One-Dimensional Unconstrained Optimization I

Asst. Prof. Dr.-Ing. Sudchai Boonto

August 20, 2024

Department of Control System and Instrumentation Engineering
King Mongkut's University of Technology Thonburi
Thailand

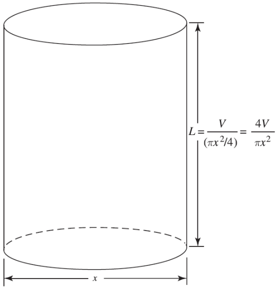
Objective

At the end of this chapter you should be able to:

- Mathematically define the optimality conditions for an unconstrained problem.
- Describe, implement, and use line-search-based methods.
- Explain the pros and cons of the various search direction methods.
- Understand the trust-region approachand how it contrasts with the line search approach.

Unconstrained optimization problems

Determine the objective function for building a minimum cost cylinderical refrigeration tank of volume 50 m^3, if the circular ends cost \$ 10 per m^2 , the cylindrical wall costs \$6 per m^2, and it costs \$80 per m^2 to refrigerate over the useful life.



$$
\begin{aligned}
f(x, L) &= (10)(2)\left(\frac{\pi x^2}{4}\right) \\
&\quad + (6)(\pi x L) + 80\left((2)\frac{\pi x^2}{4} + \pi x L\right) \\
&= 45\pi x^2 + 86\pi x L \\
L &= \frac{(50)(4)}{\pi x^2} = \frac{200}{\pi x^2} \\
f(x) &= 45\pi x^2 + \frac{17200}{x}
\end{aligned}
$$

One problem is minimize $f(x)$ for all real x. How!

Unconstrained optimization problems

We consider unconstrained optimization problems with continuous design variables,

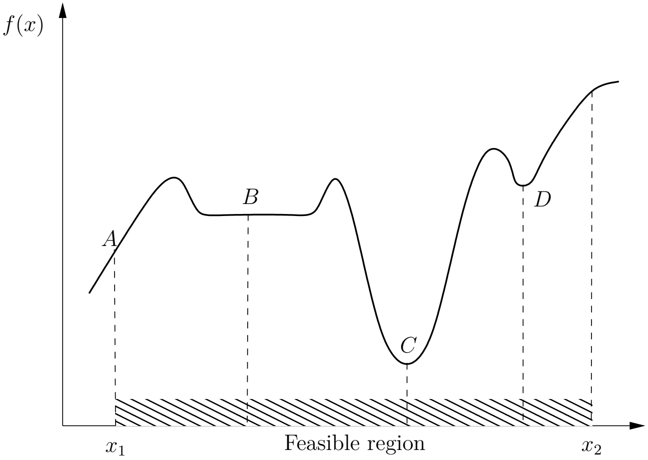
$$\underset{\mathbf{x}}{\text{minimize}}\, f(\mathbf{x}),$$

where $\mathbf{x} = [x_1, \ldots, x_n]$ is composed of the design variables that the optimization algorithm can change.

Minimum Points:

- the point $\mathbf{x}^*$ is a **weak local minimum** if there exists a $\delta > 0$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x}$ such that $|\mathbf{x} - \mathbf{x}^*| < \delta$, that is $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x}$ in a δ-neighborhood of $\mathbf{x}^*$.
- the point $\mathbf{x}^*$ is a **strong local minimum** if there exists a $\delta > 0$ such that $f(\mathbf{x}^*) < f(\mathbf{x})$ for all $\mathbf{x}$ such that $|\mathbf{x} - \mathbf{x}^*| < \delta$.
- $\mathbf{x}^*$ is a **global minimum** if $f(\mathbf{x}^*) < f(\mathbf{x})$ for all $\mathbf{x}$

Unconstrained optimization problems



Optimality Conditions

A point $\mathbf{x}^*$ is a local minimum if $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x}$ in the neighborhood of $\mathbf{x}^*$. A second-order Taylor series expansion about $\mathbf{x}^*$ for small steps of size $\mathbf{p}$ yields

$$f(\mathbf{x}^* + \mathbf{p}) = f(\mathbf{x}^*) + \nabla f(\mathbf{x}^*))^T \mathbf{p} + \frac{1}{2}\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} + \dots$$
$$f(\mathbf{x}^* - \mathbf{p}) = f(\mathbf{x}^*) - \nabla f(\mathbf{x}^*))^T \mathbf{p} + \frac{1}{2}\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} + \dots$$

For x^* to be an optimal point, we must have $f(\mathbf{x}^* + \mathbf{p}) \geq f(\mathbf{x}^*)$ for all $\mathbf{p}$. This implies that

$$\nabla f(\mathbf{x}^*)^T \mathbf{p} + \frac{1}{2}\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} \geq 0 \text{ and } -\nabla f(\mathbf{x}^*)^T \mathbf{p} + \frac{1}{2}\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} \geq 0 \qquad \forall \mathbf{p}.$$

The magnitude of $\mathbf{p}$ is small, the second term can be neglected. Therefore, we require that

$$\nabla f(\mathbf{x}^*)^T \mathbf{p} \geq 0 \text{ and } -\nabla f(\mathbf{x}^*)^T \mathbf{p} \geq 0 \quad \Longrightarrow \nabla f(\mathbf{x}^*) = 0$$

Optimality Conditions

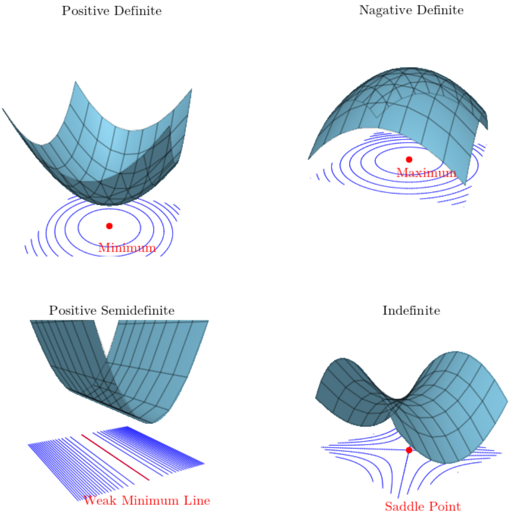
The condition $\nabla f(\mathbf{x}^*) = 0$ is called the first-order necessary optimality condition.

Because the gradient term has to be zero, we must satisfy the remaining term in the previous inequality, that is

$$\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p} \geq 0 \quad \forall \mathbf{p} \quad \text{or } \mathbf{H}(\mathbf{x}^*) \succeq 0$$

- These two conditions $\nabla f(\mathbf{x}^*) = 0$ and $\mathbf{H}(\mathbf{x}^*) \succeq 0$ are necessary conditions for a local minimum but not sufficient.
- In some direction $\mathbf{p}^T \mathbf{H}(\mathbf{x}^*)\mathbf{p}$ can be zero. We need to check the third-order term. If it is a minimum, it is a weak minimum.
- To have the sufficient optimality condition, $\mathbf{H}(\mathbf{x}^*)$ must be positive definite.

Quadratic function with different types of Hessians



Optimality Conditions

Consider the following function of two variables:

$$f(x_1, x_2) = 0.5x_1^4 + 2x_1^3 + 1.5x_1^2 + x_2^2 - 2x_1x_2$$

Let the gradient equal to zero,

$$\nabla f(x_1, x_2) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \\ \frac{\partial f}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1^3 + 6x_1^2 + 3x_1 - 2x_2 \\ 2x_2 - 2x_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

From the second row, we have $x_1 = x_2$. Substituting this in to the first equation yields

$$x_1 \left(2x_1^2 + 6x_1 + 1\right) = 0 \quad \Longrightarrow \quad x_1 = 0, -2.8223, -0.1771$$

The solution of this equation has three points: $x_A = (0, 0)$, $x_B = (-2.8223, -2.8223)$, and $x_C = (-0.1771, -0.1771)$. `(see ch2/optimal_condition.jl)`

Optimality Conditions

To clarify these three points, we need to find the Hessian matrix.

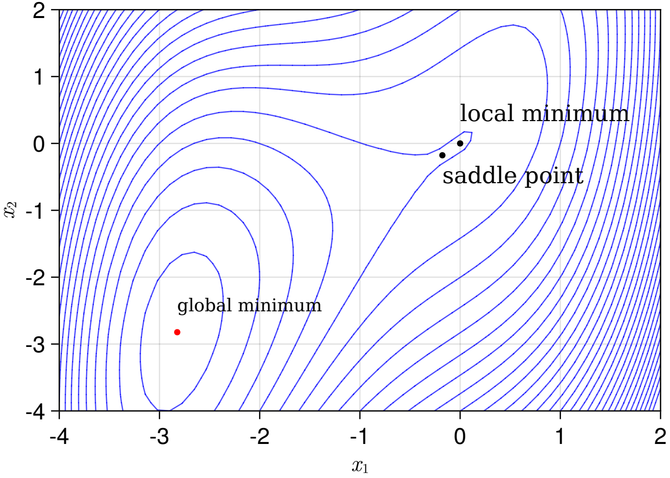
$$\mathbf{H}(x_1, x_2) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} 6x_1^2 + 12x_1 + 3 & -2 \\ -2 & 2 \end{bmatrix}$$

For each point, we have

$$\mathbf{H}(\mathbf{x}_A) = \begin{bmatrix} 3 & -2 \\ -2 & 2 \end{bmatrix}, \qquad \mathbf{H}(\mathbf{x}_B) = \begin{bmatrix} 16.9373 & -2 \\ -2 & 2 \end{bmatrix}, \qquad \mathbf{H}(\mathbf{x}_C) = \begin{bmatrix} 1.0627 & -2 \\ -2 & 2 \end{bmatrix}$$

The eigenvalues are $\lambda_A = (0.438, 4.561)$, $\lambda_B = (1.737, 17.200)$, and $\lambda_C = (-0.523, 3.586)$, respectively. The first two eigenvalues show the evidence of the local minimum points, while the last one addresses the saddle point.

Optimality Conditions



Optimality Conditions

We want to

$$\underset{x}{\text{minimize}} \qquad 45\pi x^2 - \frac{17200}{x}$$

We set

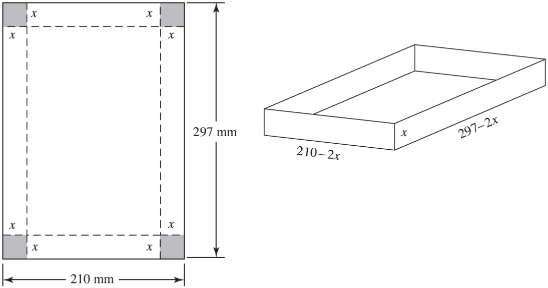
$$\nabla f = 90\pi x - \frac{17200}{x^2} = 0 \qquad \Longrightarrow \qquad x^3 = \frac{17200}{90\pi} = 60.833$$

We have $x = 3.93$ m and $L = 200/(\pi x^2) = 4.12$ m.
The cost is

$$f(x^*) = 45(x^*)^2 + \frac{17200}{x^*} = 6560$$

Since $\mathbf{H}(x^*) = 90\pi + (3(17200))/(x^*)^3 = 1132.85$, it is strictly positive. Thus the solution is a strict or strong minimum.

Determine the dimensions of an open box of maximum volume that can be constructed form an A4 sheet 210 mm $\times$ 297 mm by cutting four squares of side x from the corners and folding and gluing the edges as shown in Fig.



Volume $V = (297 - 2x)(210 - 2x)x$

Optimality Conditions

The problem is to

$$\underset{x}{\text{maximize}} \qquad V(x) = (297 - 2x)(210 - 2x)x = 62370x - 1014x^2 + 4x^3$$

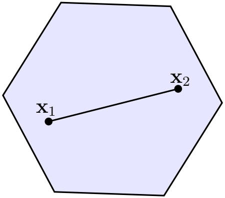
We set $f(x) = -V(x) = -62370x + 1014x^2 - 4x^3$. Setting $\nabla f(x) = 0$, we get

$$\nabla f(x) = -62370 + 2028x - 12x^2 = 0 \quad \Longrightarrow \quad x = 40.423 \text{ and } 128.577 \text{ mm.}$$

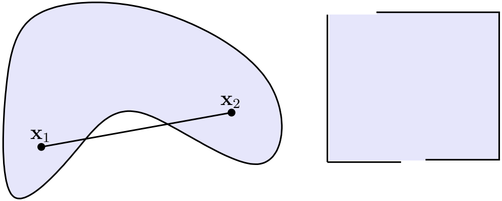
The possible solution of x is only the first one, where $x^* = 40.423$ mm. The $\mathbf{H}(x^*) = 2028 - 24x^* = 1057.848 > 0$. implies that x^* is a strict minimum of $f(x)$ or maximum of $V(x)$.
The maximum value of the box is 1128.5 cm^3.

Convex Sets

- A set $\mathbb{S}$ is called a convex set if for any two points in the set, every point on the line joining the two points is in the set.
- Alternatively, the $\mathbb{S}$ is convex if for every pair of points $\mathbf{x}_1$ and $\mathbf{x}_2$ in $\mathbb{S}$, and every α such that $0 < \alpha < 1$, the point $\alpha \mathbf{x}_1 + (1-\alpha)\mathbf{x}_2$ is in $\mathbb{S}$.
- example of convex sets:
  - the set of all real numbers $\mathbb{R}$ is a convex set.
  - any closed interval of $\mathbb{R}$ is also a convex set.
  - $\mathbf{A} = \{x \in \mathbb{R} : 0 \leq x \leq 1\}$, $\mathbf{B} = \{x \in \mathbb{R} : 2 \leq x \leq 3\}$ and $\mathbb{S} = \mathbf{A} \cup \mathbf{B}$. $\mathbb{S}$ is not a convex set.



convex set



nonconvex set

Convex Functions

- A function $f(\mathbf{x})$ defined over a convex set $\mathbb{R}_c$ is said to be convex if for every pair of points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}_c$ and every real number $0 \leq \alpha \leq 1$, the inequality

$$f(\alpha \mathbf{x}_1 + (1-\alpha)\mathbf{x}_2) \leq \alpha f(\mathbf{x}_1) + (1-\alpha) f(\mathbf{x}_2)$$

hold. If $x_1 \neq x_2$ and

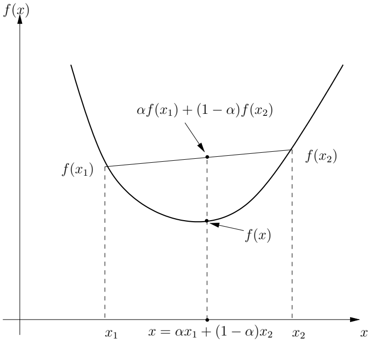
$$f(\alpha \mathbf{x}_1 + (1-\alpha)\mathbf{x}_2) < \alpha f(\mathbf{x}_1) + (1-\alpha) f(\mathbf{x}_2)$$

then $f(\mathbf{x})$ is said to be strictly convex.

- If $\psi(\mathbf{x})$ is defined over a convex set $\mathbb{R}_c$ and $f(\mathbf{x}) = -\psi(\mathbf{x})$ is convex, then $\phi(\mathbf{x})$ is said to be concave. If $f(\mathbf{x})$ is strictly convex, $\psi(\mathbf{x})$ is strictly concave.

is located in $\mathbb{R}_c$

Properties Convex Functions



Properties of Convex Functions

- If f has continuous first derivatives then f is convex over a convex set $\mathbf{S}$ if and only if for every x and y in $\mathbf{S}$, $f(y) \geq f(x) + f'(x)(y-x)$ This means that the graph of the function lies above the tangent line drawn at point show in above figure.

Properties Convex Functions

- If f has continuous second derivatives then f is convex over a convex set $\mathbf{S}$ if and only if for every x in $\mathbf{S}$,

$$f''(x) \geq 0$$

- If $f(x^*)$ is a local minimum for a convex function f on a convex set $\mathbf{S}$, then it is also a global minimum.
- If f has continuous first derivatives on a convex set $\mathbf{S}$ and for a point x^* in $\mathbf{S}$, $f'(x^*)(y - x^*)|geq0$ for every y in $\mathbf{S}$, then x^* is a global minimum point of f over $\mathbf{S}$.

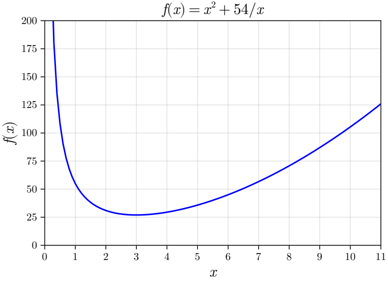
Example

Prove that $f = |x|, x \in \mathbb{R}^1$, is a convex function. Using the triangular inequality $|x + y| \leq |x| + |y|$, we have, for any two real numbers x_1 and x_2 and $0 < \alpha < 1$,

$$\begin{aligned} f(\alpha x_1 + (1-\alpha)x_2) &= |\alpha x_1 + (1-\alpha)x_2| \leq \alpha|x_1| + (1-\alpha)|x_2| \\ &\leq \alpha f(x_1) + (1-\alpha)f(x_2) \end{aligned}$$

Unimodality

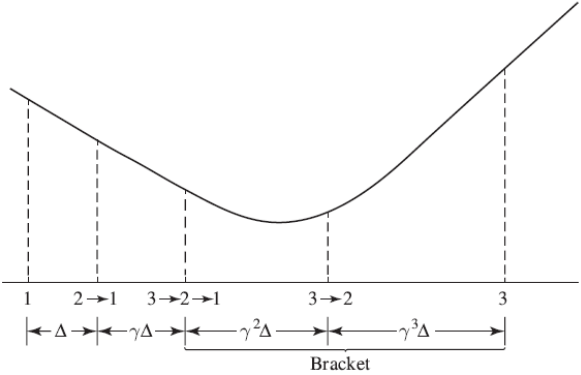
- Several of the algorithms assume unimodality of the objective function.
- A unimodal function f is one where there is a **unique** x^*, such that f is monotonically decreasing for $x \leq x^*$ and monotonically increasing for $x \geq x^*$.
- It follows from this definition that the unique global minimum is at x^* , and there are no other local minima.
- Given a unimodal function, we can bracket and inter $[a, c]$ containing the global minimum if we can find three points $a < b < c$, such that $f(a) > f(b) < f(c)$.



Finding an Initial Bracket

- When optimizing a function, we often start by first bracketing and interval containing a local minimum.
- After that, we then successively reduce the size of the bracketed interval to converge on the local minimum.
- We choose a starting point 1 with coordinate x_1 and a step size Δ in the positive direction. The distance we take is a *hyperparameter* to this algorithm. The step size Δ is 1×10^{-2}.
- We than search in the downhill direction to find a new point that exceeds the lowest point. With each step, we axpand the step size by some factor, which is another hyperparameter to the to this algorithm that is often set to $\gamma = 2$.

Finding the Initial Bracket cont.



Finding the Initial Bracket cont.

Bracketing Algorithm / Three-Point Pattern

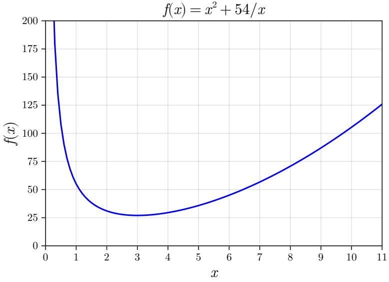
1. Set $x_2 = x_1 + \Delta$
2. Evaluate f_1 and f_2
3. If $f_2 \leq f_1$ Goto Step 5
4. Else Interchange f_1 and f_2 and x_1 and x_2, and Set $\Delta = -\Delta$
5. Set $\Delta = \gamma\Delta$, $x_3 = x_2 + \Delta$, and Evaluate f_3 at x_3
6. If $f_3 > f_2$ Goto Step 8
7. Else Rename f_2 as f_1, f_3 as f_2, x_2 as x_1, x_3 as x_2, Goto Step 5
8. Point $1, 2,$ and 3 satisfy $f_1 \geq f_2 < f_3$ (three-point pattern)

Finding the Initial Bracket cont.

Consider the problem:

$$\underset{x}{\text{minimize}} \qquad x^2 + \frac{54}{x}$$

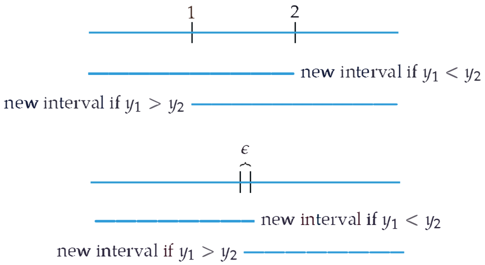
in the interval $(0, 5)$.



Using a algorithm above we have the interval $(1.28, 5.12)$ by using $\Delta = 1e - 2, \gamma = 2$. The interval guarantees that the minimum point lies in the interval.

Fibonacci Search

If we have a unimodal f bracketed by the interval $[a, b]$. Given a limit on the number of times we can query the objective function. *Fibonacci search* is guaranteed to maximally shrink the bracketed interval.

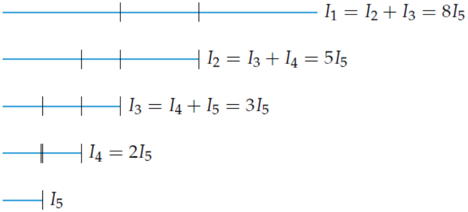


With three queries, we can shrink the interval by a factor of three. We first query f on the one-third and two-third points on the interval, discard one-third of the interval, and then sample just next to the better sample.

Fibonacci Search cont.

For n queries, the interval lengths are related to the Fibonacci sequence: $1, 1, 2, 3, 5, 8, \ldots$. The first two terms are one, and the following terms are always the sum of the previous two:

$$F_n = \begin{cases} 1, & \text{if } n \leq 2 \\ F_{n-1} + F_{n-2}, & \text{otherwise} \end{cases}$$



$$I_{n-j} = F_{j+1} I_n \qquad j = 1, 2, \ldots, n-1$$

For $j = n-1$ and $n-2$,

$$I_1 = F_n I_n \quad I_2 = F_{n-1} In$$

$$I_n = \frac{I_1}{F_n} \implies \frac{I_n}{I_1} = \frac{1}{F_n}$$

$$I_2 = \frac{F_{n-1}}{F_n} I_1$$

Fibonacci Search cont.

Consider the interval $[0, 1]$, and number of trials $n = 5$.

$$F_0 = 1, F_1 = 1, F_2 = 2, F_3 = 3, F_4 = 5, F_5 = 8, ...$$

- We have $I_2 = \frac{F_4}{F_5} I_1 = \frac{5}{8}$ and $I_3 = \frac{F_3}{F_4} I_2 = \frac{3}{5}\frac{5}{8} I_1 = \frac{3}{8}$
- The new interval is $[0, \frac{3}{8}, \frac{5}{8}, 1]$. The new interval will be either $[0, \frac{5}{8}]$ or $[\frac{3}{8}, 1]$. If the result is left hand side we have $[0, \frac{5}{8}]$
- Set four points we have $[0, \frac{2}{8}, \frac{3}{8}, \frac{5}{8}]$, then we have $[0, \frac{2}{8}, \frac{3}{8}]$ and again set four point $[0, \frac{1}{8}, \frac{2}{8}, \frac{3}{8}]$
- We have $[0, \frac{1}{8}, \frac{1}{8}, \frac{2}{8}]$. The central points coincide thus we should add a small number $\epsilon = 1e - 2$ or less. Then the interval is $[0, \frac{1}{8}, \frac{1}{8} + \epsilon, \frac{2}{8}]$ The final stage should be either $[0, \frac{1}{8} + \epsilon]$ or $[\frac{1}{8}, \frac{2}{8}]$.
- Since $n = 5$, we have $\frac{I_5}{I_1} = \frac{1}{8} = \frac{1}{F_5}$

Fibonacci Search cont.

The Fibonacci sequence can be determined analytically using *Binet's formula:*

$$F_n = \frac{\varphi^n - (1-\varphi)^n}{\sqrt{5}},$$

where $\varphi = (1+\sqrt{5})/2 \approx 1.61803$ is the *golden ratio.*
The ratio between successive values in the Fibonacci sequence is:

$$\frac{F_n}{F_{n-1}} = \varphi \frac{1 - s^{n+1}}{1 - s^n},$$

where $s = (1-\sqrt{5})/(1+\sqrt{5}) \approx -0.382$.

Fibonacci Search Algorithm

Fibonacci Search

1. **Set** the interval $[a, b]$ and the number of interval reductions n
2. **If** ϵ is given find the smallest n such that $\frac{1}{F_n} < \epsilon$.
3. **Set** $\varphi = 1.61803$, $s = (1 - \sqrt{5})/(1 + \sqrt{5})$, $\rho = 1/(\varphi(1 - s^{n+1}))/(1 - s^n))$
4. **Set** $d = \rho b + (1 - \rho)a$
5. **Set** $y_d = f(d)$
6. **For** i **In** 1 **To** $n - 1$
7.   **If** $i == n - 1$
8. $c = \epsilon a + (1 - \epsilon)d$
9.   **Else**
10. $c = \rho a + (1 - \rho)b$
11.   **EndIf**
12.   **Set** $y_c = f(c)$
13.   **If** $y_c < y_d$
14. $b, d, y_d = d, c, y_c$
15.   **Else**
16. $a, b = b, c$
17.   **EndIf**
18.   **Set** $\rho = 1/(\varphi(1 - s^{n-i+1}))/(1 - s^{n-i})$
19. **EndFor**
20. **Return** $a < b \; ? \; (a, b) : (b, a)$

Fibonacci Search cont.

In the interval reduction problem, the initial interval is given to be 4.68 units. The final interval desired is 0.01 units. Find the number of interval reductions using Fibonacci method

Solution: We need to choose the smallest n such that

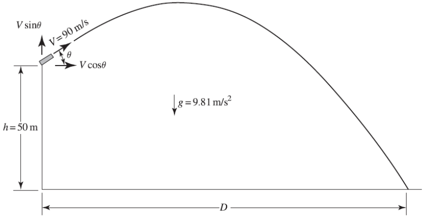
$$\frac{1}{F_n} < \frac{0.01}{4.68} \text{ or } F_n > 468$$

The Fibonacci sequence is $1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610$, we get $n = 14$. The number of interval reductions is $n - 1 = 13$.

Note: Write a program to check the result by yourself.

A projectile released from a height h at an angle θ with respect to the horizontal in a gravitational field g, shown in Fig. travels a distance D when it hits the ground. D is given by

$$D = \left(\frac{V \sin\theta}{g} + \sqrt{\frac{2h}{g} + \left(\frac{V \sin\theta}{g}\right)^2} \right) V \cos\theta$$

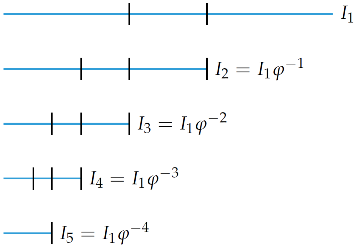


If $h = 0.5$ m, $V = 90$ m/s, $g = 9.81$ m/s^2, determine the angle θ in degrees for which the distance D is a maximum. Also calculate the maximum distance D in meters. Using the range for θ of $0°$ to $80°$ and compare your results for 7 and 19 Fibonacci interval reductions. **Note:** We are going to minimize $V = -D$. (`See: Bracket.jl`)

Golden Section Search

If we take the limit of the Fibonacci Search for large n, we see that the ratio between successive values of the Fibonacci sequence approaches the golden ratio (`https://en.wikipedia.org/wiki/Golden_ratio`):

$$\lim_{n \to \infty} \frac{F_n}{F_{n-1}} = \varphi$$



Golden Section Search Algorithm

Golden Section Search

1. **Set** the interval $[a, b]$ and the number of interval reductions n
2. **If** ϵ is given find the smallest n such that $\frac{1}{F_n} < \epsilon$.
3. **Set** $\varphi = 1.61803$, $\rho = \varphi - 1$
4. **Set** $d = \rho b + (1 - \rho)a$
5. **Set** $y_d = f(d)$
6. **For** i **In** 1 **To** $n - 1$
7. $c = \rho a + (1 - \rho)b$
8.   **Set** $y_c = f(c)$
9.   **If** $y_c < y_d$
10. $b, d, y_d = d, c, y_c$
11.   **Else**
12. $a, b = b, c$
13.   **EndIf**
14. **EndFor**
15. **Return** $a < b\ ?\ (a, b) : (b, a)$

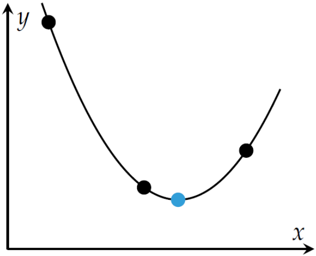
You can test the algorithm with the following function with 5 interval reductions:

$$f(x) = -e^{-x^2}$$

$$f(x) = (\sin(x) + \sin(x/2))/4$$

Quadratic Fit Search Algorithm

- **Quadratic fit search** gives our ability to analytically solve for the minimum of a quadratic function. Many local minima look quadratic when we zoom in close enough.
- Quadratic fit search iteratively fits a quadratic function to three bracketing points, solves for the minimum, chooses a new set of bracketing points, and repeats as shown in Figure below:



Given bracketing points $a < b < c$, we wish to find the coefficients p_1, p_2, and p_3 for the quadratic function q that goes through (a, y_a), (b, y_b), and (c, y_c):

$$q(x) = p_1 + p_2 x + p_3 x^2$$
$$y_a = p_1 + p_2 a + p_3 a^2$$
$$y_b = p_1 + p_2 b + p_3 b^2$$
$$y_c = p_1 + p_2 c + p_3 c^2$$

Quadratic Fit Search Algorithm cont.

In matrix form, we have

$$\begin{bmatrix} y_a \\ y_b \\ y_c \end{bmatrix} = \begin{bmatrix} 1 & a & a^2 \\ 1 & b & b^2 \\ 1 & c & c^2 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}, \qquad \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 & a & a^2 \\ 1 & b & b^2 \\ 1 & c & c^2 \end{bmatrix}^{-1} \begin{bmatrix} y_a \\ y_b \\ y_c \end{bmatrix}$$

The quadratic function is then

$$q(x) = y_a \frac{(x-b)(x-c)}{(a-b)(a-c)} + y_b \frac{(x-a)(x-c)}{(b-a)(b-c)} + y_c \frac{(x-a)(x-b)}{(c-a)(c-b)}$$

We can solve for the unique minimum by finding where the derivative is zero:

$$x^* = \frac{1}{2} \frac{y_a(b^2-c^2) + y_b(c^2-a^2) + y_c(a^2-b^2)}{y_a(b-c) + y_b(c-a) + y_c(a-b)}$$

Quadratic fit search is typically faster than golden section search. It may need safeguards for cases where the next point is very close to other points.

Quadratic Fit Search Algorithm

Quadratic Fit Search

1. Set n is a number of iteration.
2. Set $y_a, y_b, y_c = f(a), f(b), f(c)$
3. For i In 1 To $n-3$
4. Set

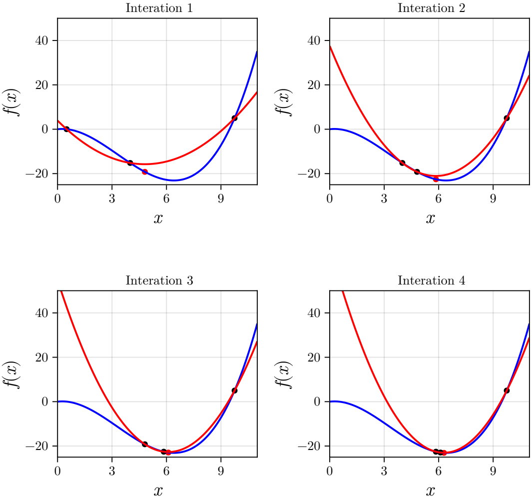
$$x = \frac{1}{2}\frac{(ya(b^2-c^2)+yb(c^2-a^2)+yc(a^2-b^2))}{(ya(b-c)+yb(c-a)+yc(a-b))}$$

$yx = f(x)$

5. If $x > b$
6. If $yx > yb$
7. $c, y_c = x, yx$
8. Else
9. $a, ya, b, yb = b, yb, x, yx$
10. EndIf

9. ElseIf $x < b$
10. If $yx > yb$
11. $a, ya = x, yx$
12. Else
13. $c, yc, b, yb = b, yb, x, yx$
14. EndIf
15. EndIf
16. EndFor
17. Return (a, b, c)

Quadratic Fit Search Algorithm



See `julia/ch3/Bracket.html`

Optimization of non-unimodal problem

- The techniques presented above, namely Fibonacci, Golden Section, and Polynomial fit method, require the function to be unimodal.
- However functions are multimodal and further, their modality cannot be ascertained a priori.
- Techniques for finding the global minimum are few, and can be broadly classified as based on deterministic or random search.
- We discuss some of them.

Shubert-Piyavskii Method

The **Shubert-Piyavskii method** is a *global optimization method* over a domain $[a, b]$, meaning it is guaranteed to converge on the global minimum of a function irrespective of any local minima or whether the function is unimodal.

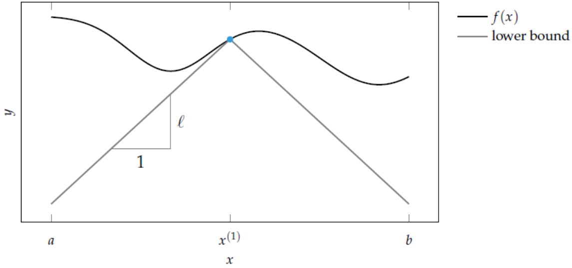
- The Shubert-Piyavskii method requires that the function be *Lipschitz continous*, meaning that it is continuous and there is an upper bound on the magnitude of its derivative. A function f is Lipshitz continuous on $[a, b]$ if there exists an $l > 0$ such that:

$$|f(x) - f(y)| \leq l|x - y| \quad \text{for all} \quad x, y \in [a, b]$$

  l is as large as the largest unsigned instantaneous rate of change the function attains on $[a, b]$.
- Given a point $(x_0, f(x_0))$, we knwo that the lines $f(x_0) - l(x - x_0)$ for $x > x_0$ and $f(x_0) + l(x - x_0)$ for $x < x_0$ form a lower bound of f.

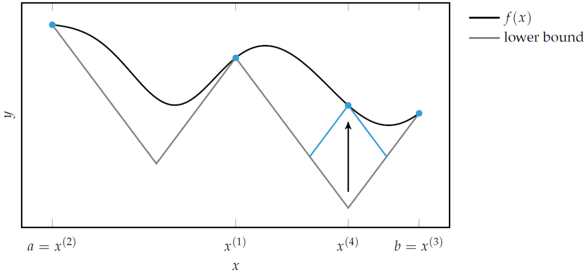
Shubert-Piyavskii Method

- The Shubert-Piyavskii method iteratively builds a tighter and tighter lower bound on the function.
- Given a valid Lipschitz constant l the algorithm begins by sampling the midpoint, $x^{(1)} = (a + b)/2$.
- A sawtooth lower bound is constructed using lines a slope $\pm l$ from this point.



- further iterations find the minimum point in the sawtooth, evaluate the function at that x value, and then use the result to update the sawtooth.

Shubert-Piyavskii Method



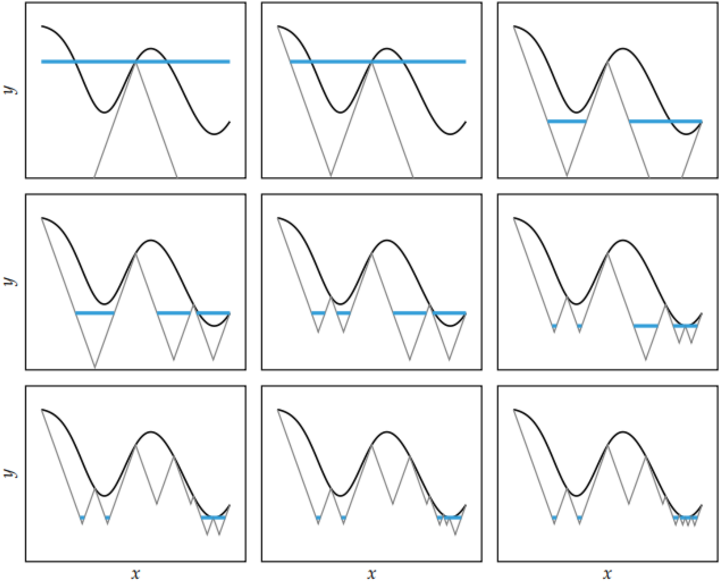
- The algorithm is stopped when the difference in height between the minimum sawtooth value and the function evaluation at that point is less than a given tolerance ϵ. For the minimum peak $(x^{(n)}, y^{(n)})$ and function evaluation $f(x^{(n)})$, we thus terminate if $y^{(n)} - f(x^{(n)}) < \epsilon$
- For every peak, an uncertainty region can be computed according to:

$$\left[x^{(i)} - \frac{1}{l}(y_{\text{min}} - y^{(i)}), x^{(i)} + \frac{1}{l}(y_{\text{min}} - y^{(i)})\right]$$

Shubert-Piyavskii Method

- The main drawback of the Shubert-Piyavskii method is that it requires knowing a valid Lipschitz constant. Large Lipschitz constants will result in poor lower bounds.
- We can use upper bounds instead of lower bounds, as well. By changing the minimum point to the maximum point in each step.

Shubert-Piyavskii Method

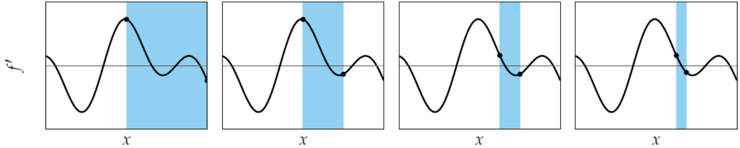


Bisection Method

The **bisection method** can be used to find roots of the function, or points where the function is zero. The **root-finding methods** can be used for optimization by applying them to the derivative of the objective, locating where $f'(x) = 0$. We must ensure that the resulting points are indeed local minima. In this method:

- The bisection method cuts the bracketed region in half with every iteration.
- The midpoint $(a + b)/2$ is evaluated, and the new bracket is formed from the midpoint and whichever side that continues to bracket a zero.
- We terminate immediately if the midpoint evaluates to zero. Otherwise we can terminate after a fixed number of iterations.
- The method is guaranteed to converge within ϵ of x^* within $\log_2\left(\frac{|b-a|}{\epsilon}\right)$ iterations, where $\log_2$ denotes the base 2 logarithm.

Bisection Method



Bisection Method

```
If a > b Then a, b = b, a EndIf
ya, yb = f(a), f(b)
If ya == 0 Then b = a EndIf
If yb == 0 Then a = b EndIf
While b − a > ϵ
    x = (a + b)/2
    y = f(x)
    If y == 0
        a, b = x, x
    ElseIf sign(y) == sign(ya)
        a = x
    Else
        b = x
    EndIf
EndWhile
Return (a, b)
```

MATLAB function `fminbnd`

`fminbnd` find minimum of single-variable function on fixed interval. It is a one-dimensional minimizer that finds a minimum for a problem specified by

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & x_1 < x < x_2 \end{aligned}$$

```
% Matlab Example
a = 9/7;
fun = @(x)sin(x-a);
x = fminbnd(fun, 1, 2*pi)
x = 5.9981
```

```
% Matlab Example
f = @(x) 2 - 2*x + exp(x)

% tolX has a default value of 1.0e-4
opts = optimset('tolX', 1.0e-6);
[xopt, fopt, ifl, out]
           = fminbnd(f, 0, 2,opts)
xopt = 0.6931;  fopt = 2.6137
```

- The algorithms that used in this function are `golden section search`, and `quadratic interpolation`.
- Try to use `optimset('Display','iter')`, and see results.

Julie Optim.jl

```
using Optim

f = x -> sin(x - 9/7); x1 = 0; x2 = 2π
result1 = optimize(f, x1, x2, Brent(), show_trace=true))
xopt1, fopt1 = Optim.minimizer(result1),  Optim.minimum(result1)

result2 = optimize(f, x1, x2, GoldenSection(),
                show_trace=true, abs_tol=1e-3)
xopt2, fopt2 = Optim.minimizer(result2),  Optim.minimum(result2)
```

Reference

1. Joaquim R. R. A. Martins, Andrew Ning, ”*Engineering Design Optimization*,” Cambridge University Press, 2021
2. Alexander Mitsos, ”*Applied Numerical Optimization*,” Lecture Note RWTH AACHEN University
3. Ashok D. Belegundu, Tirupathi R. Chandrupatla, ”Optimization Concepts and Applications in Engineering,” Cambridge University Press, 2019