UNIVERSITY AT ALBANY
STATE UNIVERSITY OF NEW YORK

CSI 401 (Fall 2025)

Numerical Methods

Lecture 12: Nonlinear Equation Solver: Bisection Method

Chong Liu

Department of Computer Science

Nov 3, 2025

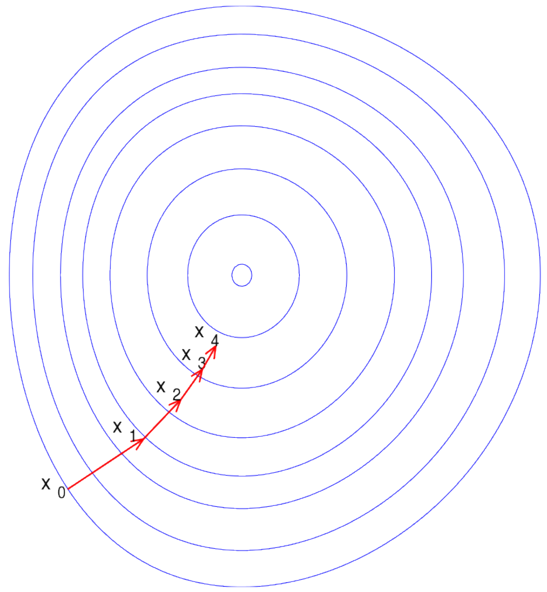
Recap: How do we optimize a continuously differentiable function in general?

- The problem: $\min_{\theta} f(\theta)$

- Discussion: How do you solve this optimization problem?

- Gradient descent in iterations

$$\theta_{t+1} = \theta_t - \eta_t \nabla f(\theta_t)$$

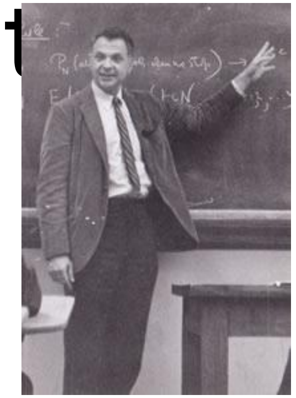


Recap: Back to linear regression: How to solve it using Gradient Descent?

- $\hat{w} = \text{argmin}_w \frac{1}{n}\sum_{i=1}^{n}(x_i^T w - y_i)^2 = \text{argmin}_w ||Xw - y||_2^2$

- In-class exercise: Write the GD updating rule for solving w.
  - $w \leftarrow w - 2\eta X^T(Xw - y)$

Recap: Stochastic Gradient Descent (Robbins-Monro 1951)



Herbert Robbins
1915 - 2001

- Gradient descent

$$\theta_{t+1} = \theta_t - \eta_t \nabla f(\theta_t)$$

- Stochastic gradient descent
  - Using a stochastic approximation of the gradient:

$$\theta_{t+1} = \theta_t - \eta_t \hat{\nabla} f(\theta_t)$$

Recap: The power of SGD

- Extremely simple:
  - A few lines of code
- Extremely scalable
  - Just a few pass of the data, no need to store the data
- Extremely general:
  - In addition to linear regression, in practice it can solve most optimization problems of differentiable functions
    - E.g., Training neural networks, Transformer, Generative Pretrained Transformer
  - Foundational algorithm of the AI revolution as we see today!

Recap: Time complexity of direct solver and GD/SGD for solving linear regression

- Direct solver
  - $O(nd^2 + d^3)$
- GD:
  - $O(ndT)$
- SGD:
  - $O(dT)$

- $T = \text{n\_iterations}$

Recap: What's Linear Programming (LP)?

- An optimization problem of linear objective functions with linear constraints.
  - Objective function can be minimized or maximized
  - Constraints can be in equalities or inequalities
  - All functions must be linear functions

- 2 examples:

$$\begin{aligned} \min\ & x_1 + 2x_2 \\ \text{s.t. } & x_1 + x_2 \leq 3 \\ & x_1 \geq 1 \end{aligned}$$

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.t. } & x_1 + x_2 = 3 \end{aligned}$$

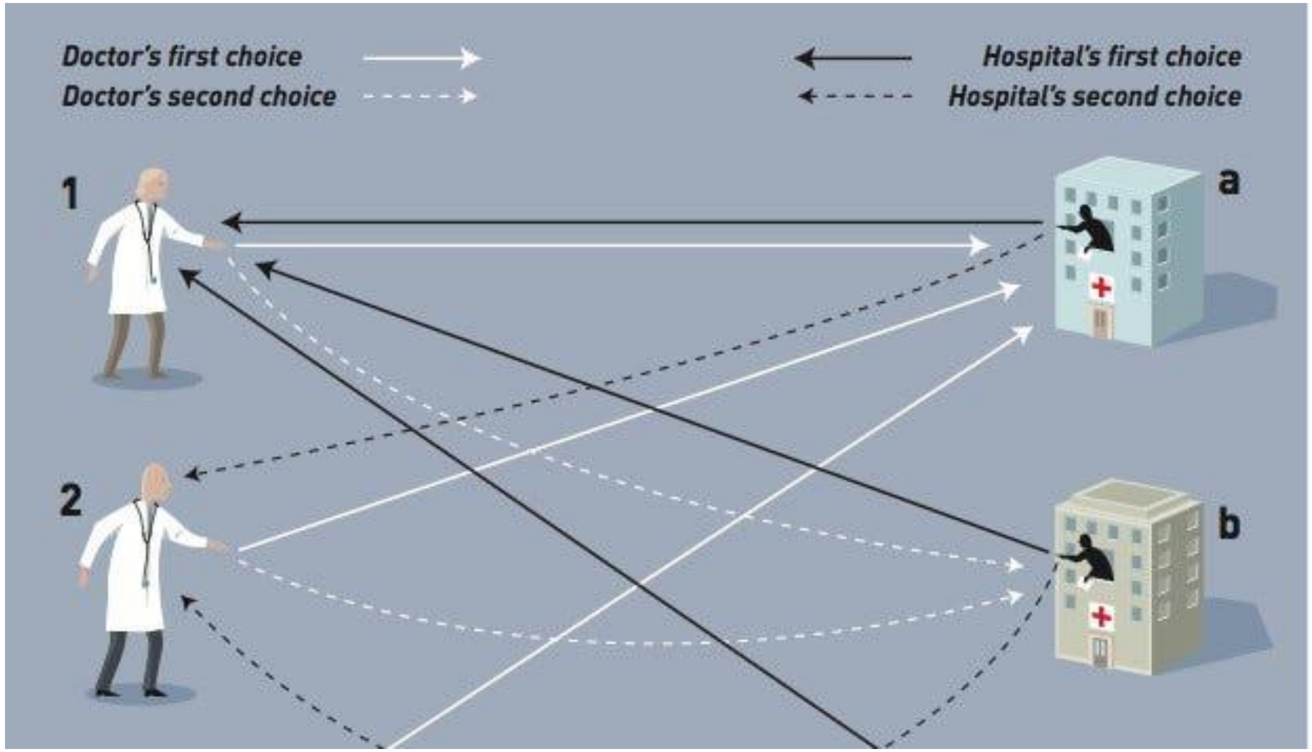
Recap: Application of LP: Matching problem

- Company (hospital) - Candidate (doctor) matching problem

- Each doctor:
  - Fits one position

- Doctors/hospitals:
  - Have their preferences

- Goal:
  - Put doctors to positions
  - Such that overall best match



Recap: Application of LP: Matching problem

- Company (hospital) – Candidate (doctor) matching problem
    - $x_{ij} \in \{0,1\}$ denotes assignment of doctor i to hospital j
    - $c_{ij} \in [0, 1]$ denotes the match between doctor i and hospital j
  - The objective function maximizes the overall match
  - 1st constraint ensures each hospital can hire a doctor
  - 2nd constraint ensures each doctor can find a job

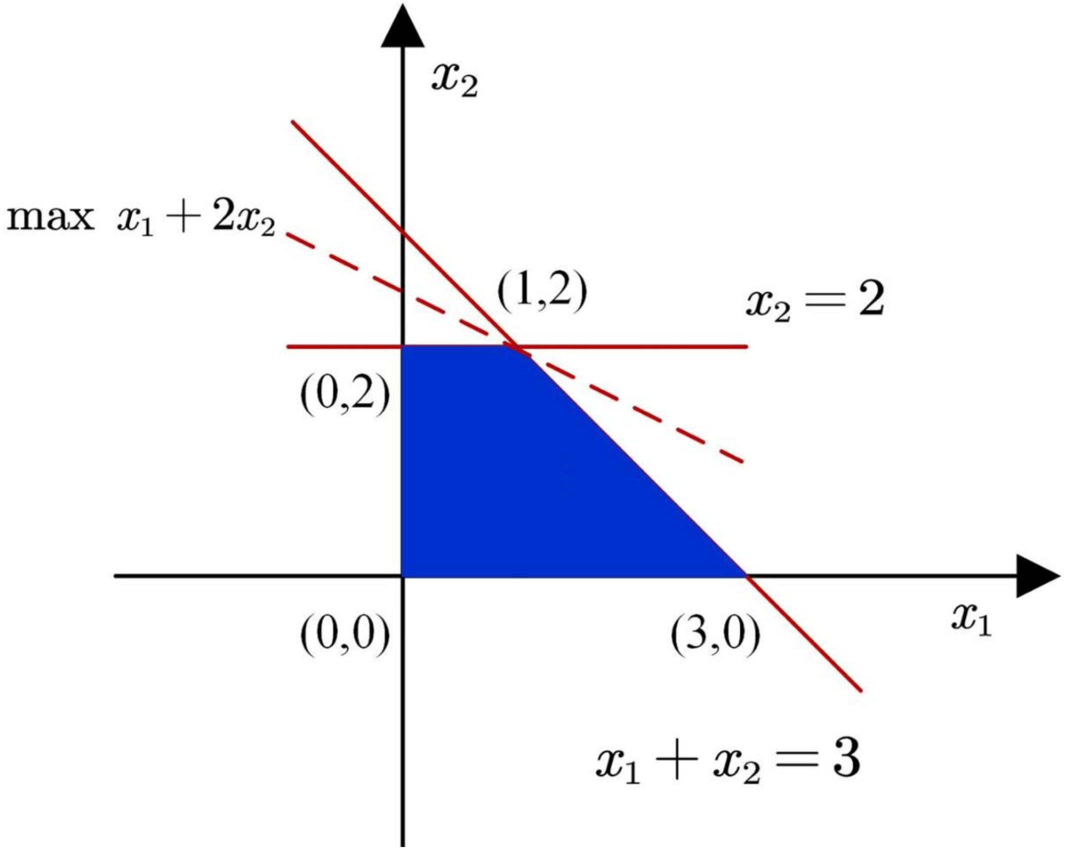
$$\max \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} x_{ij}$$

$$\text{s.t. } \sum_{i=1}^{N} x_{ij} = 1, \ \forall j = 1, 2, \ldots, N$$

$$\sum_{j=1}^{N} x_{ij} = 1, \ \forall i = 1, 2, \ldots, N$$

Recap: How to solve the LP problem?

$$\begin{aligned} \text{maximize } x_1 + 2x_2 & \\ \text{subject to } x_1 + x_2 &\leq 3 \\ x_2 &\leq 2 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$



- For most 2-d LP problems,
  1. We can draw it's feasible region
  2. And move it's objective function

- In-class exercise: Draw the feasible region defined by constraints.

Agenda

- More on linear programming
  - From primal to dual LP problems
  - Rewrite LP problems
- Non-linear equation solver: Bisection method

Dual problem of linear programming

- For every **primal** linear program, there is an associated **dual** LP that expresses the same optimization problem from a different perspective.

- Primal LP:
  - Max $c^T x$ $s.t.$ $Ax \leq b,\ x \geq 0,$
- Dual LP:
  - Min $b^T y$ $s.t.$ $A^T y \geq c,\ y \geq 0.$

- They are mathematically linked — this is not coincidence, but a property of convex optimization and linear algebra.

From primal to dual LP problems

- In-class exercise:
  - Work on the following two LP problems by drawing graphs

$$\max Z = x_1 + x_2$$

$$\text{s.t.} \begin{cases} x_1 + 2x_2 \le 8, \\ 3x_1 + 2x_2 \le 18, \\ x_1, x_2 \ge 0. \end{cases}$$

$$\min Z = 8y_1 + 18y_2$$

$$\text{s.t.} \begin{cases} y_1 + 3y_2 \ge 1, \\ 2y_1 + 2y_2 \ge 1, \\ y_1, y_2 \ge 0. \end{cases}$$

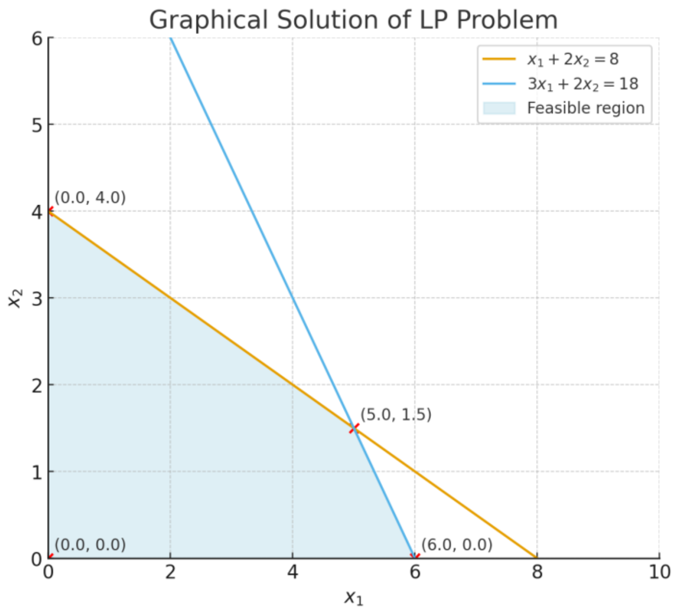
  - What can you see from their optimal Z?

From primal to dual LP problems

- Solutions to in-class exercise:

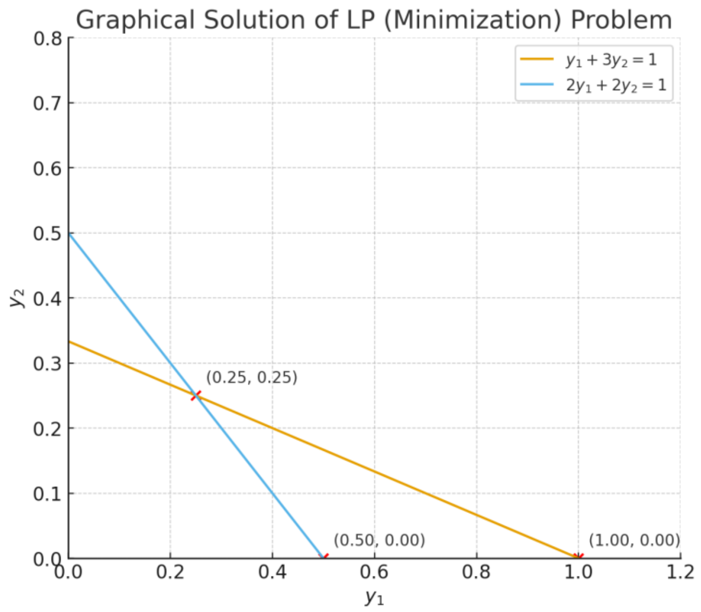
$$\max Z = x_1 + x_2$$

$$\text{s.t.} \begin{cases} x_1 + 2x_2 \leq 8, \\ 3x_1 + 2x_2 \leq 18, \\ x_1, x_2 \geq 0. \end{cases}$$



$$\min Z = 8y_1 + 18y_2$$

$$\text{s.t.} \begin{cases} y_1 + 3y_2 \geq 1, \\ 2y_1 + 2y_2 \geq 1, \\ y_1, y_2 \geq 0. \end{cases}$$



- They are primal and dual LP problems!

From primal to dual LP problems

- Primal problem:

$$\max\ z = 4x_1 + x_2 + 5x_3 + 3x_4$$

$$\begin{aligned} x_1 - x_2 - x_3 + 3x_4 &\leq 1 \\ 5x_1 + x_2 + 3x_3 + 8x_4 &\leq 55 \\ -x_1 + 2x_2 + 3x_3 - 5x_4 &\leq 3 \\ x_i &\geq 0 \end{aligned}$$

- Key idea:
  - Multiply each constraint with a non-negative multiplier and form linear combinations of constraints.

$$y_1(x_1 - x_2 - x_3 + 3x_4) + y_2(5x_1 + x_2 + 3x_3 + 8x_4) + y_3(-x_1 + 2x_2 + 3x_3 - 5x_4) \leq y_1 + 55y_2 + 3y_3.$$

$$(y_1 + 5y_2 - y_3)x_1 + (-y_1 + y_2 + 2y_3)x_2 + (-y_1 + 3y_2 + 3y_3)x_3 + (3y_1 + 8y_2 - 5y_3)x_4 \leq y_1 + 55y_2 + 3y_3.$$

- Finally, dual problem:

$$\min\ u = y_1 + 55y_2 + 3y_3$$

$$\begin{aligned} y_1 + 5y_2 - y_3 &\geq 4 \\ -y_1 + y_2 + 2y_3 &\geq 1 \\ -y_1 + 3y_2 + 3y_3 &\geq 5 \\ 3y_1 + 8y_2 - 5y_3 &\geq 3 \\ y_i &\geq 0 \end{aligned}$$

Dual problem of linear programming

- Economic Interpretation
  - The dual variables y represent **shadow prices** — the value of relaxing each constraint by one unit.
  - In a resource allocation problem, each y_i tells how much the objective (profit) would improve if resource i were increased slightly.

- Weak Duality:
  For any feasible x (primal) and y (dual), $c^T x \leq b^T y$.
  - The dual provides an upper bound (for maximization problems).

- Strong Duality:
  At the optimal solutions x^*, y^*, $c^T x^* = b^T y^*$.
  - Solving one problem solves the other — they share the same optimal value.

Dual problem of linear programming

- Why we study dual problems?

- Duality helps:
  - **Check optimality:** If primal and dual feasible solutions give the same objective, both are optimal.
  - **Perform sensitivity analysis:** Dual variables show how changes in constraints affect the outcome.
  - **Simplify computation:** Some LPs are easier to solve in dual form (e.g., when constraints >> variables).

Linear programming in practice

- Problem definition in Matlab (linprog) and Python (scipy.linprog)

$$\min_x f^T x \text{ such that } \begin{cases} A \cdot x \leq b, \\ Aeq \cdot x = beq, \\ lb \leq x \leq ub. \end{cases}$$

$$\begin{aligned} \min_x \quad & c^T x \\ \text{such that } & A_{ub} x \leq b_{ub}, \\ & A_{eq} x = b_{eq}, \\ & l \leq x \leq u, \end{aligned}$$

- My problem definition does fit them! What can I do?
  - Many ways to redefine the problem:
    - Max to min
    - Inequalities to equalities
    - Free variables to non-negative variables

Redefine a LP problem (max to min)

- Put the minus in front of the objective function to change “max” to “min”

$$
\begin{aligned}
\max x_1 &+ 2x_2 \\
\text{s.t. } x_1 + x_2 &\leq 3 \\
x_2 &\leq 2 \\
x_1 &\geq 0 \\
x_2 &\geq 0
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
\min \quad -x_1 &- 2x_2 \\
\text{s.t. } x_1 + x_2 &\leq 3 \\
x_2 &\leq 2 \\
x_1 &\geq 0 \\
x_2 &\geq 0
\end{aligned}
$$

Redefine a LP problem (inequalities to equalities)

- Introduce additional variables to change inequality constraints to equality constraints.

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.t. } & x_1 + x_2 \leq 3 \\ & x_1, x_2 \geq 0 \end{aligned} \quad \Longrightarrow \quad \begin{aligned} \max\ & x_1 + 2x_2 \\ \text{s.t. } & x_1 + x_2 + x_3 = 3 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

Redefine a LP problem (Free variables to non-negative variables)

- For free variables (variables that can be negative),
  - Redefine them as the difference between two non-negative variables

$$\max x_1 + 2x_2$$
$$\text{s.t. } x_1 + x_2 \leq 3$$

$$x_1 = x_3 - x_4, x_2 = x_5 - x_6$$

$$\begin{aligned} \max x_3 - x_4 + 2x_5 - 2x_6 \\ \text{s.t. } x_3 - x_4 + x_5 - x_6 \leq 3 \\ x_3, x_4, x_5, x_6 \geq 0 \end{aligned}$$

Beyond linear programming

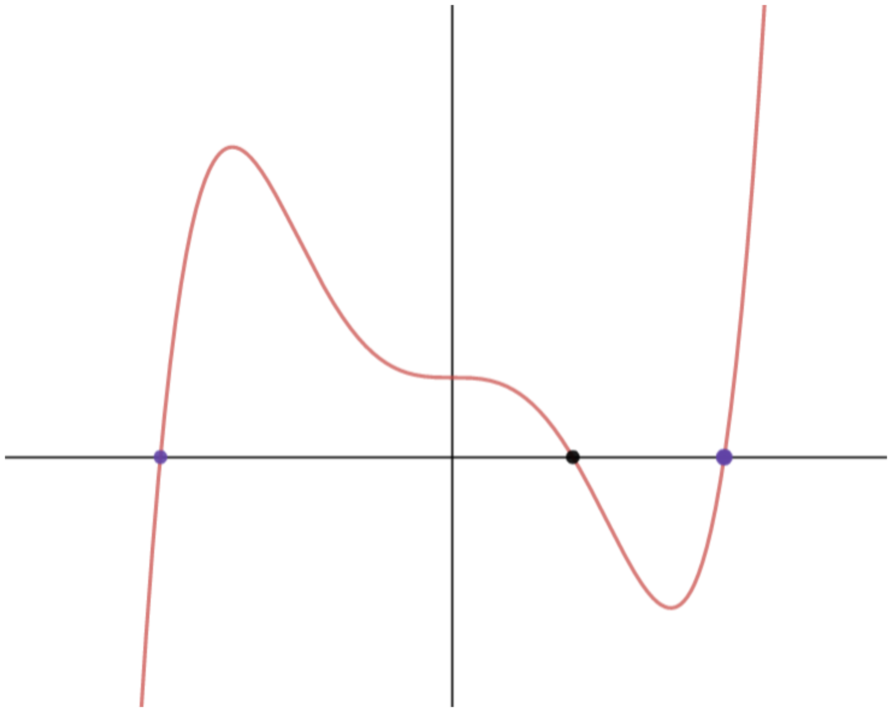
- LP builds of the foundations of
  - Nonlinear Programming
  - Network Flow Optimization
  - Integer Programming
  - Robust Optimization
  - Stochastic Programming
  - Semidefinite Programming

Checkpoint – Since Lecture 4

- Linear systems:
  - Non-iterative solvers:
    - Gaussian elimination, Gauss-Jordan Elimination, LU Decomposition
  - Iterative solvers:
    - Jacobi, Gauss-Seidel
  - Eigenvalues and Eigenvectors: Power method
  - Conditions of Linear Systems
- Linear regression:
  - Squared loss
  - Solvers:
    - Direct solver
    - GD/SGD
- Optimization:
  - GD/SGD
  - Linear programming

Nonlinear equation solver

- Problem statement: find a root of $F(x) = 0$ within an interval $[a, b]$
  - where F is a continuous nonlinear function
  - Discussion: any ideas?



Nonlinear equation solver: Bisection method

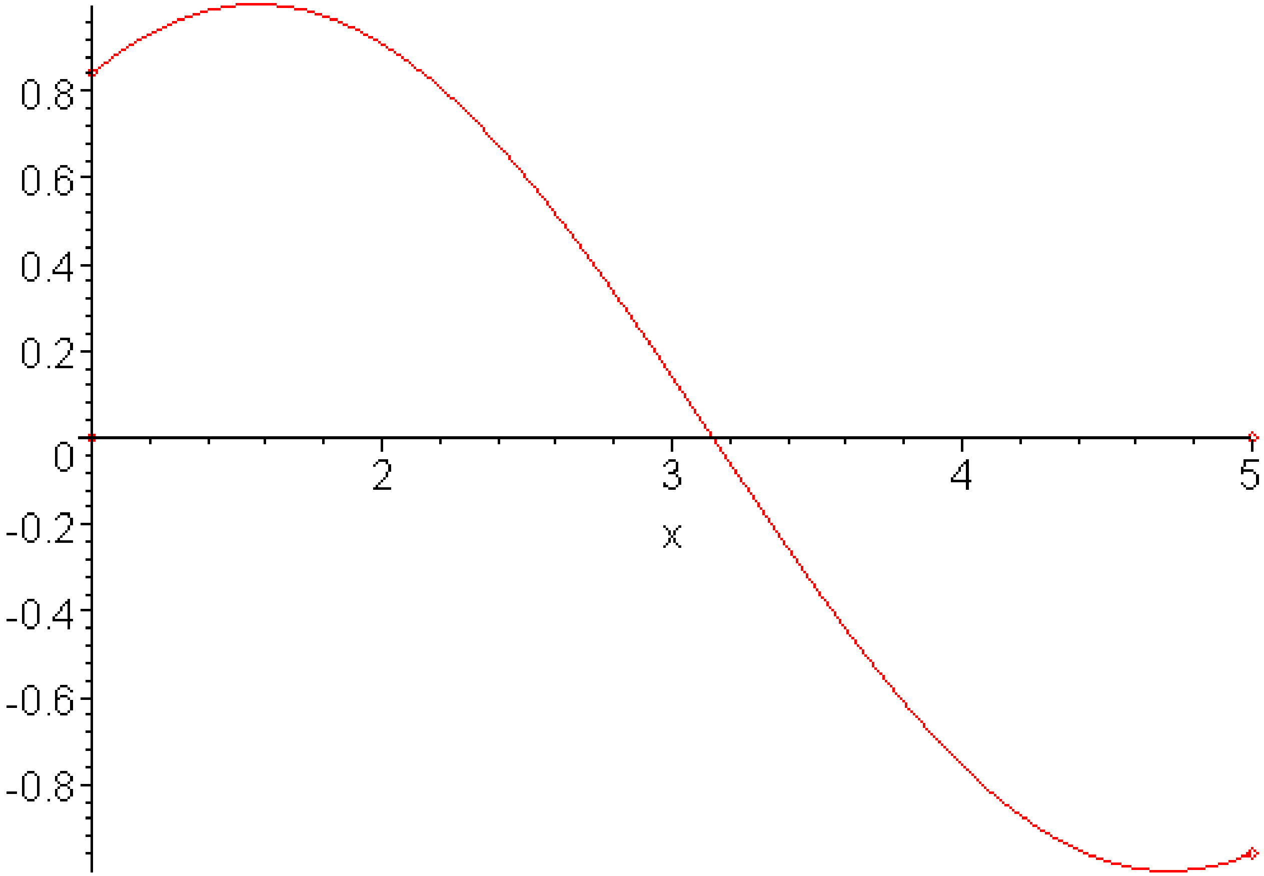
- Key idea:
  - In every iteration, we cut the interval in half while still maintaining the property that the endpoints have opposite signs. This allows us to conclude that we're getting closer and closer to a root.
- Algorithm:

1. Preprocessing: If $F(a) = 0$ or $F(b) = 0$, output whichever one was 0 and terminate. If $F(a) < 0 < F(b)$, then set $inc = 1$. Otherwise, set $inc = 0$.
2. Compute $z = \frac{a+b}{2}$, the midpoint of the interval $[a, b]$.
3. If $F(z) = 0$, return z and terminate.
4. If $inc = 0$ (so $F(a) > 0 > F(b)$):
   (a) If $F(z) < 0$, then set $b = z$.
   (b) If $F(z) > 0$, then set $a = z$.
5. If $inc = 1$ (so $F(a) < 0 < F(b)$):
   (a) If $F(z) < 0$, then set $b = z$.
   (b) If $F(z) > 0$, then set $a = z$.

After k iterations, we output the midpoint of the resulting interval.

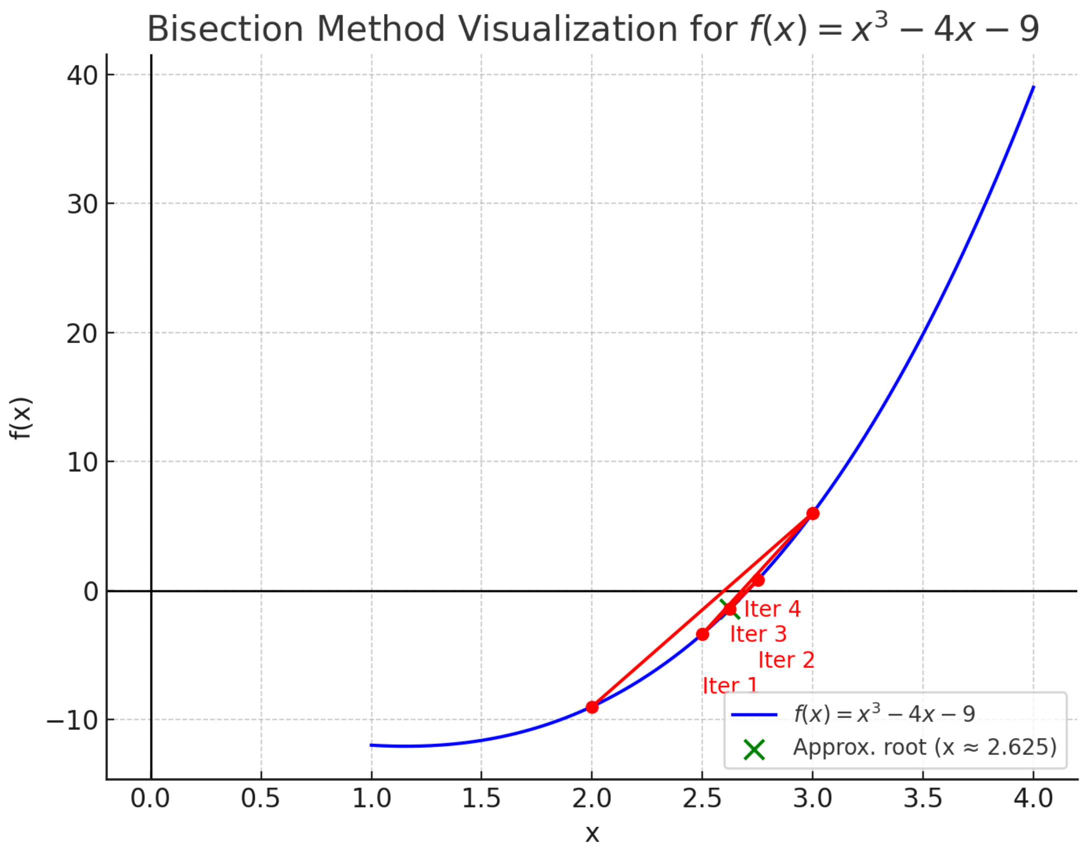
Illustration of the bisection method

- Initial interval: [1, 5]
- 3 steps in each iteration:
  - Given a, b, find midpoint
  - Check midpoint value
  - Update a or b



In-class exercise – calculators needed

- Use the bisection method to find a root of the equation
  - $f(x) = x^3 - 4x - 9 = 0$
  - in the interval $[2, 3]$, 3 iterations.
- Solutions:
  - Check $f(2) = -9$ and $f(3) = 6$, so there is a solution in $[2, 3]$
  - Iteration 1: $mid = 2.5$, $f(2.5) = -3.375$, new interval $[2.5, 3]$
  - Iteration 2: $mid = 2.75$, $f(2.75) = 0.7969$, new interval $[2.5, 2.75]$
  - Iteration 3: $mid = 2.625$, $f(2.625) = -1.43$, new interval $[2.625, 2.75]$
  - Output: 2.6875
- True root ≈ 2.706



Convergence of bisection method

- Key idea in analysis:
  - Let us call the initial interval $[a, b]$. After every iteration, the interval $[a, b]$ is cut in half. Thus, the length of the k-th interval is given by $l_k = \frac{b-a}{2^k}$
- **Theorem:**
  - There exists some root x of F between a and b such that the output m of the algorithm satisfies
    - $|m - x| \leqslant \frac{b-a}{2^{k+1}}$
  - In other words, if we fix some desired accuracy ϵ > 0, then
    - $|m - x| \leqslant \epsilon$
  - provided that $k \geqslant \log_2\left(\frac{b-a}{\epsilon}\right) - 1$
  - Since the absolute error is divided by a positive constant in every iteration, we say that the bisection search converges linearly to the solution.