Computational Methods
CMSC/AMSC/MAPL 460

Solving nonlinear equations and zero finding

Ramani Duraiswami,
Dept. of Computer Science
Finding zeroes of functions

• Where does it arise?
• Solving functional equations
  – Polynomials: Quadratic, cubic, quadric, quintic …
    • Galois in 1830 proved that there is no finite sequence of rational operations
      plus square/cube roots that can solve quintic or higher equations.
    • Aside: Galois died in a duel at a very young age (<21)
      http://www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Galois.html
  – Minimization or maximization of a function
    • Recall if $f(x)$ has a minimum or maximum, $df/dx=0$
  – Intersection of curves
  – Others
    • Integration formulas …
The simplest algorithm: Bisection

• Suppose we know that
  – \( f \) is continuous in an interval \([a,b]\)
  – \( f(a) > 0 \) and \( f(b) < 0 \) OR \( f(a)< 0 \) and \( f(b) > 0 \)

• What does this tell us about \( f \) in the interval \([a,b]\)?
  – By continuity, there must be at least one zero somewhere in between!
  – Hold on to this fact and squeeze the interval till we bracket the zero!

• Evaluate \( f((a+b)/2) \).
  – If it has the same sign as \( f(a) \), then at least one zero is in \([(a+b)/2, b]\)
  – If it has the same sign as \( f(b) \), then at least one zero is in \([a,(a+b)/2]\)

• Repeat until a zero is obtained, or the interval is small enough.
Example

• Solve $x=2^{1/2}$;
  – Find $x_*$ for which $f(x):x^2 - 2$ has a zero
  – Evaluate $f(1)$ and $f(2)$
  – We know $f(1)<0$ and $f(2)>0$ \([1,2]\)
  – Next guess $1\frac{1}{2}$ : $f(1\frac{1}{2}) >0$ \([1,1\frac{1}{2}]\)
  – Next guess $1\frac{1}{4}$ : $f(1\frac{1}{4}) <0$ \([1\frac{1}{4},1\frac{1}{2}]\)
  – Next guess $1\ 3/8$ : $f(1\ 3/8) <0$ \([1\ 3/8,1\frac{1}{2}]\)
  – …
    \[\frac{3}{8}, \frac{5}{16}, \frac{13}{32}, \frac{27}{64}, \ldots\]

• Will the algorithm ever stop?
  – Always will converge in floating point
  – After 52 steps $a = 1.41421356237309$ $b = 1.41421356237310$
  – Difference smaller than machine epsilon

• This algorithm needs one function evaluation per iteration
Convergence analysis

• For iterative algorithms, we want to know how the error decreases after each iteration

• Here the imprecision in locating the root (or the error), approximately halves at each step

• What is the trend in convergence

• Error equals \((x_k - x_\star) = e_k\)

\[
e_k = e_{k-1}/2
\]

\[
e_k = e_0/2^k = e_02^{-k}
\]

• Take logs

\[
\log e_k = \log e_0 - k \log 2
\]

– Semilog plot shows linear rate

– What is the slope here?

• This algorithm is said to have linear convergence
Another algorithm

• Note that in bisection we take the half-way point no matter how close \(f(a)\) or \(f(b)\) maybe to zero
• Instead let us fit a straight line joining \(f(a)\) and \(f(b)\)
• Find where it becomes zero
• Recall the straight line is

\[
g(x) = f(a) + (x - a) \frac{(f(b) - f(a))}{(b - a)}
\]

\[
g(a) = f(a) \quad g(b) = f(a) + f(b) - f(a) = f(b)
\]

• Set \(g(x) = 0\)

\[
x_\star = a - f(a) \frac{(b - a)}{(f(b) - f(a))}
\]

Evaluate \(f(x_\star)\)

Depending on sign of \(f(x_\star)\) replace \(a\) or \(b\) with \(x_\star\)
Modified secant method

• Algorithm is a modified secant method
• Requires one function evaluation per iteration
  – Convergence is superlinear
    \[ e_k = c \ e_{k-1}^a \]
    \[ e_k = c \ (c e_{k-2}^a)^a = C e_0^{-ka} \]
    Here \( a \) is the golden ratio \((1+\sqrt{5})/2\)

• What is a secant?
  – In trigonometry it is the function defined as
    \[ \sec(z) = 1/\cos(z) \]
  – Here the use is more from the geometry of a circle
    • A SECANT is a line that intersects a circle in exactly two points.
    • Every secant forms a chord.
Secant method

- In bisection and the modified secant method we were required to first bracket a zero.
- This can be time consuming … and is indeed the hard part of minimization.
- On the other hand once this is done we have ensured convergence.
- Instead in the secant method choose two points.
- Fit straight line and evaluate its zero.
- Choose next point and repeat.
Secant method

\[ x_{k+1} = x_k - f(x_k)(x_k - x_{k-1})/(f(x_k) - f(x_{k-1})) \]

- When it converges, the convergence is super linear
- \( e_{k+1} = c \cdot e_k^\phi \)
- \( \phi = (5^{1/2} + 1)/2 = 1.62 \ldots \)
- Each step the error is raised to a power >1
- Convergence to zero occurs quickly
- But, convergence is not guaranteed till we are near the zero
Newton’s method

• Several ways to derive: We choose Taylor series
• I want \( f(x_\ast) = 0 \)
• But I have \( f(x_k) \) which is not zero
• Let me guess that \( f(x_k + h) \) will be zero
• \( f(x_k + h) = f(x_k) + hf'(x_k) + O(h^2) \)
• Ignore terms of \( O(h^2) \)
  – Approximate curve locally as straight line
  – When will this not work?
• Solve \( f(x_k) + hf''(x_k) = 0 \) for \( h \)
• So \( h = -\frac{f(x_k)}{f''(x_k)} \)
• So \( x_{k+1} = x_k + h = x_k - \frac{f(x_k)}{f''(x_k)} \)
• Repeat until convergence
• Apply Newton method to square root
• \(X = \sqrt{a}\)
• \(f(x) = x^2 - a\)
• \(f'(x) = 2x\)
• \(x_{k+1} = x_k + h = x_k - \frac{(x_k^2 - a)}{2x_k}\)
• \(\text{Guess } \sqrt{2} = 1\)
• \(1 - \frac{(1 - 2)}{2} = 1.5\)
• \(1.5 - \frac{(2.25 - 2)}{3} = 1.5 - 0.0833 = 1.4167\)
• ...
• \(\text{Converges rapidly within 5 iterations to machine precision}\)
Convergence analysis

- For iterative algorithms, we want to know how the error decreases after each iteration
- We also want to check how much each iteration costs
- Optimal algorithm is the one that achieves a given error for a given cost
- Algorithm convergence function evaluations per step
  - Bisection  Linear  One
  - Secant  Superlinear  One
  - Newton  Quadratic  Two
- Which method is better?
- Define better:
  - Bisection guaranteed to converge, but slow
  - Secant one evaluation per step Newton: two
  - Newton quadratic convergence Secant super linear
  - Newton needs derivative, which may be unavailable
Comparing convergence

• Suppose cost of function evaluations for derivative and function are similar

• Then let Newton method converge in \( n \) steps to error \( \tau \)

• So \( e_0^{2n} \leq \tau \)
  – Take logs: \( 2n \log e_0 \leq \log \tau \)
  – So \( 2n \geq |\log \tau| / |\log e_0| \) \( n \geq (2)^{-1} (|\log \tau| / |\log e_0|) \)

• Secant will require \( s \) steps to ensure \( e_0^{1.62s} \leq \tau \)
  – For secant: \( s \geq (1.62)^{-1} (|\log \tau| / |\log e_0|) \)

• Cost of Newton is \( 2n \) while that of secant is \( s \)

• Which is larger?

• \( \frac{\text{Cost}_{\text{Newton}}}{\text{Cost}_{\text{Secant}}} = 2n/s = 1/(1.62)^{-1} = 1.62 > 1 \)
  – So Secant is cheaper!
Infinite cycles

- Newton's method could iterate forever!
  \[ x_{n+1} = x_n - f(x_n)/f'(x_n) \]
- cycles back and forth around point \( a \) if
  \[ x_{n+1} - a = -(x_n - a) \]
- \[ x - a - \frac{f(x)}{f'(x)} = -(x - a) \]
  - Rewrite as an ODE for \( f \)
    \[ \frac{f'(x)}{f(x)} = \frac{1}{2(x - a)} \]
  - Solution
    \[ f(x) = \text{sign}(x - a) \sqrt{|x - a|} \]
  - Such cycles could exist with secant methods as well.