Nonlinear Systems

Problem setting: Find roots

$$f(x) = 0.$$

Often, methods are iterative.

Example. Compute quare roots

$$x^2 - A = 0.$$

A geometric approach: Find the side of the square whose area is A.
Start with a rectangle whose one side is x_c, then the other side is A/x_c so that its area is A.
Make the rectangle more square:

$$x_+ = \frac{1}{2}\left(x_c + \frac{A}{x_c}\right)$$

Then $x_c = x_+$ and iterate.

Three issues to deal with

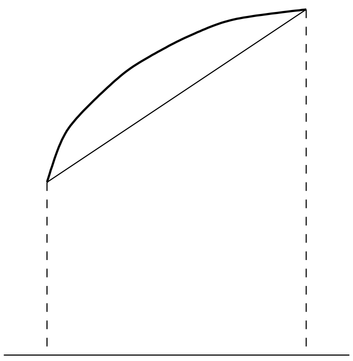
- Initialization (x_0)
- Convergence ($x_k \to x_*$?) and rate (how fast?)
- Termination

Initialization (square root problem): Present

$$A = m \times 4^e, \qquad 0.25 \le m < 1$$

then $\sqrt{A} = \sqrt{m} \times 2^e$. Assuming $0.25 \le A < 1$.
Linear interpolation of $f(A) = \sqrt{A}$ at $A = [0.25, 1]$:

$$p(A) = (1 + 2A)/3.$$



It can be shown that

$$\left|p(A) - \sqrt{A}\right| \leq 0.05$$

Initial value: $x_0 = (1 + 2A)/3$.

Convergence:

$$x_{k+1} = \frac{1}{2}\left(x_k + \frac{A}{x_k}\right)$$

Denote the error $e_k = |x_k - \sqrt{A}|$, then

$$\begin{aligned} e_{k+1} &= |x_{k+1} - \sqrt{A}| = \frac{1}{2}\left(\frac{x_k - \sqrt{A}}{\sqrt{x_k}}\right)^2 \\ &= \frac{1}{2|x_k|}e_k^2. \end{aligned}$$

It can be shown that $0.5 \leq x_k \leq 1.0$.

Since $e_0 = |x_0 - \sqrt{A}| = |p(A) - \sqrt{A}| \leq 0.05$,

$$e_k \leq e_{k-1}^2 \leq \cdots \leq e_0^{2^k} \leq (0.05)^{2^k}$$

We have shown the convergence.

How fast?

Rate: quadratic $e_{k+1} \leq ce_k^2$, each iteration doubles the accuracy.

Termination:

$$e_4 \leq e_0^{16} \leq (0.05)^{16}$$

Four iterations are enough for IEEE double precision (10^{-16}).

1 Bisection method

If $f(a)f(b) \le 0$ and $f(x)$ is continuous on $[a, b]$, then $f(x)$ has a root on $[a, b]$.

```
while (b-a)>tol
   m = (a+b)/2;
   if f(a)*f(m)<=0
      b = m;
   else
      a = m;
   end;
end;
r = (a + b)/2;
```

Two problems in the above algorithm:

- The while-loop may not terminate (why?)
- Two function evaluations per iteration

A better algorithm:

```
fa = f(a);
while (b-a)>tol + eps*max(|a|,|b|)
   m = (a + b)/2;
   fm = f(m);
   if fa*fm<=0
      b = m;
   else
      a = m; fa = fm;
   end;
end;
r = (a + b)/2;
```

Convergence:

Since $|a_k - b_k| \le |a - b|/2^k$, $x_* \in [a_k, b_k]$, and $x_k = (a_k + b_k)/2$, we have

$$e_k = |x_k - x_*| \le \frac{b-a}{2^{k+1}} \;\to\; 0$$

Rate: linear

$$e_{k+1} \leq ce_k$$

Improve accuracy by 1 bit per iteration or 1 decimal digit for every three or so iterations.

2 Newton's Method

The tangent line of $f(x)$ at x_c:

$$y = f(x_c) + (x - x_c)f'(x_c).$$

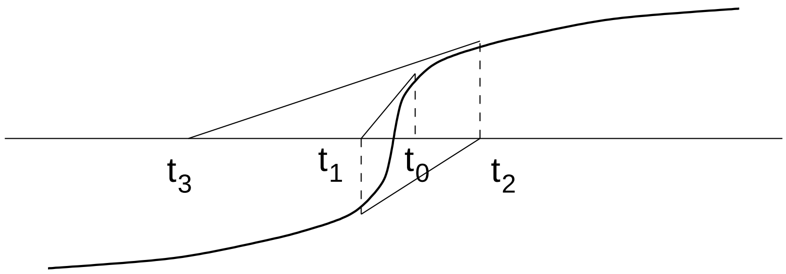
Set $y = 0$,

$$x_+ = x_c - \frac{f(x_c)}{f'(x_c)}$$

Example. Square root problem revisited, $f(x) = x^2 - A$

$$x_+ = x_c - \frac{x_c^2 - A}{2x_c} = \frac{1}{2}\left(x_c + \frac{A}{x_c}\right)$$

Convergence. No guarantee of convergence (unlike bisection), e.g., $f(x) = \arctan(x)$ ($|x_0| > 1.3917$).



Conditions for convergence (qualitative):

- x_0 close enough to x_*
- $f'(x)$ does not change sign near x_*
- $f(x)$ is not too nonlinear near x_*

Newton method is *local*.

Globalizing Newton method: combining bisection and Newton methods

Bracketing interval $[a, b]$, and $x_c = a$ or b
if $x_+ = x_c - f(x_c)/f'(x_c) \in [a, b]$
    bracketing interval $[a, x_+]$ or $[x_+, b]$
else
    $m = (a + b)/2$;
    bracketing interval $[a, m]$ or $[m, b]$
end

Termination criteria. Any one of:

- $(b_k - a_k) < \delta$
- $|f(x_c)| < \delta$
- Too many function evaluations

Avoiding derivatives. Approximation

$$f'(x_c) \approx \frac{f(x_c + \delta_c) - f(x_c)}{\delta_c}.$$

Choice of δ_c?

Example. Secant method ($\delta_c = x_- - x_c$)

$$f'(x_c) \approx \frac{f(x_c) - f(x_-)}{x_c - x_-},$$

then

$$x_+ = x_c - f(x_c)\frac{x_c - x_-}{f(x_c) - f(x_-)}.$$

Usually, the convergence rate (if it converges) is

$$\frac{1 + \sqrt{5}}{2} \approx 1.6$$

The secant method can be viewed as using x_c and x_- to construct a linear interpolation

$$f(x_c) + (x - x_c)\frac{f(x_c) - f(x_-)}{x_c - x_-}.$$

The root of this linear function is used as an improvement over x_c.

Inverse Quadratic Interpolation

The secant method linearly interpolates two previous points. What if we use quadratic interpolation by using three previous points? Let a, b, and c be three previous points and $p(x)$ is the quadratic interpolation:

$$a = p(f(a)), \quad b = p(f(b)), \quad c = p(f(c)),$$

then $p(0)$ gives an approximation of the zero of $f(x)$. The trouble with this method is that polynomial interpolation requires $f(a)$, $f(b)$, and $f(c)$ distinct. For example, if we try to compute $\sqrt{2}$ by finding the zero of $f(x) = x^2 - 2$ and $a = -2$, $b = 0$, and $c = 2$, then $f(a) = f(c)$ and the interpolation is undefined. If we start near this singular situation, say with $a = -2.001$, $b = 0$, $c = 1.999$, the next iterate is near $x = 500$. A practical method combines the reliable bisection method with the secant method and the inverse quadratic interpolation.

3 Muller's Method

This method can be used to find both real and complex roots. It is a generalization of the secant method. Given three estimates x_0, x_1, x_2, we construct a quadratic interpolation in Newton's form:

$$f(x_2) + (x - x_2)f[x_2, x_1] + (x - x_2)(x - x_1)f[x_2, x_1, x_0].$$

Rearranging the above Newton's divided difference form into a quadratic polynomial form

$$f(x_2) + w(x - x_2) + f[x_2, x_1, x_0](x - x_2)^2,$$

where

$$\begin{aligned} w &= f[x_2, x_1] + (x_2 - x_1)f[x_2, x_1, x_0] \\ &= f[x_2, x_1] + f[x_2, x_1] - f[x_2, x_1]. \end{aligned}$$

This quadratic polynomial has two roots. We choose the one that is the closest to the most current estimate x_2.

Usually, the convergence rate (if it converges) is approximately 1.84.

4 Systems of Nonlinear Equations

$$\begin{aligned} f_1(x_1, ..., x_n) &= 0 \\ f_2(x_1, ..., x_n) &= 0 \\ &\vdots \\ f_n(x_1, ..., x_n) &= 0 \end{aligned}$$

Denote

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

$\mathbf{f}$: vector-valued function, $\mathbf{x}$: vector.

Newton method.

$$\mathbf{x}_+ = \mathbf{x}_c + \mathbf{s}_c$$

where $\mathbf{s}_c$ is the solution of

$$\mathbf{f}(\mathbf{x}_c) + J(\mathbf{x}_c)\mathbf{s}_c = 0,$$

i.e., $\mathbf{s}_c = -J^{-1}(\mathbf{x}_c)\mathbf{f}(\mathbf{x}_c)$, where $J(\mathbf{x}_c)$ is the Jacobian of $\mathbf{f}$ at $\mathbf{x}_c$:

$$J(\mathbf{x}) = \left[\frac{\partial f_i}{\partial x_j}\right] = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

Avoiding derivatives. The jth column of $J(\mathbf{x})$

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_j} \\ \vdots \\ \frac{\partial f_n}{\partial x_j} \end{bmatrix} = \frac{\partial \mathbf{f}}{\partial x_j}$$

can be approximated by

$$\frac{\mathbf{f}(x_1, ..., x_j + \delta, x_{j+1}, ..., x_n) - \mathbf{f}(x_1, ..., x_n)}{\delta},$$

for some small δ.

Summary

- Issues in an iterative method: Initialization, convergence and rate of convergence, termination. The example of computing square root.
- Bisection method: Numerical termination problem.
- Newton's method: Initial value, convergence problems.
- Muller's method: Find complex root from real estimates.
- System of nonlinear equations: Generalization of Newton's method

Software packages

IMSL zporc, zplrc, zpocc

MATLAB roots

NAG c02agf, c02aff

NAPACK czero

References

Kendall E. Atkinson, *An Introduction to Numerical Analysis*, 2nd Edition, John Wiley & Sons, 1989.

R. P. Brent, *Algorithms for Minimization without Derivatives*, Prentice-Hall, Englewood Cliffs, NJ, 1973.

George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler, *Computer Methods for Mathematical Computations*, Prentice-Hall, Inc., 1977.