# Optimization

# Optimization

**Goal:** Find the **minimizer** $\boldsymbol{x}^*$ that minimizes the **objective (cost) function** $f(\boldsymbol{x}): \mathcal{R}^n \to \mathcal{R}$

## Unconstrained Optimization

$$f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} f(\boldsymbol{x})$$

## Constrained Optimization

$$f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} f(\boldsymbol{x})$$

$$\text{s.t. } \mathbf{g}(\boldsymbol{x}) = \mathbf{0}$$ → Equality constraints

$$\mathbf{h}(\boldsymbol{x}) \leq \mathbf{0}$$ → Inequality constraints

# Optimization

- **What if we are looking for a maximizer $\boldsymbol{x}^*$?**

$$f(\boldsymbol{x}^*) = \max_{\boldsymbol{x}} f(\boldsymbol{x})$$

We can instead solve the minimization problem

$$f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}}(-f(\boldsymbol{x}))$$

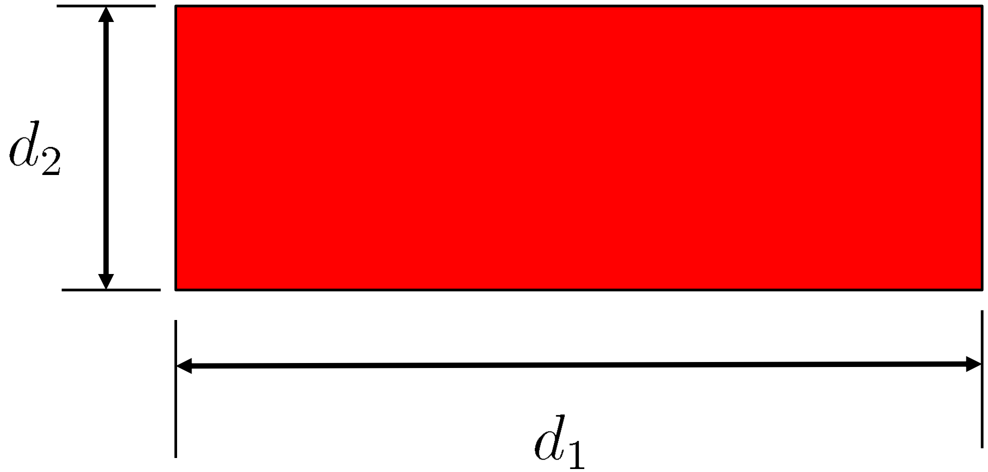
- **What if constraint is $\mathbf{h}(\boldsymbol{x}) > \mathbf{0}$?**

- **What if method only has inequality constraints?**

# Calculus problem: maximize the rectangle area subject to perimeter constraint

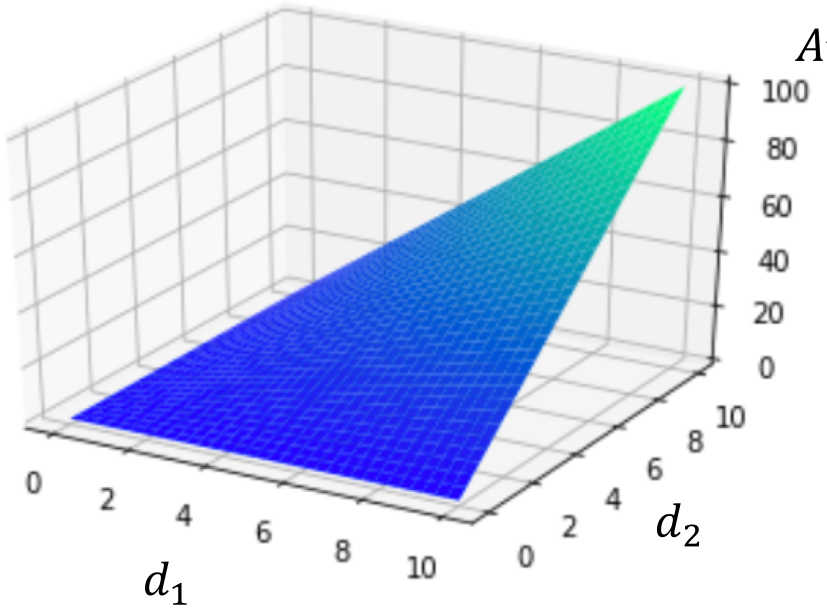
$$\max_{\boldsymbol{d} \in \mathcal{R}^2} \quad f(d_1, d_2) = d_1 \times d_2$$

$$\text{such that} \quad g(d_1, d_2) = 2\,(d_1 + d_2) - 20 \leq 0$$

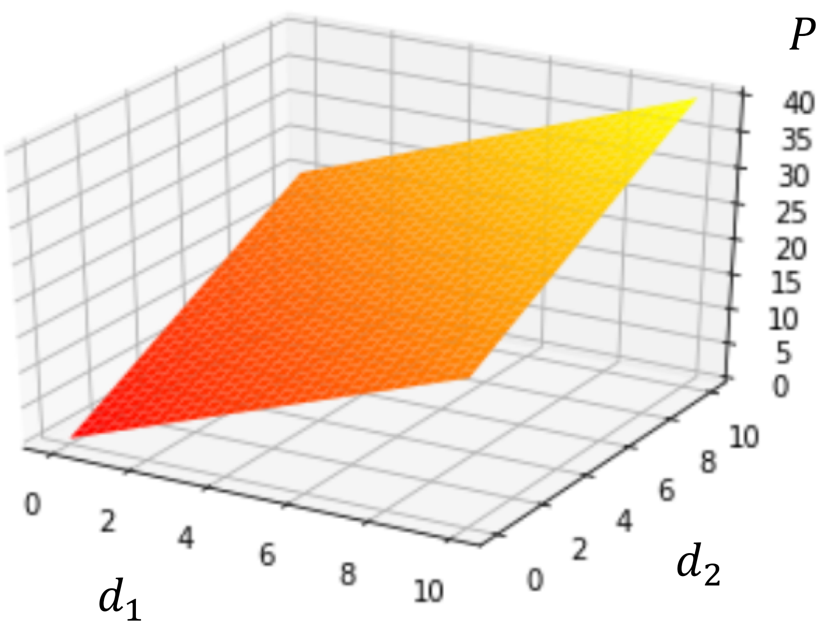


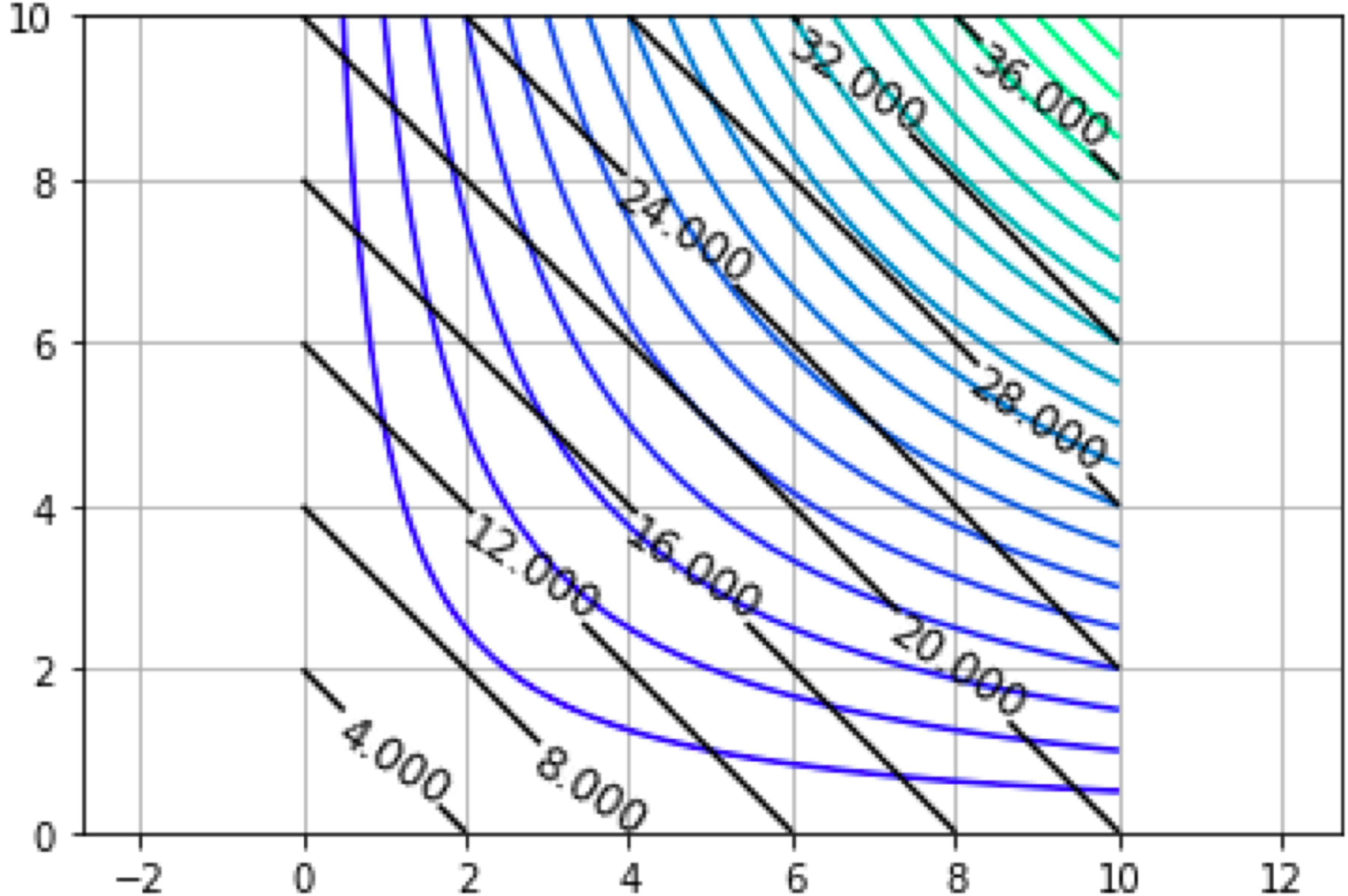

Demo: Constrained-Problem-2D

$$Area = d_1 d_2$$



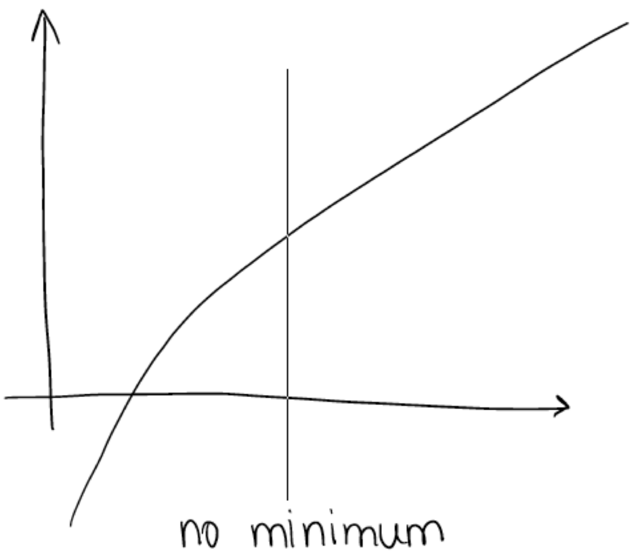

$$Perimeter = 2(d_1 + d_2)$$

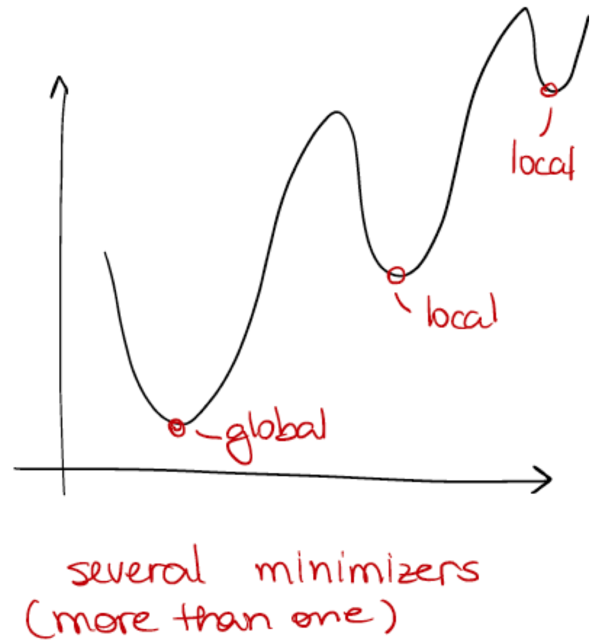




10
8
6
4
2
0
−2
0
2
4
6
8
10
12
4.000
8.000
12.000
16.000
20.000
24.000
28.000
32.000
36.000

# Does the solution exists? Local or global solution?



# Types of optimization problems

$$f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} f(\boldsymbol{x})$$

$f$: nonlinear, continuous and smooth

**Gradient-free methods**

Evaluate $f(\boldsymbol{x})$

**Gradient (first-derivative) methods**

Evaluate $f(\boldsymbol{x}), \boldsymbol{\nabla} f(\boldsymbol{x})$

**Second-derivative methods**

Evaluate $f(\boldsymbol{x}), \boldsymbol{\nabla} f(\boldsymbol{x}), \boldsymbol{\nabla}^2 f(\boldsymbol{x})$

# Taking derivatives…

$$f: \mathbb{R}^n \to \mathbb{R} \qquad f(\underset{\sim}{x}) = f(x_1, x_2, \dots, x_n)$$

$$\underset{\sim}{\nabla} f(\underset{\sim}{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \to \text{gradient}$$

$$\underset{=}{\nabla}^2 f(\underset{\sim}{x}) = \underset{=}{H}_f(\underset{\sim}{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_1 \partial x_3} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 \partial x_3} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & & & \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}_{n \times n \text{ matrix}}$$

# What is the optimal solution?

$$f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} f(\boldsymbol{x})$$

**(First-order) Necessary condition**

$$f'(x) = 0$$

$$\boldsymbol{\nabla} f(\boldsymbol{x}) = \boldsymbol{0}$$

**(Second-order) Sufficient condition**

$$f''(x) > 0$$

$\boldsymbol{\nabla}^2 f(\boldsymbol{x}) = \boldsymbol{H_f}$ is positive definite

$$\min_{\underset{\sim}{x}} f(\underset{\sim}{x})$$

First-order necessary condition

$\rightarrow \nabla f(\underset{\sim}{x}) = \underset{\sim}{0}$

Second-order sufficient condition

$\rightarrow \underset{\approx}{H}_f$ is positive definite

| eigenvalues of $H_f(x^*)$ | $H_f(x^*)$ | critical point $x^*$ |
|---|---|---|
| all positive | pos. def. | minimizer |
| all negative | neg. def. | maximizer |
| neg. and pos. | indefinite | saddle |

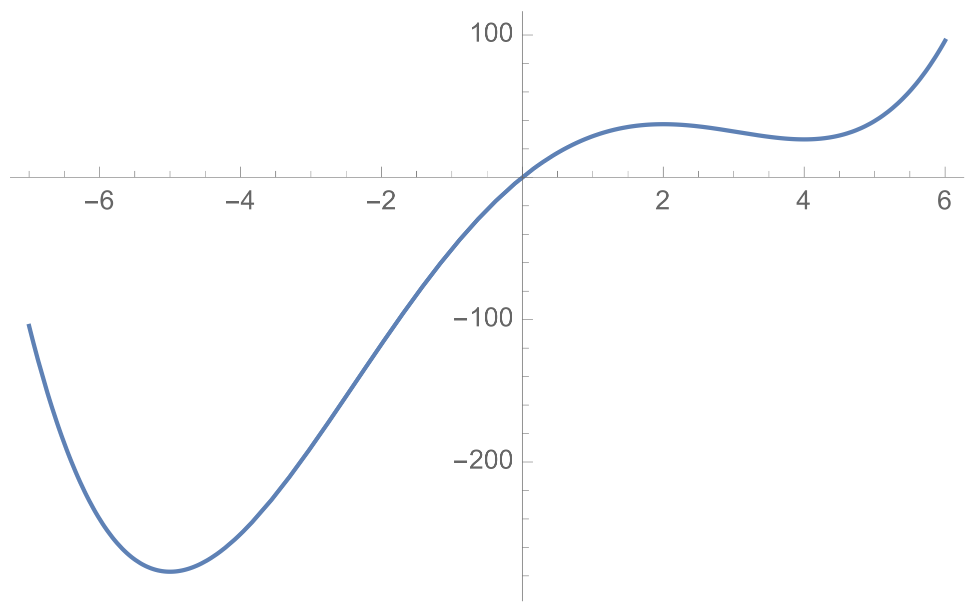
# Example (1D)

Consider the function $f(\boldsymbol{x}) = \frac{x^4}{4} - \frac{x^3}{3} - 11\,x^2 + 40x$

Find the stationary point and check the sufficient condition

$$f'(x) = \frac{4x^3}{4} - \frac{3x^2}{3} - 22x + 40$$

$$f''(x) = 3x^2 - 2x - 22$$



# Example (ND)

Consider the function $f(x_1, x_2) = 2x_1^3 + 4x_2^2 + 2x_2 - 24x_1$

Find the stationary point and check the sufficient condition

$$\nabla f = \begin{bmatrix} 6x_1^2 - 24 \\ 8x_2 + 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \rightarrow \begin{matrix} x_1 = \pm 2 \\ x_2 = -0.25 \end{matrix}$$

$$H_f = \begin{bmatrix} 12x_1 & 0 \\ 0 & 8 \end{bmatrix}$$

$$x^* = \begin{bmatrix} 2 \\ -0.25 \end{bmatrix} \rightarrow H_f = \begin{bmatrix} 24 & 0 \\ 0 & 8 \end{bmatrix}$$

positive eigenvalues

$\Rightarrow$ pos. def!

minimum

$$x^* = \begin{bmatrix} -2 \\ -0.25 \end{bmatrix} \rightarrow H_f = \begin{bmatrix} -24 & 0 \\ 0 & 8 \end{bmatrix} \Rightarrow \text{indefinite}$$

saddle

# Optimization in 1D: Golden Section Search

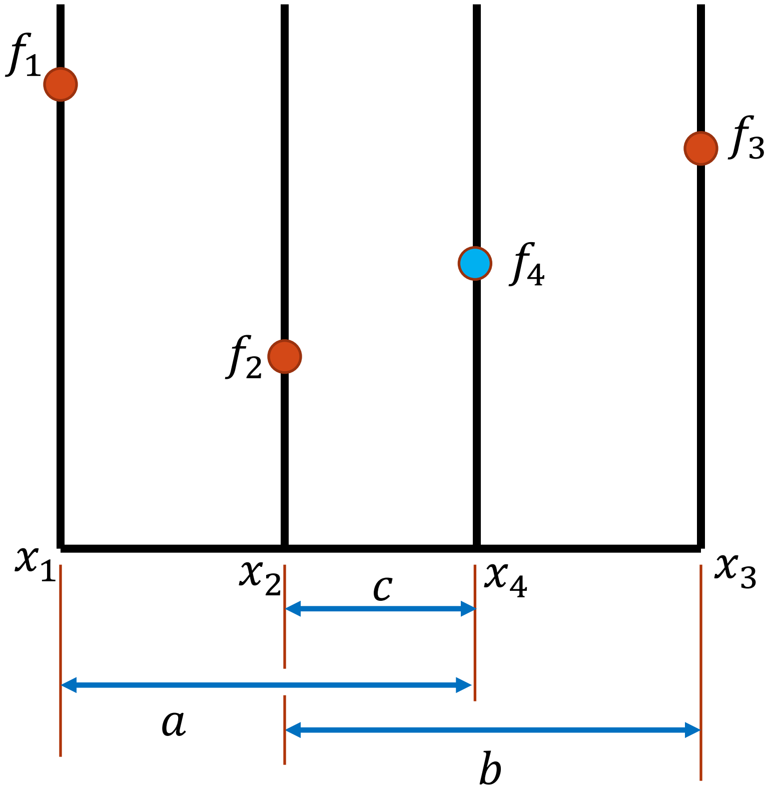
- Similar idea of bisection method for root finding
- Needs to bracket the minimum inside an interval
- Required the function to be unimodal

A function $f\colon \mathcal{R} \to \mathcal{R}$ is unimodal on an interval $[a, b]$

✓ There is a unique $\boldsymbol{x}^* \in [a, b]$ such that $f(\boldsymbol{x}^*)$ is the minimum in $[a, b]$

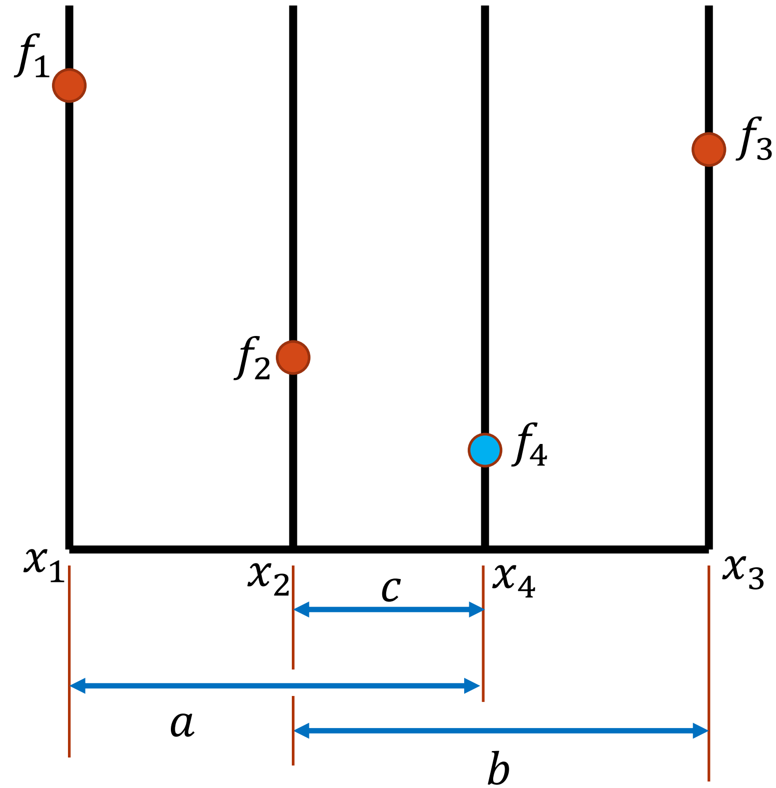
✓ For any $x_1, x_2 \in [a, b]$ with $x_1 < x_2$

- $x_2 < \boldsymbol{x}^* \Longrightarrow f(x_1) > f(x_2)$
- $x_1 > \boldsymbol{x}^* \Longrightarrow f(x_1) < f(x_2)$




$f_2 < f_4$

$x^* \epsilon [x_1, x_4]$

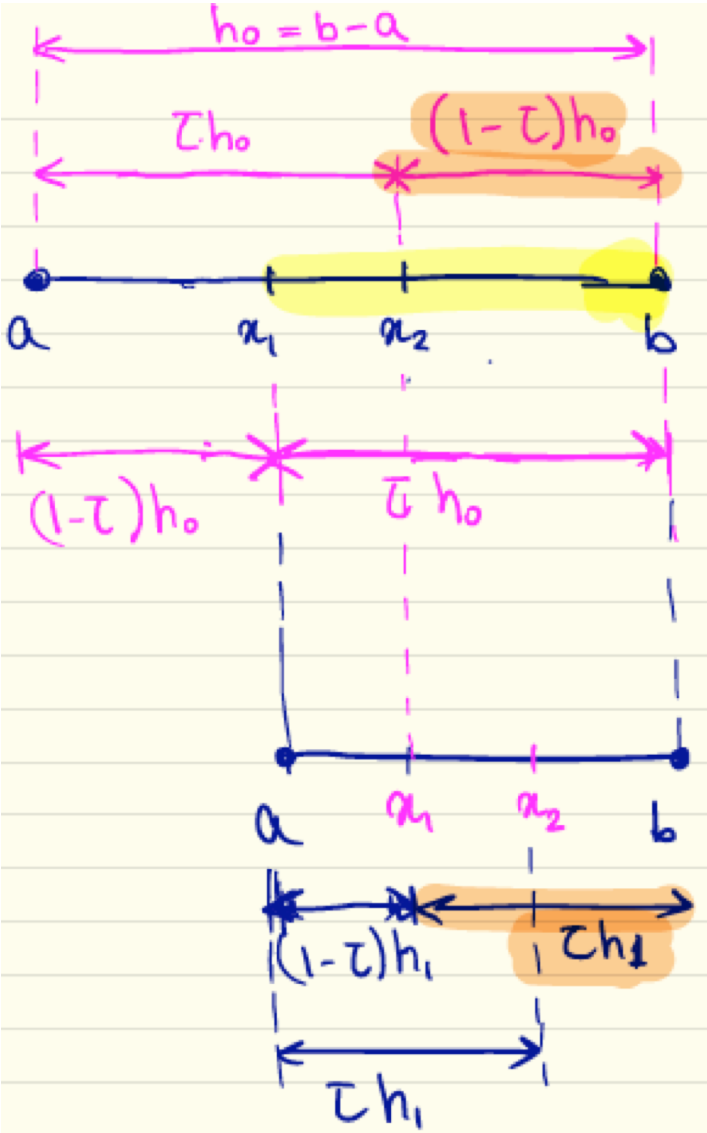



$f_2 > f_4$

$x^* \epsilon [x_2, x_3]$

Such method would in general require 2 new function evaluations per iteration. How can we select the points $x_2, x_4$ such that only one function evaluation is required?

# Golden Section Search




**Propose:**

$x_1 = a + (1-\tau)h_0$

$x_2 = a + \tau h_0$

Evaluate $f_1 = f(x_1)$

$f_2 = f(x_2)$

if $(f_1 > f_2)$:

$a = x_1$

$x_1 = x_2$ → already have func. value!

$h_1 = b - a$

$x_2 = a + \tau h_1$

$f_2 = f(x_2)$ → only one

if $(f_1 < f_2)$:

$b = x_2$

$x_2 = x_1$

$x_1 = a + (1-\tau)h_1$

$f_1 = f(x_1)$

# Golden Section Search

Demo: Golden Section Proportions

What happens with the length of the interval after one iteration?

$$h_1 = \tau\, h_o$$

Or in general: $h_{k+1} = \tau\, h_k$

**Hence the interval gets reduced by $\boldsymbol{\tau}$**
(for bisection method to solve nonlinear equations, $\tau$=0.5)

For recursion:

$$\tau\, h_1 = (1 - \tau)\, h_o$$
$$\tau\, \tau\, h_o = (1 - \tau)\, h_o$$
$$\tau^2 = (1 - \tau)$$
$$\boldsymbol{\tau = 0.618}$$

# Golden Section Search

- Derivative free method!

- Slow convergence:

$$\lim_{k \to \infty} \frac{|e_{k+1}|}{|e_k|} = 0.618 \qquad r = 1\ (linear\ convergence)$$

- Only one function evaluation per iteration

# Iclicker question

Consider running golden section search on a function that is unimodal. If golden section search is started with an initial brakcet of $[-10, 10]$, what is the length of the new bracket after 1 iteration?

A) 20
B) 10
C) 12.36
D) 7.64

# Newton's Method

Using Taylor Expansion, we can approximate the function $f$ with a quadratic function about $x_0$

$$f(x) \approx f(x_0) + f'(x_0)\,(x - x_0) + \frac{1}{2} f''(x_0)\,(x - x_0)^2$$

And we want to find the minimum of the quadratic function using the first-order necessary condition

$$f'(x) = 0 \Rightarrow f'(x_0) + f''(x_0)(x - x_0) = 0$$

$$h = (x - x_0) \Rightarrow h = \frac{-f'(x_0)}{f''(x_0)}$$

Note that this is the same as the step for the Newton's method to solve the nonlinear equation $f'(x) = 0$

# Newton's Method

- **Algorithm:**

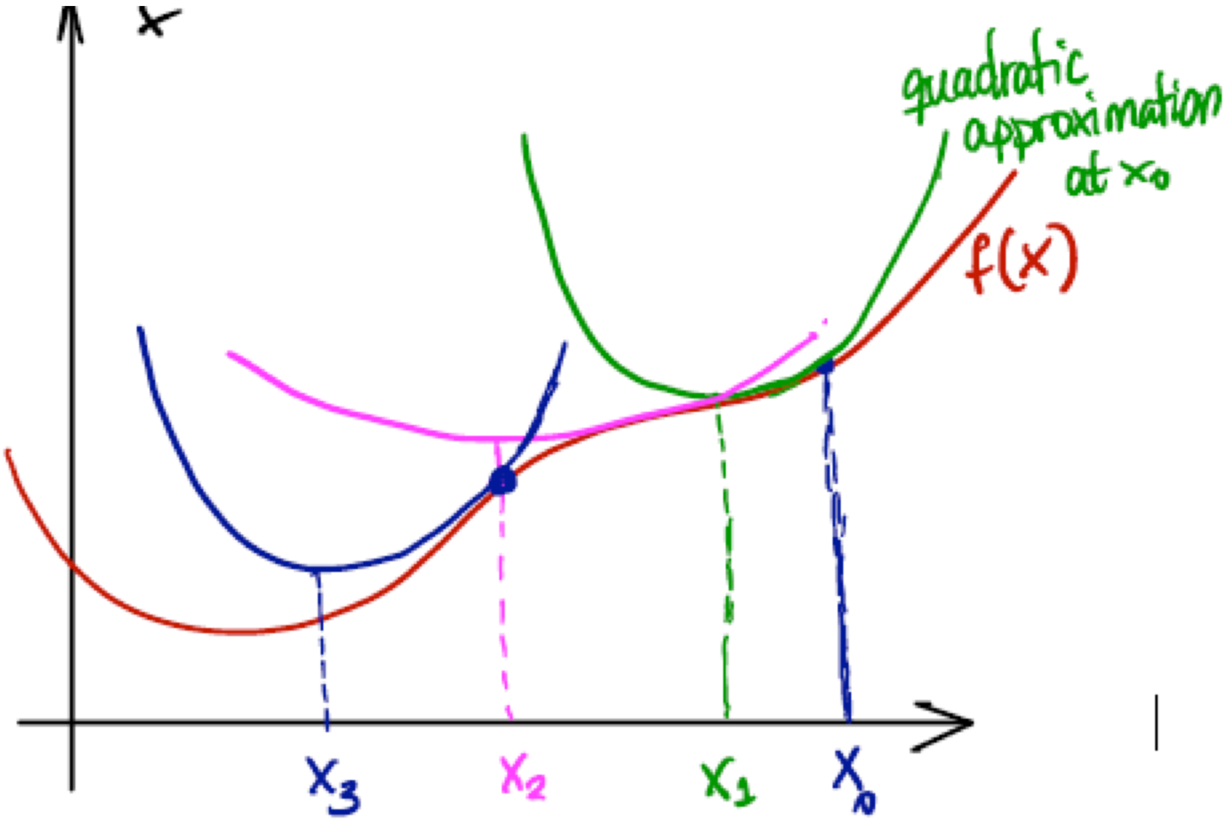
$x_0 =$ starting guess

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

- **Convergence:**
  - Typical quadratic convergence
  - Local convergence (start guess close to solution)
  - May fail to converge, or converge to a maximum or point of inflection

Demo: ”Newton's method in 1D”
And “Newton's method Initial Guess”

# Newton's Method (Graphical Representation)



# Iclicker

Consider the function $f(x) = 4\,x^3 + 2\,x^2 + 5\,x + 40$

If we use the initial guess $x_0 = 2$, what would be the value of $x$ after one iteration of the Newton's method?

*A)* $x_1 = 2.852$
*B)* $x_1 = 1.147$
*C)* $x_1 = 3.173$
*D)* $x_1 = 0.827$
E) NOTA

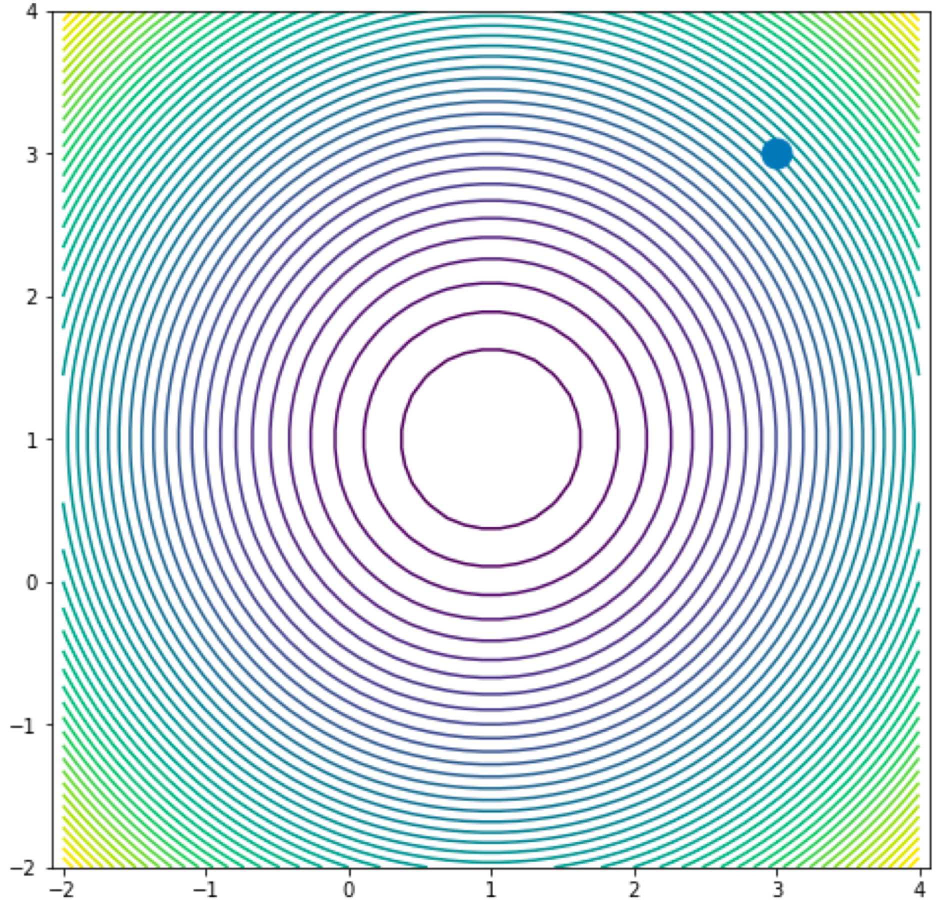
# Optimization in ND: Steepest Descent Method

Given a function $f(\boldsymbol{x}): \mathcal{R}^n \rightarrow \mathcal{R}$ at a point $\boldsymbol{x}$, the function will decrease its value in the direction of steepest descent: $-\boldsymbol{\nabla} f(\boldsymbol{x})$

*Iclicker question:*

What is the steepest descent direction?

$$f(x_1, x_2) = (x_1 - 1)^2 + (x_2 - 1)^2$$



# Steepest Descent Method

Start with initial guess:

$$\boldsymbol{x}_0 = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

Check the update:

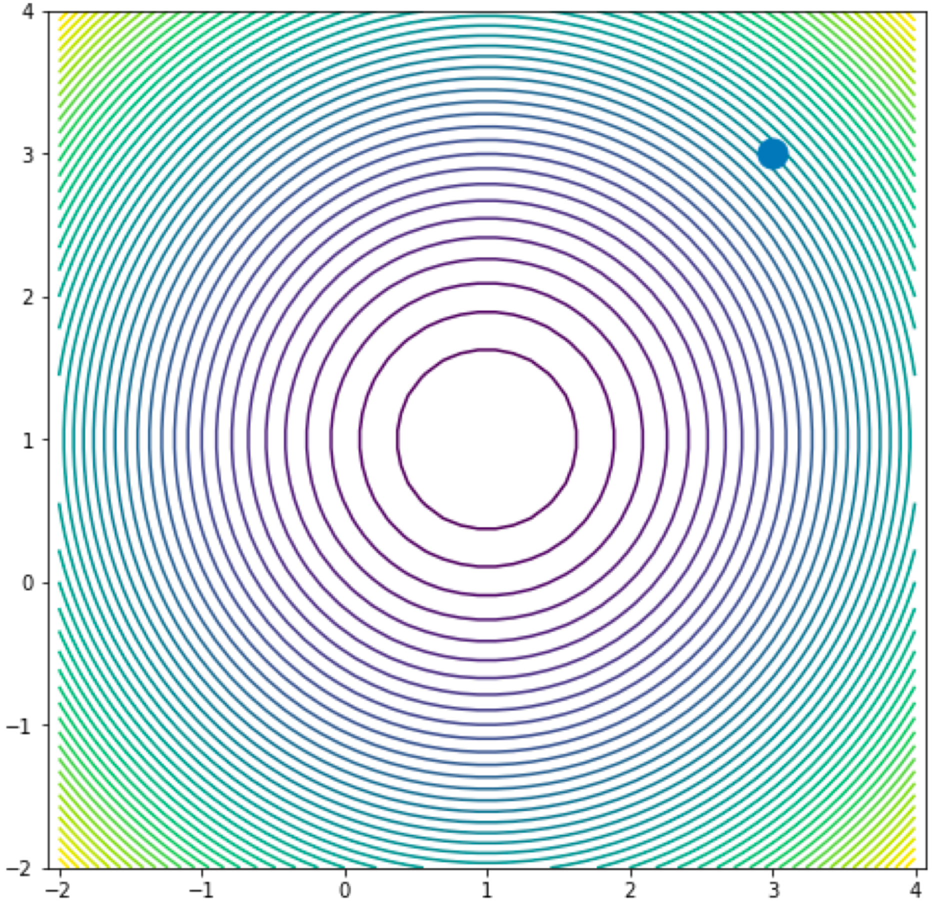
$$\boldsymbol{x}_1 = \boldsymbol{x}_0 - \boldsymbol{\nabla} f(\boldsymbol{x}_0)$$

$$\boldsymbol{\nabla} f(\boldsymbol{x}) = \begin{bmatrix} 2(x_1 - 1) \\ 2(x_2 - 1) \end{bmatrix}$$

$$\boldsymbol{x}_1 = \begin{bmatrix} 3 \\ 3 \end{bmatrix} - \begin{bmatrix} 4 \\ 4 \end{bmatrix} = -\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

How far along the gradient direction should we go?

$$f(x_1, x_2) = (x_1 - 1)^{\mathbf{2}} + (x_2 - 1)^{\mathbf{2}}$$



# Steepest Descent Method

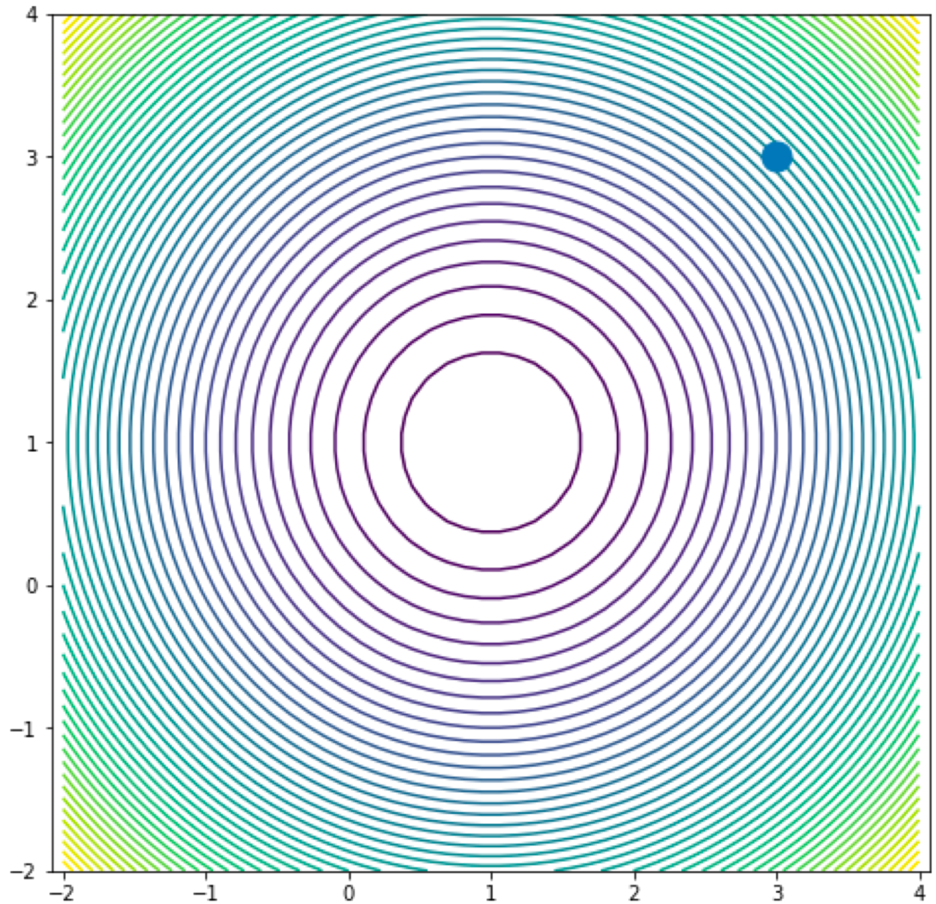
Update the variable with:

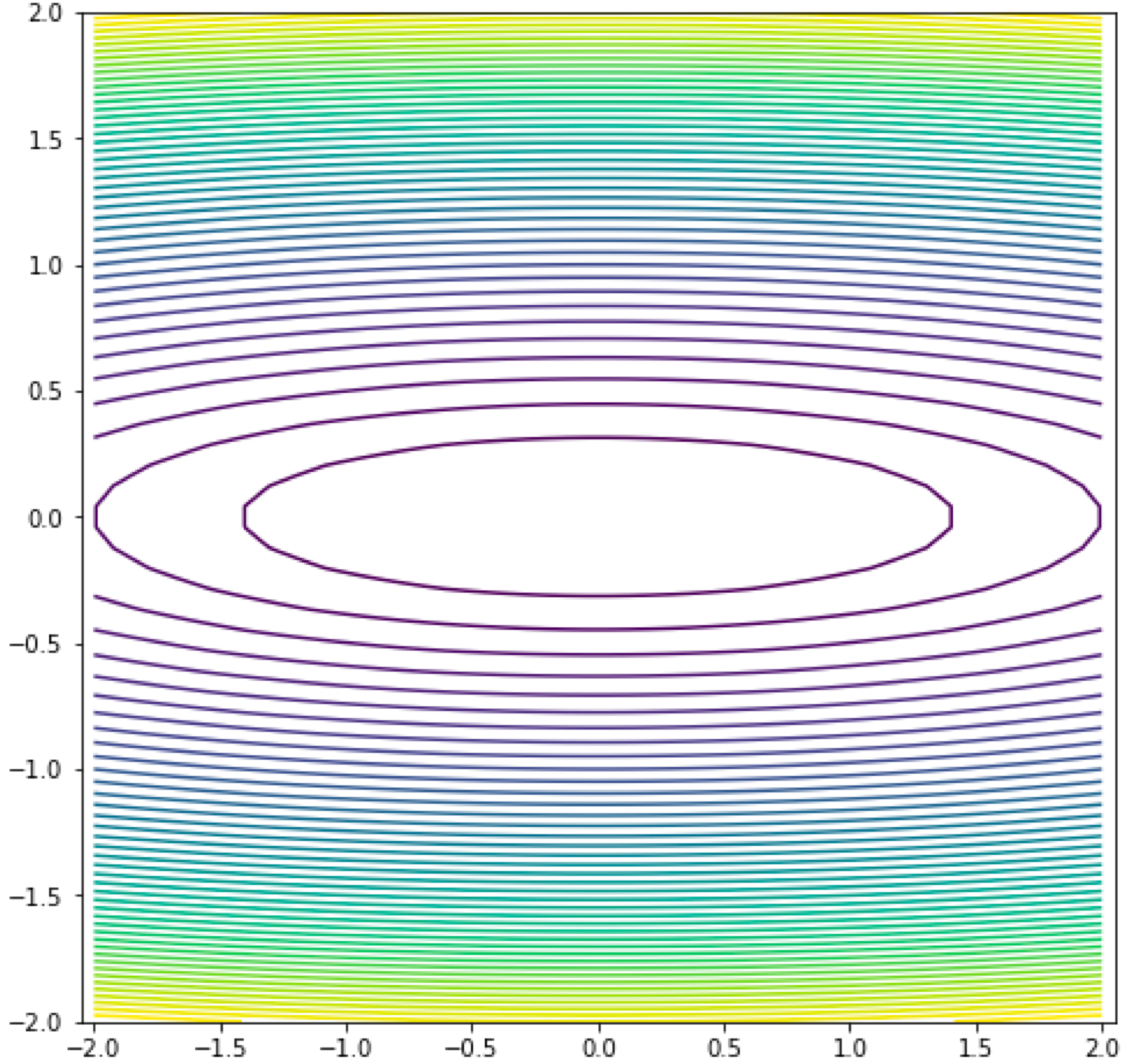
$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha_k \boldsymbol{\nabla} f(\boldsymbol{x}_k)$$

How far along the gradient should we go? What is the "best size" for $\alpha_k$?

A) 0
B) 0.5
C) 1
D) 2
E) Cannot be determined

$$f(x_1, x_2) = (x_1 - 1)^2 + (x_2 - 1)^2$$





2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0

# Steepest Descent Method

**Algorithm:**

Initial guess: $\boldsymbol{x}_0$

Evaluate: $\boldsymbol{s}_k = -\boldsymbol{\nabla} f(\boldsymbol{x}_k)$

Perform a line search to obtain $\alpha_k$ (for example, Golden Section Search)

$$\alpha_k = \underset{\alpha}{\operatorname{argmin}} f(\boldsymbol{x}_k + \alpha\, \boldsymbol{s}_k)$$

Update: $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k\, \boldsymbol{s}_k$

# Line search

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k)$$

what is $\alpha_k$ such that $f(x_{k+1})$ is minimized?

$$\min_{\alpha} f(\underbrace{x_k - \alpha \nabla f(x_k)}_{x_{k+1}})$$

Necessary condition: $\frac{df}{d\alpha} = 0$

Chain rule:

$$\frac{df}{d\alpha} = \frac{df}{dx_{k+1}} \frac{dx_{k+1}}{d\alpha} \qquad \frac{dx_{k+1}}{d\alpha} = -\nabla f(x_k)$$

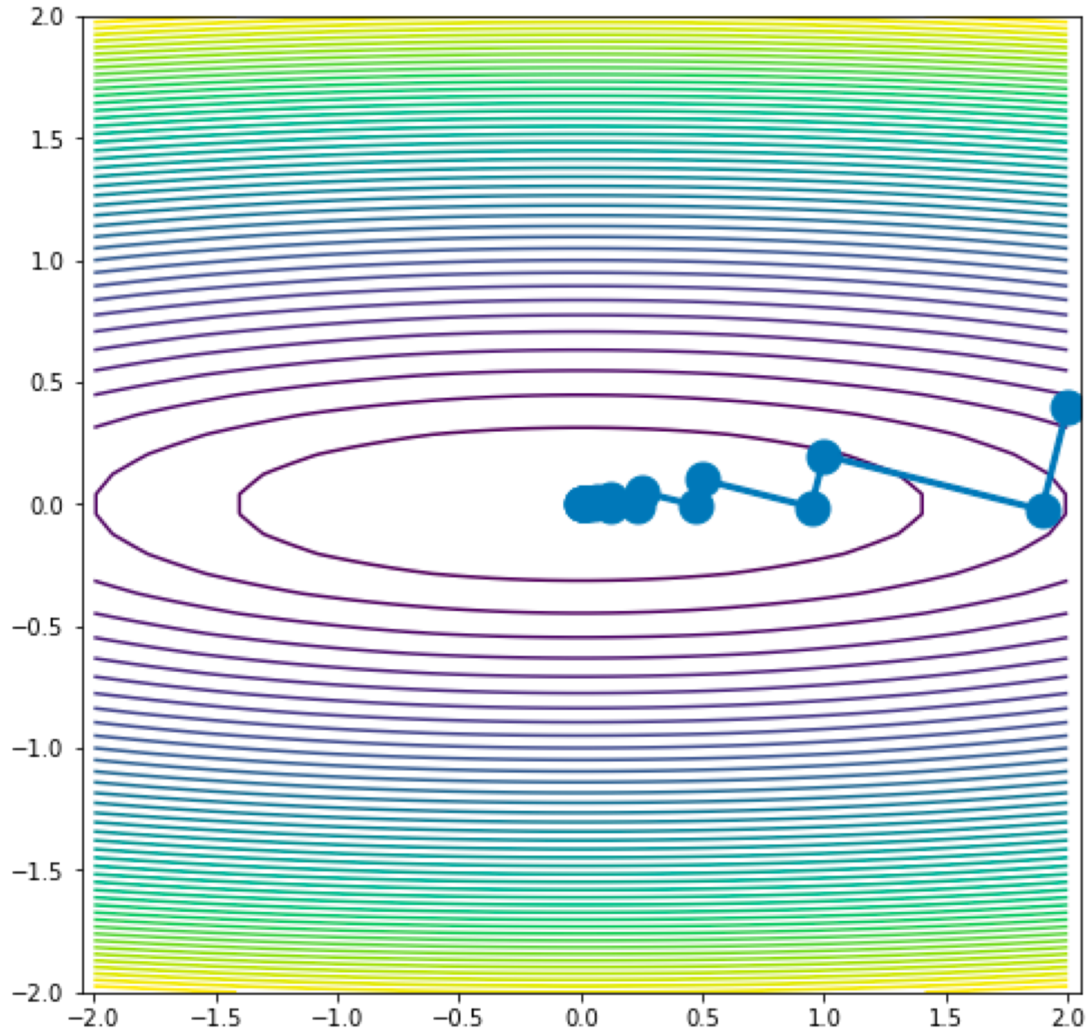
$$= -\nabla f(x_{k+1})^T \nabla f(x_k) = 0$$

$\nabla f(x_{k+1})$ is orthogonal to $\nabla f(x_k)$!

# Steepest Descent Method

**Demo:** Steepest Descent

**Convergence:** linear




Demo: "Steepest Descent"

# Iclicker question:

Consider minimizing the function

$$f(x_1, x_2) = 10(x_1)^3 - (x_2)^2 + x_1 - 1$$

Given the initial guess

$$x_1 = 2,\ x_2 = 2$$

what is the direction of the first step of gradient descent?

A) $\begin{bmatrix} -61 \\ 4 \end{bmatrix}$

C) $\begin{bmatrix} -120 \\ 4 \end{bmatrix}$

B) $\begin{bmatrix} -61 \\ 2 \end{bmatrix}$

D) $\begin{bmatrix} -121 \\ 4 \end{bmatrix}$

# Newton's Method

Using Taylor Expansion, we build the approximation:

$$f(\boldsymbol{x}+\boldsymbol{s}) \approx f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T \boldsymbol{H_f}(\boldsymbol{x})\,\boldsymbol{s} = \hat{f}(\boldsymbol{s})$$

And we want to find the minimum $\hat{f}(\boldsymbol{s})$, so we enforce the first-order necessary condition

$$\nabla \hat{f}(\boldsymbol{s}) = \boldsymbol{0} \Rightarrow \nabla f(\boldsymbol{x}) + \frac{1}{2} 2\, \boldsymbol{H_f}(\boldsymbol{x})\,\boldsymbol{s} = 0$$

$$\Rightarrow \quad \boldsymbol{H_f}(\boldsymbol{x})\,\boldsymbol{s} = -\nabla f(\boldsymbol{x})$$

Which becomes a system of linear equations where we need to solve for the Newton step $\boldsymbol{s}$

# Newton's Method

**Algorithm:**

Initial guess: $\boldsymbol{x}_0$

Solve: $\boldsymbol{H}_{\boldsymbol{f}}(\boldsymbol{x}_k)\,\boldsymbol{s}_k = -\boldsymbol{\nabla} f(\boldsymbol{x}_k)$

Update: $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}_k$

Note that the Hessian is related to the curvature and therefore contains the information about how large the step should be.

# Iclicker question

To find a minimum of the function $f(x, y) = 3x^2 + 2y^2$, which is the expression for one step of Newton's method?

A) $\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \end{bmatrix} - \begin{bmatrix} 6 & 0 \\ 0 & 4 \end{bmatrix}^{-1} \begin{bmatrix} 6x_k \\ 4y_k \end{bmatrix}$

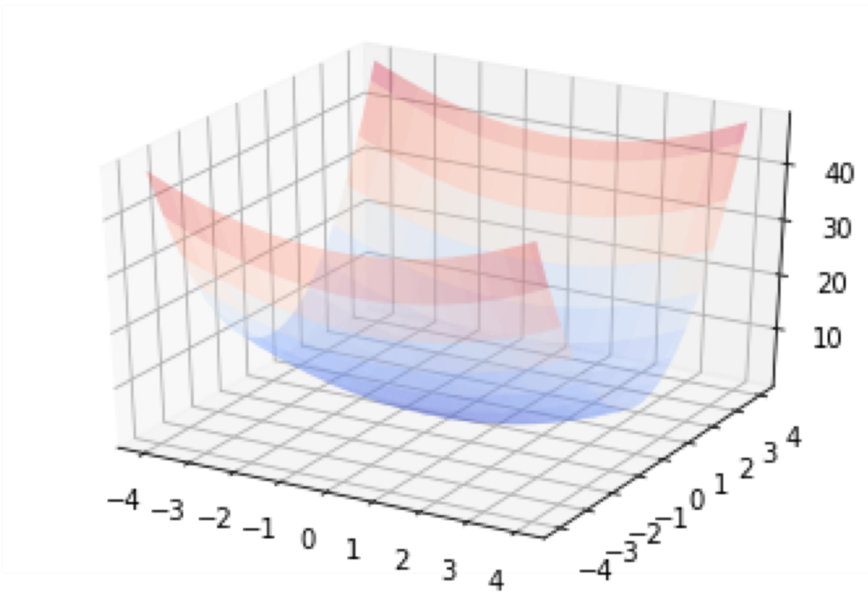
B) $\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = -\begin{bmatrix} 6 & 0 \\ 0 & 4 \end{bmatrix}^{-1} \begin{bmatrix} 6x_k \\ 4y_k \end{bmatrix}$

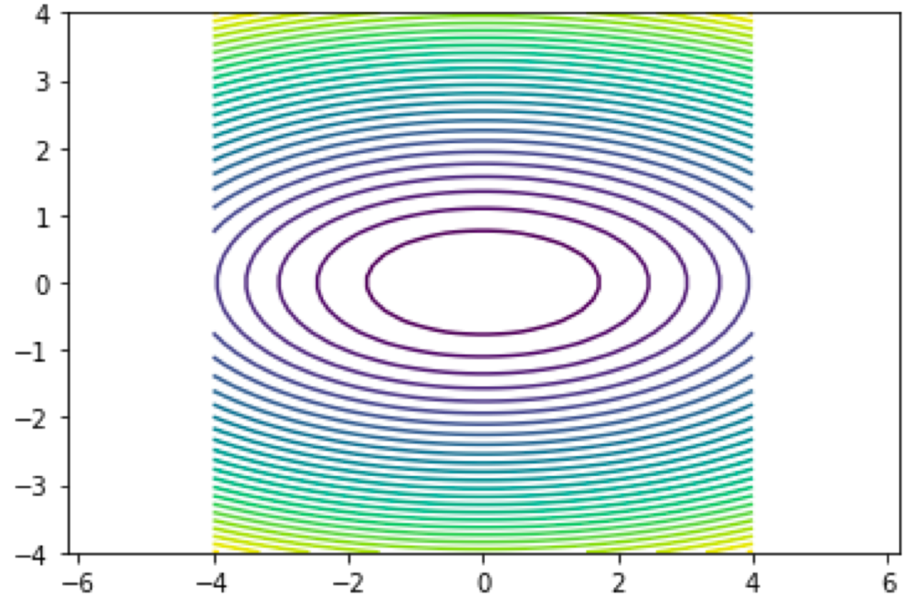
C) $\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} 6 & 0 \\ 0 & 4 \end{bmatrix}^{T} \begin{bmatrix} 6x_k \\ 4y_k \end{bmatrix}$

D) $\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \end{bmatrix} - \begin{bmatrix} 6 & 0 \\ 0 & 4 \end{bmatrix}^{T} \begin{bmatrix} 6x_k \\ 4y_k \end{bmatrix}$

# Iclicker question:

$$f(x, y) = 0.5x^2 + 2.5y^2$$




When using the Newton's Method to find the minimizer of this function, estimate the number of iterations it would take for convergence?

A) 1   B) 2-5   C) 5-10   D) More than 10   E) Depends on the initial guess

# Newton's Method Summary

**Algorithm:**

Initial guess: $\boldsymbol{x}_0$

Solve: $\boldsymbol{H}_{\boldsymbol{f}}(\boldsymbol{x}_k)\,\boldsymbol{s}_k = -\nabla f(\boldsymbol{x}_k)$

Update: $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}_k$

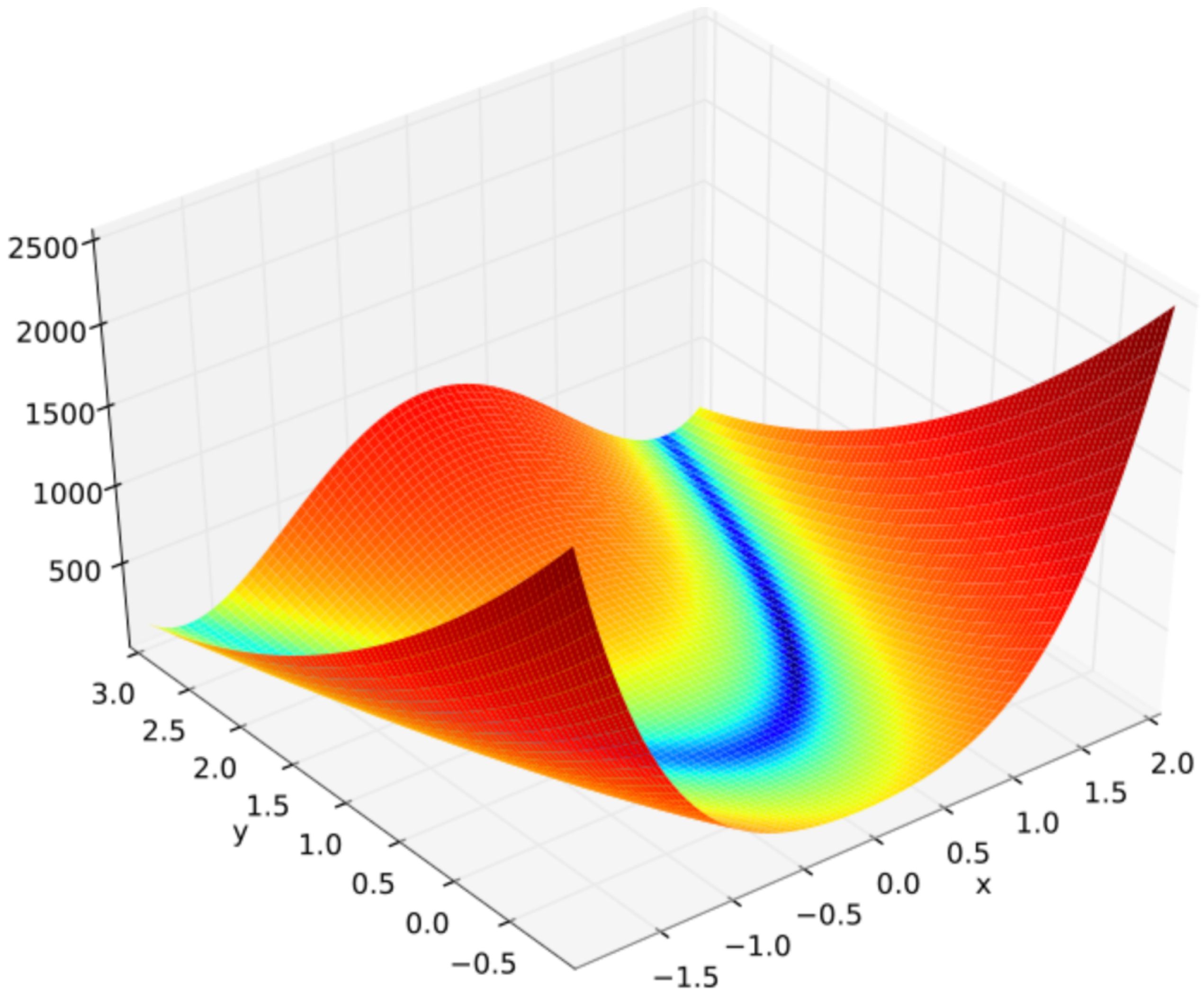
**About the method…**

- Typical quadratic convergence ☺
- Need second derivatives ☹
- Local convergence (start guess close to solution)
- Works poorly when Hessian is nearly indefinite
- Cost per iteration: $O(n^3)$

Demo: "Newton's method in n dimensions"

# Example:

https://en.wikipedia.org/wiki/Rosenbrock_function



# Iclicker question:

Recall Newton's method and the steepest descent method for minimizing a function $f(\boldsymbol{x}): \mathcal{R}^n \rightarrow \mathcal{R}$. How many statements below describe the Newton Method's only (not both)?

1. Convergence is linear
2. Requires a line search at each iteration
3. Evaluates the Gradient of $f(\boldsymbol{x})$ at each iteration
4. Evaluates the Hessian of $f(\boldsymbol{x})$ at each iteration
5. Computational cost per iteration is $O(n^3)$

A) 1  B) 2  C) 3  D) 4  E) 5