CS 357 - 15 Solving Nonlinear Equations

Boyang Li (boyangl3)

Root of a function: For a function $f(x) : \mathbb{R} \to \mathbb{R}$, the root of a function is all $x \in \mathbb{R}$ such that $f(x) = 0$. For a multi-dimensional function, the root is all vector $\mathbf{x}$ in the domain such that $f(\mathbf{x}) = \vec{0}$.

Solution of an Equation: For an equation $f(x) = y$, we can construct a new function such that $\tilde{f}(x) = f(x) - y$m then the roots of $\tilde{f}(x)$ are the solutions of $f(x) = y$. This applies to multi-dimensional equations as well.

General Methods: Sometimes it is not easy to apply theorems to some nonlinear equations, so we need to approximate the solution. The basic logistic is "Initial Guess - Check Residual - Iteration".

Convergence: Iterative methods converges with rate r if

$$\lim_{k\to\infty} \frac{\|e_{k+1}\|}{\|e_k\|^r} = C$$

where e_i is the error at iteration i, and C is a constant.

The speed of convergence is determined by the value of r:

r	Type of convergence	Accurate digits
1	Linear	Gain constant accurate digits every iteration
$1 < r < 2$	Superlinear	
2	Quadratic	Double the number of accurate digits every iteration

Bisection Method:

- **Iteration Steps:**
  1. Take 2 points a and b and make sure $f(a) \cdot f(b) < 0$.
  2. Take the midpoint of $[a, b]$, $c = \dfrac{a+b}{2}$.
  3. Evaluate $f(c)$ and use c to replace either a or b, to make the signs of 2 endpoints are still opposite.
- **Convergence:** Linear convergence, $e_k = \dfrac{b-a}{2^k}$ and $C = \dfrac{1}{2}$.
- **Cost:** Except the fist iteration, every iteration requires one new function evaluation.
- **Drawback:** Applicable to functions that we can get the points a and b such that $f(a)$ and $f(b)$ have opposite signs.

Newton's Method:

- **Motivation:** $f(x_k + h_k) \approx f(x_k) + f'(x_k) \cdot h_k$ (First 2 terms of Taylor's expansion)
- **Iteration Steps:**
  1. $f(x_k) + f'(x_k) \cdot h_k = 0 \rightarrow h_k = -\dfrac{f(x_k)}{f'(x_k)}$ (h_k is called Newton step)
  2. $x_{k+1} = x_k + h_k = x_k - \dfrac{f(x_k)}{f'(x_k)}$ (This step is called Newton update)
- **Convergence:** Quadratic convergence.
- **Cost:** Each iteration we need to evaluate 2 functions, $f(x_k)$ and $f'(x_k)$.
- **Drawbacks:**
  - Cost of computing functions
  - Requires differentiable functions
  - Initial guess need to be close to the solution/root, otherwise, it will be divergence.

Secant Method:

- **Relation with Newton's Method:** Sometimes we cannot evaluate the derivative of a function directly, so we need to approximate the derivative by applying

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

- **Iteration Steps:** Same as Newton's Method.
- **Convergence:** Superlinear convergence, $r = \dfrac{\sqrt{5}+1}{2}$.
- **Cost:** Except the first iteration, every iteration requires one function evaluation.
- **Drawbacks:**
  - Same as Newton's methods but does not require a derivative
  - Not converge as quickly as Newton's method
  - Need 2 initial guesses near the root

1D Summary (From CS357 Online Textbook):

Method	Update	Convergence	Cost
Bisection	Check signs of $f(a)$ and $f(b)$ $$t_k = \frac{\lvert b-a\rvert}{2^k}$$	Linear ($r = 1$ and $c = 0.5$)	One function evaluation per iteration, no need to compute derivatives
Secant	$$x_{k+1} = x_k + h$$ $$h = -f(x_k)/dfa$$ $$dfa = \frac{f(x_k) - f(x_{k-1})}{(x_k - x_{k-1})}$$	Superlinear ($r = 1.618$), local convergence properties, convergence depends on the initial guess	One function evaluation per iteration (two evaluations for the initial guesses only), no need to compute derivatives
Newton	$$x_{k+1} = x_k + h$$ $$h = -f(x_k)/f'(x_k)$$	Quadratic ($r = 2$), local convergence properties, convergence depends on the initial guess	Two function evaluations per iteration, requires first order derivatives

N-D Newton's Method:

- **Motivation:** $f(\mathbf{x}_k + \mathbf{s}_k) \approx f(\mathbf{x}_k) + \mathbb{J}(\mathbf{x}_k) \cdot \mathbf{s}_k$ (Introduced from 1-D Newton's Method, where $\mathbb{J}(\mathbf{x})$ is the Jacobian matrix of $f(\mathbf{x})$.
- **Iteration Steps:**
  1. $f(\mathbf{x}_k) + \mathbb{J}(\mathbf{x}_k) \cdot \mathbf{s}_k = 0 \rightarrow \mathbf{s}_k = -\mathbb{J}(\mathbf{x}_k)^{-1} \cdot f(\mathbf{x}_k)$ ($\mathbf{s}_k$ can also be calculated by solving the system $\mathbb{J}(\mathbf{x}_k) \cdot \mathbf{s}_k = -f(\mathbf{x}_k)$).
  2. $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k = \mathbf{x}_k - \mathbb{J}(\mathbf{x}_k)^{-1} \cdot f(\mathbf{x}_k)$
- **Cost:** The cost of calculating $\mathbf{s}$ by solving the linear system is $\mathcal{O}(n^3)$, the cost of calculating $\mathbb{J}(\mathbf{x})$ is $\mathcal{O}(n^2)$.
- **Drawbacks:**
  - Like 1D, only converges locally
  - Expensive to compute Jacobian matrix and solve linear system.