Nonlinear Equations

Given function \( f \), we seek value \( x \) for which

\[ f(x) = 0 \]

Solution \( x \) is root of equation, or zero of function \( f \)

So problem is known as root finding or zero finding

Example of nonlinear equation in one dimension

\[ x^2 - 4 \sin(x) = 0 \]

for which \( x = 1.9 \) is one approximate solution

Example of system of nonlinear equations in two dimensions

\[
\begin{align*}
  x_1^2 - x_2 + 0.25 &= 0 \\
  -x_1 + x_2^2 + 0.25 &= 0
\end{align*}
\]

for which \( x = [0.5, 0.5] \) is solution vector

Example: One Dimension

Nonlinear equations can have any number of solutions

- \( \exp(x) + 1 = 0 \) has no solution
- \( \exp(-x) - x = 0 \) has one solution
- \( x^2 - 4 \sin(x) = 0 \) has two solutions
- \( x^3 + 6x^2 + 11x - 6 = 0 \) has three solutions
- \( \sin(x) = 0 \) has infinitely many solutions

Nonlinear Equations

Two important cases

- Single nonlinear equation in one unknown, where

\[ f : \mathbb{R} \rightarrow \mathbb{R} \]

Solution is scalar \( x \) for which \( f(x) = 0 \)

- System of \( n \) coupled nonlinear equations in \( n \) unknowns, where

\[ f : \mathbb{R}^n \rightarrow \mathbb{R}^n \]

Solution is vector \( x \) for which all components of \( f \) are zero simultaneously, \( f(x) = 0 \)

Examples: Nonlinear Equations

Existence and Uniqueness

Existence and uniqueness of solutions are more complicated for nonlinear equations than for linear equations

- For function \( f : \mathbb{R} \rightarrow \mathbb{R} \), bracket \([a, b]\) is interval \([a, b]\) for which sign of \( f \) differs at endpoints

- If \( f \) is continuous and \( \text{sign}(f(a)) \neq \text{sign}(f(b)) \), then Intermediate Value Theorem implies there is \( x^* \in [a, b] \) such that \( f(x^*) = 0 \)

- There is no simple analog for \( n \) dimensions

Example: Systems in Two Dimensions

\[
\begin{align*}
  x_1^2 - x_2 + \gamma &= 0 \\
  -x_1 + x_2^2 + \gamma &= 0
\end{align*}
\]

\( \gamma = 0.5 \)

\( \gamma = 0.25 \)

\( \gamma = -0.5 \)

\( \gamma = -1.9 \)
Convergence Rate

- For general iterative methods, define error at iteration $k$ by
  \[ e_k = x_k - x^* \]
  where $x_k$ is approximate solution and $x^*$ is true solution
- For methods that maintain interval known to contain solution, rather than specific approximate value for solution, take error to be length of interval containing solution
- Sequence converges with rate $r$ if
  \[ \lim_{k \to \infty} \frac{\|e_{k+1}\|}{\|e_k\|^r} = C \]
  for some finite nonzero constant $C$

Sensitivity and Conditioning

- Conditioning of root finding problem is opposite to that for evaluating function
- Absolute condition number of root finding problem for root $x^*$ of $f: \mathbb{R} \to \mathbb{R}$ is $1/|f'(x^*)|$
- Root is ill-conditioned if tangent line is nearly horizontal
- In particular, multiple root ($m > 1$) is ill-conditioned
- Absolute condition number of root finding problem for root $x^*$ of $f: \mathbb{R}^n \to \mathbb{R}^n$ is $\| J_f(x^*)^{-1} \|$, where $J_f$ is Jacobian matrix of $f$, $J_f(x)_{ij} = \partial f_i(x)/\partial x_j$
- Root is ill-conditioned if Jacobian matrix is nearly singular

Multiplicity

- If $f(x^*) = f'(x^*) = f''(x^*) = \cdots = f^{(m-1)}(x^*) = 0$ but $f^{(m)}(x^*) \neq 0$ (i.e., with derivative is lowest derivative of $f$ that does not vanish at $x^*$), then root $x^*$ has \textit{multiplicity} $m$

![Graph of $x^2 - 2x + 1$ and $x^3 - 3x^2 + 3x - 1$]

- If $m = 1$ ($f(x^*) = 0$ and $f'(x^*) \neq 0$), then $x^*$ is \textit{simple} root

Well-conditioned

Ill-conditioned

Convergence Rate, continued

- Some particular cases of interest
  - $r = 1$: \textit{linear} ($C < 1$)
  - $r > 1$: \textit{superlinear}
  - $r = 2$: \textit{quadratic}

Example: Bisection Method

\[
f(x) = x^2 - 4 \sin(x) = 0
\]

<table>
<thead>
<tr>
<th>$a$</th>
<th>$f(a)$</th>
<th>$b$</th>
<th>$f(b)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00000</td>
<td>-2.6355844</td>
<td>3.000000</td>
<td>8.435520</td>
</tr>
<tr>
<td>1.00000</td>
<td>-2.6355844</td>
<td>2.000000</td>
<td>0.362810</td>
</tr>
<tr>
<td>1.50000</td>
<td>-1.7399800</td>
<td>2.000000</td>
<td>0.362810</td>
</tr>
<tr>
<td>1.75000</td>
<td>-0.8734444</td>
<td>2.000000</td>
<td>0.362810</td>
</tr>
<tr>
<td>1.87500</td>
<td>-0.3007180</td>
<td>2.000000</td>
<td>0.362810</td>
</tr>
<tr>
<td>1.87500</td>
<td>-0.3007180</td>
<td>1.937500</td>
<td>0.019849</td>
</tr>
<tr>
<td>1.90625</td>
<td>-0.143255</td>
<td>1.937500</td>
<td>0.019849</td>
</tr>
<tr>
<td>1.921875</td>
<td>-0.062406</td>
<td>1.937500</td>
<td>0.019849</td>
</tr>
<tr>
<td>1.926562</td>
<td>-0.021454</td>
<td>1.937500</td>
<td>0.019849</td>
</tr>
<tr>
<td>1.933594</td>
<td>-0.000846</td>
<td>1.937500</td>
<td>0.019849</td>
</tr>
<tr>
<td>1.933594</td>
<td>-0.000846</td>
<td>1.935457</td>
<td>0.004320</td>
</tr>
<tr>
<td>1.933594</td>
<td>-0.000846</td>
<td>1.934082</td>
<td>0.001736</td>
</tr>
</tbody>
</table>
Bisection Method, continued
- Bisection method makes no use of magnitudes of function values, only their signs.
- Bisection is certain to converge, but does so slowly.
- At each iteration, length of interval containing solution reduced by half, convergence rate is linear, with $r = 1$ and $C = 0.5$.
- One bit of accuracy is gained in approximate solution for each iteration of bisection.
- Given starting interval $[a, b]$, length of interval after $k$ iterations is $(b - a) / 2^k$, so achieving error tolerance of $\text{tol}$ requires $k \approx \log_2 \left( \frac{b - a}{\text{tol}} \right)$ iterations, regardless of function $f$ involved.

Example: Fixed-Point Problems
If $f(x) = x^2 - x - 2$, then fixed points of each of functions
- $g_1(x) = x^2 - 2$
- $g_2(x) = \sqrt{x + 2}$
- $g_3(x) = 1 + \frac{2}{x}$
- $g_4(x) = \frac{x^2 + 2}{2x - 1}$
are solutions to equation $f(x) = 0$.

Convergence of Fixed-Point Iteration
- If $x^* = g(x^*)$ and $|g'(x^*)| < 1$, then there is interval containing $x^*$ such that iteration $x_{k+1} = g(x_k)$ converges to $x^*$ if started within that interval.
- If $|g'(x^*)| > 1$, then iterative scheme diverges.
- Asymptotic convergence rate of fixed-point iteration is usually linear, with constant $C = |g'(x^*)|$.
- But if $g'(x^*) = 0$, then convergence rate is at least quadratic.

Fixed-Point Problems
**Fixed point** of given function $g : \mathbb{R} \rightarrow \mathbb{R}$ is value $x$ such that $x = g(x)$.
- Many iterative methods for solving nonlinear equations use fixed-point iteration scheme of form $x_{k+1} = g(x_k)$ where fixed points for $g$ are solutions for $f(x) = 0$.
- Also called functional iteration, since function $g$ is applied repeatedly to initial starting value $x_0$.
- For given equation $f(x) = 0$, there may be many equivalent fixed-point problems $x = g(x)$ with different choices for $g$.

Newton’s Method
- Truncated Taylor series
  $$f(x + h) \approx f(x) + f'(x)h$$
  is linear function of $h$ approximating $f$ near $x$.
- Replace nonlinear function $f$ by this linear function, whose zero is $h = -f(x)/f'(x)$.
- Zeros of original function and linear approximation are not identical, so repeat process, giving Newton's method.

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$
Secant Method

For each iteration, Newton’s method requires evaluation of both function and its derivative, which may be inconvenient or expensive.

In secant method, derivative is approximated by finite difference using two successive iterates, so iteration becomes

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

Convergence rate of secant method is normally superlinear, with \( r \approx 1.618 \).

Secant Method, continued

Example: Secant Method

Use secant method to find root of

\[ f(x) = x^2 - 4 \sin(x) = 0 \]

Taking \( x_0 = 1 \) and \( x_1 = 3 \) as starting guesses, we obtain

\[ \begin{array}{cccc}
  x & f(x) & h \\
  1.000000 & -2.365884 & \\
  3.000000 & 8.435520 & -1.561930 \\
  1.438070 & -1.896774 & 0.286735 \\
  1.724805 & -0.977706 & 0.305029 \\
  2.029833 & 0.534305 & -0.107789 \\
  1.922044 & -0.061523 & 0.111130 \\
  1.933174 & -0.000306 & 0.000383 \\
  1.933754 & 0.000000 & -0.00004 \\
  1.933754 & 0.000000 & 0.000000 \\
\end{array} \]

Higher-Degree Interpolation

Secant method uses linear interpolation to approximate function whose zero is sought.

Higher convergence rate can be obtained by using higher-degree polynomial interpolation.

For example, quadratic interpolation (Muller’s method) has superlinear convergence rate with \( r \approx 1.839 \).

Unfortunately, using higher degree polynomial also has disadvantages:

- Interpolating polynomial may not have real roots
- Roots may not be easy to compute
- Choice of root to use as next iterate may not be obvious
Inverse Interpolation

- Good alternative is inverse interpolation, where $x_k$ are interpolated as function of $y_k = f(x_k)$ by polynomial $p(y)$, so next approximate solution is $p(0)$
- Most commonly used for root finding is inverse quadratic interpolation

**Example: Inverse Quadratic Interpolation**

**Example: Inverse Quadratic Interpolation**

- Use inverse quadratic interpolation to find root of
  
  $$f(x) = x^2 - 4 \sin(x) = 0$$
  
- Taking $x = 1, 2, 3$ as starting values, we obtain

<table>
<thead>
<tr>
<th>$x$</th>
<th>$f(x)$</th>
<th>$h$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000000</td>
<td>-2.365884</td>
<td></td>
</tr>
<tr>
<td>2.000000</td>
<td>0.362810</td>
<td></td>
</tr>
<tr>
<td>3.000000</td>
<td>8.435520</td>
<td></td>
</tr>
<tr>
<td>1.886318</td>
<td>-0.243433</td>
<td>-0.113682</td>
</tr>
<tr>
<td>1.939558</td>
<td>0.030786</td>
<td>0.053240</td>
</tr>
<tr>
<td>1.933742</td>
<td>-0.000060</td>
<td>-0.005815</td>
</tr>
<tr>
<td>1.933754</td>
<td>0.000000</td>
<td>0.000011</td>
</tr>
<tr>
<td>1.933754</td>
<td>0.000000</td>
<td>0.000000</td>
</tr>
</tbody>
</table>

- Convergence rate is normally $\approx 1.839$

  < interactive example >

Linear Fractional Interpolation

- Interpolation using rational fraction of form
  
  $$\phi(x) = \frac{x - a}{b - x}$$

  is especially useful for finding zeros of functions having horizontal or vertical asymptotes
- $\phi$ has zero at $x = a$, vertical asymptote at $x = b$, and horizontal asymptote at $y = 1/v$
- Given approximate solution values $a, b, c$, with function values $f_a, f_b, f_c$, next approximate solution is $c + h$, where
  
  $$h = \frac{(a - c)(b - c)(f_a - f_c)}{(a - c)(f_a - f_b) + (b - c)(f_c - f_b)}$$

  - Convergence rate is normally $\approx 1.839$, same as for quadratic interpolation (inverse or regular)

  < interactive example >

Safeguarded Methods

- For polynomial $p(x)$ of degree $n$, one may want to find all $n$ of its zeros, which may be complex even if coefficients are real
- Several approaches are available
  - Use root-finding method such as Newton’s or Muller’s method to find one root, deflate it out, and repeat
  - Form companion matrix of polynomial and use eigenvalue routine to compute all its eigenvalues
  - Use method designed specifically for finding all roots of polynomial, such as Jenkins-Traub
Solving systems of nonlinear equations is much more difficult than scalar case because:
- Wider variety of behavior is possible, so determining existence and number of solutions or good starting guess is much more complex.
- There is no simple way, in general, to guarantee convergence to desired solution or to bracket solution to produce absolutely safe method.
- Computational overhead increases rapidly with dimension of problem.

Cost per iteration of Newton’s method for dense problem in $n$ dimensions is substantial:
- Solving system $E$.
- Evaluating at new point, $s_k$.

For Newton step $s_k$, then take as next iterate:
$$x_{k+1} = x_k + s_k$$

### Example: Newton’s Method

- In $n$ dimensions, Newton’s method has form:
  $$x_{k+1} = x_k - J(x_k)^{-1}f(x_k)$$
- where $J(x)$ is Jacobian matrix of $f$.
- In practice, we do not explicitly invert $J(x_k)$, but instead solve linear system:
  $$J(x_k)s_k = -f(x_k)$$
- for Newton step $s_k$, then take as next iterate:

### Evaluating at new point:
$$f(x) = \begin{bmatrix} 3x_1^2 + 2x_2 - 2 \\ 4x_1^2 + 4x_2 - 4 \end{bmatrix} = 0$$
- Jacobian matrix is $J_f(x) = \begin{bmatrix} 6x_1 & 2 \\ 8x_1 & 4 \end{bmatrix}$.
- If we take $x_0 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$, then:
  $$f(x_0) = \begin{bmatrix} 3 \\ 13 \end{bmatrix}, \quad J_f(x_0) = \begin{bmatrix} 1 & 2 \\ 2 & 16 \end{bmatrix}$$
- Solving system:
  $$\begin{bmatrix} 3 & 2 \\ 2 & 13 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} -3 \\ -13 \end{bmatrix}$$
  gives $s_0 = \begin{bmatrix} -1.83 \\ -0.58 \end{bmatrix}$.

### Convergence of Newton’s Method

- Differentiating corresponding fixed-point operator:
  $$g(x) = x - J(x)^{-1}f(x)$$
  and evaluating at solution $x^*$ gives:
  $$G(x^*) = I - (J(x^*)^{-1}J(x^*) + \sum_{i=1}^n f_i(x^*)H_i(x^*)) = O$$
  where $H_i(x)$ is component matrix of derivative of $J(x)^{-1}$.
- Convergence rate of Newton’s method for nonlinear systems is normally quadratic, provided Jacobian matrix $J(x^*)$ is nonsingular.
- But it must be started close enough to solution to converge.

### Secant Updating Methods

- Secant updating methods reduce cost by:
  - Using function values at successive iterates to build approximate Jacobian and avoiding explicit evaluation of derivatives.
  - Updating factorization of approximate Jacobian rather than refactoring it each iteration.
- Most secant updating methods have superlinear but not quadratic convergence rate.
- Secant updating methods often cost less overall than Newton’s method because of lower cost per iteration.
**Broyden's Method**

- Broyden’s method is typical secant updating method

- Beginning with initial guess $x_0$ for solution and initial approximate Jacobian $B_0$, following steps are repeated until convergence

  $x_0 = $ initial guess  
  $B_0 = $ initial Jacobian approximation

  for $k = 0, 1, 2, \ldots$

  Solve $B_k s_k = -f(x_k)$ for $s_k$

  $x_{k+1} = x_k + s_k$

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

  $B_{k+1} = B_k + ((y_k - B_k s_k s^T_k)/(s^T_k s_k))$

  end

Example: Broyden’s Method

- Use Broyden’s method to solve nonlinear system

  $f(x) = \begin{bmatrix} x_1^2 + x_2^2 - 2 \\ x_1^2 + 4x_2^2 - 4 \end{bmatrix} = 0$

- If $x_0 = [1 \ 2]^T$, then $f(x_0) = [3 \ 13]^T$, and we choose

  $B_0 = J_f(x_0) = \begin{bmatrix} 1 & 2 \\ 2 & 16 \end{bmatrix}$

- Solving system

  $\begin{bmatrix} 1 & 2 \\ 2 & 16 \end{bmatrix} s_0 = \begin{bmatrix} -3 \\ -13 \end{bmatrix}$

  gives $s_0 = \begin{bmatrix} -1.83 \\ 0.58 \end{bmatrix}$, so $x_1 = x_0 + s_0 = \begin{bmatrix} -0.83 \\ 1.42 \end{bmatrix}$

Example, continued

- Evaluating at new point $x_1$ gives $f(x_1) = \begin{bmatrix} 0.472 \\ -8.28 \end{bmatrix}$

- From updating formula, we obtain

  $B_1 = \begin{bmatrix} 1 & 2 \\ 2 & 16 \end{bmatrix} + \begin{bmatrix} -0.34 & 0 \\ 0 & -0.74 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ -0.34 & 15.3 \end{bmatrix}$

  gives $s_1 = \begin{bmatrix} 0.59 \\ -0.30 \end{bmatrix}$, so $x_2 = x_1 + s_1 = \begin{bmatrix} 0.24 \\ 1.12 \end{bmatrix}$

Robust Newton-Like Methods

- Newton’s method and its variants may fail to converge when started far from solution

- Safeguards can enlarge region of convergence of Newton-like methods

- Simplest precaution is damped Newton method, in which new iterate is

  $x_{k+1} = x_k + \alpha_k s_k$

  where $s_k$ is Newton (or Newton-like) step and $\alpha_k$ is scalar parameter chosen to ensure progress toward solution

- Parameter $\alpha_k$ reduces Newton step when it is too large, but $\alpha_k = 1$ suffices near solution and still yields fast asymptotic convergence rate

Trust-Region Methods

- Another approach is to maintain estimate of trust region, where Taylor series approximation, upon which Newton’s method is based, is sufficiently accurate for resulting computed step to be reliable

- Adjusting size of trust region to constrain step size when necessary usually enables progress toward solution even starting far away, yet still permits rapid convergence once near solution

- Unlike damped Newton method, trust region method may modify direction as well as length of Newton step

- More details on this approach will be given in Chapter 6