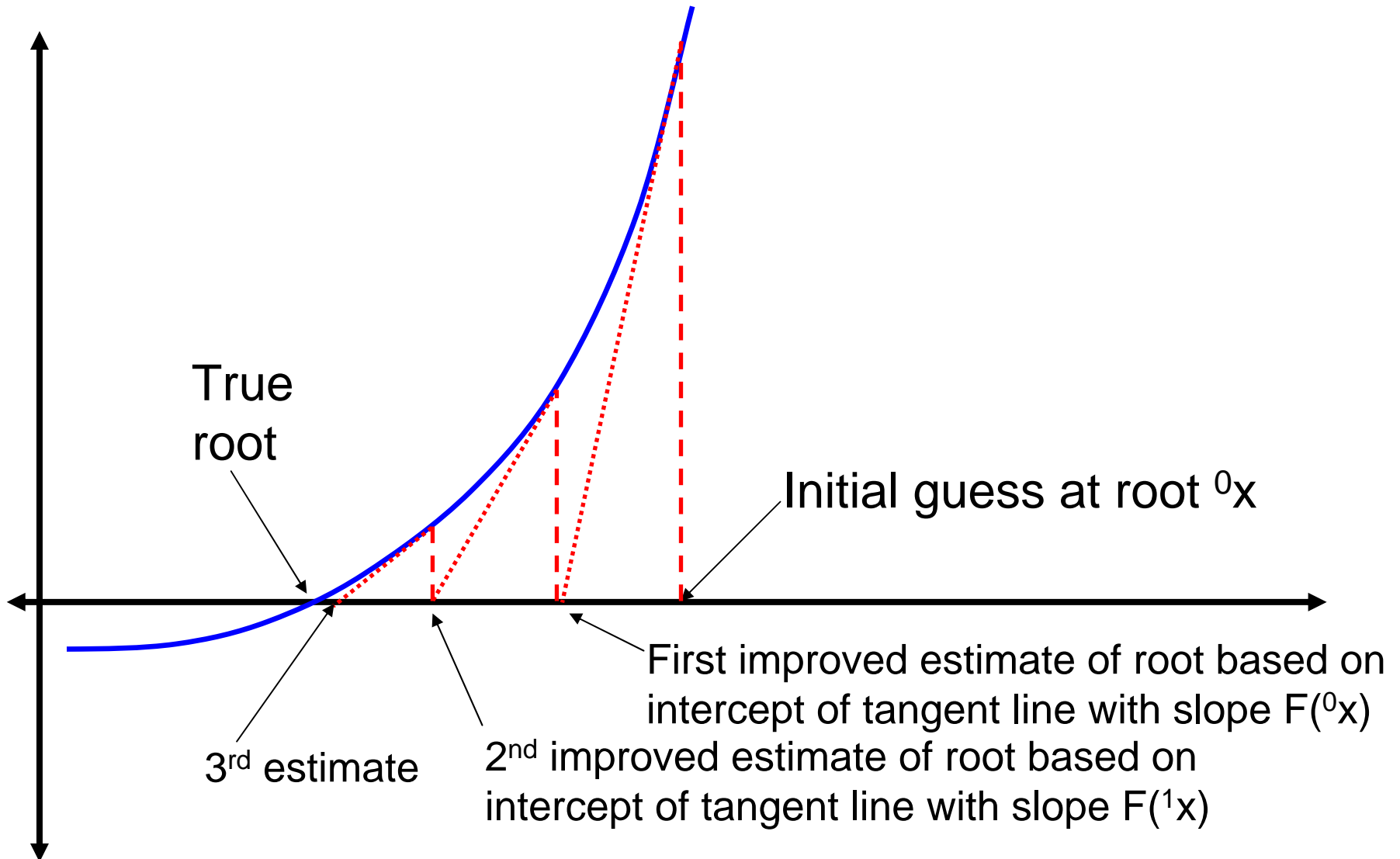
The Newton-Raphson Method (Part II)

- Exploit the structure of the Taylor series expansion to improve a guess at the value of the root
- Make an initial guess at a value for the root: x^0
- Use the Taylor series expansion to find a better approximation to the root:

$$x^1 = x^0 - \frac{F(x^0)}{F'(x^0)}$$

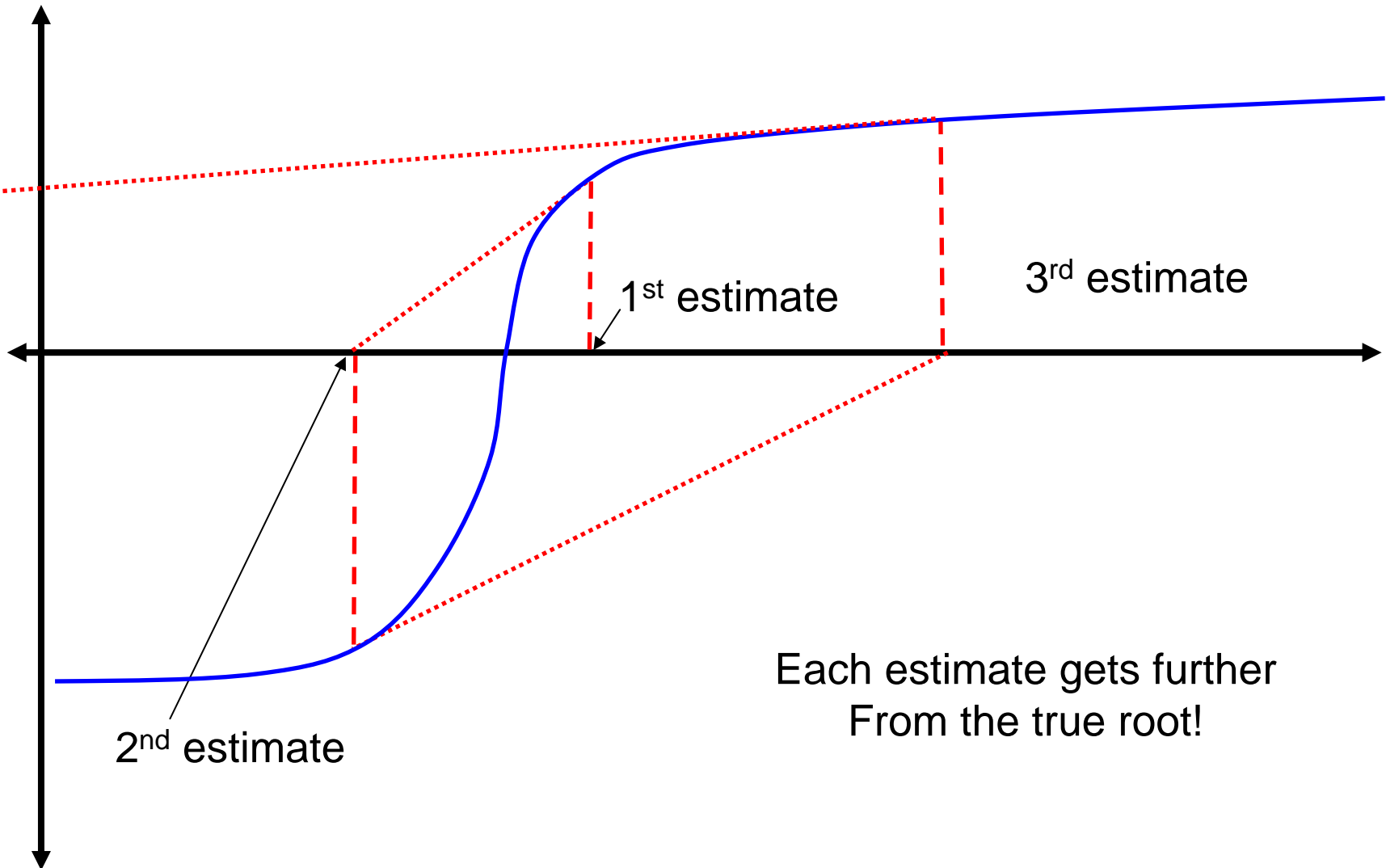
- Use x^1 as an improved guess at the root and employ the Taylor series expansion again to get a better guess x^2
- Repeat process until the answer is accurate enough

A Geometrical Interpretation of Newton-Raphson Iteration



Newton-Raphson Iteration is not foolproof!

Example of a simple function that will defeat Newton-Raphson Iteration:



Stopping Criterion

- When do we stop iterating?
- Could stop when $F(x) < \text{some tolerance}$
- This method may fail if you don't understand the behavior of $F(x)$ near the root
- Could check to see if relative change between “old” and “new” estimates is small
- Choice depends on the problem
- Visually verify root by inspecting plot of function if possible

Newton-Raphson Program

```
! Purpose: Find the root of (x^2)-8x+16 via Newton-Raphson Iteration
! Author: F. Douglas Swesty
! Date: 10/7/2005
program newton_raphson
implicit none ! Turn off implicit typing
real :: xold ! Old estimate of root
real :: xnew ! New estimate of root
real :: fold ! Value of F(xold)
real :: fpold ! Value of dF/dx(xold)
real :: xchange ! Relative change in x
integer :: niter=1 ! Number of iterations
integer, parameter :: niter_max = 30 ! Maximum # of iterations allowed

xold = 3.2 ! Initial guess at root

do while (niter < niter_max) ! Initiate loop
  fold = (xold**2)-8.0*xold+16.0 ! Evaluate function at old estimate
  fpold = 2.0*xold-8.0 ! Evaluate derivative at old estimate
  if(abs(fpold) < 1.0e-20) then ! If derivative is zero the method fails
    write(*,*) ' fpold is approx. zero'
    write(*,*) ' method failed'
    stop
  endif
  xnew = xold - fold/fpold ! Calculate new estimate
  xchange = abs((xnew-xold)/xold) ! Calculate relative change
  if(xchange < 1.0e-5) then
    write(*,*) ' root is ',xnew
    stop
  else
    xold = xnew ! Update the "old" guess to the "new" guess
    niter = niter+1 ! Update iteration counter
  endif
enddo ! Terminate loop
write(*,*) ' Method failed!'

stop ! Stop execution of the program
end program newton_raphson
```

What can we do when Newton-Raphson fails?

- Recast N-R iteration as:

$$x_r \approx x - \delta x$$

- Try restricting the step size δx

$$\delta x = \frac{F(x)}{F'(x)}$$

$$x_r \approx x - C\delta x$$

where $0 < C < 1$

- Reduce C until $|F^{(n+1)}x| < |F^{(n)}x|$
- These methods may slow convergence
- These methods may not work!

An Alternative Method: bisection

- Assume that $F(x)$ is continuous
- Suppose we have two values x_L and $x_H > x_L$ such that $F(x_L)$ and $F(x_H)$ have opposite signs
- The Intermediate Value Theorem tells us that there must a point x_r , where $x_L < x_r < x_H$, such that $F(x_r) = 0$
 - Root is contained in the interval (x_L, x_H)
 - We say the root is bracketed by x_L and x_H
- Bisection Algorithm:
 1. Choose x_L and x_H such that root is bracketed
 2. Choose the midpoint $x_M = (x_L + x_H)/2$
 3. If $F(x_M) = 0$ then x_M is the root
 4. If $F(x_M)$ and $F(x_L)$ have opposite signs then the Intermediate Value Theorem tells us that $x_L < x_r < x_M$
 5. Otherwise $x_M < x_r < x_H$
 6. Repeat process (starting with step 1) on subinterval containing the root until answer is accurate enough
- Algorithm easily implemented in a program with loops and conditional constructs

A comparison of the Methods

- When Newton-Raphson works it is usually much faster
- Error usually decreases quadratically
- However, Newton-Raphson iteration can fail for a variety of reasons
- Newton-Raphson can easily be extended to multi-variable problems
- Bisection is usually slower than N-R iteration
- However, bisection is usually robust
- Bisection cannot be extended to multivariable problems