Plan for today

- Nonlinear Optimization: Algorithms
  - Unconstrained Single Var
  1. Bisection Search
  2. Newton Search

Announcements:
- Exam 2 Review Problems posted

- Integer Programming
  - Power of integer variables in formulations

NLP: ALGORITHMS

… where we see algorithms to solve single variable unconstrained nonlinear optimization problems

MOTIVATIONS

… why do we even need algorithms?
Why can't we use the theory?

Motivations for algorithm

- Suppose we want to solve $\max f(x)$

  where $f(x)$ is a differentiable concave function
- If we can find a point x^* such that $f'(x^*) = 0$, then it must be a global maximum

- We may not have an easy way to find such a stationary point
  - If $f(x)$ is a complicated function, there may not be an easy closed form expression for x^*
  - E.g., consider $f(x) = xe^{1-x} - x^6$
  - Is this function concave?

Is this function concave?

$$f(x) = xe^{1-x} - x^6$$

$$f'(x) = e^{1-x}(1-x) - 6x^5$$
$$f''(x) = e^{1-x}(x-2) - 30x^4$$

Motivations for algorithm

- Suppose we want to solve $\max f(x)$

  where $f(x)$ is a differentiable concave function
- If we can find a point x^* such that $f'(x^*) = 0$, then it must be a global maximum

- We may not have an easy way to find such a stationary point
  - If $f(x)$ is a complicated function, there may not be an easy closed form expression for x^*
  - E.g., consider $f(x) = xe^{1-x} - x^6$
    - This function is concave

$$f'(x) = e^{1-x}(1 - x) - 6x^5$$

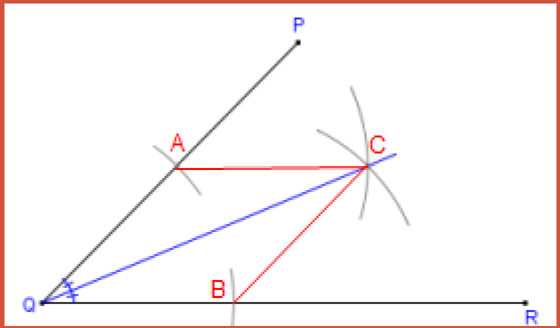
Question: How do we find a root of $f'(x) = 0$?

We will find a value x that is numerically close to the value of the root

NLP: UNCONSTRAINED, SINGLE VAR

How to find x such that $f'(x) = 0$?

1. Bisection search



One-Var Unconstrained Optimization

- **Idea:** Suppose we have located two points x_L and x_U (with $x_L < x_U$) such that:
  - $f'(x) > 0$ at $x = x_L$
  - $f'(x) < 0$ at $x = x_U$

- Since $f(x)$ is differentiable, there must be a point x^* in the interval (x_L, x_U) such that $f'(x) = 0$ at $x = x^*$

- Question: How can we find this x^*?

Trap x^* within a narrow interval

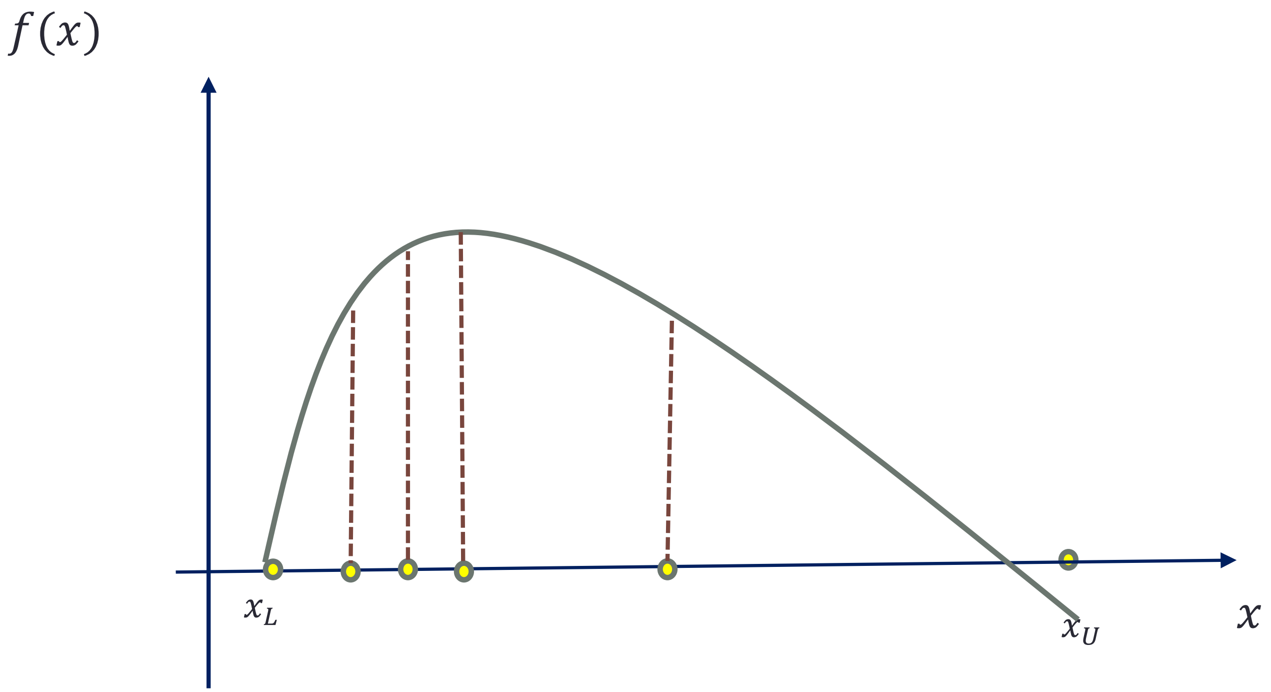
Bisection Search

- In each iteration, we reduce the width of our interval by half
  - Repeat this process until the width is sufficiently narrow
  - If width is sufficiently narrow, then we should be "close enough" to x^*

Bisection Search Algorithm:

- **Initialization:** Choose some $\epsilon > 0$ and identify initial value of x_L and x_U such that $f'(x_L) > 0$ and $f'(x_U) < 0$
- **Iteration:**
  1. Let $x_m = \frac{x_L + x_U}{2}$; we have 3 cases:
     1. If $f'(x_m) > 0$, set $x_L = x_m$
     2. If $f'(x_m) < 0$, set $x_U = x_m$
     3. If $f'(x_m) = 0$, set $x^* = x_m$ and terminate
  2. If $x_U - x_L \leq 2\epsilon$, then return $x_m = \frac{x_L + x_U}{2}$, which must be within a distance of ϵ from x^*; otherwise return to step 1

Bisection Search: Geometric Visualization



Bisection Search: Numerical Example

$$x_m := \frac{x_L + x_U}{2}$$

1. If $f'(x) > 0$ at x_m, set $x_L = x_m$
2. If $f'(x) < 0$ at x_m, set $x_U = x_m$
3. If $f'(x) = 0$ at x_m, set $x^* = x_m$ and terminate

- $\max f(x) = -4x^4 - 5x^2 + 3x$
- $f'(x) = -16x^3 - 10x + 3$
- $f''(x) = -48x^2 - 10$

- Note that $f''(x) < 0$ for all x and hence $f(x)$ is concave

- For this example, we will choose $\epsilon = 0.01, x_L = 0, x_U = 1$

Iteration	x_L	x_U	x_m	$f'(x_m)$	$\frac{x_U - x_L}{2}$
1	0	1	0.5	-4	0.5

Bisection Search: Numerical Example

$$x_m := \frac{x_L + x_U}{2}$$

1. If $f'(x) > 0$ at x_m, set $x_L = x_m$
2. If $f'(x) < 0$ at x_m, set $x_U = x_m$
3. If $f'(x) = 0$ at x_m, set $x^* = x_m$ and terminate

Iteration	x_L	x_U	x_m	$f'(x_m)$	$\frac{x_U - x_L}{2}$
1	0	1	0.5	-4	0.5
2	0	0.5	0.25	0.25	0.25
3	0.25	0.5	0.375	-1.59375	0.125
4	0.25	0.375	0.3125	-0.61328	0.0625
5	0.25	0.3125	0.28125	-0.16846	0.03125
6	0.25	0.28125	0.265625	0.04388	0.015625
7	0.265625	0.28125	0.273438	-0.06149	0.007813

↑ x_m is sufficiently close to the true x^*

Bisection Search

- Finding initial lower and upper bounds (x_L, x_U):
  - Choose some current point x_{curr}
  - If $f'(x_{curr}) > 0$, then x_{curr} is a lower bound, so set $x_L = x_{curr}$
  - To find an upper bound, choose some $\alpha > 0$ and find $f'(x_{curr} + \alpha)$
  - If $f'(x_{curr} + \alpha) < 0$, then $x_U = x_{curr} + \alpha$, otherwise increase α and try again
  - Note: If $f'(x_{curr}) < 0$, then x_{curr} is an upper bound

    choose $\alpha < 0$ to search for a lower bound

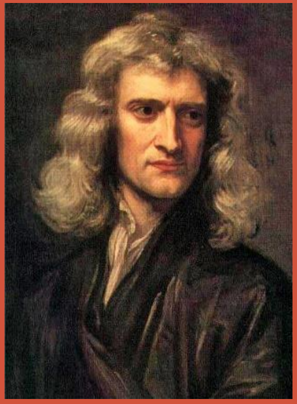
Bisection Search: Advantages and Limitations

- Pros:
  - Simple and easy to implement
  - Works even if the objective function is not necessarily concave, but unimodal
- Con:
  - The objective function needs to be differentiable

NLP: UNCONSTRAINED, SINGLE VAR

How to find x such that $f'(x) = 0$?

2. Newton Search



Motivations for algorithm

- Suppose we want to solve $\max f(x)$

  where $f(x)$ is a differentiable concave function
- If we can find a point x^* such that $f'(x^*) = 0$, then it must be a global maximum

- We may not have an easy way to find such a stationary point
  - If $f(x)$ is a complicated function, there may not be an easy closed form expression for x^*
  - E.g., consider $f(x) = xe^{1-x} - x^6$
    - This function is concave

$$f'(x) = e^{1-x}(1-x) - 6x^5$$
$$f''(x) = e^{1-x}(x-2) - 30x^4$$

Question: How do we find a root of $f'(x) = 0$?

We will find a value x that is numerically close to the value of the root

Newton Search: Overview

- Bisection search only uses information about the first derivative of $f(x)$
- Newton's method uses information about the second derivative as well
  - Creates a quadratic approximation of $f(x)$ around the current solution
  - Uses this quadratic approximation to approximate a new solution
  - If the new solution is "sufficiently close" to the previous solution, then the algorithm stops

Newton Search: Quadratic Approximation

- Suppose our current solution is x_i
  - A quadratic approximation of $f(x)$ near x_i is found by the Taylor series:
    $$f(x) \cong f(x_i) + f'(x_i)(x - x_i) + \frac{1}{2}f''(x_i)(x - x_i)^2$$
  - To find a new solution, take the derivative (wrt x) and set it to zero
    $$f'(x) \cong f'(x_i) + f''(x_i)(x - x_i) = 0$$
  - Hence, $x_{i+1} = x_i - \frac{f'(x_i)}{f''(x_i)}$ is the next solution
- Repeat this process until the new solution is x_{i+1} "close enough" to x_i

Newton Search: The Algorithm

Newton's Method Algorithm

- **Initialization:** Choose some $\epsilon > 0$, identify an initial solution x_0 and let $i = 0$
- **Iteration:**

1. Compute $x_{i+1} = x_i - \frac{f'(x_i)}{f''(x_i)}$
2. If $|x_{i+1} - x_i| < \epsilon$, then OUTPUT x_{i+1} and STOP
   Otherwise let $i = i + 1$ and go to step 1

Newton Search: Numerical Example

- $\max f(x) = -4x^4 - 5x^2 + 3x$
- $f'(x) = -16x^3 - 10x + 3$
- $f''(x) = -48x^2 - 10$

$$x_{i+1} := x_i - \frac{f'(x_i)}{f''(x_i)}$$

- Note that $f''(x) < 0$ and hence $f(x)$ is concave

- For this example, we will choose $\epsilon = 0.01$ and let $x_0 = 0$

Iteration	x_i	$f'(x_i)$	$f''(x_i)$	x_{i+1}	$\lvert x_{i+1} - x_i \rvert$
1	0	3	-10	0.3	0.3
2	0.3	-0.432	-14.32	0.26983	0.0301
3	0.26983	-0.01267	-13.4949	0.26889	0.0009

Newton Search: The Algorithm

Newton's Method Algorithm

- **Initialization:** Choose some $\epsilon > 0$, identify an initial solution x_0 and let $i = 0$
- **Iteration:**

1. Compute $x_{i+1} = x_i - \frac{f'(x_i)}{f''(x_i)}$
2. If $|x_{i+1} - x_i| < \epsilon$, then OUTPUT x_{i+1} and STOP
   Otherwise let $i = i + 1$ and go to step 1

Alternative interpretation:

We are trying to find the root of the function $g(x) \coloneqq f'(x)$

1. Start from some point x_0, pretend that $g(x)$ is proportional to x (linear in x), i.e.,

$$g(x) = g'(x_0)x + c$$

and ask: how much should we change x (moving from x_0 to x_1) to drive $g(x)$ to 0:

$$g(x_0) - 0 = g'(x_0)(x_0 - x_1), \text{ so } x_1 = x_0 - \frac{g(x_0)}{g'(x_0)}$$

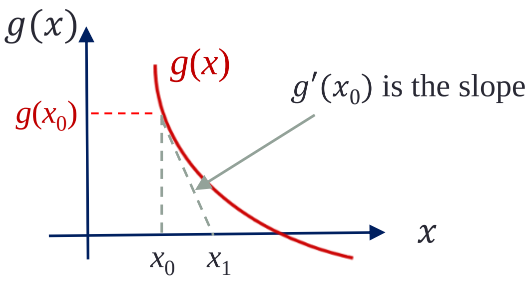
2. Take x_1 as the new x_0 and repeat until x_0 and x_1 are sufficiently close

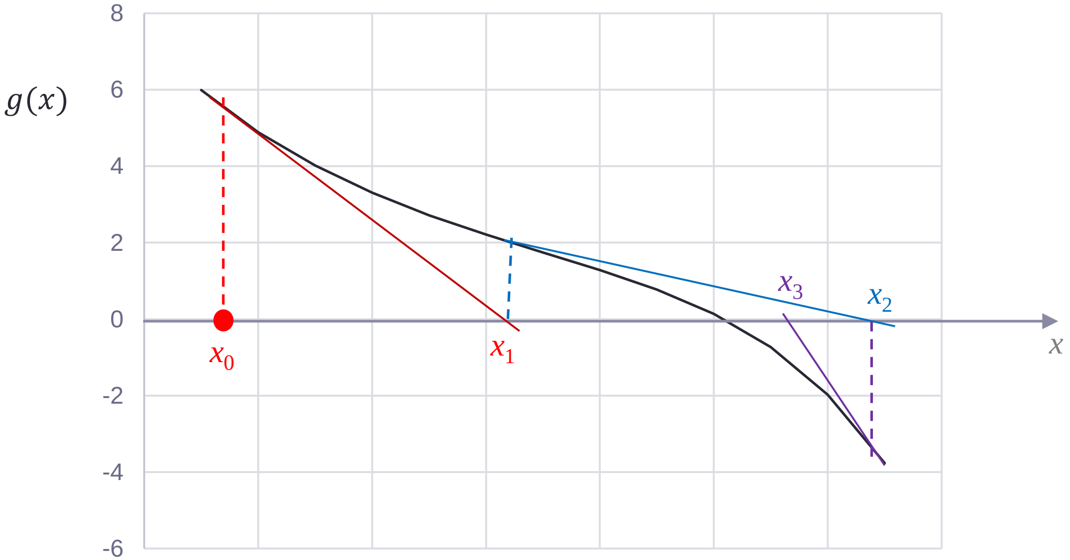
Newton Search: Geometric Visualization

Geometric description:

We want to find a root of $f'(x) = 0$. Let $g(x) = f'(x)$

1. From current point x_0, slide down/up the slope to the x axis
   - The slope is given by the gradient/derivative, $g'(x)$ at $x = x_0$
2. Use the intersection as the next point x_1, repeat 1 until the change in the x value in the next iteration is tiny

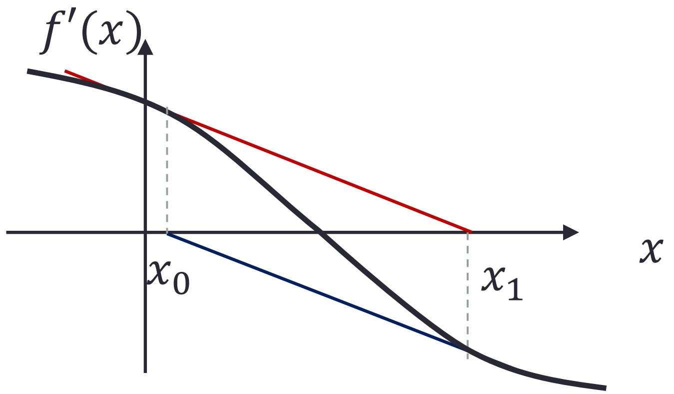




Newton's method is applicable to find a root of any function $g(x)$

Newton Search: Advantages and Limitations

- Pros:
  - Newton's method tends to converge faster than bisection search
  - Can you think of cases, where convergence to the root will be very fast?
- Cons:
  - To use the method for optimizing $f(x)$, we need $g'(x) = f''(x)$, which means that $f(x)$ needs to be twice differentiable
  - In some other cases, convergence is very slow or never happens (depends on the starting point)



$$\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots$$

INTEGER PROGRAMMING

… where we see the power of integer variables and algorithms to solve optimization problems involving integer variables

Linear Programming Formulation

- A Linear Programming Formulation is an optimization problem formulation in which
    - the objective function and
    - all constraints

  are linear functions of decision variables, which are real variables

- More specifically, in a linear programming, the objective function and all constraints satisfy the following four conditions
    - Proportionality
    - Additivity
    - Divisibility ← Recall: Each decision variable can take any real value (not necessarily integers) as long as all constraints are satisfied
    - Certainty

What if we have a linear program requiring variables to only take integer values?
Such formulations are known as integer programs

Integer Programming

An **integer programming (IP)** model is similar to a linear programming model, except all variables have to take integer values

- if all variables have to be either 0 or 1, the model is sometimes referred to **Binary IP (BIP)**
- if some variables have to be integers while others can take real values, the model is referred to **Mixed IP (MIP)**
- when both objective and constraints in an IP are linear, the IP model is also referred to as a **Integer Linear Program (ILP) or Mixed ILP (MILP)**
- If objectives or constraints are nonlinear then it is known as **non-linear IP**
- Non-linear IP is usually very difficult to solve (ILP is hard enough)

Linear Programming (LP)

$$\begin{gathered} \max\ Z = 3x_1 + 5x_2 \\ 3x_1 + 4x_2 \le 25 \\ x_1 + x_2 \le 20 \\ x_1 \ge 0, x_2 \ge 0 \end{gathered}$$

Integer Programming (IP)

$$\begin{gathered} \max\ Z = 3x_1 + 5x_2 \\ 3x_1 + 4x_2 \le 25 \\ x_1 + x_2 \le 20 \\ x_1, x_2 \text{ integers} \end{gathered}$$

Binary Integer Programming (BIP)

$$\begin{gathered} \max\ Z = 3x_1 + 5x_2 \\ 3x_1 + 4x_2 \le 25 \\ x_1 + x_2 \le 20 \\ x_1, x_2 \in \{0,1\} \end{gathered}$$

Mixed Integer Programming (MIP)

$$\begin{gathered} \max\ Z = 3x_1 + 5x_2 \\ 3x_1 + 4x_2 \le 25 \\ x_1 + x_2 \le 20 \\ x_1 \ge 0, x_2 \text{ integer} \end{gathered}$$

Recall:
- A function is **linear** if it satisfies additivity and proportionality
- Constraint $f(x) \le b_i$ is linear if f is linear

$$\begin{gathered} \max\ Z = 3x_1^2 + 5x_1x_2 \\ 3x_1 + 4x_2 \le 25 \\ x_1 + x_2 \le 20 \\ x_1 \ge 0, x_2 \text{ integer} \end{gathered}$$

Non-linear IP

POWER OF INTEGER VARIABLES

IP Formulations

… where we see the power of integer variables in mathematical modeling

Logical relationships, e.g., exclusion/inclusion

Examples:

1. Cake or Ice cream?

u_1: satisfaction of eating a cake, say $u_1 = 3$
u_2: satisfaction of eating ice cream, say $u_2 = 5$
Can eat at most one of the two
Goal: Maximize satisfaction

$$\max u_1 x_1 + u_2 x_2$$
$$x_1 + x_2 \leq 1$$
$$x_1, x_2 \in \{0,1\}$$

Logical relationship:
by saying yes to ice cream, you're saying no to cake (exclusion)

2. Course selection

Can select from *n* courses
First *m* courses are required core courses
Have to take at least three core courses
Cannot spend more than *h* hours per week
g_i: credit from course *i*
t_i: time to be spent per week on course *i*

Goal: maximize total credits

$$\max g_1 x_1 + g_2 x_2 + \cdots + g_n x_n$$
$$x_1 + x_2 + \cdots + x_m \geq 3$$
$$t_1 x_1 + t_2 x_2 + \cdots + t_n x_n \leq h$$
$$x_1, \ldots, x_n \in \{0,1\}$$

Logical relationship:
certain number of elements (3 core courses) must be a part of the solution (inclusion)

Fixed Investment Problem

B-Mobile is considering using exactly one of the two technologies to build a network

- LTE1 has a fixed cost $100k and
  can serve up to 1000 customers at a cost of $70 per customer
- LTE2 has a fixed cost $80k and
  can serve up to 2000 customers at a cost of $90 per customer
- each customer generates a revenue of $150

Question: how many customers to serve and by which technology

$y_i = 1$ if technology i is selected, $i = 1,2$
$x_i =$ number of customers served by technology i, $i = 1,2$

(obj. in $1000s)

$$\max 0.08\,x_1 + 0.06x_2 - 100y_1 - 80y_2$$
$$x_1 \le 1000y_1$$
$$x_2 \le 2000y_2$$
$$y_1 + y_2 = 1$$
$$x_1, x_2 \ge 0$$
$$y_1, y_2 \in \{0,1\}$$
x_1, x_2 integers

Sequential decisions

Example:

Suppose that you have \$20m to invest in two projects. The return from each is as follows:

Project 1: 1st installment (max \$7m): -3%, 2nd installment (max \$8m): 10%, 3rd installment (max \$5m): 5%

Project 2: 1st installment (max \$10m): 8%, 2nd installment (max \$10m): 4%

Fine-print: Investment will be counted towards next installment only if the previous installment's complete max is invested

Question: how much should you invest in each project in each installment?

What if you try to formulate this as a LP?

project 1
x_1: 1st installment, up to 7m
x_2: 2nd installment, up to 8m
x_3: 3rd installment, up to 5m
project 2
x_4: 1st installment, up to 10m
x_5: 2nd installment, up to 10m

$$\max Z = -0.03x_1 + 0.1x_2 + 0.05x_3 + 0.08x_4 + 0.04x_5$$
$$x_1 \le 7$$
$$x_2 \le 8$$
$$x_3 \le 5$$
$$x_4 \le 10$$
$$x_5 \le 10$$
$$x_1 + x_2 + x_3 + x_4 + x_5 \le 20$$
$$x_1, x_2, x_3, x_4, x_5 \ge 0$$

Optimum will be $x_1^* = 0, x_2^* = 8, x_3^* = 2, x_4^* = 10, x_5^* = 0$

Wait a min…