计算物理

Lecture 6

傅子文

Today's Lecture

- Functions and roots: Part II
  - Global bracket finding techniques
  - Secant root finder
  - Müller-Brent root finder
  - Example

Global Bracket finding

- Dr Clarke's Global Bracket Finding Algorithm!
  - The explanation is somewhat lengthy, don't be too concerned if you don't "get it all" first time out – however the fundamental concepts are simple

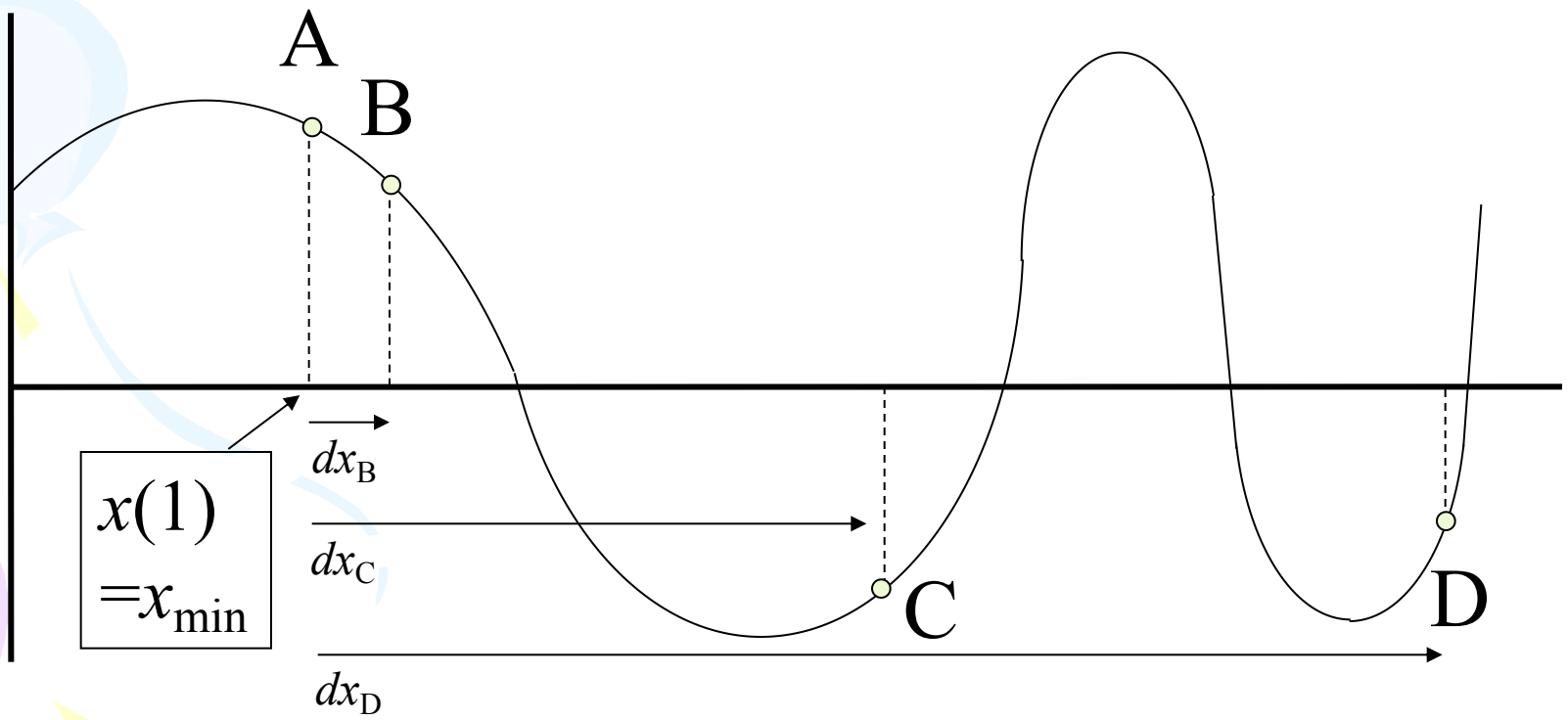
- **Broad-brush**
  - Calculate array $x(n)$
    - Values of independent variable x at which $f(x)$ will be evaluated
    - These values won't necessarily be equally spaced, in fact we might expect them not to be
  - Store all values of $f(x(n))$
    - Evaluate for all n, although this may need to be a large number
  - Apply root finder to each pair of values of $f(x(i))$
    - If $(f(x(i))*f(x(i+1) < 0$ → root between this pair
    - If no root, skip to next pair
  - Report all roots found

Issues to address

- How do we know how large to make the domain of x we consider?
  - We need limits on "x_{min}", "x_{max}"
  - Some problems have natural limits (see example later)
  - Roots may be all positive in which case only need upper bound
  - For polynomials ($\sum a_n x^n$) for large x, x^n term dominates
    - $a_{n-1}+a_n x \approx 0$ so $x_{max} \geq -a_{n-1}/a_n$
    - For smallest root, $a_0+a_1 x \approx 0$ so $x_{min} \leq -a_0/a1$
  - However, in many cases we may just have to guess!

How many points to we need?

- This is equivalent to answering “how big does the spacing between points need to be?”



Selecting the step size *dx*

- To be conservative we want to pick our first step $x(2)=x(1)+dx(1)$ to be such that there are no roots between $x(1)$ and $x(2)$
  - dx_B achieves this if A corresponds to the first point

- At each stage we should ***adapt*** $dx(i)$, so that if $x(i+1)=x(i)+dx(i)$ we want to be reasonably sure there isn't more than one root between the two points
  - dx_C achieves this but looks slightly dangerous if we used it generally – it could cover two roots on the right of the diagram
  - dx_D is just too big

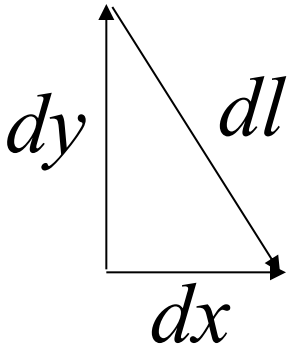
Evaluating an optimal *dx*

- Needs to be sensitive to local changes in the function
- *dx* needs to contain ≤ 1 root
- Finding an optimal *dx* is a bit like "groping around in the dark"
  - Start with small steps
  - If the function feels "straight" take bigger steps
  - If the function has rapid changes in direction then take smaller steps

Follow the function "length" to find *dx*

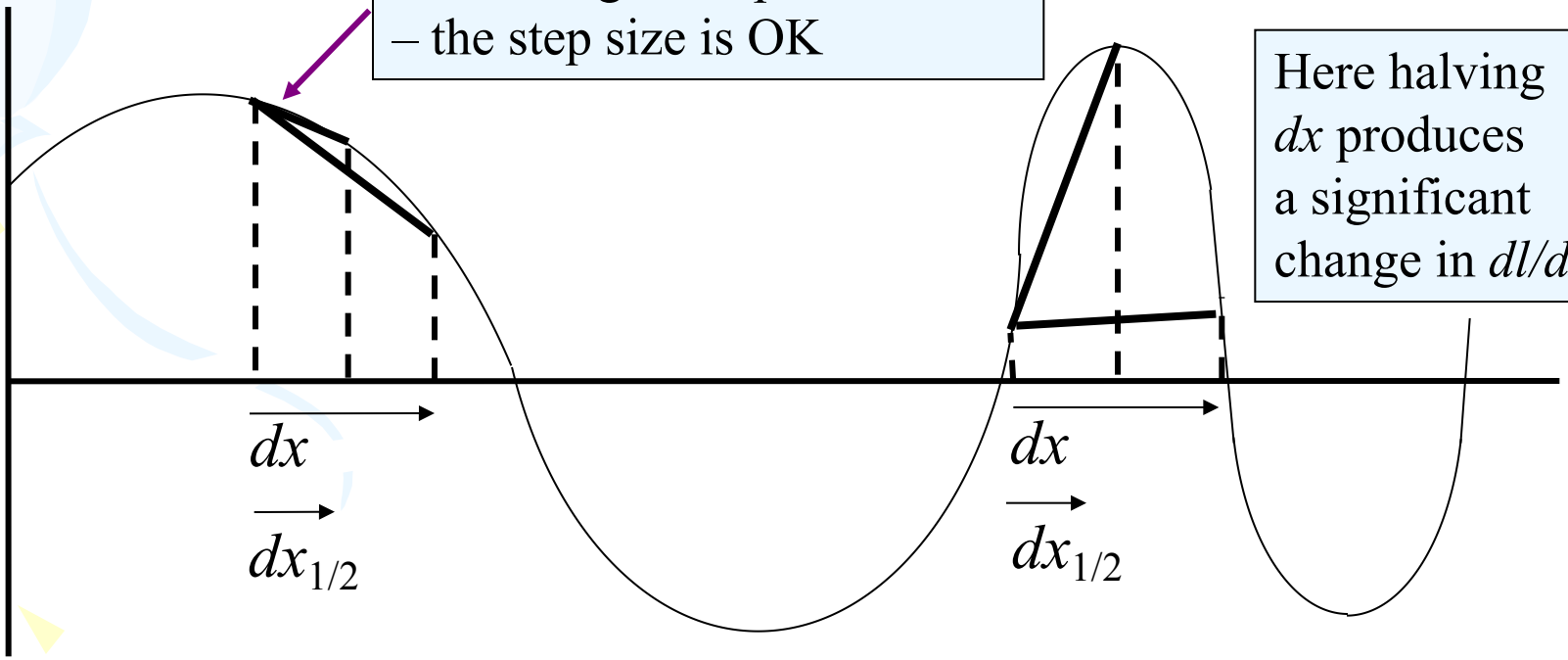
$$\text{Since } \delta l^2 = \delta x^2 + \delta y^2 = \delta x^2\left(1+\left[\frac{\delta y}{\delta x}\right]^2\right) = \delta x^2\left(1+f'(x)^2\right)$$

$$\Rightarrow \frac{\delta l}{\delta x} = \sqrt{1+f'(x)^2}$$



In this part of the function halving the size of *dx* doesn't have to big an impact on *dl/dx* – the step size is OK

Here halving *dx* produces a significant change in *dl/dx*



Quick summary of conditions

- If $dl/dx \sim dl_{1/2}/dx_{1/2}$ then dx is OK
  - If dl/dx and $dl_{1/2}/dx_{1/2}$ are very close then we may actually be able to increase the size of dx
- If dl/dx and $dl_{1/2}/dx_{1/2}$ are "very different" then dx needs to be decreased
  - We try using small dx until we find a separation that produces the first condition
- *Remember: we are using dl/dx to tell us what is happening to the value of the function locally*

Slight modification – not intuitive

- We really want to consider the change in path length as a function of the relative change in *dx* and *x*
- So rather than using *f'(x)=df/dx* we use

$$f'(x) = \frac{\delta f(x) / \tilde{f}(x)}{\delta x / \tilde{x}} = \frac{\tilde{x}}{\tilde{f}(x)} \frac{\delta f}{\delta x}$$

- This scales the derivative in terms of the *x* and *y* values
  - Although for numerical stability reasons we don't quite use *x* and *f*(*x*) (see next slide)
- The numerical value of the derivative is given by

$$\frac{\delta f}{\delta x} = \frac{f(x(i)+\delta x) - f(x(i))}{\delta x}$$

Efficiency & stability concerns

- Using $\tilde{x} = \max(1.0, x(i))$
- Reduces the number of function calls significantly near $x=0$
  - This was found by trial and error
- Using $\tilde{f}(x) = \max(|f(x(i)+\delta x)|, |f(x(i))|)$
- Doesn't have a huge advantage except making divide by zero less likely

Final part of algorithm

- We have the mechanisms to find *x(n)* and the function evaluations *f(n)*
  - Apply root finding to each pair using following algorithm

```
do i=1,n-1
  if (f(i)*f(i+1) .lt. 0.0) then
  find root with preferred method and report
 end if
end do
```

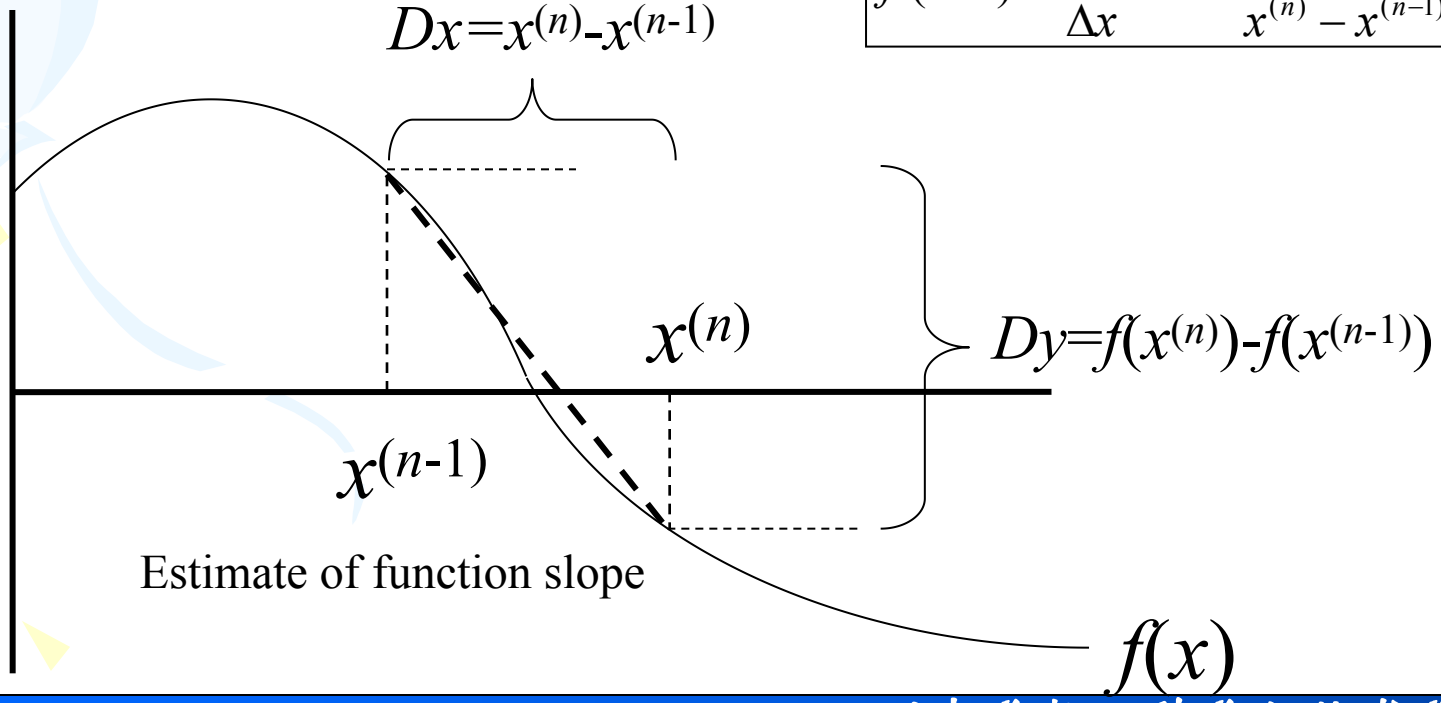
- I think I will set a homework question with a detailed explanation of the algorithm

Secant Method - Preliminaries

- One of the key issues in N-R is that you need the derivative
  - May not have it if the function is defined numerically
  - Derivative could be extremely hard to calculate

- You can get an estimate of the local slope using values of $x^{(n)}$,$x^{(n-1)}$, $f(x^{(n)})$ and $f(x^{(n-1)})$

$$f'(x^{(n)}) \approx \frac{\Delta y}{\Delta x} = \frac{f(x^{(n)}) - f(x^{(n-1)})}{x^{(n)} - x^{(n-1)}}$$



Secant Method

- So once you initially select a pair of points bracketing the root $(x^{(n-1)}, x^{(n)})$, $x^{(n+1)}$ is given by

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

- You then iterate to a specific tolerance value exactly as in NR
- This method can converge almost as quickly as NR

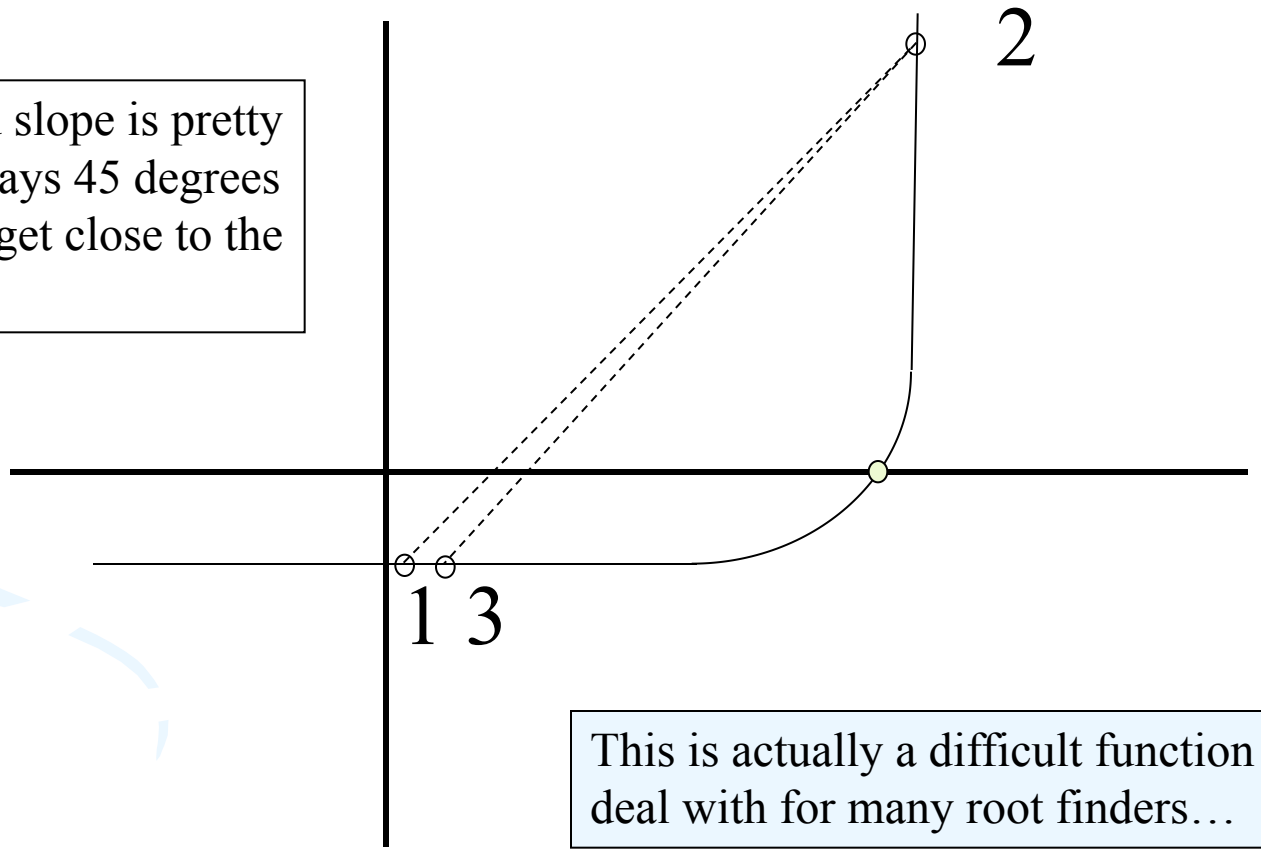
Issues with the Secant Method

- Given the initial values $x_l(=x^{(n-1)}),x_u(=x^{(n)})$ that bracket the root the calculation of $x^{(n+1)}$ is *local*
  - Easy to program, all values are easily stored

- Difficulty is with finding suitable x_l,x_u since this is a global problem
  - Determined by $f(x)$ over its entire range
  - Consequently difficult to program (no perfect method)

Case where Secant will be poor

Estimated slope is pretty much always 45 degrees until you get close to the Root.



$x^{10}-1=0$

This is actually a difficult function to deal with for many root finders…

Secant method is also fast

Advantages

- better-than-linear convergence near simple root
- linear convergence near multiple root
- no derivative needed

Disadvantages

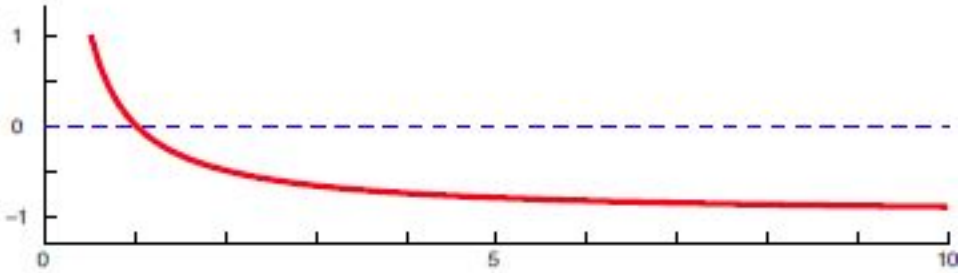
- iterates may diverge
- no practical & rigorous error bound

Bracket finding Strategies

- Graph the function
  - May require 1000's of points and thus function calls to determine visually where the function crosses the x-axis
- "Exhaustive searching" – a global bracket finder
  - Looks for changes in the sign of $f(x)$
  - Need small steps so that no roots are missed
  - Need still to take large enough steps that this doesn't take forever

Without bracketing, an iteration can jump far away

```
>> f = @(x) 1/x - 1;
>> secant(f,[0.5,10],4);

k         x_k              f(x_k)
0           0.5     1.00000e+00
1            10    -9.00000e-01
2           5.5    -8.18182e-01
3         -39.5    -1.02532e+00
4        183.25    -9.94543e-01
```



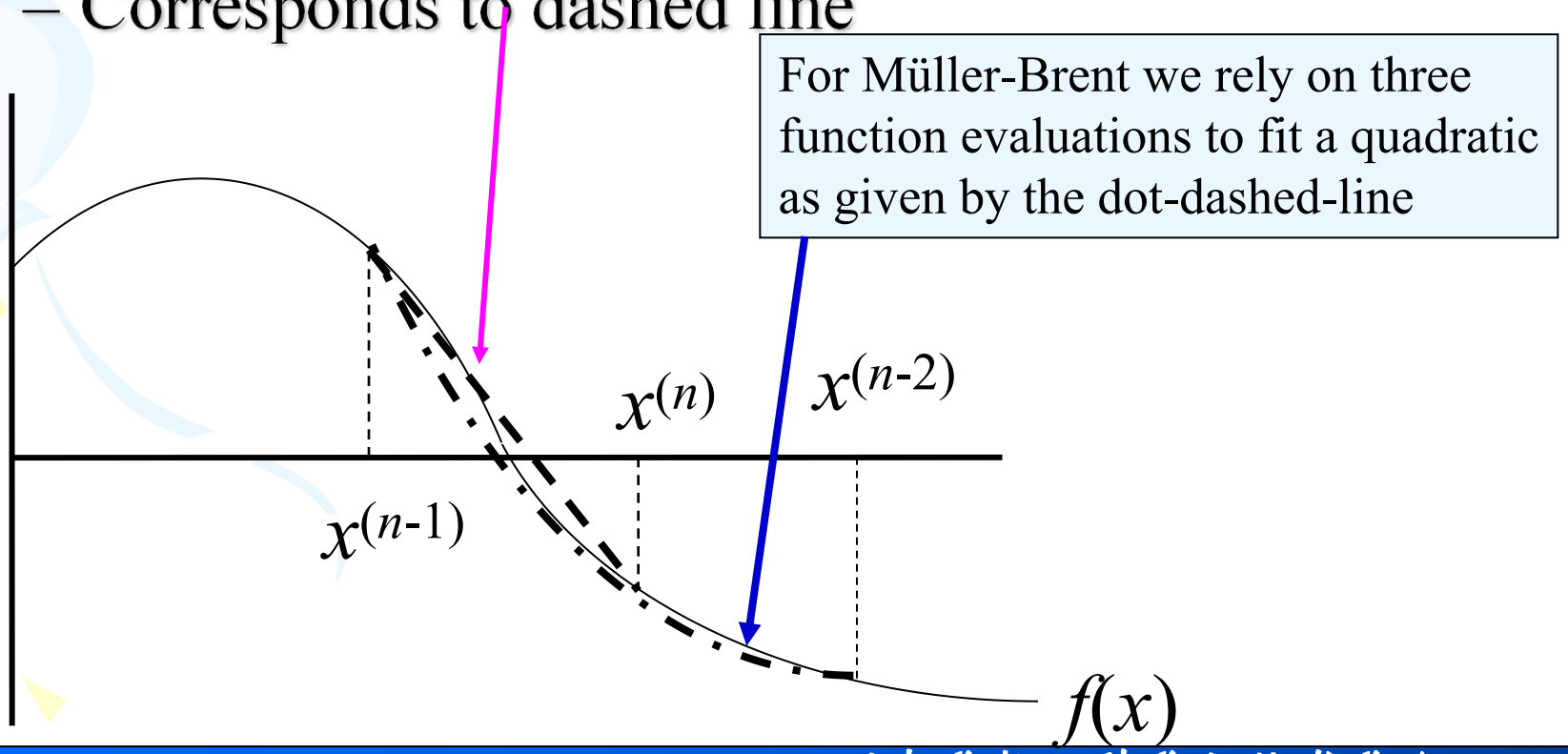

```
>> df = @(x) -1/x^2;
>> newtonraphson(f,df,10,3);

k            x_k       f(x_k)     f'(x_k)           dx
0             10    -9.00e-01    -0.01000           90
1            -80    -1.01e+00    -0.00016         6480
2          -6560    -1.00e+00    -0.00000  4.30402e+07
3  -4.30467e+07    -1.00e+00    -0.00000  1.85302e+15
```

Müller-Brent method

- Most expensive part of root finding algorithms is frequently the function evaluation
- In Secant method *n+1* guess is based upon function evaluations from *n* and *n-1* steps
  - We throw away the *n-2* step
  - With only two function evaluations the fitting method must be linear
- We could use *n-2* step to give us three points that we use for quadratic interpolation

Secant method compared to Müller-Brent

- Secant method fits a straight line between $(x^{(n)}, f(x^{(n)}))$ and $(x^{(n-1)}, f(x^{(n-1)}))$
  - Corresponds to dashed line



Fitting a quadratic

- Three points is enough data to fit a function of the form

$$q(x) = a(x - x^{(n)})^2 + b(x - x^{(n)}) + c$$

- Since we know q(x) passes through all three points we have

$$q(x^{(n)}) = f(x^{(n)}) \Rightarrow c = f(x^{(n)})$$

$$q(x^{(n-1)}) = f(x^{(n-1)}) = a(x^{(n-1)} - x^{(n)})^2 + b(x^{(n-1)} - x^{(n)}) + f(x^{(n)})$$

$$q(x^{(n-2)}) = f(x^{(n-2)}) = a(x^{(n-2)} - x^{(n)})^2 + b(x^{(n-2)} - x^{(n)}) + f(x^{(n)})$$

- Which gives two equations for two unknowns,

Solving for *a* & *b*

- Given the two equations we can eliminate *a* or *b* to get equations for the two constants

$$a = \frac{(x^{(n-1)} - x^{(n)})[f(x^{(n-2)}) - f(x^{(n)})] - (x^{(n-2)} - x^{(n)})[f(x^{(n-1)}) - f(x^{(n)})]}{(x^{(n-2)} - x^{(n-1)})(x^{(n-2)} - x^{(n)})(x^{(n-1)} - x^{(n)})}$$

$$b = \frac{(x^{(n-2)} - x^{(n)})^2[f(x^{(n-1)}) - f(x^{(n)})] - (x^{(n-1)} - x^{(n)})^2[f(x^{(n-2)}) - f(x^{(n)})]}{(x^{(n-2)} - x^{(n-1)})(x^{(n-2)} - x^{(n)})(x^{(n-1)} - x^{(n)})}$$

- We now have *a,b,c* so we can use the general formula to calculate the zeros for $x^{(n+1)}$-$x^{(n)}$

$$x^{(n+1)} - x^{(n)} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

General quadratic solution issues

- Recall we talked about the problems of differencing very similar numbers
  - Precision is lost due to the fixed number of digits in the mantissa
- We use the same method discussed in the previous lectures for improving stability
- If $b \geq 0$ then take + root

$$x^{(n+1)} = x^{(n)} + \frac{-b+\sqrt{b^2-4ac}}{2a} \times \frac{-b-\sqrt{b^2-4ac}}{-b-\sqrt{b^2-4ac}} = x^{(n)} - \frac{2c}{b+\sqrt{b^2-4ac}}$$

$b < 0$ then

$$x^{(n+1)} = x^{(n)} + \frac{-b-\sqrt{b^2-4ac}}{2a} \times \frac{-b+\sqrt{b^2-4ac}}{-b+\sqrt{b^2-4ac}} = x^{(n)} + \frac{2c}{-b+\sqrt{b^2-4ac}}$$

Full Müller-Brent

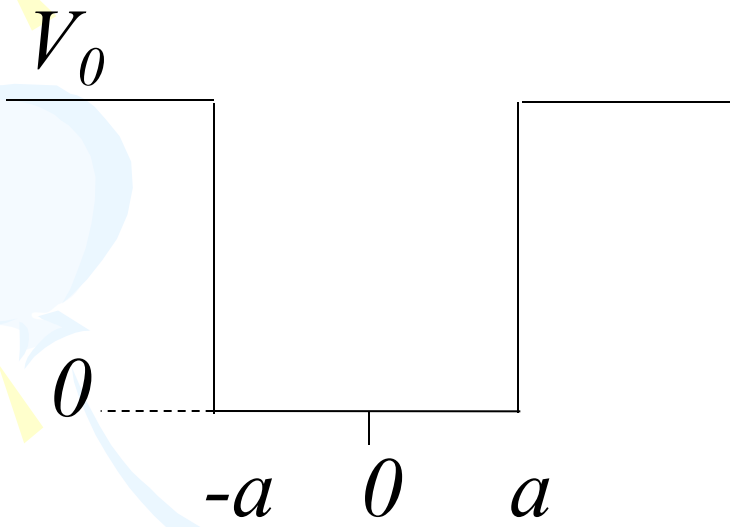
- The previous two equations for $x^{(n+1)}$ give the guesses for the root
  - Can suffer from similar problems to NR if too close to extremum though
  - (This part of algorithm is due to Muller)
- Brent added the option of a bisection step to improve behaviour
  - Same idea as the hybrid NR-bisection algorithm we discussed

Higher order fitting methods

- In practice quadratic is as high as we need to go
- Firstly, if the quadratic fit guesses a root outside the bracket then the cubic fit will not do any better
- Secondly, if it converges the quadratic fits usually takes about 3 steps
  - The cubic fit would require 4 steps just to calculate the initial points required for the guess $(x^{(n-3)},x^{(n-2)},x^{(n-1)},x^{(n)})$
- You can think of this the following way
  - *By the time we can fit a cubic a quadratic method will likely have the root anyway…*

Example of root finding from QM

- Consider the problem of a square well potential



$$V(x) = \begin{cases} 0 & -a < x < a \\ V_0 & \text{elsewhere} \end{cases}$$

Time independent Schrödinger equation is

$$\frac{-\hbar^2}{2m}\frac{\partial^2\psi}{\partial x^2} + V\psi = E\psi$$

Substitute potential

$$\Rightarrow \frac{\partial^2\psi}{\partial x^2} = \begin{cases} \frac{-2m}{\hbar^2} E\psi & -a < x < a \\ \frac{2m}{\hbar^2}(V_0 - E)\psi & \text{elsewhere} \end{cases} \Bigg\} \text{equation (1)}$$

- Solving the pair of equations (1) involves fitting a sinusoidal solution inside the square well and a decaying exponential solution outside it. The two solutions ψ and ψ' must be continuous at $x=\pm a$.

- We won't go through the details of the solution process, but two types of solution can be found, the parameters of which are given by transcendental equations

The solution classes

- Class I

$$\xi \tan \xi = \sqrt{\eta^2 - \xi^2}$$

- This gives even solutions for ψ

- Class II

$$\xi \cot \xi = -\sqrt{\eta^2 - \xi^2}$$

- This gives odd solutions for ψ

- Note,

$$\xi = \sqrt{2mE}\,\frac{a}{h}$$

$$\eta = \sqrt{2mV_0}\,\frac{a}{h}$$

We thus solve these equations to find x – this will give the energy levels of the solutions, E. However, we do not know a priori where the roots will be, so we need to use a bracket Finder.

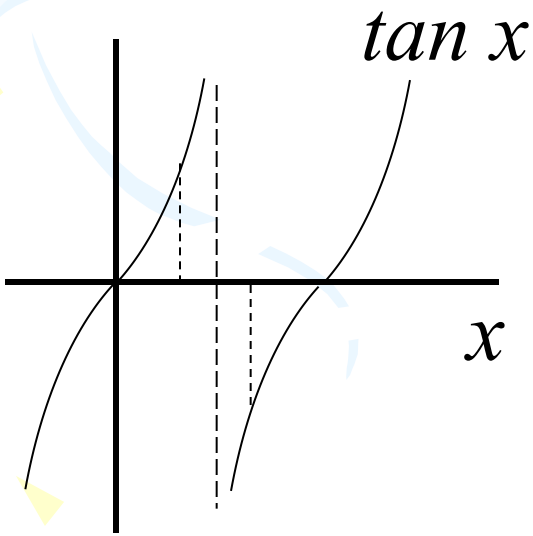
Root finding

- We recast the two solutions classes as equations

$$f(x) = x\tan x - \sqrt{h^2 - x^2} \qquad \text{and}$$

$$g(x) = x\cot x + \sqrt{h^2 - x^2}$$

- What problems might we expect?

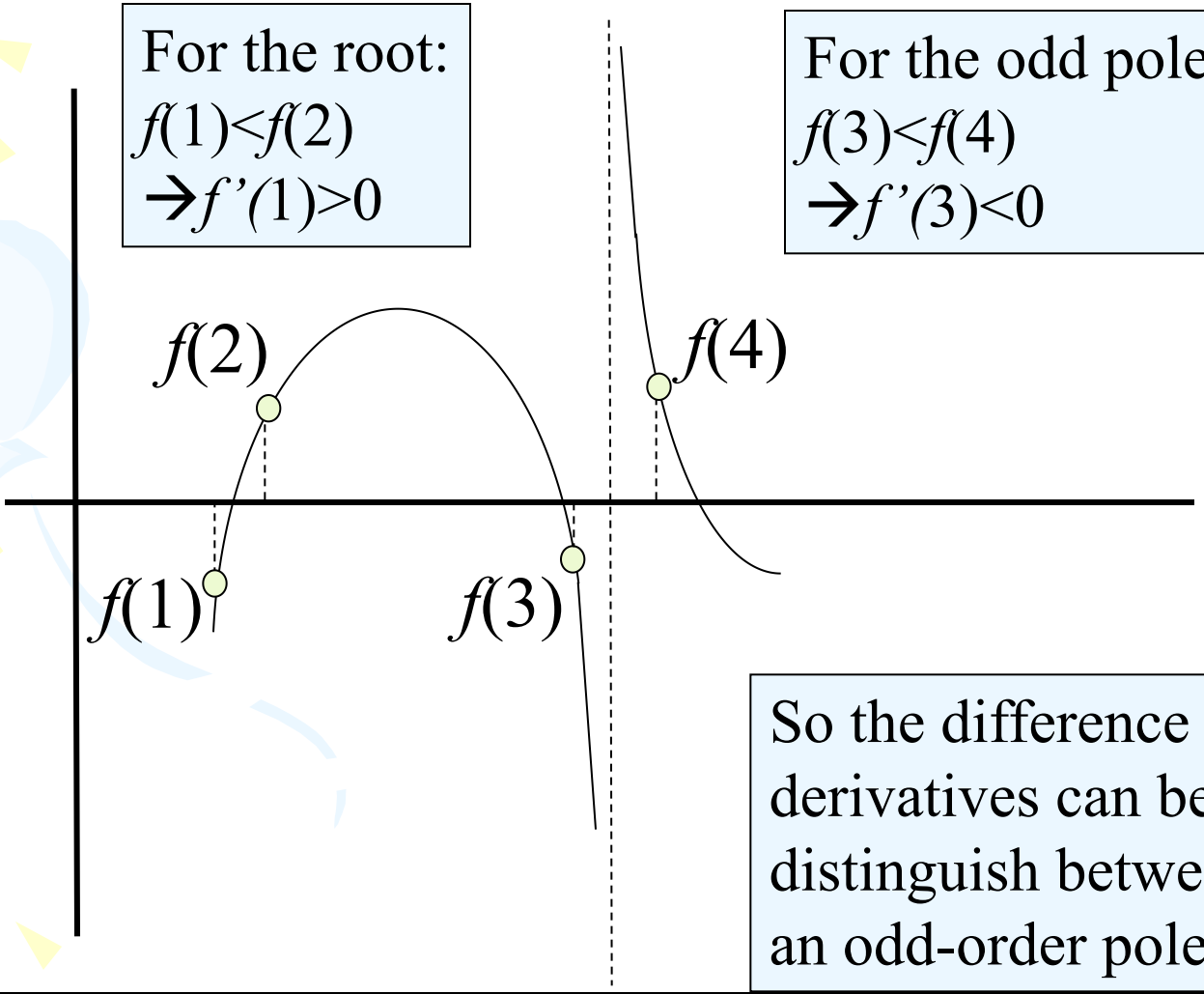


Both ***tan*** and ***cot*** have odd ordered poles that change sign. As we discussed in the last lecture this can lead to singularities being interpreted as roots – we need to look out for this.

Strategies for dealing with poles

For the root:
$f(1)<f(2)$
$\rightarrow f'(1)>0$

For the odd pole:
$f(3)<f(4)$
$\rightarrow f'(3)<0$



So the difference in signs of the derivatives can be used to distinguish between a root and an odd-order pole.

Strategy 2: multiply by new function

- We can multiply $f(\xi) = \xi \tan\xi - \sqrt{\eta^2 - \xi^2}$ by ***cos x*** to give $\tilde{f}(\xi) = \xi \sin\xi - \cos\xi\sqrt{\eta^2 - \xi^2}$

- Multiply $g(\xi) = \xi \cot\xi + \sqrt{\eta^2 - \xi^2}$ by ***sin x*** to give $\tilde{g}(\xi) = \xi \cos\xi + \sin\xi\sqrt{\eta^2 - \xi^2}$

- These functions will share the same roots as *g* & *f*
- WARNING: watch out for any spurious roots that are introduced by multiplying by the new function!

Make sure you check the roots explicitly

Consider $\tilde{g}(x)$

- By inspection we can see that x=0 is a root of this equation
- However, if we examine $g(x)$ as $x \to 0$

$$\lim_{\xi \to 0} g(\xi) = \lim_{\xi \to 0} \left(\xi \frac{\cos \xi}{\sin \xi} + \sqrt{\eta^2 - \xi^2} \right)$$

$$= \lim_{\xi \to 0} \left(\frac{\xi}{\sin \xi} \right) \lim_{\xi \to 0} (\cos \xi) + \lim_{\xi \to 0} \left(\sqrt{\eta^2 - \xi^2} \right)$$

$$= 1 + \eta \neq 0$$

Always be aware that multiplying by a new function may induce spurious roots.

Summary

- Global bracket finding is difficult
  - Bracket finder given here is a sound first approach, but it isn't perfect
  - Key step is locally adaptive evaluation of the step between evaluation points
- The Muller-Brent method improves upon Newton-Raphson by using a quadratic fit
- Distinguishing between poles and roots is possible provided one is prepared to look at derivatives
- Be careful we multiplying by additional functions to remove poles – this can induce spurious roots

Next Lecture

- Interpolation

Exercise 2

- *seek the roots of* $f(x)=2-x-e^{-x}$
  1) Secant root finder
  2) Müller-Brent root finder

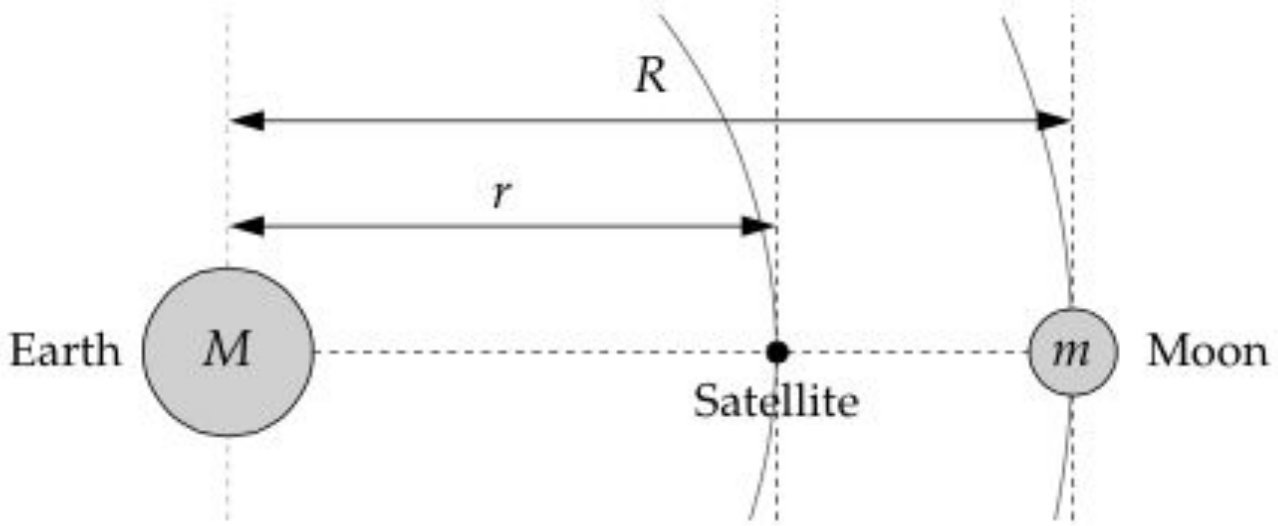
Homwork： 04/3/2019

Problem 1: Use Full Müller-Brent to find the roots of

$$f(x) = x\tan x - \sqrt{h^2 - x^2}$$
$$g(x) = x\cot x + \sqrt{h^2 - x^2}$$

Here h=0.5, 1.0

Problem 2: **The Lagrange point**: There is a magical point between the Earth and the Moon, called the $L1$ Lagrange point, at which a satellite will orbit the Earth in perfect synchrony with the Moon, staying always in between the two. This works because the inward pull of the Earth and the outward pull of the Moon combine to create exactly the needed centripetal force that keeps the satellite in its orbit. Here's the setup:



a) Assuming circular orbits, and assuming that the Earth is much more massive than either the Moon or the satellite, show that the distance r from the center of the Earth to the $L1$ point satisfies

$$\frac{GM}{r^2} - \frac{Gm}{(R-r)^2} = \omega^2 r$$

where M and m are the Earth and Moon masses, G is Newton's gravitational constant, and ω is the angular velocity of both the Moon and the satellite.

b) The equation above is a fifth-order polynomial equation in r (also called a quintic equation). Such equations cannot be solved exactly in closed form, but it's straightforward to solve them numerically. Write a program that uses either Newton's method or the secant method to solve for the distance r from the Earth to the $L1$ point. Compute a solution accurate to at least four significant figures.

The values of the various parameters are:
$G = 6.674 \times 10^{-11}$ m3kg^{-1}s^{-2},
$M = 5.974 \times 10^{24}$ kg,
$m = 7.348 \times 10^{22}$ kg,
$R = 3.844 \times 10^{8}$ m,
$\omega = 2.662 \times 10^{-6}$ s^{-1}.

You will also need to choose a suitable starting value for r, or two starting values if you use the secant method.

Problem 3. The roots of a polynomial: Consider the sixth-order polynomial

$$P(x) = 924x^6 - 2772x^5 + 3150x^4 - 1680x^3 + 420x^2 - 42x + 1.$$

There is no general formula for the roots of a sixth-order polynomial, but one can find them easily enough using a computer.

a) Make a plot of $P(x)$ from x=0 to x=1 and by inspecting it find rough values for the six roots of the polynomial—the points at which the function is zero.

b) Write a code to solve for the positions of all six roots to at least ten decimal places of accuracy, using Newton's method.

c) Use the Strategies discussed in this lecture to seek all the six roots (extra credit)

Note that the polynomial in this example is just the sixth Legendre polynomial mapped onto the interval from zero to one.