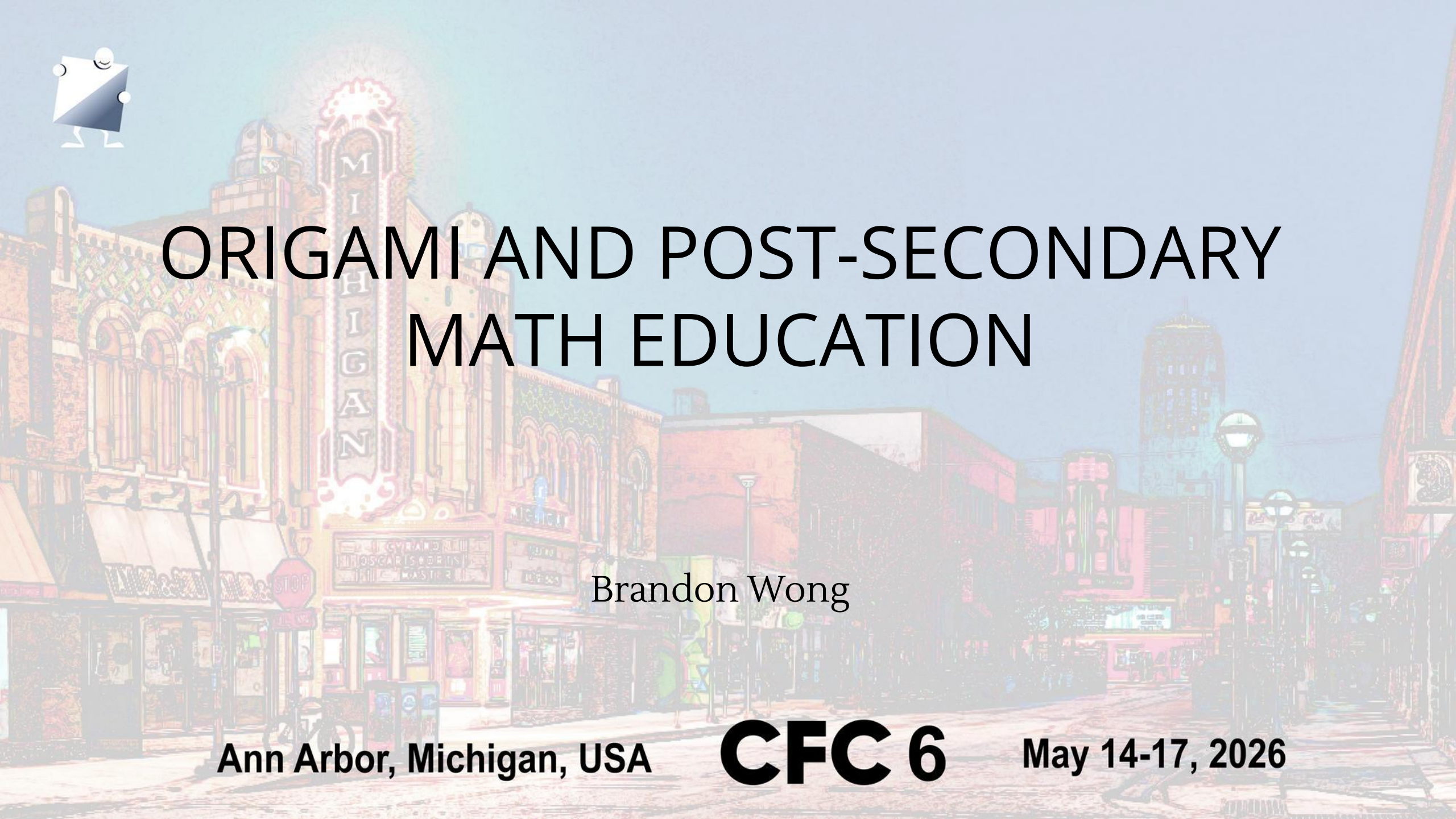


ORIGAMI AND POST-SECONDARY MATH EDUCATION

Brandon Wong

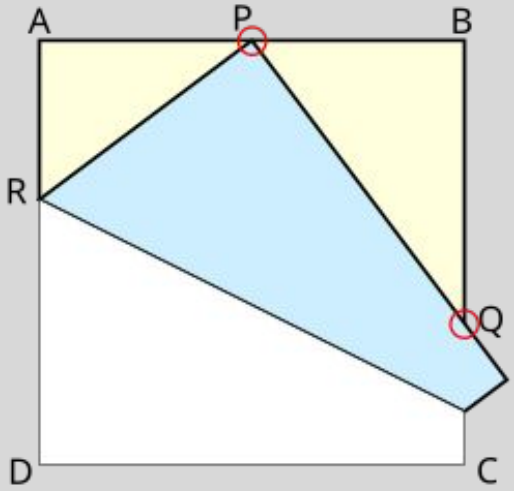
Ann Arbor, Michigan, USA CFC 6 May 14-17, 2026

CONTENTS

BACKGROUND: ORIGAMI IN MATH

Primary school

Secondary school



Post-secondary

?

Academia

- **Tortilla-tortilla constraint** (i, j, k, l): Creases e_{ij}, e_{kl} overlap, so there are points $p \in e_{ij}$ and $q \in e_{kl}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $P' \subset F_i \cup F_j \cup e_{ij}$ and $Q' \subset F_k \cup F_l \cup e_{kl}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P' \cap e_{ij}$, $q' \in Q' \cap e_{kl}$ such that $f(p') = f(q')$. For ε' sufficiently small, there are points $p^+ \in F_i \cap P'$, $q^+ \in F_k \cap Q'$ with $f(p^+) = f(q^+)$ and points $p^- \in F_j \cap P'$, $q^- \in F_l \cap Q'$ with $f(p^-) = f(q^-)$ such that the distances from p^+, p^- to p' along P', and from q^+, q^- to q' along Q', are all ε'. Then $\lambda(p^+, q^+) = \lambda(p^-, q^-)$ by the tortilla-tortilla condition. Because $p^+ \in F_i$, $p^- \in F_j$, $q^+ \in F_k$, and $q^- \in F_l$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jl}$.
- **Taco-tortilla constraint** (i, j, k): F_k and crease e_{ij} overlap, so there are points $p \in F_k$ and $q \in e_{ij}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $P' \subset F_k$ and $Q' \subset F_i \cup F_j \cup e_{ij}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P', q' \in Q' \cap e_{ij}$ such that $f(p') = f(q')$. For ε' sufficiently small there are points $q^+ \in F_i \cap Q', q^- \in F_j \cap Q', p^+ \in F_k \cap P'$ with $f(q^+) = f(q^-) = f(p^+)$ such that the distances from p^+ to p' along P', and from q^+, q^- to q' along Q', are all ε'. Then $\lambda(q^+, p^+) = \lambda(q^-, p^+)$ by the taco-tortilla condition. Because $q^+ \in F_i$, $q^- \in F_j$, and $p^+ \in F_k$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jk}$.
- **Taco-tortilla constraint** (i, j, k, l): Creases e_{ij}, e_{kl} overlap, so there are points $q \in e_{ij}$ and $p \in e_{kl}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $Q' \subset F_i \cup F_j \cup e_{ij}$ and $P' \subset F_k \cup F_l \cup e_{kl}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P' \cap e_{kl}$, $q' \in Q' \cap e_{ij}$ such that $f(p') = f(q')$. For ε' sufficiently small, there are points $q^+ \in F_i \cap Q', q^- \in F_j \cap Q', p^+ \in F_k \cap P'$ with $f(q^+) = f(q^-) = f(p^+)$ such that the distances from p^+ to p' along P', and from q^+, q^- to q' along Q', are all ε'. By the taco-tortilla condition, $\lambda(p^+, q^+) = \lambda(p^-, q^-)$. Because $p^+ \in F_i$, $p^- \in F_j$, and $q^+ \in F_k$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jl}$.

BRANCHES OF MATH*

Calculus

Linear algebra

Differential equations

Discrete math

Optimization

Abstract algebra

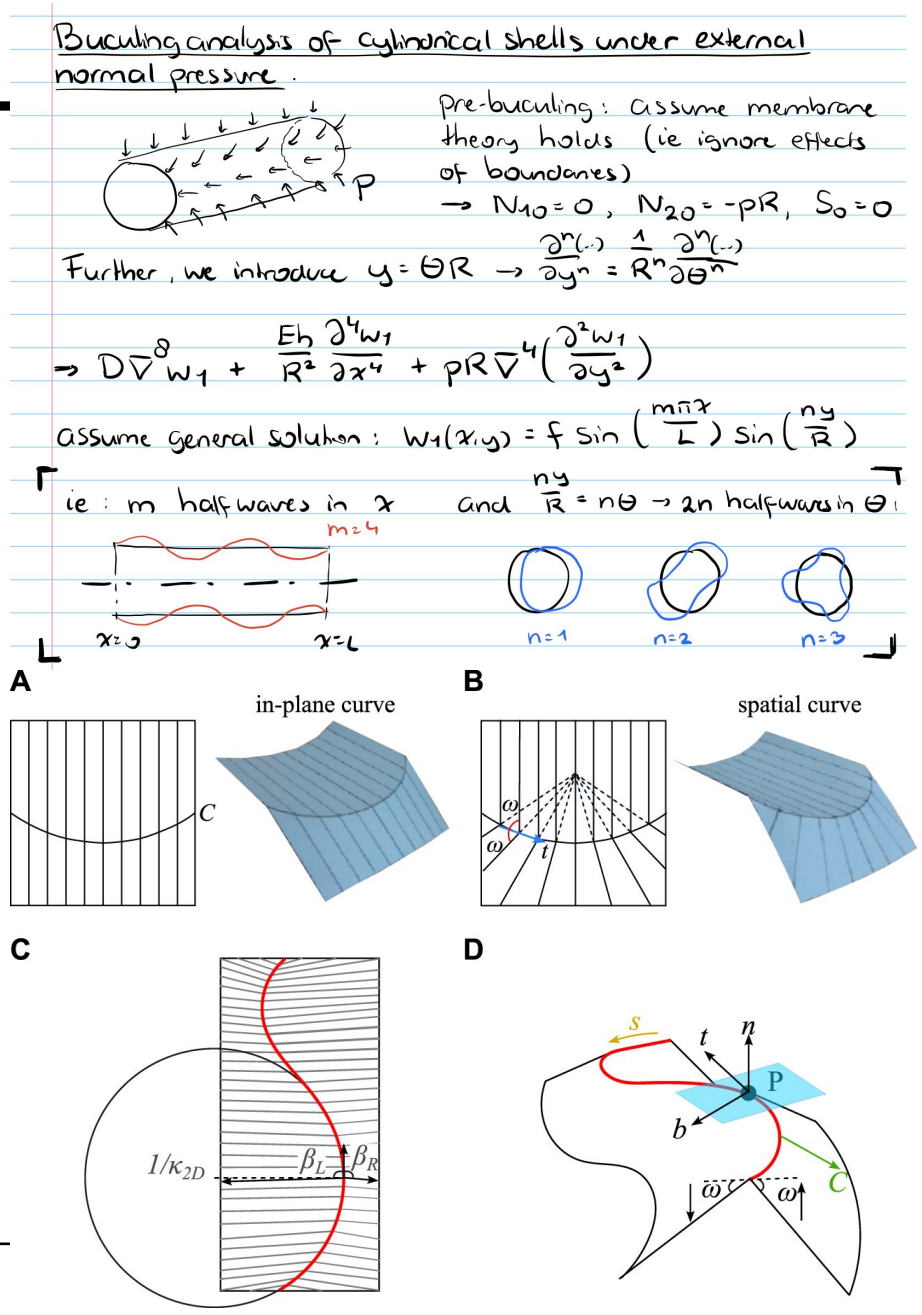
*greatly simplified, these are based on common undergraduate math courses

CALCULUS

Lowkey can’t think of any direct connections, but calculus is the foundation of the rest of the fields

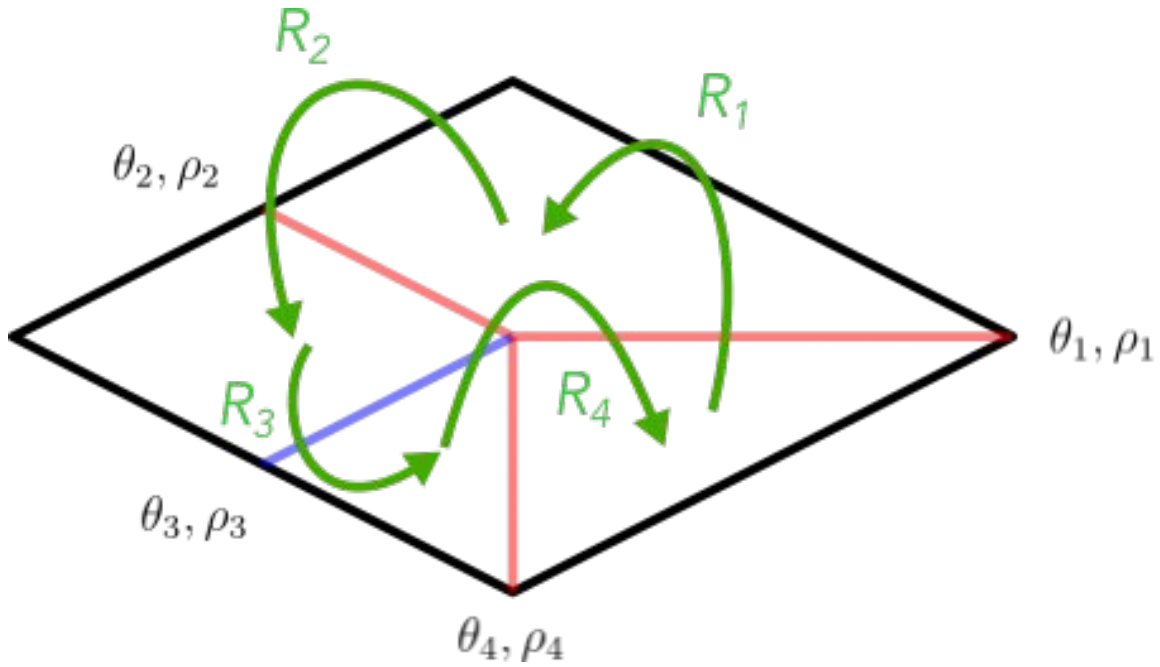
CALCULUS + DIFFERENTIAL EQUATIONS

- Core to engineering, mechanics of materials, dynamics
  - Paper can be modelled as a membrane/plate-like solid
- Differential geometry: curved crease folding



LINEAR ALGEBRA

- Quaternions and rotation matrices
  - Useful for rendering folded states in 2D and 3D
  - 3D Kawasaki's theorem

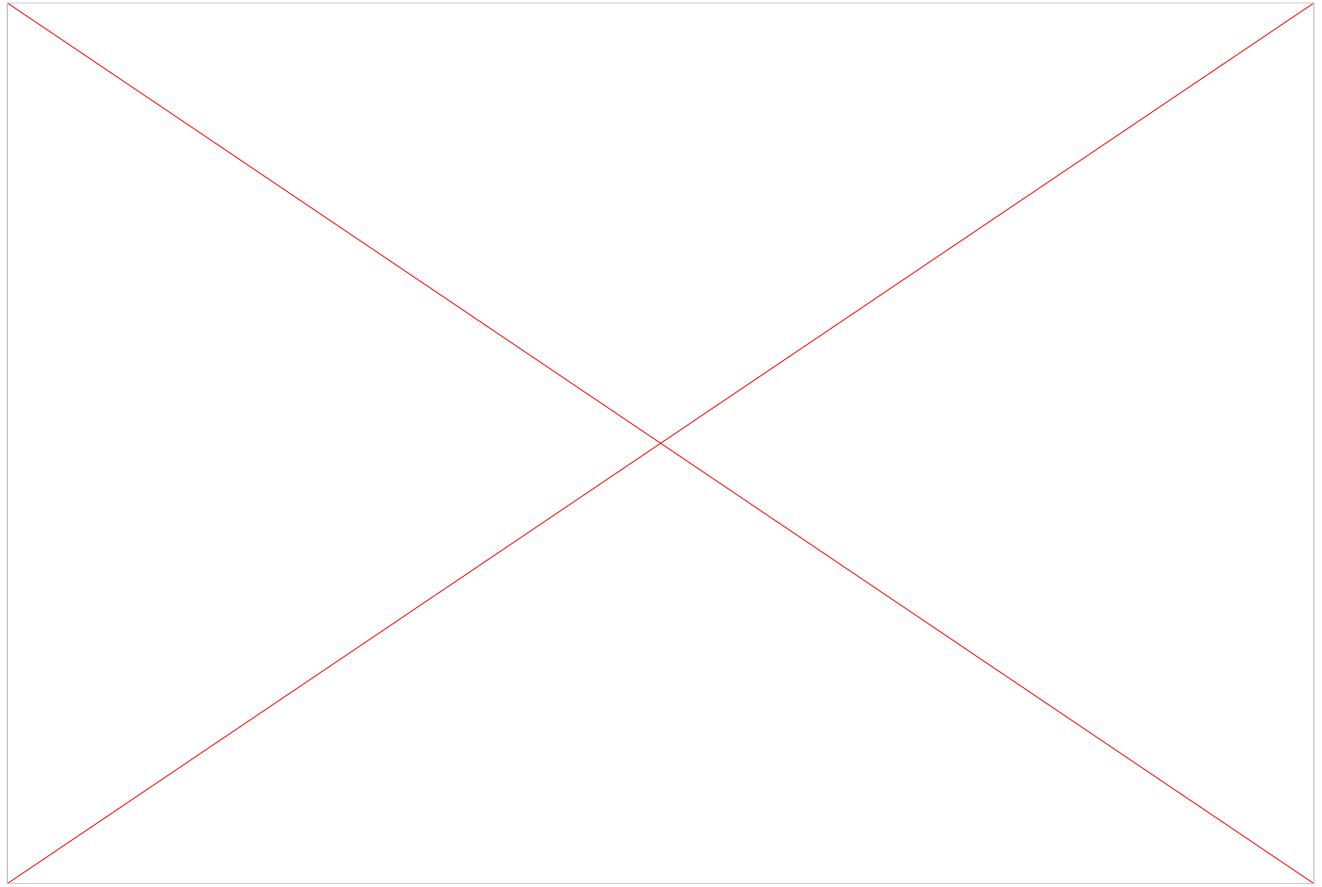


$$R(\theta,\rho) = \begin{bmatrix} 1-(1-\cos\rho)\sin^2\theta & (1-\cos\rho)\cos\theta\sin\theta & \sin\rho\sin\theta \\ (1-\cos\rho)\cos\theta\sin\theta & 1-(1-\cos\rho)\cos^2\theta & -\sin\rho\cos\theta \\ -\sin\rho\sin\theta & \sin\rho\cos\theta & \cos\rho \end{bmatrix}$$

$$R_1 R_2 R_3 \ldots R_n = I_3$$

DISCRETE MATH

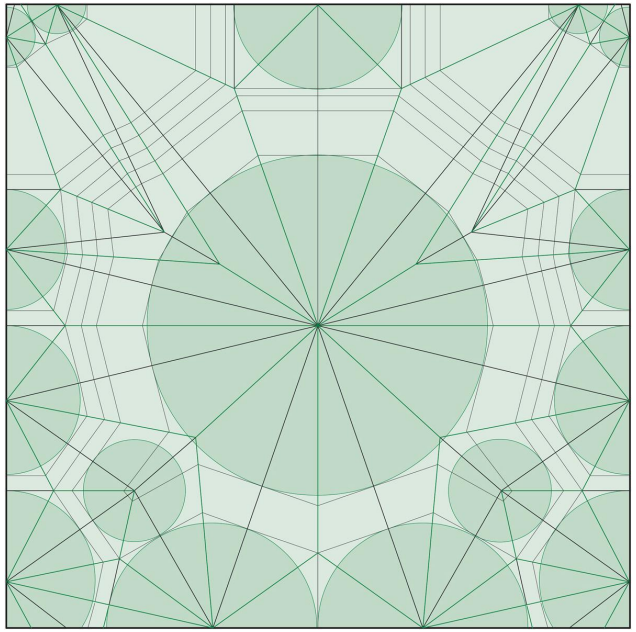
- Graph theory: literally anything to do with crease patterns
  - Also anything to do with trees
- Combinatorics: enumeration/brute force generation of folds
- Logic: layer ordering, global flat foldability (self intersection)

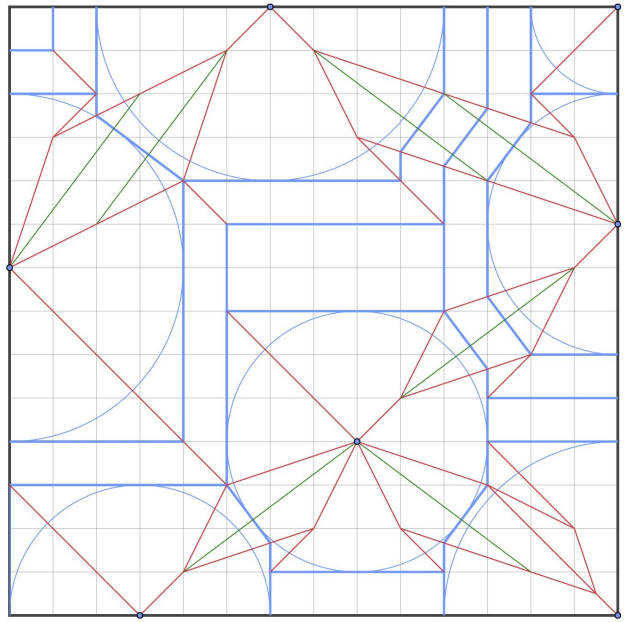


OPTIMIZATION

Most origami design problems can be formulated as optimization

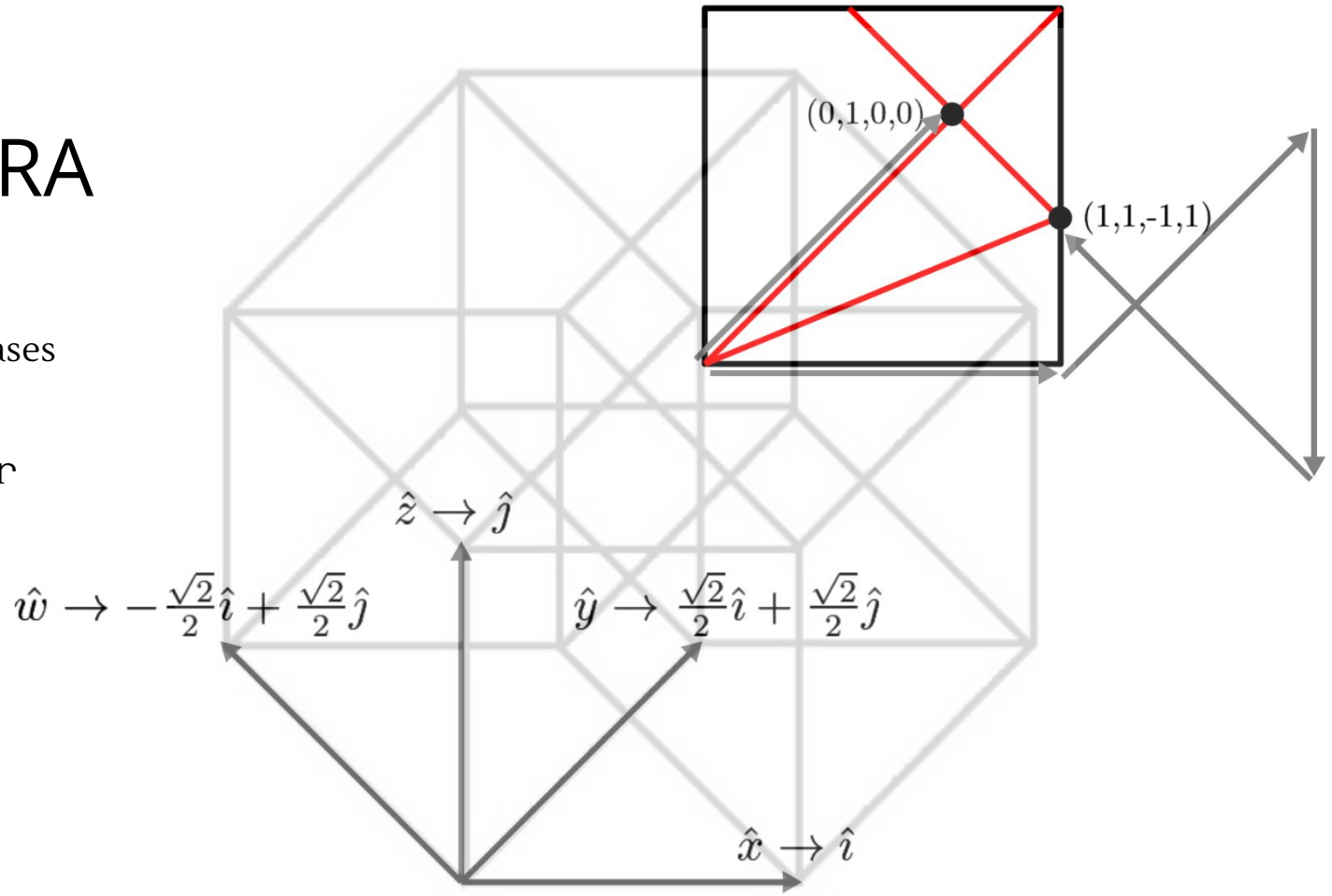
- Constrained non-linear optimization: TreeMaker, pure circle packing
- Mixed-integer optimization: box pleating and 22.5 (stay tuned for Sunday 👀)





ABSTRACT ALGEBRA

- Algebra over fields
  - Ex: binary folded/unfolded creases can be expressed as GF(2)
- 22.5 as a quadratic field extension



OTHER TOPICS (PREVIEW OF 22.5 TALK)

Boolean satisfiability (SAT)

Mixed Integer Linear Programming (MILP)

Linear algebra, computational geometry

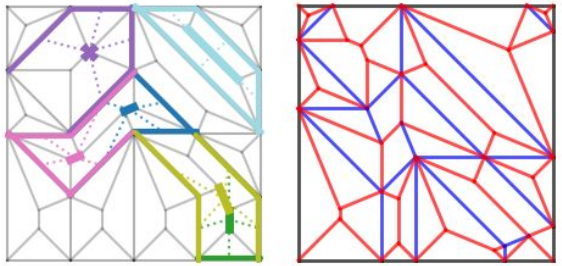
Spectral graph theory

Heat transfer

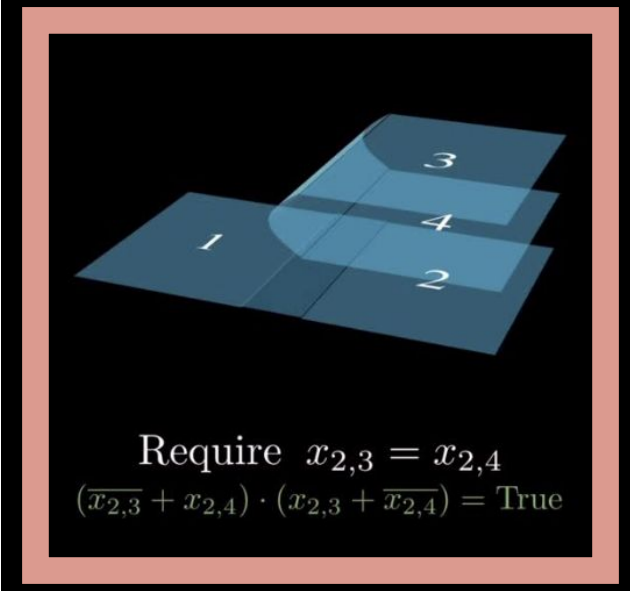
Facebook AI Similarity Search (FAISS)

$$Z(t) = \mathrm{tr}(\Phi e^{-t\Lambda}\Phi^T)$$

$$Z(t) = \sum_{i=1}^{N} e^{-t\lambda_i}$$



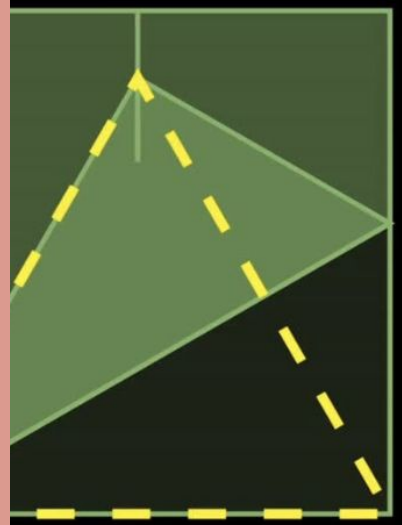
$$\begin{array}{l} \text{Basic constraints} \\ \text{Edge Integrity} \\ \text{Quadruplets} \\ \\ \text{Incircle Clearance} \\ \\ \text{Anti-bowtie} \end{array} \underbrace{\left[\begin{array}{ccccc} A & 0 & 0 & 0 & 0 \\ \hdashline M_{edges} & 0 & 0 & 0 & 0 \\ \hdashline -N_{quad} & C \cdot I_{quad} & N_{incircle} & -I_{quad} & 0 \\ N_{quad} & C \cdot I_{quad} & -N_{incircle} & I_{quad} & 0 \\ \hdashline N_{invader} & C \cdot I_{inv} & -N_{incircle} & -I_{inv} & 0 \\ \hdashline 0 & 0 & 0 & 0 & -S_{sector} \\ N_{bowtie} & 0 & 0 & 0 & C \cdot I_{bowtie} \end{array}\right]}_{M} \underbrace{\left[\begin{array}{c} \vec{x} \\ \hdashline \vec{z}_{quads} \\ \vec{P}_{incircles} \\ \vec{r}_{incircles} \\ \vec{b}_{bowtie} \end{array}\right]}_{\vec{X}} \leq \underbrace{\left[\begin{array}{c} \mathbf{0} \\ \hdashline -\boldsymbol{\epsilon} \\ \mathbf{C} \\ \mathbf{C} \\ \mathbf{C} \\ -\mathbf{1} \\ \mathbf{C} - \boldsymbol{\epsilon} \end{array}\right]}_{\vec{B}}$$



189K

2.9M

- **Tortilla-tortilla constraint** (i, j, k, l): Creases e_{ij}, e_{kl} overlap, so there are points $p \in e_{ij}$ and $q \in e_{kl}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $P' \subset F_i \cup F_j \cup e_{ij}$ and $Q' \subset F_k \cup F_l \cup e_{kl}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P' \cap e_{ij}$, $q' \in Q' \cap e_{kl}$ such that $f(p') = f(q')$. For ε' sufficiently small, there are points $p^+ \in F_i \cap P'$, $q^+ \in F_k \cap Q'$ with $f(p^+) = f(q^+)$ and points $p^- \in F_j \cap P'$, $q^- \in F_l \cap Q'$ with $f(p^-) = f(q^-)$ such that the distances from p^+, p^- to p' along P', and from q^+, q^- to q' along Q', are all ε'. Then $\lambda(p^+, q^+) = \lambda(p^-, q^-)$ by the tortilla-tortilla condition. Because $p^+ \in F_i$, $p^- \in F_j$, $q^+ \in F_k$, and $q^- \in F_l$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jl}$.
- **Taco-tortilla constraint** (i, j, k, k): F_k and crease e_{ij} overlap, so there are points $p \in F_k$ and $q \in e_{ij}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $P' \subset F_k$ and $Q' \subset F_i \cup F_j \cup e_{ij}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P', q' \in Q' \cap e_{ij}$ such that $f(p') = f(q')$. For ε' sufficiently small there are points $q^+ \in F_i \cap Q', q^- \in F_j \cap Q', p^+ \in F_k \cap P'$ with $f(q^+) = f(q^-) = f(p^+)$ such that the distances from p^+ to p' along P', and from q^+, q^- to q' along Q', are all ε'. Then $\lambda(q^+, p^+) = \lambda(q^-, p^+)$ by the taco-tortilla condition. Because $q^+ \in F_i$, $q^- \in F_j$, and $p^+ \in F_k$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jk}$.
- **Taco-tortilla constraint** (i, j, k, l): Creases e_{ij}, e_{kl} overlap, so there are points $q \in e_{ij}$ and $p \in e_{kl}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $Q' \subset F_i \cup F_j \cup e_{ij}$ and $P' \subset F_k \cup F_l \cup e_{kl}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P' \cap e_{kl}$, $q' \in Q' \cap e_{ij}$ such that $f(p') = f(q')$. For ε' sufficiently small, there are points $q^+ \in F_i \cap Q', q^- \in F_j \cap Q', p^+ \in F_k \cap P'$ with $f(q^+) = f(q^-) = f(p^+)$ such that the distances from p^+ to p' along P', and from q^+, q^- to q' along Q', are all ε'. By the taco-tortilla condition, $\lambda(p^+, q^+) = \lambda(p^-, q^-)$. Because $p^+ \in F_i$, $p^- \in F_j$, and $q^+ \in F_k$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jl}$.



equilateral triangle.

A BRIDGE GOES BOTH WAYS

Math -> Origami

- Bring math nerds into origami by applying concepts they are interested in

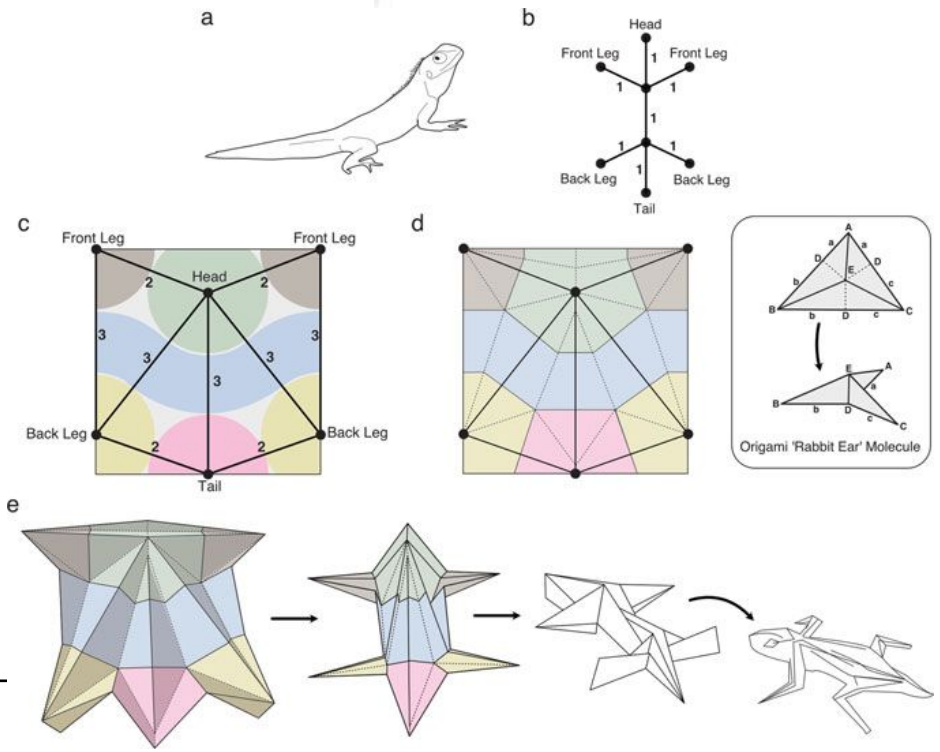
Origami -> Math

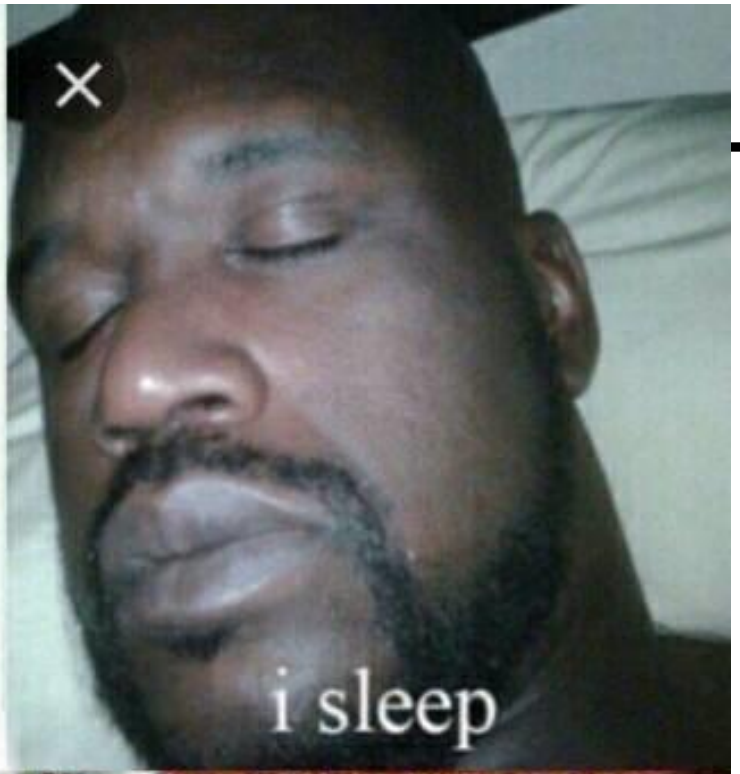
- Teach math to origami nerds by motivating/contextualizing with origami

Math world

Origami world

$$\min_{w,b,\xi} \quad \frac{1}{2} w^t w + C \sum_{i=1}^{N} \xi_i$$
$$\text{s.t.} \quad y_i(w\phi(x_i + b)) + \xi_i - 1$$
$$\xi \geq 0$$





EXAMPLE LESSON

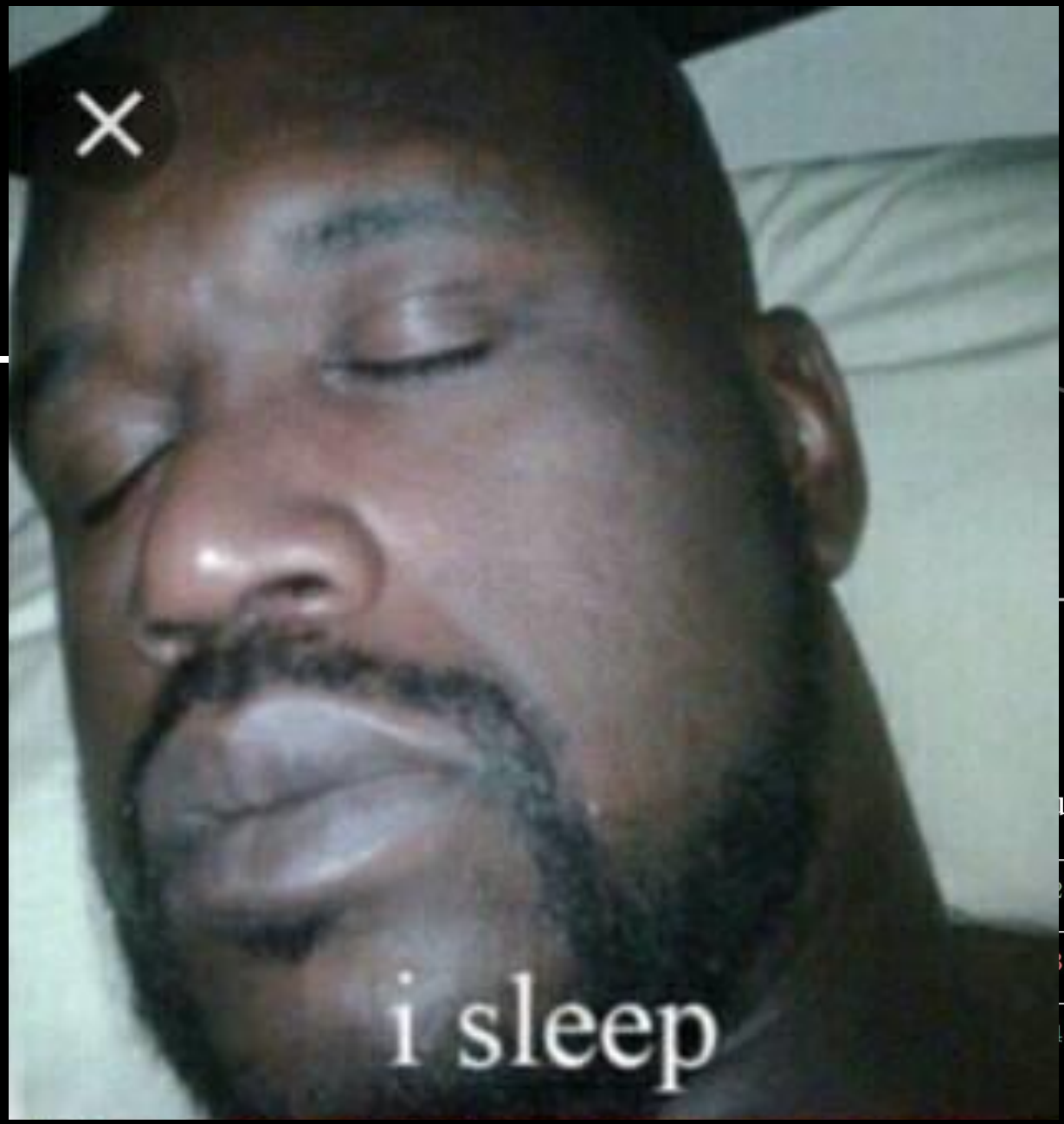
What makes this more compelling than a typical math lesson?

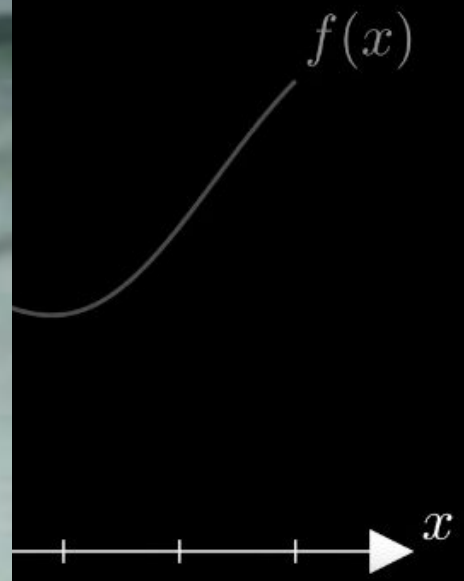
How could this get a math student interested in origami?

CONSTRAI
D NONLIN
OPTIMIZAT
N

Brandon Wong

CFC6, Ann Arbor





maximize σ

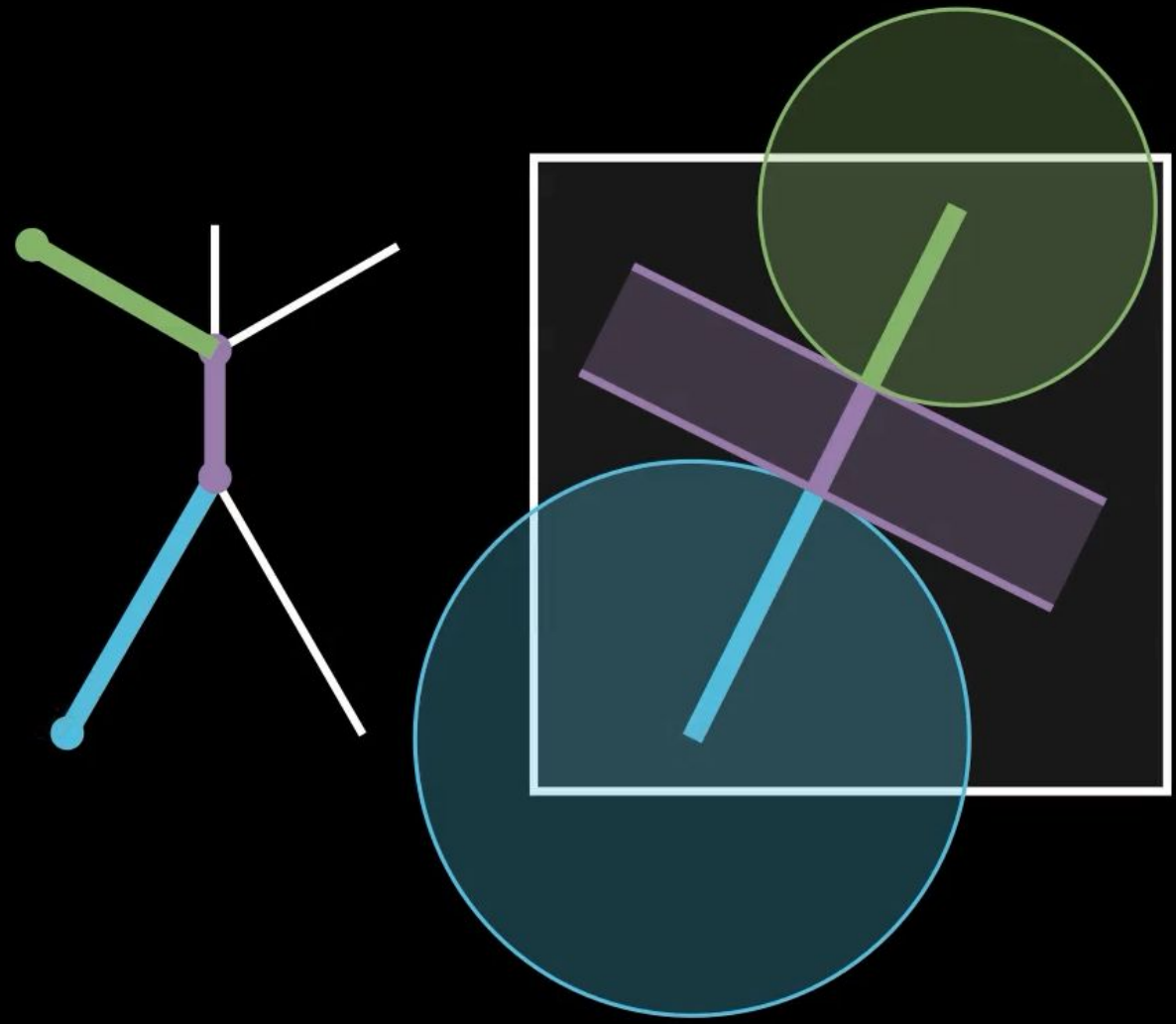
alities:

$$\sqrt{(\quad - y_1)^2}$$

$$\sqrt{(\quad - y_1)^2}$$

$$\sqrt{(\quad - y_1)^2}$$

UN*PACKING* THE MATH OF TREEMAKER

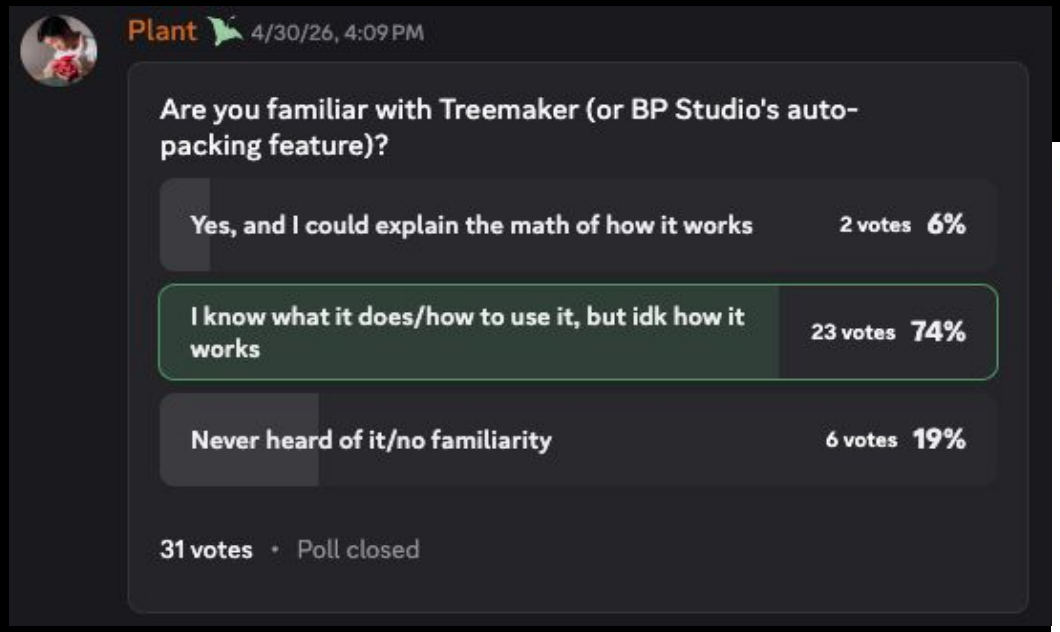


Brandon Wong

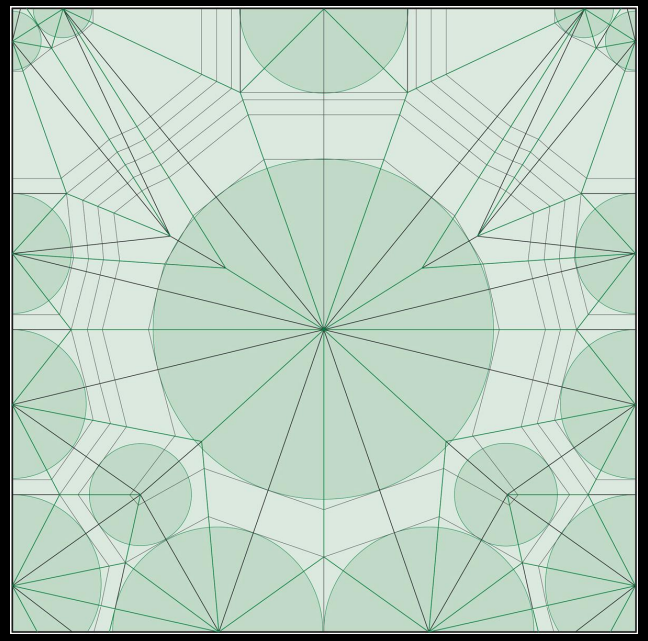
CFC6, Ann Arbor

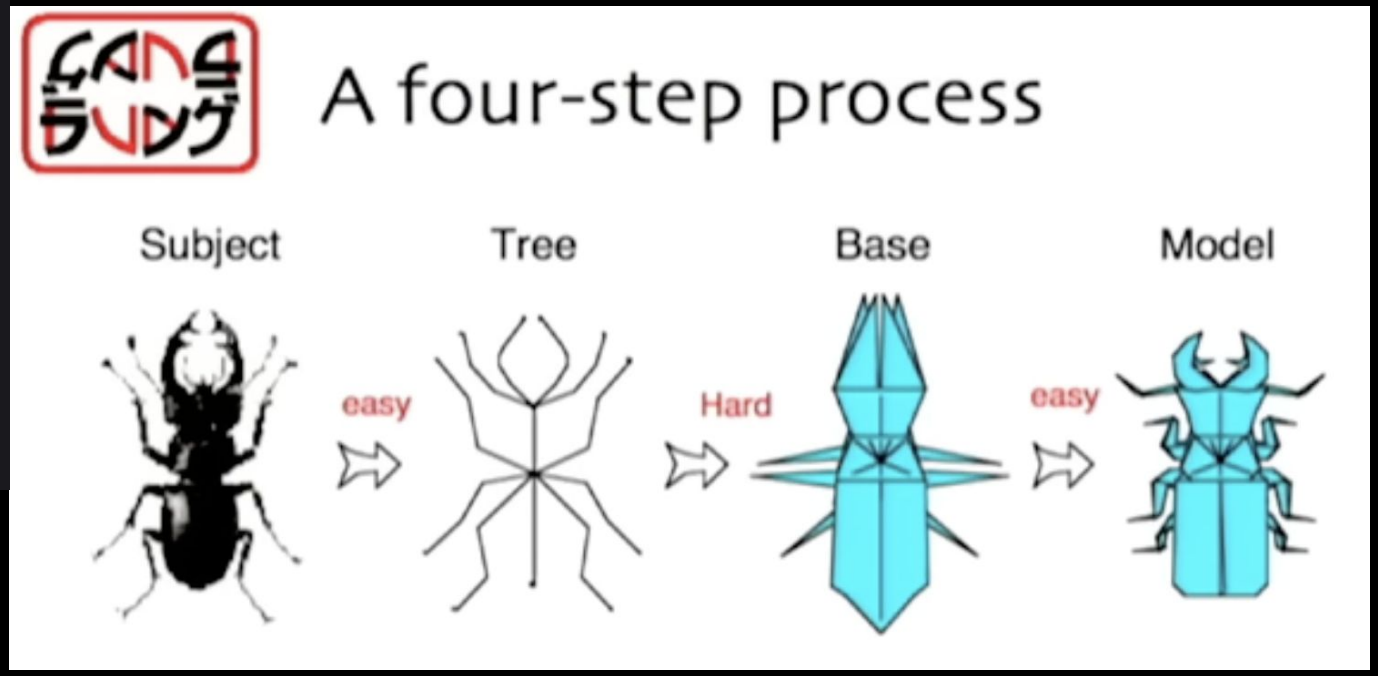
Background

Treemaker has been around for around 20 years, yet very few users know how it works



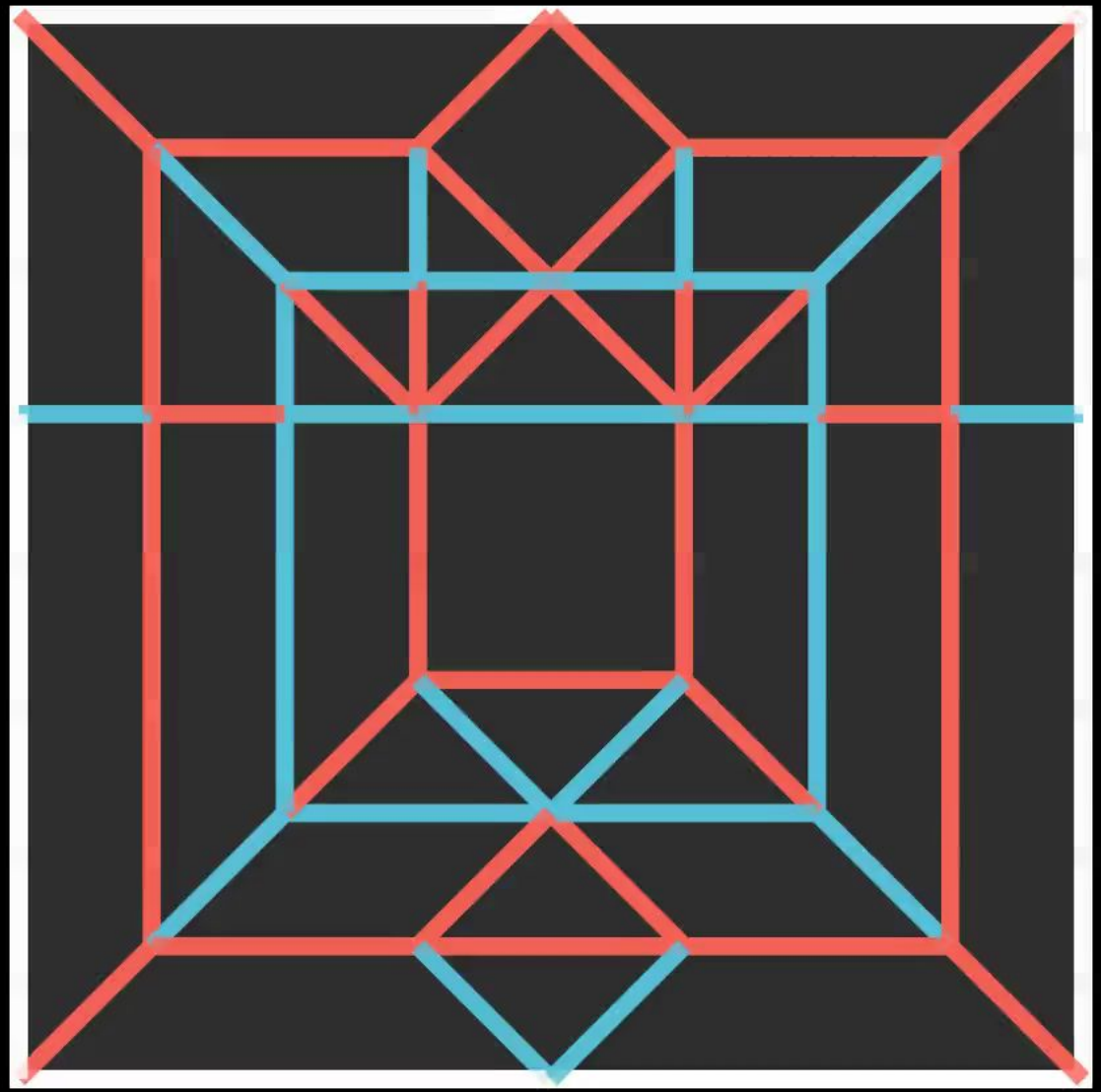
Poll conducted in Origami-Dan (mostly young advanced folders)

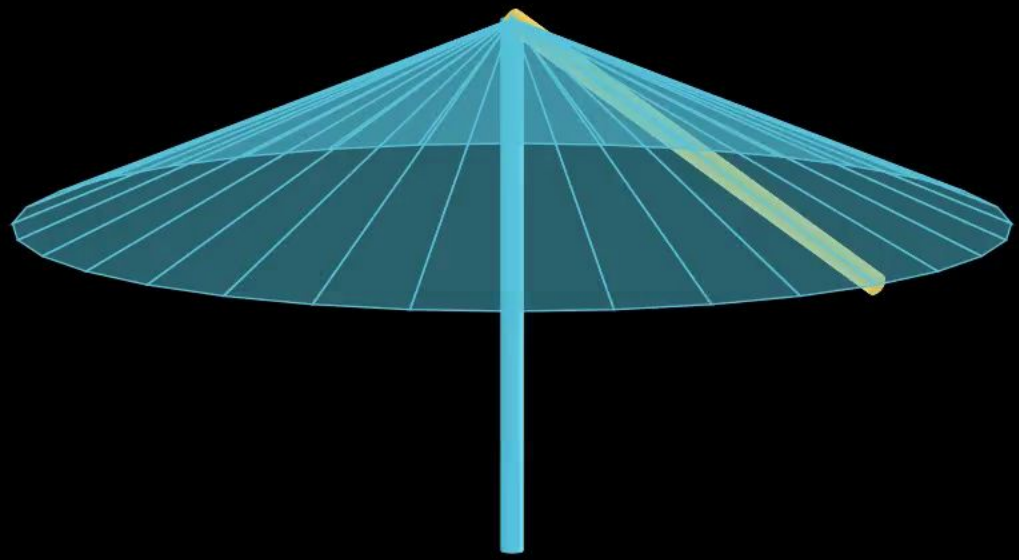


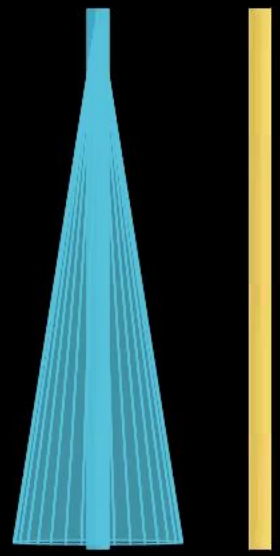


Problem formulation

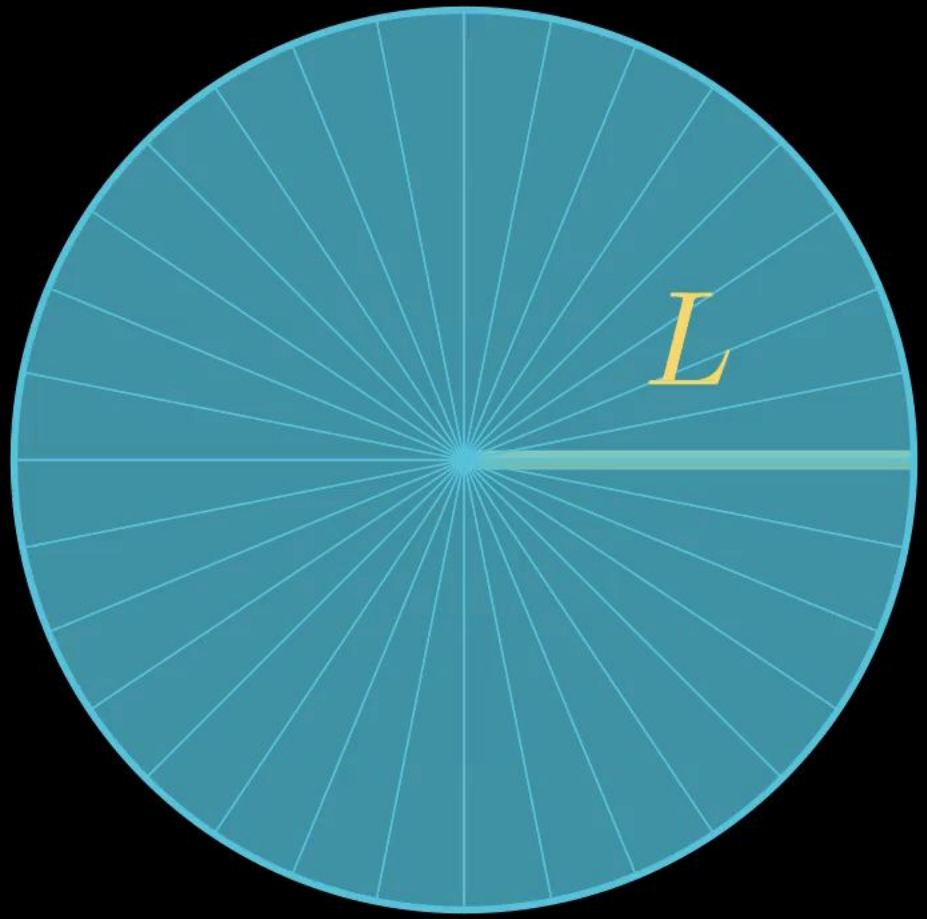
Tree -> optimization problem

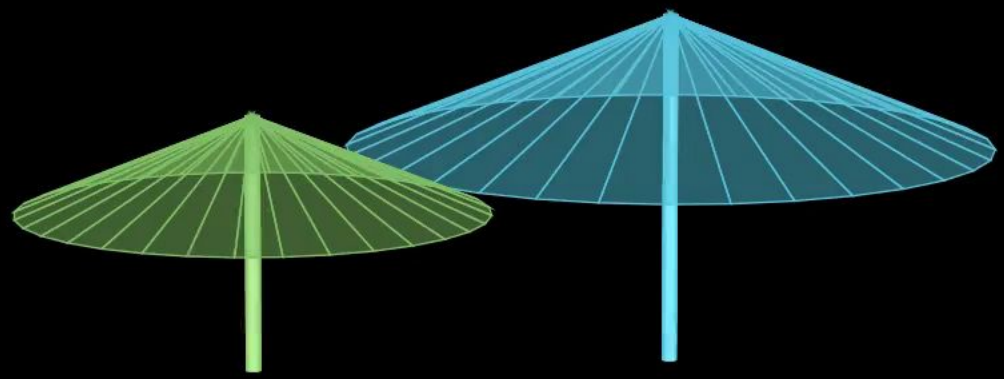


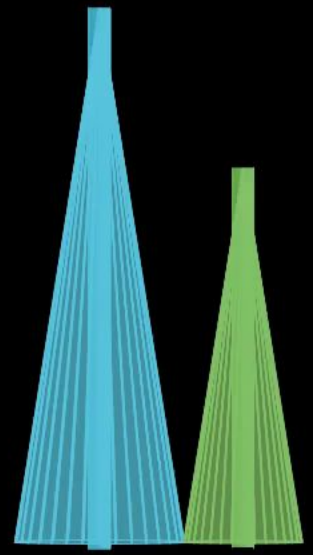


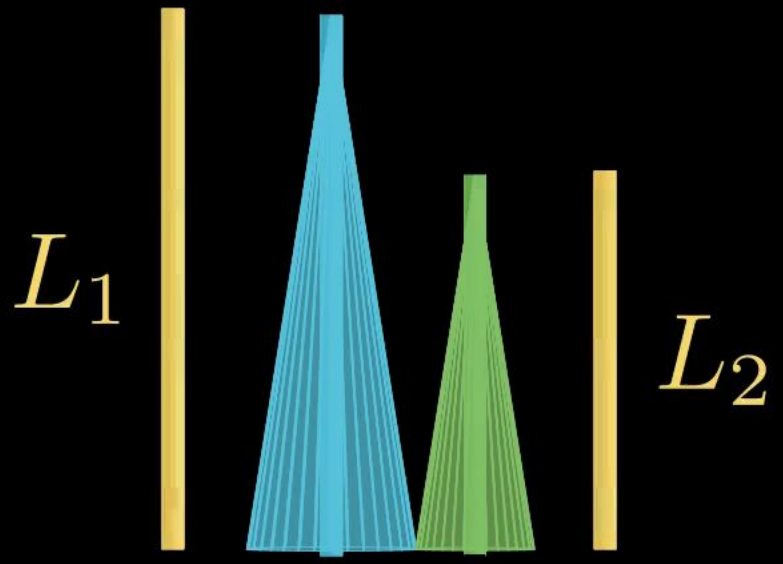


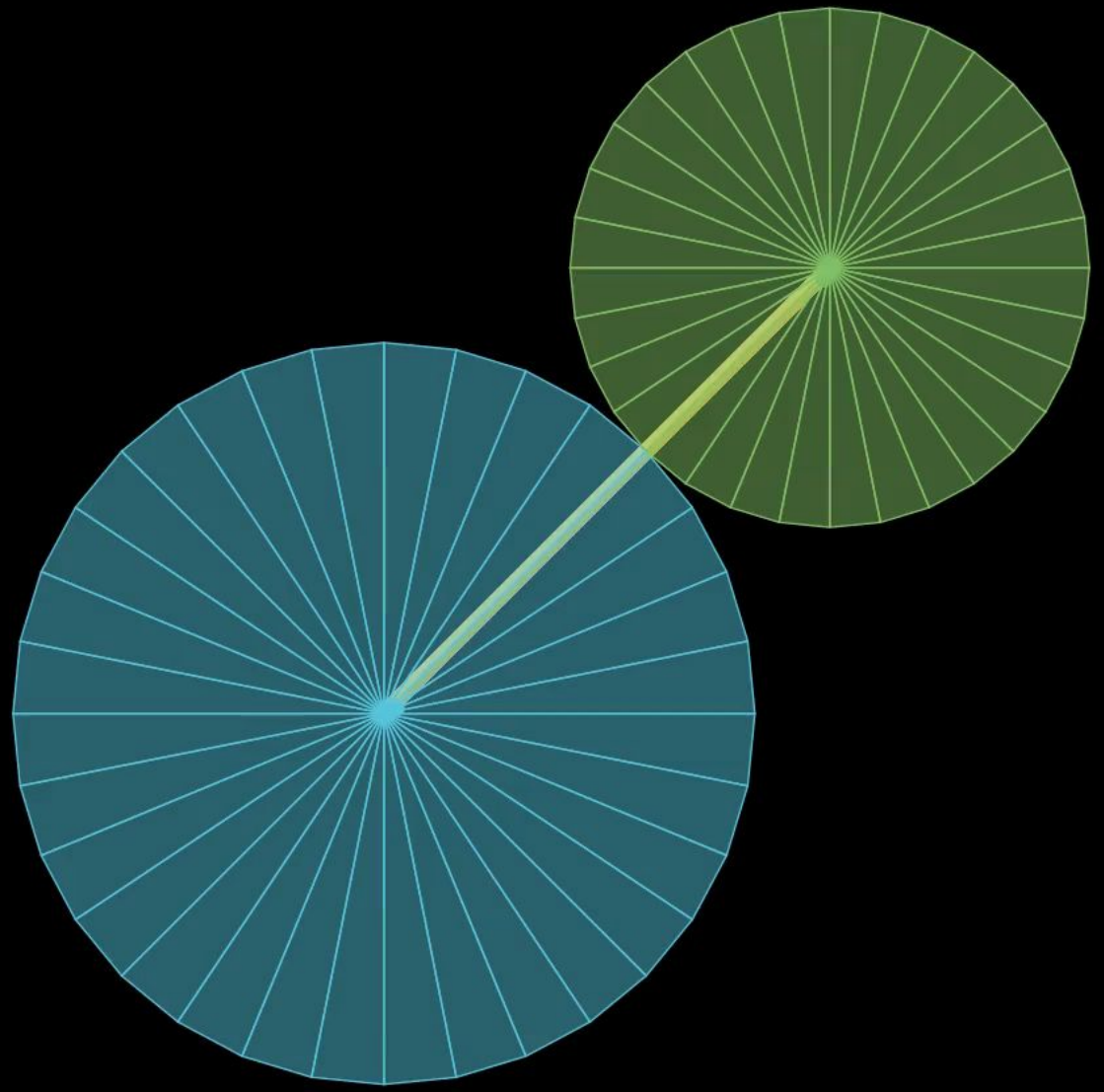

L

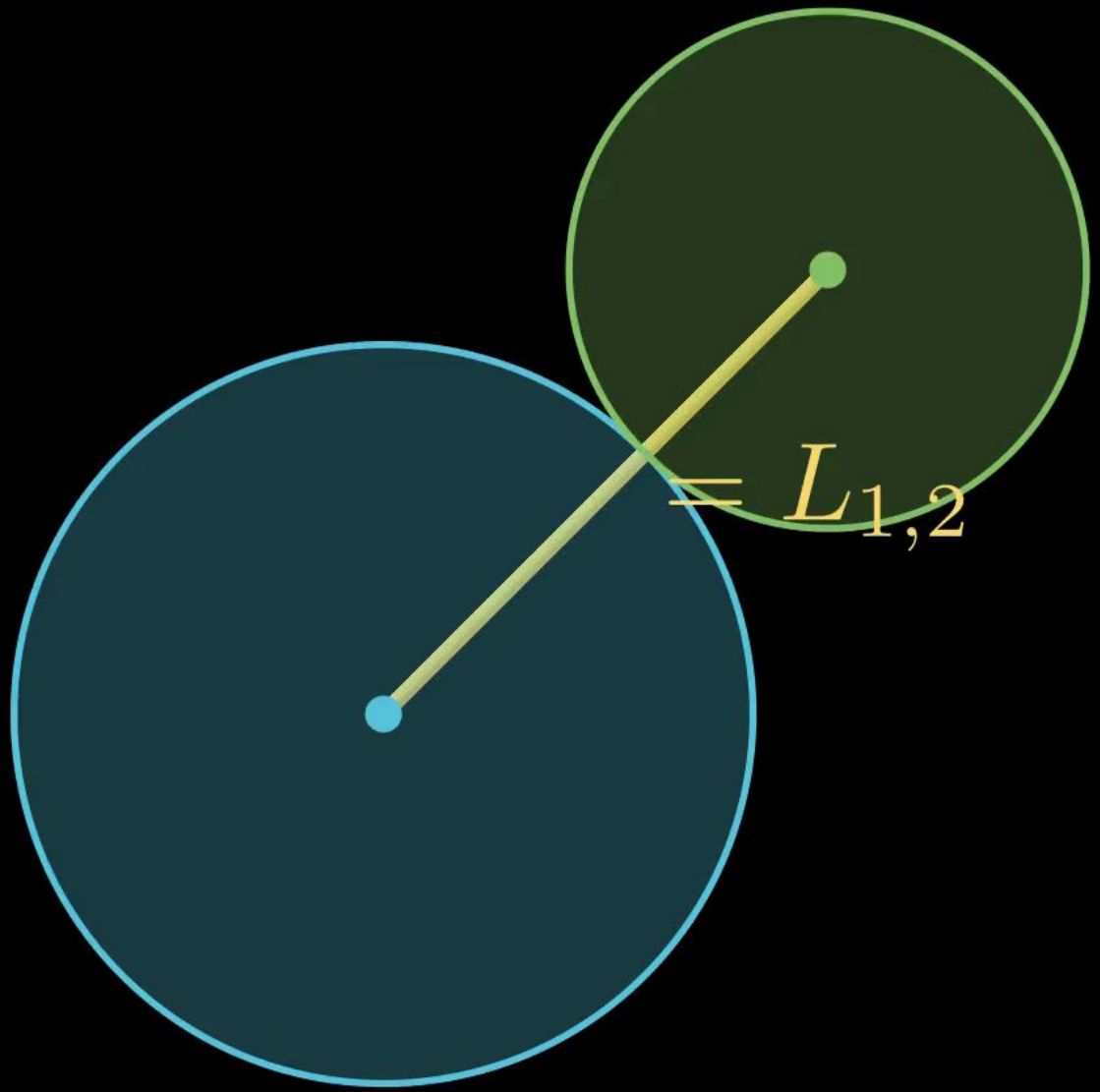

L

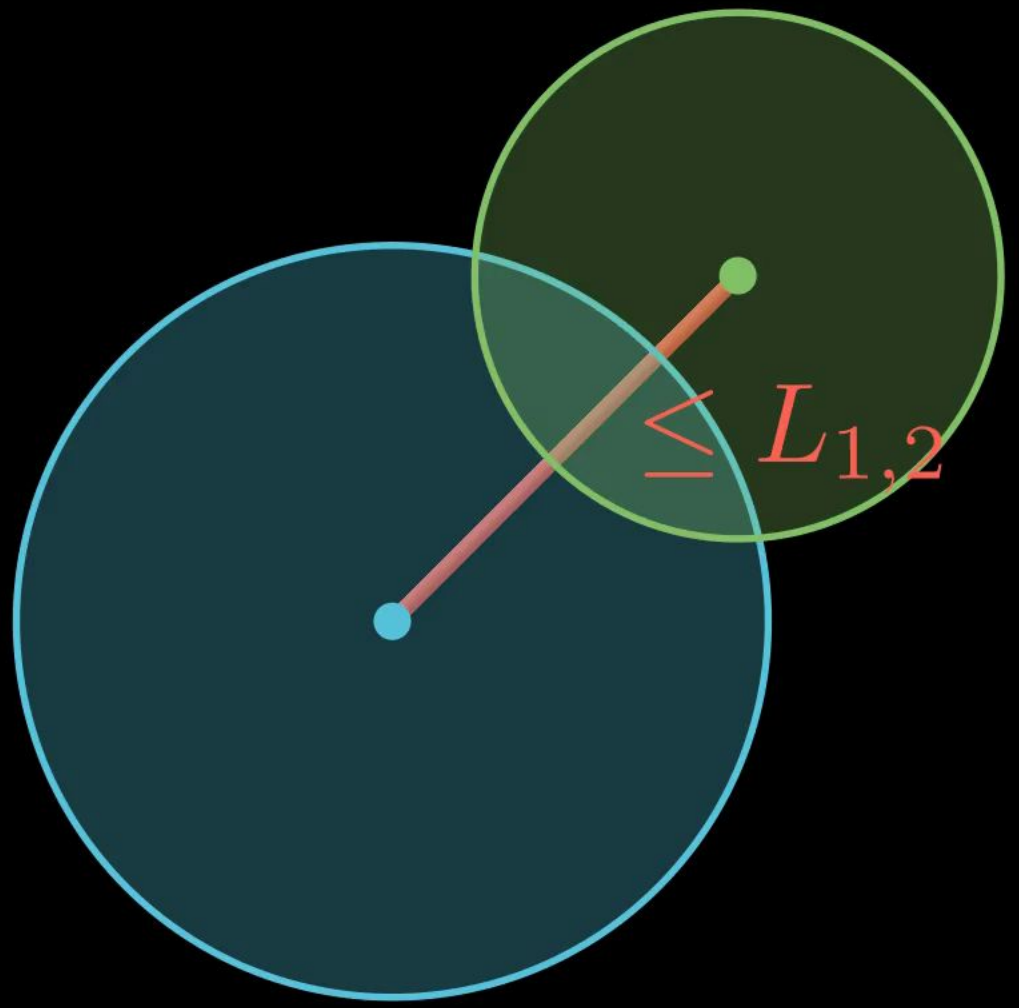


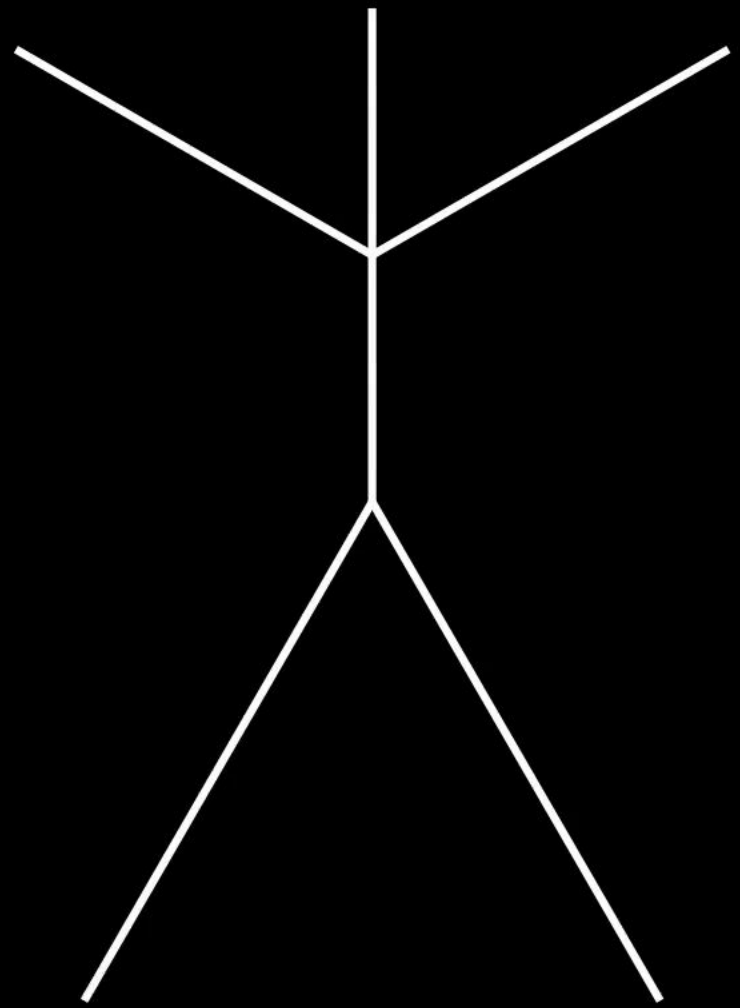


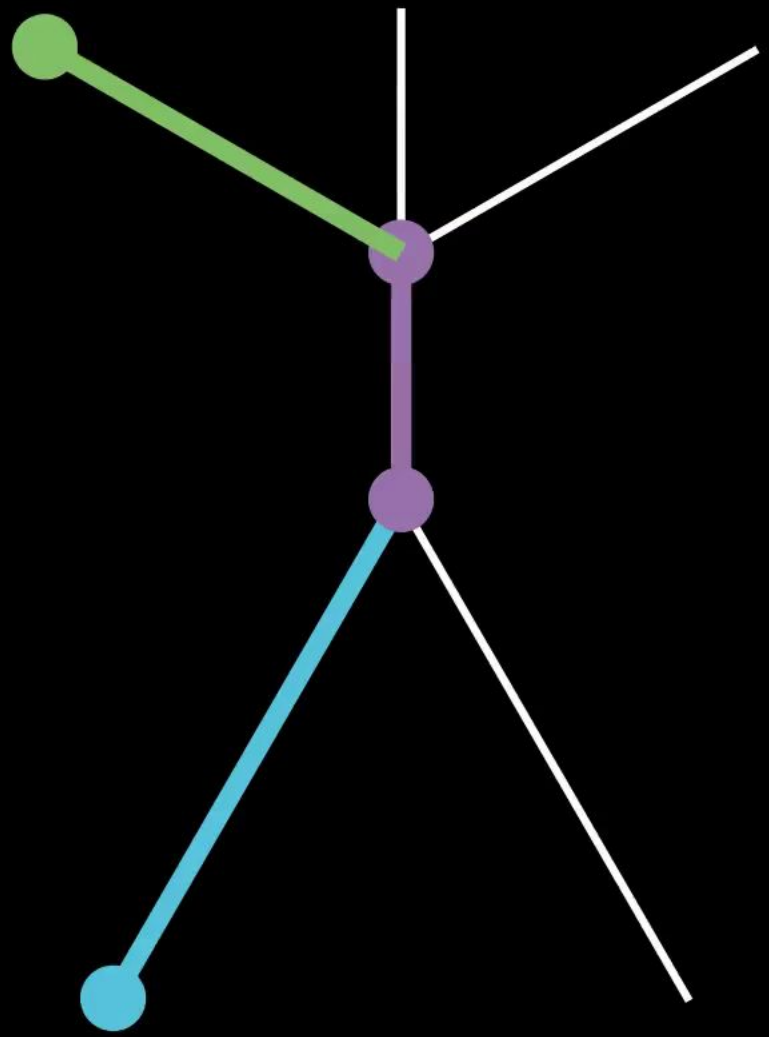

L_1
L_2

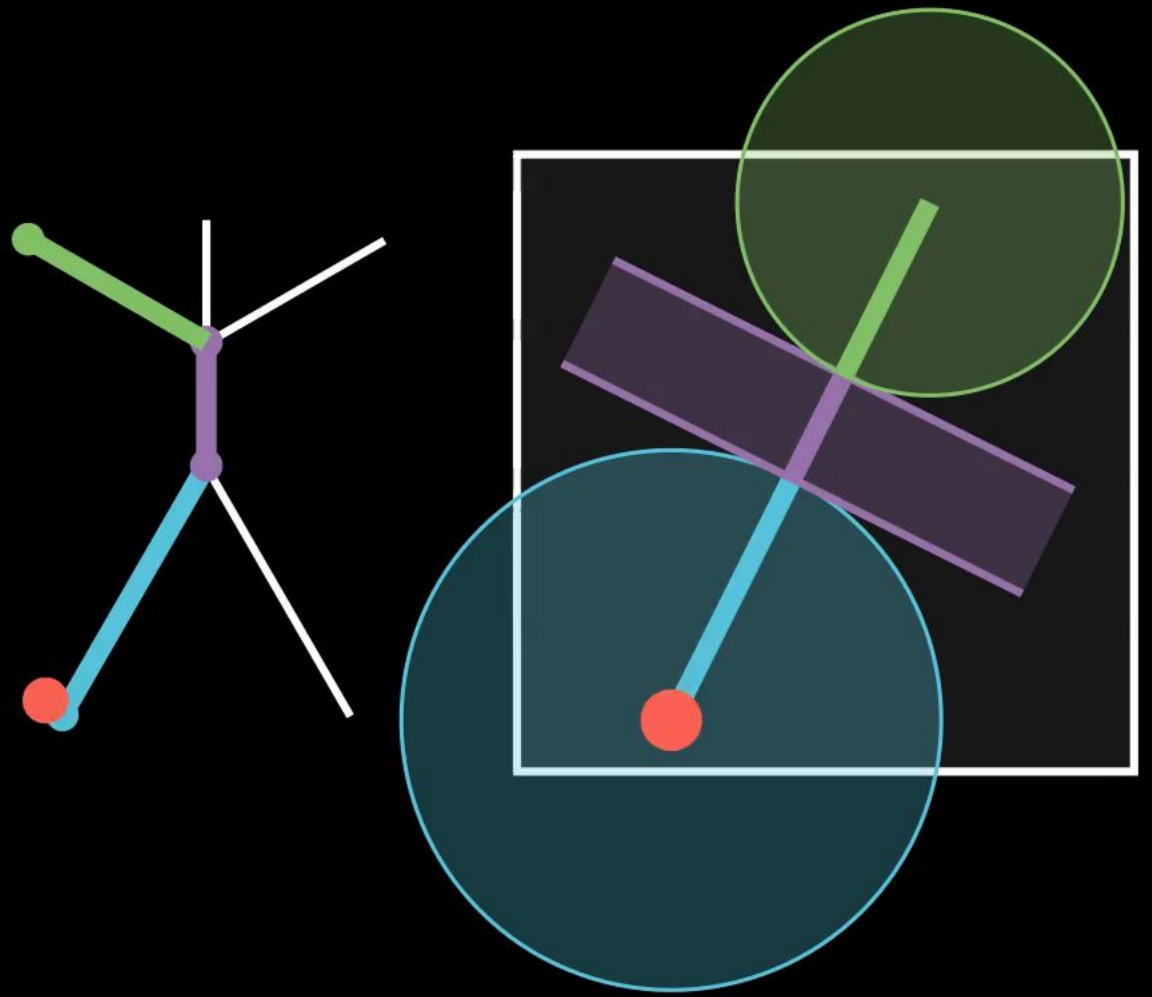


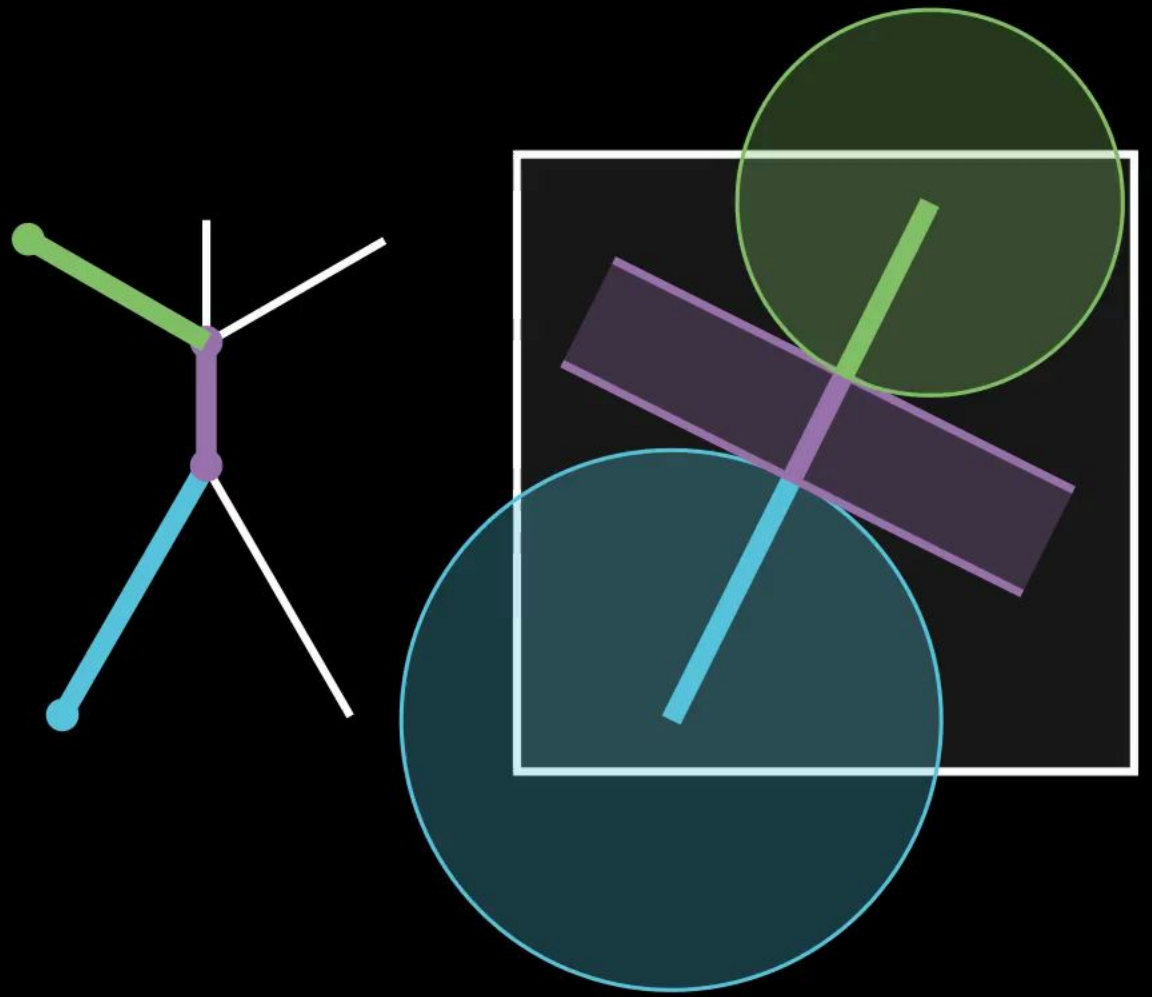

$= L_{1,2}$

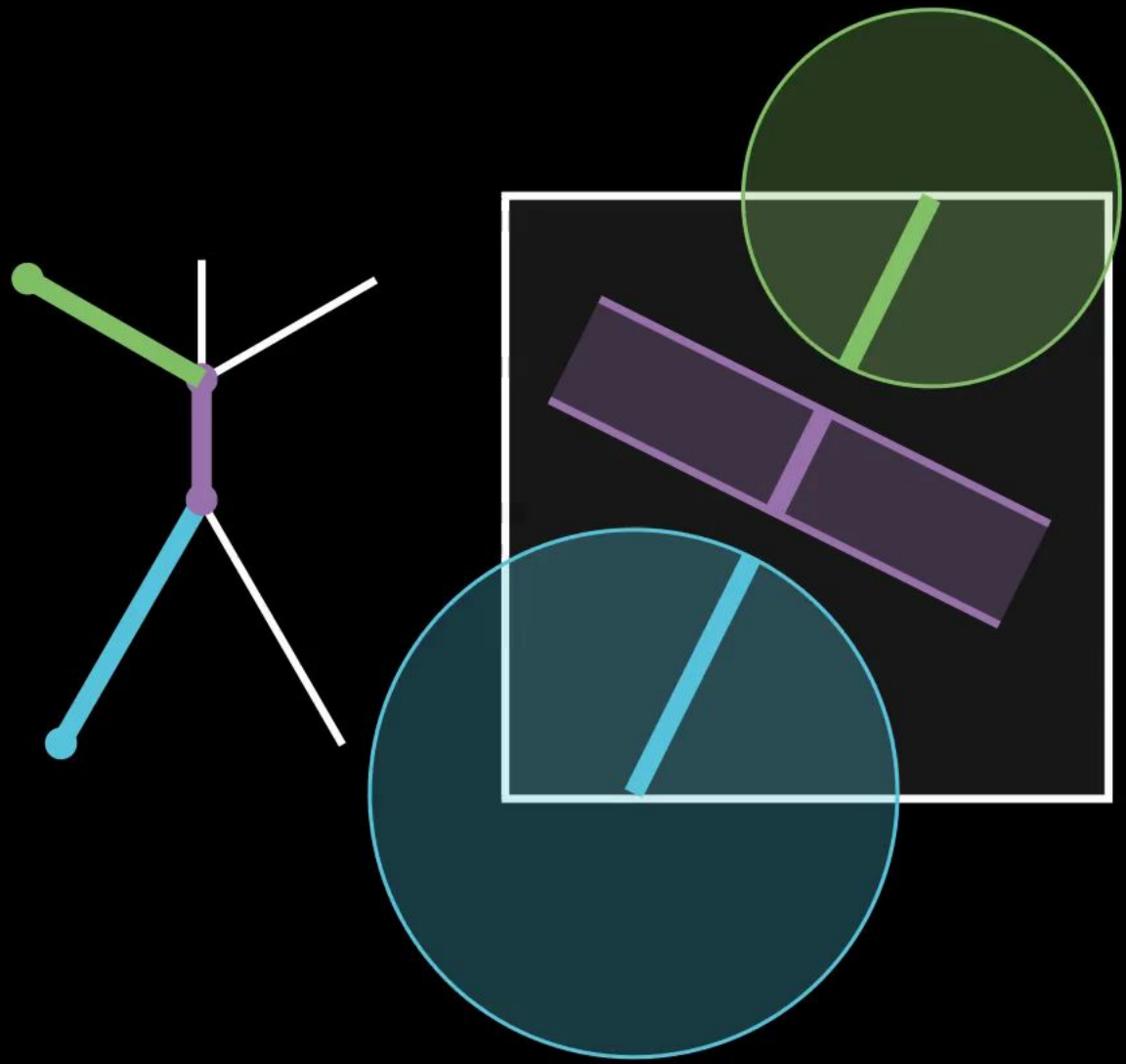

$\leq L_{1,2}$

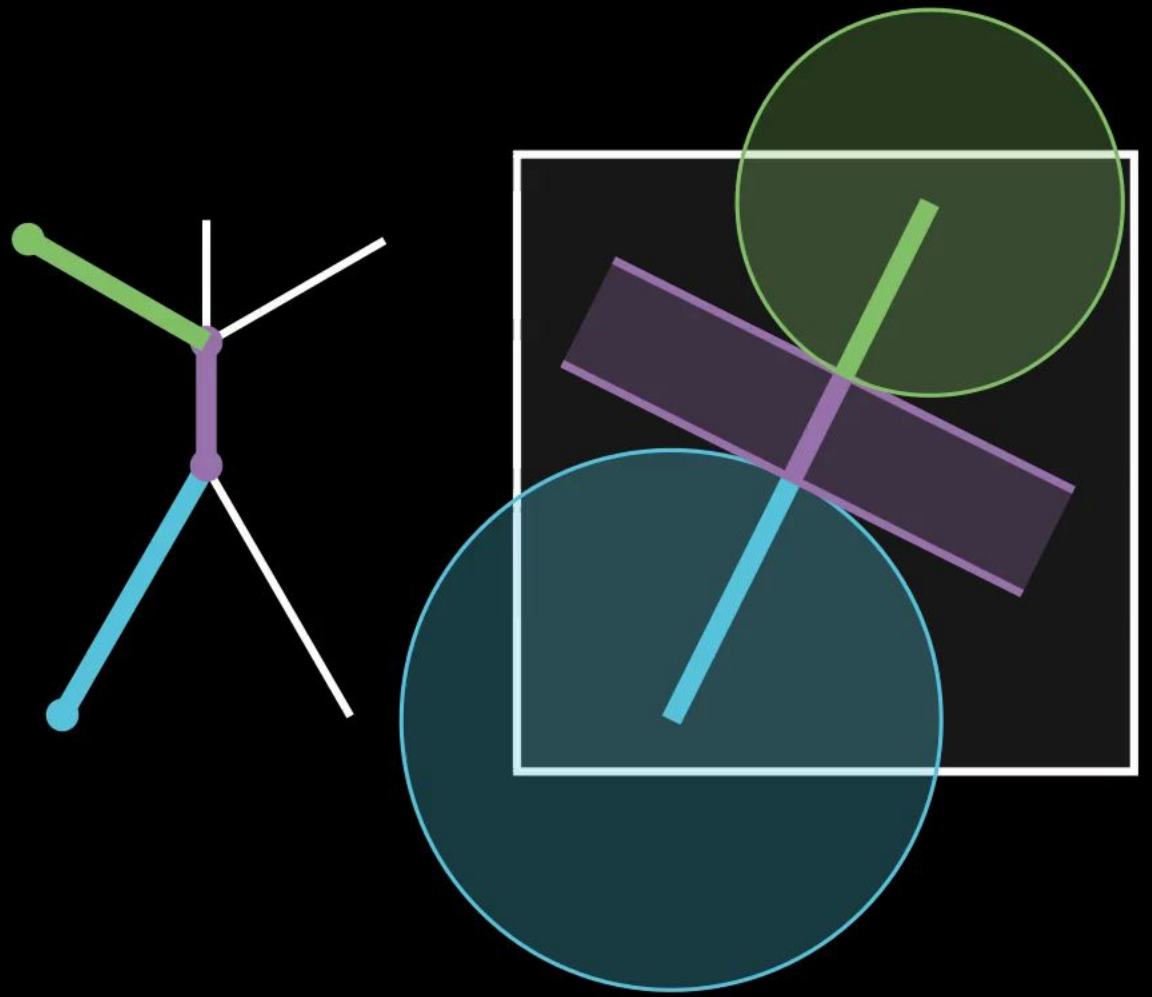


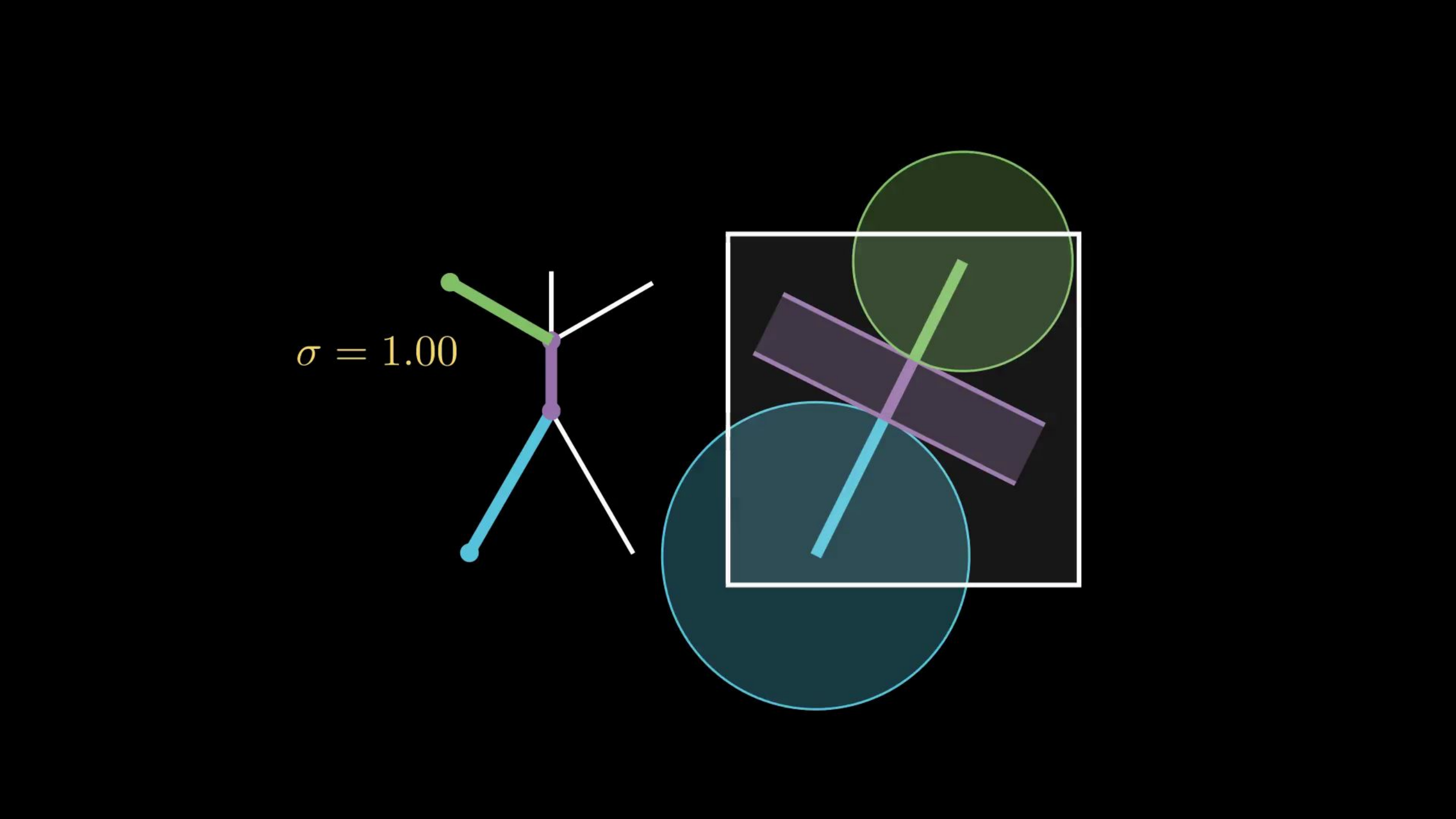


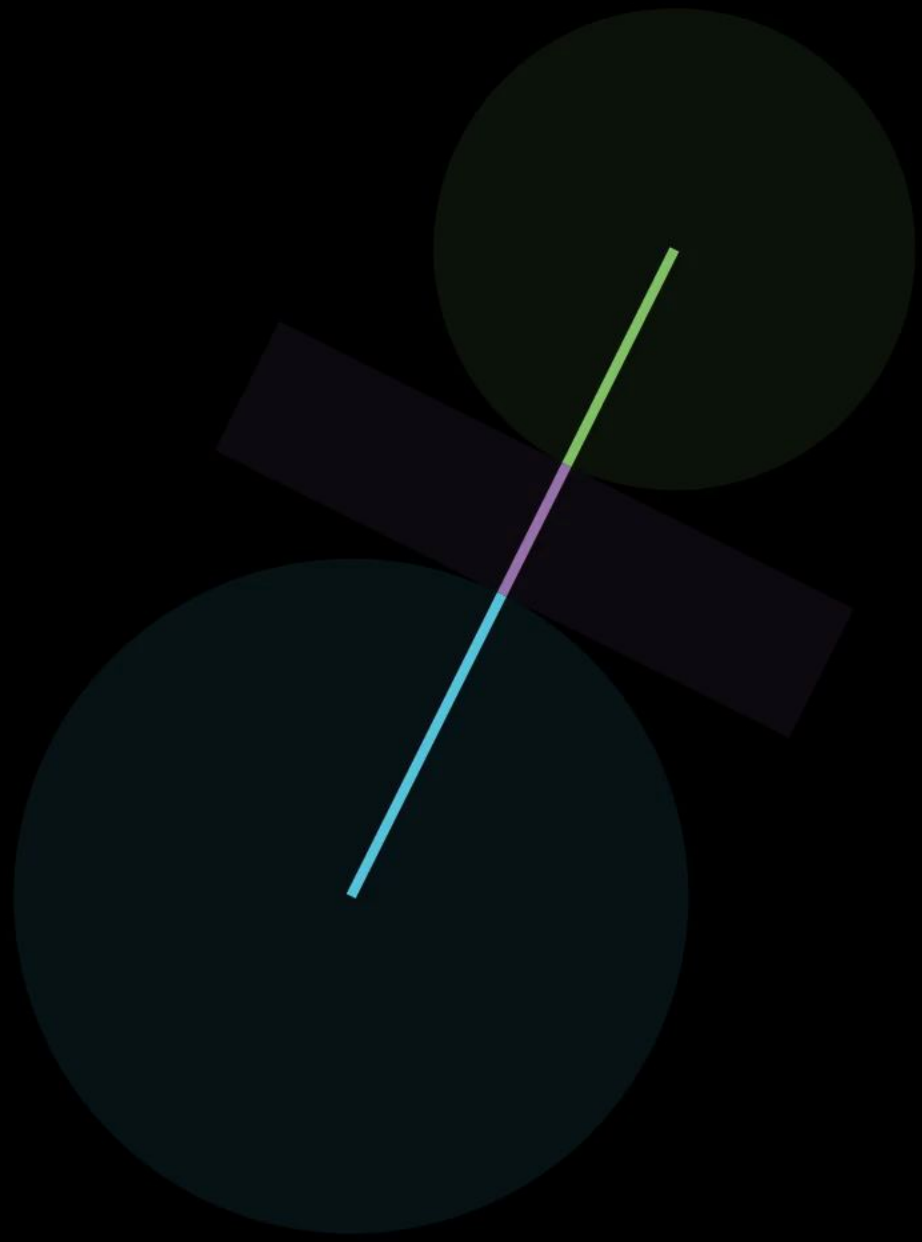


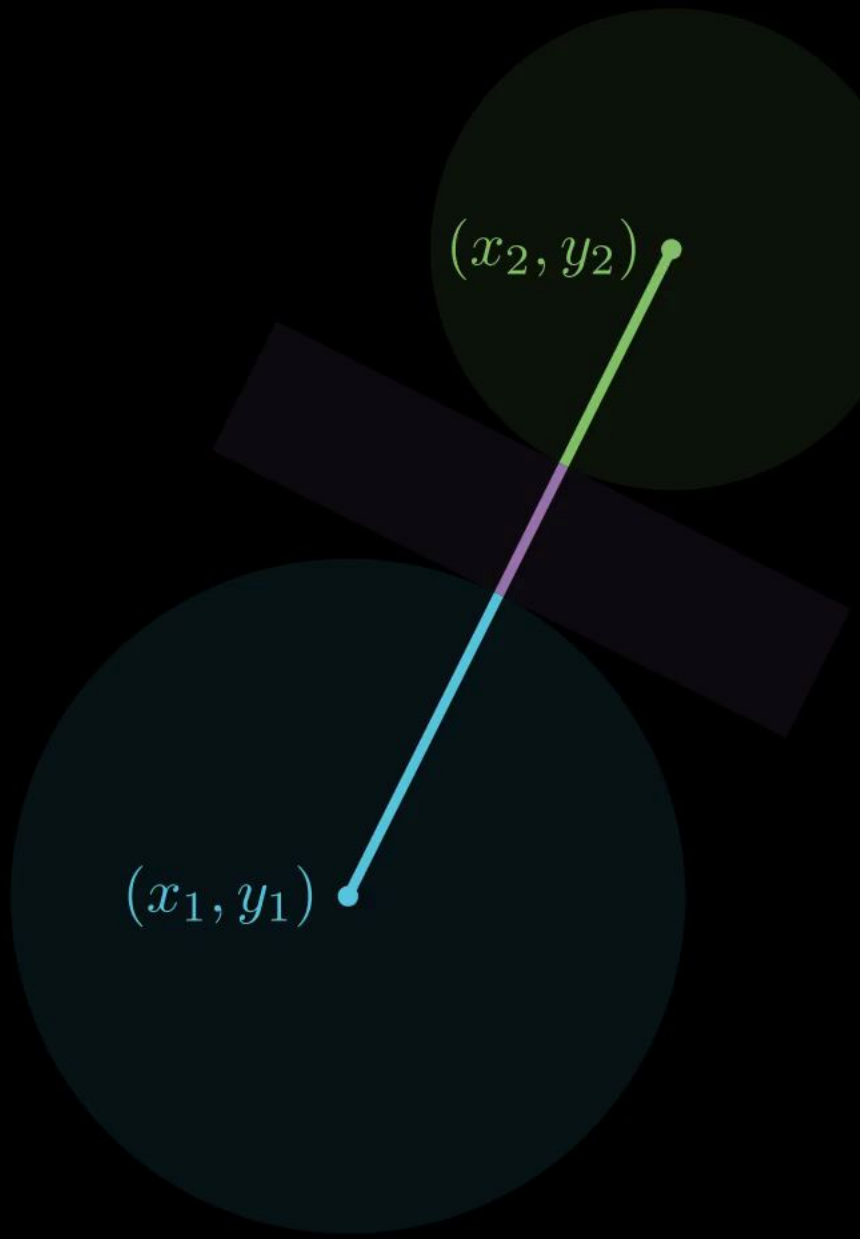


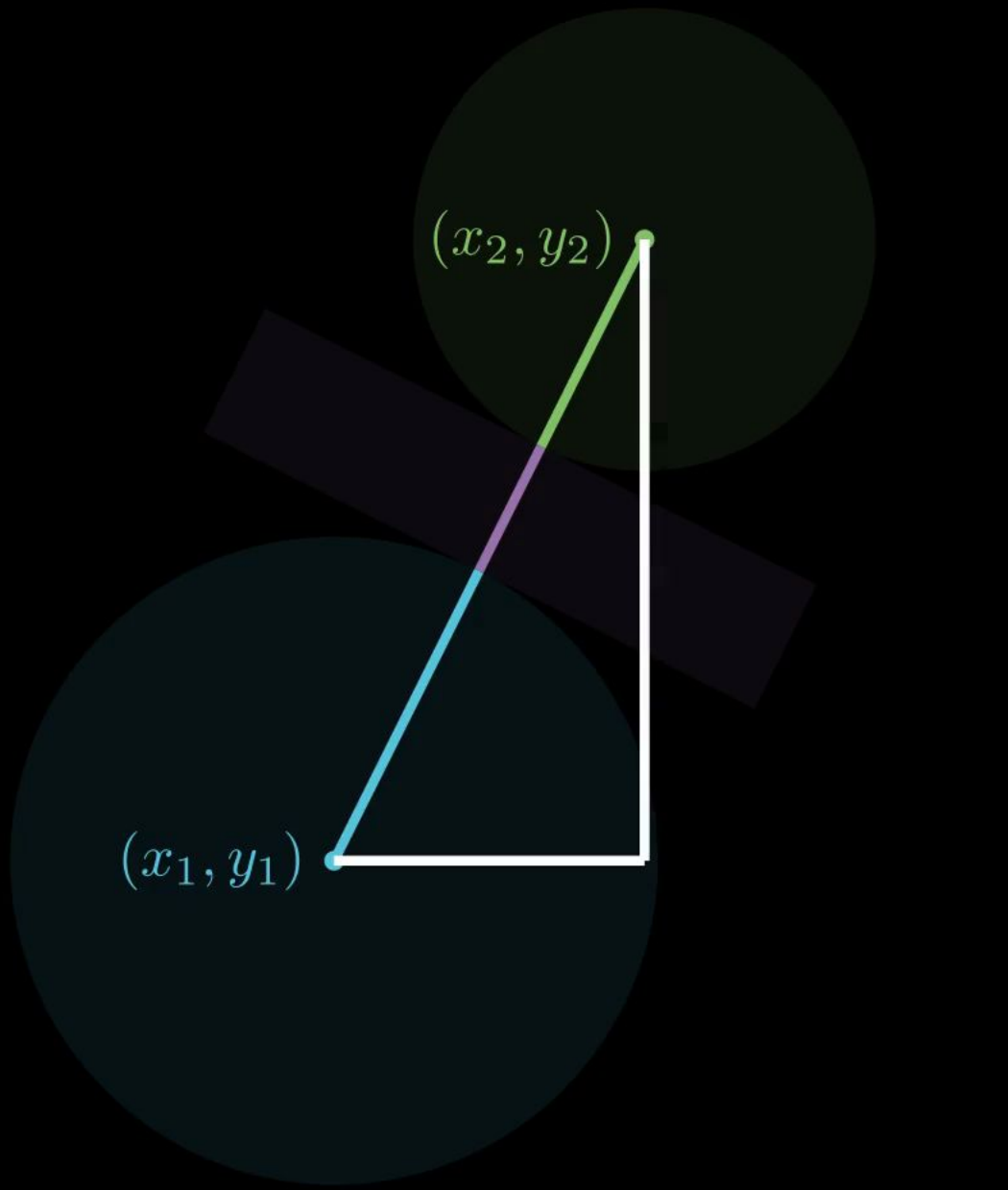


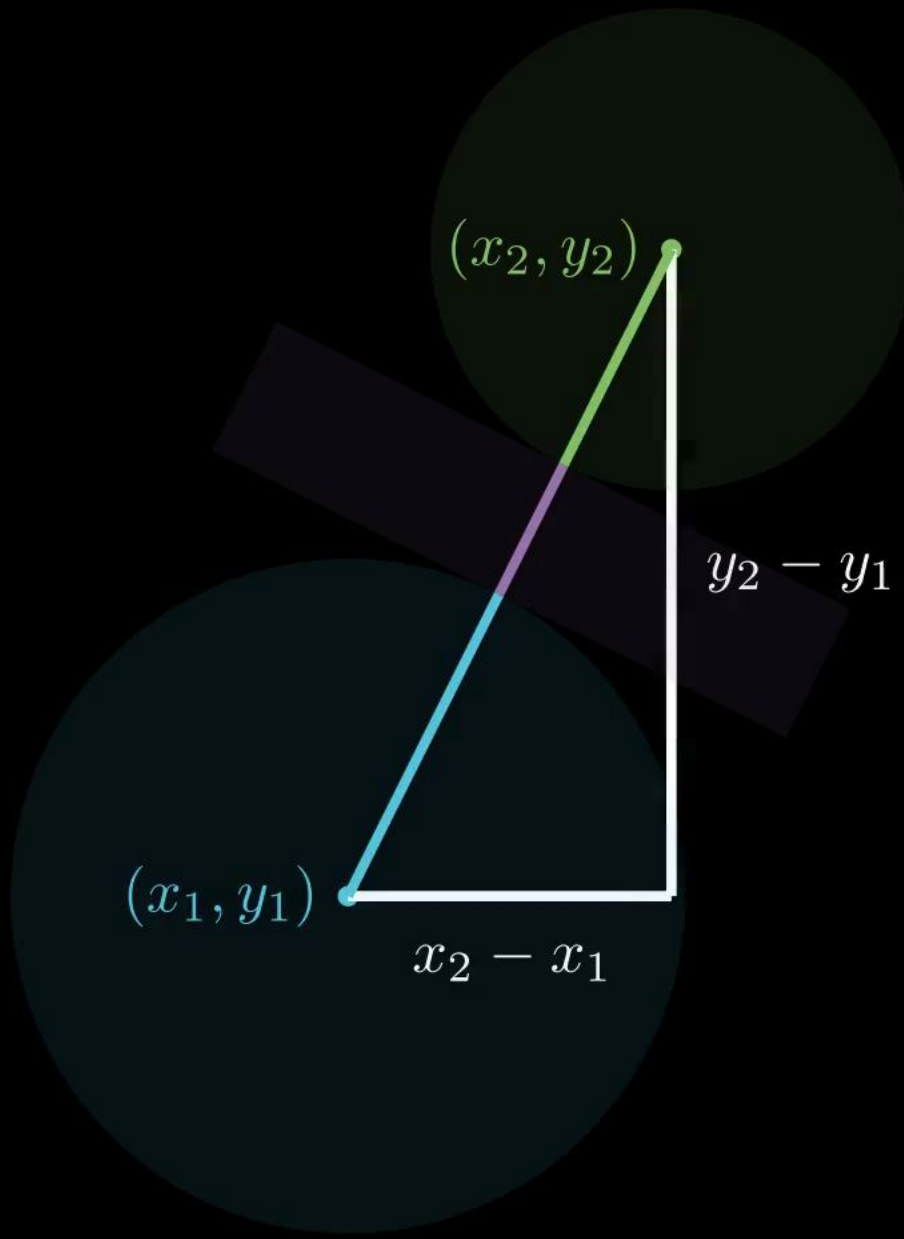



$\sigma = 1.00$

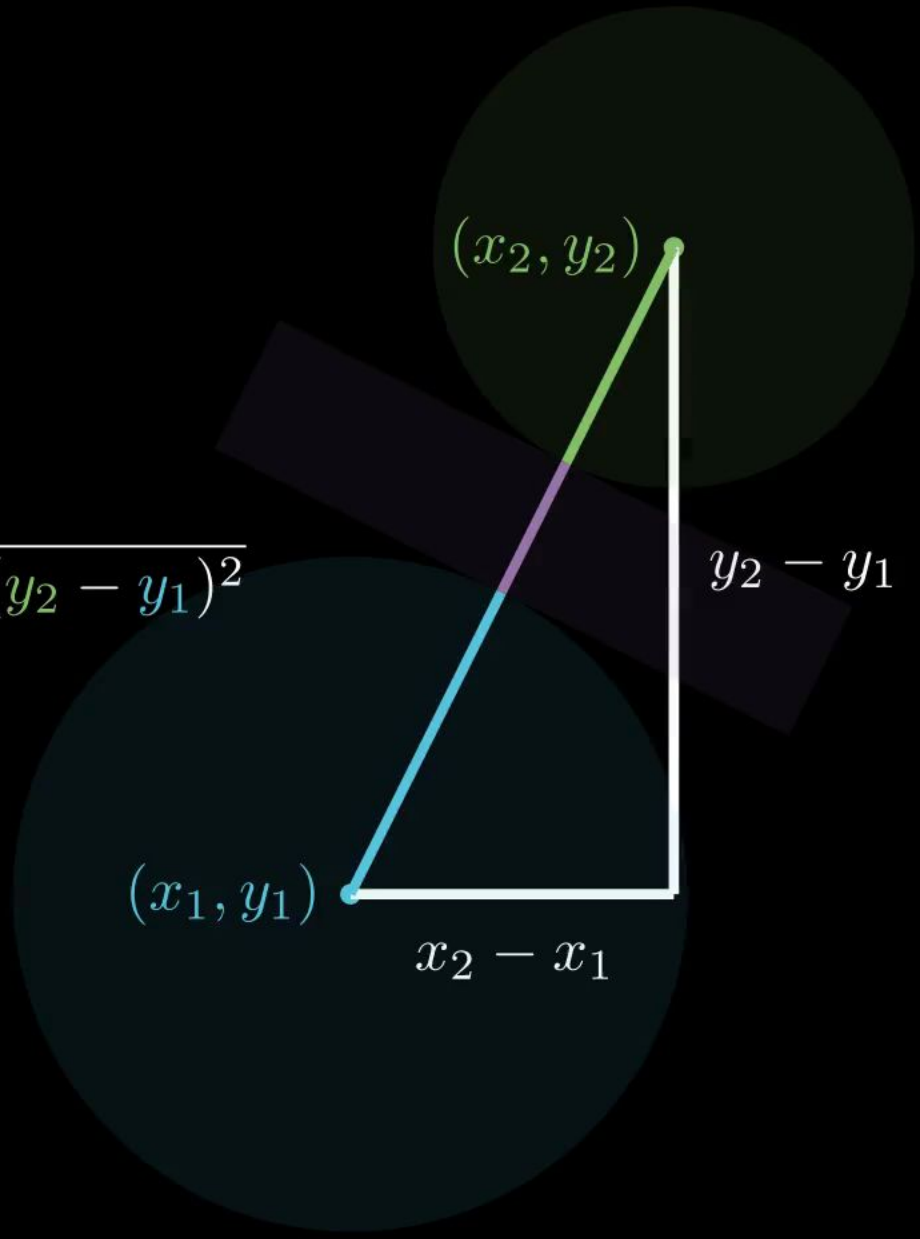



(x_2, y_2)
(x_1, y_1)

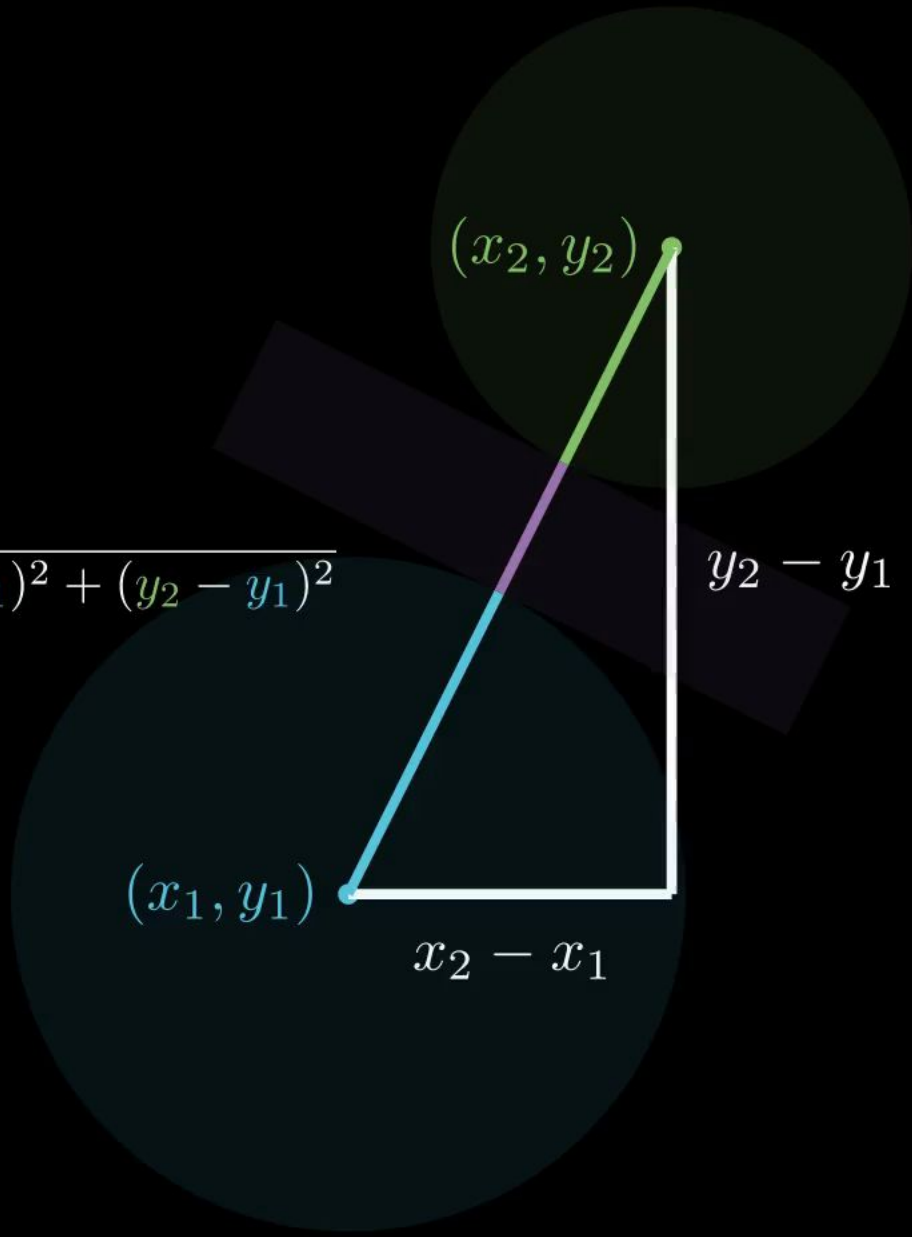

(x_2, y_2)
(x_1, y_1)


(x_2, y_2)
$y_2 - y_1$
(x_1, y_1)
$x_2 - x_1$

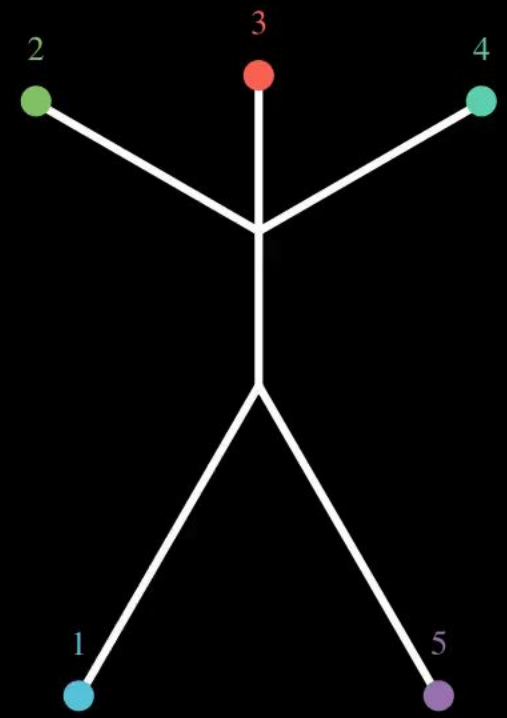
$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

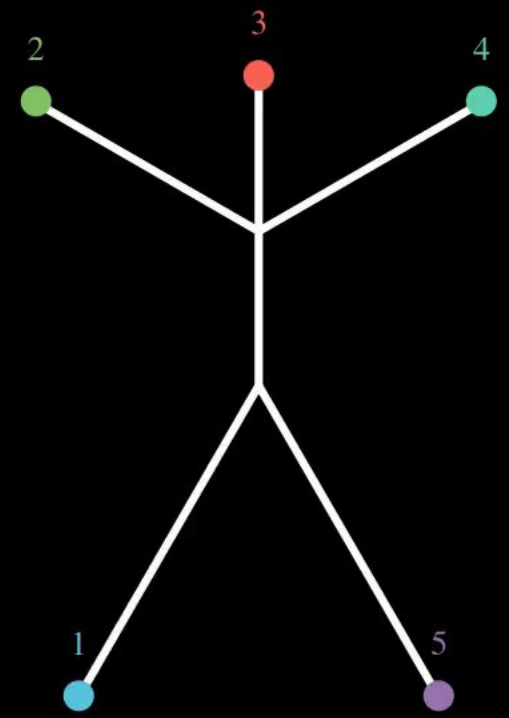


$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

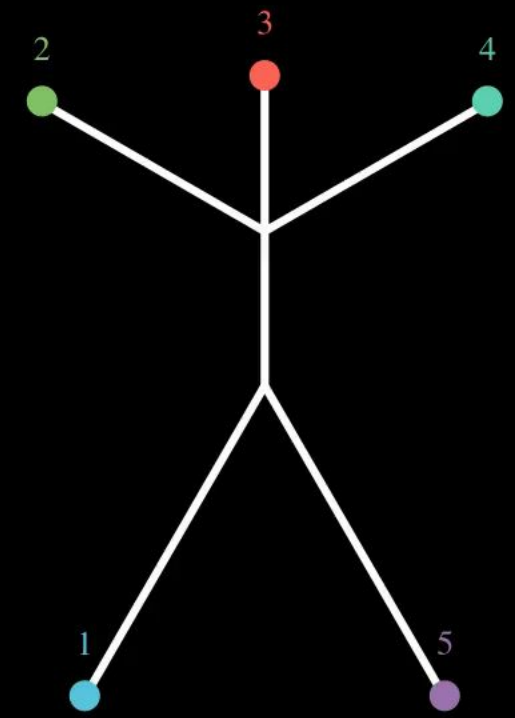


$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$


2
3
4
1
5



$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
$$\sigma L_{1,3} \leq \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$
$$\sigma L_{1,4} \leq \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2}$$
$$\sigma L_{1,5} \leq \sqrt{(x_5 - x_1)^2 + (y_5 - y_1)^2}$$
$$\sigma L_{2,3} \leq \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2}$$
$$\sigma L_{2,4} \leq \sqrt{(x_4 - x_2)^2 + (y_4 - y_2)^2}$$
$$\sigma L_{2,5} \leq \sqrt{(x_5 - x_2)^2 + (y_5 - y_2)^2}$$
$$\sigma L_{3,4} \leq \sqrt{(x_4 - x_3)^2 + (y_4 - y_3)^2}$$
$$\sigma L_{3,5} \leq \sqrt{(x_5 - x_3)^2 + (y_5 - y_3)^2}$$
$$\sigma L_{4,5} \leq \sqrt{(x_5 - x_4)^2 + (y_5 - y_4)^2}$$

$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
$$\sigma L_{1,3} \leq \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$
$$\sigma L_{1,4} \leq \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2}$$
$$\sigma L_{1,5} \leq \sqrt{(x_5 - x_1)^2 + (y_5 - y_1)^2}$$
$$\sigma L_{2,3} \leq \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2}$$
$$\sigma L_{2,4} \leq \sqrt{(x_4 - x_2)^2 + (y_4 - y_2)^2}$$
$$\sigma L_{2,5} \leq \sqrt{(x_5 - x_2)^2 + (y_5 - y_2)^2}$$
$$\sigma L_{3,4} \leq \sqrt{(x_4 - x_3)^2 + (y_4 - y_3)^2}$$
$$\sigma L_{3,5} \leq \sqrt{(x_5 - x_3)^2 + (y_5 - y_3)^2}$$
$$\sigma L_{4,5} \leq \sqrt{(x_5 - x_4)^2 + (y_5 - y_4)^2}$$

Choose $x_1, y_1, x_2, y_2, \ldots, \sigma$ to maximize σ

subject to the following inequalities:

$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$\sigma L_{1,3} \leq \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$

$$\sigma L_{1,4} \leq \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2}$$

$$\vdots$$

Choose $\vec{x}$ to maximize σ

subject to the following inequalities:

$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
$$\sigma L_{1,3} \leq \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$
$$\sigma L_{1,4} \leq \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2}$$
$$\vdots$$

where $\vec{x} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ \sigma \end{bmatrix}$

Choose $\vec{x}$ to minimize $-\sigma$

subject to the following inequalities:

$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
$$\sigma L_{1,3} \leq \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$
$$\sigma L_{1,4} \leq \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2}$$
$$\vdots$$

where $\vec{x} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ \sigma \end{bmatrix}$

Choose $\vec{x}$ to minimize $f(\vec{x})$

subject to the following inequalities:

$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$\sigma L_{1,3} \leq \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$

$$\sigma L_{1,4} \leq \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2}$$

$$\vdots$$

where $\vec{x} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ \sigma \end{bmatrix}$

where $f(\vec{x}) = -\sigma$

$$\min_{\vec{x}} \; f(\vec{x})$$

s.t.

$$\begin{aligned}
\sigma L_{1,2} &\leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\
\sigma L_{1,3} &\leq \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2} \\
\sigma L_{1,4} &\leq \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2} \\
&\vdots
\end{aligned}$$

where $\vec{x} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ \sigma \end{bmatrix}$

where $f(\vec{x}) = -\sigma$

$$\min_{\vec{x}} \; f(\vec{x})$$

s.t.

$$\sigma L_{1,2} - \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \leq 0$$

$$\sigma L_{1,3} - \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2} \leq 0$$

$$\sigma L_{1,4} - \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2} \leq 0$$

$$\vdots$$

where $\vec{x} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ \sigma \end{bmatrix}$

where $f(\vec{x}) = -\sigma$

$$\min_{\vec{x}} \; f(\vec{x})$$

s.t.

$$g_1(\vec{x}) \leq 0$$
$$g_2(\vec{x}) \leq 0$$
$$g_3(\vec{x}) \leq 0$$
$$\vdots$$

where $\vec{x} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ \sigma \end{bmatrix}$

where $f(\vec{x}) = -\sigma$

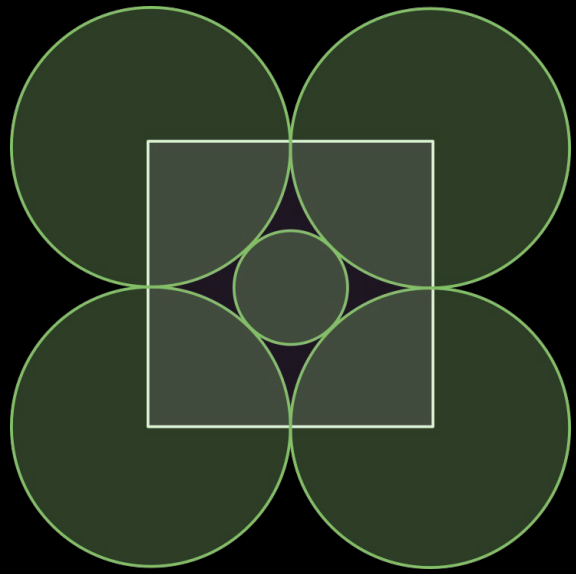
$$\min_{\vec{x}} \; f(\vec{x})$$

s.t.

$$g_1(\vec{x}) \leq 0$$
$$g_2(\vec{x}) \leq 0$$
$$g_3(\vec{x}) \leq 0$$
$$\vdots$$

Solve using open-source solver library

$$\vec{x} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ \sigma \end{bmatrix}$$



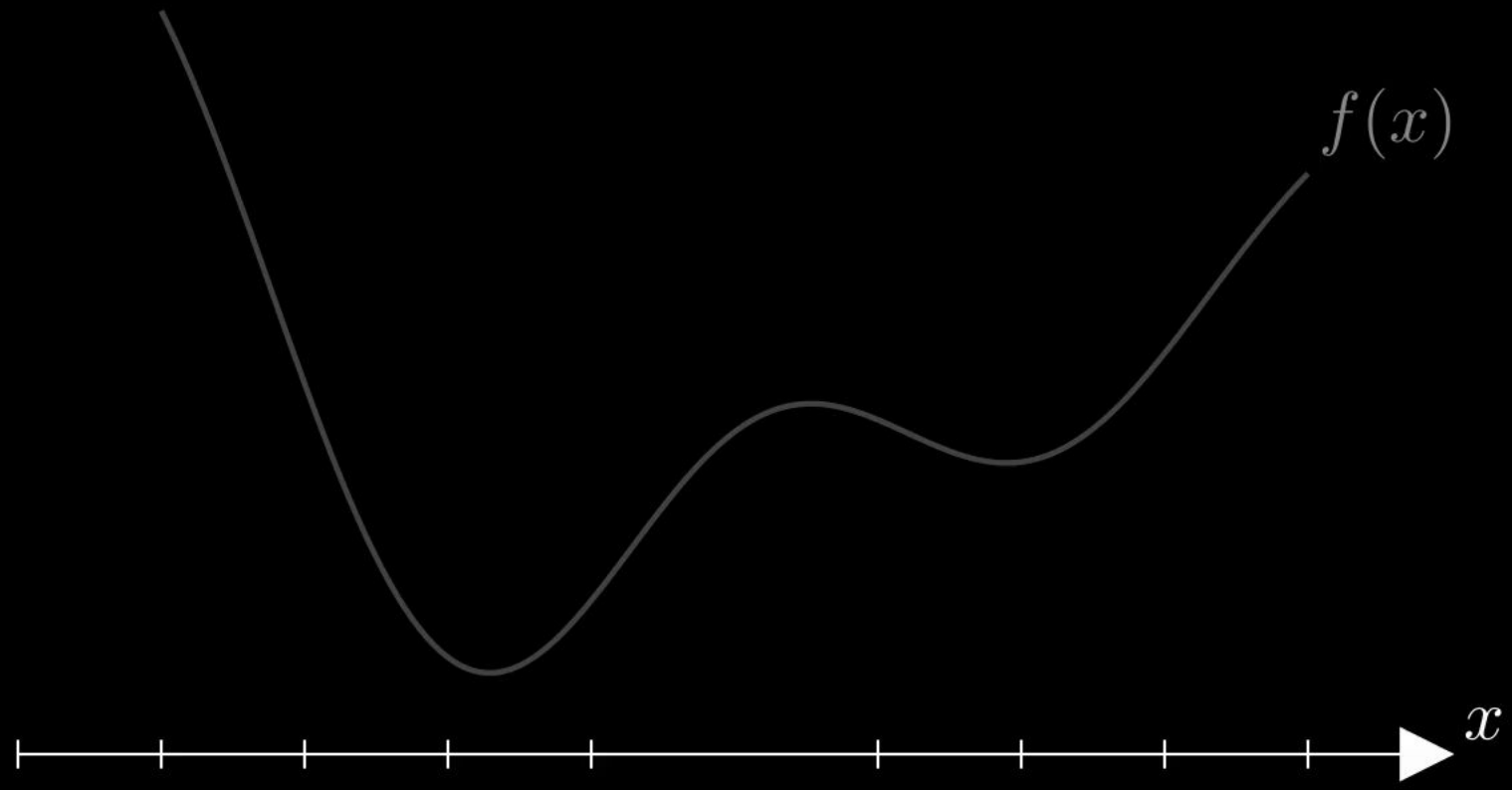
Optimization problem

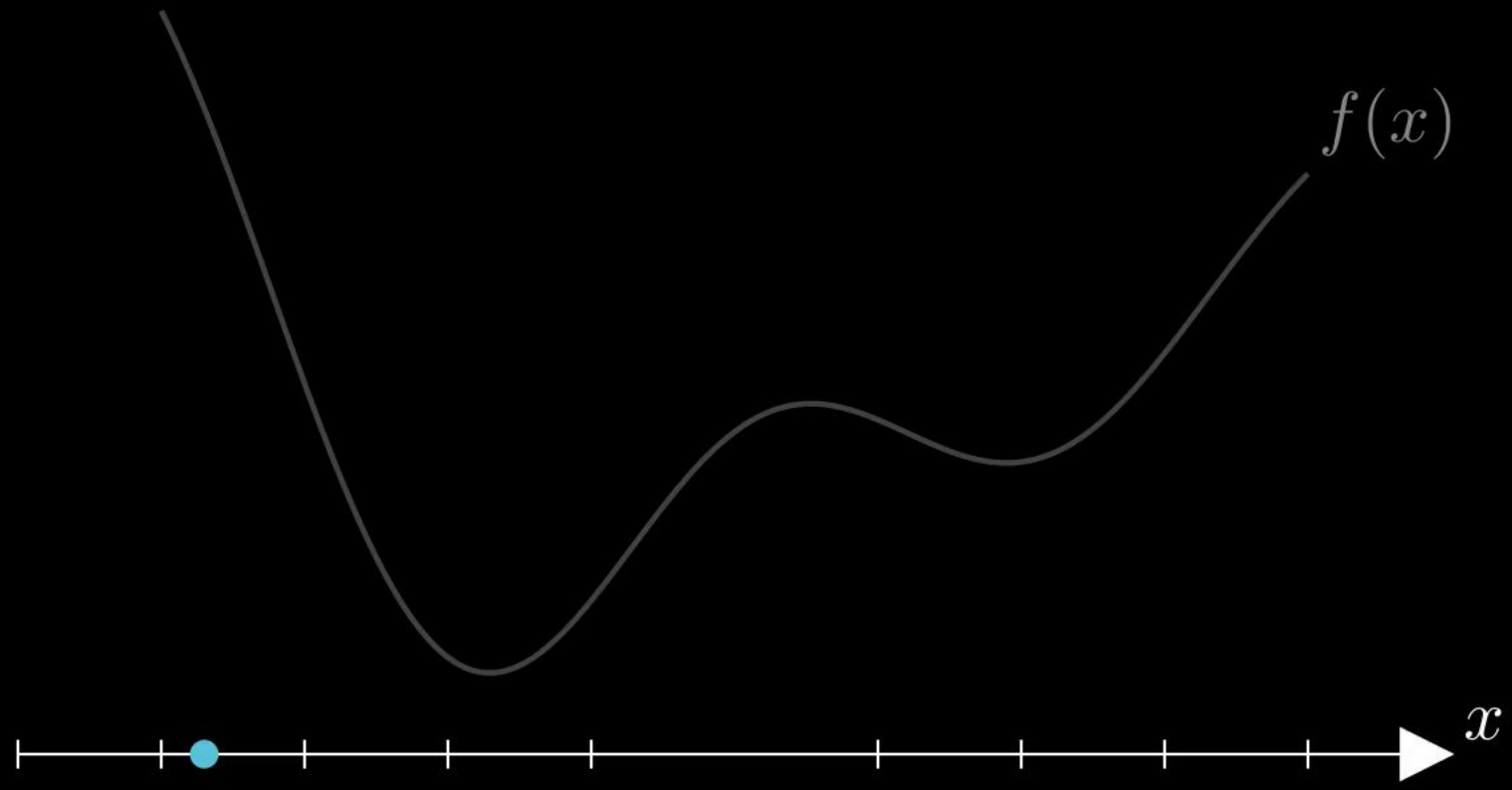
Optimal position vector

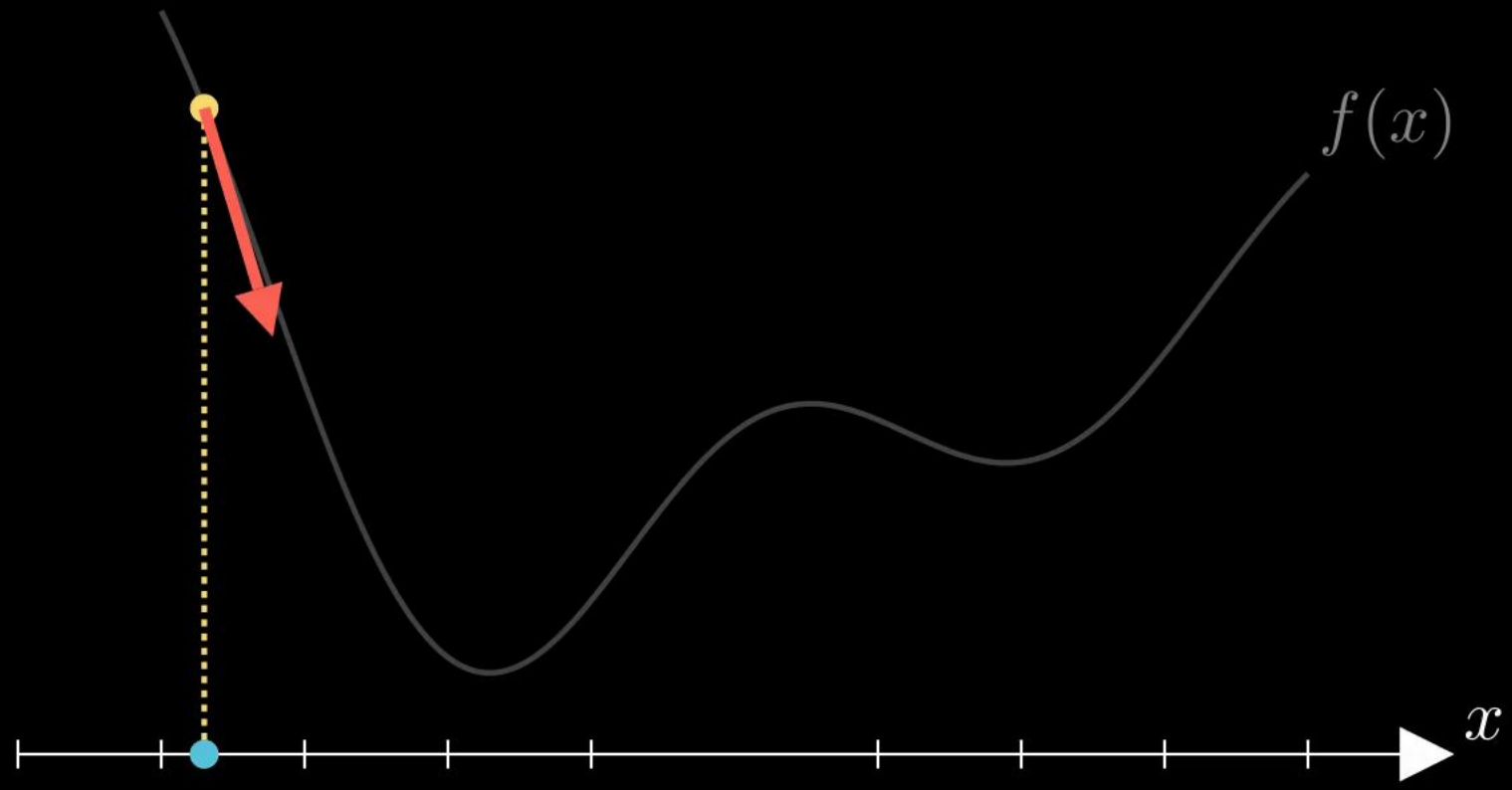
Reconstruct packing

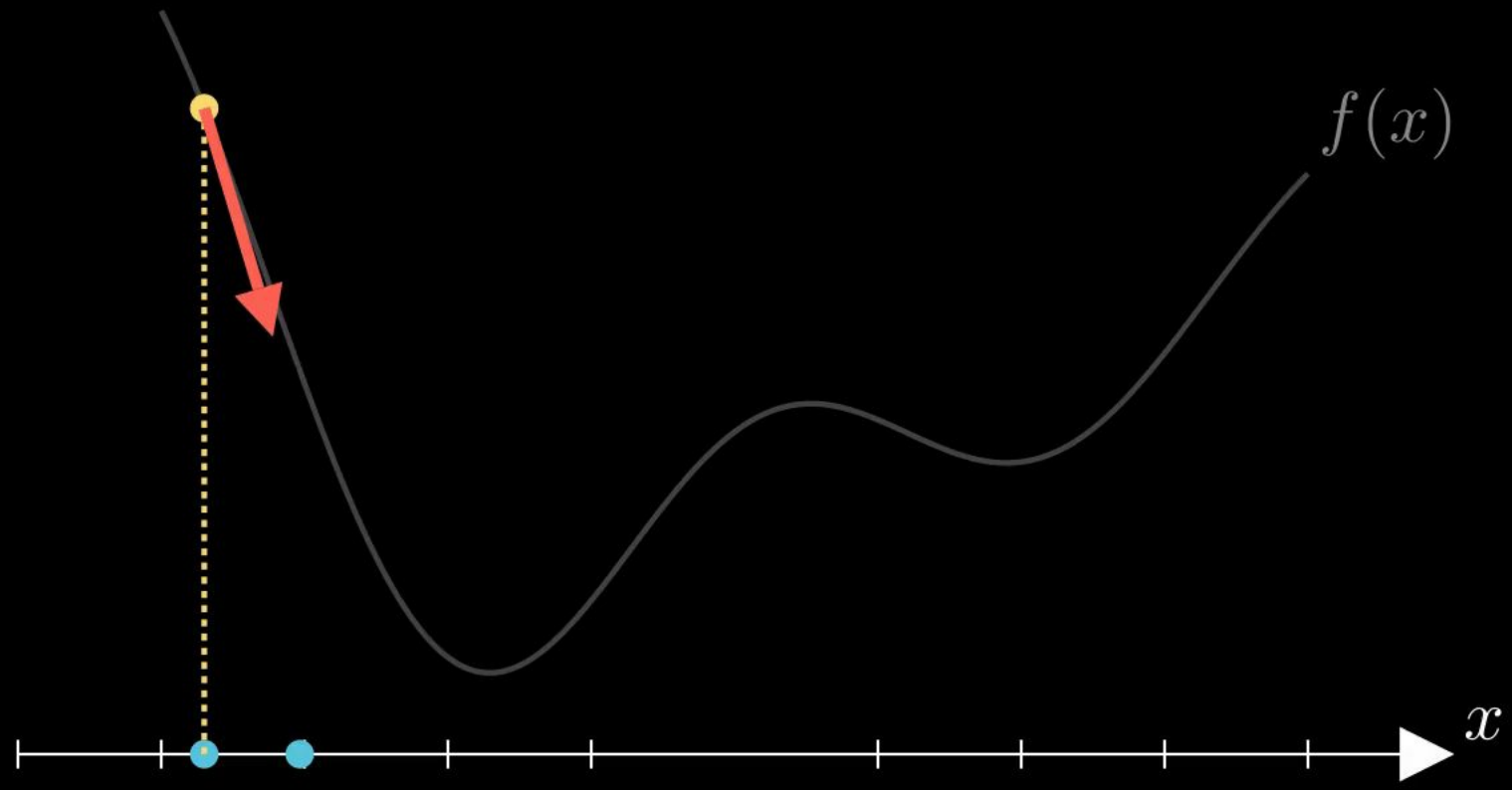
$$\min_{\vec{x}} \; f(\vec{x})$$

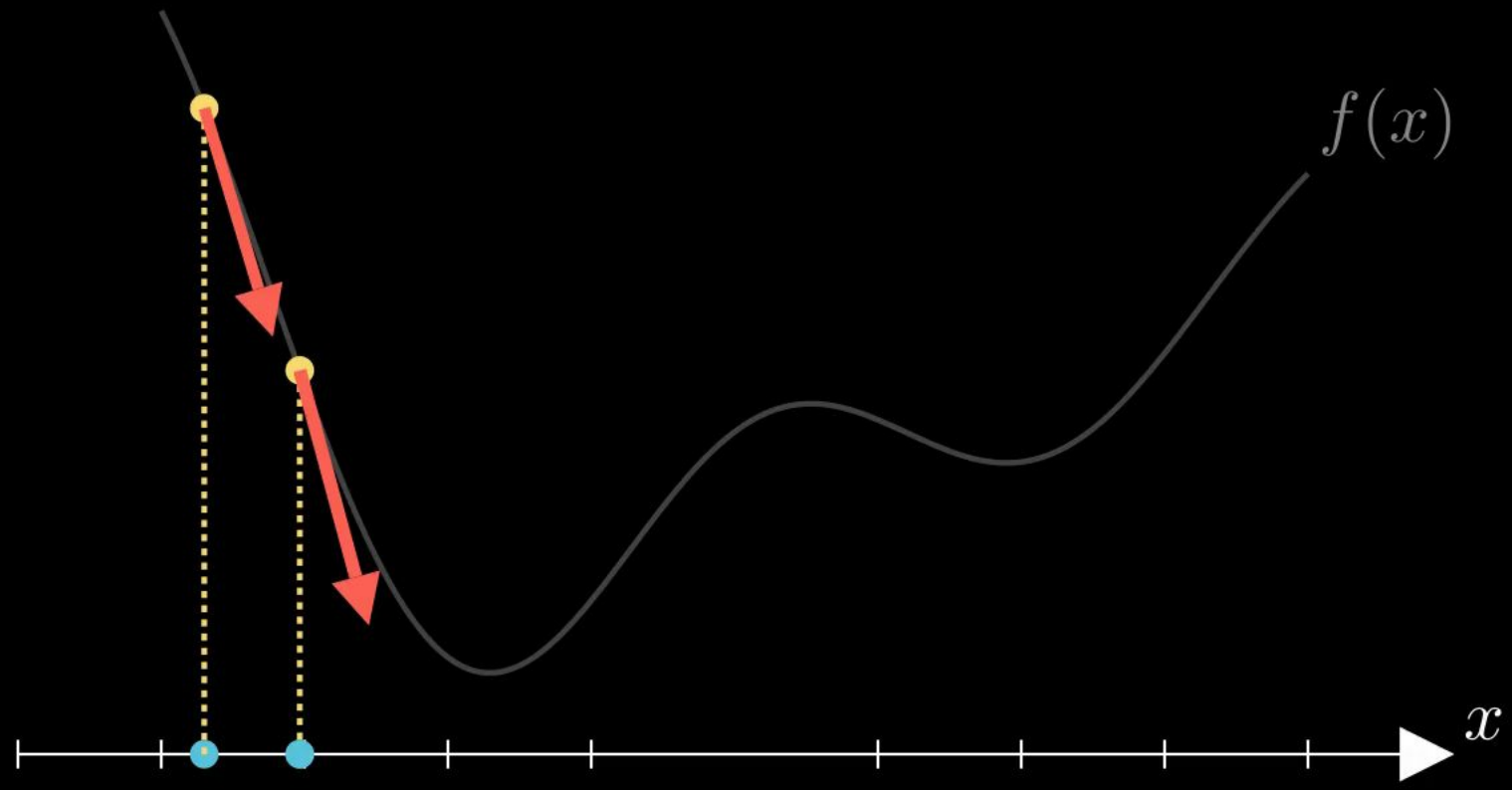
1. Single-variable optimization

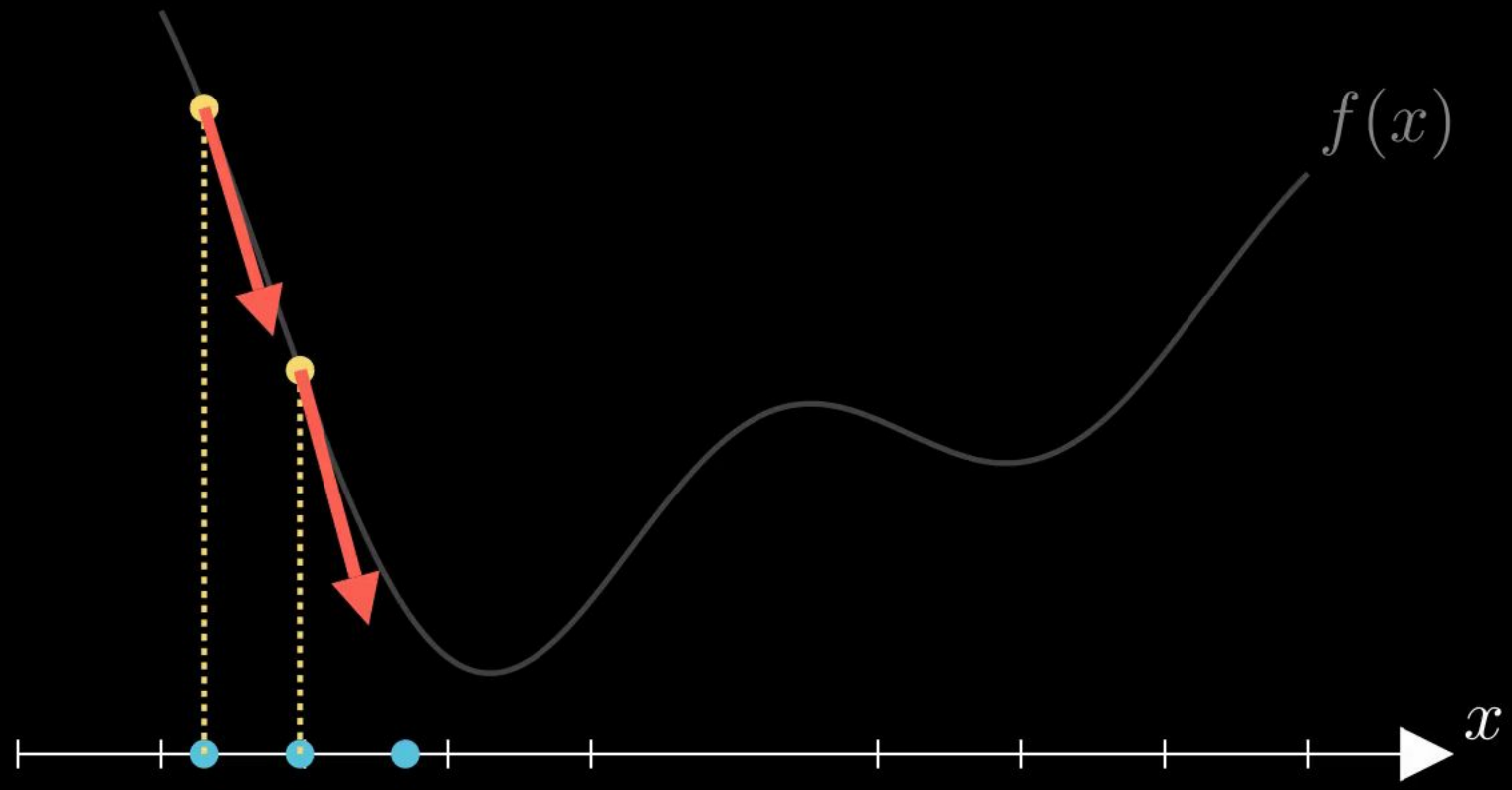

$f(x)$
x

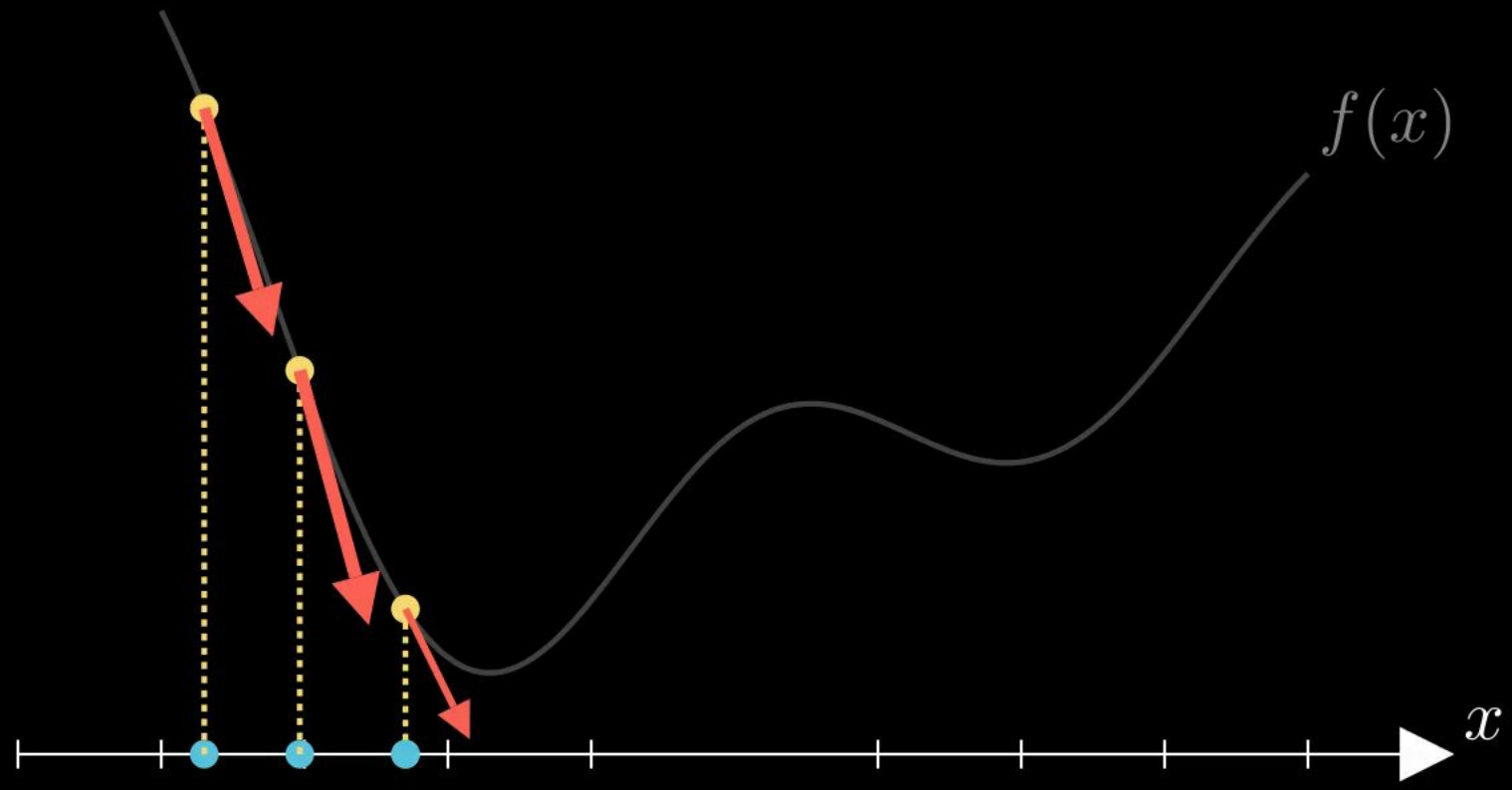

$f(x)$
x

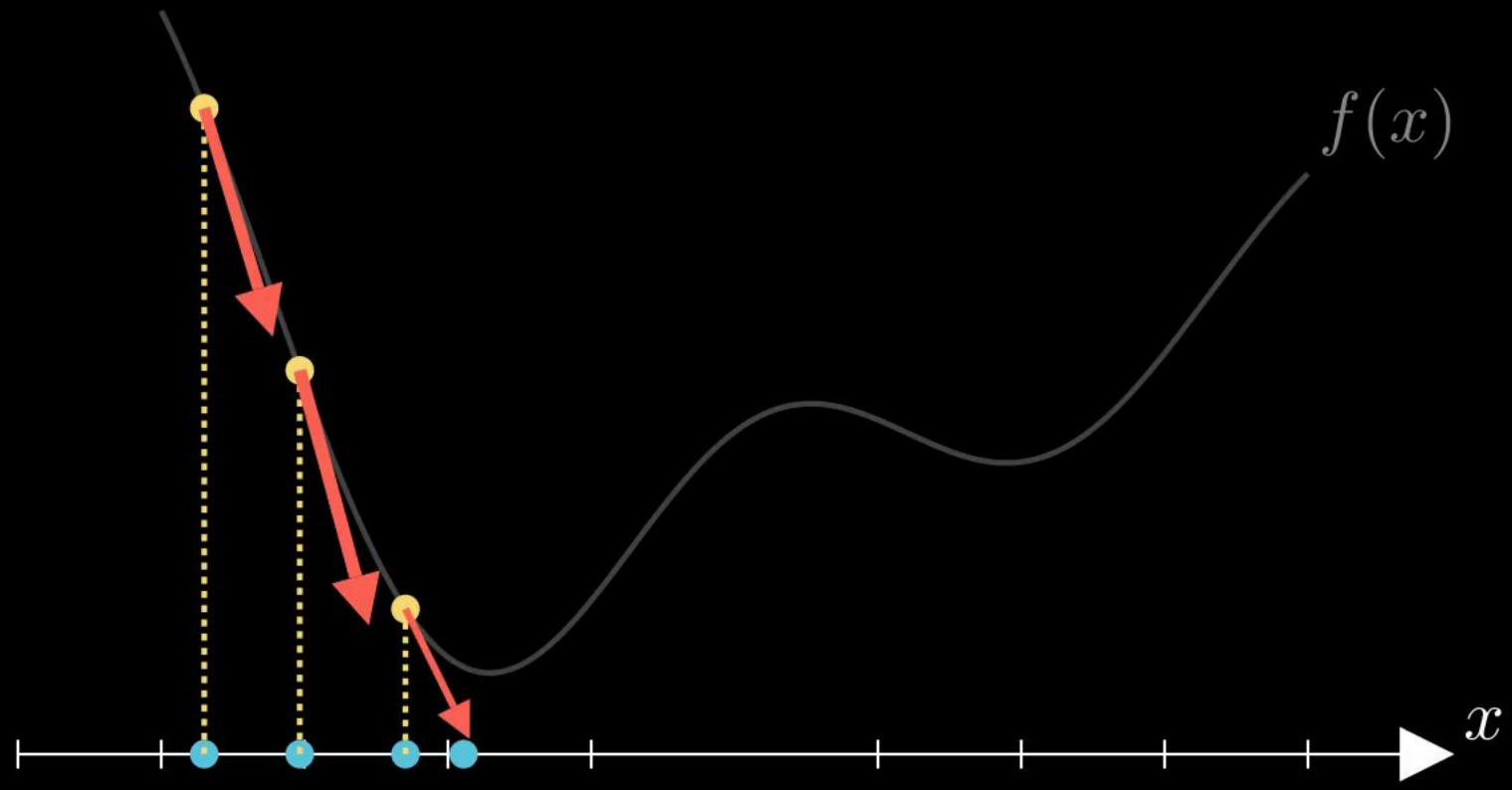

$f(x)$
x

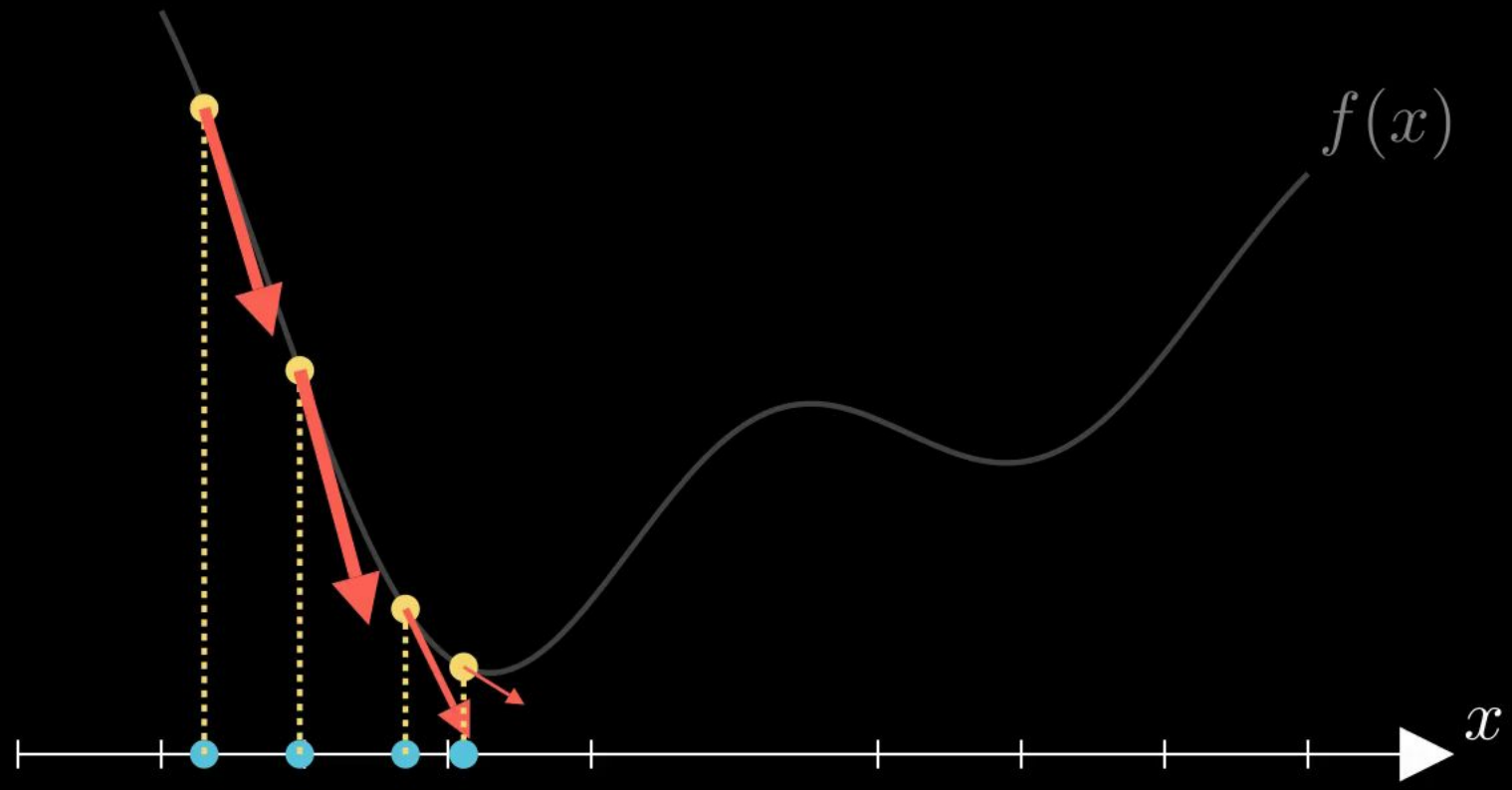

$f(x)$
x

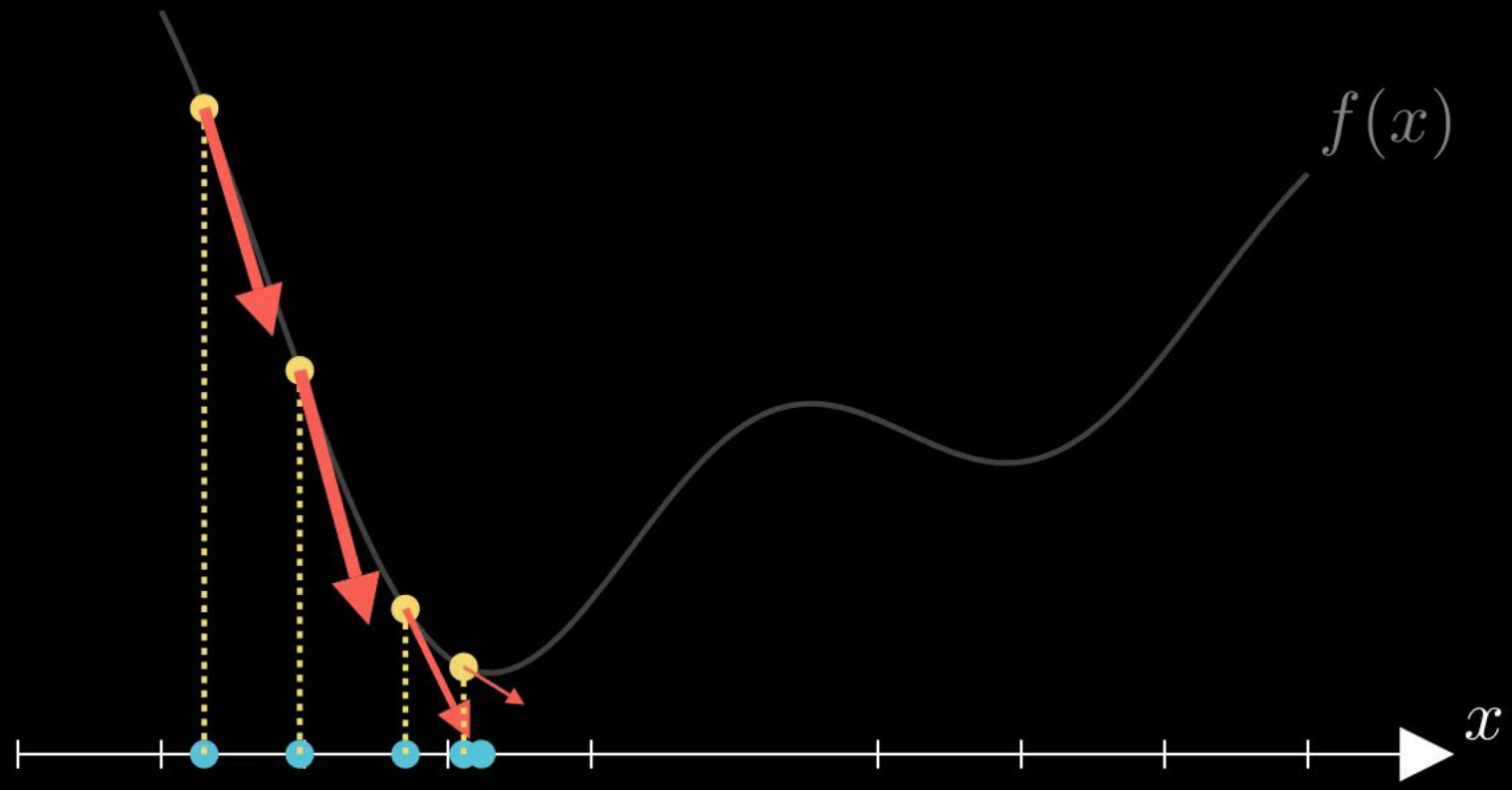

$f(x)$
x

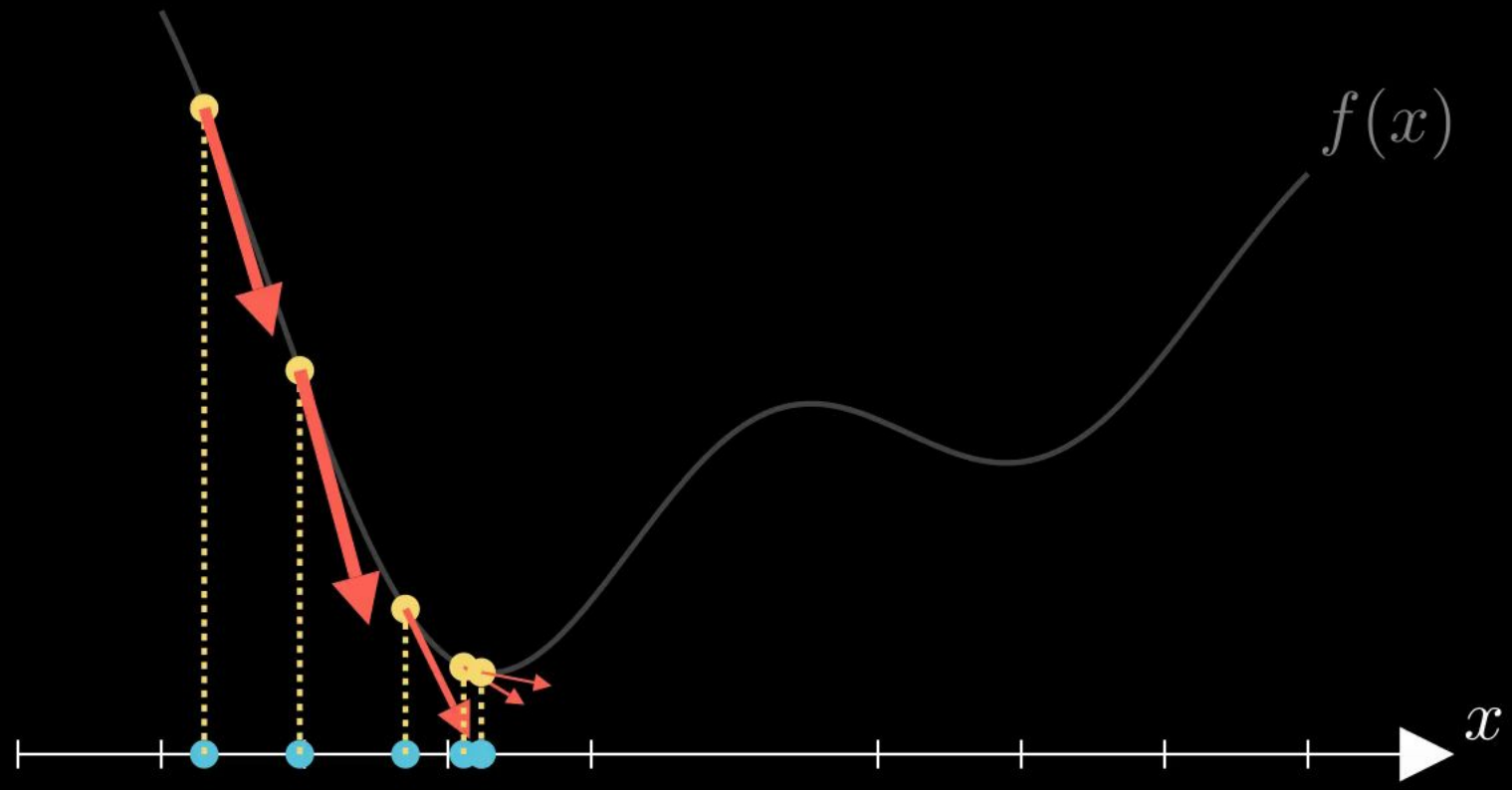

$f(x)$
x

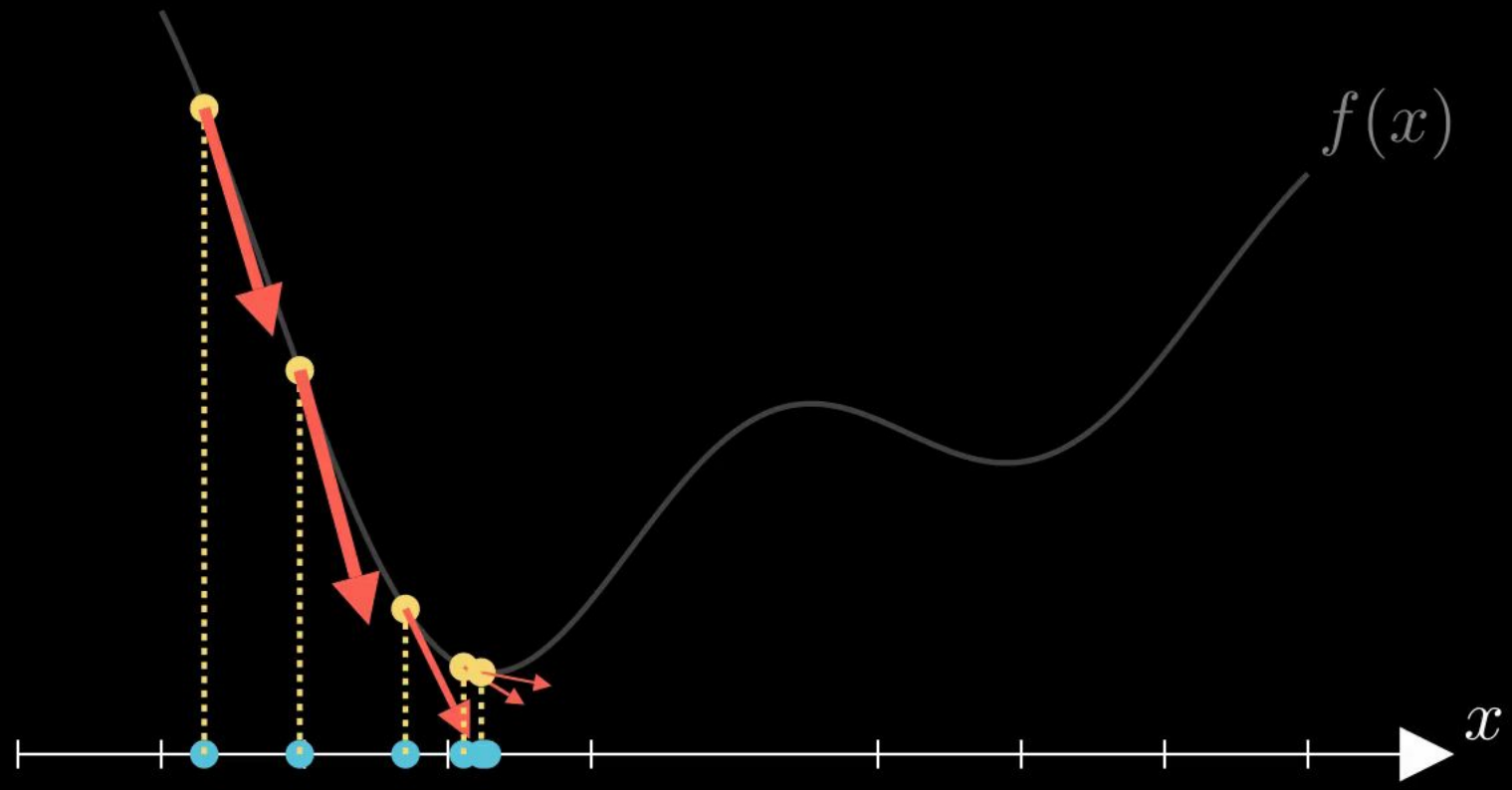

$f(x)$
x

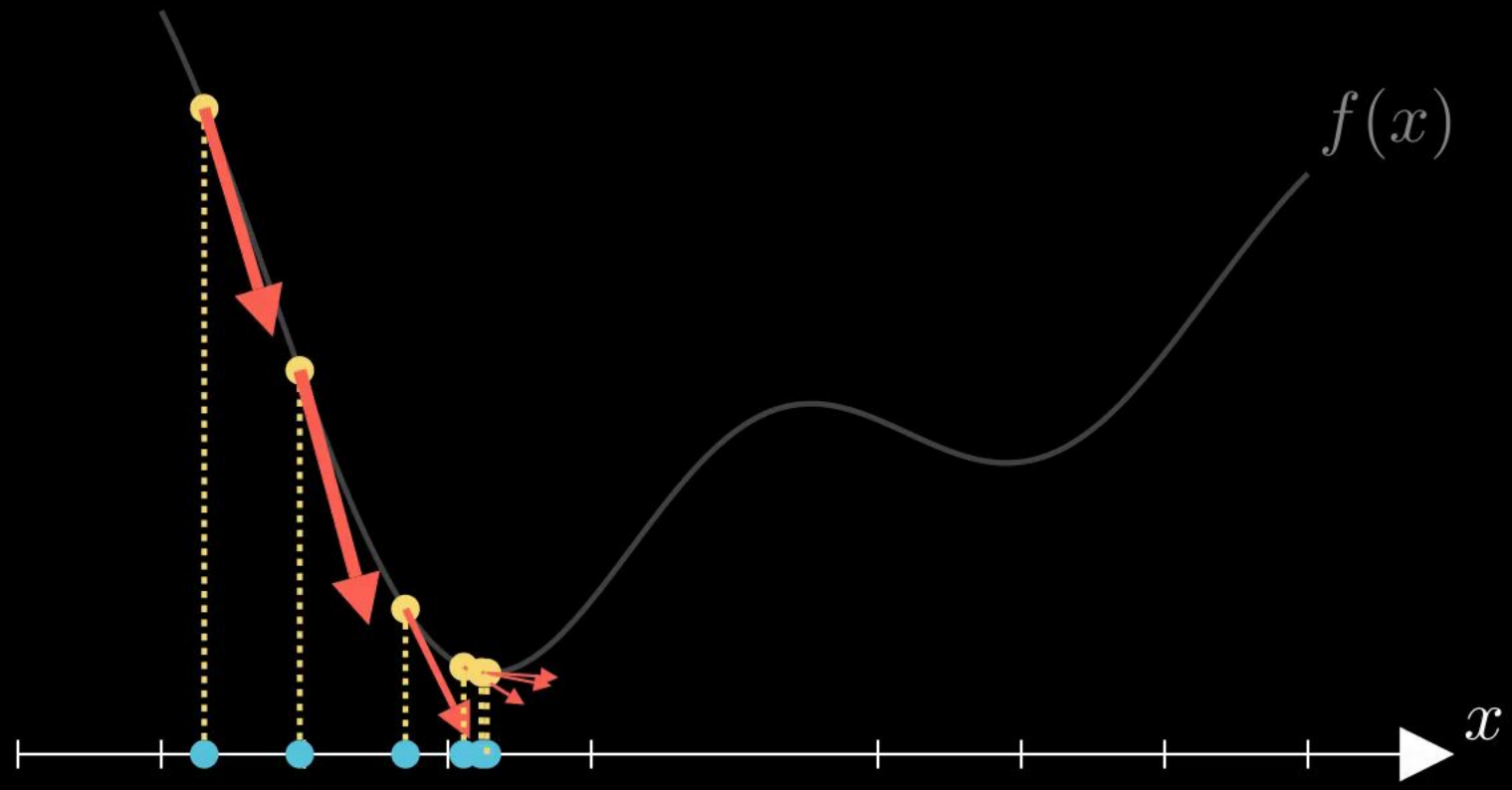

$f(x)$
x

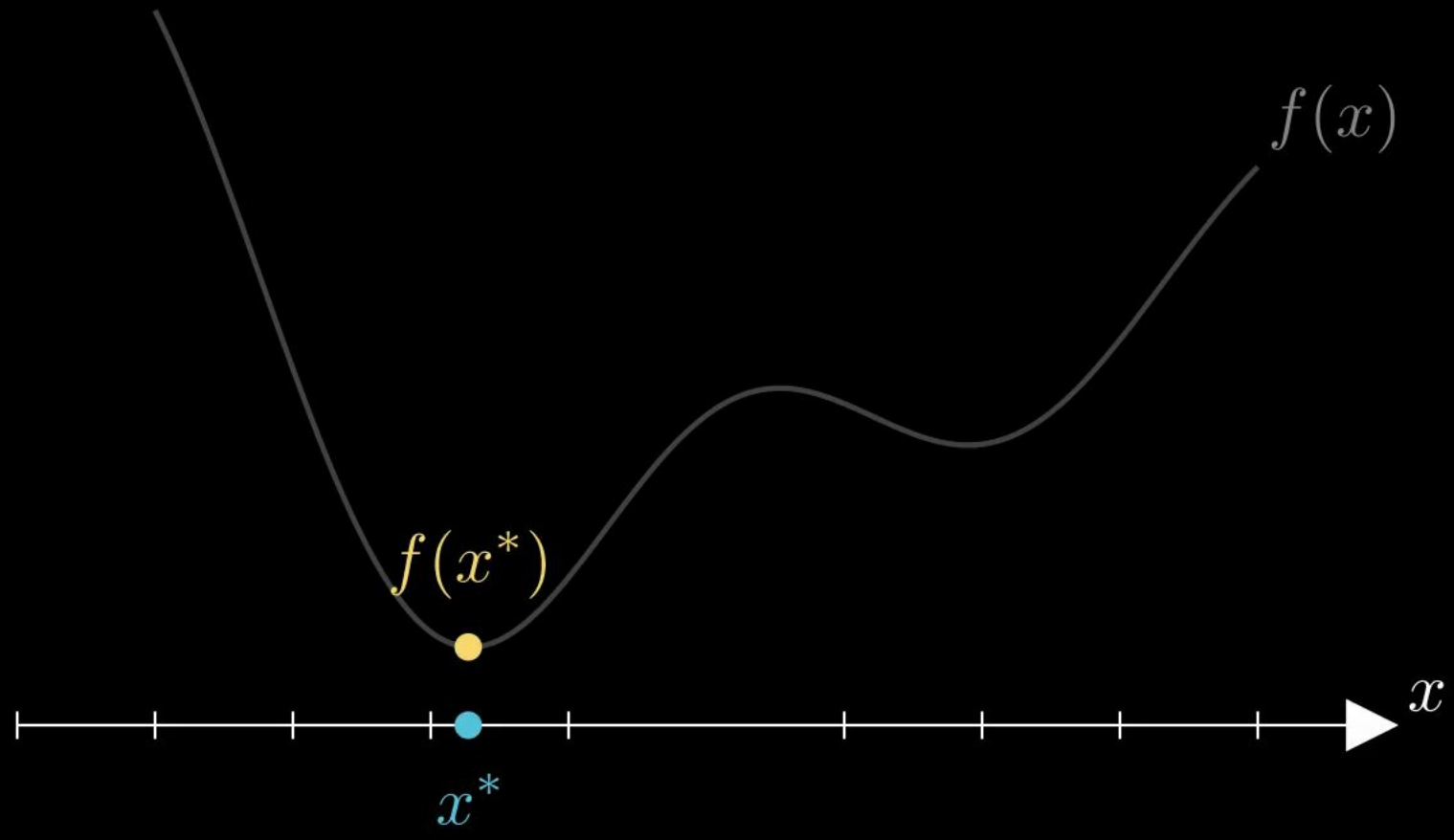

$f(x)$
x

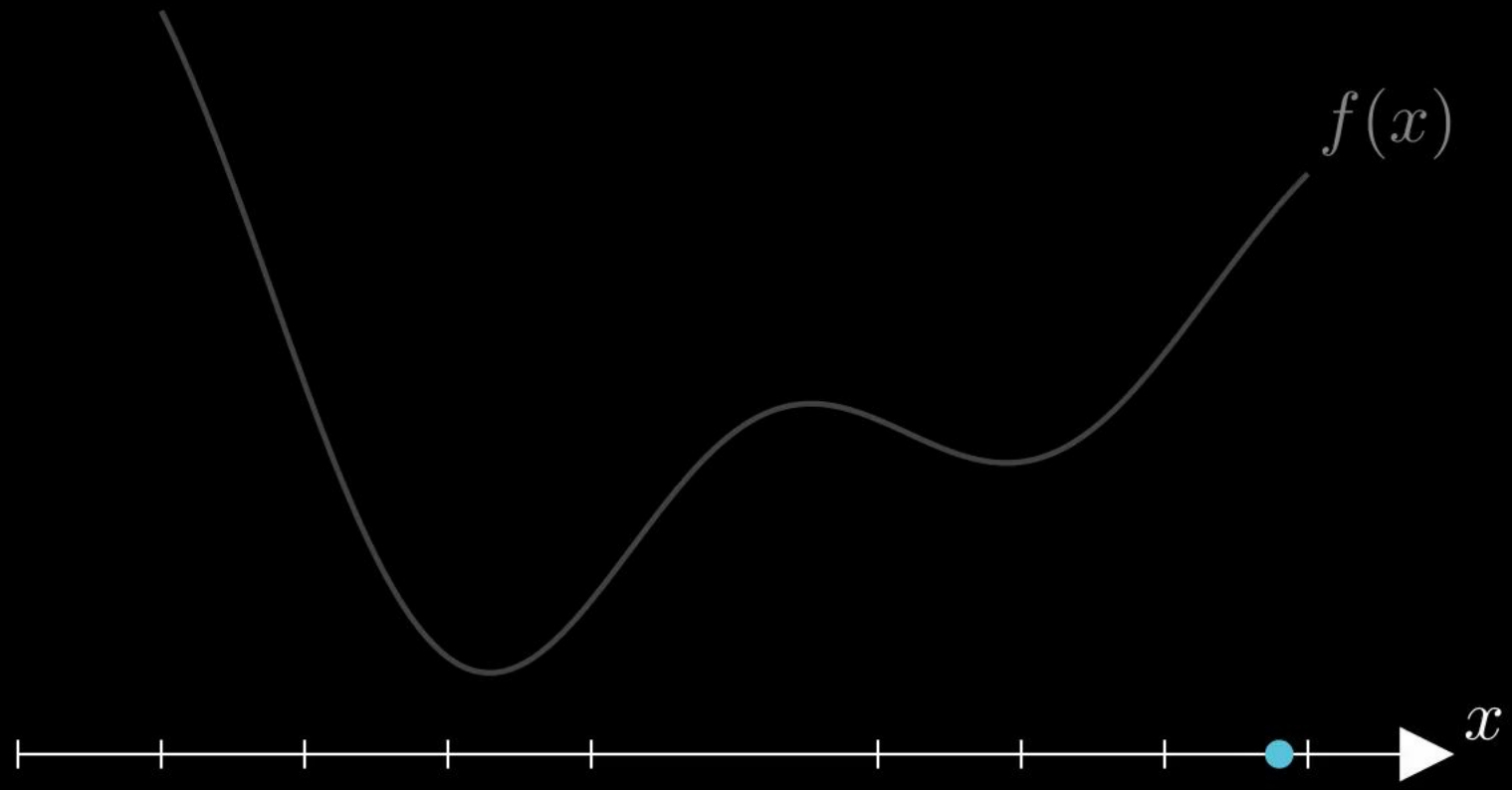

$f(x)$
x

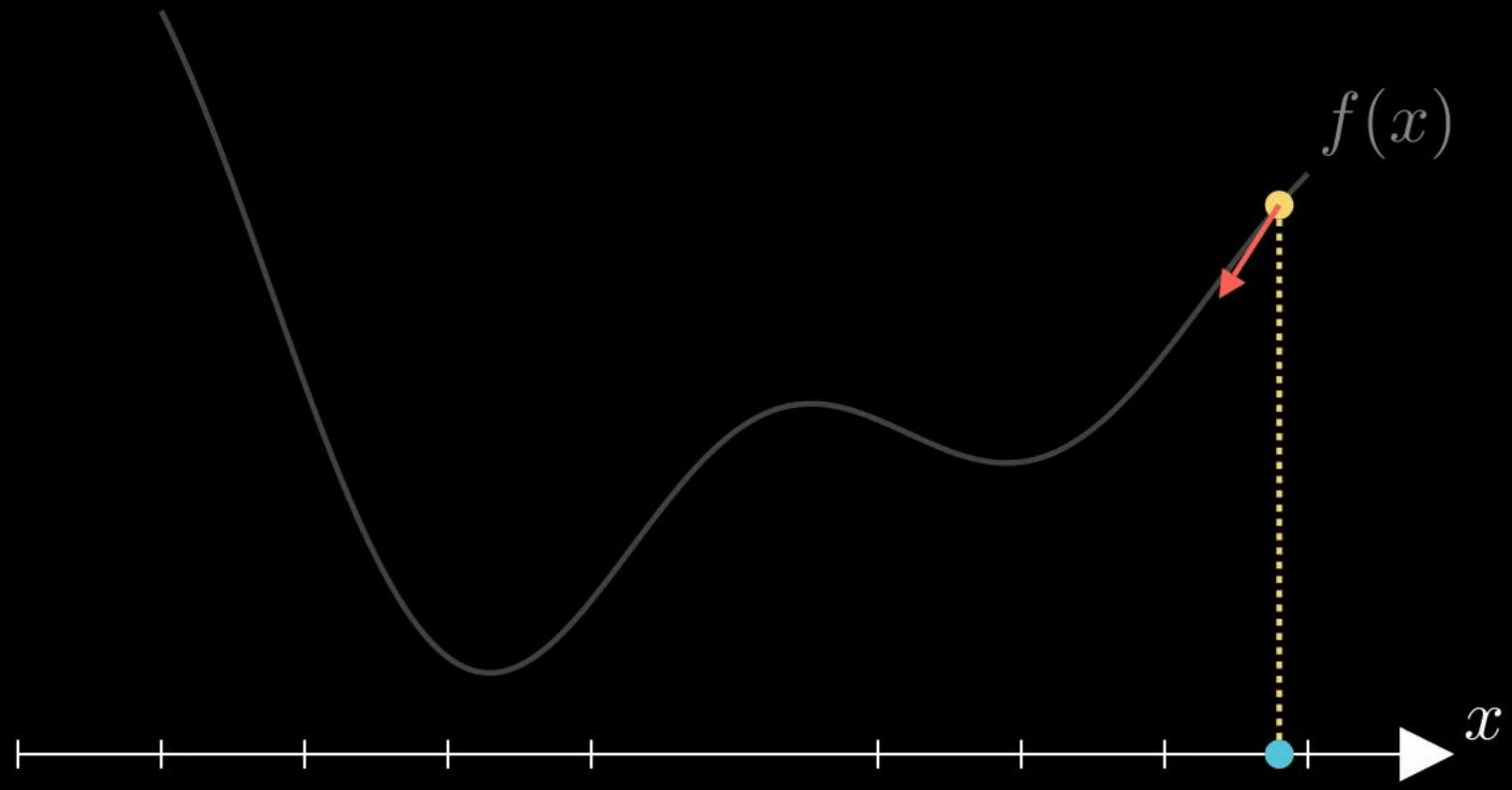

$f(x)$
x

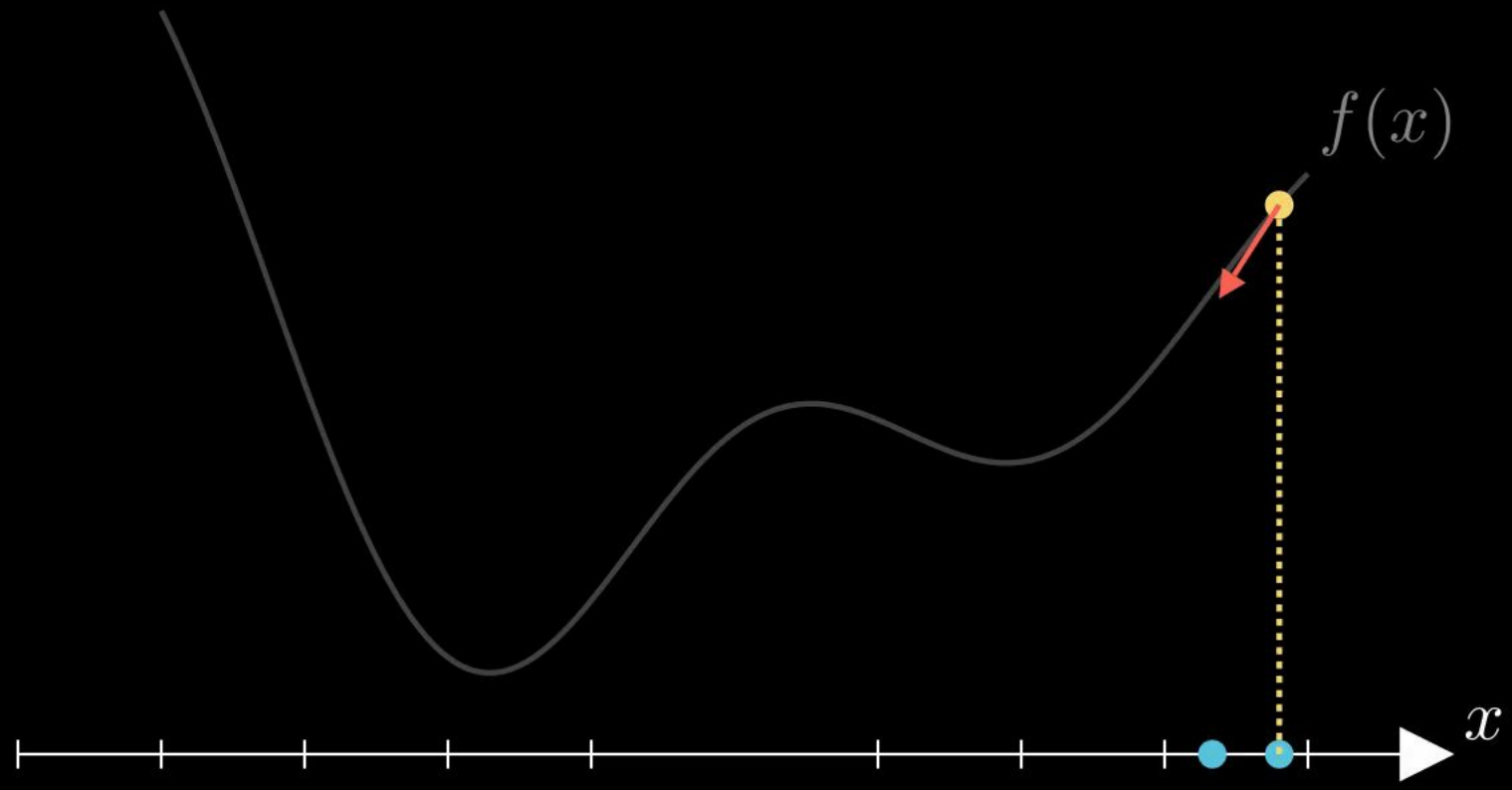

$f(x)$
x

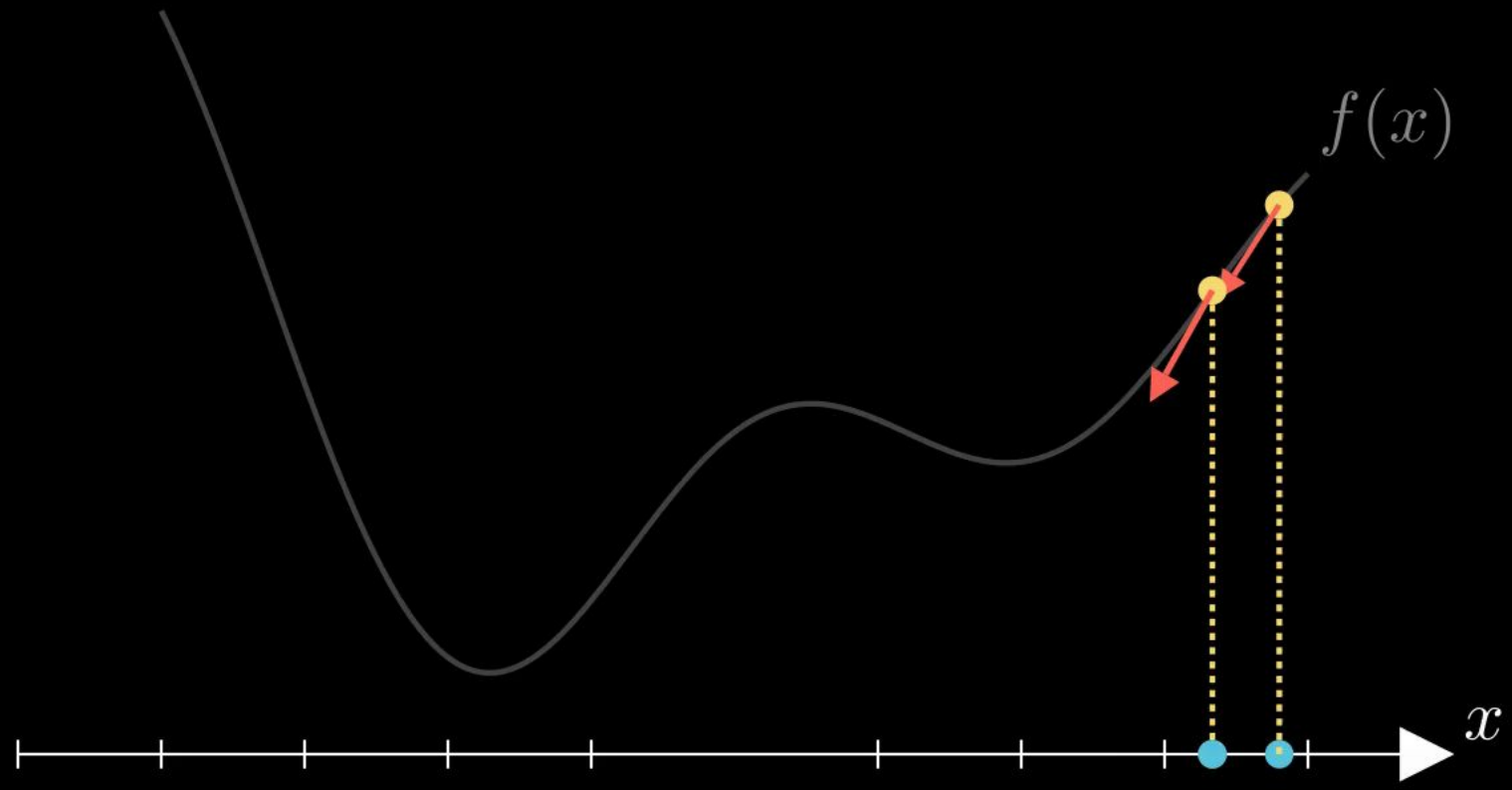

$f(x)$
x

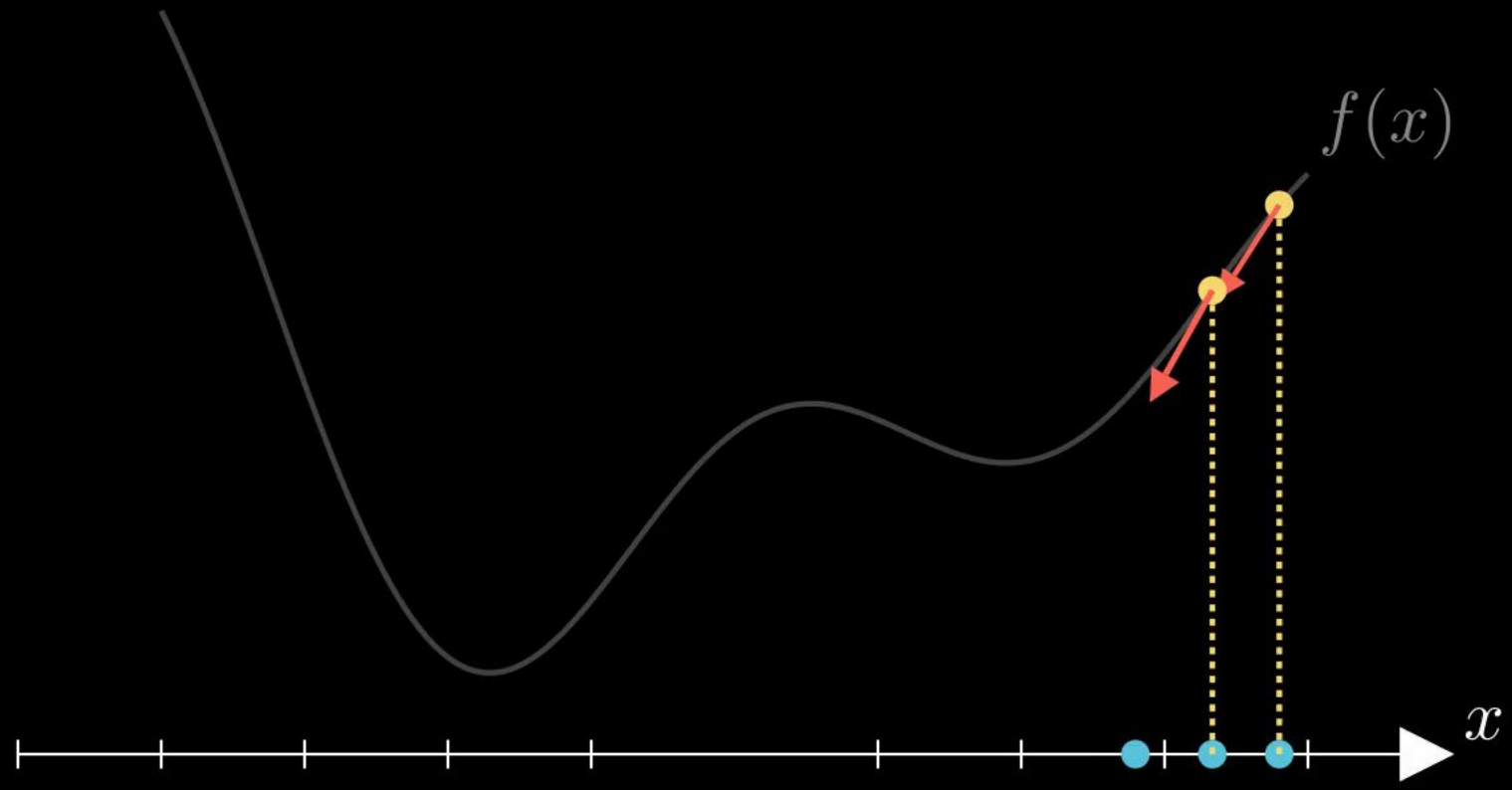

$f(x)$
$f(x^*)$
x
x^*

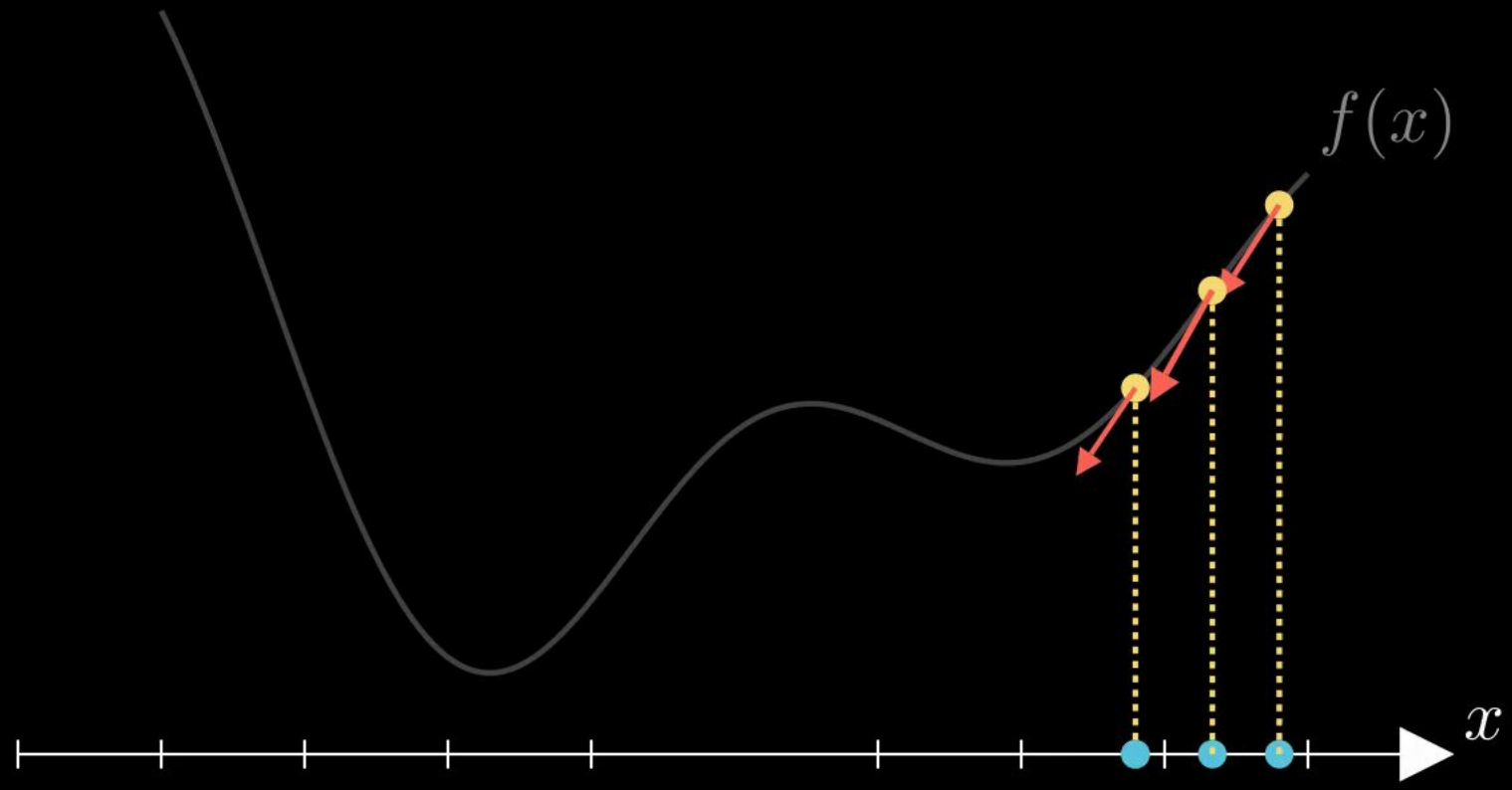

$f(x)$
x

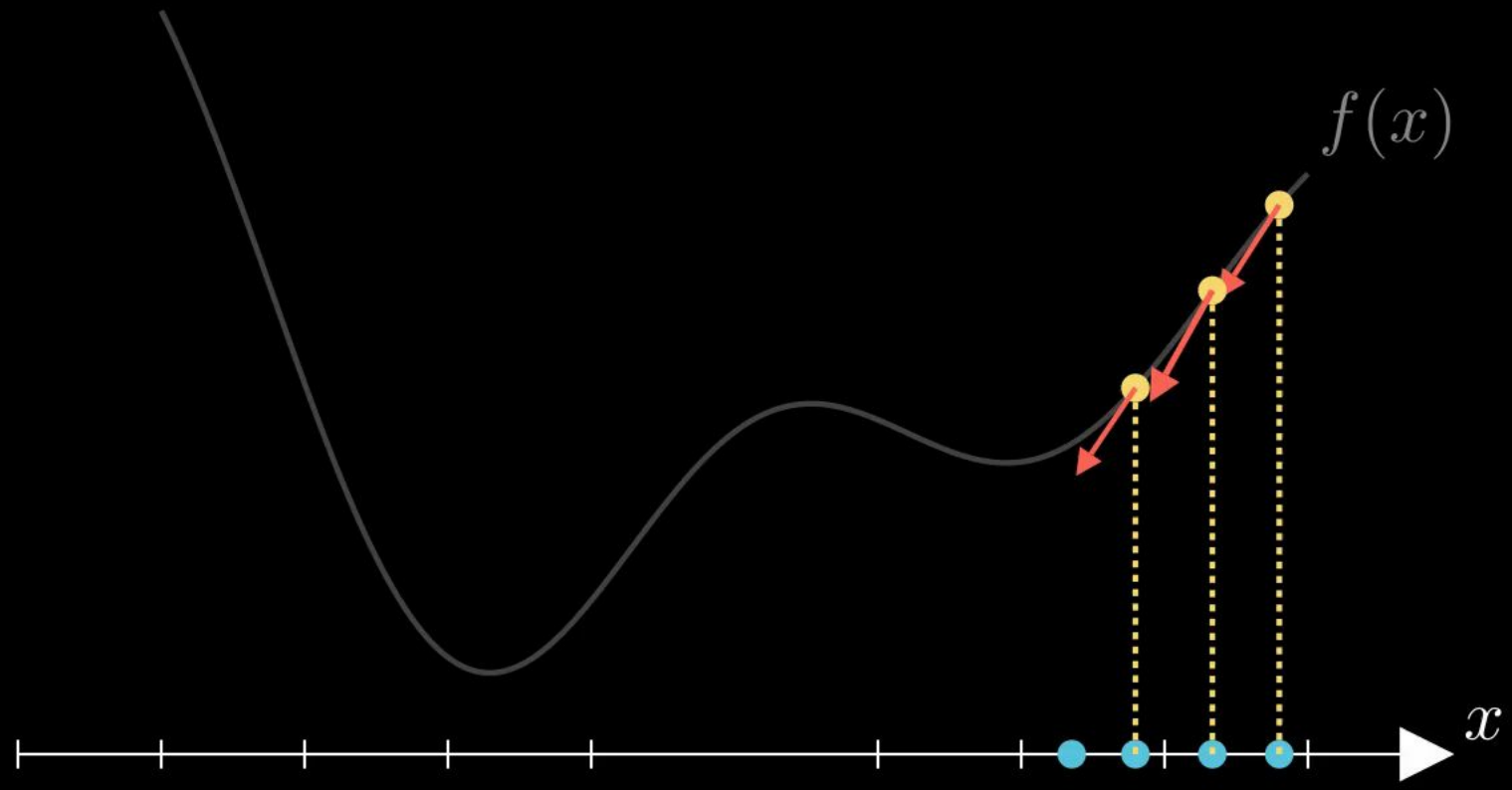

$f(x)$
x

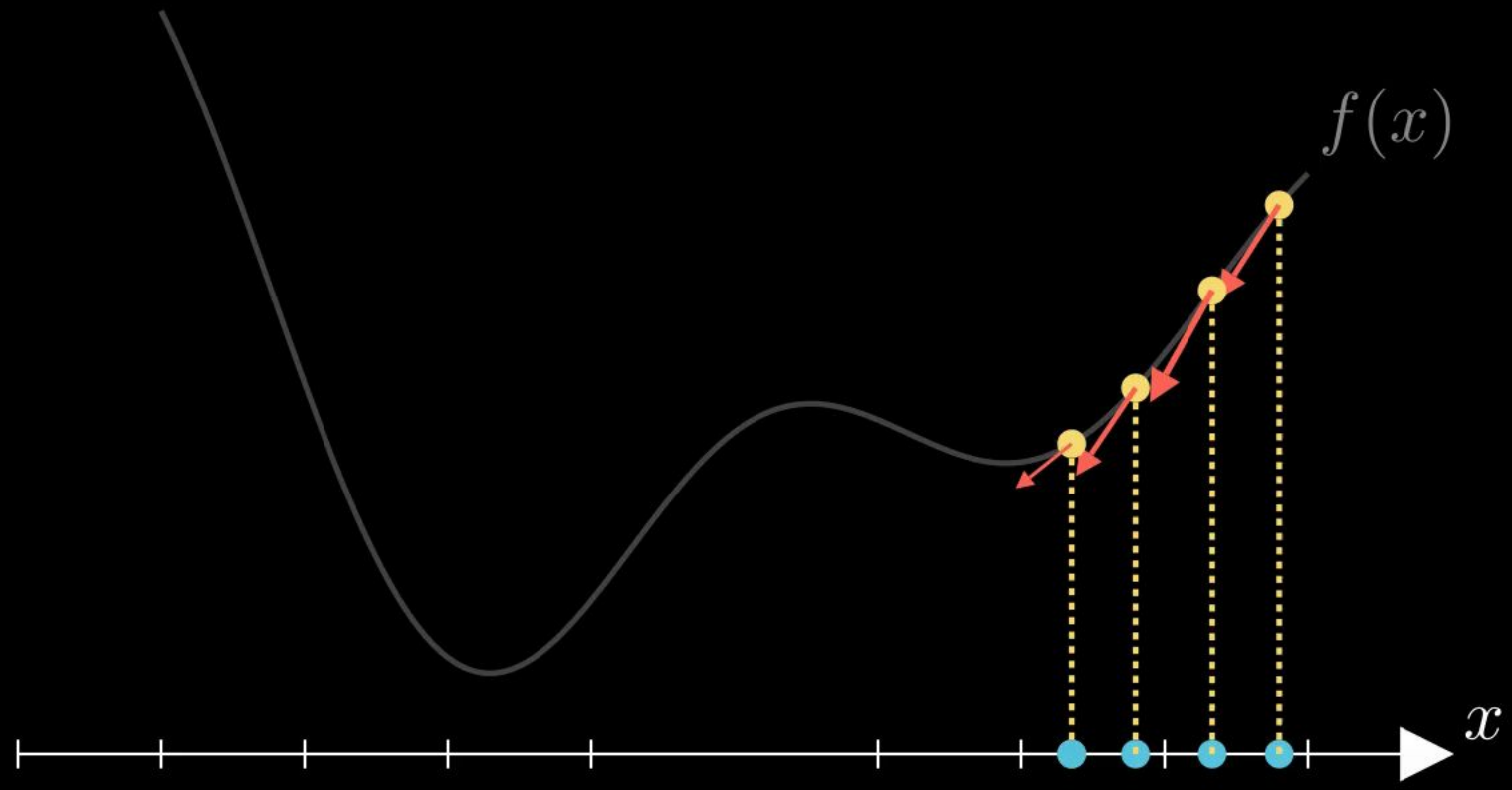

$f(x)$
x

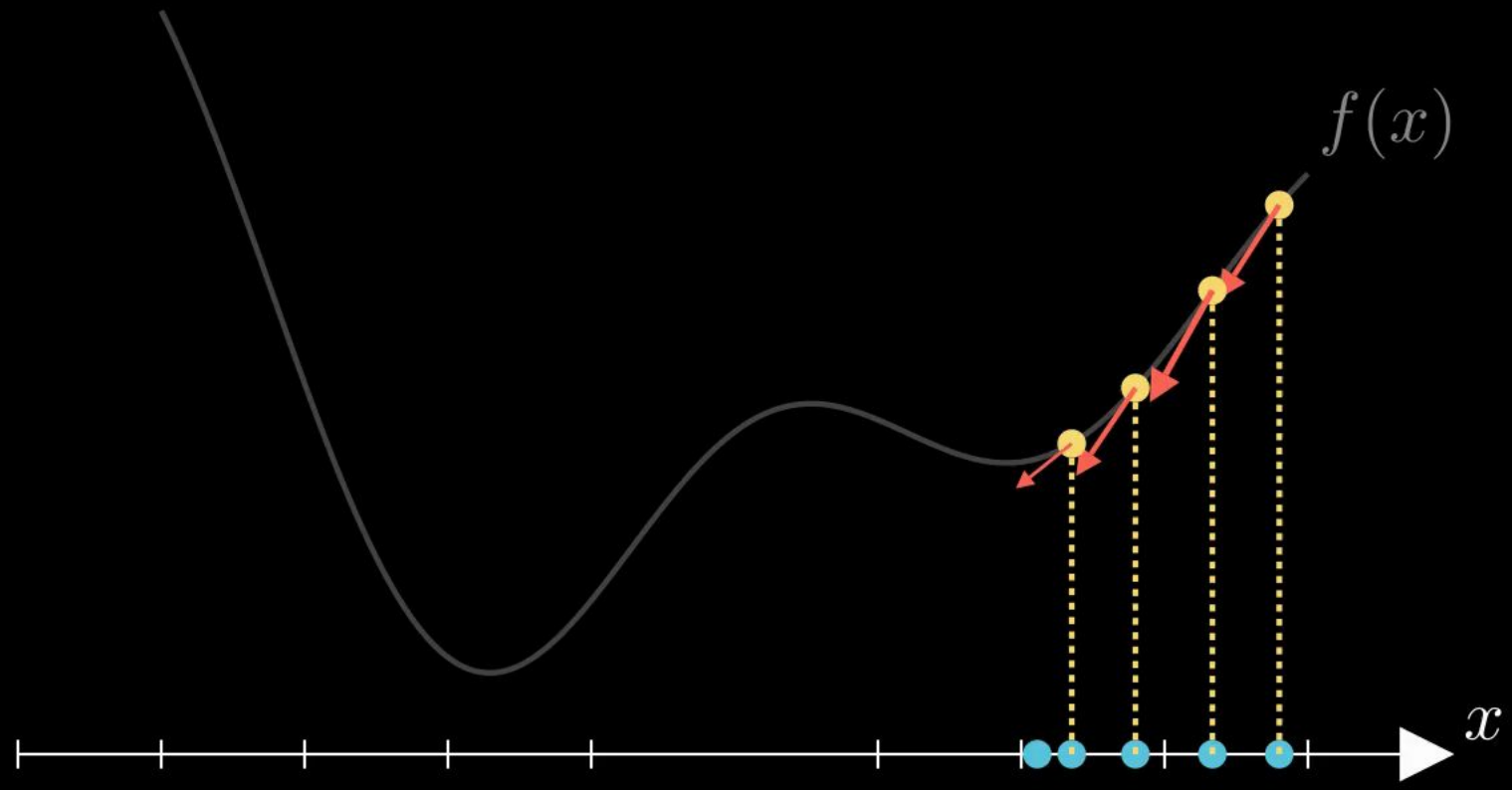

$f(x)$
x

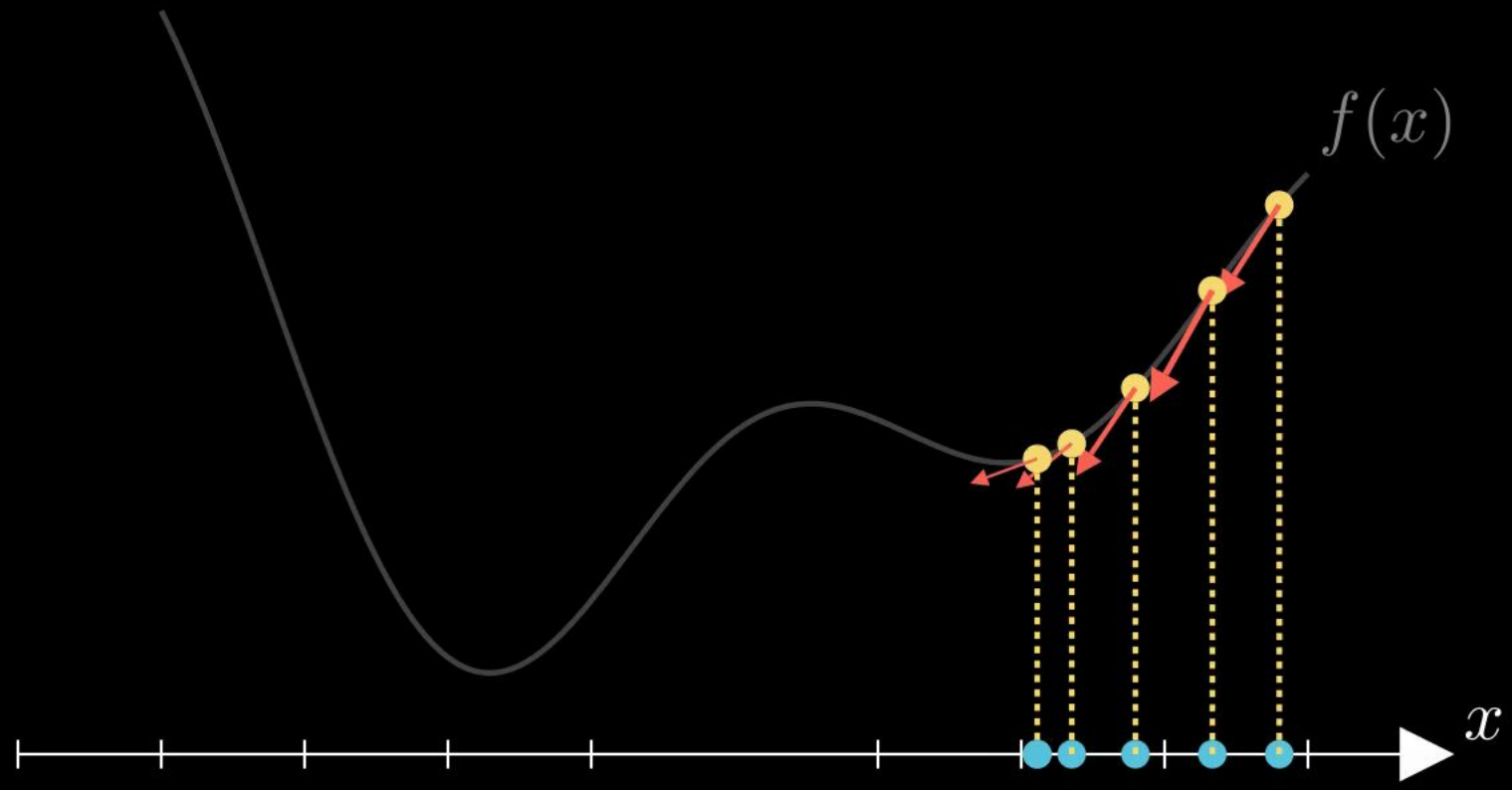

$f(x)$
x

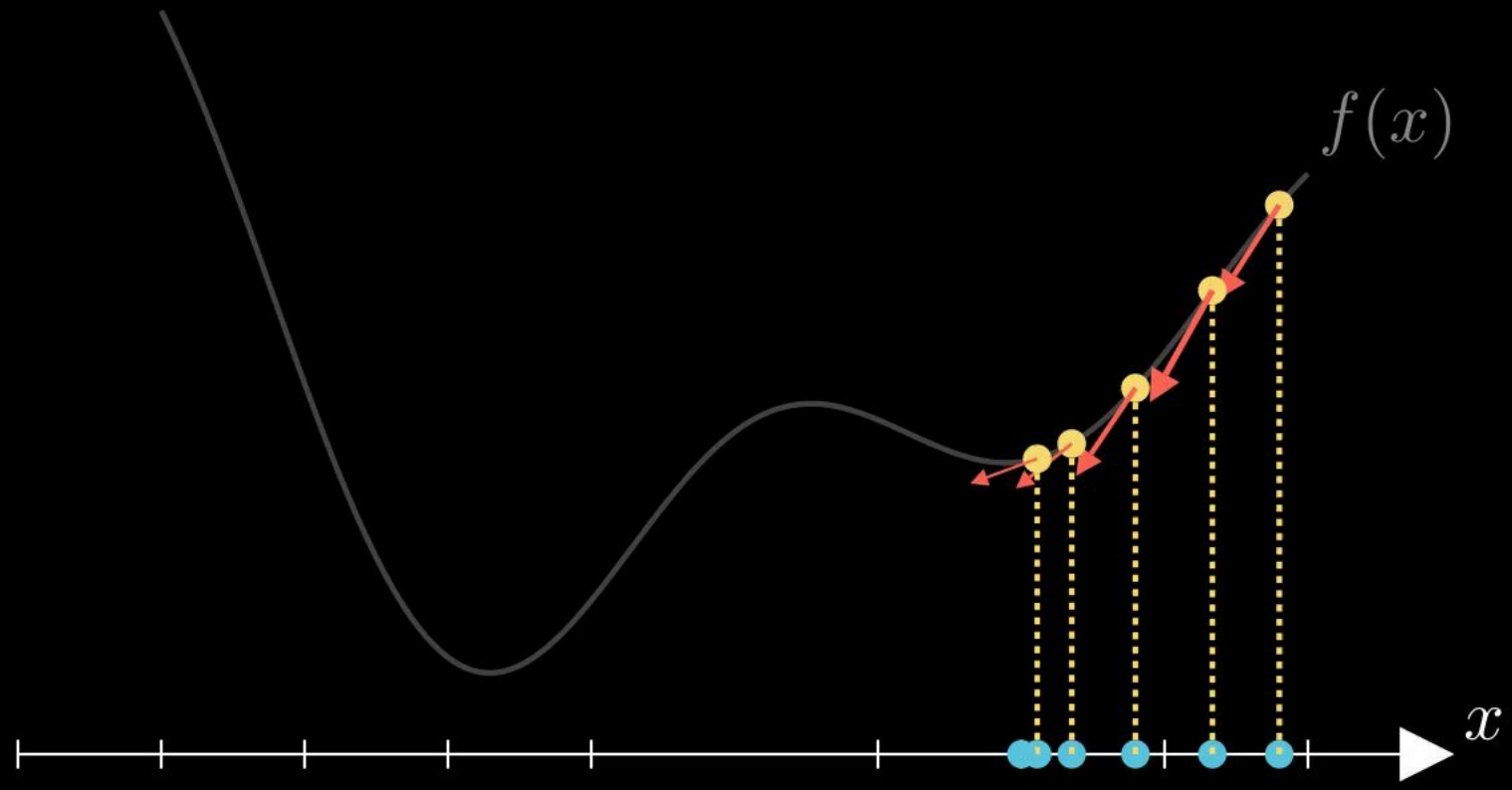

$f(x)$
x

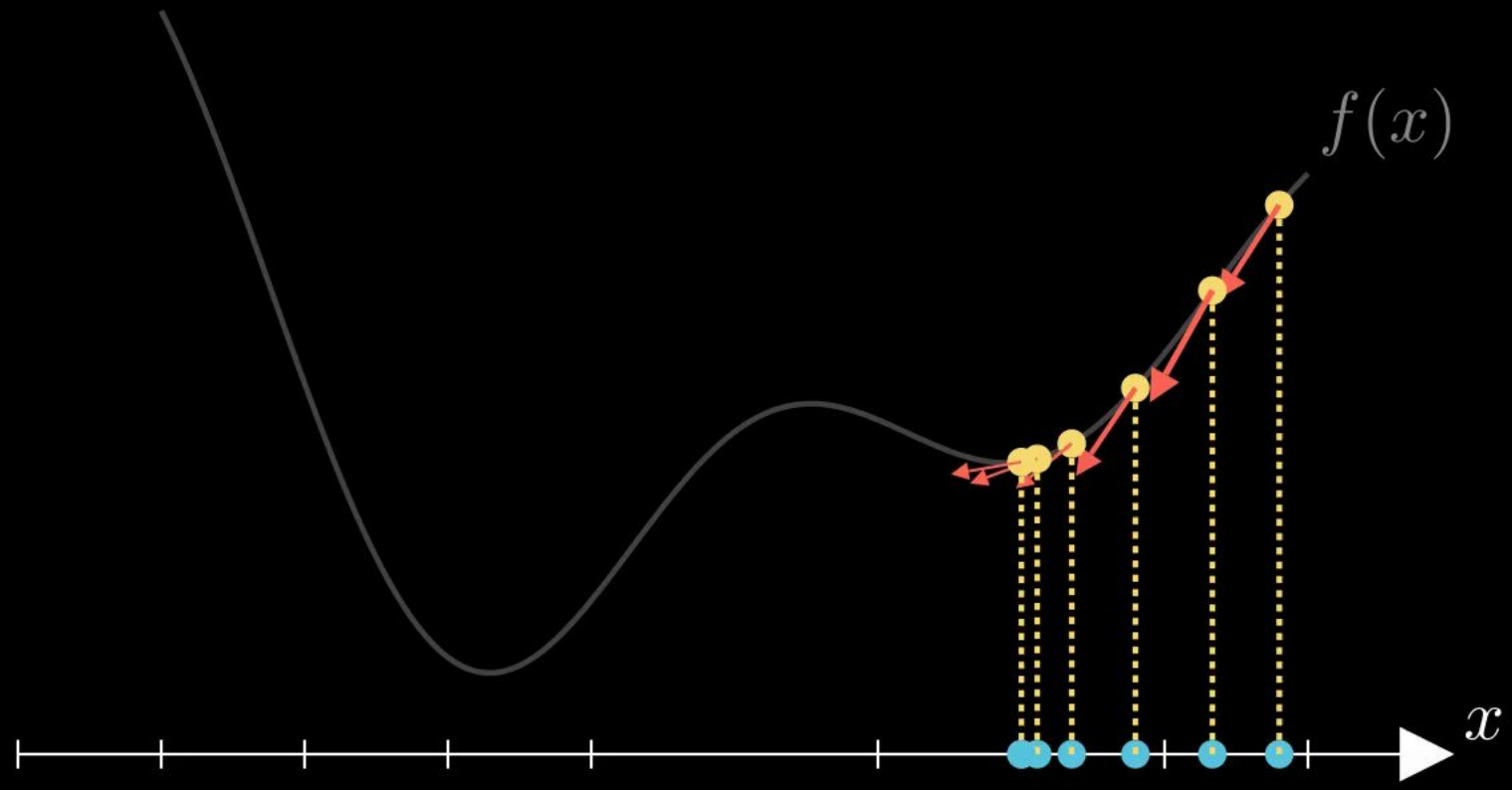

$f(x)$
x

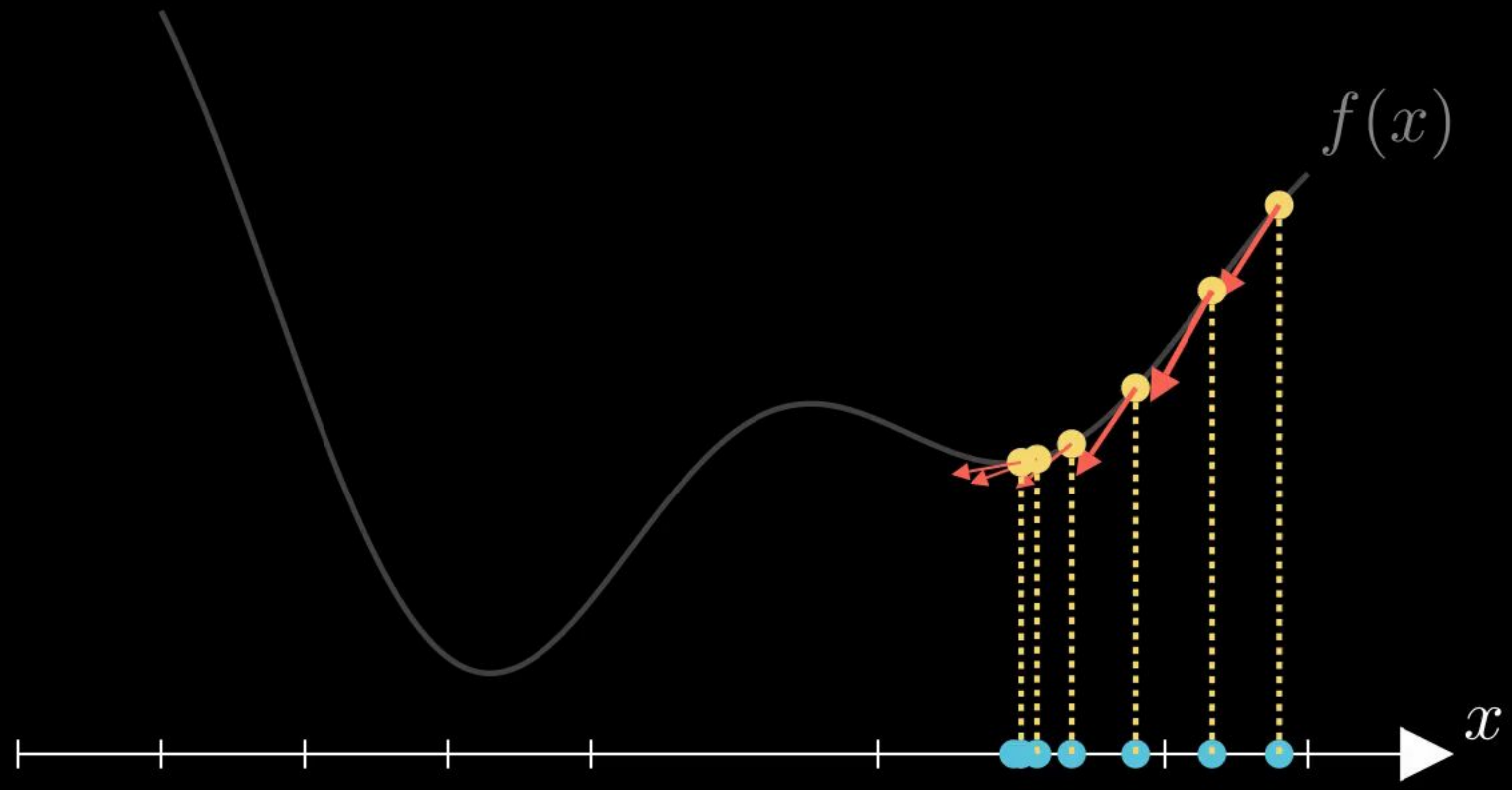

$f(x)$
x

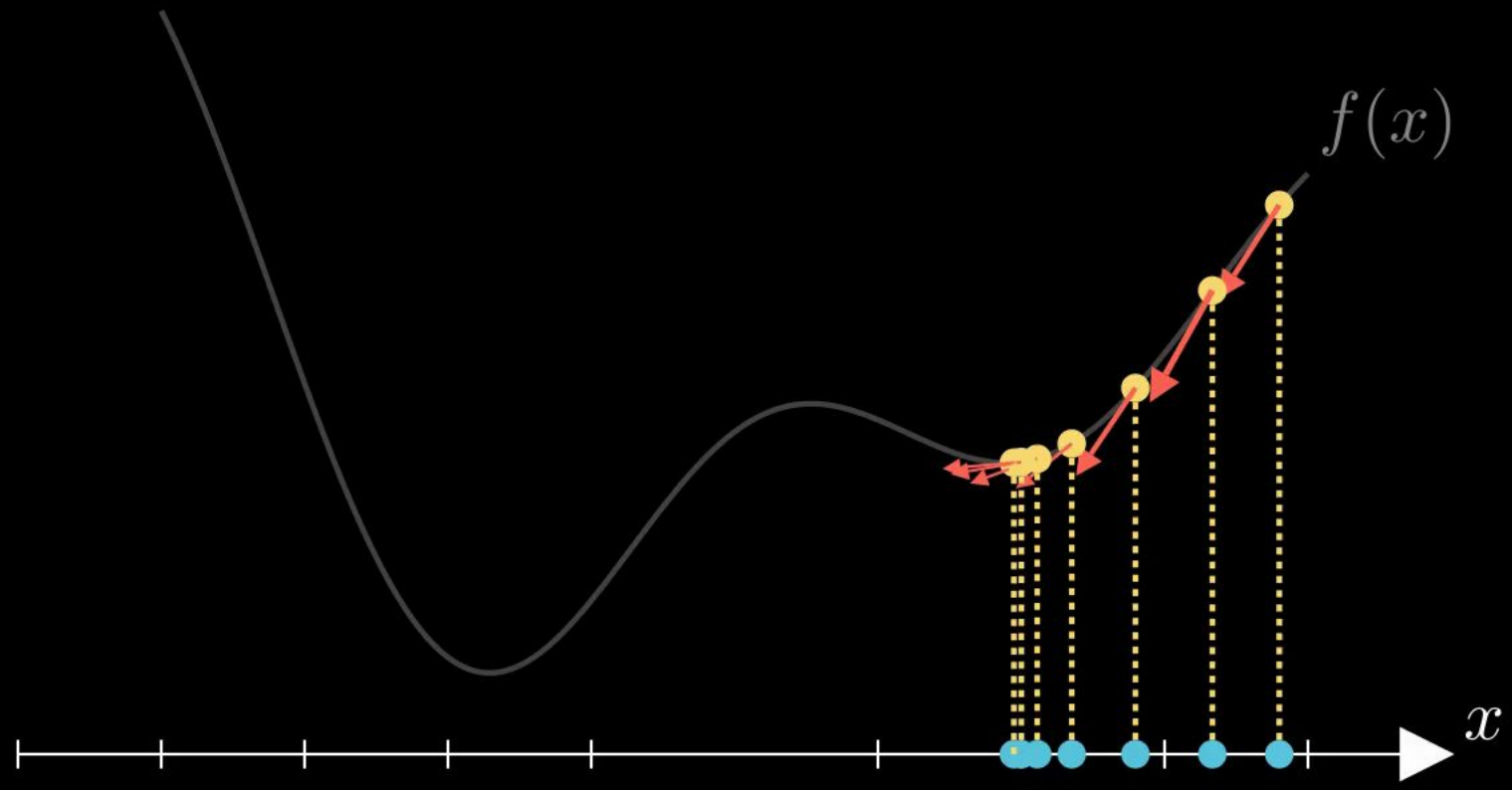

$f(x)$
x

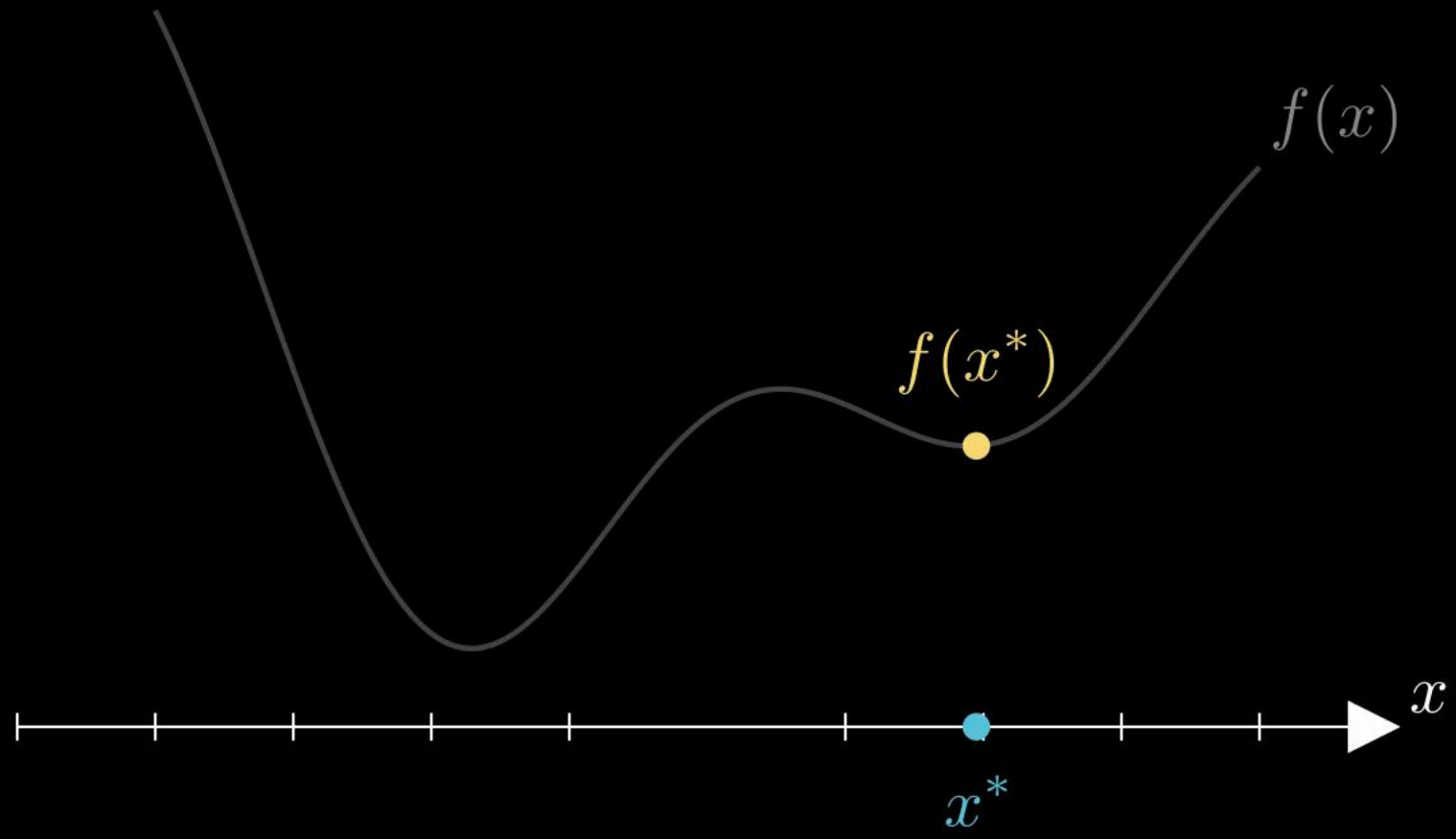

$f(x)$
x


$f(x)$
x


$f(x)$
x


$f(x)$
x


$f(x)$
x

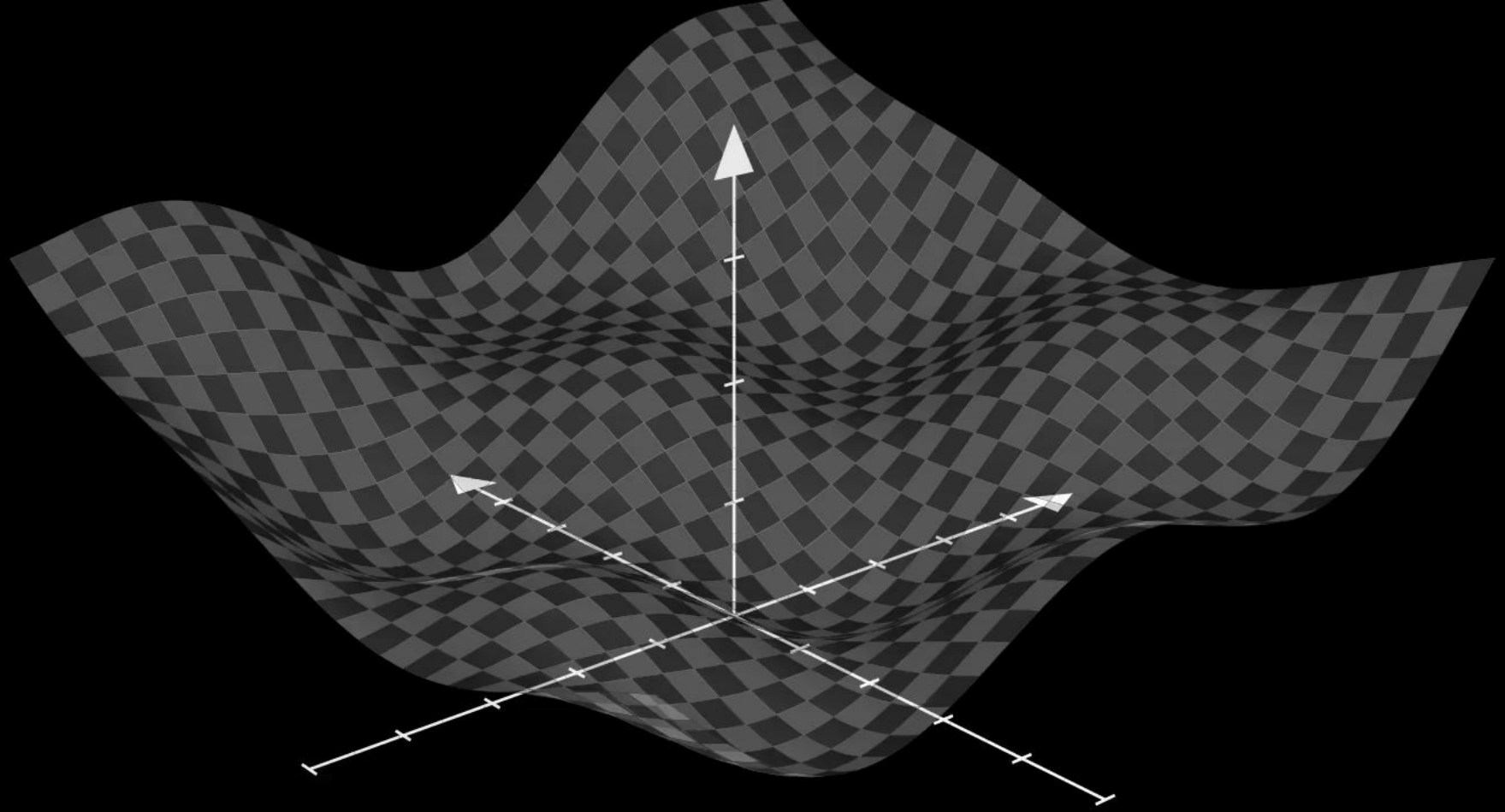

$f(x)$
$f(x^*)$
x
x^*

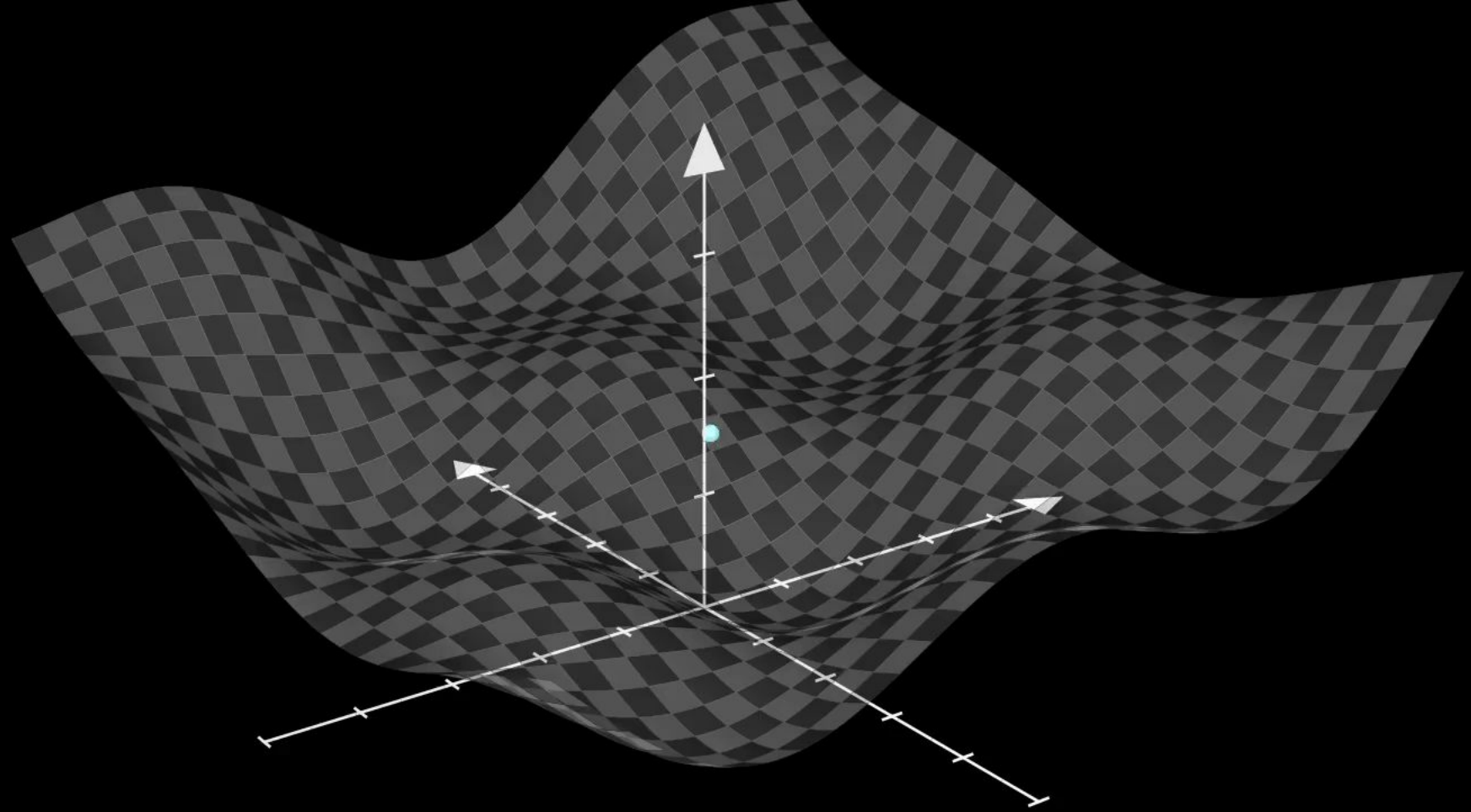
$$\min_{\vec{x}} \; f(\vec{x})$$

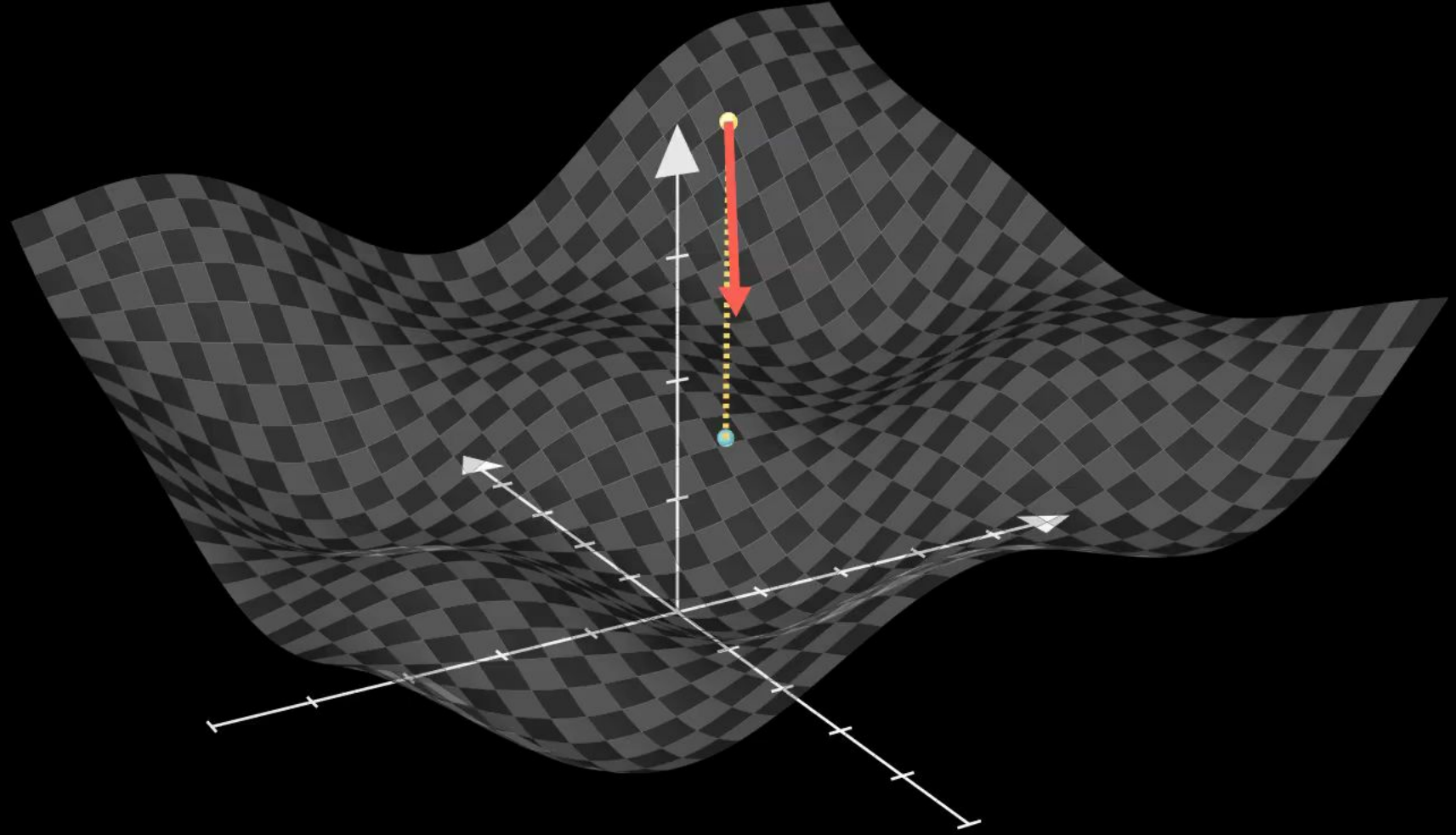
s.t.

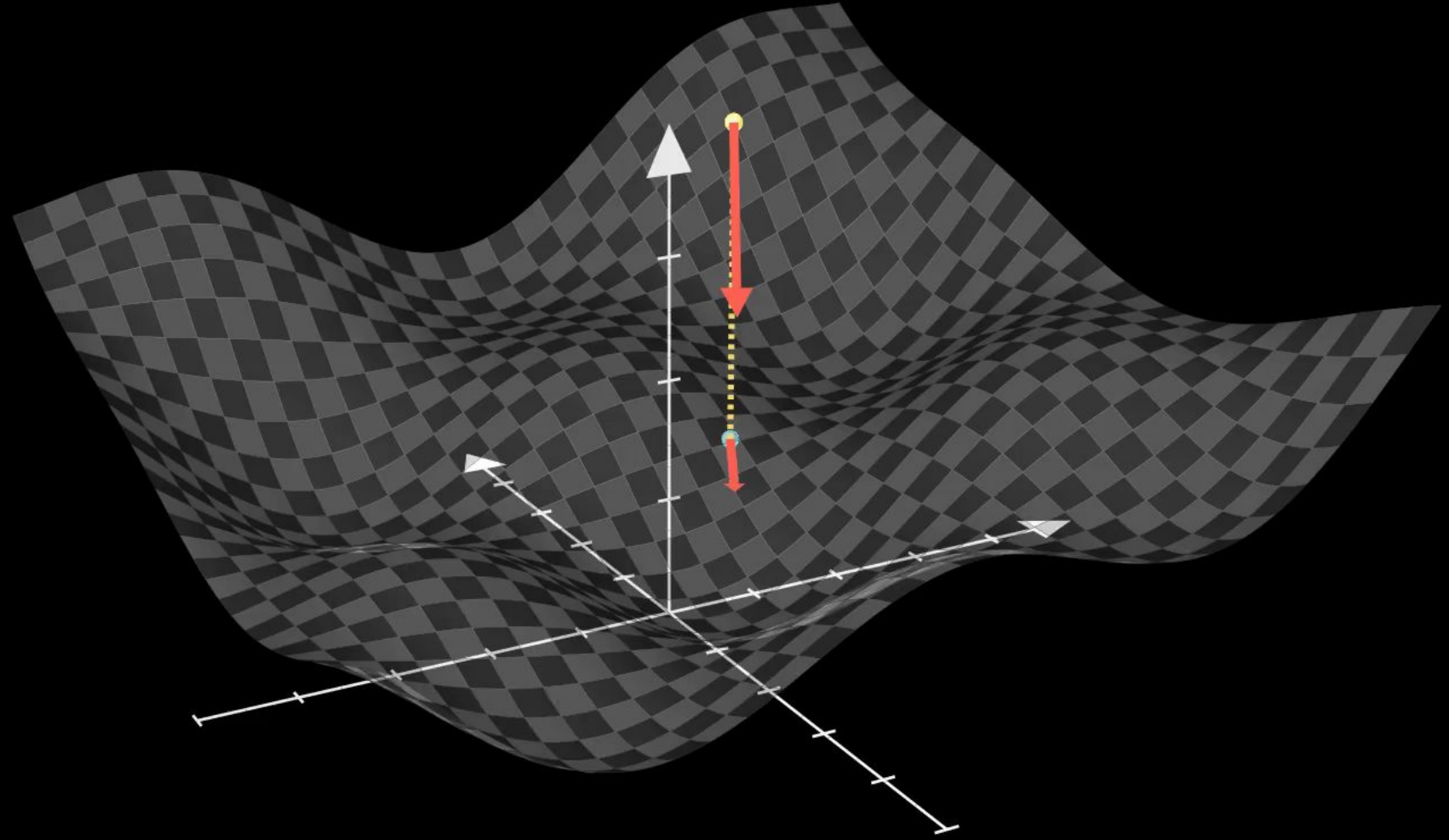
$$g_1(\vec{x}) \leq 0$$
$$g_2(\vec{x}) \leq 0$$
$$g_3(\vec{x}) \leq 0$$
$$\vdots$$

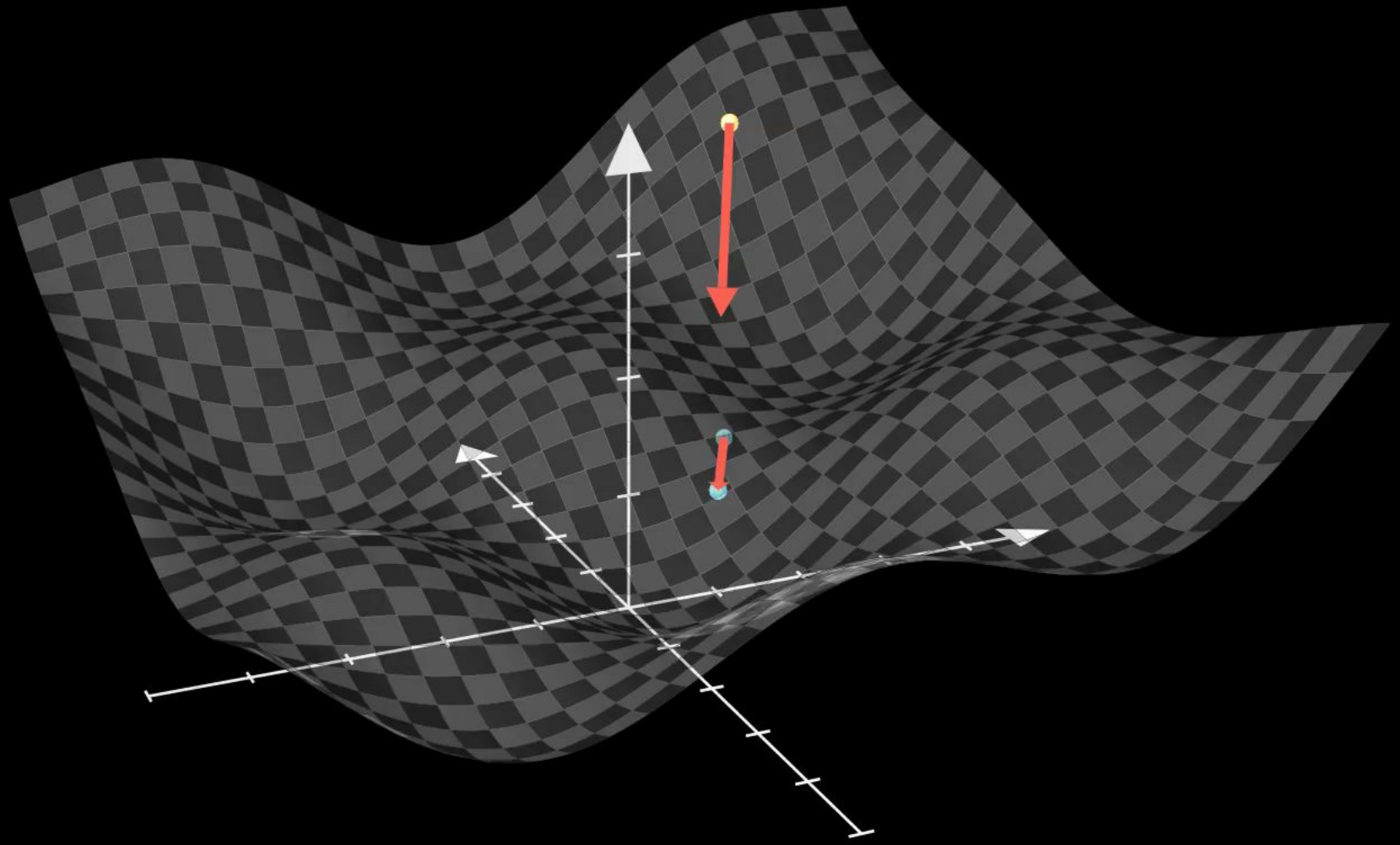
2. Multi-variable optimization

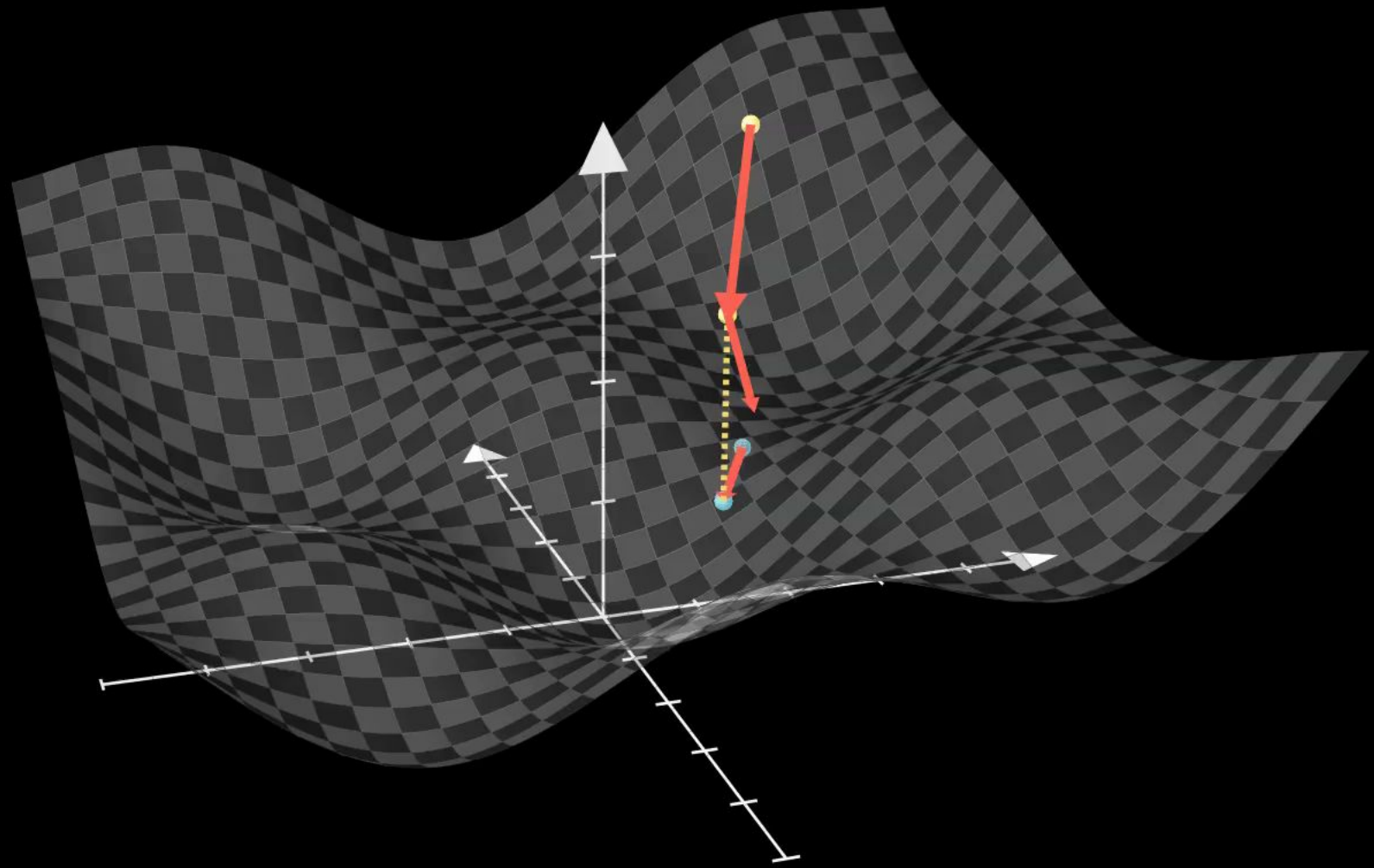


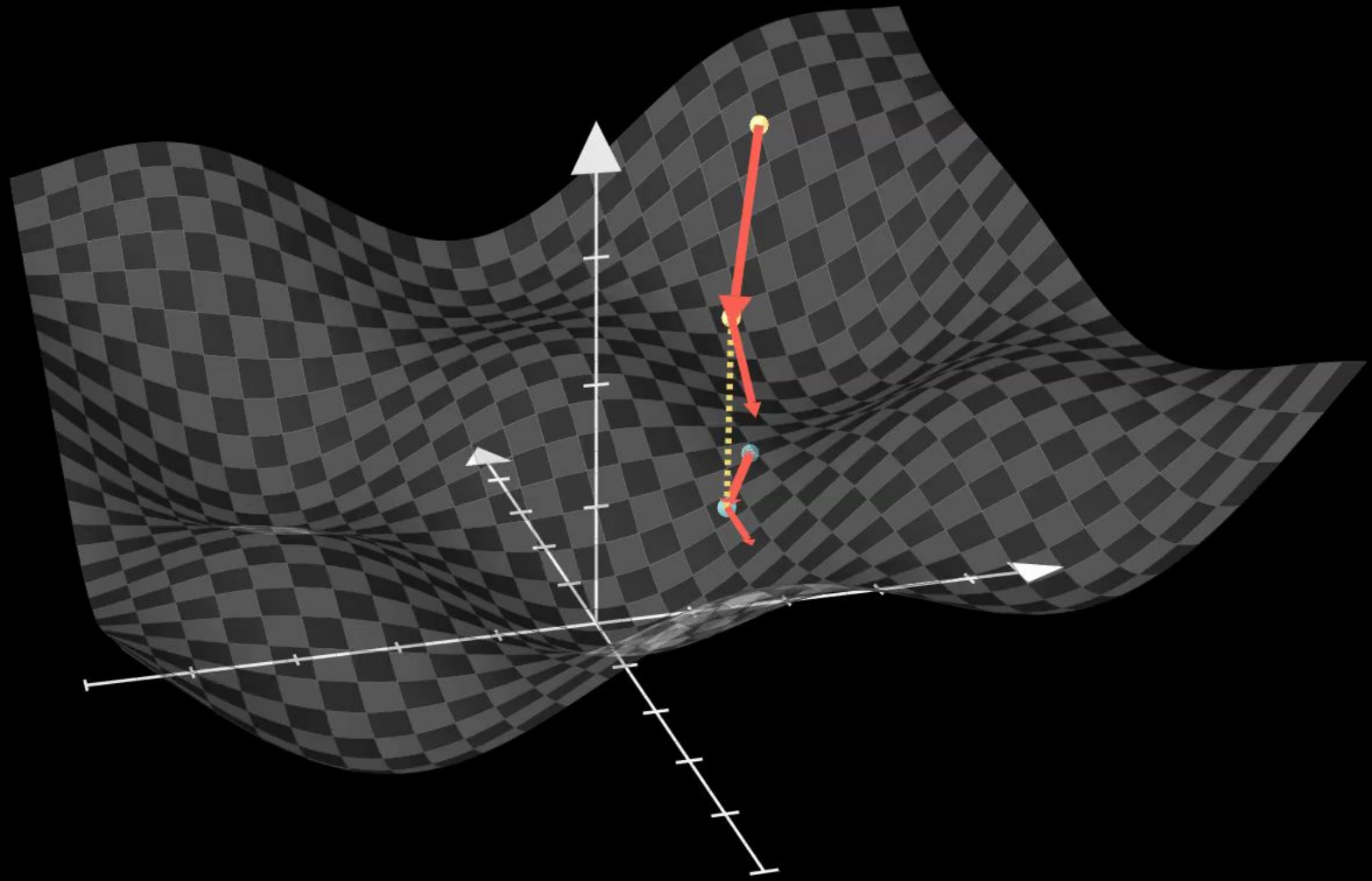


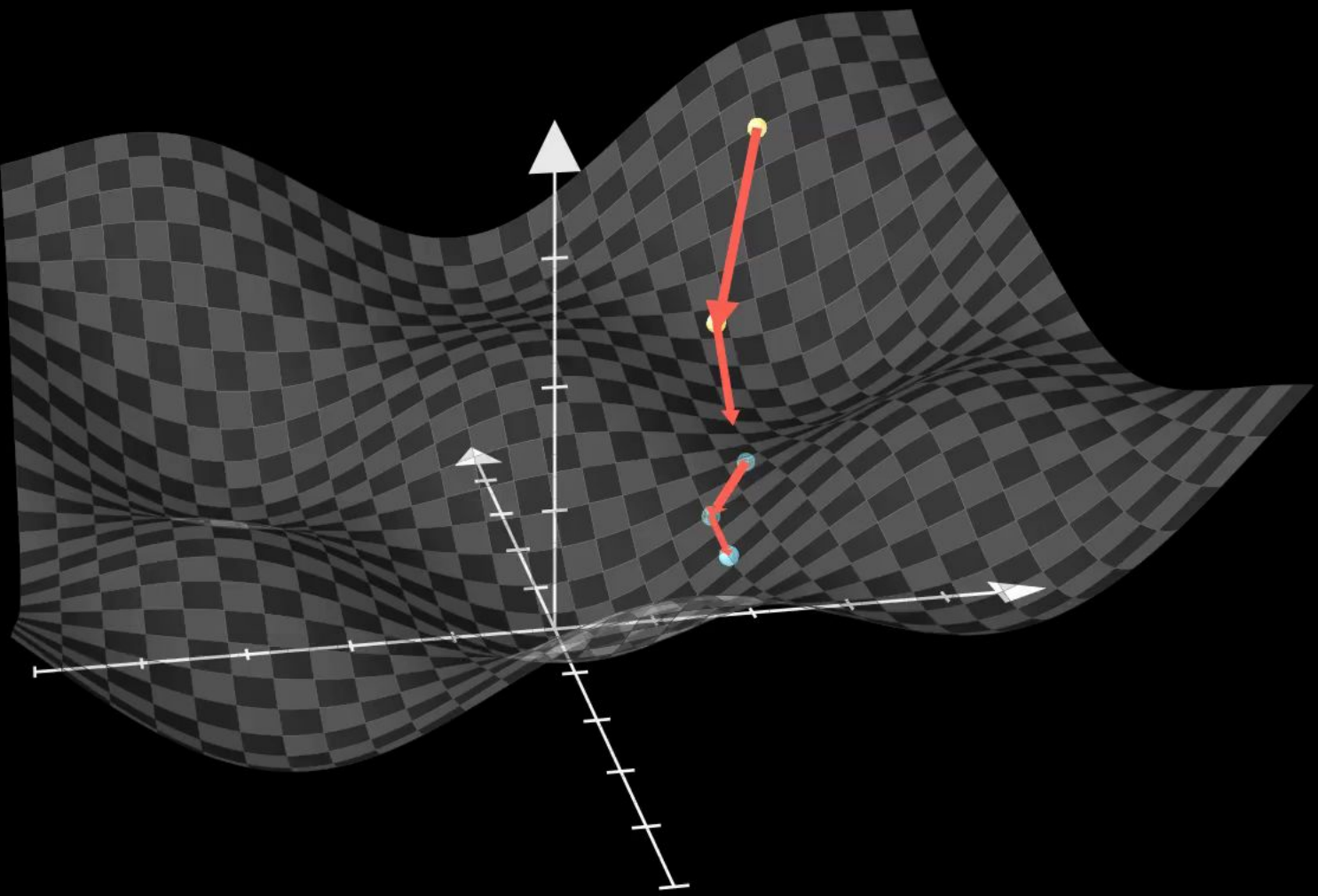


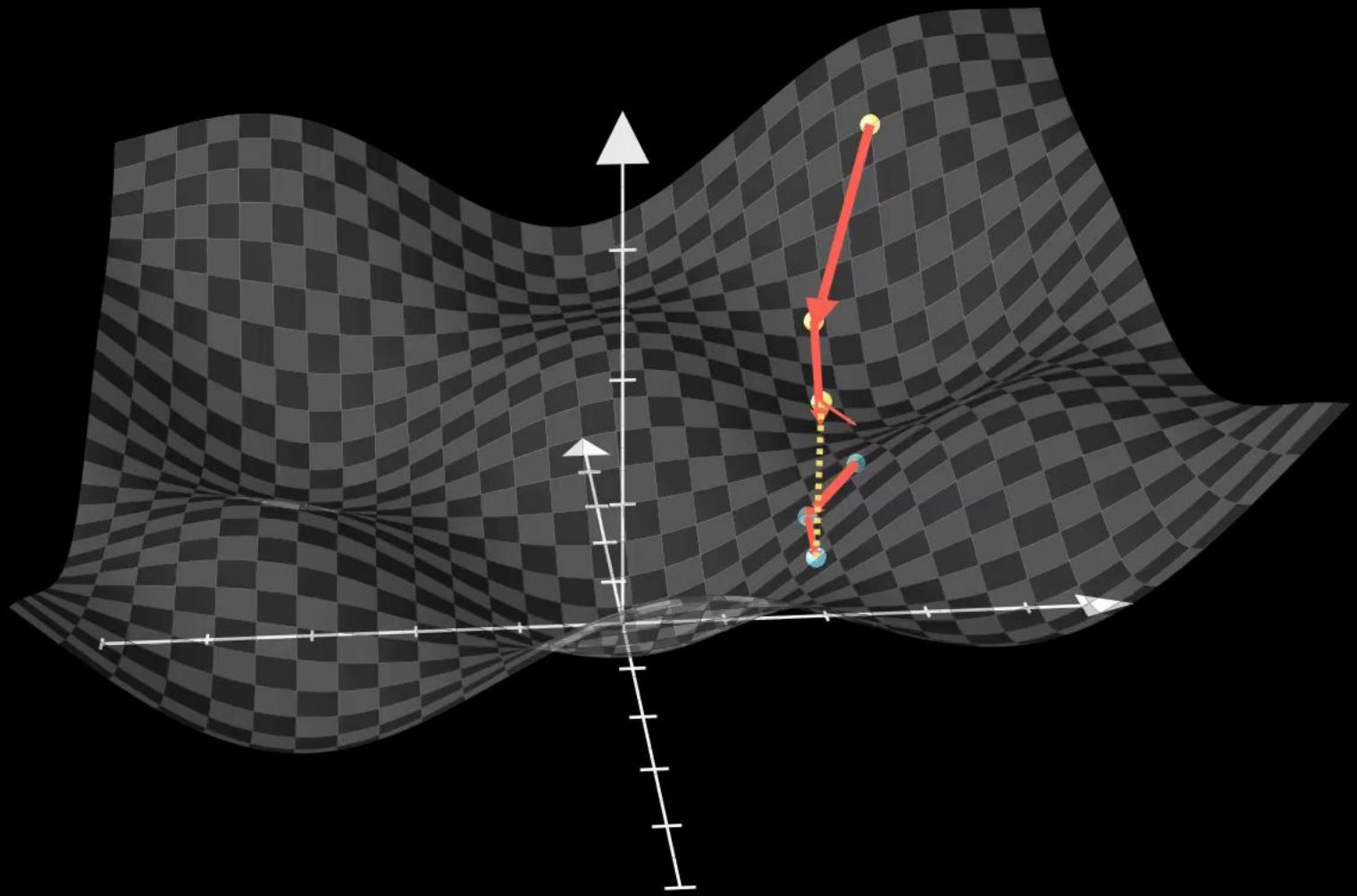


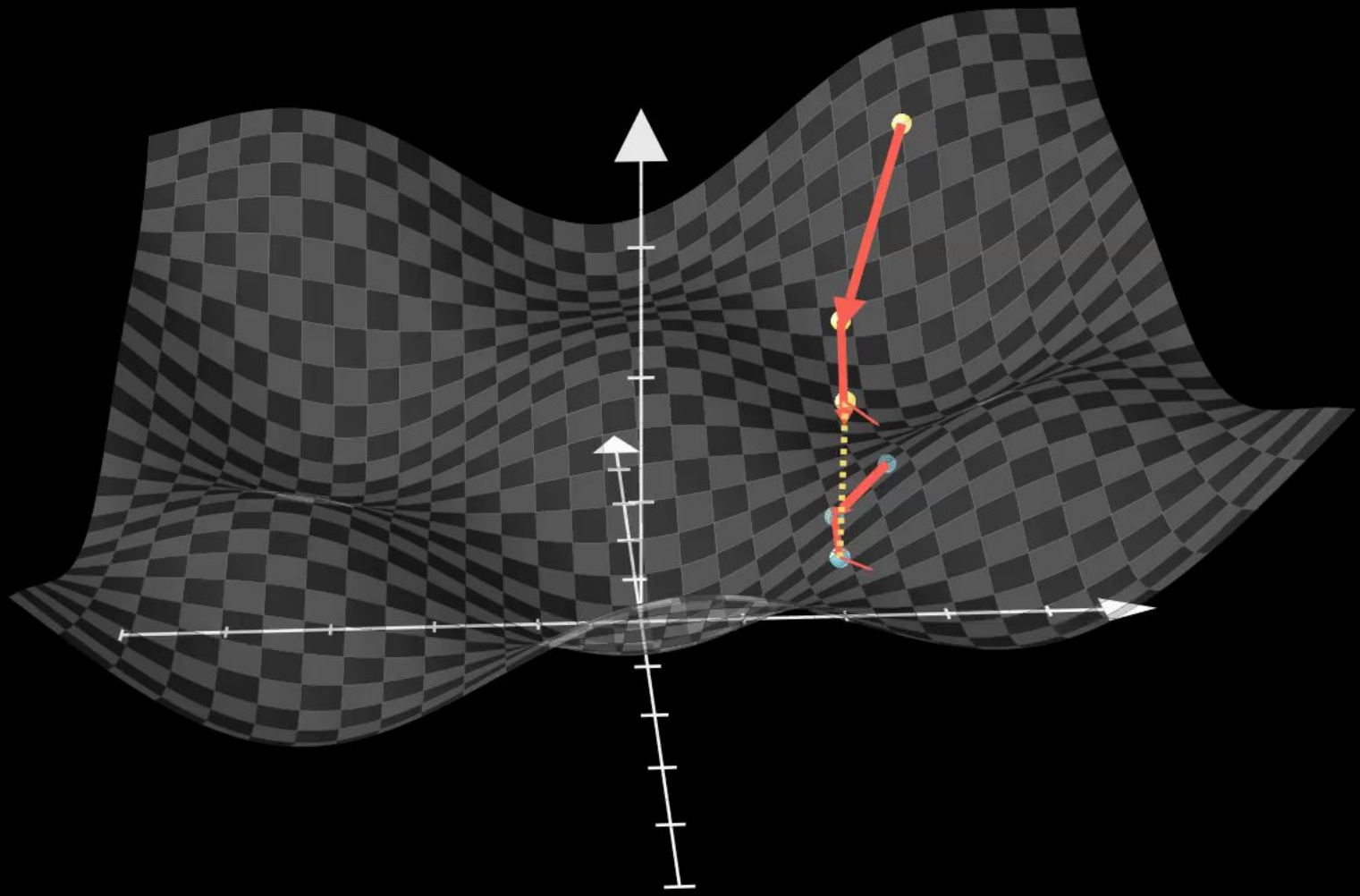


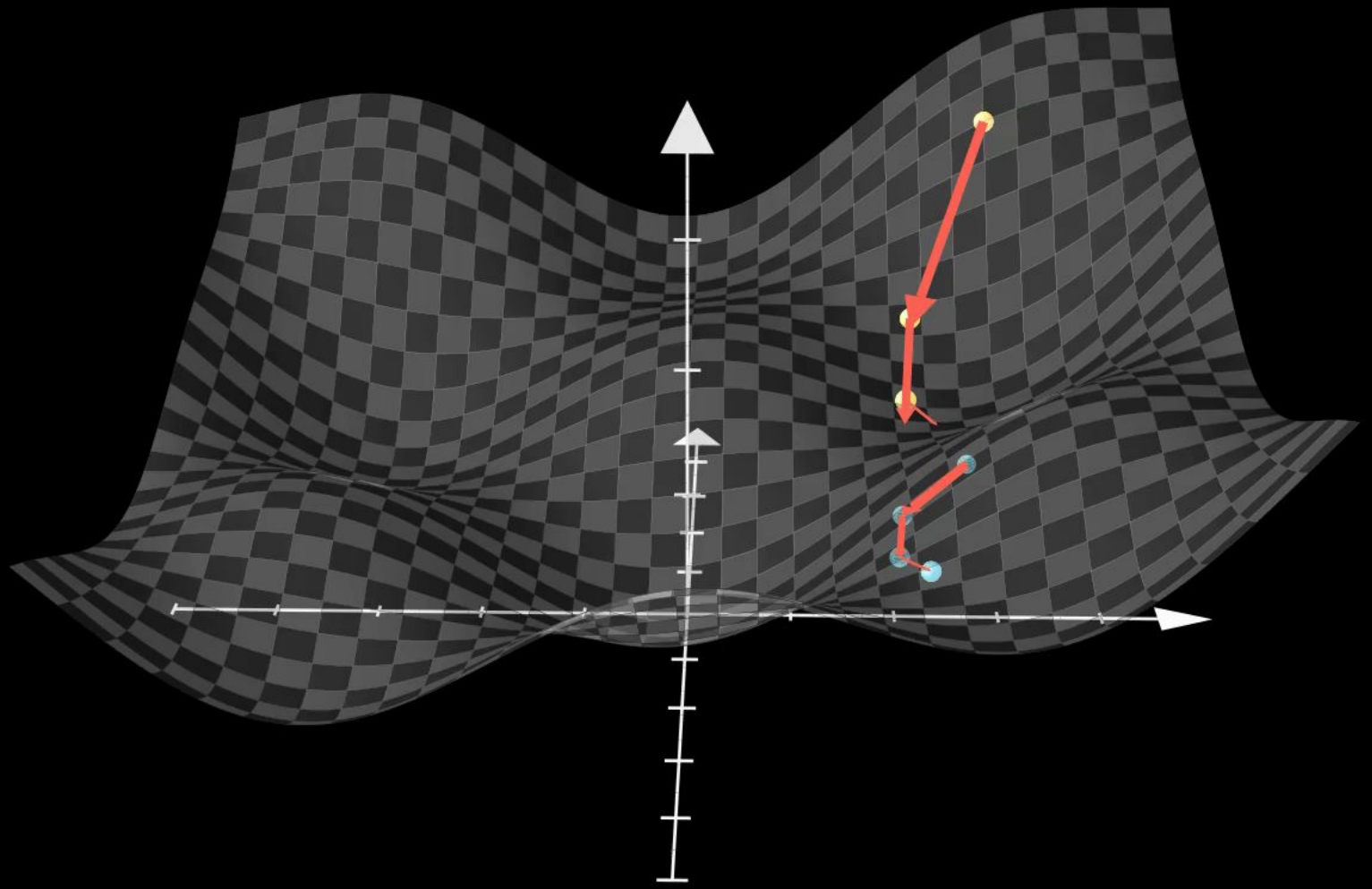


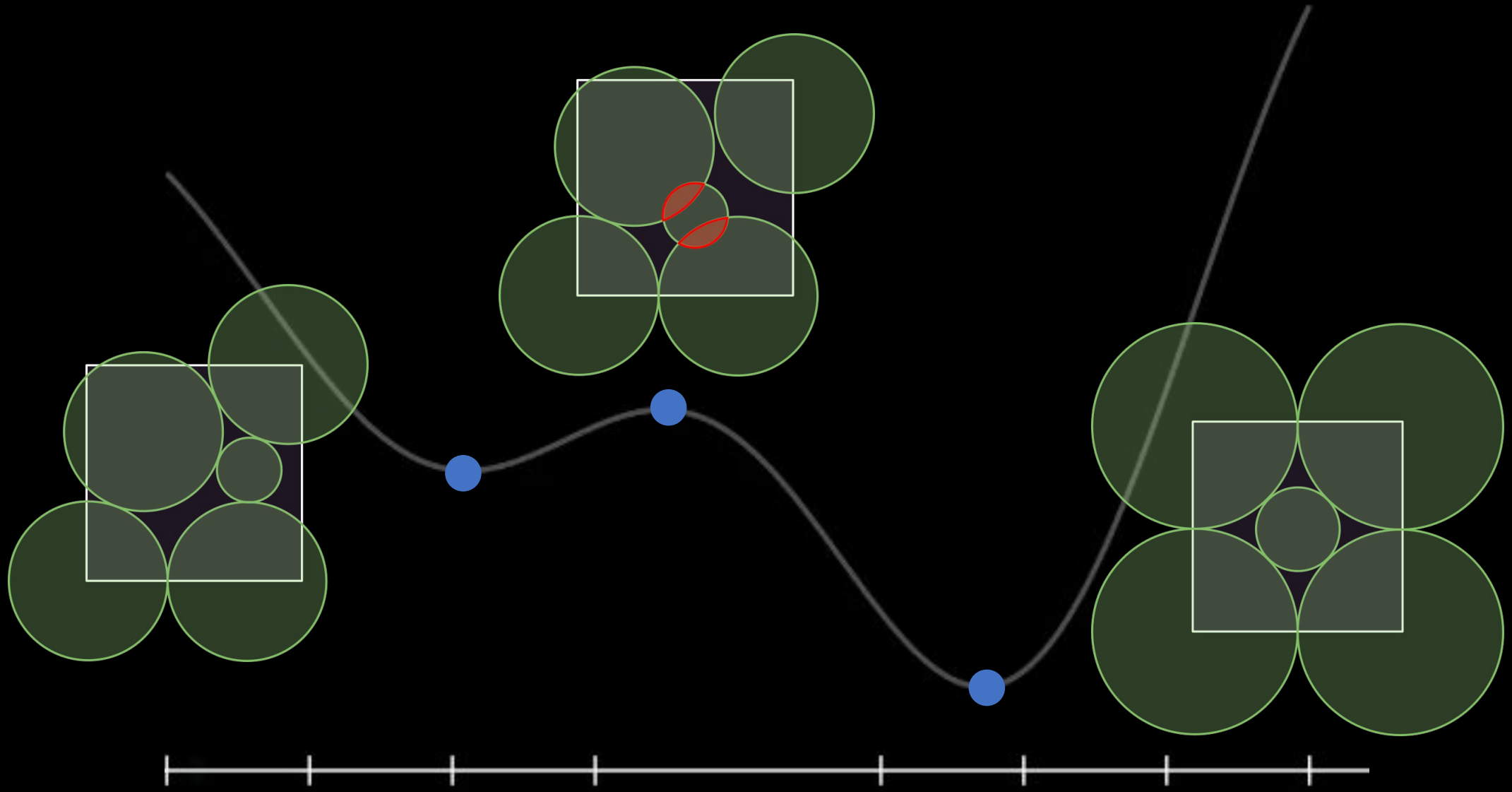










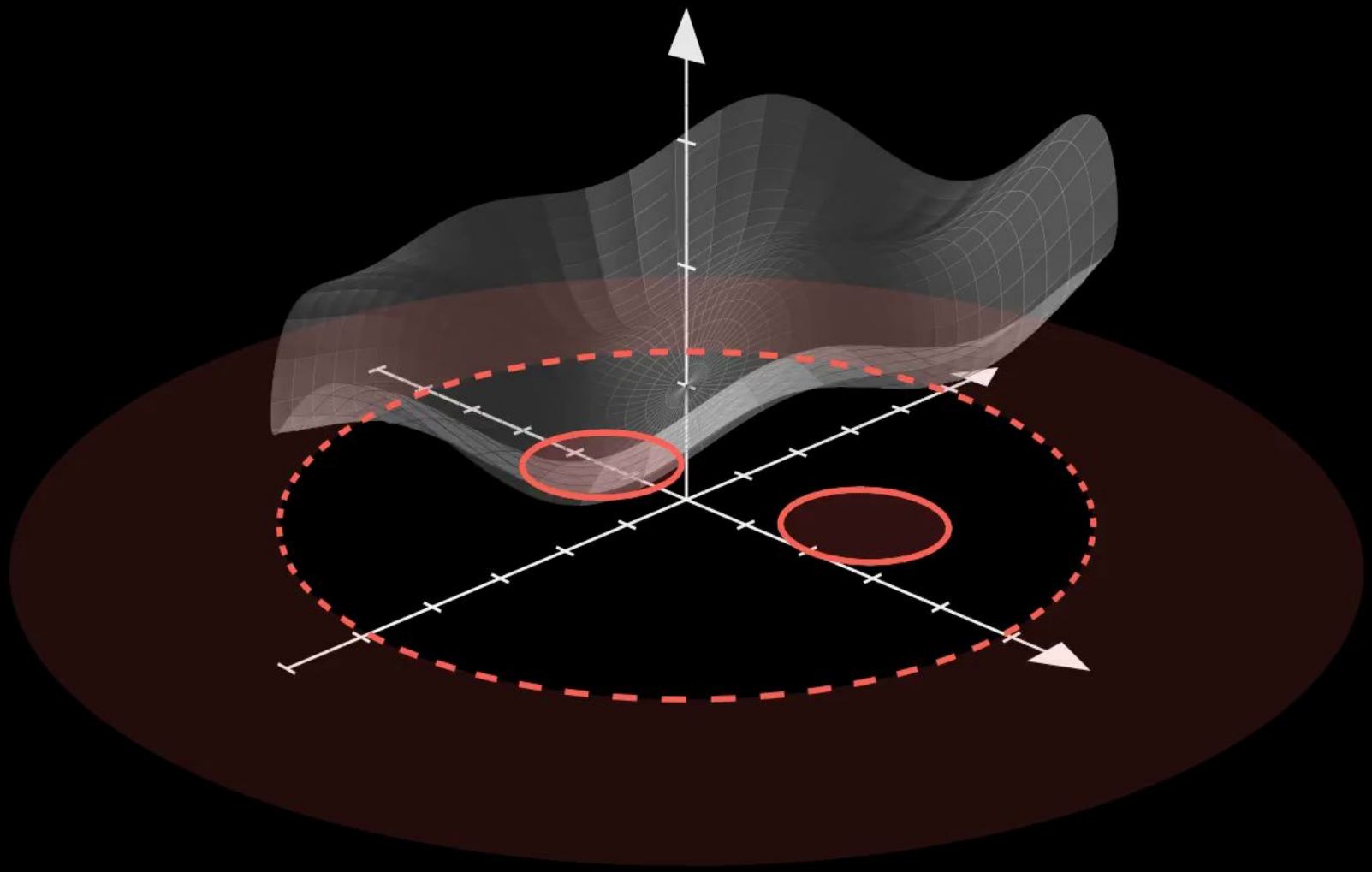


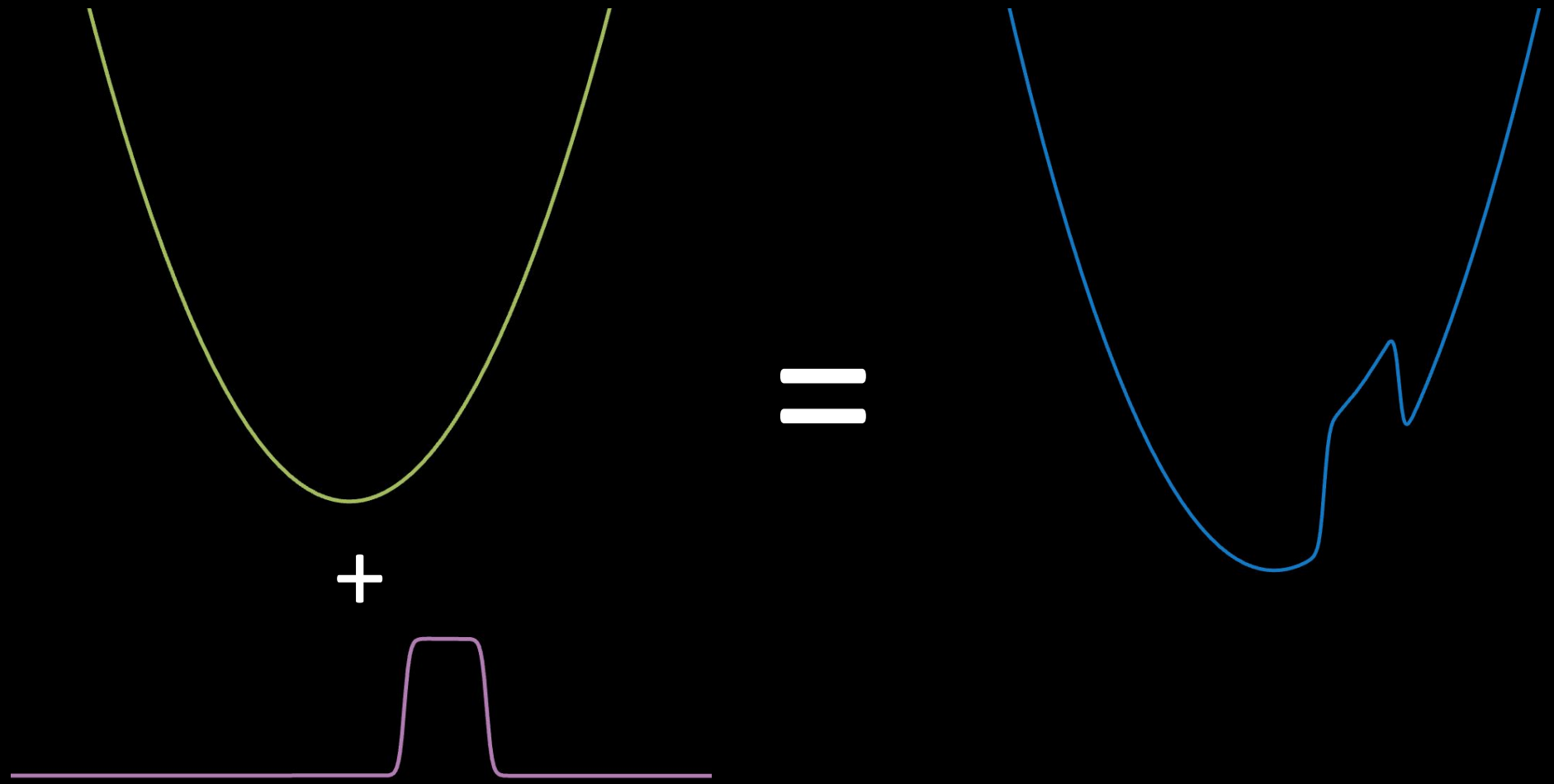
$$\min_{\vec{x}} f(\vec{x})$$

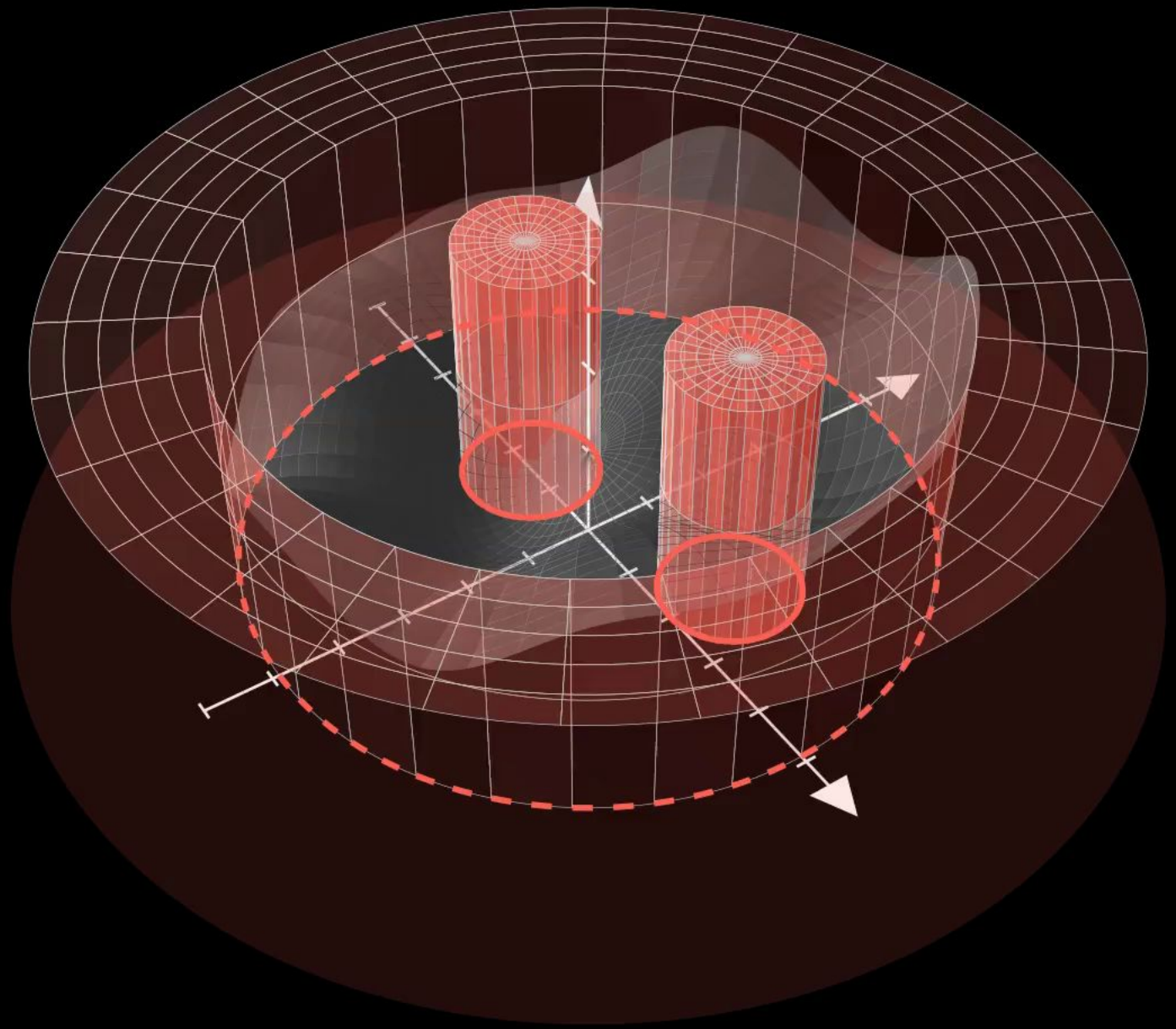
s.t.

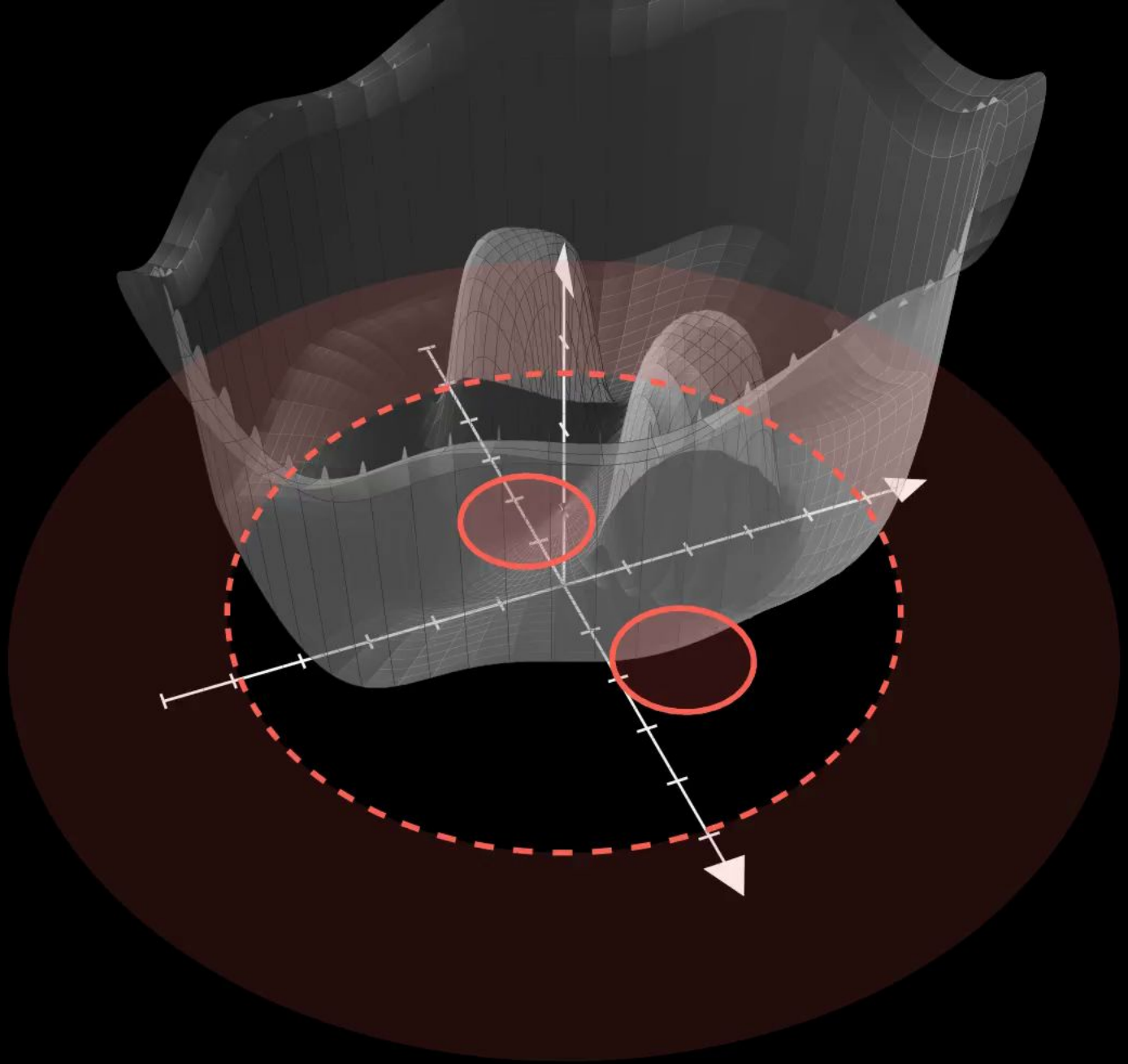
$$g_1(\vec{x}) \leq 0$$
$$g_2(\vec{x}) \leq 0$$
$$g_3(\vec{x}) \leq 0$$
$$\vdots$$

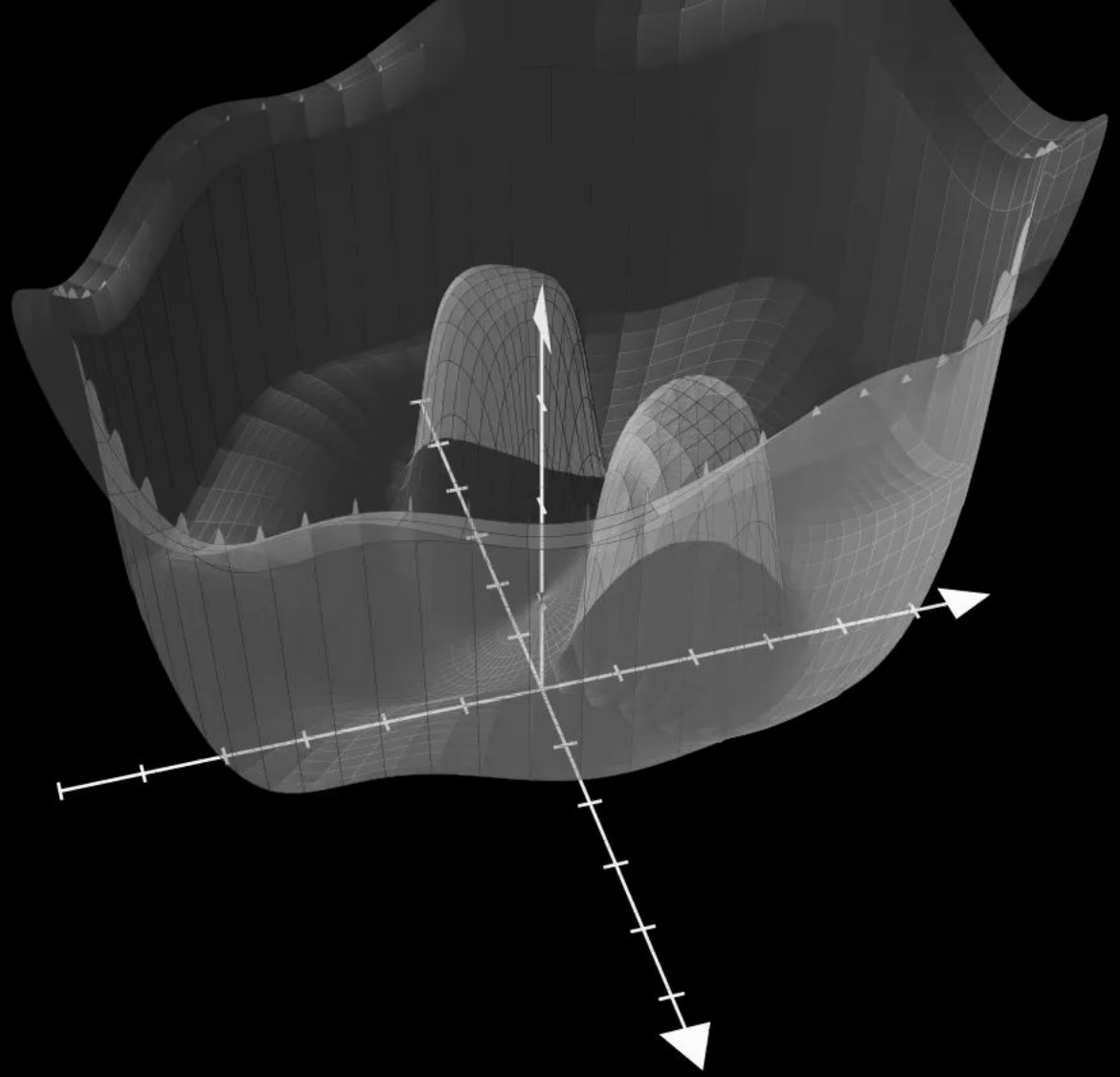
3. Constrained optimization





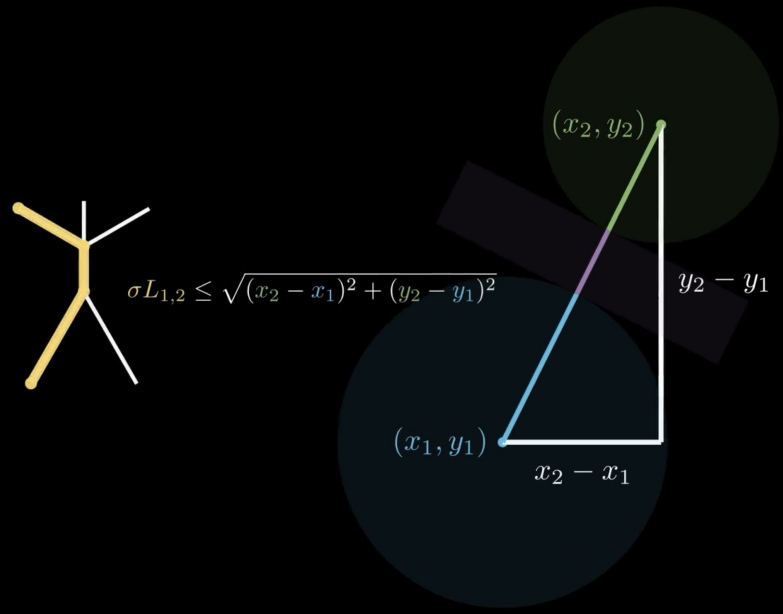






Putting it all together

Tree -> optimization problem -> optimal packing



$$\sigma L_{1,2} \leq \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
$$\sigma L_{1,3} \leq \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$
$$\sigma L_{1,4} \leq \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2}$$
$$\sigma L_{1,5} \leq \sqrt{(x_5 - x_1)^2 + (y_5 - y_1)^2}$$
$$\sigma L_{2,3} \leq \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2}$$
$$\sigma L_{2,4} \leq \sqrt{(x_4 - x_2)^2 + (y_4 - y_2)^2}$$
$$\sigma L_{2,5} \leq \sqrt{(x_5 - x_2)^2 + (y_5 - y_2)^2}$$
$$\sigma L_{3,4} \leq \sqrt{(x_4 - x_3)^2 + (y_4 - y_3)^2}$$
$$\sigma L_{3,5} \leq \sqrt{(x_5 - x_3)^2 + (y_5 - y_3)^2}$$
$$\sigma L_{4,5} \leq \sqrt{(x_5 - x_4)^2 + (y_5 - y_4)^2}$$

$$\min_{\vec{x}} f(\vec{x})$$

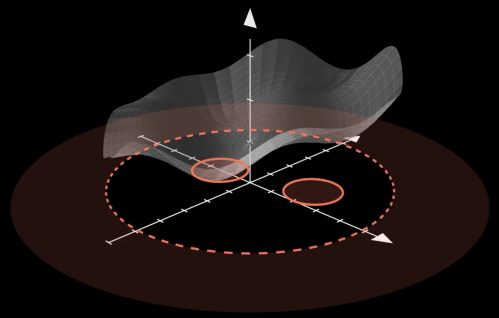
s.t.

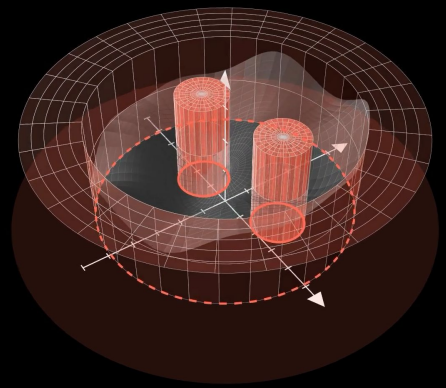
$$g_1(\vec{x}) \leq 0$$
$$g_2(\vec{x}) \leq 0$$
$$g_3(\vec{x}) \leq 0$$
$$\vdots$$

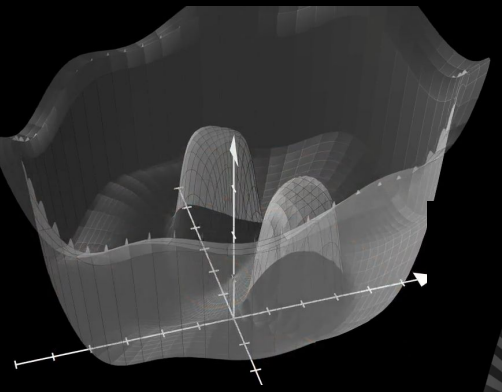
$$\min_{\vec{x}} f(\vec{x})$$

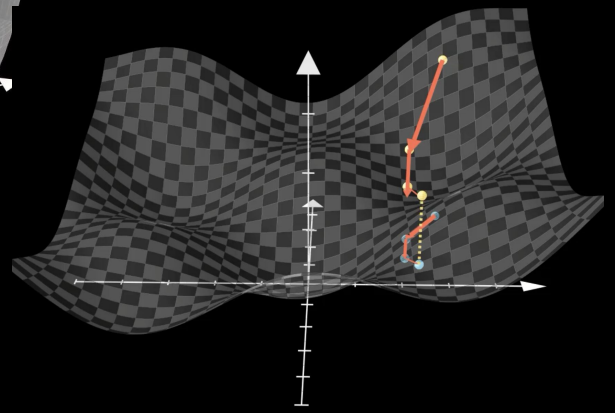
s.t.

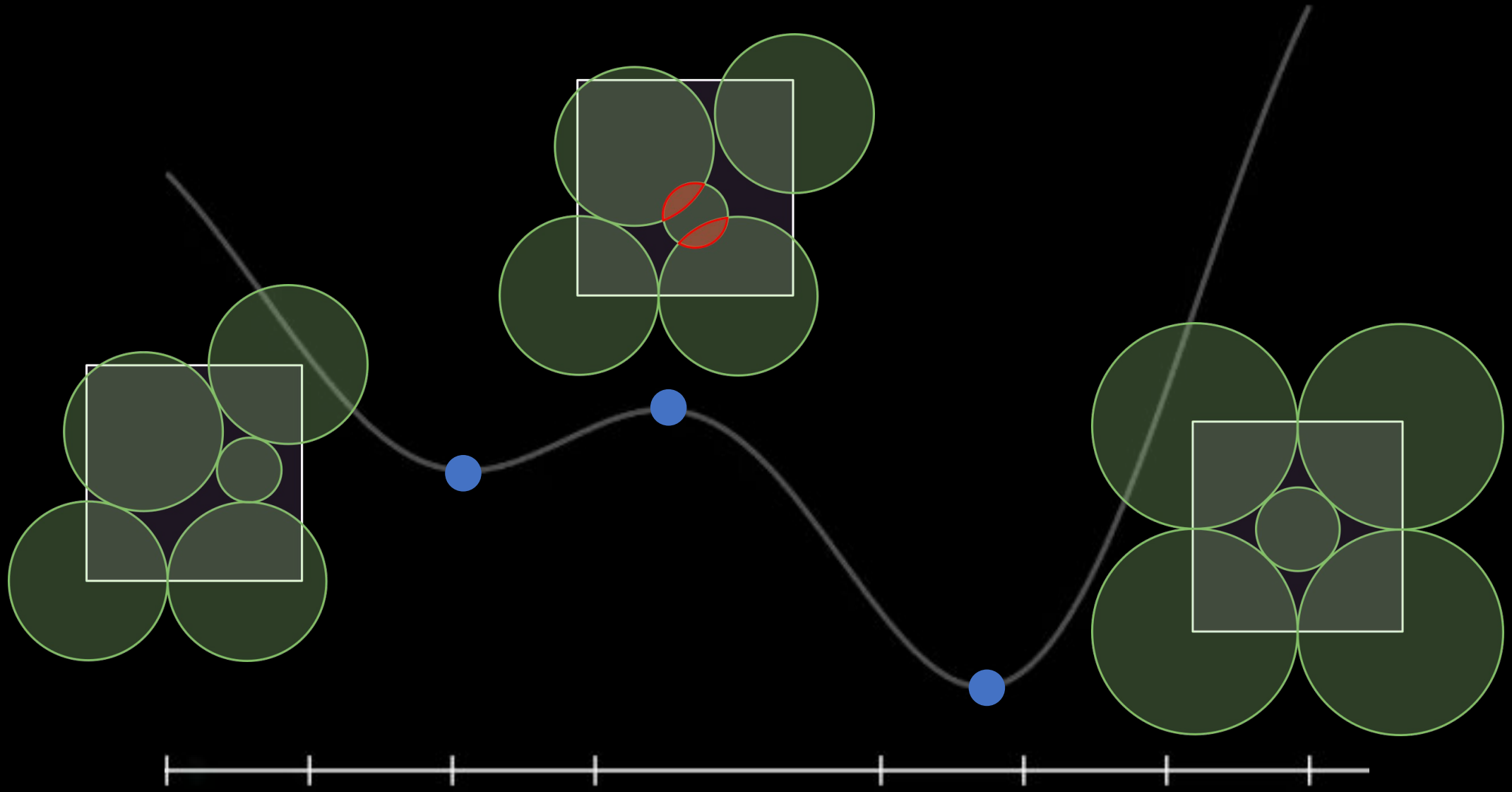
$$g_1(\vec{x}) \leq 0$$
$$g_2(\vec{x}) \leq 0$$
$$g_3(\vec{x}) \leq 0$$
$$\vdots$$







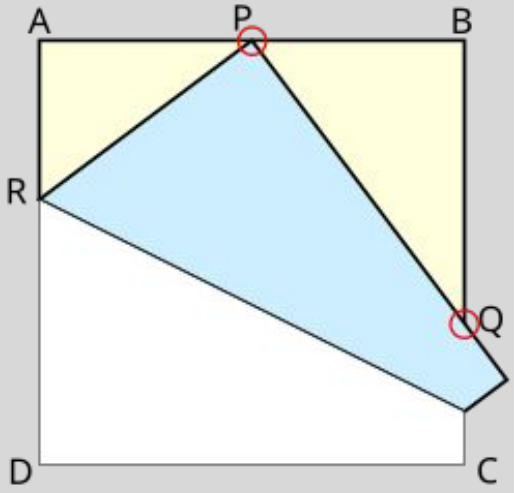




(End of lesson)

Primary school

Secondary school



Post-secondary

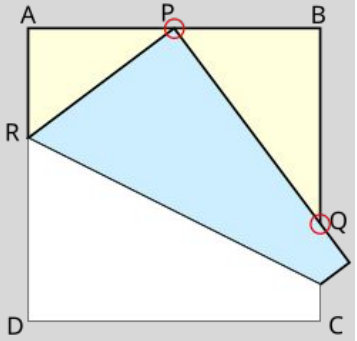
?

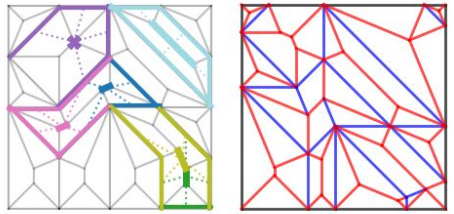
Academia

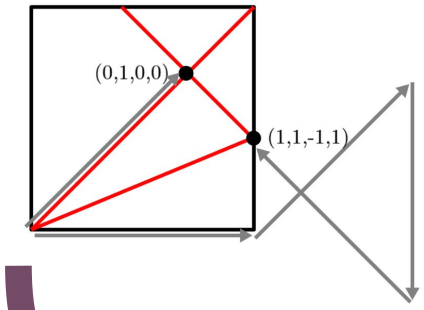
- **Tortilla-tortilla constraint** (i, j, k, l): Creases e_{ij}, e_{kl} overlap, so there are points $p \in e_{ij}$ and $q \in e_{kl}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $P' \subset F_i \cup F_j \cup e_{ij}$ and $Q' \subset F_k \cup F_l \cup e_{kl}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P' \cap e_{ij}$, $q' \in Q' \cap e_{kl}$ such that $f(p') = f(q')$. For ε' sufficiently small, there are points $p^+ \in F_i \cap P'$, $q^+ \in F_k \cap Q'$ with $f(p^+) = f(q^+)$ and points $p^- \in F_j \cap P'$, $q^- \in F_l \cap Q'$ with $f(p^-) = f(q^-)$ such that the distances from p^+, p^- to p' along P', and from q^+, q^- to q' along Q', are all ε'. Then $\lambda(p^+, q^+) = \lambda(p^-, q^-)$ by the tortilla-tortilla condition. Because $p^+ \in F_i$, $p^- \in F_j$, $q^+ \in F_k$, and $q^- \in F_l$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jl}$.
- **Taco-tortilla constraint** (i, j, k, k): F_k and crease e_{ij} overlap, so there are points $p \in F_k$ and $q \in e_{ij}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $P' \subset F_k$ and $Q' \subset F_i \cup F_j \cup e_{ij}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P', q' \in Q' \cap e_{ij}$ such that $f(p') = f(q')$. For ε' sufficiently small there are points $q^+ \in F_i \cap Q', q^- \in F_j \cap Q', p^+ \in F_k \cap P'$ with $f(q^+) = f(q^-) = f(p^+)$ such that the distances from p^+ to p' along P', and from q^+, q^- to q' along Q', are all ε'. Then $\lambda(q^+, p^+) = \lambda(q^-, p^+)$ by the taco-tortilla condition. Because $q^+ \in F_i$, $q^- \in F_j$, and $p^+ \in F_k$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jk}$.
- **Taco-tortilla constraint** (i, j, k, l): Creases e_{ij}, e_{kl} overlap, so there are points $q \in e_{ij}$ and $p \in e_{kl}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $Q' \subset F_i \cup F_j \cup e_{ij}$ and $P' \subset F_k \cup F_l \cup e_{kl}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P' \cap e_{kl}$, $q' \in Q' \cap e_{ij}$ such that $f(p') = f(q')$. For ε' sufficiently small, there are points $q^+ \in F_i \cap Q', q^- \in F_j \cap Q', p^+ \in F_k \cap P'$ with $f(q^+) = f(q^-) = f(p^+)$ such that the distances from p^+ to p' along P', and from q^+, q^- to q' along Q', are all ε'. By the taco-tortilla condition, $\lambda(p^+, q^+) = \lambda(p^-, q^-)$. Because $p^+ \in F_i$, $p^- \in F_j$, and $q^+ \in F_k$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jl}$.

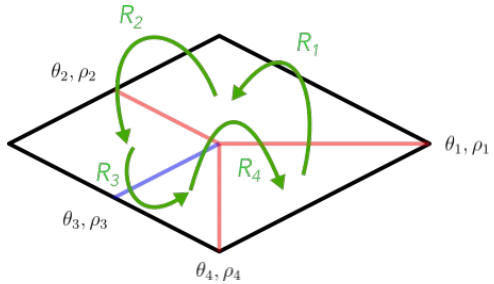
Primary school

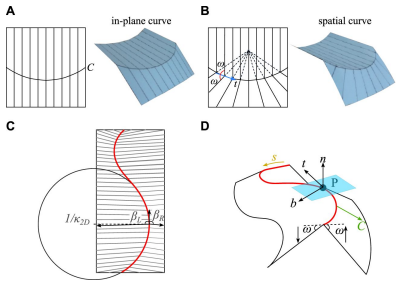
Secondary school



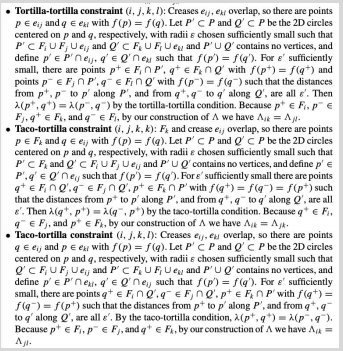








Academia

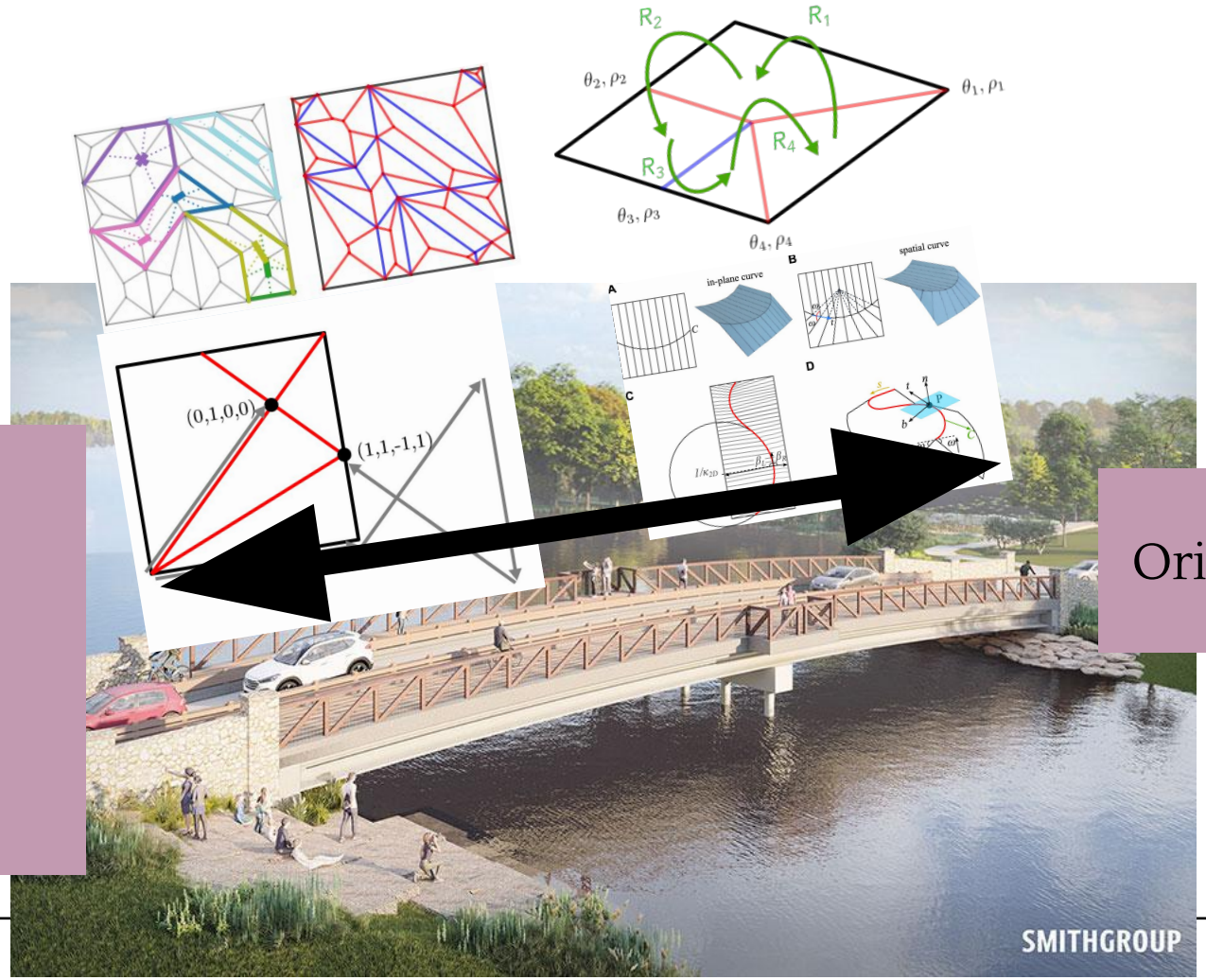


- **Tortilla-tortilla constraint** (i, j, k, l): Creases e_{ij}, e_{kl} overlap, so there are points $p \in e_{ij}$ and $q \in e_{kl}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $P' \subset F_i \cup F_j \cup e_{ij}$ and $Q' \subset F_k \cup F_l \cup e_{kl}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P' \cap e_{ij}$, $q' \in Q' \cap e_{kl}$ such that $f(p') = f(q')$. For ε' sufficiently small, there are points $p^+ \in F_i \cap P'$, $q^+ \in F_k \cap Q'$ with $f(p^+) = f(q^+)$ and points $p^- \in F_j \cap P'$, $q^- \in F_l \cap Q'$ with $f(p^-) = f(q^-)$ such that the distances from p^+, p^- to p' along P', and from q^+, q^- to q' along Q', are all ε'. Then $\lambda(p^+, q^+) = \lambda(p^-, q^-)$ by the tortilla-tortilla condition. Because $p^+ \in F_i$, $p^- \in F_j$, $q^+ \in F_k$, and $q^- \in F_l$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jl}$.
- **Taco-tortilla constraint** (i, j, k, k): F_k and crease e_{ij} overlap, so there are points $p \in F_k$ and $q \in e_{ij}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $P' \subset F_i \cup F_j \cup e_{ij}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P', q' \in Q' \cap e_{ij}$ such that $f(p') = f(q')$. For ε' sufficiently small there are points $q^+ \in F_i \cap Q', q^- \in F_j \cap Q', p^+ \in F_k \cap P'$ with $f(q^+) = f(q^-) = f(p^+)$ such that the distances from p^+ to p' along P', and from q^+, q^- to q' along Q', are all ε'. Then $\lambda(q^+, p^+) = \lambda(q^-, p^+)$ by the taco-tortilla condition. Because $q^+ \in F_i$, $q^- \in F_j$, and $p^+ \in F_k$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jk}$.
- **Taco-tortilla constraint** (i, j, k, l): Creases e_{ij}, e_{kl} overlap, so there are points $q \in e_{ij}$ and $p \in e_{kl}$ with $f(p) = f(q)$. Let $P' \subset P$ and $Q' \subset P$ be the 2D circles centered on p and q, respectively, with radii ε chosen sufficiently small such that $Q' \subset F_i \cup F_j \cup e_{ij}$ and $P' \subset F_k \cup F_l \cup e_{kl}$ and $P' \cup Q'$ contains no vertices, and define $p' \in P' \cap e_{kl}$, $q' \in Q' \cap e_{ij}$ such that $f(p') = f(q')$. For ε' sufficiently small, there are points $q^+ \in F_i \cap Q', q^- \in F_j \cap Q', p^+ \in F_k \cap P'$ with $f(q^+) = f(q^-) = f(p^+)$ such that the distances from p^+ to p' along P', and from q^+, q^- to q' along Q', are all ε'. By the taco-tortilla condition, $\lambda(p^+, q^+) = \lambda(p^-, q^-)$. Because $p^+ \in F_i$, $p^- \in F_j$, and $q^+ \in F_k$, by our construction of Λ we have $\Lambda_{ik} = \Lambda_{jl}$.

This is a huge gap/opportunity!

COMMUNITY BRIDGE

Math world



Origami world

My personal journey as a math student + origami nerd

Some things I folded at age 11

My origami design notebook, age 15

THANK YOU
QUESTIONS / OPEN DISCUSSION

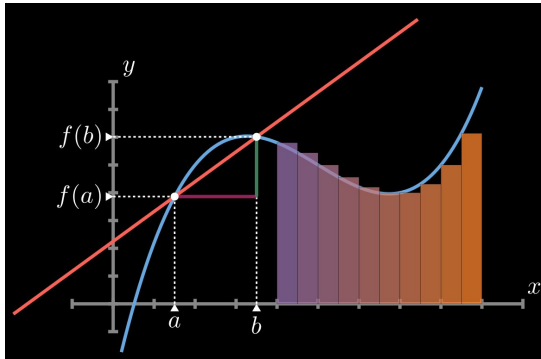
Ann Arbor, Michigan, USA CFC 6 May 14-17, 2026

REFERENCES AND IMAGE SOURCES

- https://www.origametria.com/storage/files/common/Origametria-ClipF1.mp4
- https://en.wikipedia.org/wiki/Mathematics_of_paper_folding#/media/File:Haga_theorem_1.svg
- Hugo A. Akitaya, Erik D. Demaine, and Jason S. Ku, "Computing Flat-Folded States", in *Origami^{8}: Proceedings of the 8th International Meeting on Origami in Science, Mathematics and Education (OSME 2024)*, Melbourne, Australia, July 16–18, 2024,
- Wim van Reese, 2.081 Plates and Shells lecture notes
- Song K, Li H, Li Y, Ma J and Zhou X (2024) A review of curved crease origami: design, analysis, and applications. Front. Phys. 12:1393435. doi: 10.3389/fphy.2024.1393435
- https://youtu.be/fTezUtQV62k?si=wNuReMG-Za3P1jua
- https://www.langorigami.com/
- https://youtu.be/7EuiXb6hFAM?si=bczI_VsOGl8Ibhw9
- Robert Lang, The math and magic of origami. https://youtu.be/NYKcOFQCeno?si=Oc7Y3SxPITTrmK1w

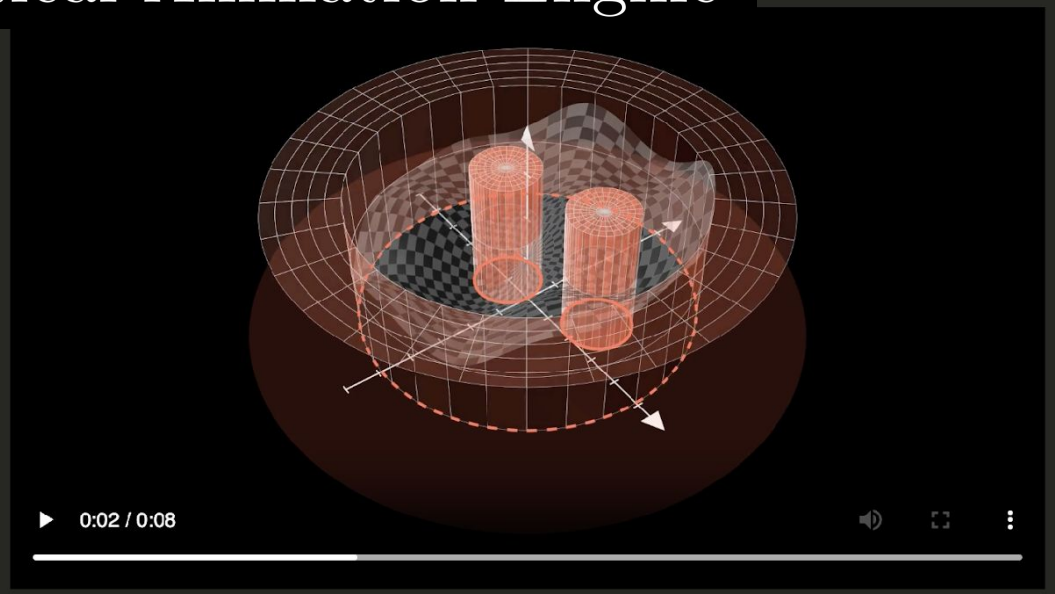
IF MORE TIME TO WORK ON PRESENTATION

- Update the 2d gradient descent: red/blue color by z. implement sgd/momentum.
- Also animate gradient descent on the constrained case
  - For easier debugging, separate the optimization from the animation. Run the optimizer, save results as json, see if it looks good, then animate.
- add captions/titles to the animations



Manim

Mathematical Animation Engine



How I made this presentation and previous math videos

```python
class GradientDescentConstraints(ThreeDSlide):
    def construct(self):
        self.play(Create(axes), Create(xy_constraints), Create(surface_base))
        self.begin_ambient_camera_rotation(rate=0.1)
        self.next_slide()

        # --- 4. The "Hard Wall" Volumes ---
        h = 5.5
        wall_style = {"u_range": [0, TAU], "v_range": [0, h], "resolution": (30, 2), "color": RED, "stroke_width": 0.5, "checkerboard_colors": [RED, RED_E]}
        roof_style = {"v_range": [0, TAU], "resolution": (5, 30), "color": RED, "stroke_width": 0.5, "checkerboard_colors": [RED, RED_E]}

        w_in1 = Surface(lambda th, z: axes.c2p(2 + np.cos(th), 1 + np.sin(th), z), fill_opacity=0.6, **wall_style)
        r_in1 = Surface(lambda r, th: axes.c2p(2 + r*np.cos(th), 1 + r*np.sin(th), h), u_range=[0, 1], fill_opacity=0.6, **roof_style)

        w_in2 = Surface(lambda th, z: axes.c2p(-1.5 + np.cos(th), np.sin(th), z), fill_opacity=0.6, **wall_style)
        r_in2 = Surface(lambda r, th: axes.c2p(-1.5 + r*np.cos(th), r*np.sin(th), h), u_range=[0, 1], fill_opacity=0.6, **roof_style)

        w_out = Surface(lambda th, z: axes.c2p(4.9*np.cos(th), 4.9*np.sin(th), z), fill_opacity=0.2, **wall_style)
        r_out = Surface(lambda r, th: axes.c2p(r*np.cos(th), r*np.sin(th), h), u_range=[4.9, 7.0], fill_opacity=0.2, **roof_style)

        hard_walls = VGroup(w_in1, r_in1, w_in2, r_in2, w_out, r_out)

        self.play(
            FadeIn(hard_walls),
            self.camera.phi_tracker.animate.set_value(45 * DEGREES),
            self.camera.zoom_tracker.animate.set_value(0.7),
            run_time=2
        )
        self.next_slide()

        # --- 5. Transition to Softmax Plateaus ---
        surface_plateaus = Surface(
            lambda r, theta: axes.c2p(r * np.cos(theta), r * np.sin(theta),
```

APPENDIX