September 2, 2014

Chapter 3: Nonlinear equations in One Variable

Uri M. Ascher and Chen Greif
Department of Computer Science
The University of British Columbia
{ascher,greif}@cs.ubc.ca

Slides for the book
A First Course in Numerical Methods (published by SIAM, 2011)
`http://bookstore.siam.org/cs07/`

Goals of this chapter

- To develop useful methods for a basic, simply stated problem, including such favourites as fixed point iteration and Newton's method;
- to develop and assess several algorithmic concepts that are prevalent throughout the field of numerical computing;
- to study basic algorithms for minimizing a function in one variable.

Outline

- Bisection method
- Fixed point iteration
- Newton's method and variants
- Minimizing a function in one variable

Roots of continuous functions

- For a given continuous function $f = f(x)$, consider the problem

  $$f(x) = 0,$$

  where x is the independent variable in $[a, b]$.
- If x^* is value of x where f vanishes then x^* is a **root** or **zero** of f.
- *How many roots?* This depends not only on the function but also on the interval.
  Example: $f(x) = \sin(x)$ has one root in $[-\pi/2, \pi/2]$, two roots in $[-\pi/4, 5\pi/4]$, and no roots in $[\pi/4, 3\pi/4]$.
- Why study a nonlinear problem before addressing linear ones?!
  - One linear equation, e.g. $ax = b$, is too simple (solution: $x = b/a$, duh!), whereas a system of linear equations (Chapter 5 and 7) can have many complications.
  - Several important general methods can be described in a simple context.
  - Several important algorithm properties can be defined and used in a simple context.

Desirable algorithm properties

Generally for a nonlinear problem, must consider an iterative method: starting with initial iterate (guess) x_0, generate sequence of iterates $x_1, x_2, \ldots, x_k, \ldots$ that hopefully converge to a root x^*.

Desirable properties of a contemplated iterative method are:

- Efficient: requires a small number of function evaluations.
- Robust: fails rarely, if ever. Announces failure if it does fail.
- Requires a minimal amount of additional information such as the derivative of f.
- Requires f to satisfy only minimal smoothness properties.
- Generalizes easily and naturally to many equations in many unknowns.

Like many other wish-lists, this one is hard to fully satisfy...

Outline

- Bisection method
- Fixed point iteration
- Newton's method and variants
- Minimizing a function in one variable

Bisection method

- Simple
- Safe, robust
- Requires only that f be continuous
- Slow
- Hard to generalize to systems

Bisection method development

- Given $a < b$ such that $f(a) \cdot f(b) < 0$, there must be a root in $[a, b]$. Refer to $[a, b]$ as the uncertainty interval.
- So, at each iteration, evaluate $f(p)$ at $p = \frac{a+b}{2}$ and check the sign of $f(a) \cdot f(p)$.
  If positive, set $a \leftarrow p$, if negative set $b \leftarrow p$.
  Note: only one evaluation of the function f per iteration.
- This reduces the length of the uncertainty interval by factor 0.5 at each iteration. So, setting $x_n = p$, the error after n iterations satisfies $|x^* - x_n| \leq \frac{b-a}{2} \cdot 2^{-n}$.
- Stopping criterion: given (absolute) tolerance `atol`, require $\frac{b-a}{2} \cdot 2^{-n} \leq \texttt{atol}$.
- This allows a priori determination of the number of iterations n: unusual in algorithms for nonlinear problems.

bisect function

```
function [p,n] = bisect(func,a,b,fa,fb,atol)
if (a >= b) | (fa*fb >= 0) | (atol <= 0)
   disp('something wrong with the input:  quitting');
   p = NaN; n=NaN;
   return
end

n = ceil ( log2 (b-a) - log2 (2*atol));
for k=1:n
   p = (a+b)/2;
   fp = feval(func,p);
   if fa * fp < 0
     b = p;
     fb = fp;
   else
     a = p;
     fa = fp;
   end
end
p = (a+b)/2;
```

Outline

- Bisection method
- Fixed point iteration
- Newton's method and variants
- Minimizing a function in one variable

Fixed point iteration

This is an intuitively appealing approach which often leads to simple algorithms for complicated problems.

1. Write given problem

   $$f(x) = 0$$

   as

   $$g(x) = x,$$

   so that $f(x^*) = 0$ iff $g(x^*) = x^*$.

2. Iterate:

   $$x_{k+1} = g(x_k), \quad k = 0, 1, \ldots,$$

   starting with guess x_0.

It's all in the choice of the function g.

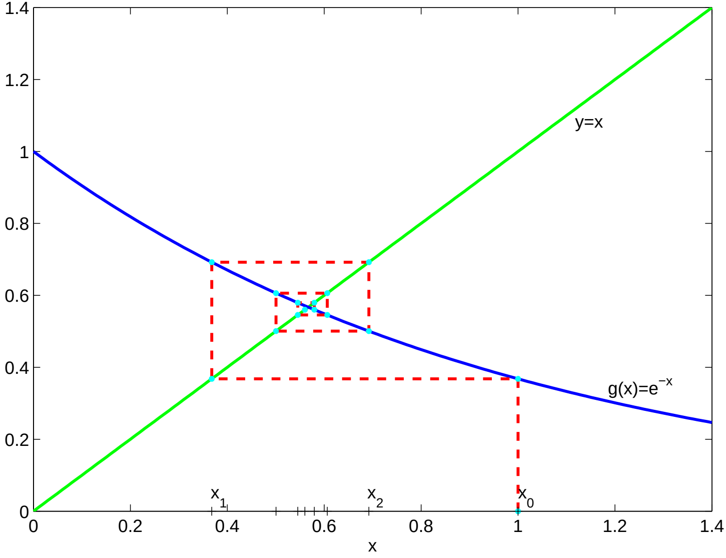
Choosing the function g

- Note: there are many possible choices g for the given f: this is a family of methods.
- Examples:

$$
\begin{aligned}
g(x) &= x - f(x), \\
g(x) &= x + 2f(x), \\
g(x) &= x - f(x)/f'(x) \quad (\text{assuming f}' \text{ exists and f}'(\text{x}) \neq 0).
\end{aligned}
$$

  The first two choices are simple, the last one has potential to yield fast convergence (we'll see later).
- Want resulting method to
  - be simple;
  - converge; and
  - do it rapidly.

Graphical illustration, $x = e^{-x}$, starting from $x_0 = 1$



Fixed Point Theorem

If $g \in C[a,b]$, $g(a) \geq a$ and $g(b) \leq b$, then there is a fixed point x^* in the interval $[a,b]$.
If, in addition, the derivative g' exists and there is a constant $\rho < 1$ such that the derivative satisfies

$$|g'(x)| \leq \rho \quad \forall\, x \in (a,b),$$

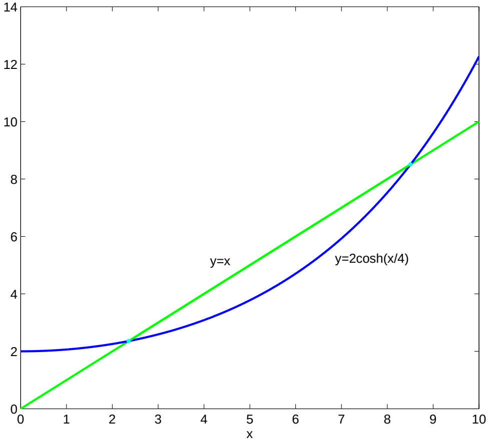
then the fixed point x^* is unique in this interval.

Convergence of fixed point iteration

- Assuming $\rho < 1$ as for the fixed point theorem, obtain
$$|x_{k+1} - x^*| = |g(x_k) - g(x^*)| \le \rho |x_k - x^*|.$$
- This is a **contraction** by factor ρ.
- So
$$|x_{k+1} - x^*| \le \rho |x_k - x^*| \le \rho^2 |x_{k-1} - x^*| \le \ldots \le \rho^{k+1} |x_0 - x^*| \to 0.$$
- The smaller ρ the faster convergence is.

Example: cosh with two roots

$$f(x) = g(x) - x, \quad g(x) = 2\cosh(x/4)$$



Fixed point iteration with g

For tolerance 1.e-8:

- Starting at $x_0 = 2$ converge to x_1^* in 16 iterations.
- Starting at $x_0 = 4$ converge to x_1^* in 18 iterations.
- Starting at $x_0 = 8$ converge to x_1^* (even though x_2^* is closer to x_0).
- Starting at $x_0 = 10$ obtain **overflow** in 3 iterations.

Note: bisection yields both roots in 27 iterations.

Rate of convergence

- Suppose we want $|x_k - x^*| \approx 0.1|x_0 - x^*|$.
- Since $|x_k - x^*| \leq \rho^k |x_0 - x^*|$, want

$$\rho^k \approx 0.1,$$

  i.e., $k \log_{10} \rho \approx -1$.
- Define the **rate of convergence** as

$$rate = -\log_{10} \rho.$$

- Then it takes about $k = \lceil 1/rate \rceil$ iterations to reduce the error by more than an order of magnitude.

Return to cosh example

- Bisection: $rate = -\log_{10} 0.5 \approx .3 \quad \Rightarrow \quad k = 4$.
- For the root x_1^* of fixed point example, $\rho \approx 0.31$ so

$$rate = -\log_{10} 0.31 \approx .5, \quad \Rightarrow \quad k = 2.$$

- For the root x_2^* of fixed point example, $\rho > 1$ so

$$rate = -\log_{10}(\rho) < 0, \quad \Rightarrow \quad \text{no convergence.}$$

Outline

- Bisection method
- Fixed point iteration
- Newton's method and variants
- Minimizing a function in one variable

Newton's method

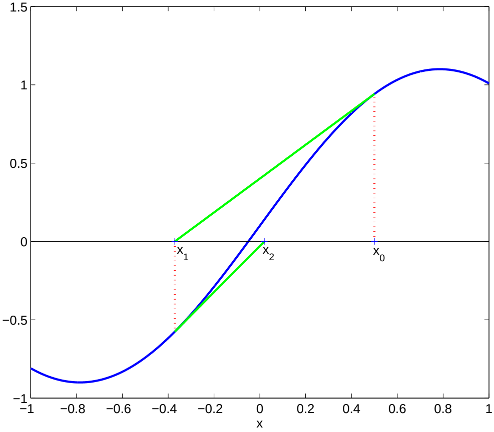
This fundamentally important method is everything that bisection is not, and vice versa:

- Not so simple
- Not very safe or robust
- Requires more than continuity on f
- Fast
- Automatically generalizes to systems

Derivation

- By Taylor series,
$$f(x) = f(x_k) + f'(x_k)(x - x_k) + f''(\xi(x))(x - x_k)^2/2.$$
- So, for $x = x^*$
$$0 = f(x_k) + f'(x_k)(x^* - x_k) + \mathcal{O}\left((x^* - x_k)^2\right).$$
- The method is obtained by neglecting nonlinear term, defining $0 = f(x_k) + f'(x_k)(x_{k+1} - x_k)$, which gives the iteration step
$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, 2, \ldots.$$

A geometric interpretation



Next iterate is x-intercept of the tangent line to f at current iterate.

Example: cosh with two roots

- The function

$$f(x) = 2\cosh(x/4) - x$$

  has two solutions in the interval $[2, 10]$.
- Newton's iteration is

$$x_{k+1} = x_k - \frac{2\cosh(x_k/4) - x_k}{0.5\sinh(x_k/4) - 1}.$$

- For absolute tolerance 1.e-8:
  - Starting from $x_0 = 2$ requires 4 iterations to reach x_1^*.
  - Starting from $x_0 = 4$ requires 5 iterations to reach x_1^*.
  - Starting from $x_0 = 8$ requires 5 iterations to reach x_2^*.
  - Starting from $x_0 = 10$ requires 6 iterations to reach x_2^*.
- Tracing the iteration's progress:

k	0	1	2	3	4	5
$f(x_k)$	-4.76e-1	8.43e-2	1.56e-3	5.65e-7	7.28e-14	1.78e-15

- Note that the number of significant digits essentially doubles at each iteration (until the 5th, when roundoff error takes over).

Speed of convergence

A given method is said to be

- **linearly convergent** if there is a constant $\rho < 1$ such that

$$|x_{k+1} - x^*| \le \rho |x_k - x^*| \ ,$$

  for all k sufficiently large;

- **quadratically convergent** if there is a constant M such that

$$|x_{k+1} - x^*| \le M |x_k - x^*|^2 \ ,$$

  for all k sufficiently large;

- **superlinearly convergent** if there is a sequence of constants $\rho_k \to 0$ such that

$$|x_{k+1} - x^*| \le \rho_k |x_k - x^*|,$$

  for all k sufficiently large.

Convergence theorem for Newton's method

If $f \in C^2[a, b]$ and there is a root x^* in $[a, b]$ such that $f(x^*) = 0$, $f'(x^*) \neq 0$, then there is a number δ such that, starting with x_0 from anywhere in the neighborhood $[x^* - \delta, x^* + \delta]$, Newton's method converges quadratically.

Idea of proof:

- Expand $f(x^*)$ in terms of a Taylor series about x_k;
- divide by $f'(x_k)$, rearrange, and replace $x_k - \frac{f(x)}{f'(x_k)}$ by x_{k+1};
- find the relation between $e_{k+1} = x_{k+1} - x^*$ and $e_k = x_k - x^*$.

Secant method

- One potential disadvantage of Newton's method is the need to *know and evaluate* the derivative of f.
- The secant method circumvents the need for explicitly evaluating this derivative.
- Observe that near the root (assuming convergence)

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

- So, define Secant iteration

$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}, \quad k = 0, 1, 2, \ldots.$$

- Note the need for two initial starting iterates x_0 and x_1: a *two-step method*.

Example: cosh with two roots

$$f(x) = 2\cosh(x/4) - x.$$

Same absolute tolerance 1.e-8 and initial iterates as before:

- Starting from $x_0 = 2$ and $x_1 = 4$ requires 7 iterations to reach x_1^*.
- Starting from $x_0 = 10$ and $x_1 = 8$ requires 7 iterations to reach x_2^*.

k	0	1	2	3	4	5	6
$f(x_k)$	2.26	-4.76e-1	-1.64e-1	2.45e-2	-9.93e-4	-5.62e-6	1.30e-9

Observe superlinear convergence: much faster than bisection and simple fixed point iteration, yet not quite as fast as Newton's iteration.

Newton's method as a fixed point iteration

- If $g'(x^*) \neq 0$ then fixed point iteration converges linearly, as discussed before, as $\rho > 0$.
- Newton's method can be written as a fixed point iteration with

$$g(x) = x - \frac{f(x)}{f'(x)}.$$

From this we get $g'(x^*) = 0$.

- In such a situation the fixed point iteration may converge faster than linearly: indeed, Newton's method converges quadratically under appropriate conditions.

Outline

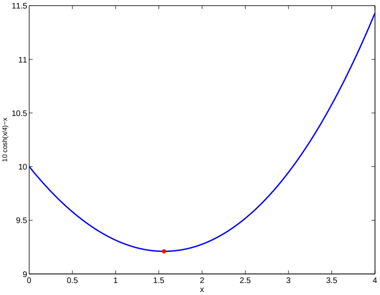
- Bisection method
- Fixed point iteration
- Newton's method and variants
- Minimizing a function in one variable

Minimizing a function in one variable

- Optimization is a vast subject, only some of which is covered in Chapter 9. Here, we just consider the simplest situation of minimizing a smooth function in one variable.
- **Example**: find $x = x^*$ that minimizes

$$\phi(x) = 10\cosh(x/4) - x.$$

- From the figure below, this function has no zeros but does appear to have one minimizer around $x = 1.6$.



Conditions for optimum and algorithm

- Necessary condition for an optimum:
  Suppose $\phi \in C^2$ and denote $f(x) = \phi'(x)$. Then a zero of f is a critical point of ϕ, i.e., where

  $$\phi'(x^*) = 0.$$

  To be a minimizer or a maximizer, it is necessary for x^* to be a critical point.
- Sufficient condition for an optimum:
  A critical point x^* is a minimizer if also $\phi''(x^*) > 0$.
- Hence, an algorithm for finding a minimizer is obtained by using one of the methods of this chapter for finding the roots of $\phi'(x)$, then checking for each such root x^* if also $\phi''(x^*) > 0$.
- Note: rather than finding all roots of ϕ' first and checking for minimum condition later, can do things more carefully and wisely, e.g. by sticking to steps that decrease $\phi(x)$.

Example

To find a minimizer for

$$\phi(x) = 10\cosh(x/4) - x,$$

1. Calculate gradient

$$f(x) = \phi'(x) = 2.5\sinh(x/4) - 1$$

2. Find root of $\phi'(x) = 0$ using any of our methods, obtaining

$$x^* \approx 1.56014.$$

3. Second derivative

$$\phi''(x) = 2.5/4\cosh(x/4) > 0 \quad \text{for all } x,$$

so x^* is a minimizer.