Topic 02: Root Finding

Part of “Numerical Methods for Differential Equations”, Colin Macdonald, cbm@math.ubc.ca.

Textbooks:
Burden and Faires;
Ascher and Greif, A *First Course in Numerical Methods*.

Root Finding

Iterative techniques for solving $f(x) = 0$ for $x$.

*Bisection:* start with an interval $[a, b]$ bracketing the root. Evaluate the midpoint. Replace one end, maintaining a root bracket. Linear convergence. Slow but **robust**.

*Newton’s Method:* $x_{k+1} = x_k - f(x_k)/f'(x_k)$. Faster, quadratic convergence (number of correct decimals places doubles each iteration).

Downsides of Newton’s Method: need derivative info, and additional smoothness. Convergence usually not guaranteed unless “sufficiently close”: not **robust**.

Rates of convergence

linear, superlinear and quadratic convergence.

Systems

$f(x) = 0$, but now $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$.

This is a system of nonlinear equations. Denote a solution as $\alpha \in \mathbb{R}^n$.

Derivation: Taylor expansion about $x$

$$0 = f(\alpha) = f(x) + J(x)(\alpha - x) + \text{h.o.t.}$$

where $J(x)$ is the Jacobian matrix...

Pretend h.o.t. are 0, so instead of $\alpha$ we find $x_{k+1}$:

$$0 = f(x_k) + J(x_k)(x_{k+1} - x_k)$$

In principle, can rearrange to solve for $x_k$ but better to solve

$$J_k \delta = -f(x_k)$$

That is, solve “$Ax = b$”. Then update:

$$x_{k+1} := x_k + \delta$$
Optimization


A huge area, concerned with minimizing (equiv. maximizing) a function \( f(x, y, z) \) subject to equality constraints \( h_{\text{eq}}(x, y, z) = 0 \) and inequality constraints \( h_{\text{ineq}}(x, y, z) < 0 \).

Just to scratch the surface... Consider scalar function of vector argument \( f(\vec{x}) \) and no constraints. From calculus: find min/max points by setting the derivative (gradient \( g(x) = \nabla f \)) equal to zero. Then use Newton’s method on the gradient.

\[
0 = g(x_k) + H(x_k)(x_{k+1} - x_k),
\]

or

\[
x_{k+1} = x_k - H^{-1}(x_k)g(x_k),
\]

\[
H(x_k)d_x = -\nabla f(x_k).
\]

We will need the Hessian matrix \( H \). For convergence, want \( H \) symmetric positive definite.

**Line search**

What about local max or saddle etc?

Find direction \( d_k \) then search along a line to find the best \( x_{k+1} = x_k + \tau_k d_k \). If not best, at least with \( x_{k+1} < x_k \).

**Proxy**

Alternative approach: build a local parabolic proxy for the surface \( f(\vec{x}) \), and minimize the proxy.

Possibly use a SPD matrix \( B_k \) instead of \( H \). Leads to “quasi-Newton methods”.