Want to associate a model with observed features

Model could be a line

Model could be a line

Model could be a circle

Model could be a circle

Model could be an arbitrary shape

Model could be an arbitrary shape

Model could be a transformation

Model could be a transformation

Main Questions

What model(s) represent the set of features best?

Main Questions

What model(s) represent the set of features best?

How many model instances are there?

Main Questions

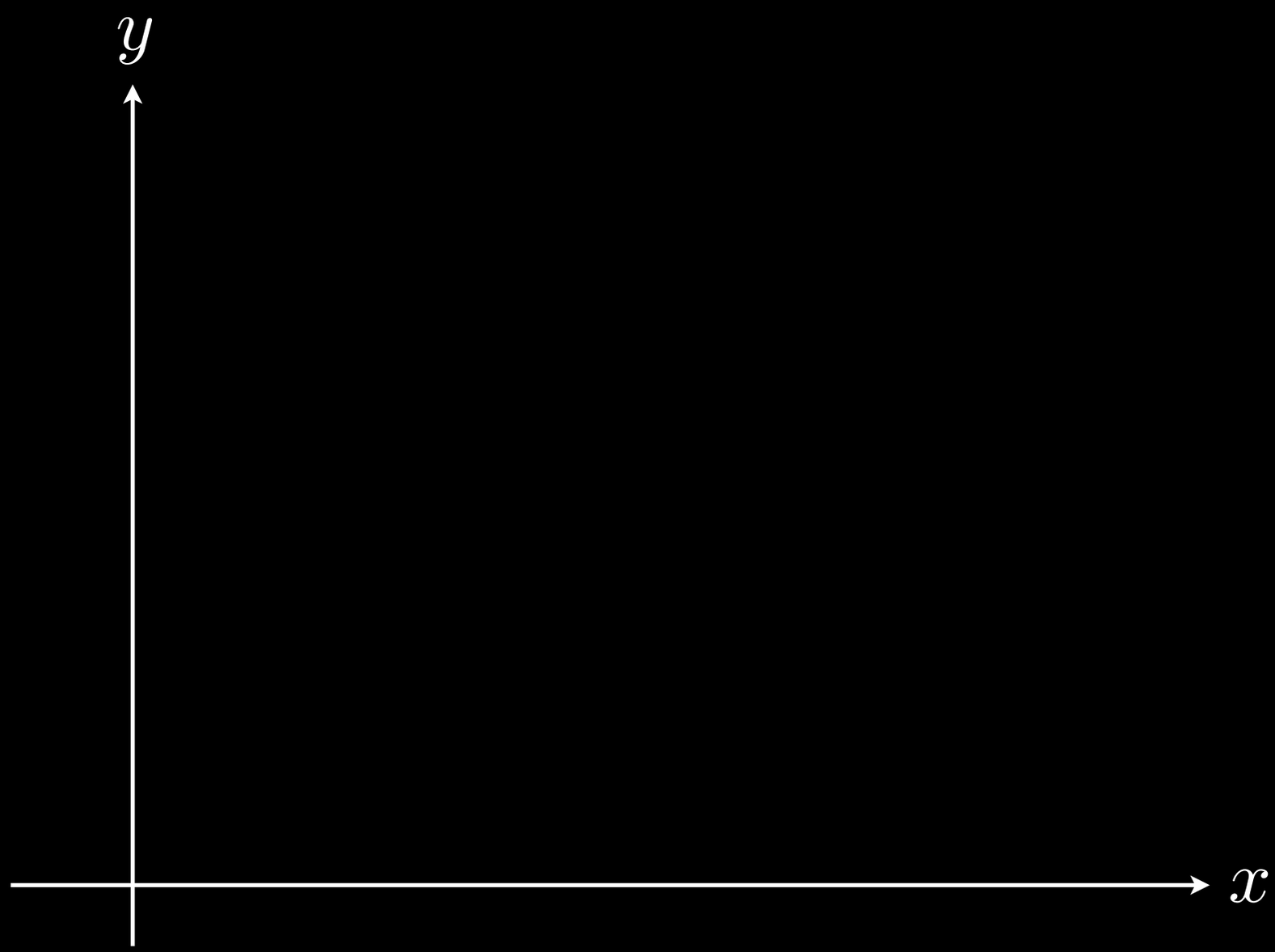
What model(s) represent the set of features best?

How many model instances are there?

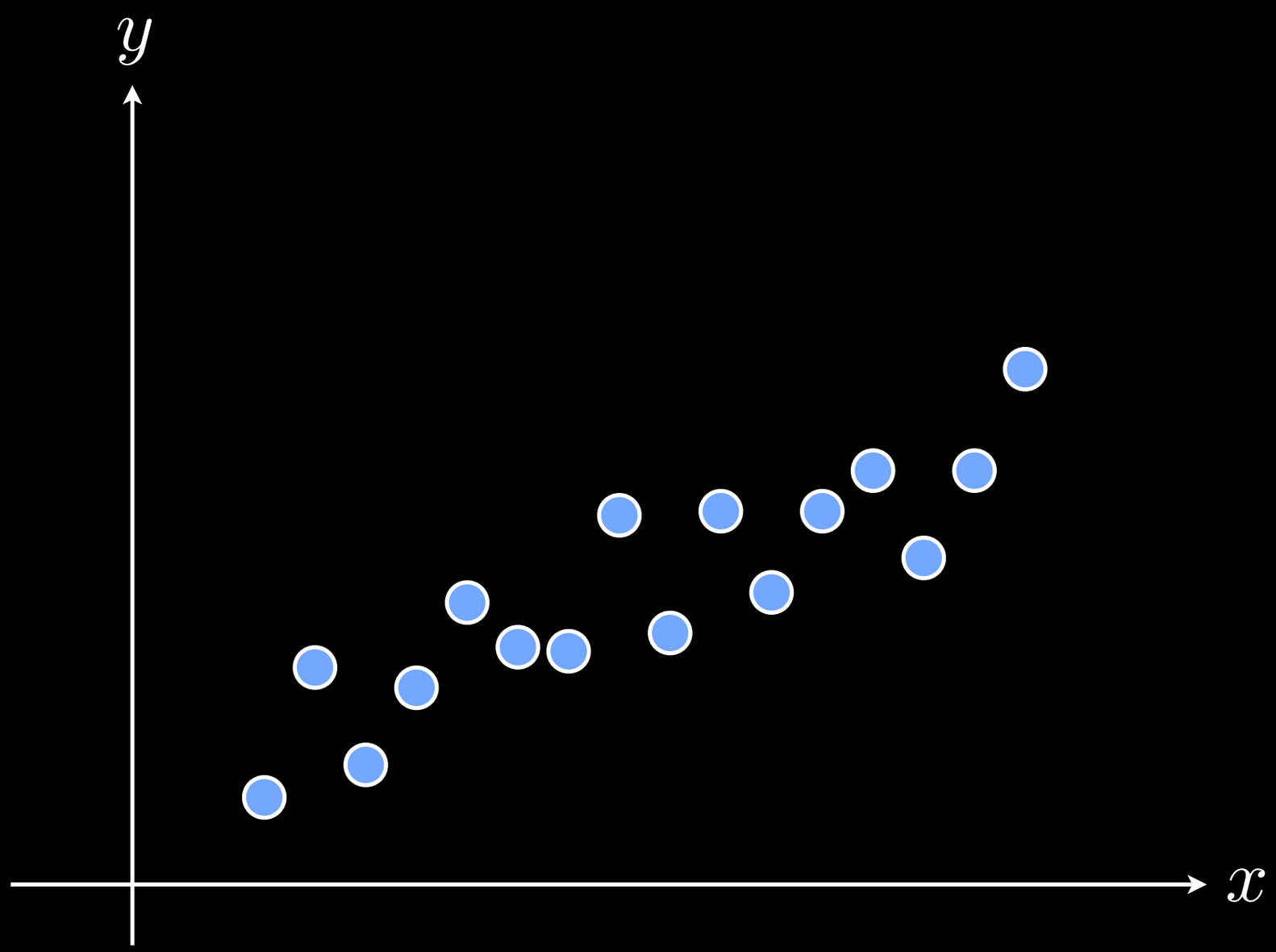
Which of several model instances gets which feature?

Challenges

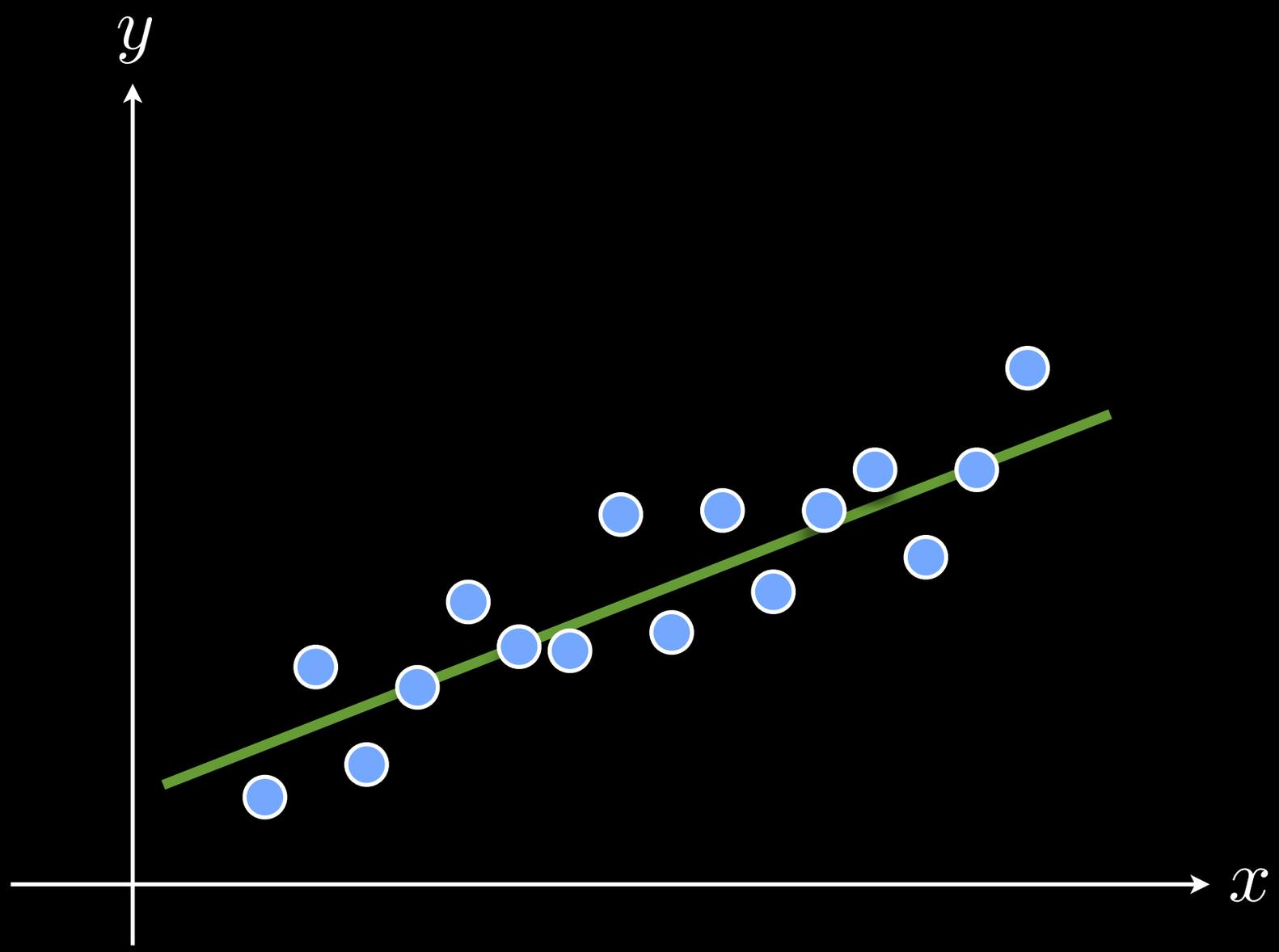
Noisy data



Noisy data



Noisy data



RYERSON UNIVERSITY

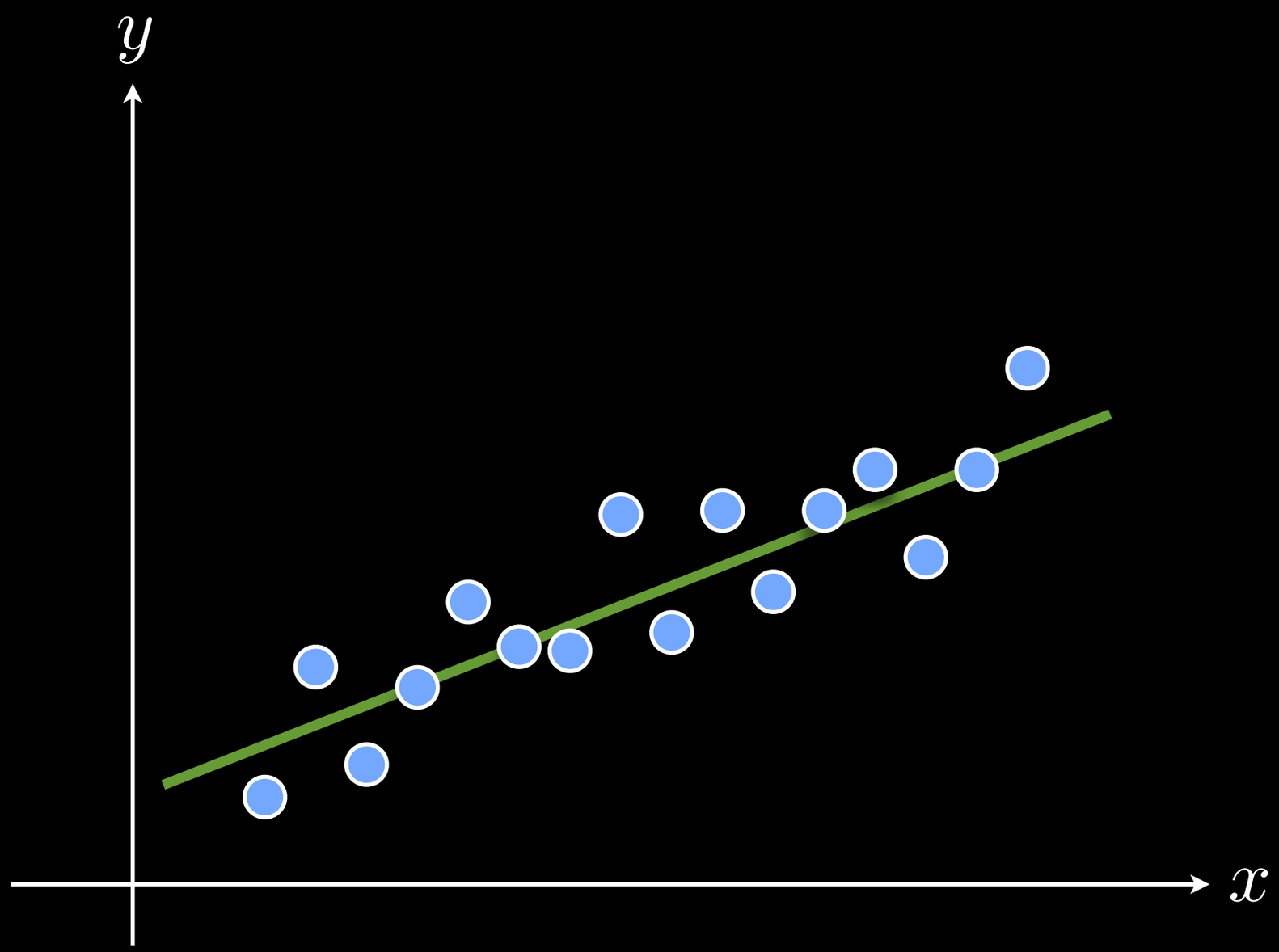
MAPLE LEAF GARDENS

RYERSON UNIVERSITY

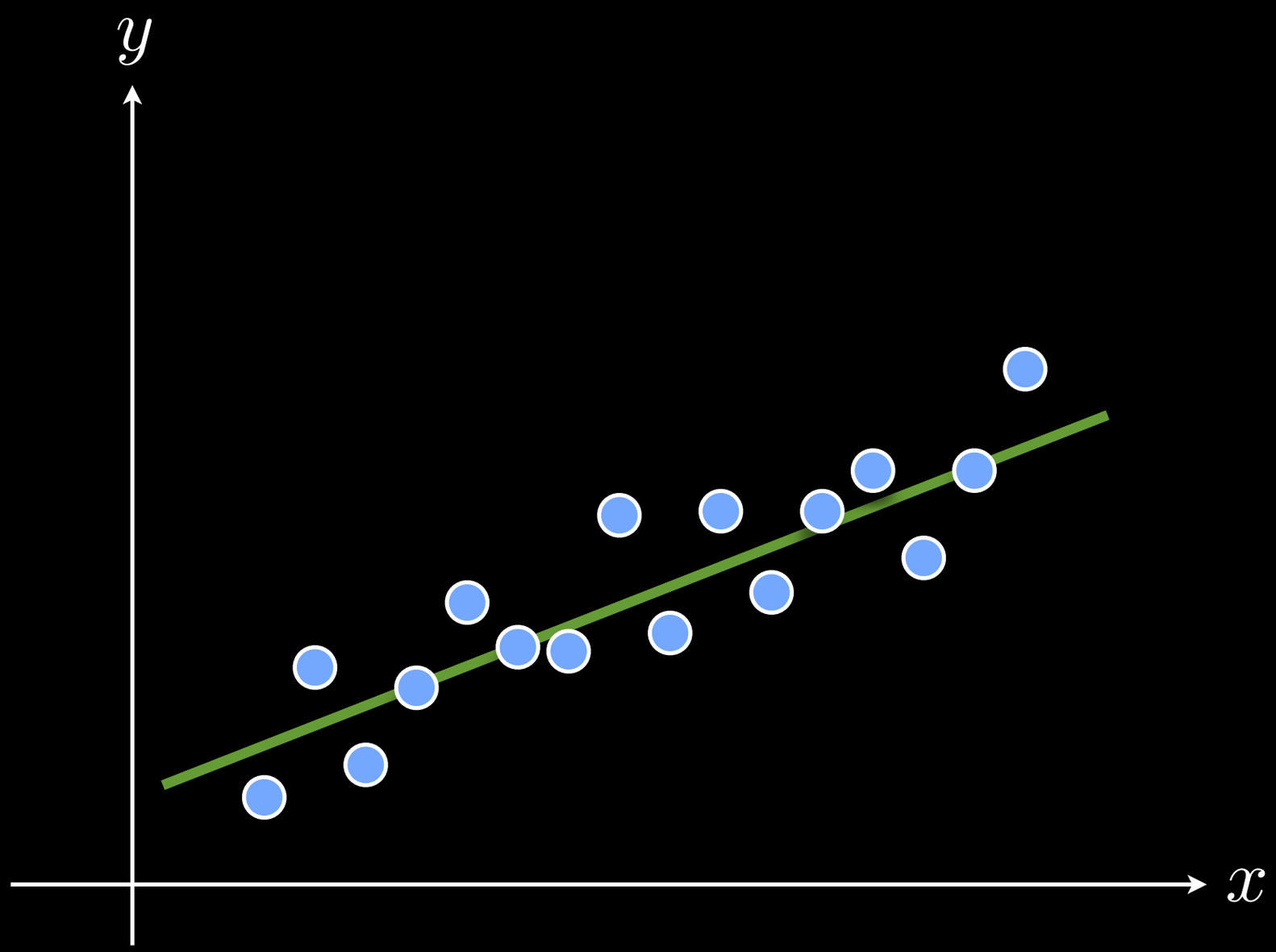

MAPLE LEAF GARDENS
RYERSON
UNIVERSITY


MAPLE LEAF GARDENS
RYERSON
UNIVERSITY

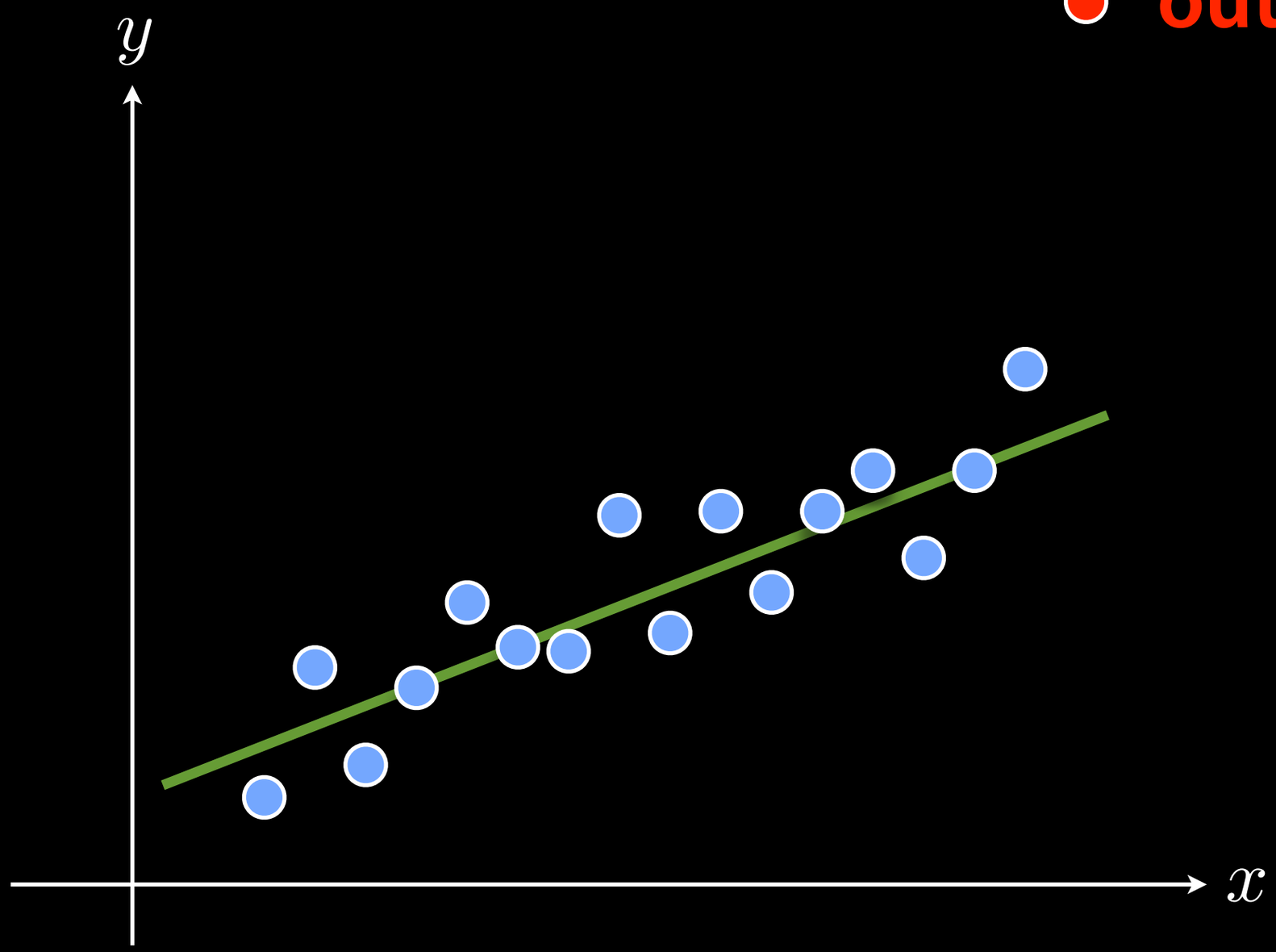
Noisy data



Outliers in data



Outliers in data



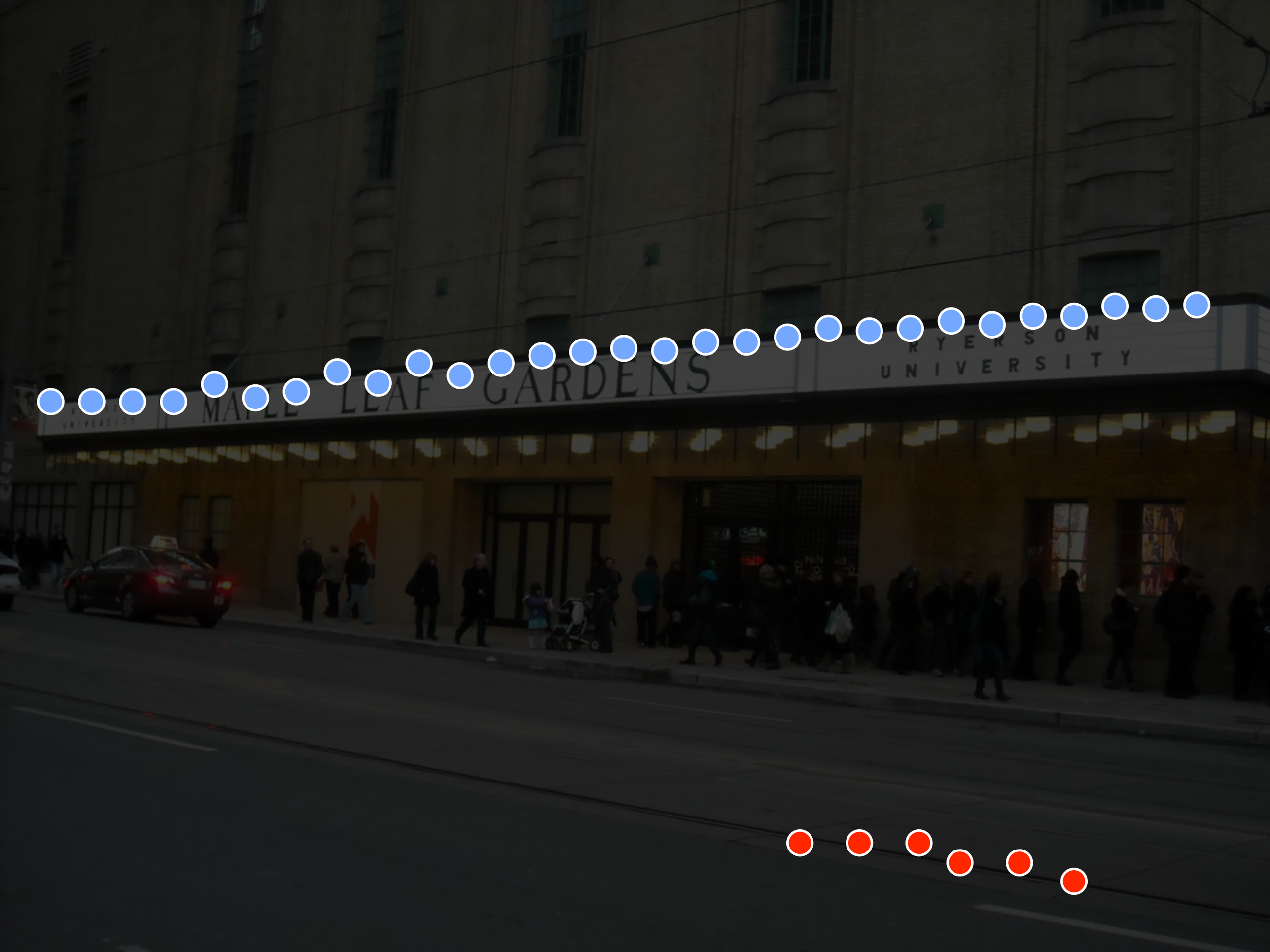
outlier

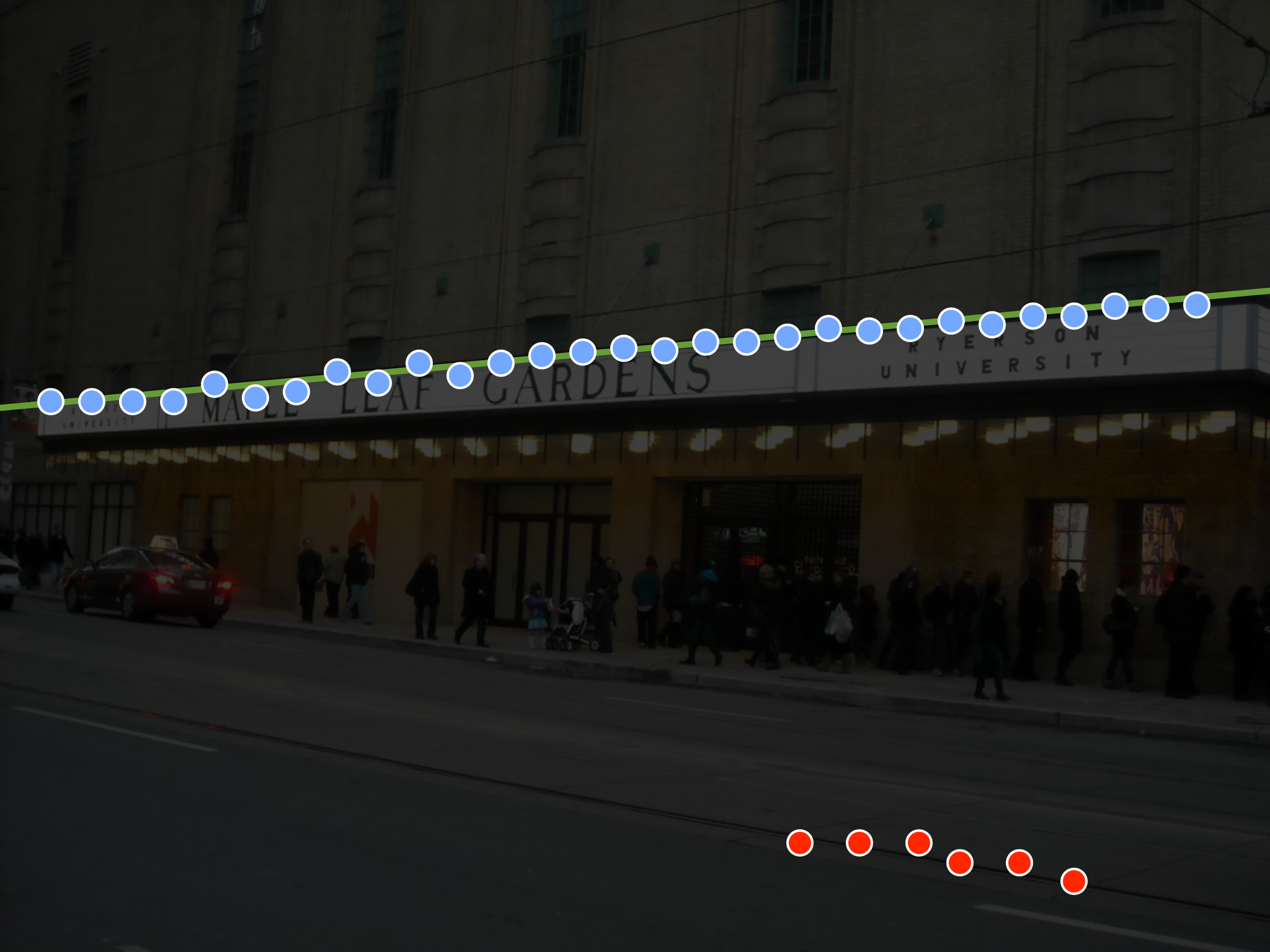
RYERSON UNIVERSITY

MAPLE LEAF GARDENS

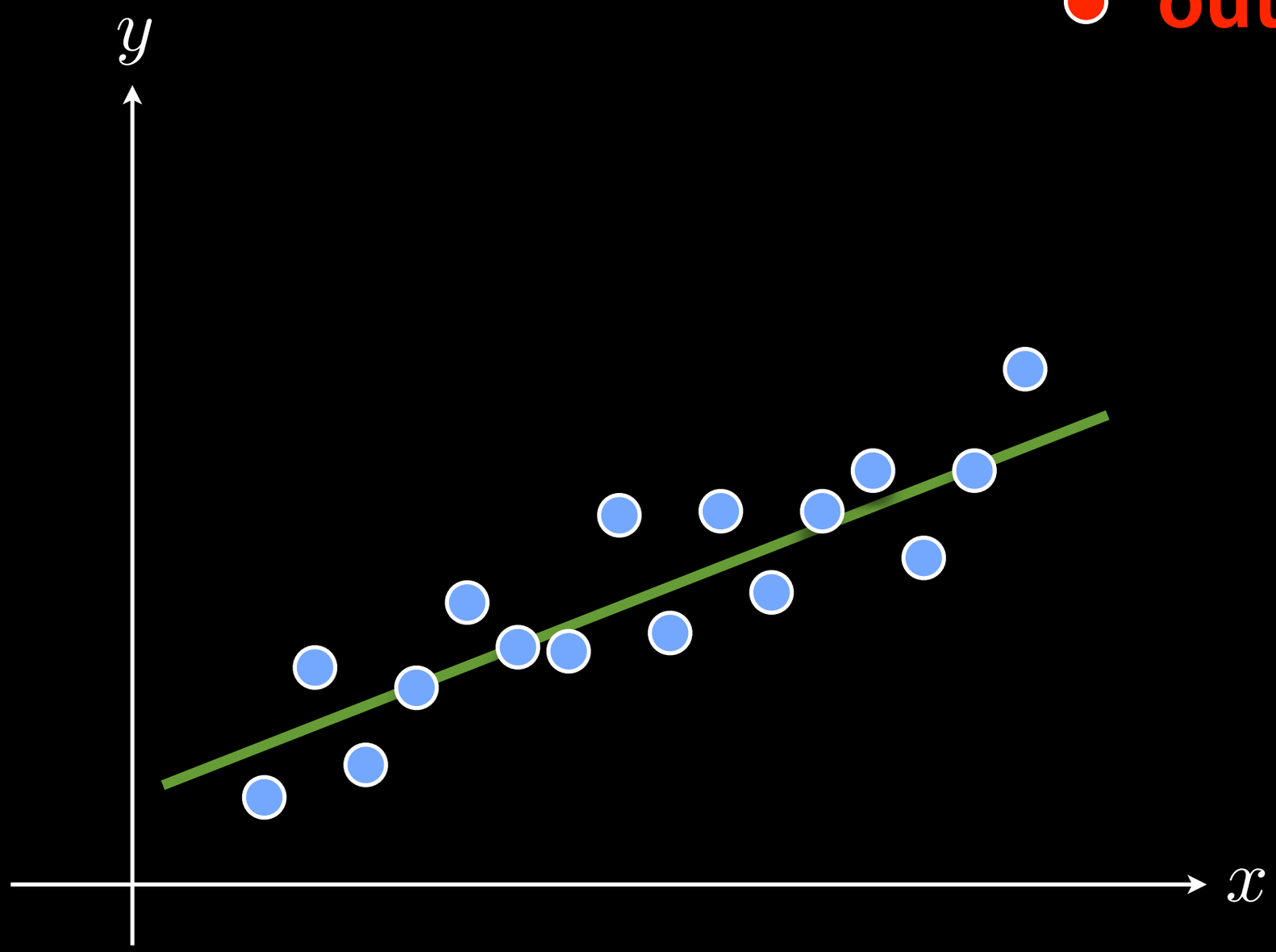
RYERSON UNIVERSITY


MAPLE LEAF GARDENS
RYERSON
UNIVERSITY


MAPLE LEAF GARDENS
RYERSON
UNIVERSITY

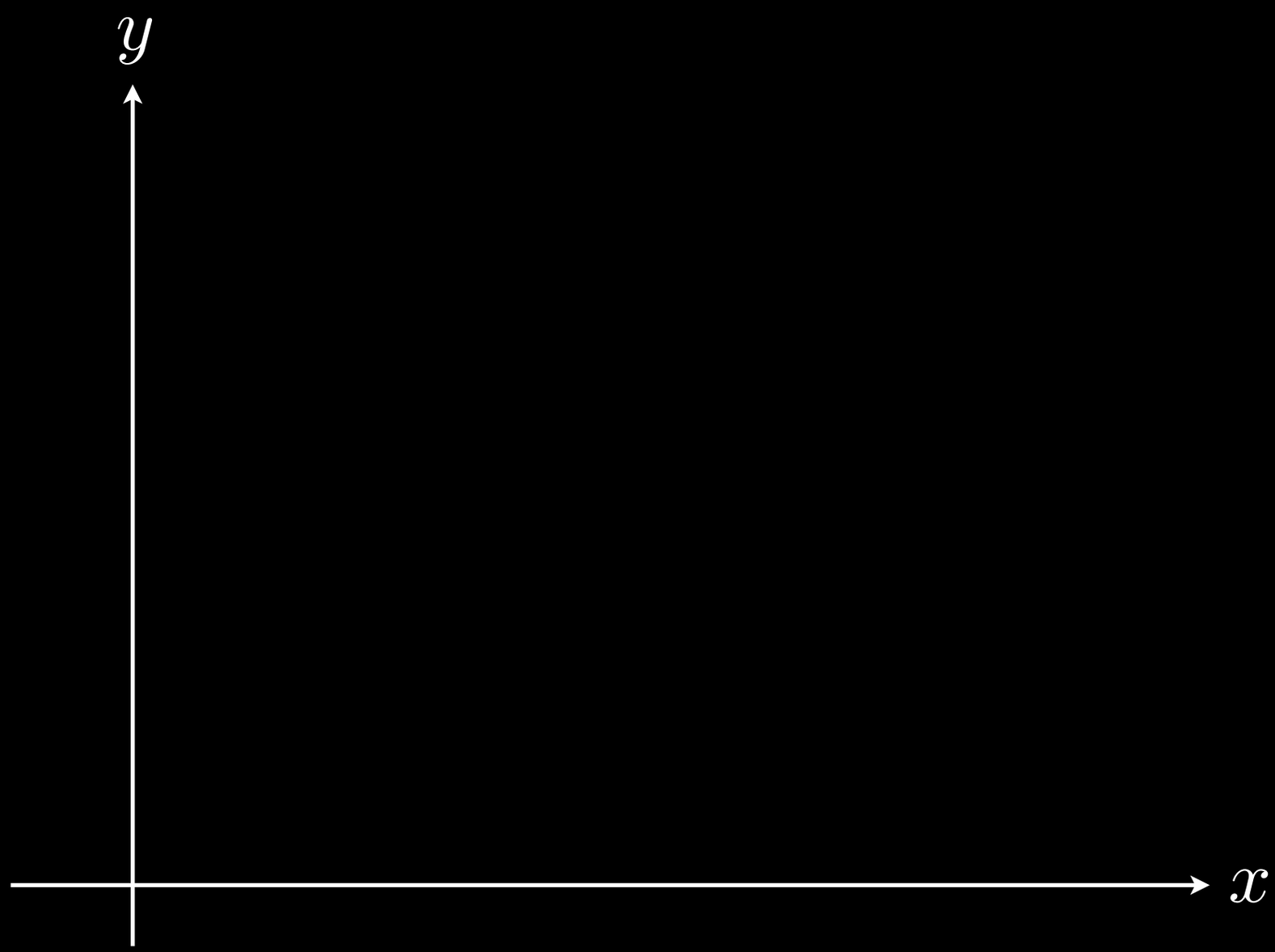

MAPLE LEAF GARDENS
RYERSON
UNIVERSITY

Outliers in data

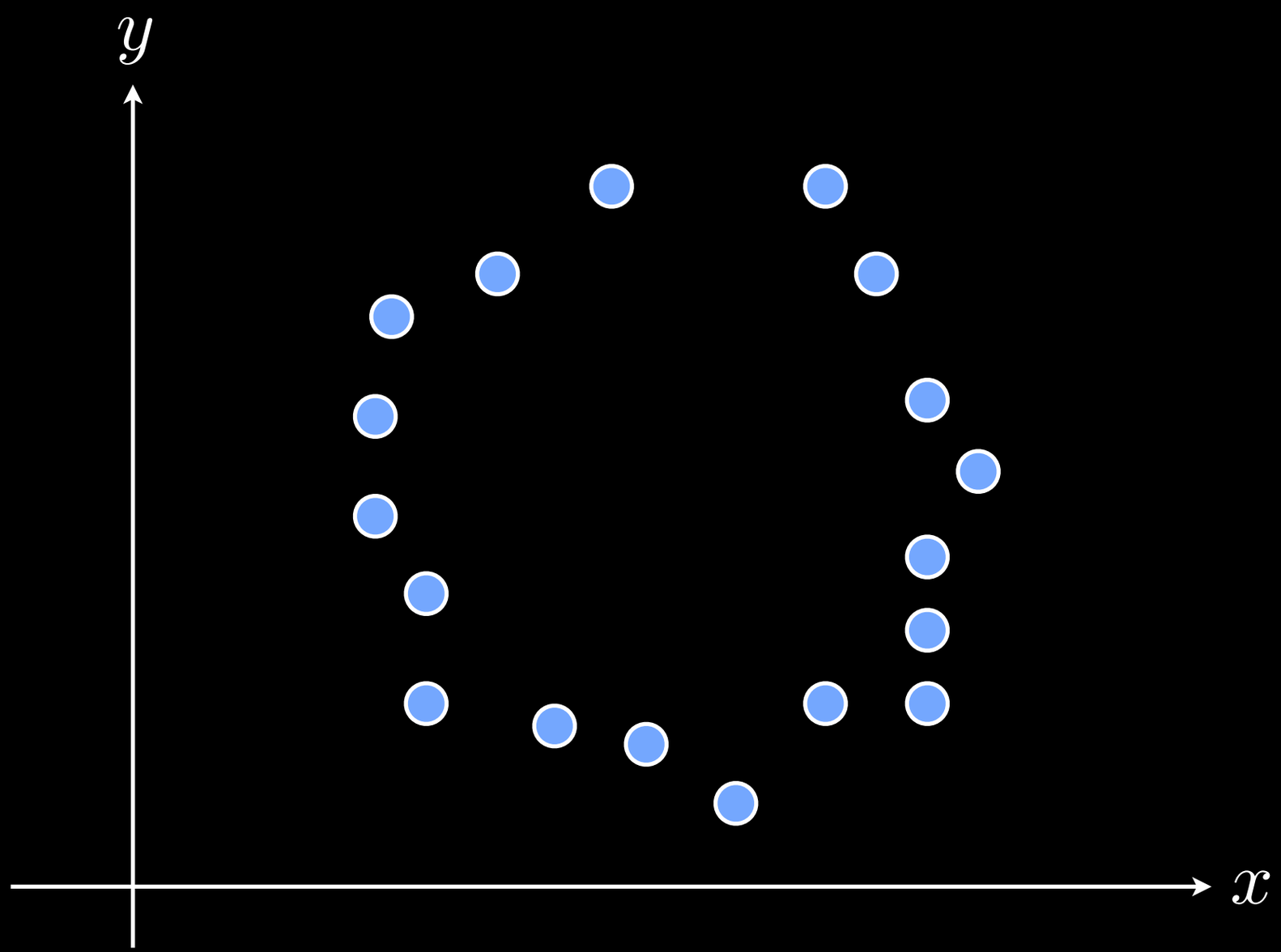


outlier

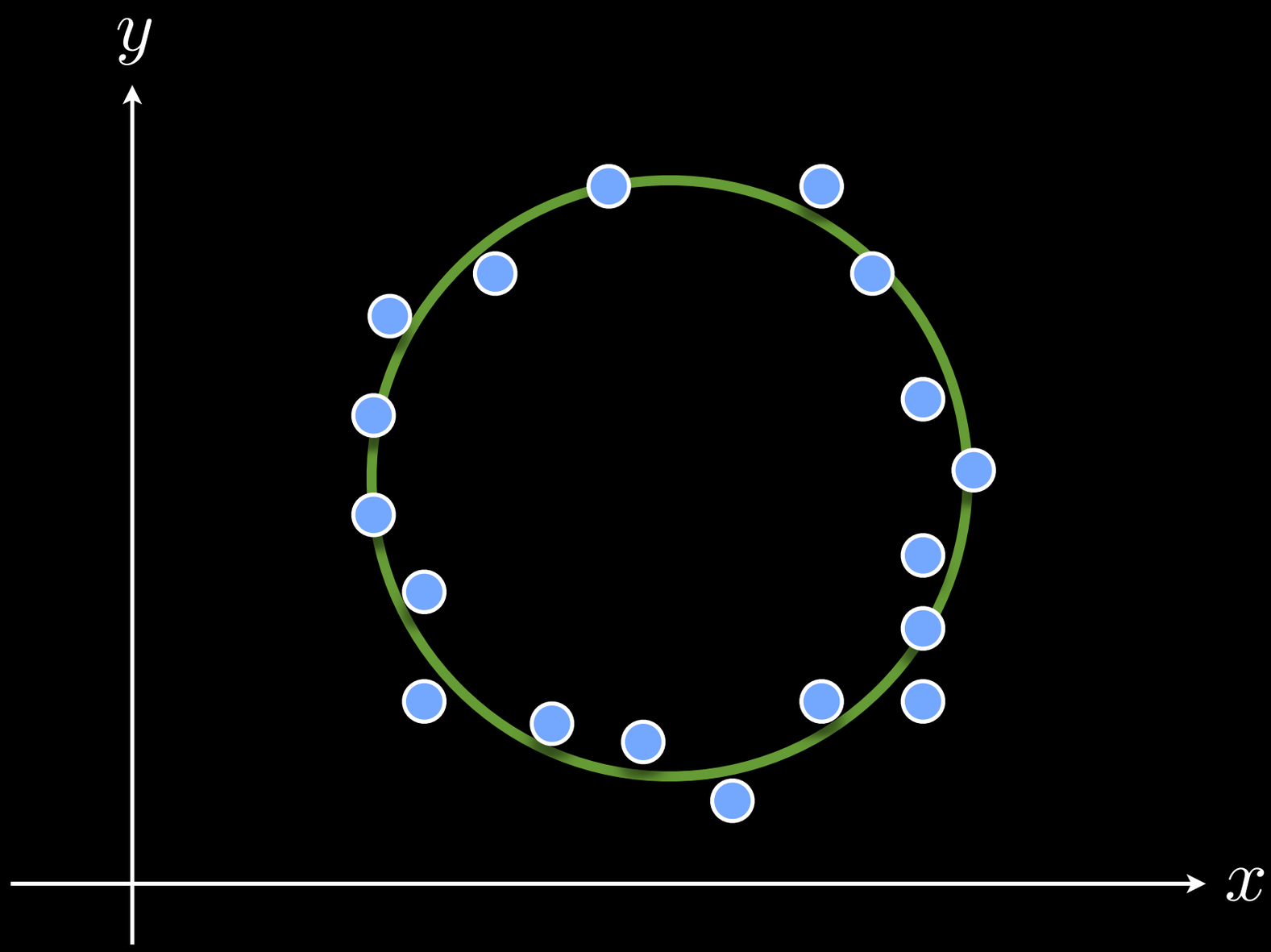
Missing data



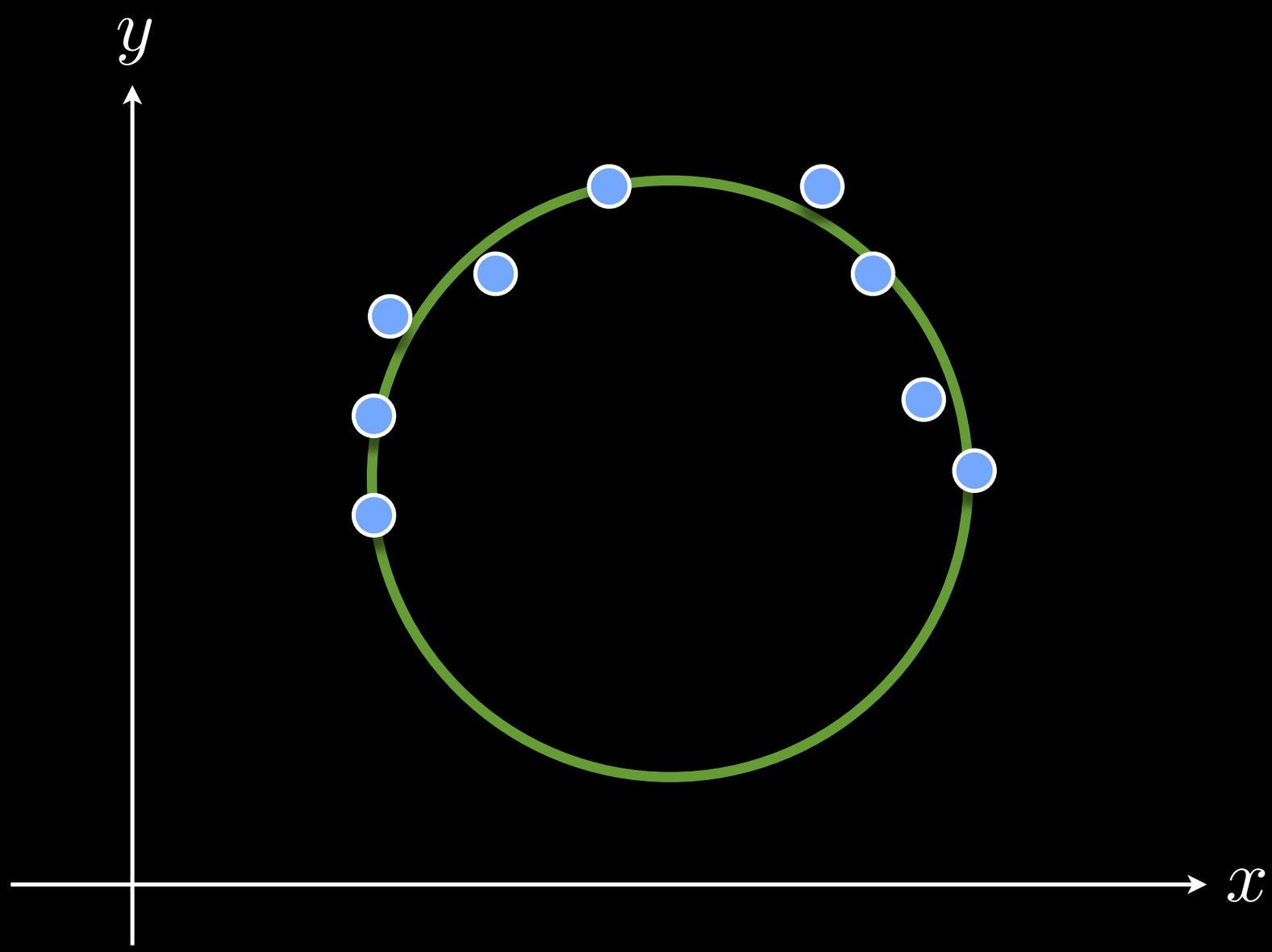
Missing data



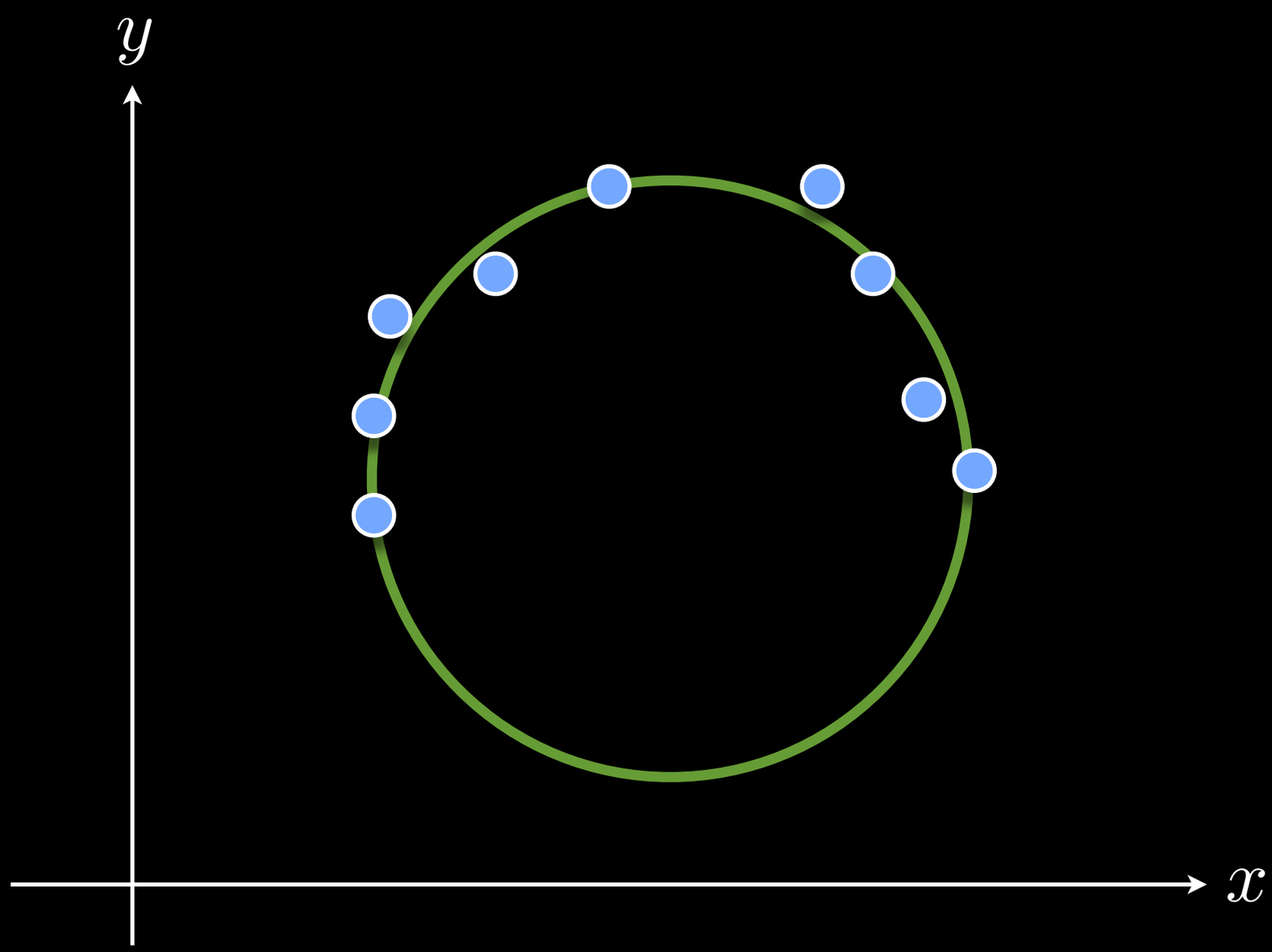
Missing data



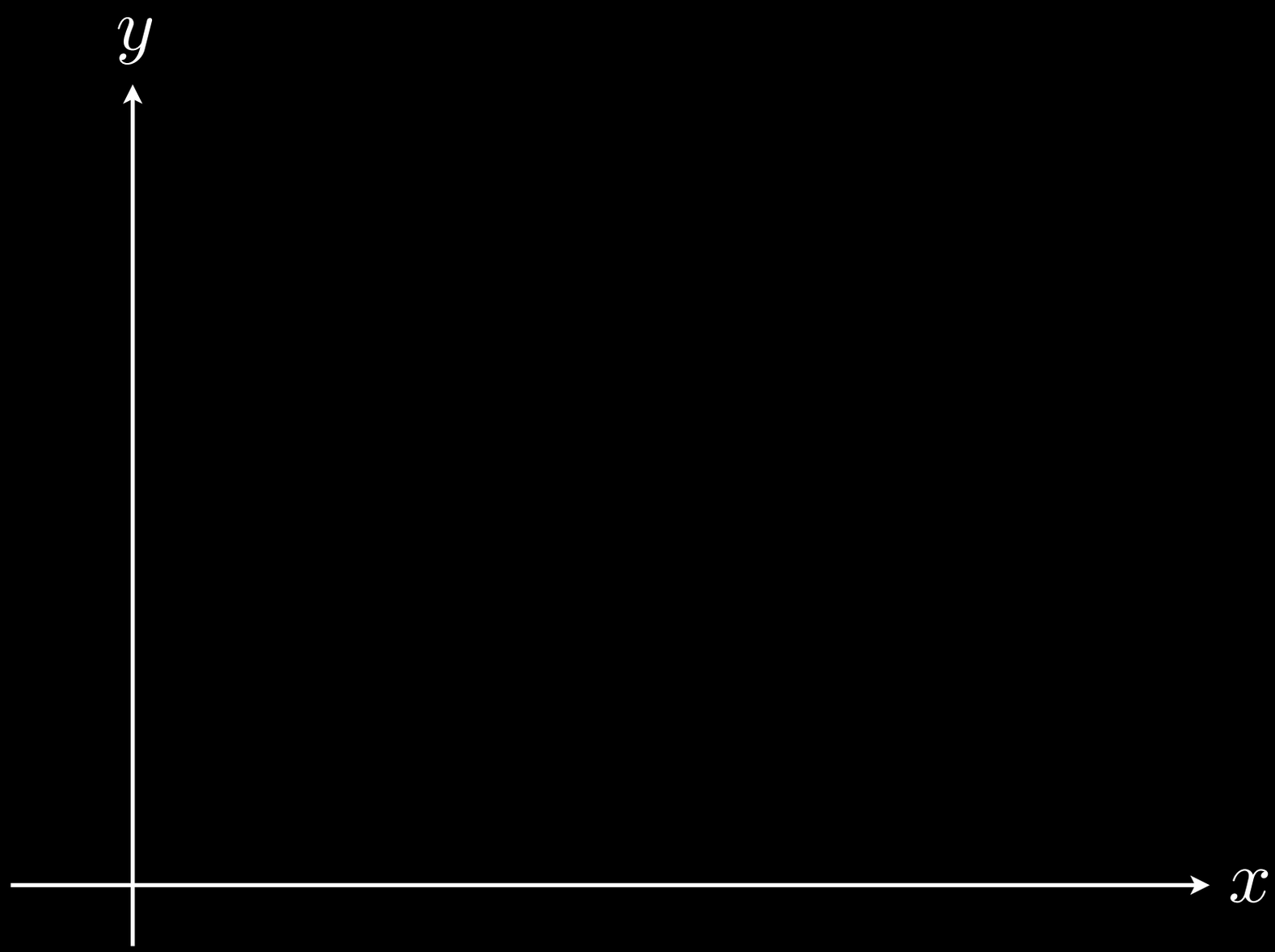
Missing data



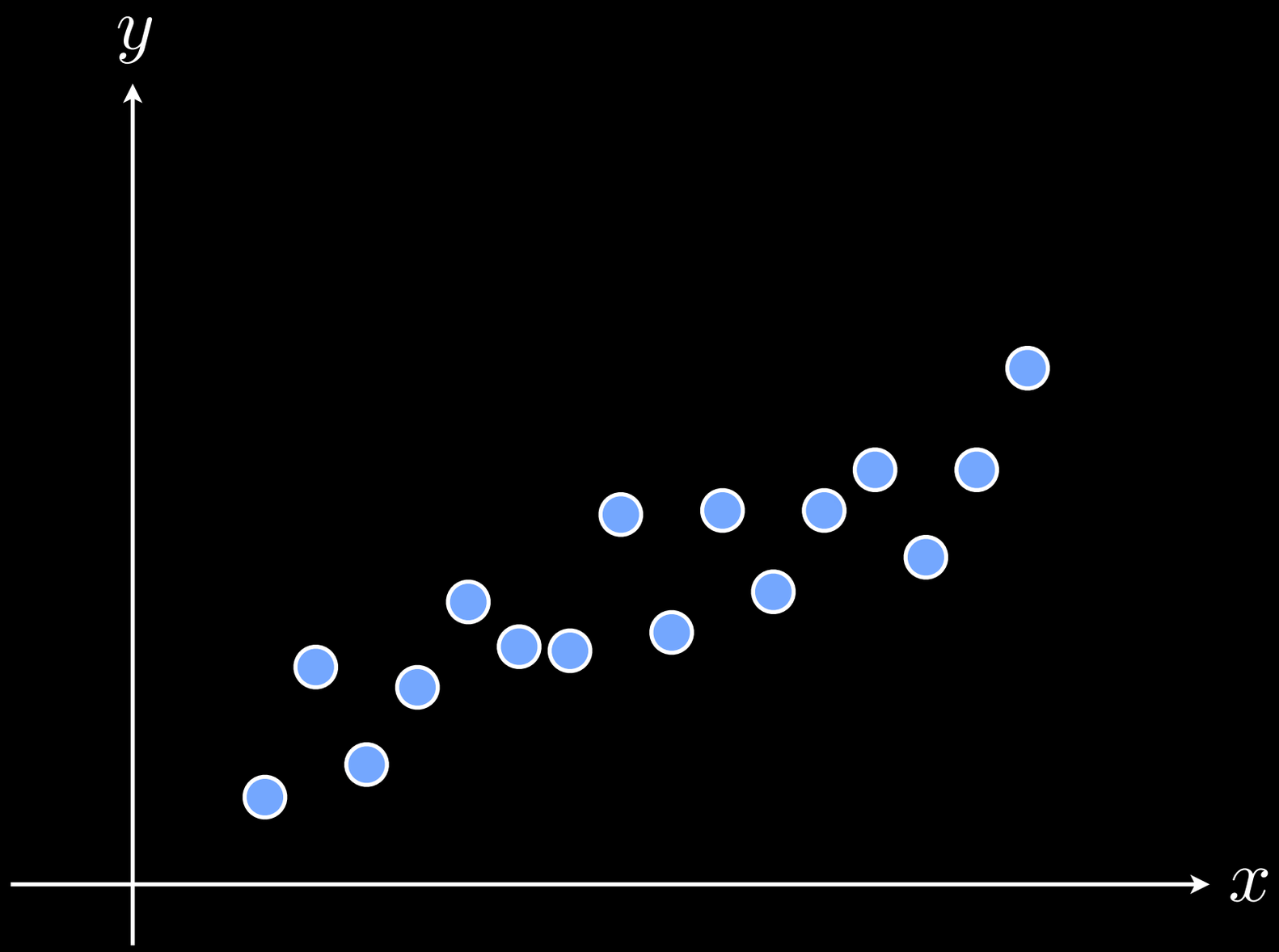
Missing data



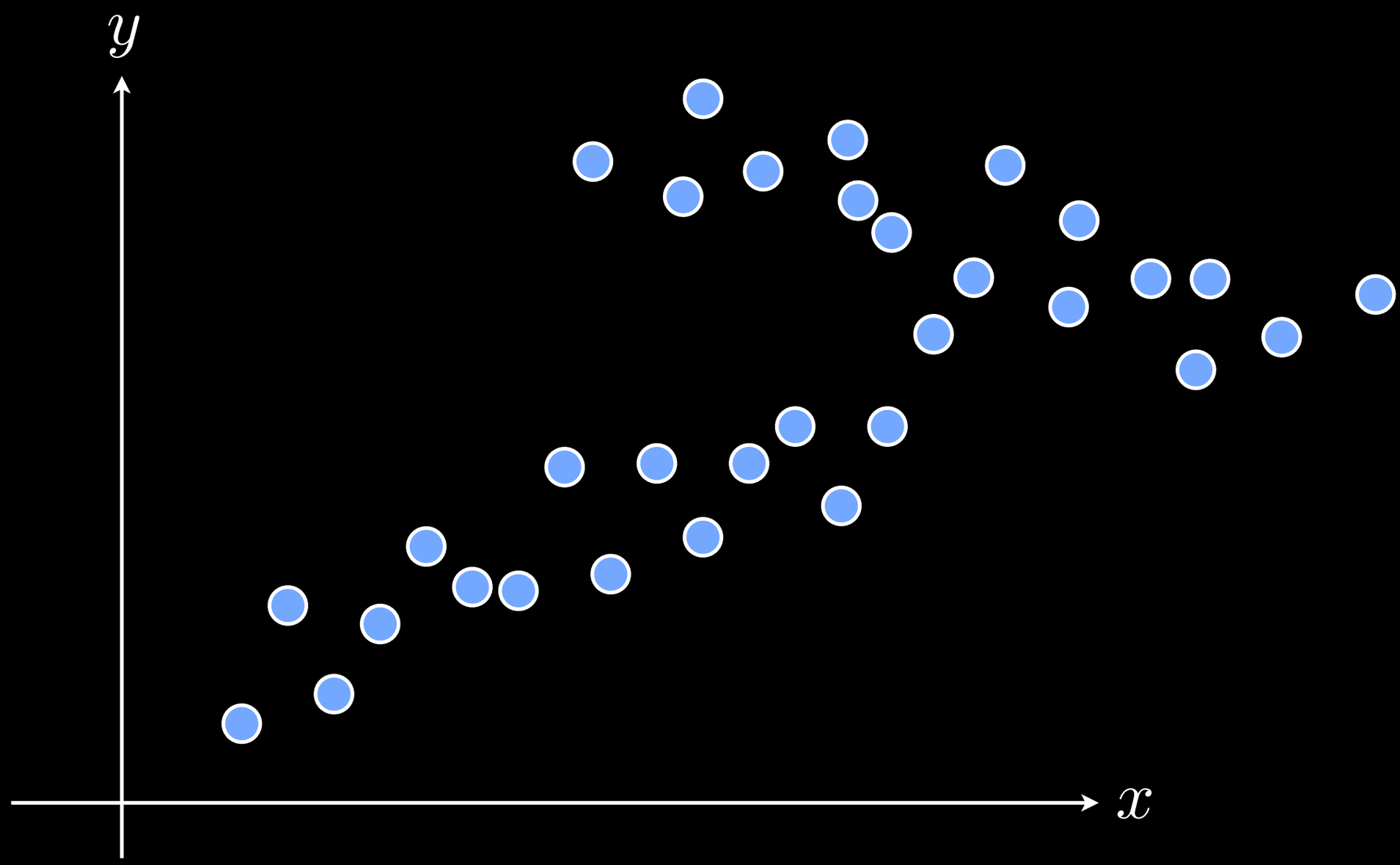
Feature binding



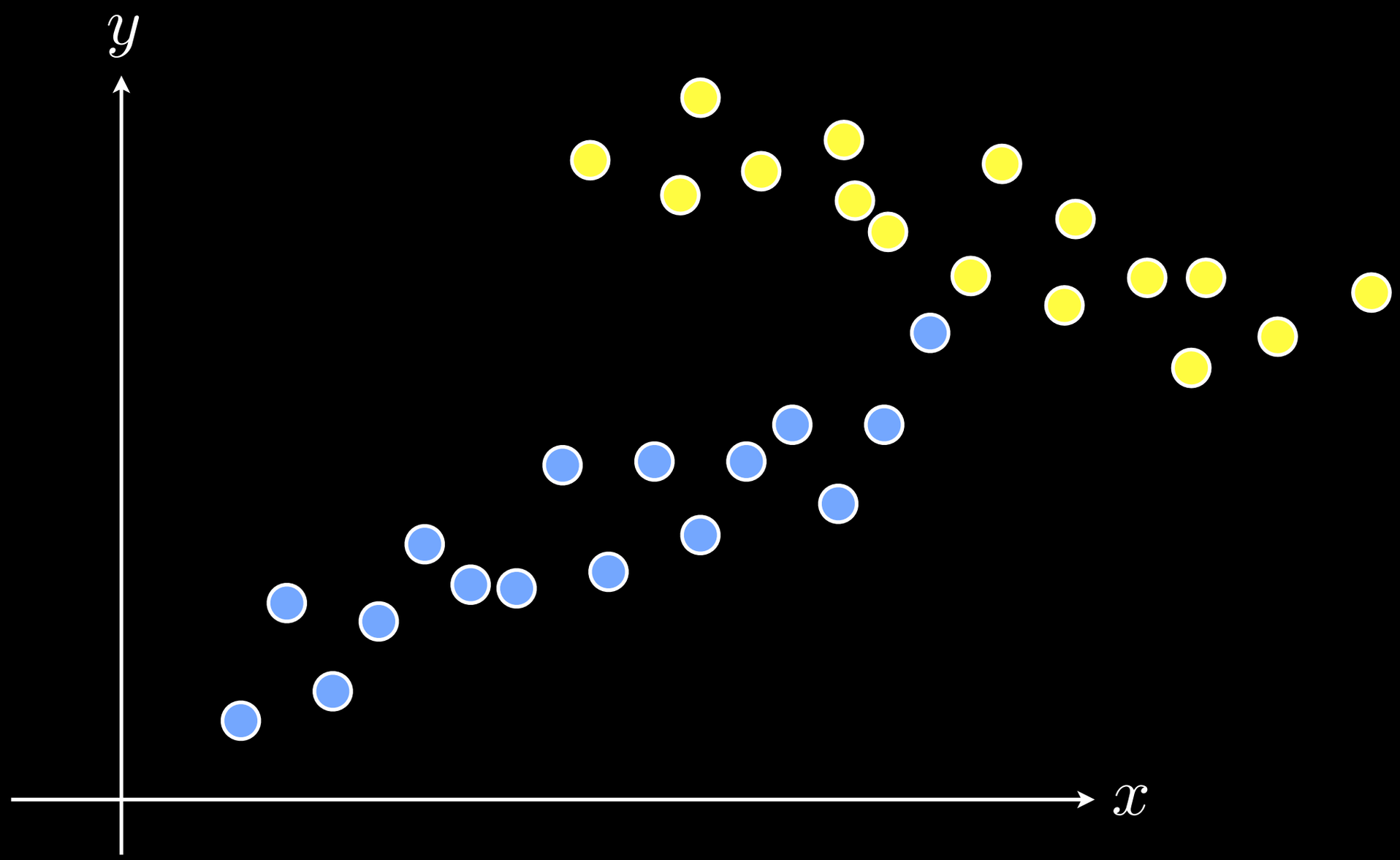
Feature binding



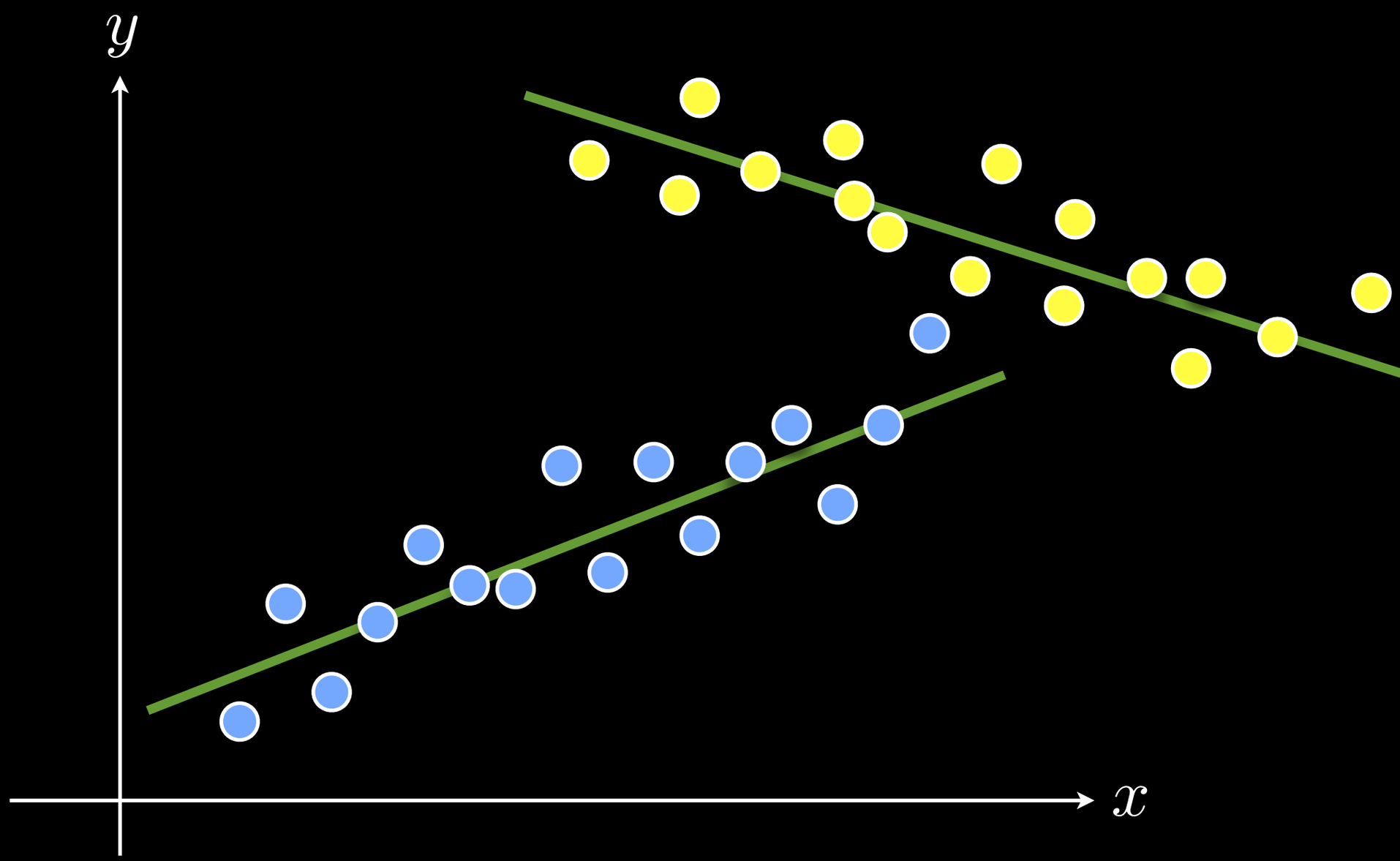
Feature binding



Feature binding

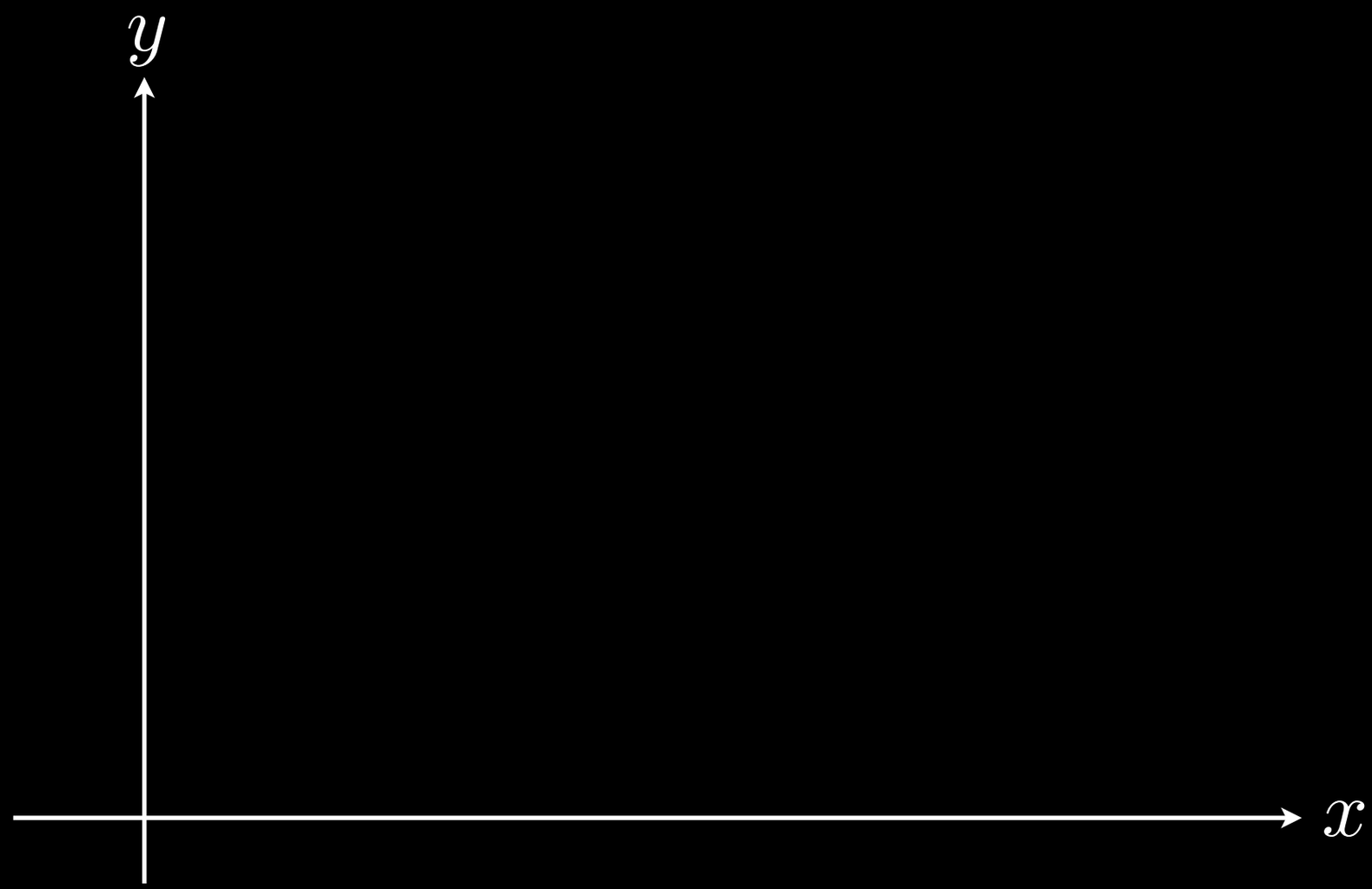


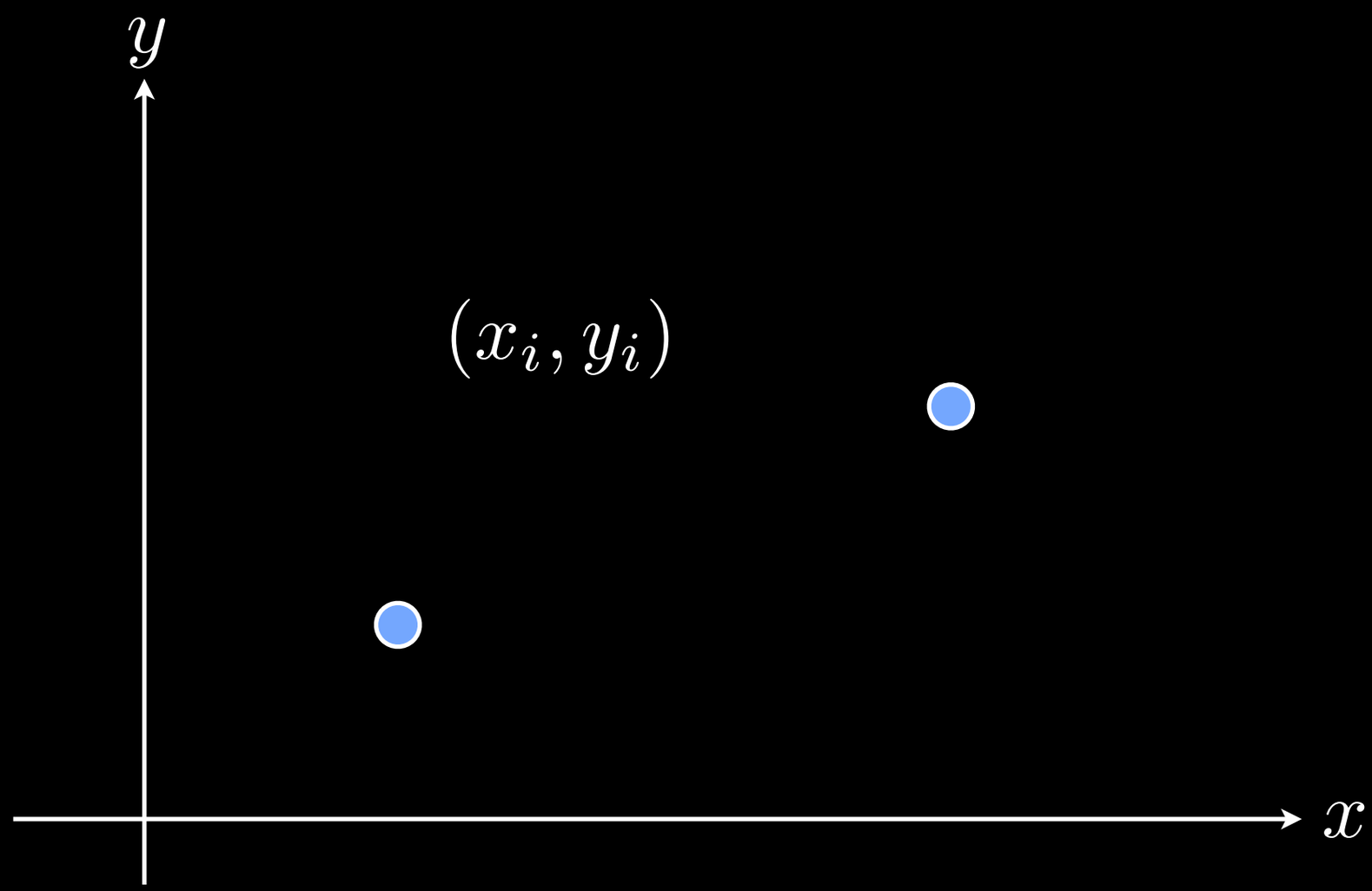
Feature binding

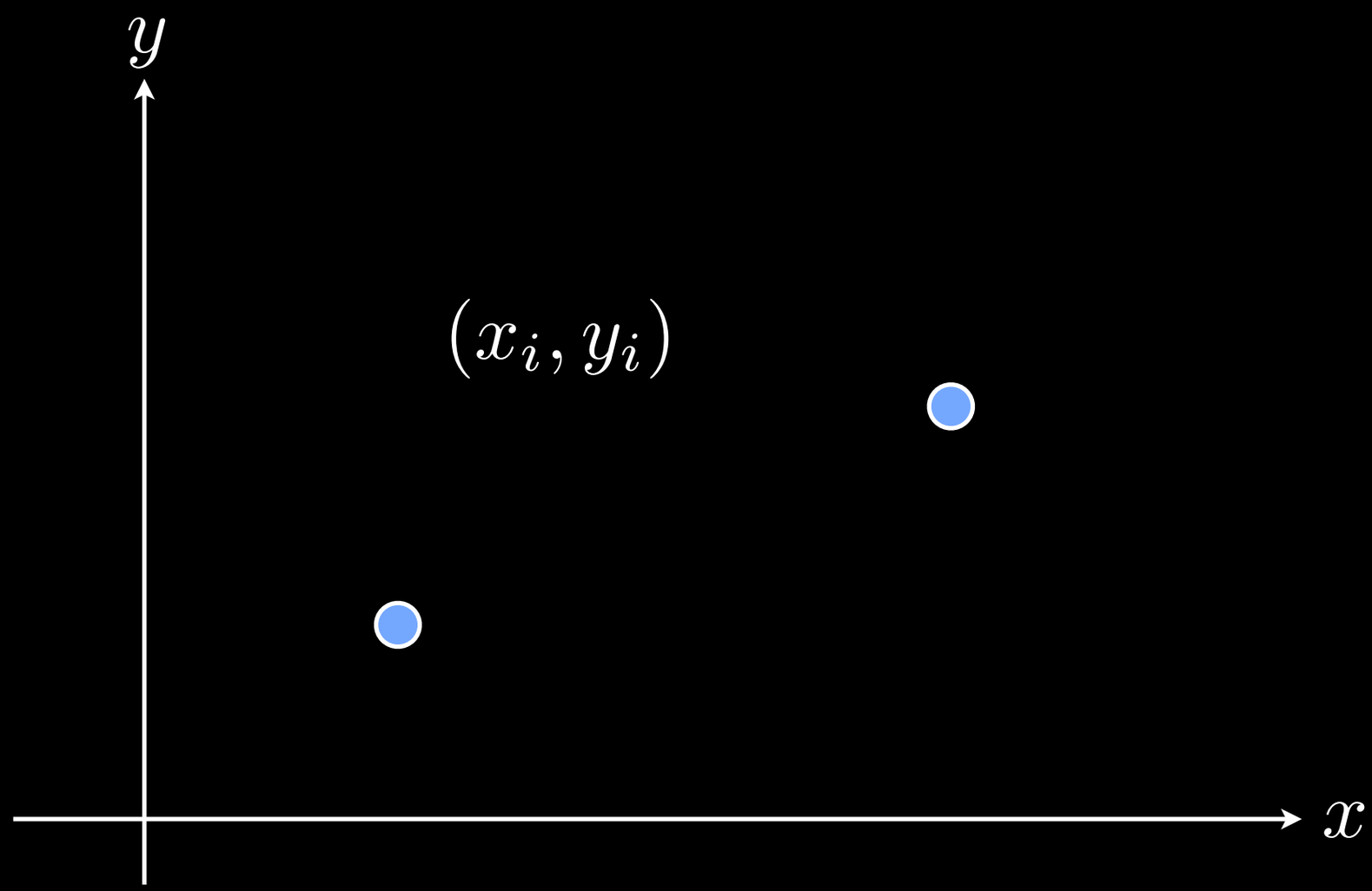


Least Squares

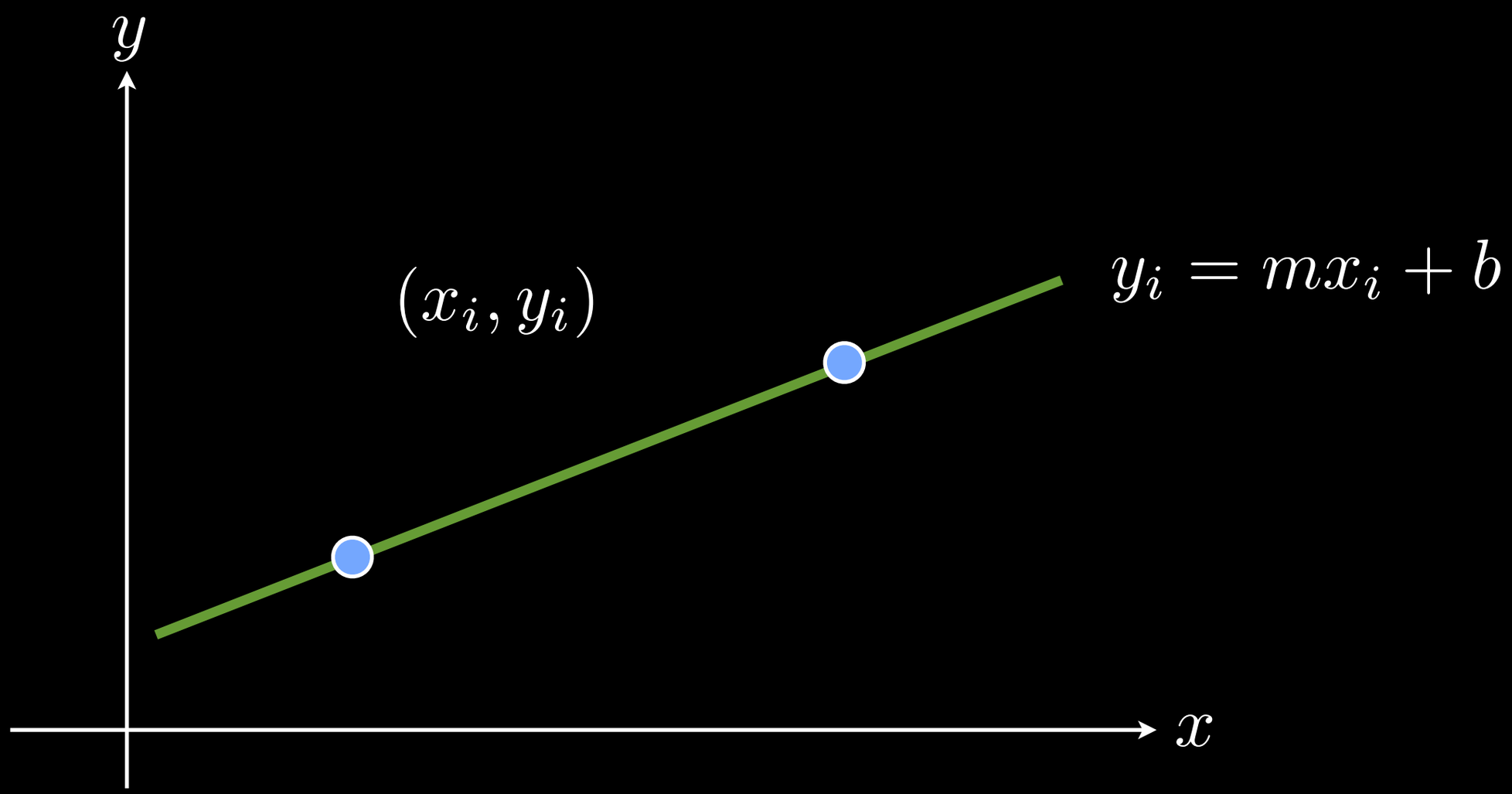
fitting


y
x

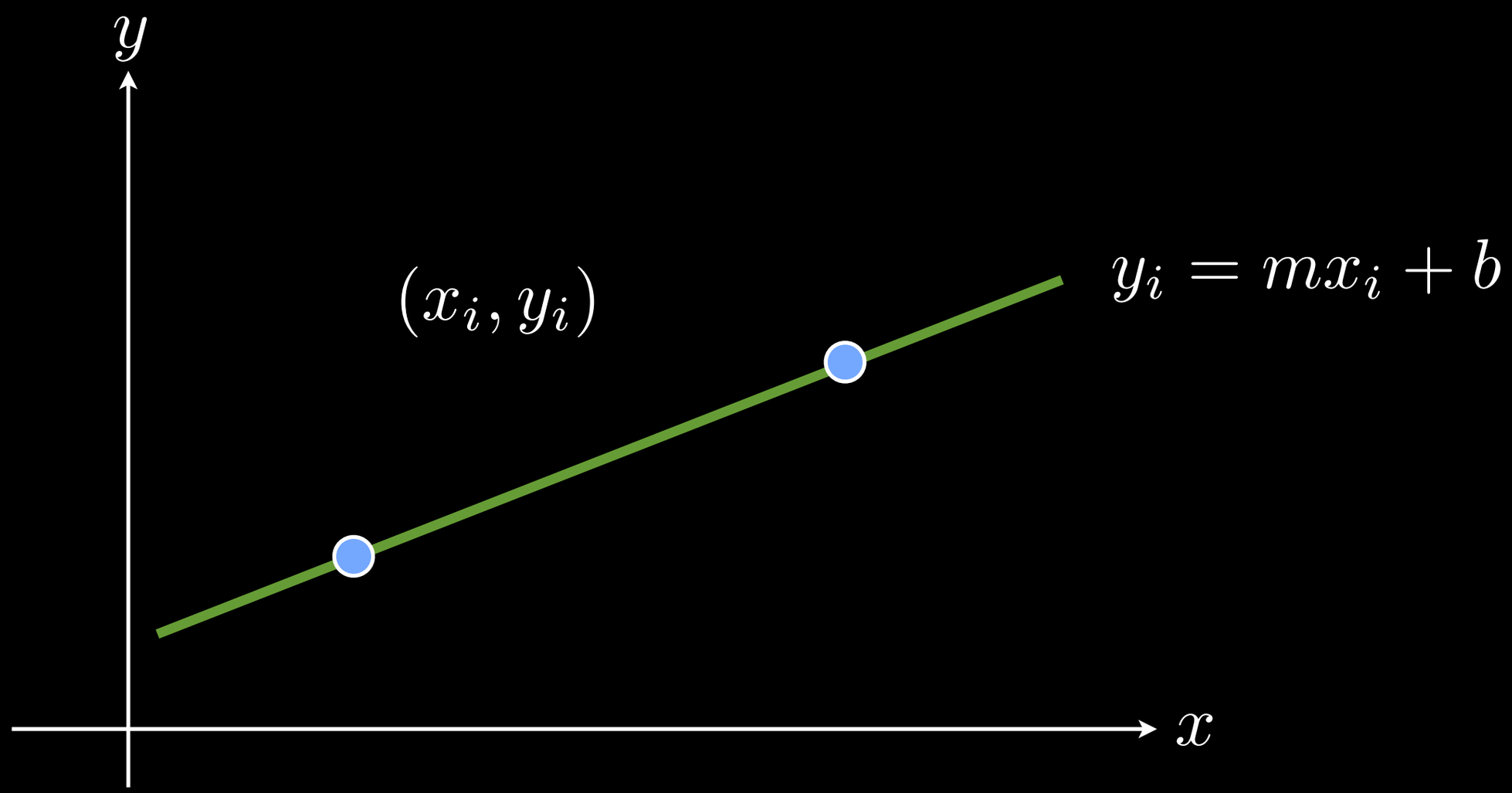

y
(x_i, y_i)
x



How do we find the line?

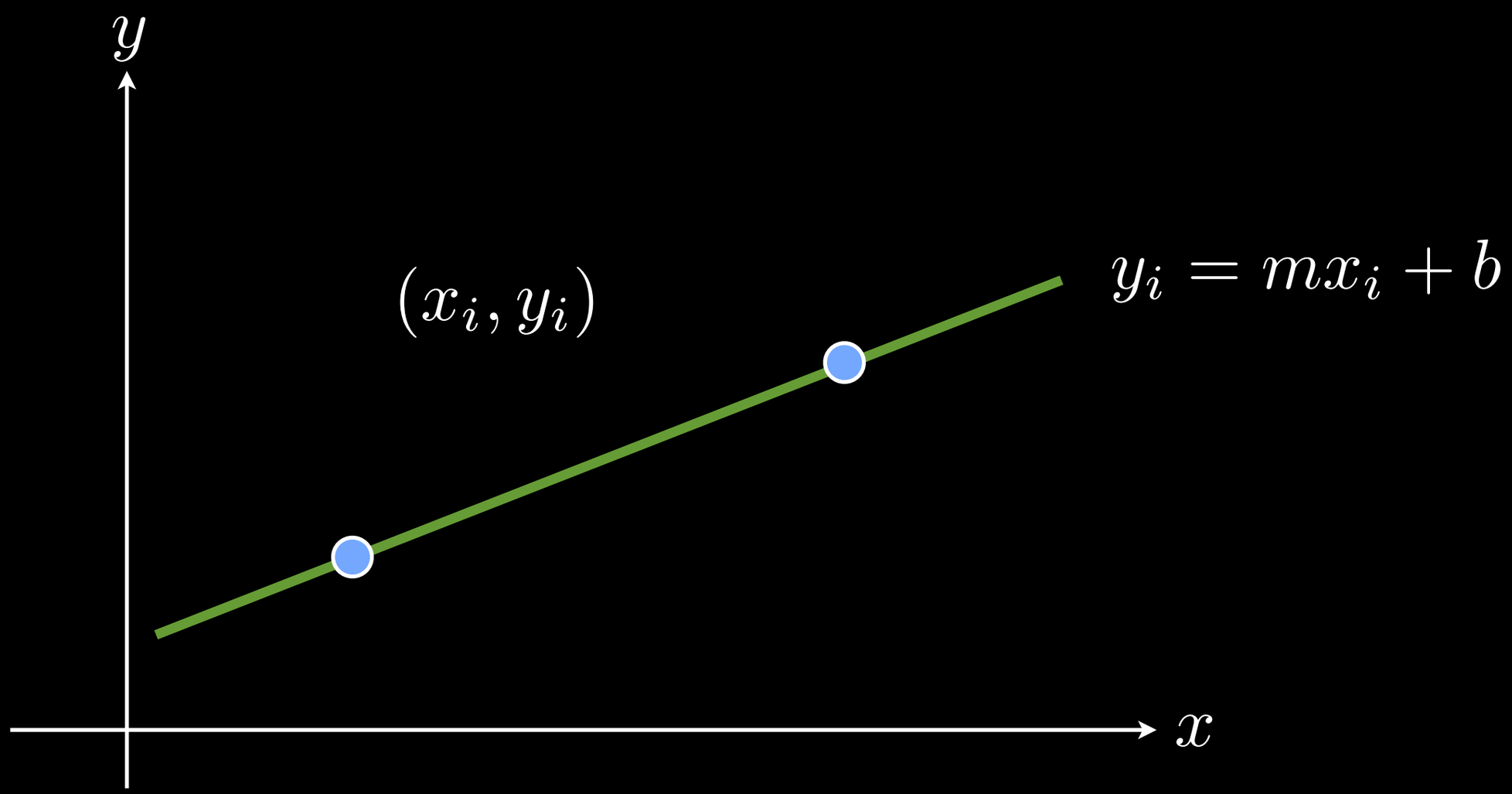


How do we find the line?



How do we find the line?

$$y_0 = mx_0 + b$$
$$y_1 = mx_1 + b$$



How do we find the line?

$$\begin{pmatrix} x_0 & 1 \\ x_1 & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix}$$

How do we find the line?

$$\begin{pmatrix} x_0 & 1 \\ x_1 & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix}$$

How do we find the line?

$$\underbrace{\begin{pmatrix} x_0 & 1 \\ x_1 & 1 \end{pmatrix}}_{\mathbf{A}} \begin{pmatrix} m \\ b \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix}$$

How do we find the line?

$$\underbrace{\begin{pmatrix} x_0 & 1 \\ x_1 & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} m \\ b \end{pmatrix}}_{\mathbf{p}} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix}$$

How do we find the line?

$$\underbrace{\begin{pmatrix} x_0 & 1 \\ x_1 & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} m \\ b \end{pmatrix}}_{\mathbf{p}} = \underbrace{\begin{pmatrix} y_0 \\ y_1 \end{pmatrix}}_{\mathbf{b}}$$

How do we find the line?

$$\begin{pmatrix} x_0 & 1 \\ x_1 & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix}$$

$$\mathbf{A} \qquad \mathbf{p} \qquad \mathbf{b}$$

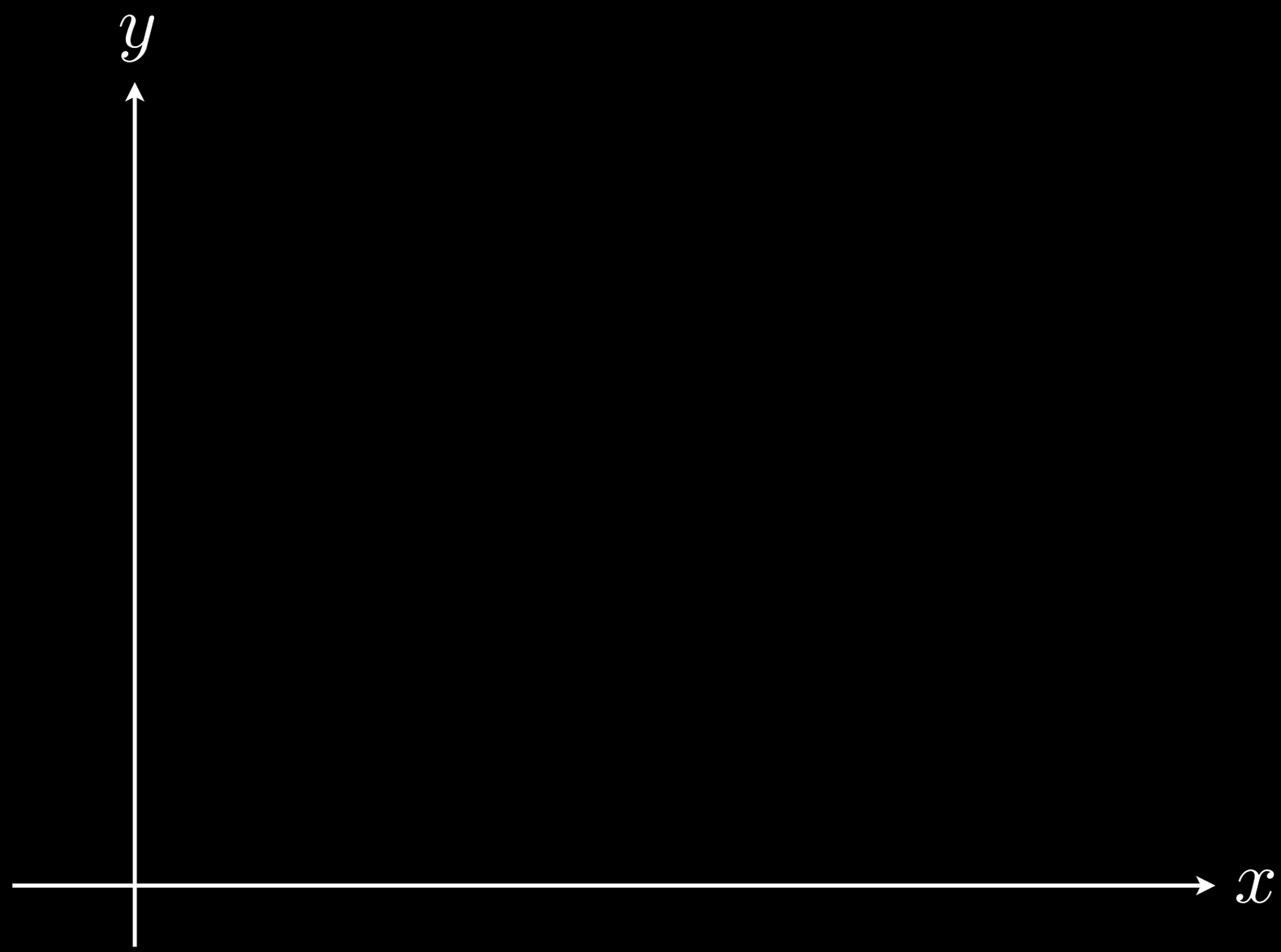
What is the solution for the line parameters?

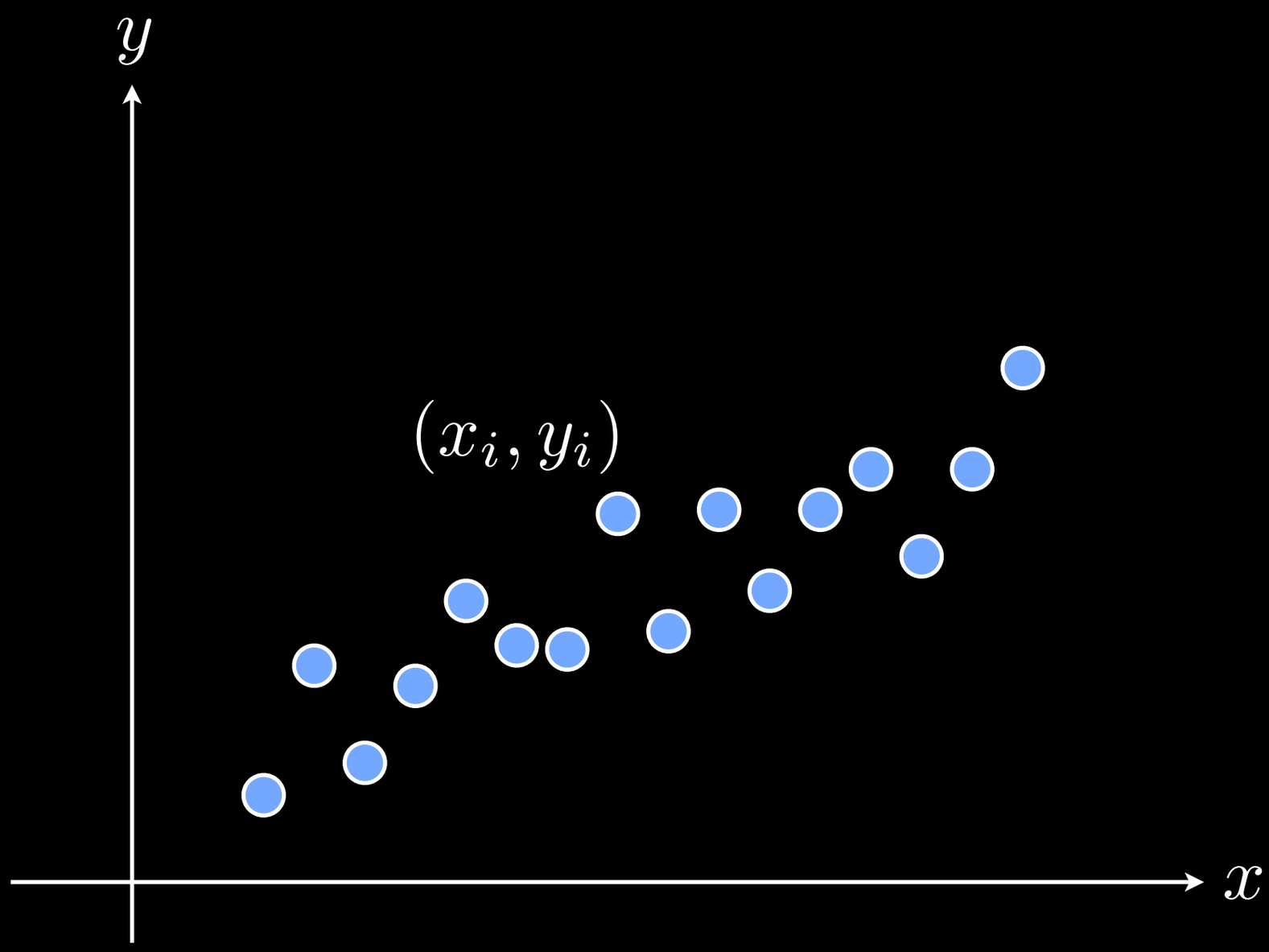
How do we find the line?

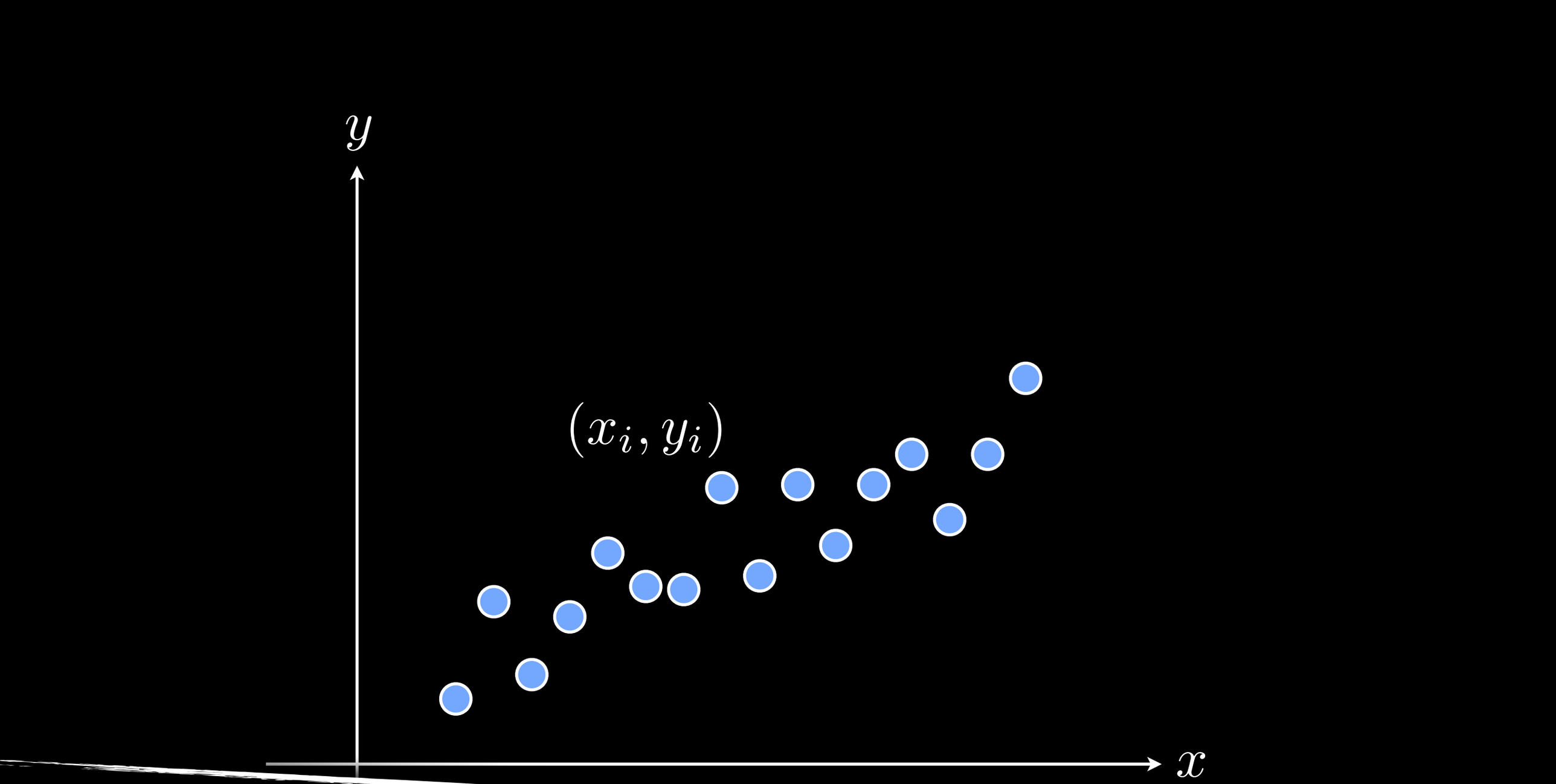
$$\underbrace{\begin{pmatrix} x_0 & 1 \\ x_1 & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} m \\ b \end{pmatrix}}_{\mathbf{p}} = \underbrace{\begin{pmatrix} y_0 \\ y_1 \end{pmatrix}}_{\mathbf{b}}$$

What is the solution for the line parameters?

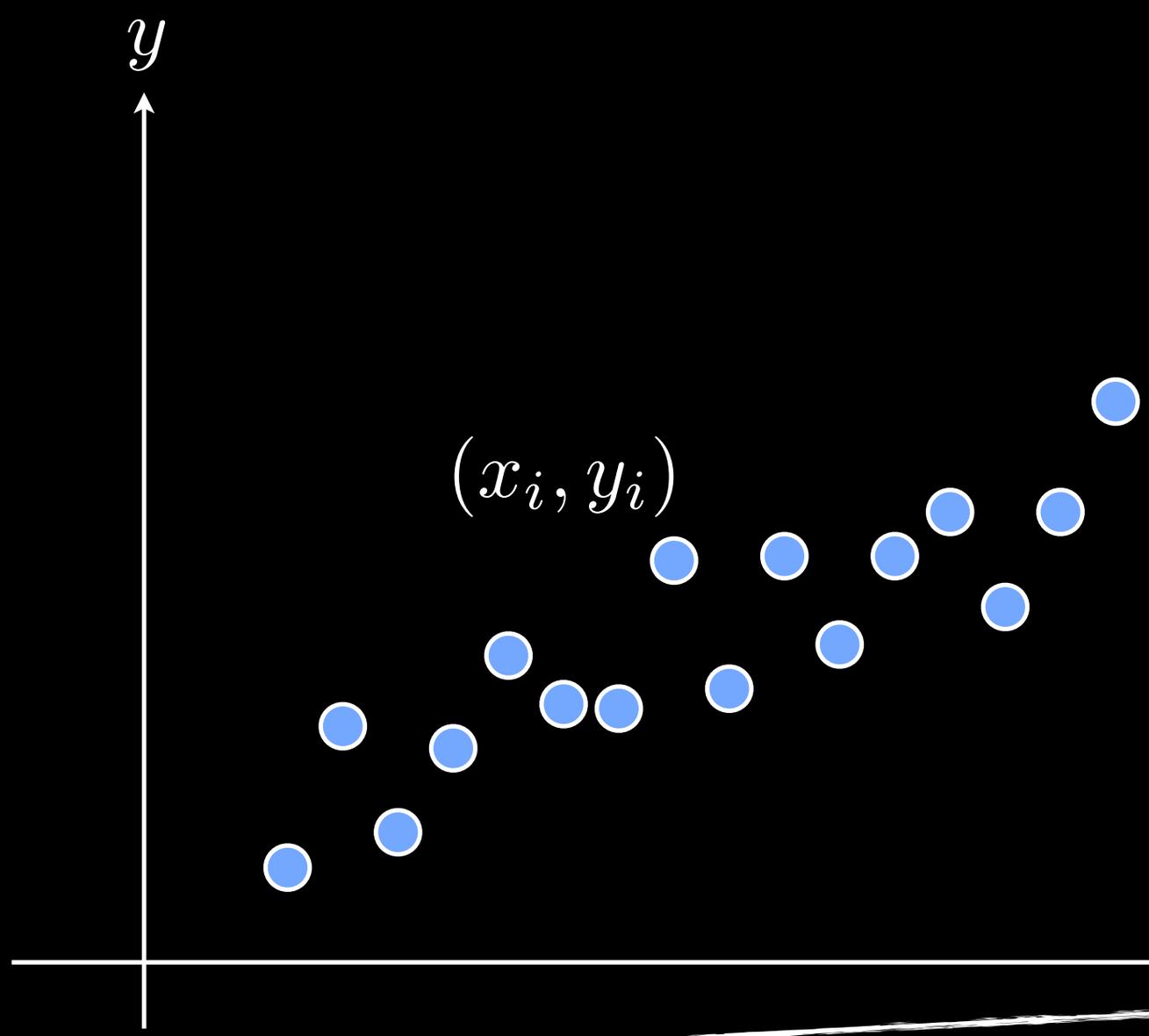
$$\mathbf{p} = \mathbf{A}^{-1}\mathbf{b}$$


y
x

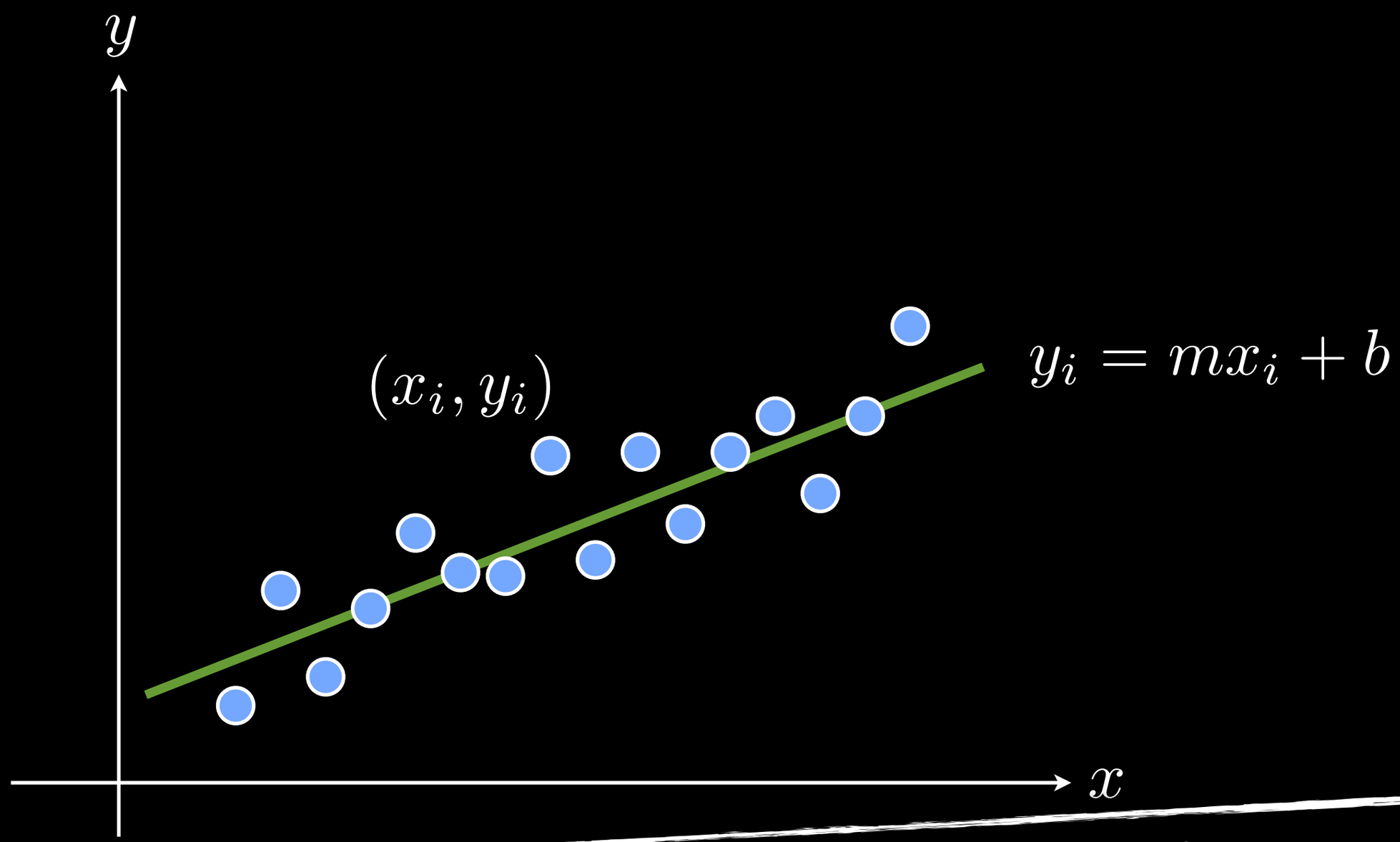

y
(x_i, y_i)
x



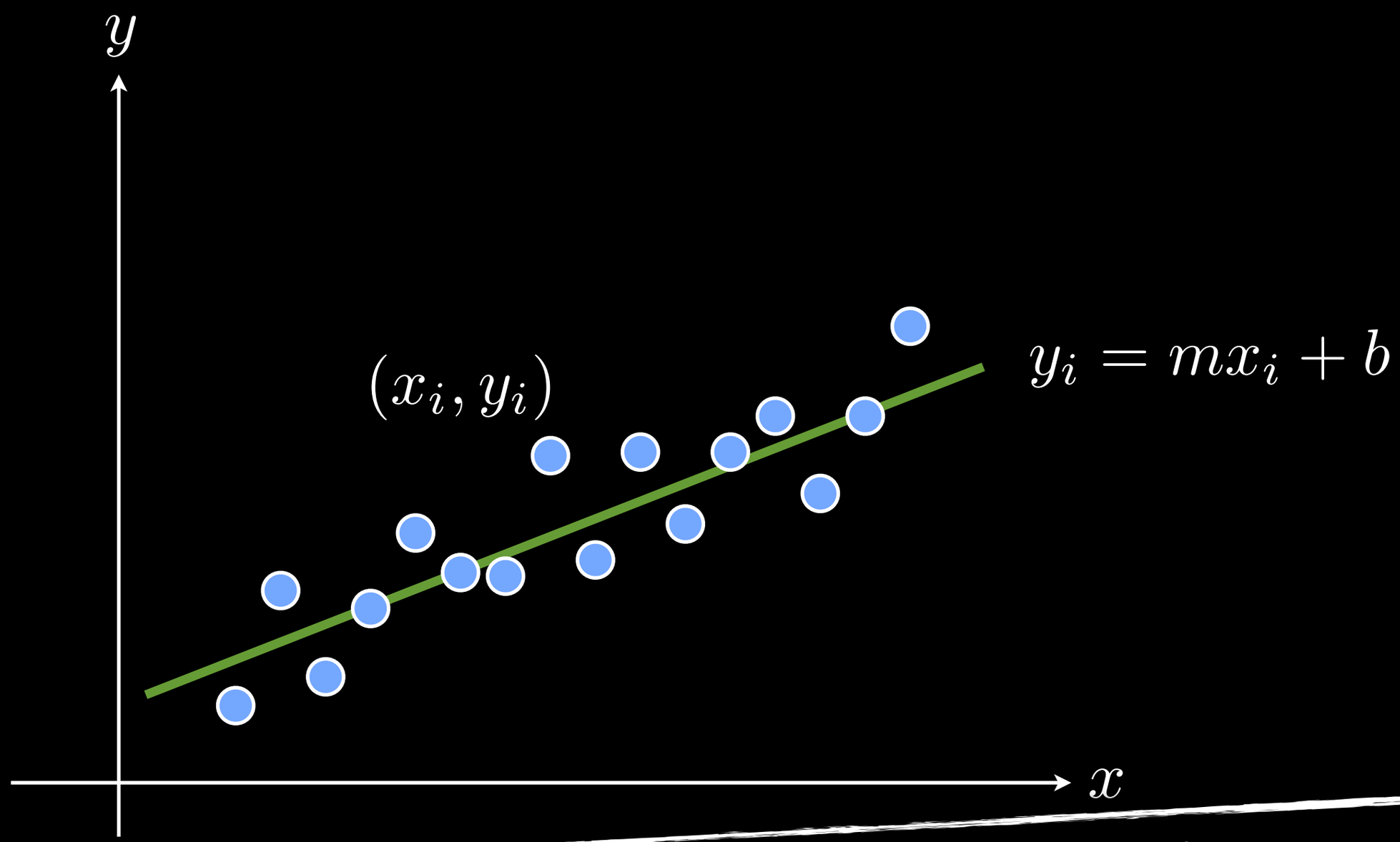
How do we fit a line?



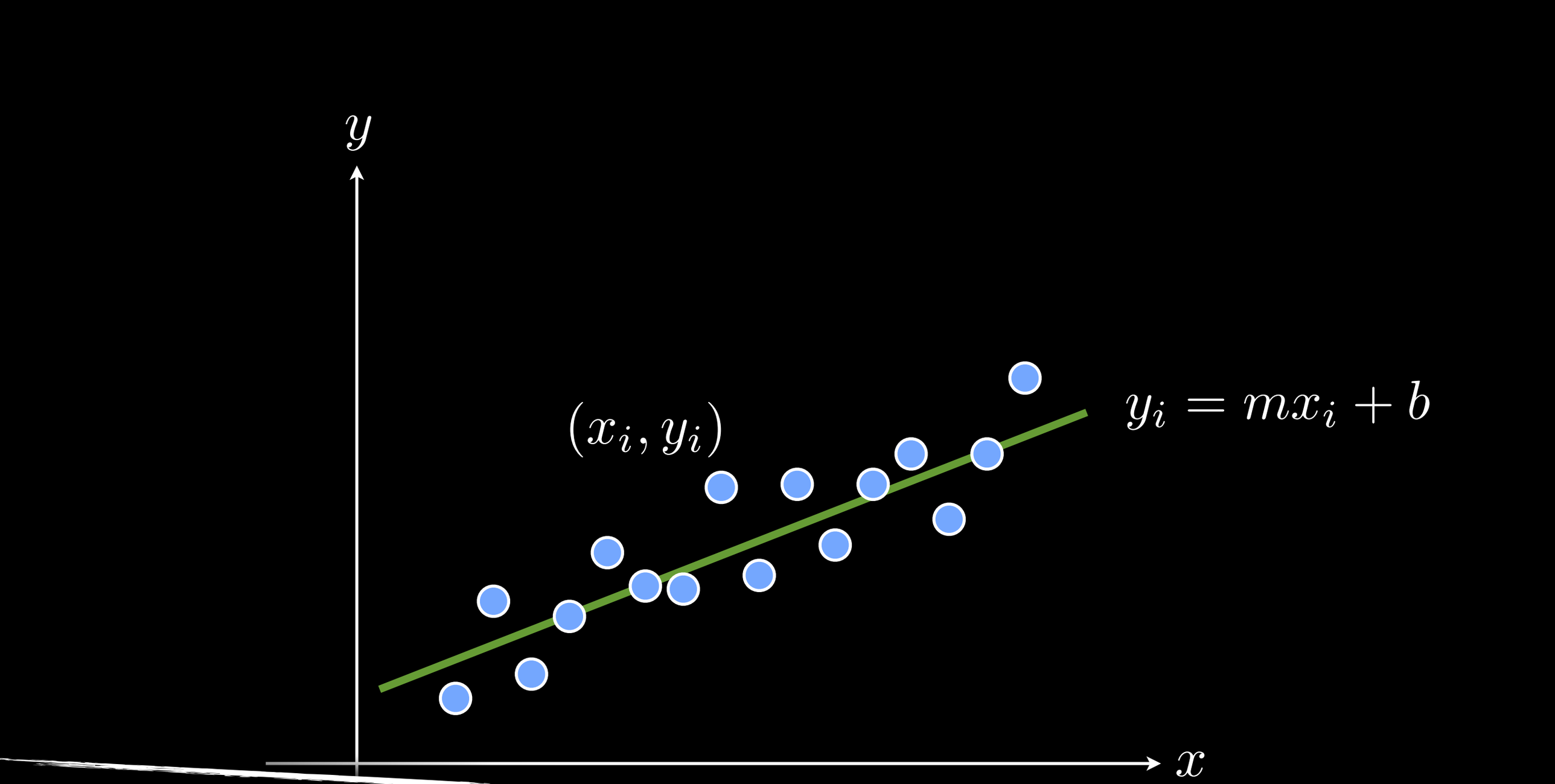
Find “best” fitting line that minimizes the “distances” to the data points



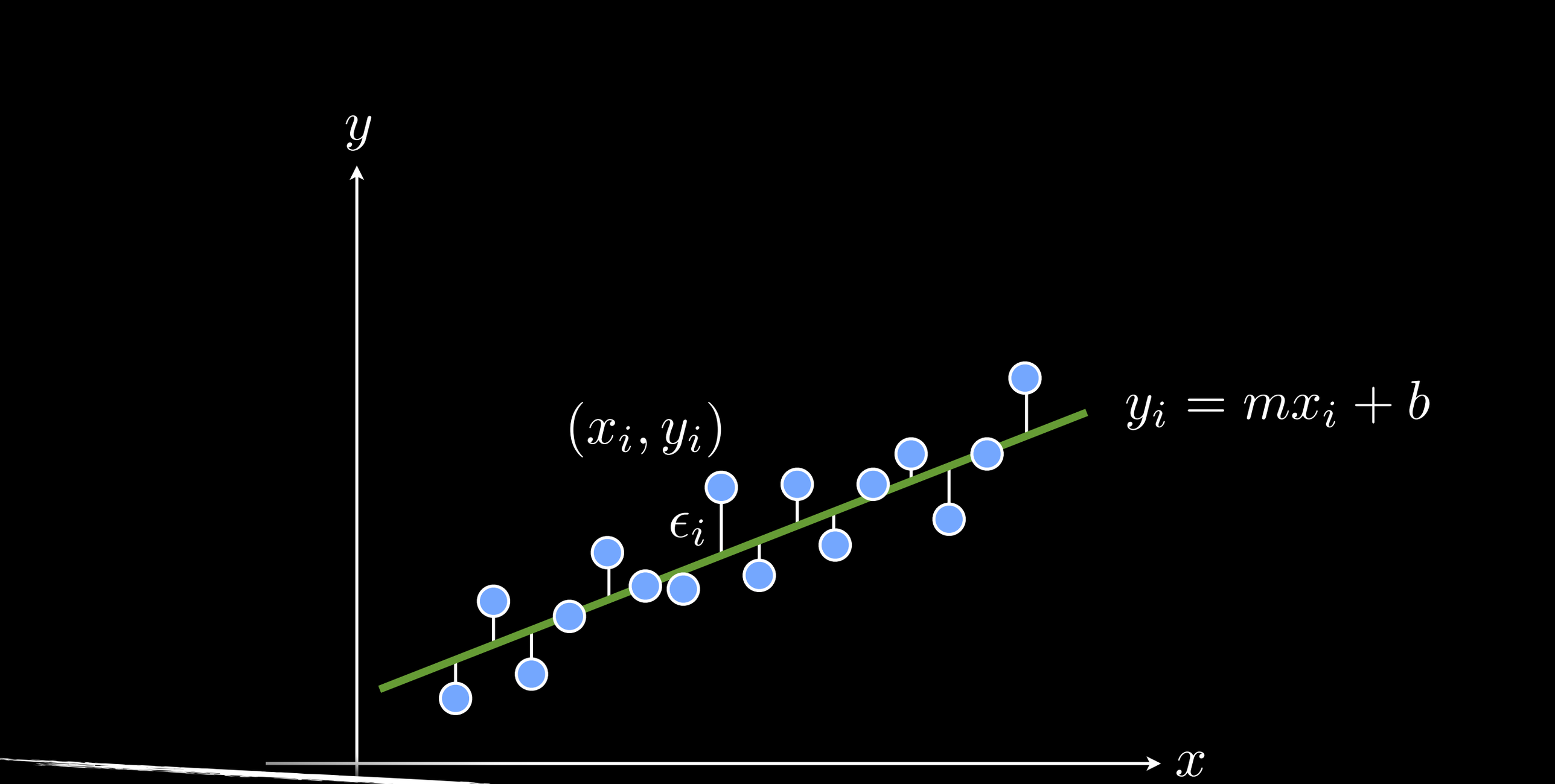
Find “best” fitting line that minimizes the “distances” to the data points



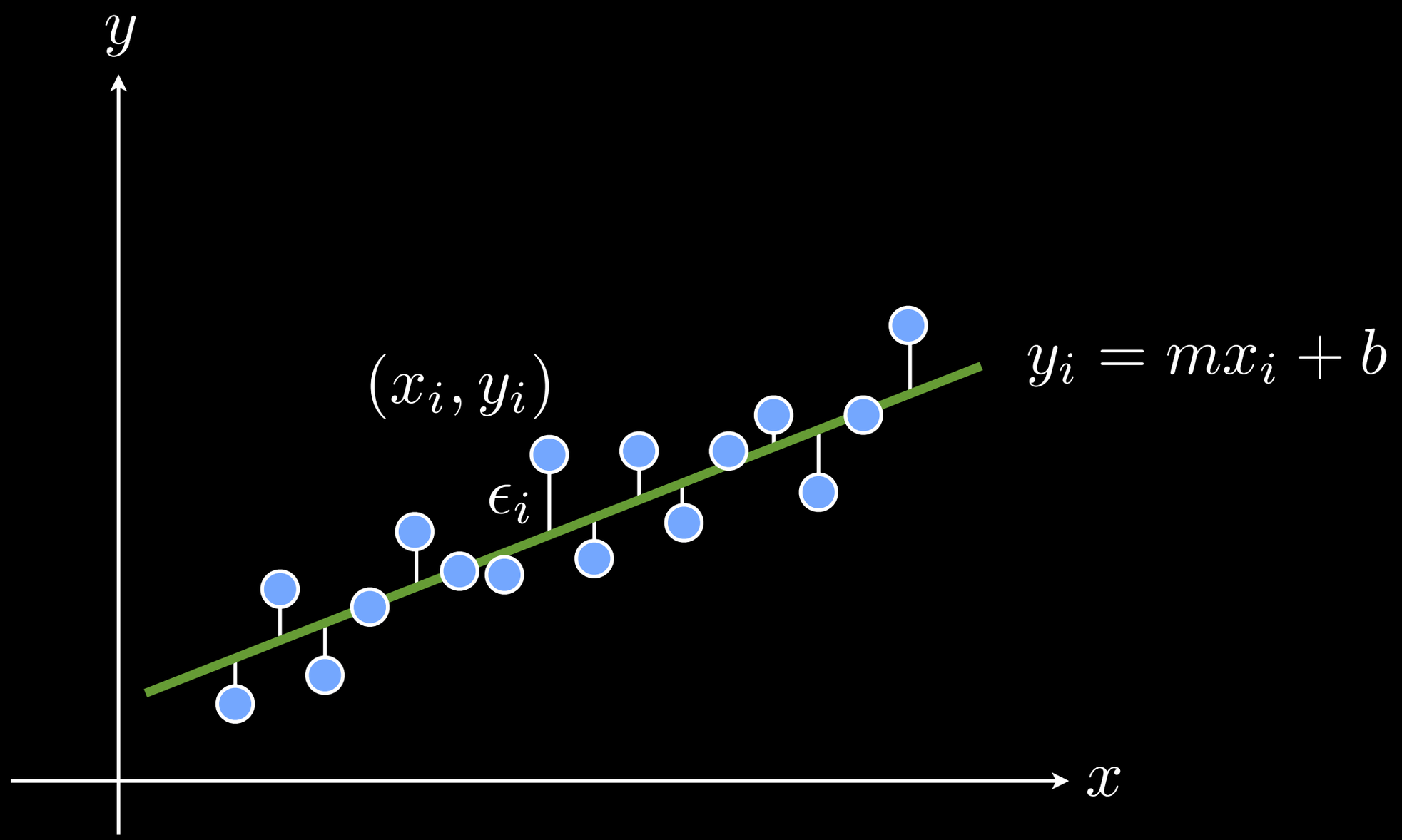
Find “best” fitting line that minimizes the “distances” to the data points



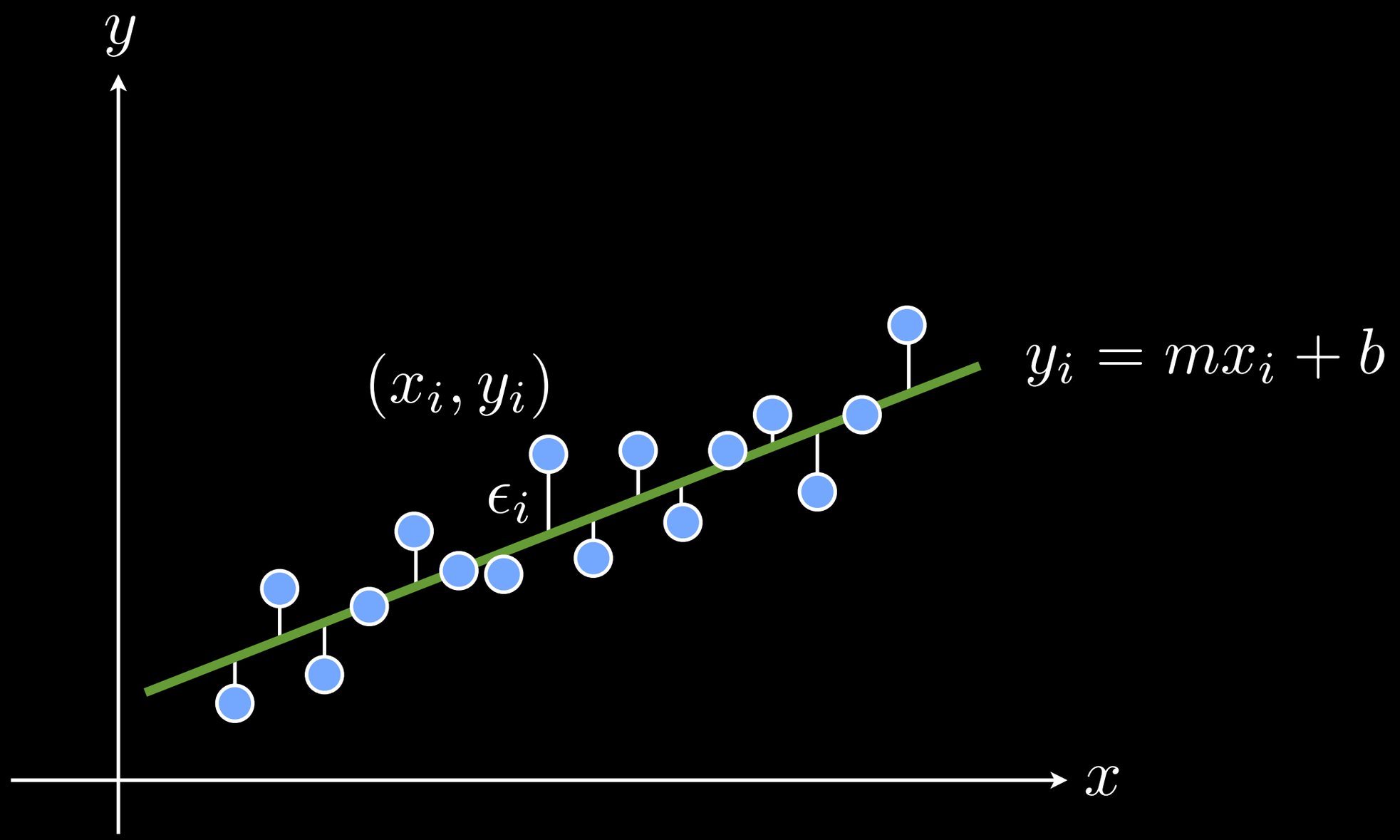
How do we measure the distance to the line?



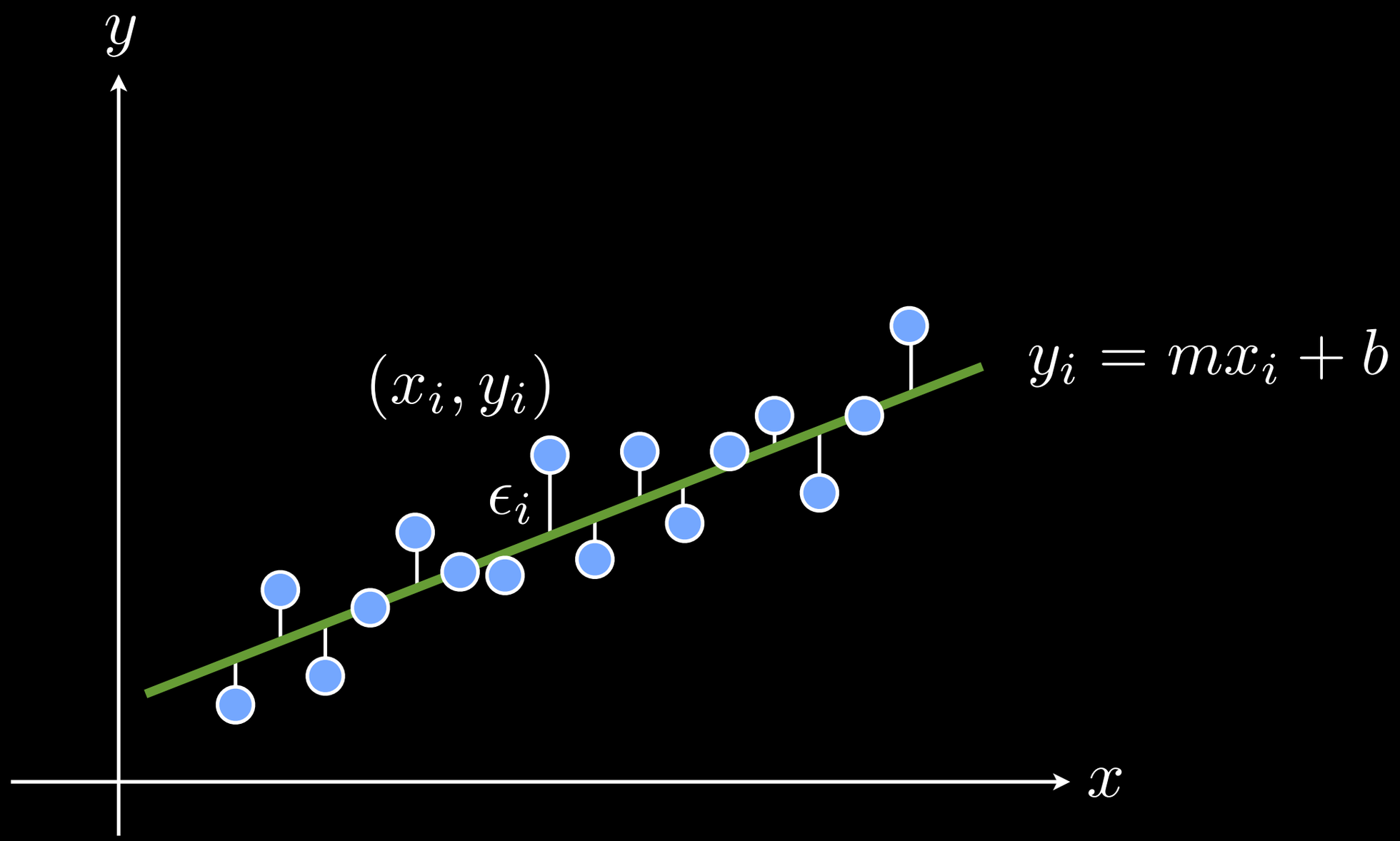
How do we measure the distance to the line?



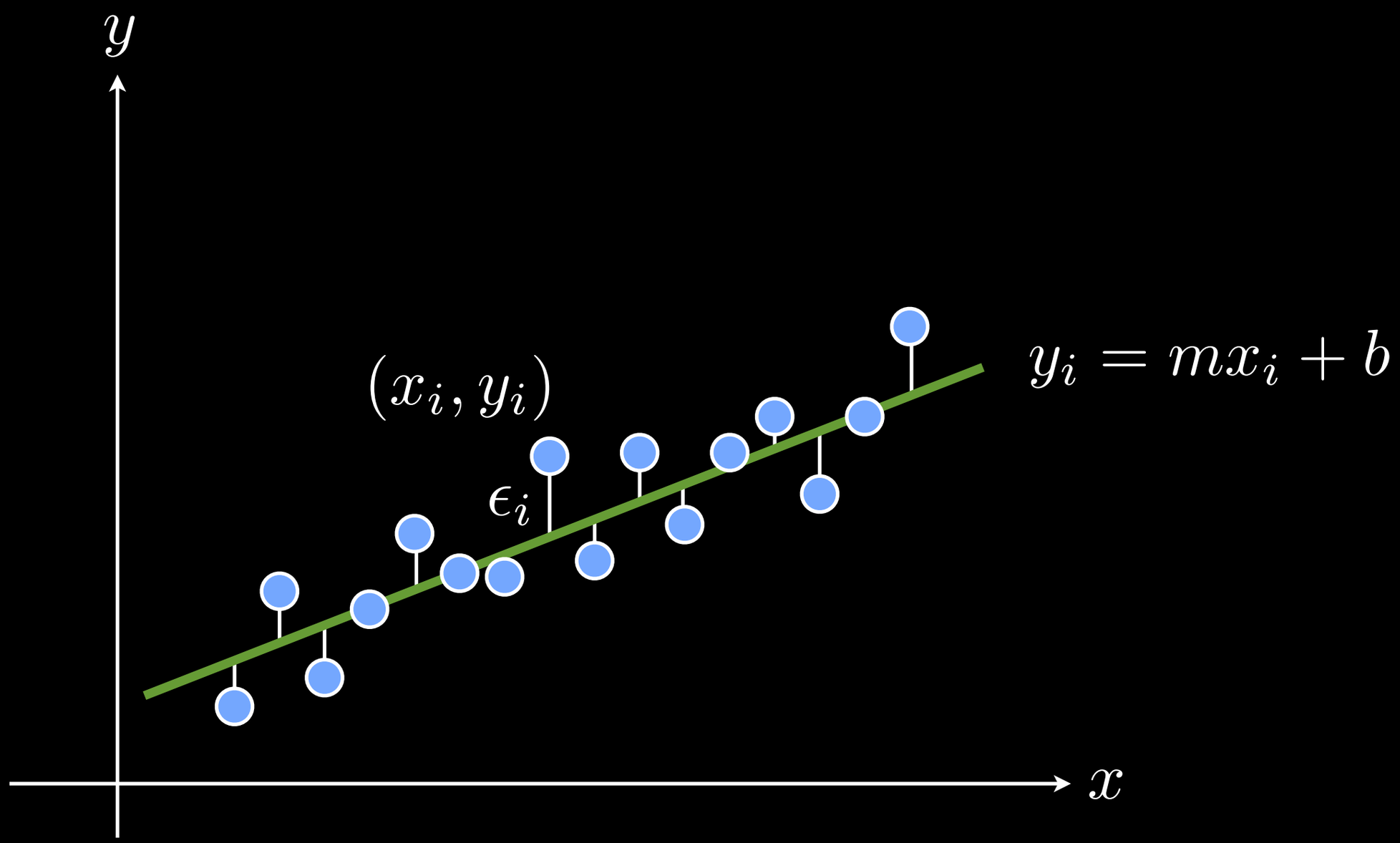
$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$



$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$



$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$



$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

unknown

$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

$$= \sum_{i=1}^{N} \left(y_i - \begin{pmatrix} x_i & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \right)^2$$

$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

$$= \sum_{i=1}^{N} \left(y_i - \begin{pmatrix} x_i & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \right)^2$$

$$= \| \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} - \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \|^2$$

$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

$$= \sum_{i=1}^{N} \left(y_i - \begin{pmatrix} x_i & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \right)^2$$

$$= \| \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{b}} - \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \|^2$$

$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

$$= \sum_{i=1}^{N} \left(y_i - \begin{pmatrix} x_i & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \right)^2$$

$$= \| \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{b}} - \underbrace{\begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix}}_{\mathbf{A}} \begin{pmatrix} m \\ b \end{pmatrix} \|^2$$

$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

$$= \sum_{i=1}^{N} \left(y_i - \begin{pmatrix} x_i & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \right)^2$$

$$= \| \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{b}} - \underbrace{\begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} m \\ b \end{pmatrix}}_{\mathbf{x}} \|^2$$

$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

$$= \sum_{i=1}^{N} \left(y_i - \begin{pmatrix} x_i & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \right)^2$$

$$= \| \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{b}} - \underbrace{\begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} m \\ b \end{pmatrix}}_{\mathbf{x}} \|^2$$

$$= \|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2$$

$$\epsilon = \sum_{i=1}^{N} (y_i - mx_i - b)^2$$

$$= \sum_{i=1}^{N} \left(y_i - \begin{pmatrix} x_i & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \right)^2$$

$$= \| \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{b}} - \underbrace{\begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} m \\ b \end{pmatrix}}_{\mathbf{x}} \|^2$$

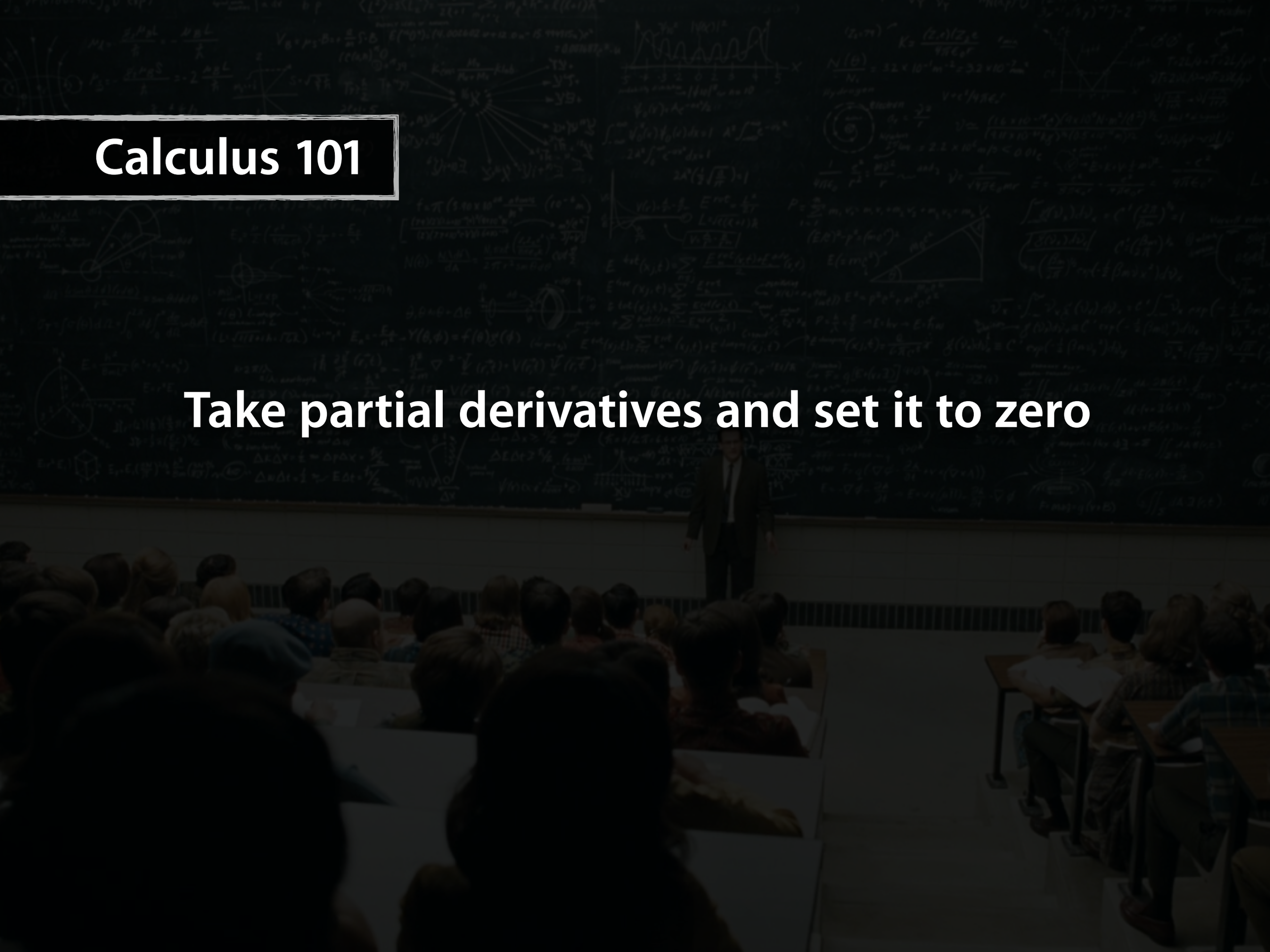
$$= \|\mathbf{b} - \mathbf{A}\mathbf{x}\|^2$$

How do we minimize ϵ ?

Calculus 101

Calculus 101

Take partial derivatives and set it to zero

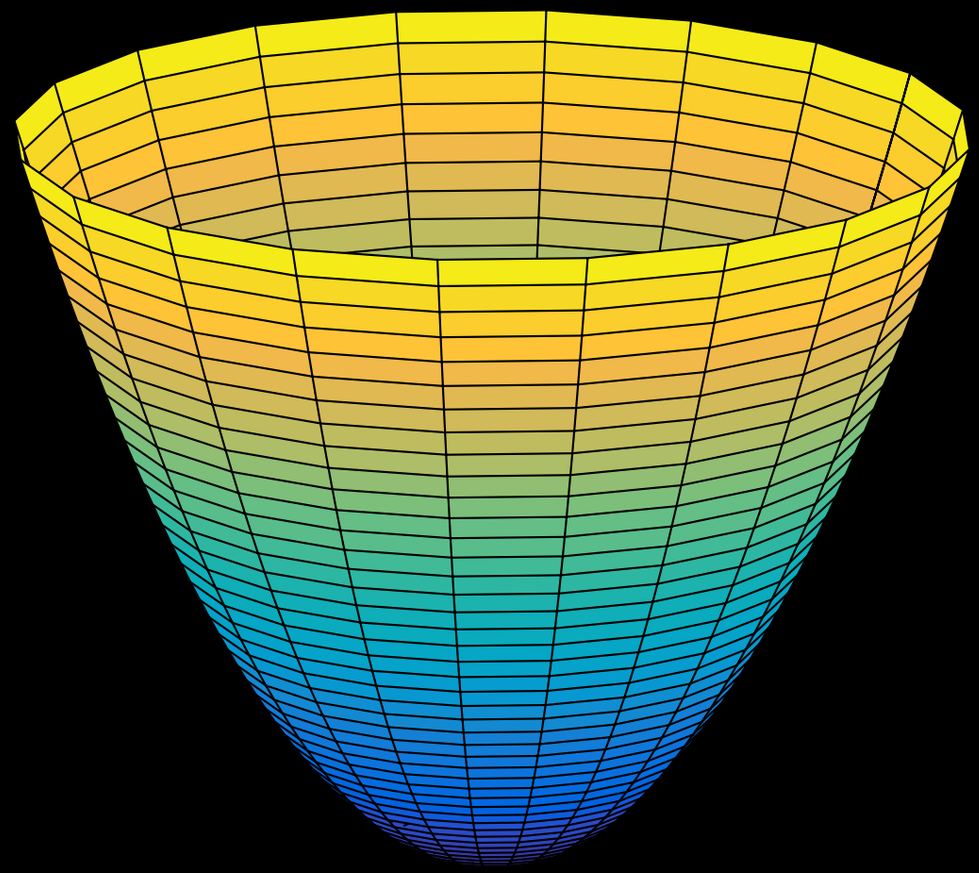


Calculus 101

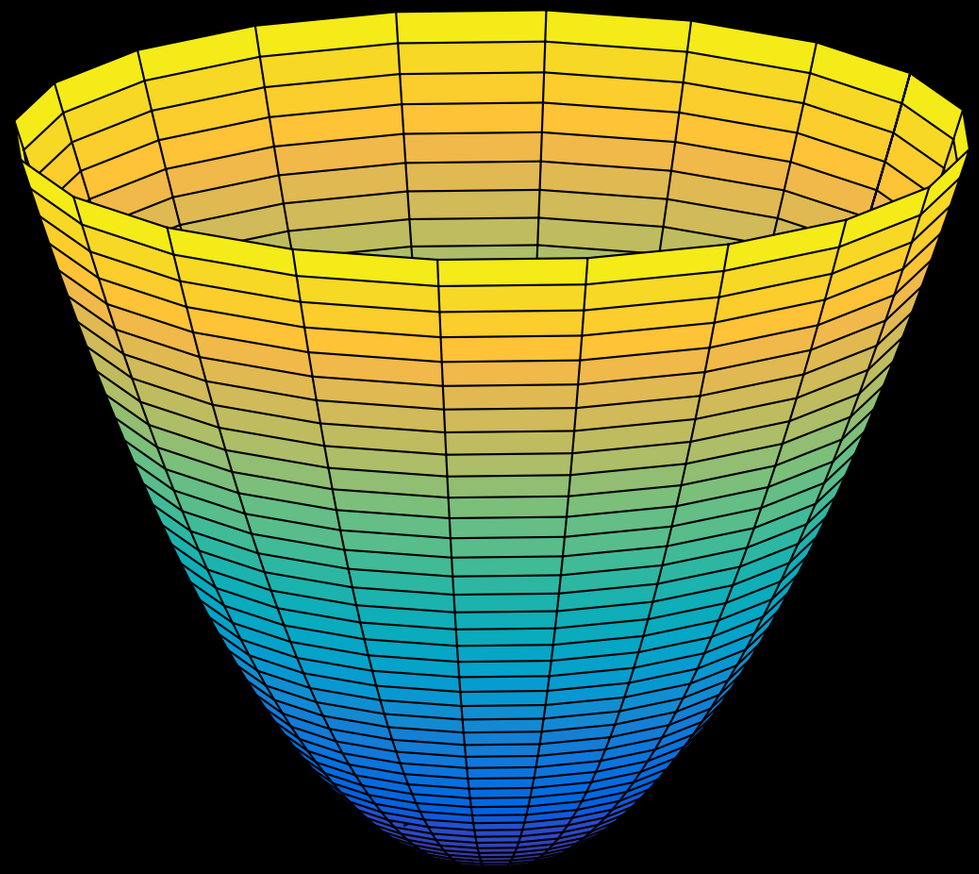
Take partial derivatives and set it to zero

$$\frac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$$

Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$

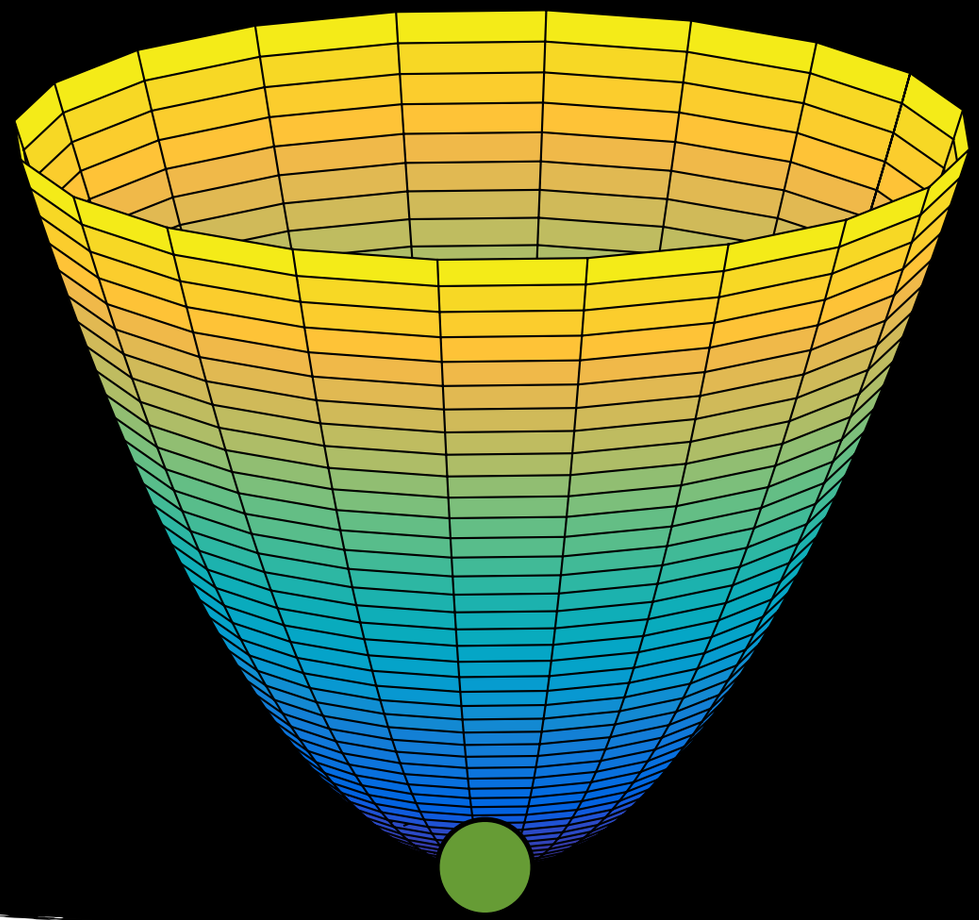


Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$



gradient is equal to zero at extrema

Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$



gradient is equal to zero at extrema

Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$

Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$

$$\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \mathbf{b}$$

Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$

$\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \mathbf{b}$ **Normal equation**

Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$

$\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \mathbf{b}$ **Normal equation**

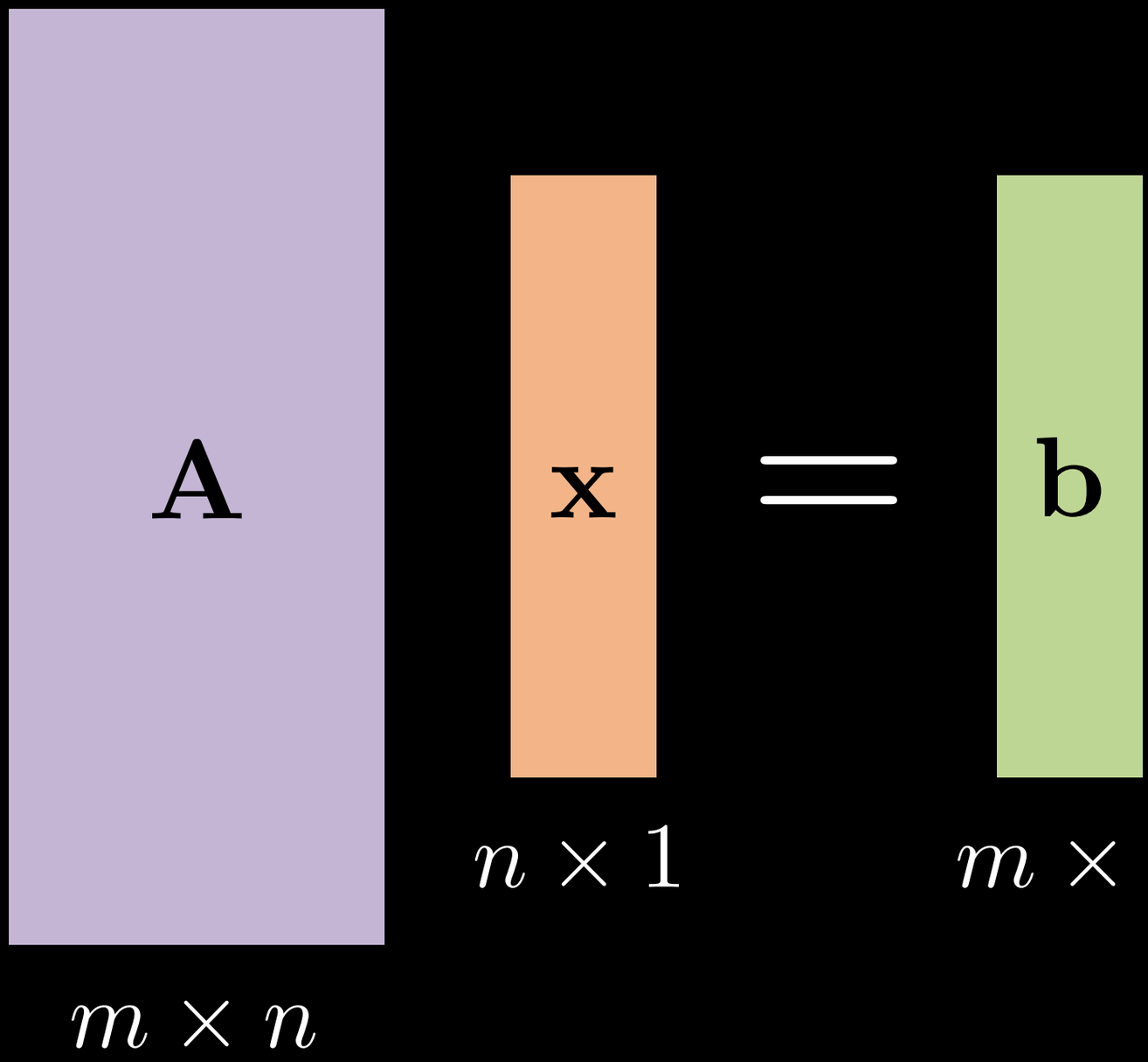
Least Squares Solution $\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$

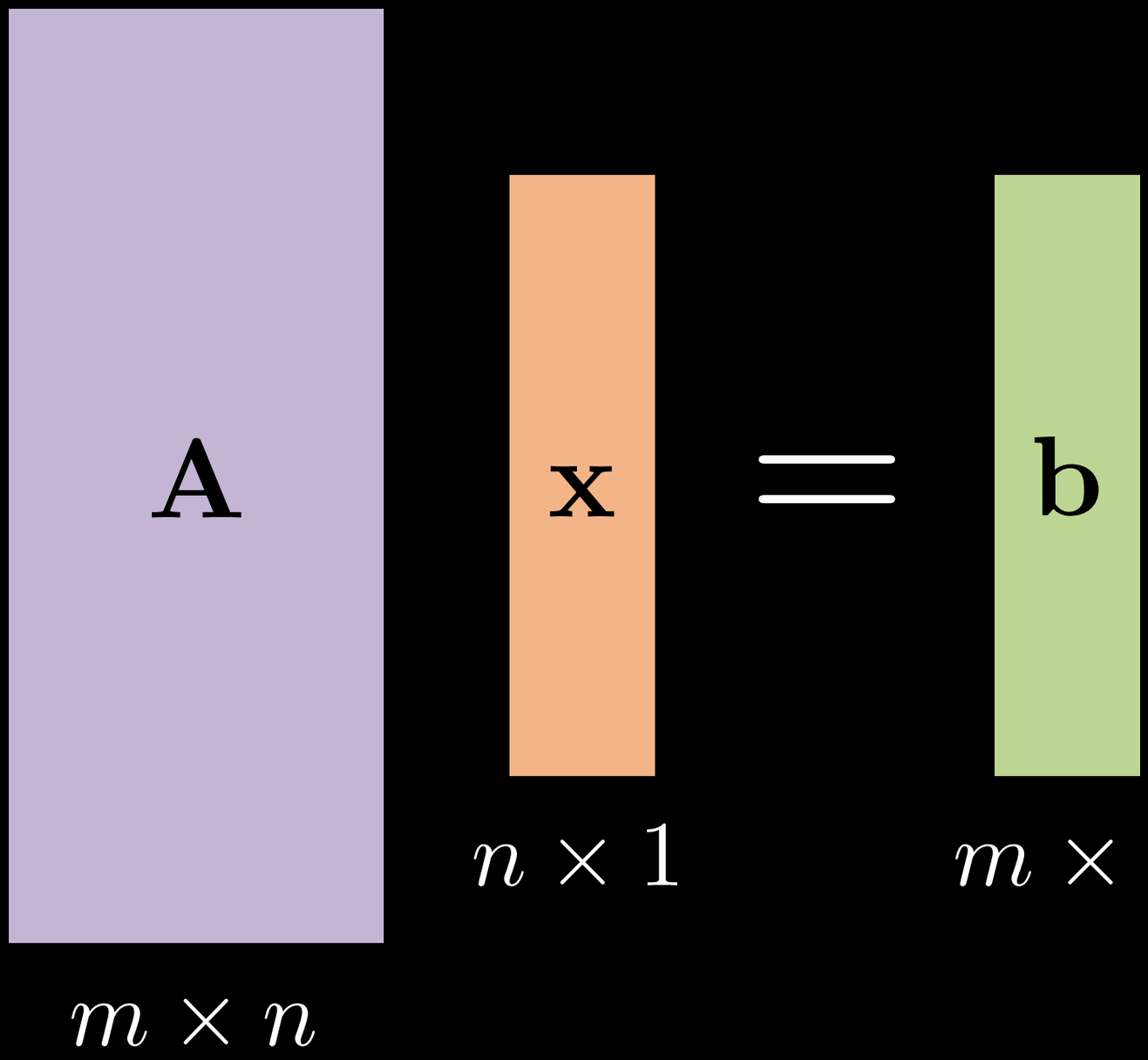
Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$

$\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \mathbf{b}$ **Normal equation**

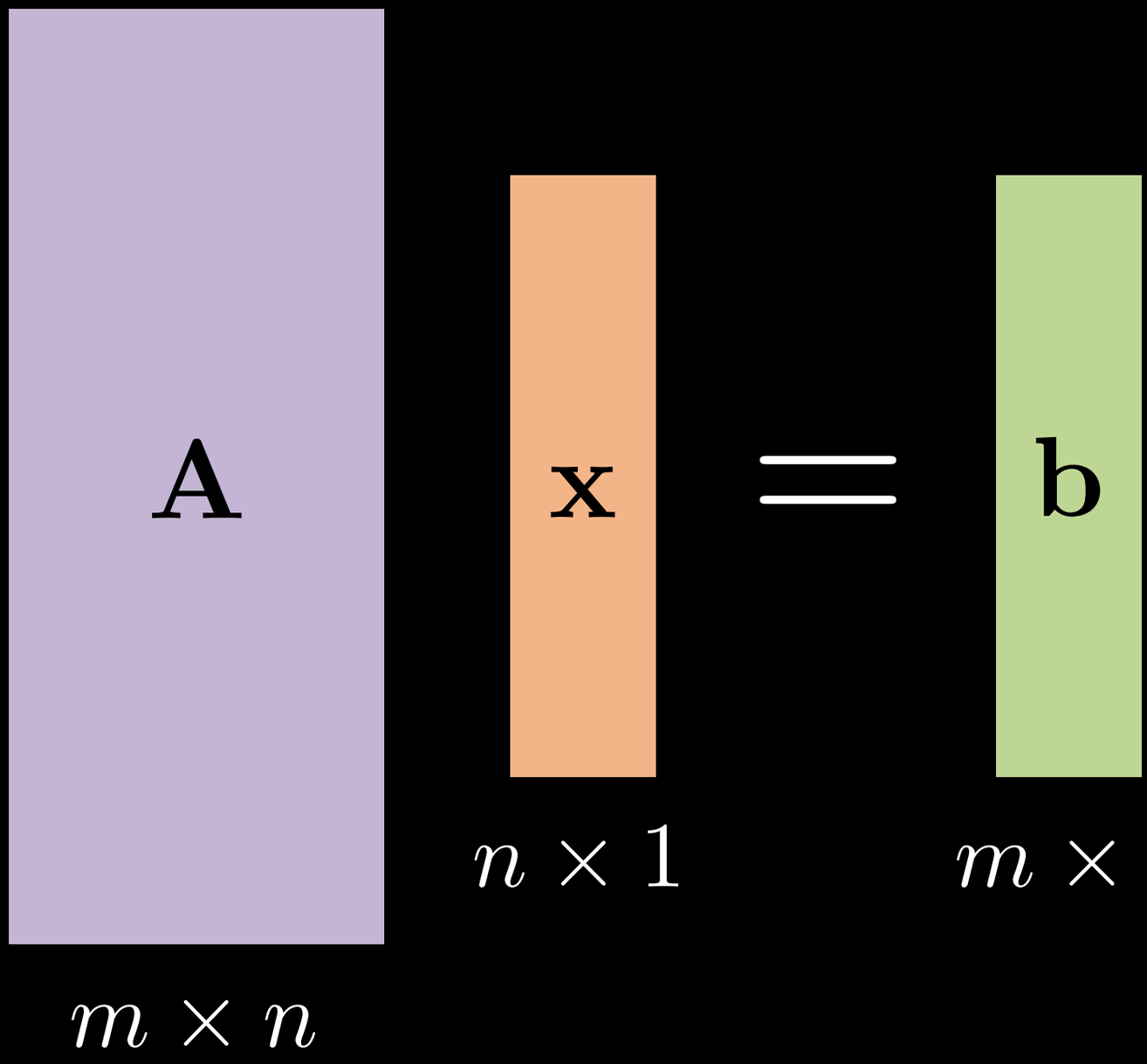
Least Squares Solution $\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$

pseudo-inverse of $\mathbf{A}$


A
x
=
b
$m \times n$
$n \times 1$
$m \times 1$



$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$$



$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$$

“ If you understand something in only one way, then you don’t really understand it at all.”

— Marvin Minsky

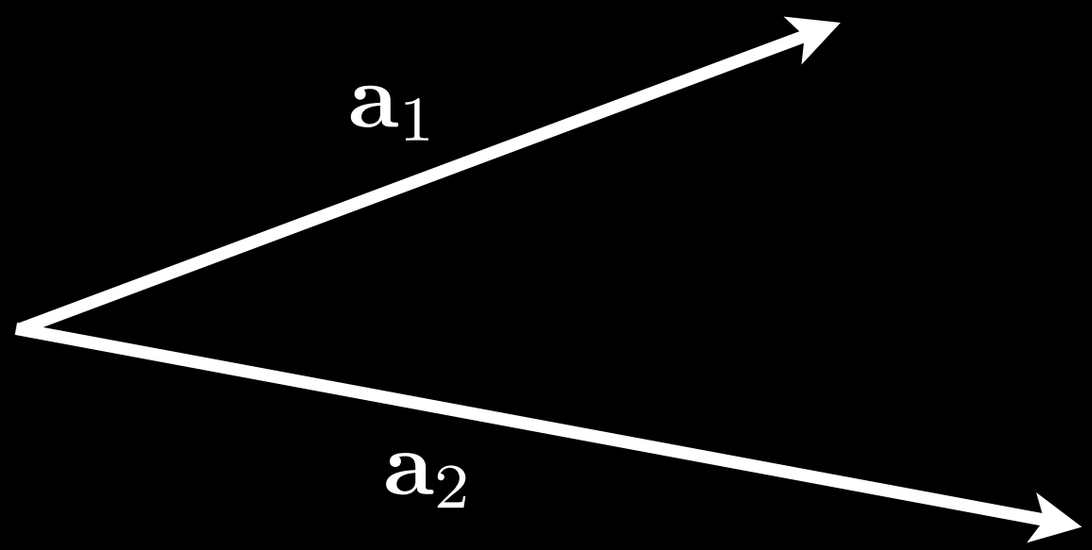
“ If you understand something in only one way, then you don’t really understand it at all.”

— Marvin Minsky

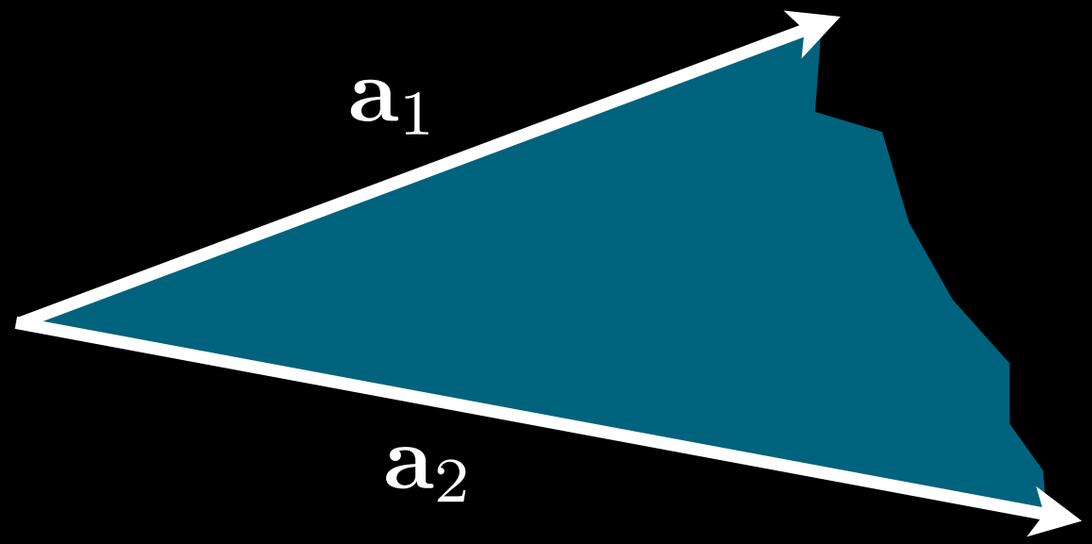
Linear algebra view of linear least-squares

Goal: Make $\|\mathrm{b} - \mathrm{A}\mathrm{x}\|$ as small as possible

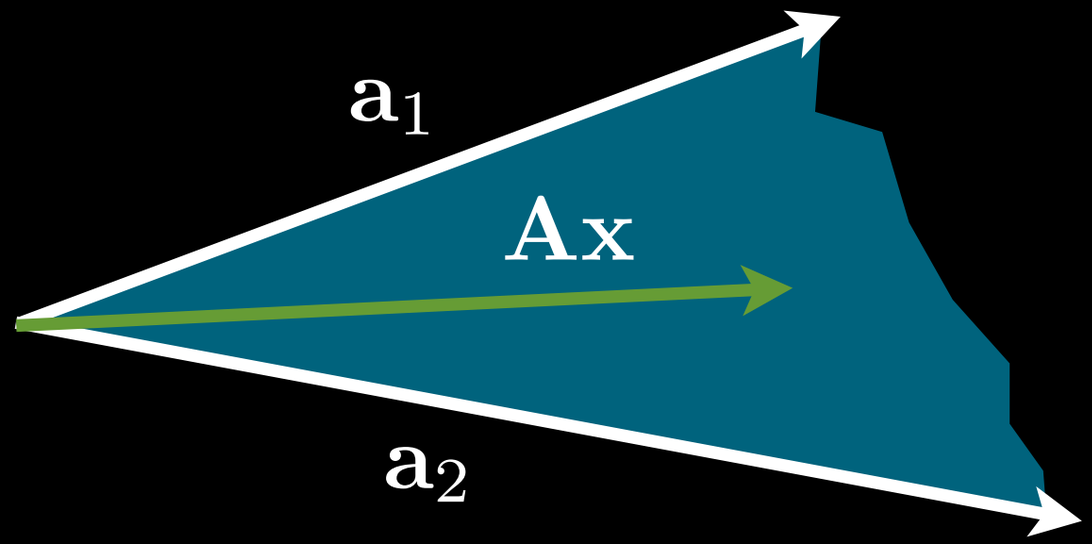
Goal: **Make $\|\mathrm{b} - \mathrm{A}\mathrm{x}\|$ as small as possible**



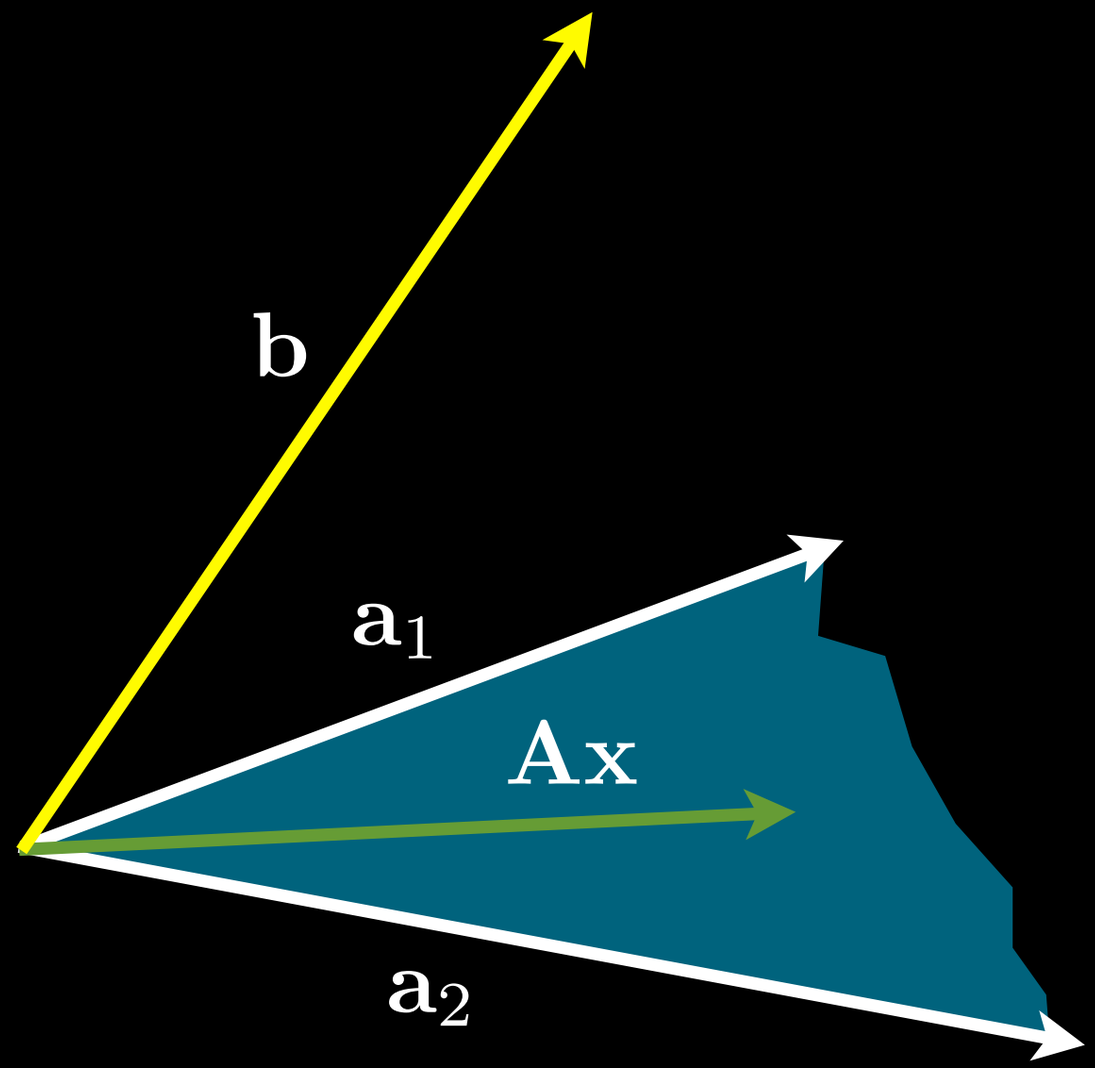
Goal: **Make $\|\mathrm{b} - \mathrm{Ax}\|$ as small as possible**



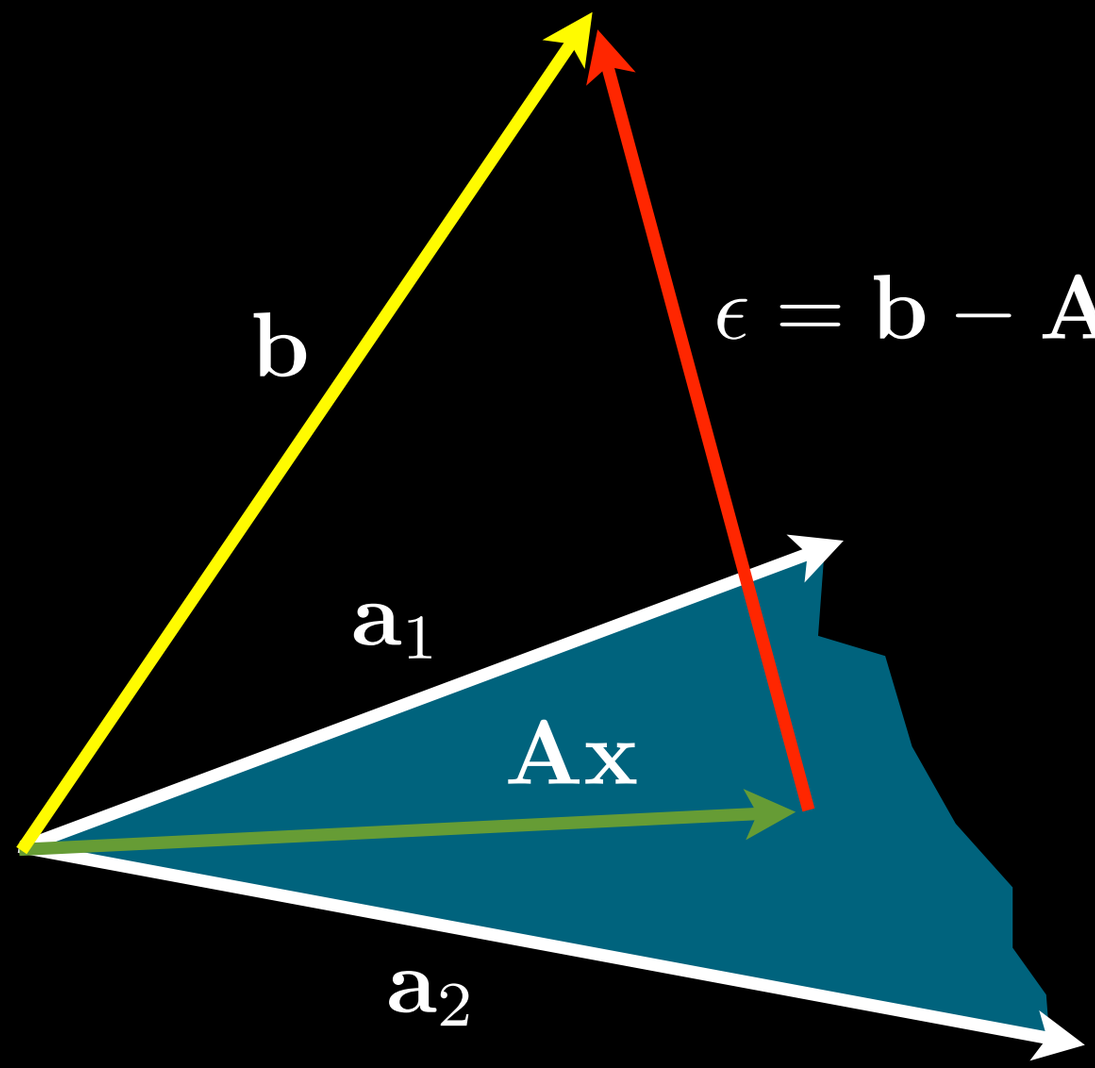
Goal: Make $\|\mathrm{b} - \mathrm{Ax}\|$ as small as possible



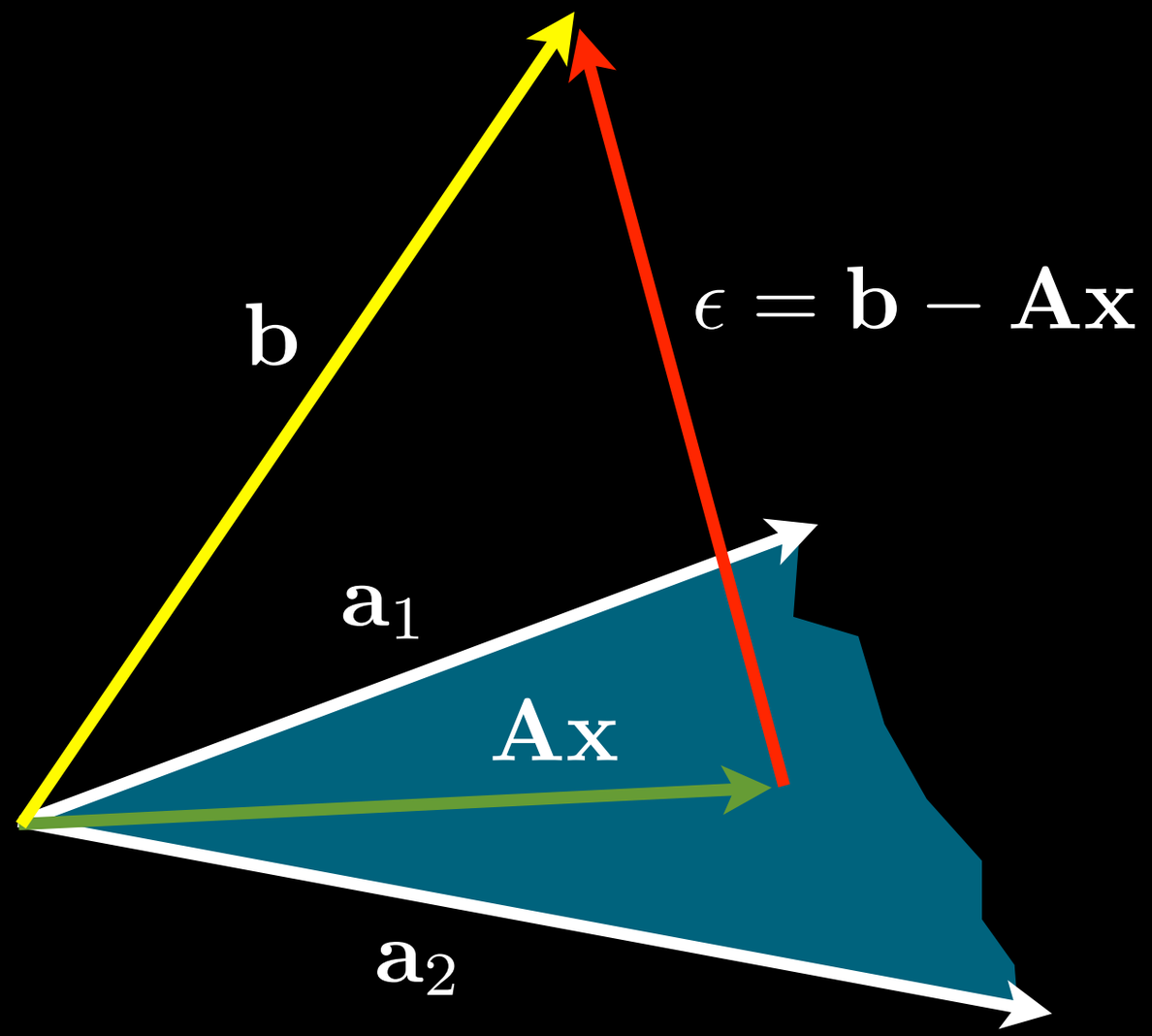
Goal: Make $\|\mathrm{b} - \mathrm{Ax}\|$ as small as possible



Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible

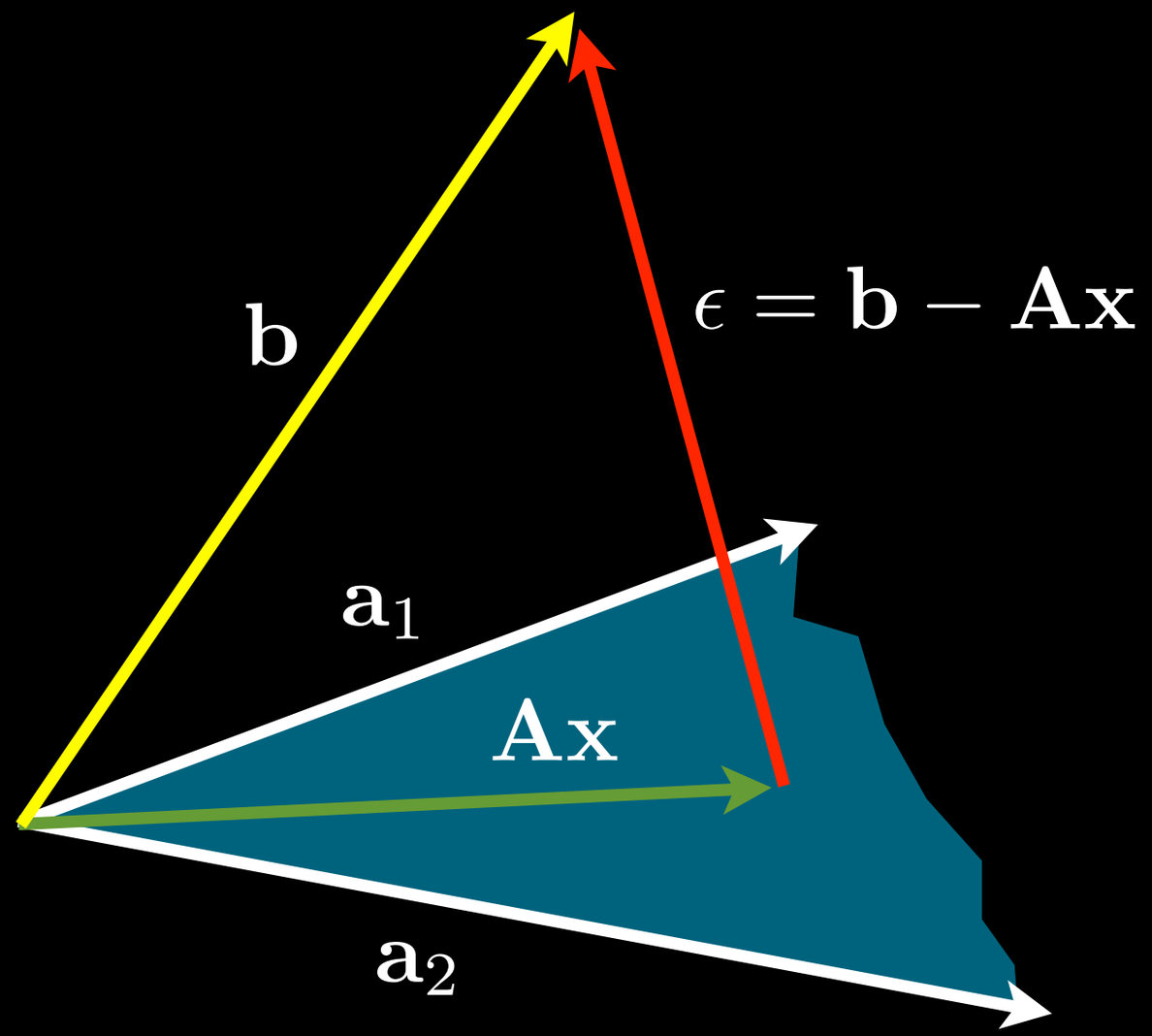


Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



How do we minimize the length of ϵ ?

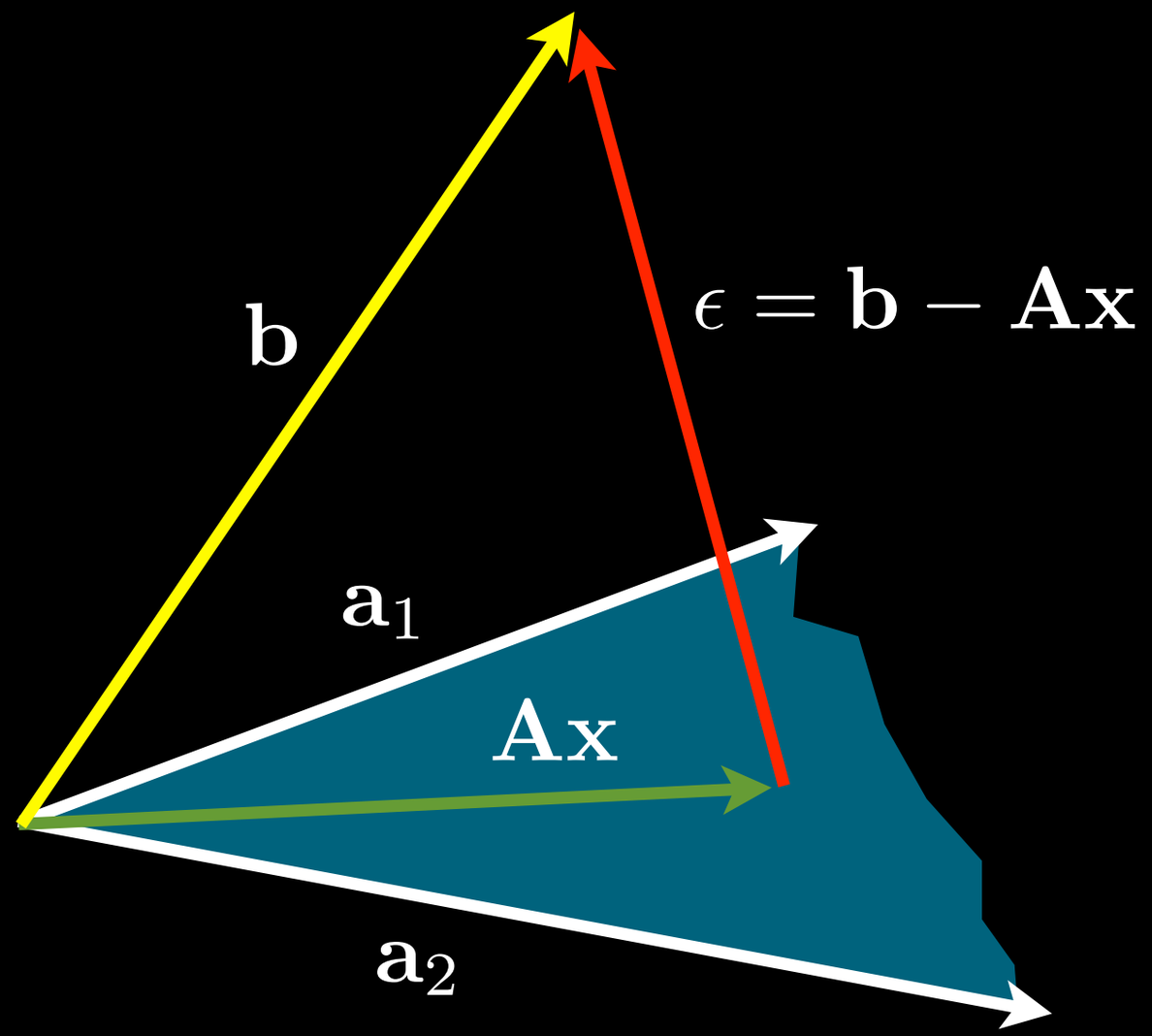
Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



How do we minimize the length of ϵ ?

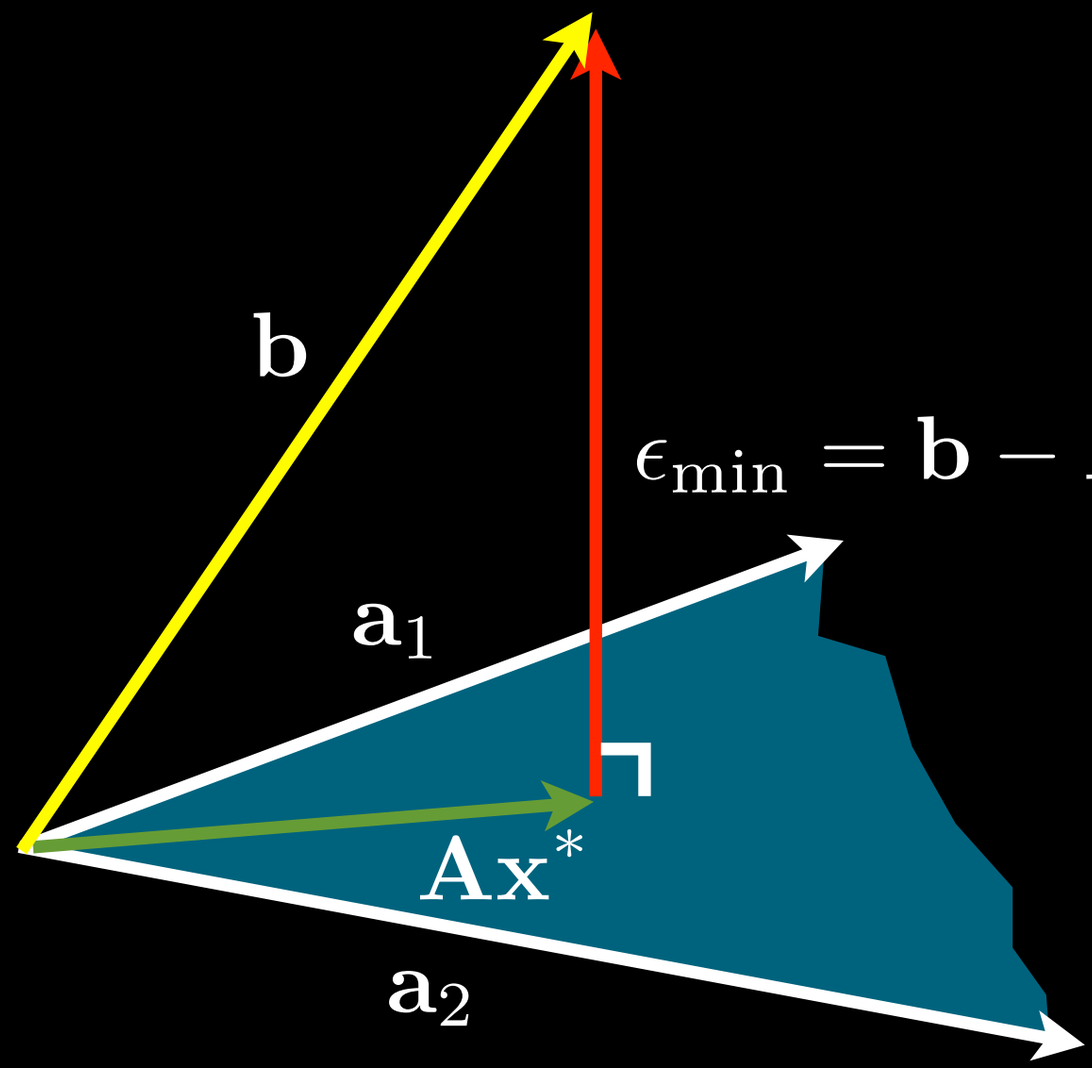
Choose the closest point to $\mathbf{b}$ in the plane

Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



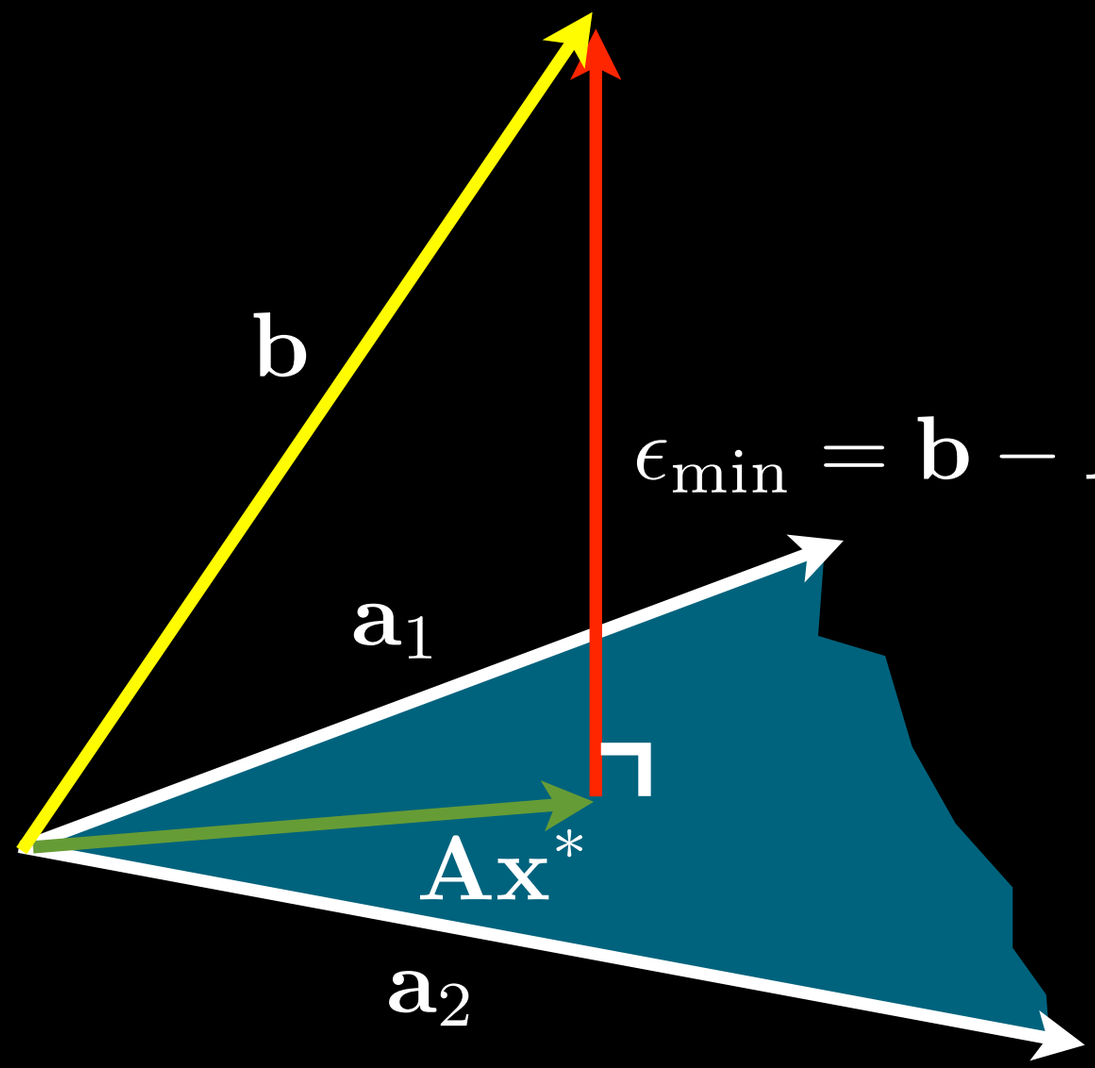
$\epsilon_{\min}$ is perpendicular to plane

Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



$\epsilon_{\min}$ **is perpendicular to plane**

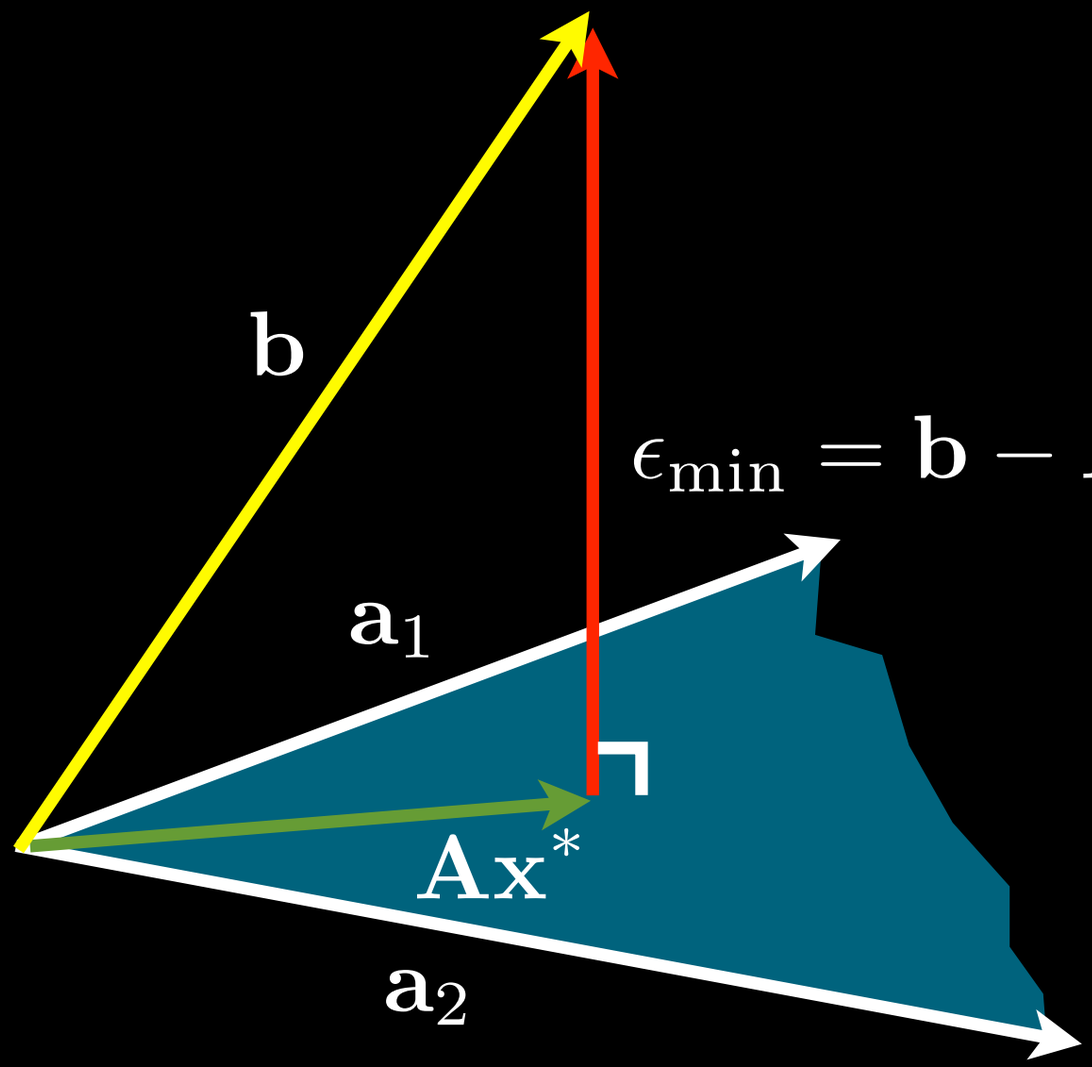
Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



$\epsilon_{\min}$ **is perpendicular to plane**

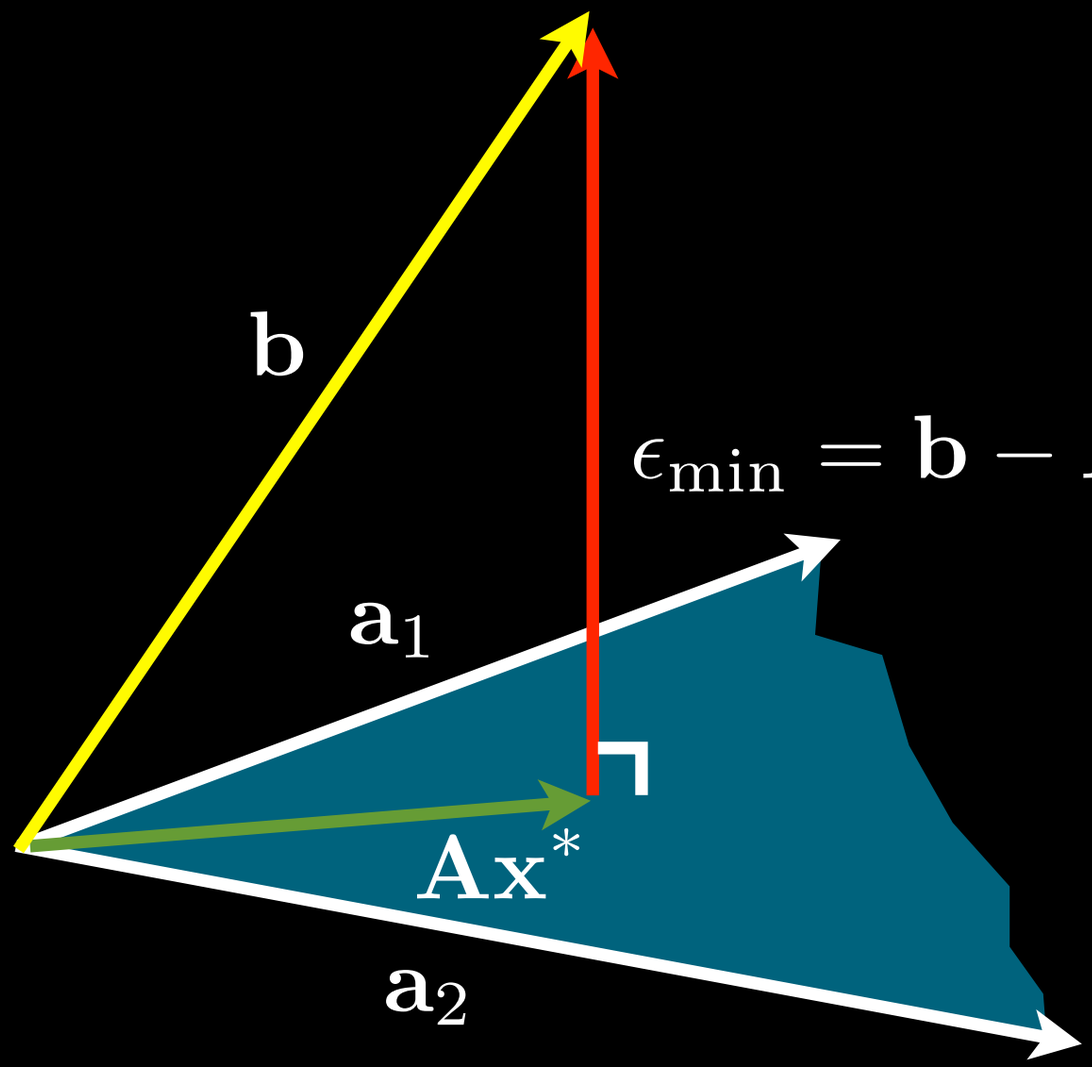
$$\mathbf{A}^\top(\mathbf{b} - \mathbf{A}\mathbf{x}) = \mathbf{0}$$

Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



$$\mathbf{A}^\top(\mathbf{b} - \mathbf{A}\mathbf{x}) = \mathbf{0}$$

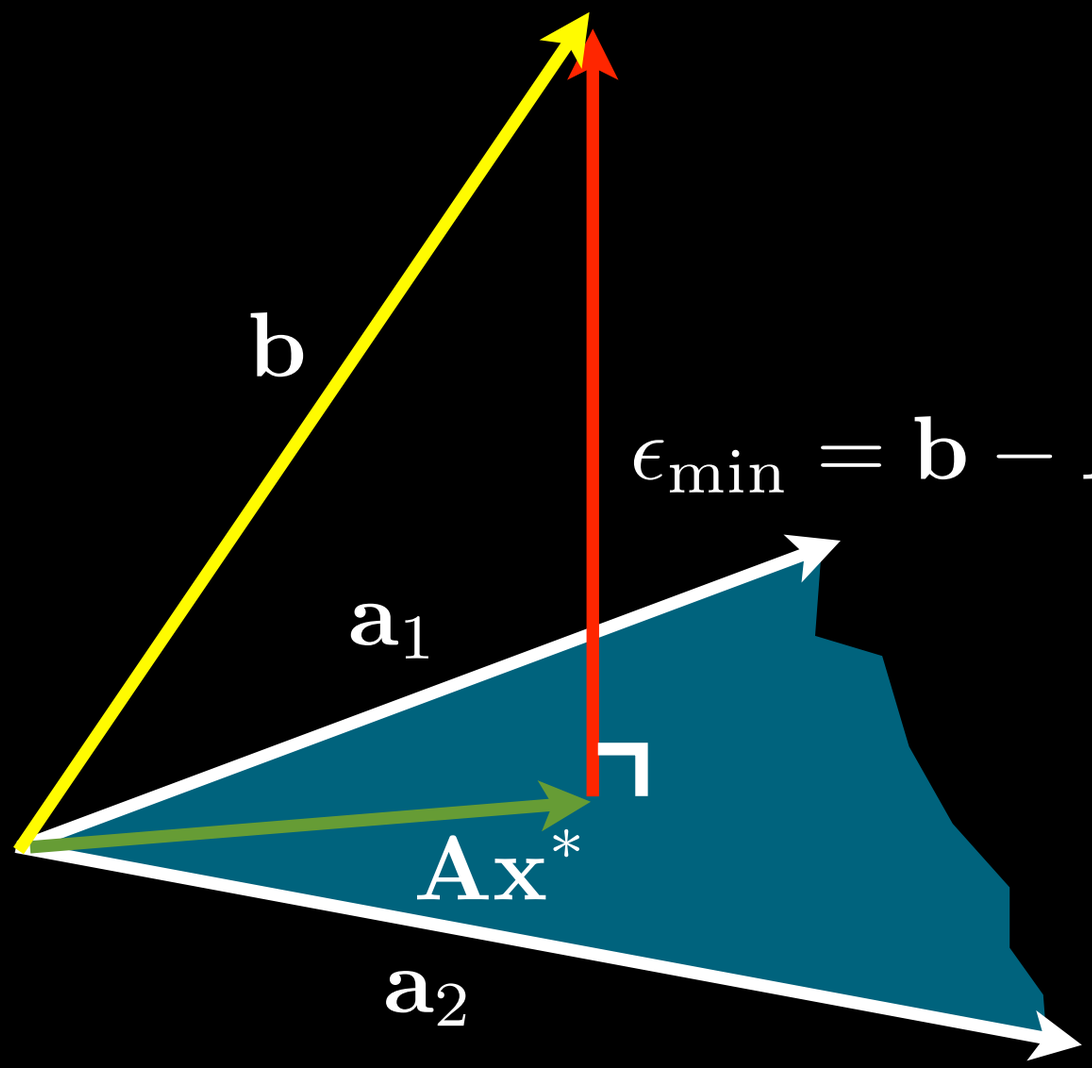
Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



$$\mathbf{A}^\top(\mathbf{b} - \mathbf{A}\mathbf{x}) = \mathbf{0}$$

$$\mathbf{A}^\top\mathbf{b} - \mathbf{A}^\top\mathbf{A}\mathbf{x} = \mathbf{0}$$

Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible

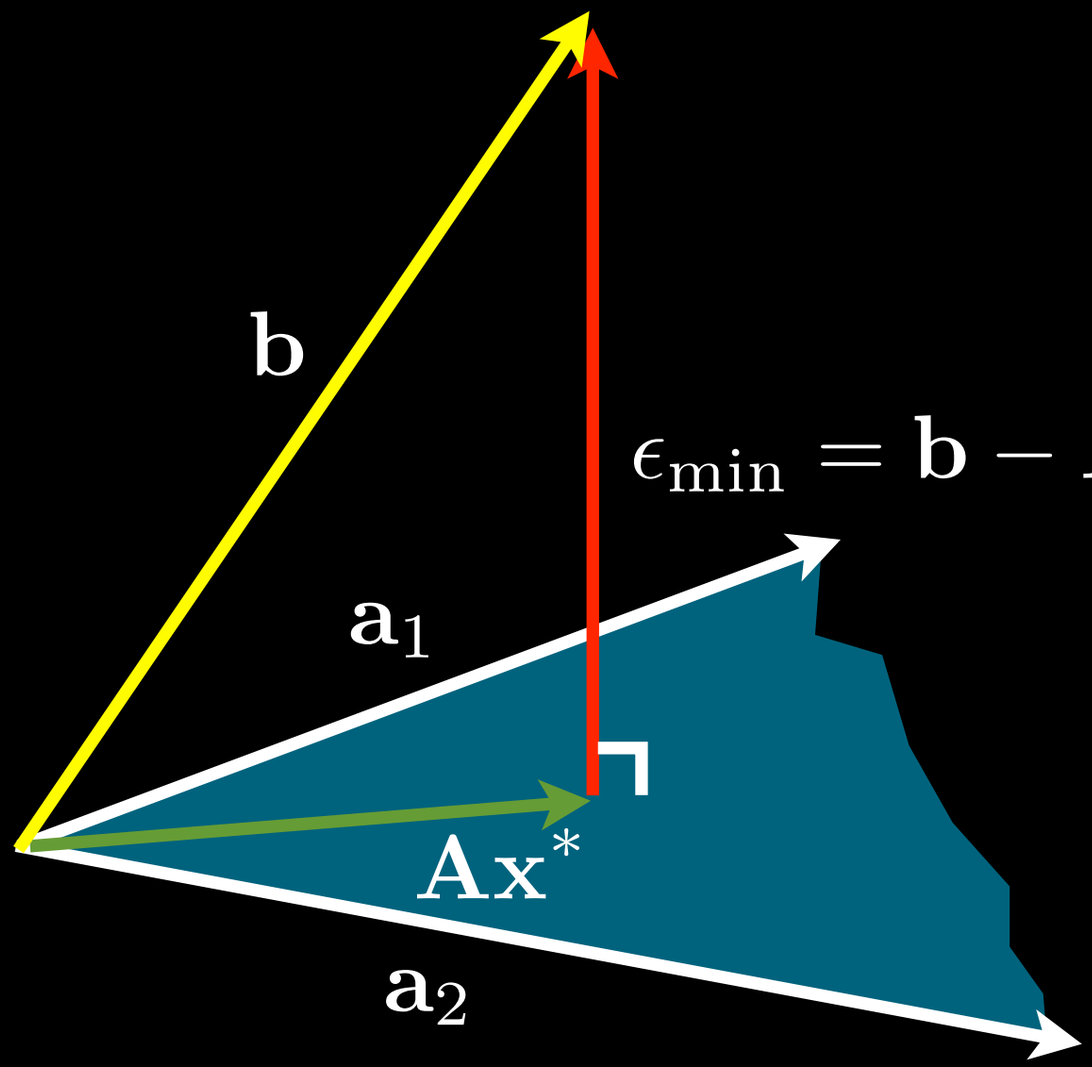


$$\mathbf{A}^\top(\mathbf{b} - \mathbf{A}\mathbf{x}) = \mathbf{0}$$

$$\mathbf{A}^\top\mathbf{b} - \mathbf{A}^\top\mathbf{A}\mathbf{x} = \mathbf{0}$$

$$\mathbf{x} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{b}$$

Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



$$\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$$

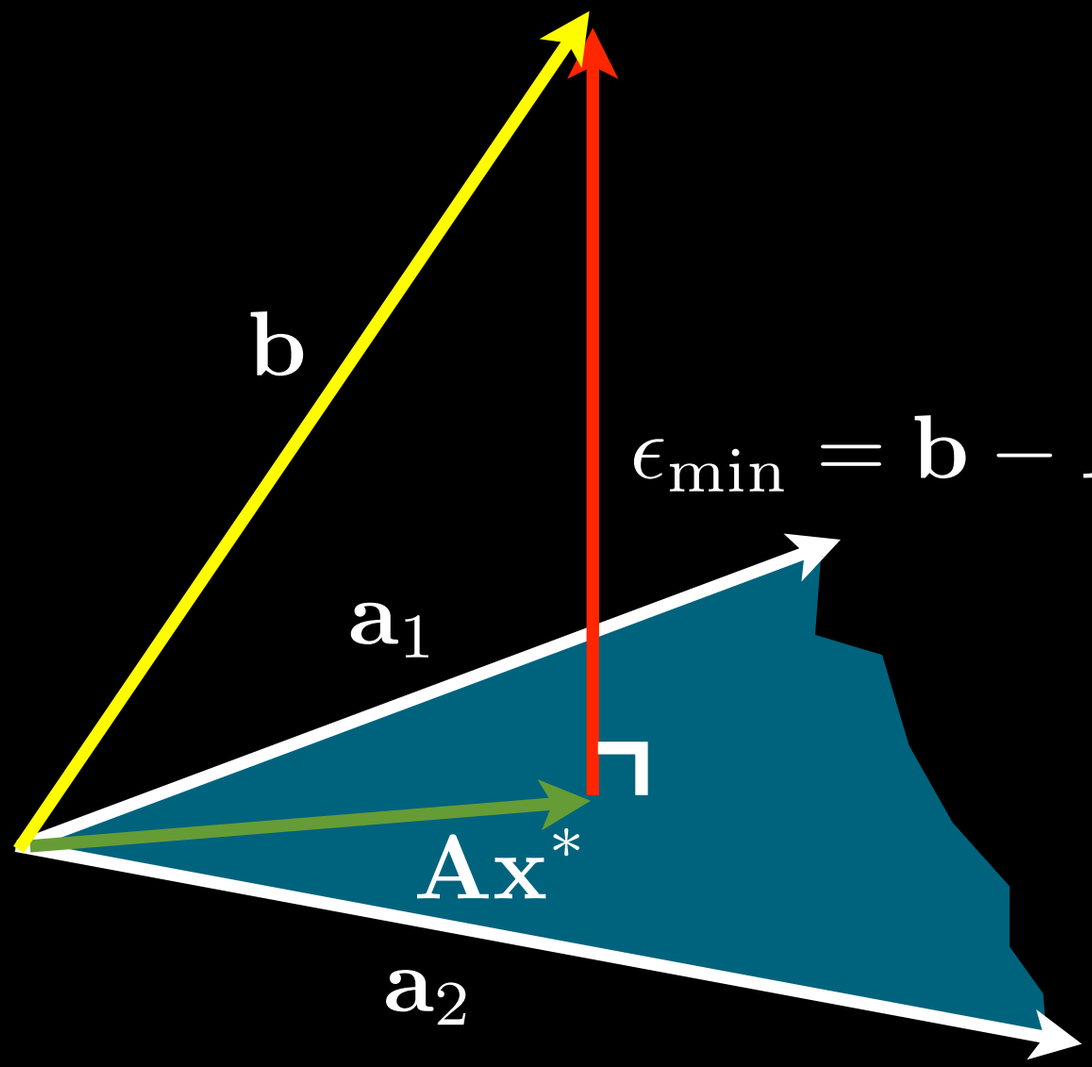
Does this look familiar?

Solve $\dfrac{\partial \epsilon}{\partial \mathbf{x}} = \mathbf{0}$

$$\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \mathbf{b}$$ **Normal equation**

Least Squares Solution $\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$

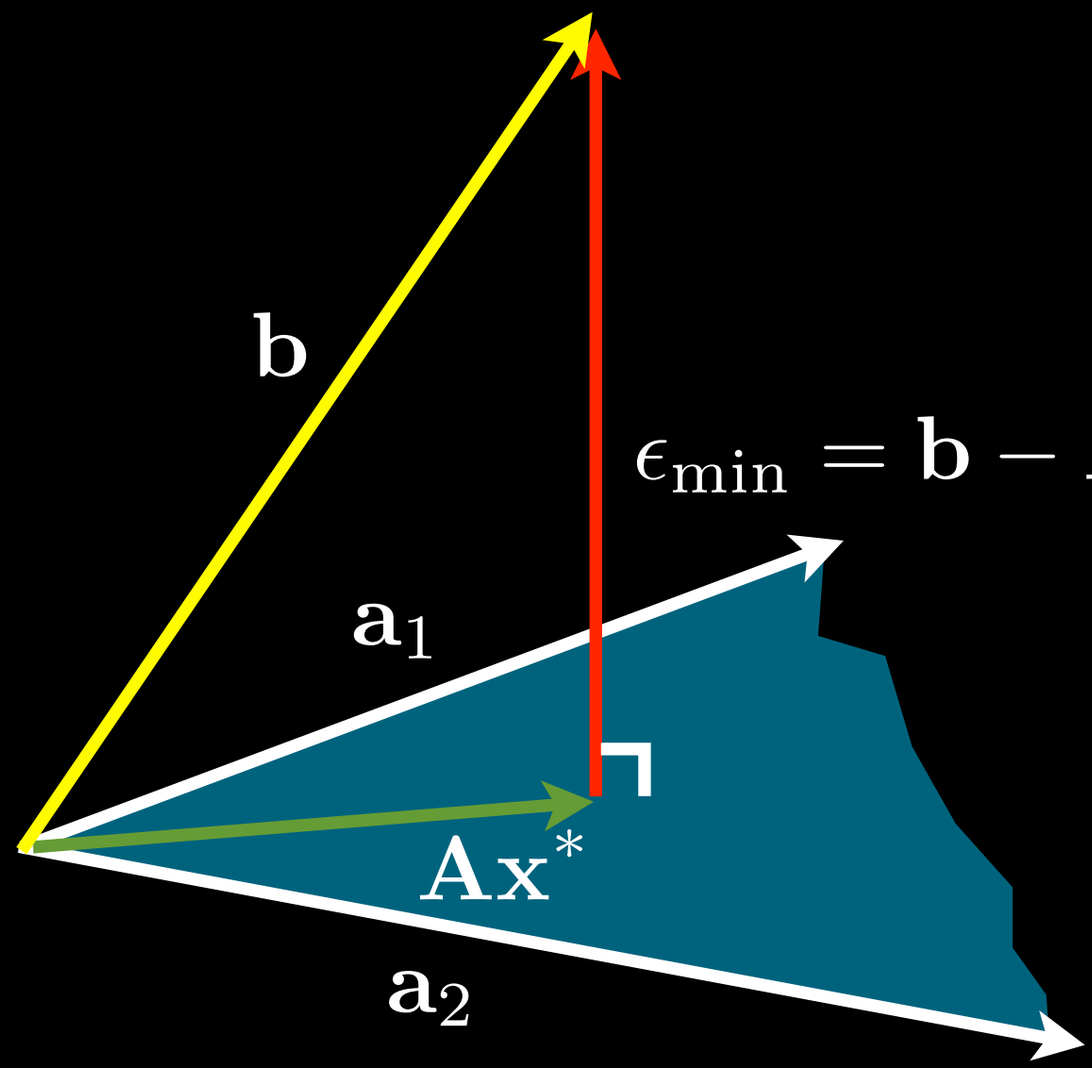
Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



$$\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$$

Does this look familiar?

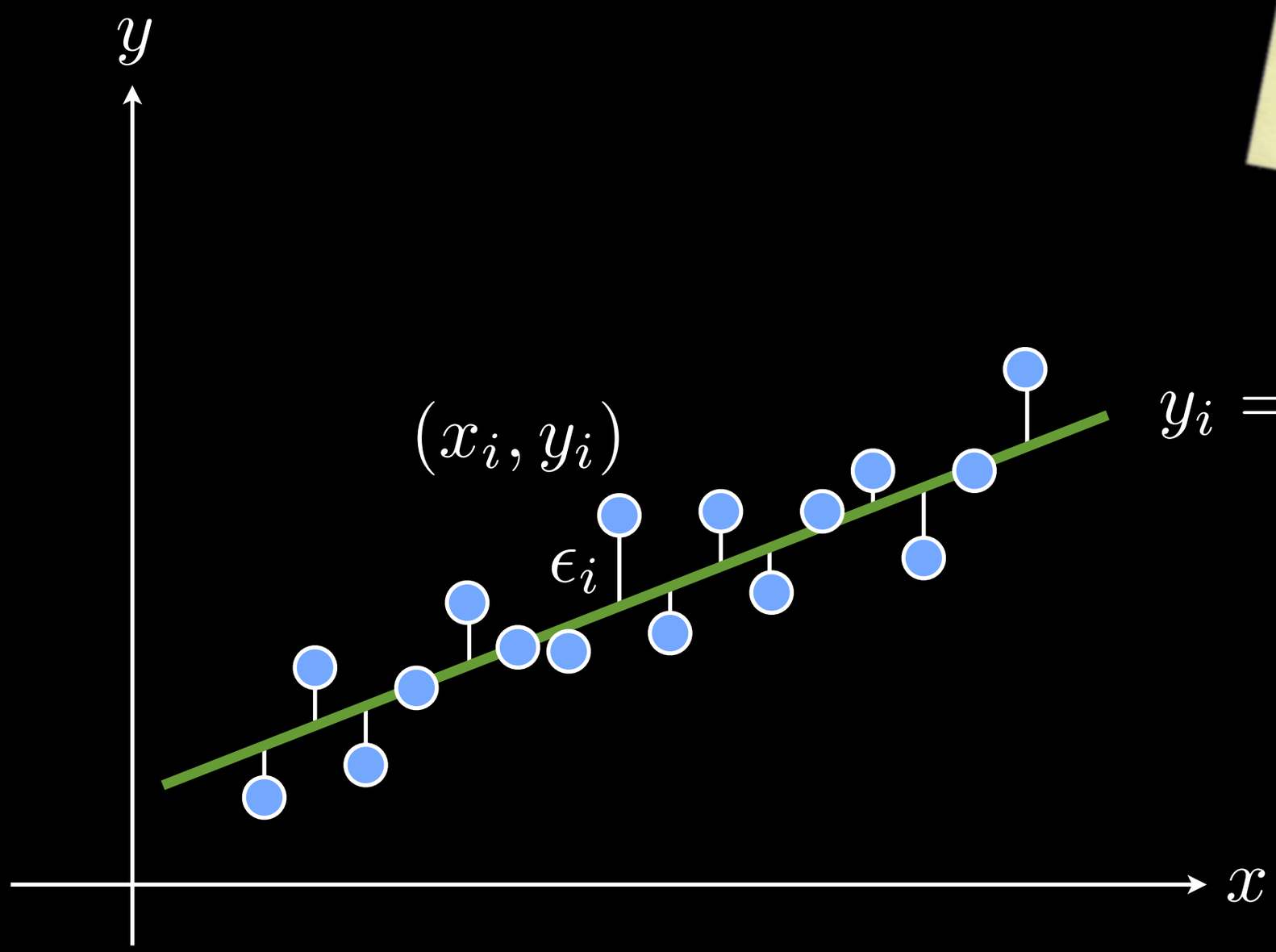
Goal: Make $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$ as small as possible



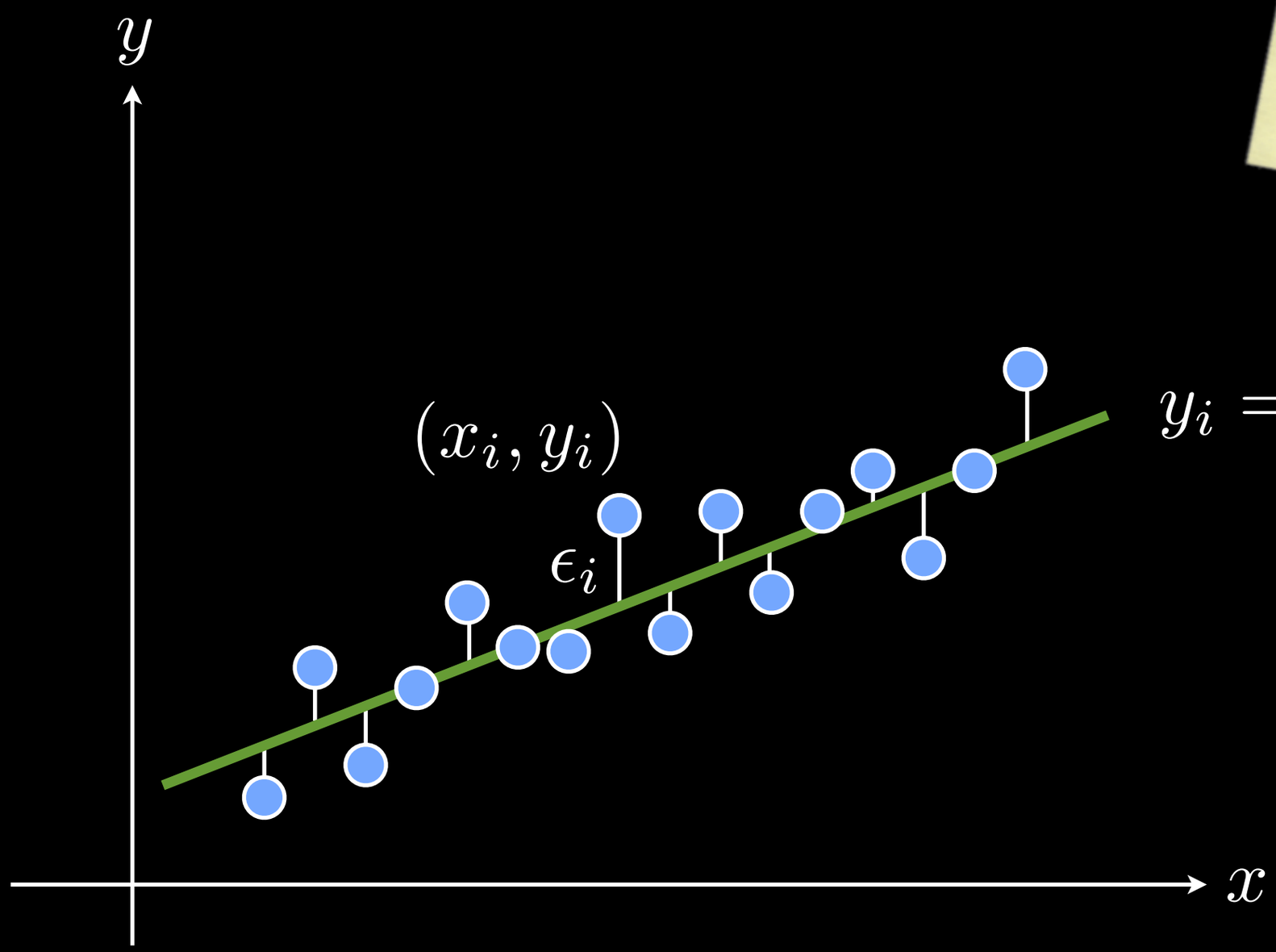
Least Squares Solution

$$\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$$

Issues

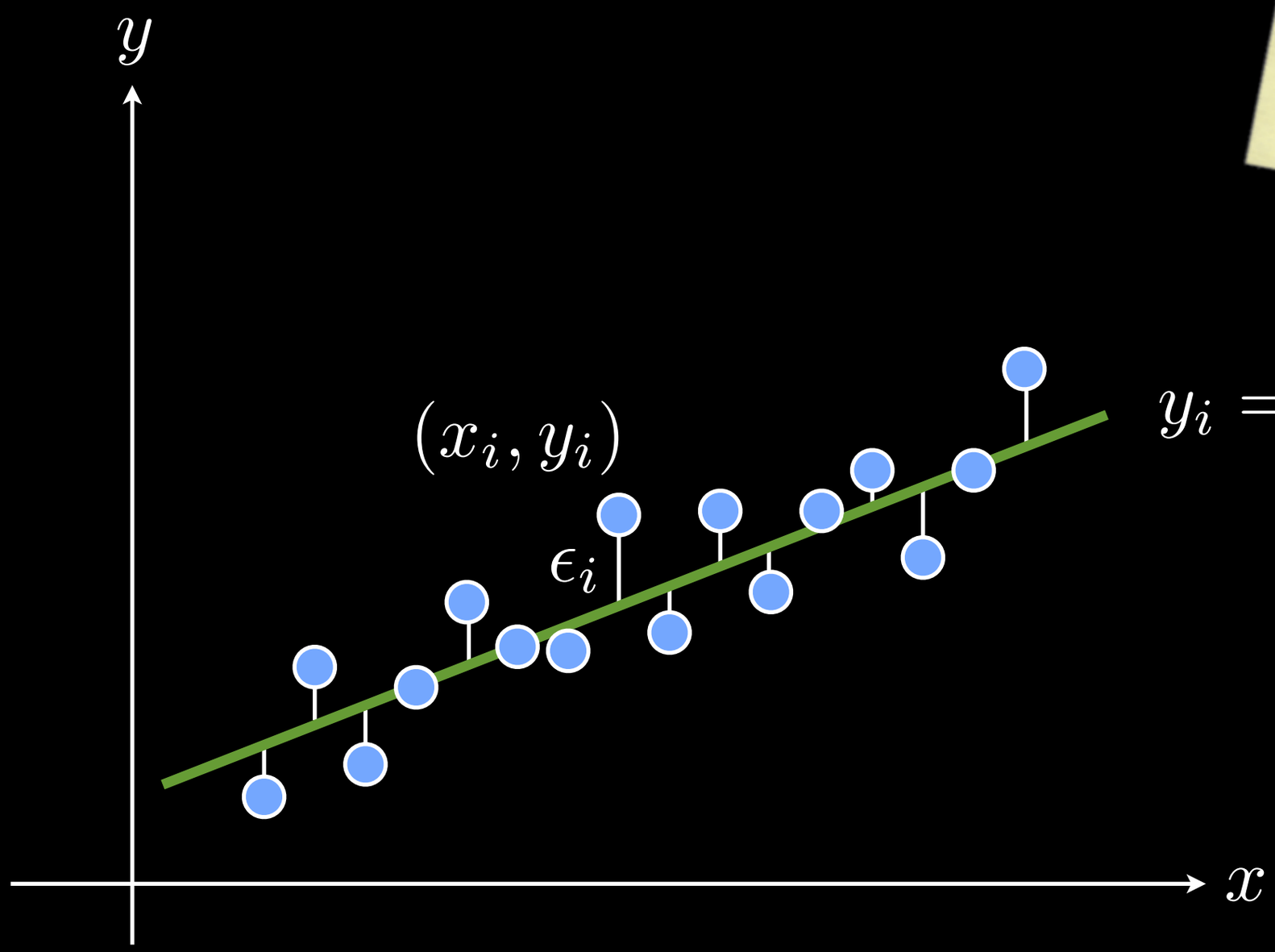


Issues



Not rotation invariant

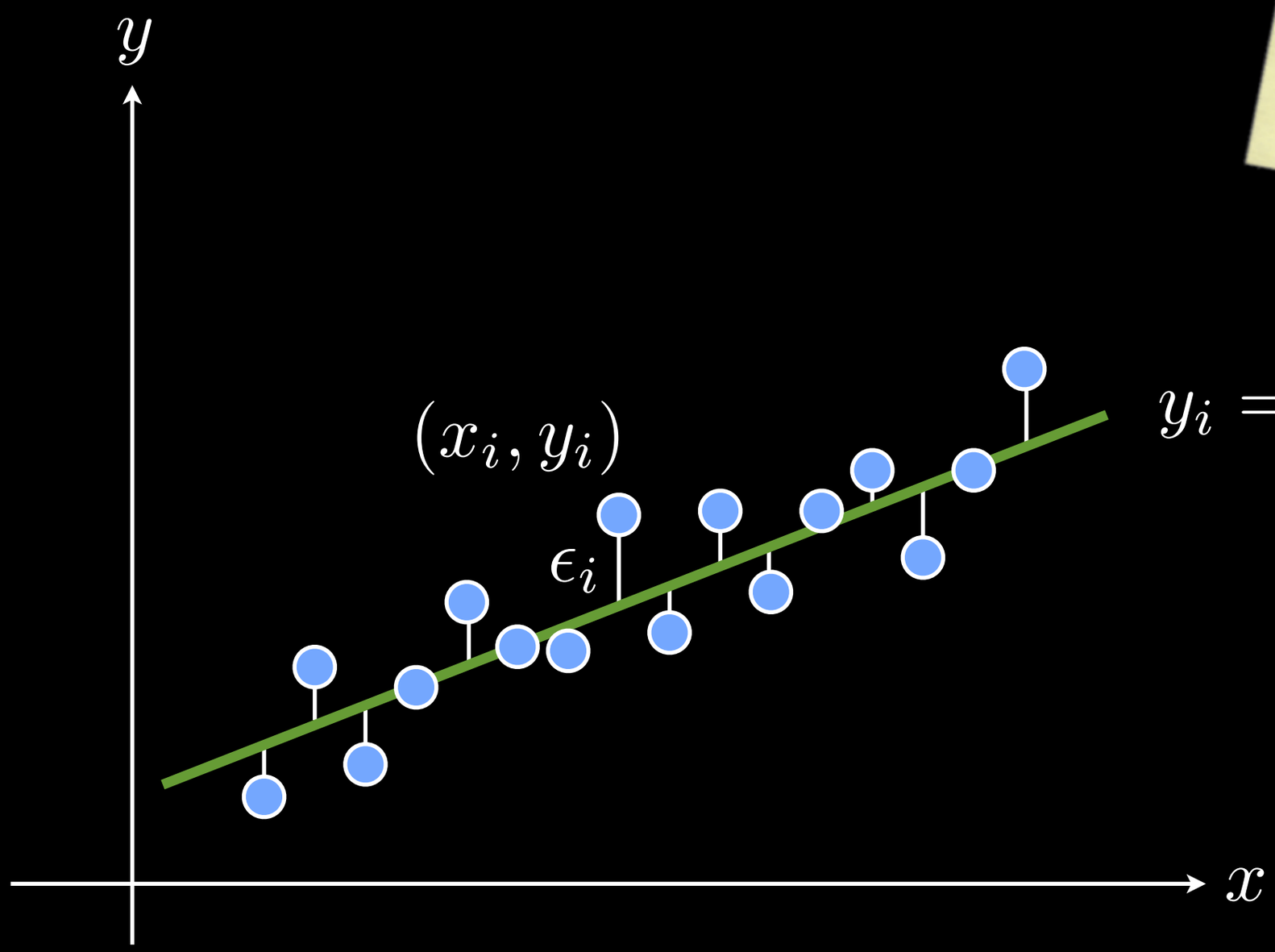
Issues



Not rotation invariant

Fails completely for vertical lines

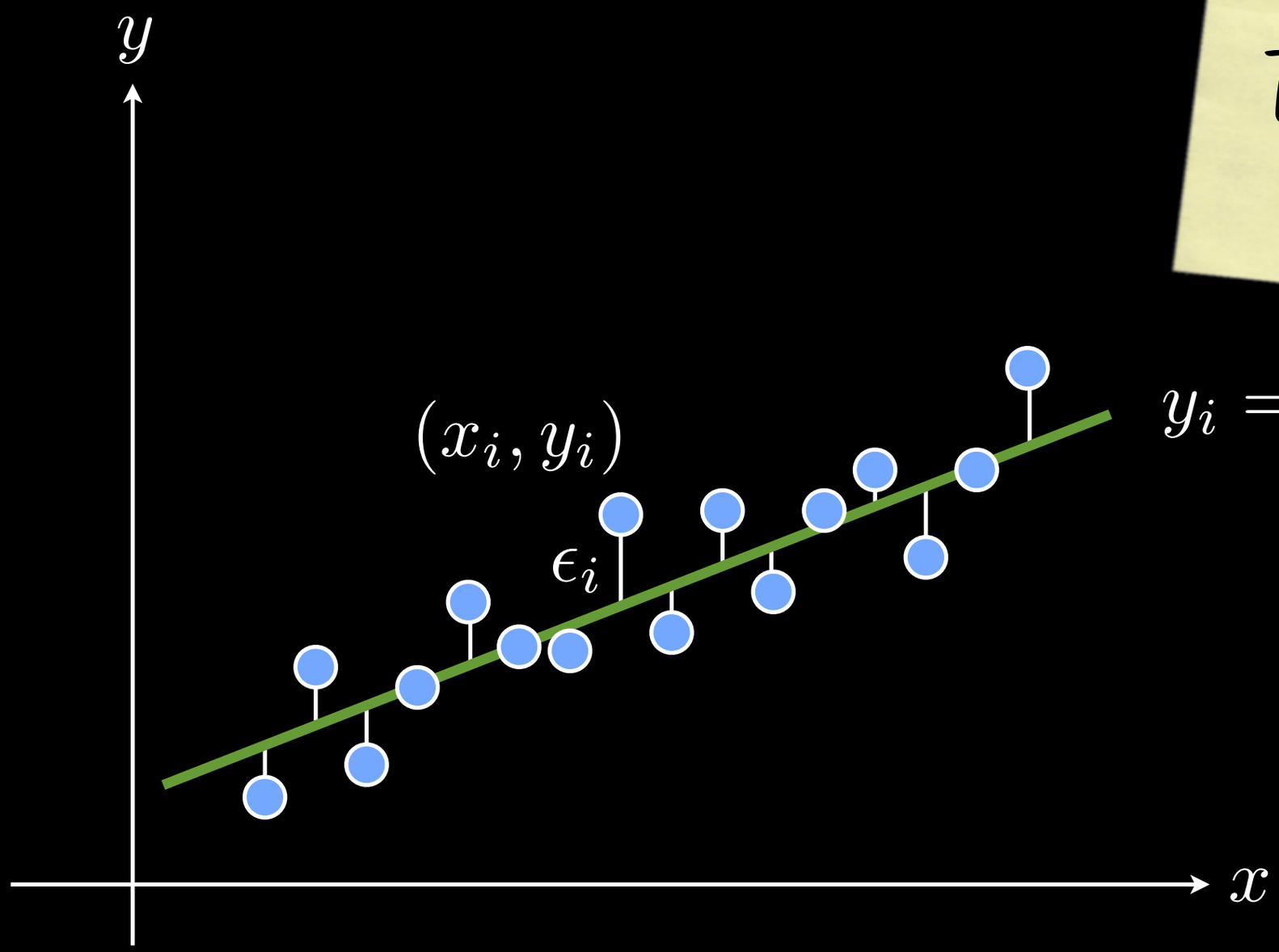
Issues



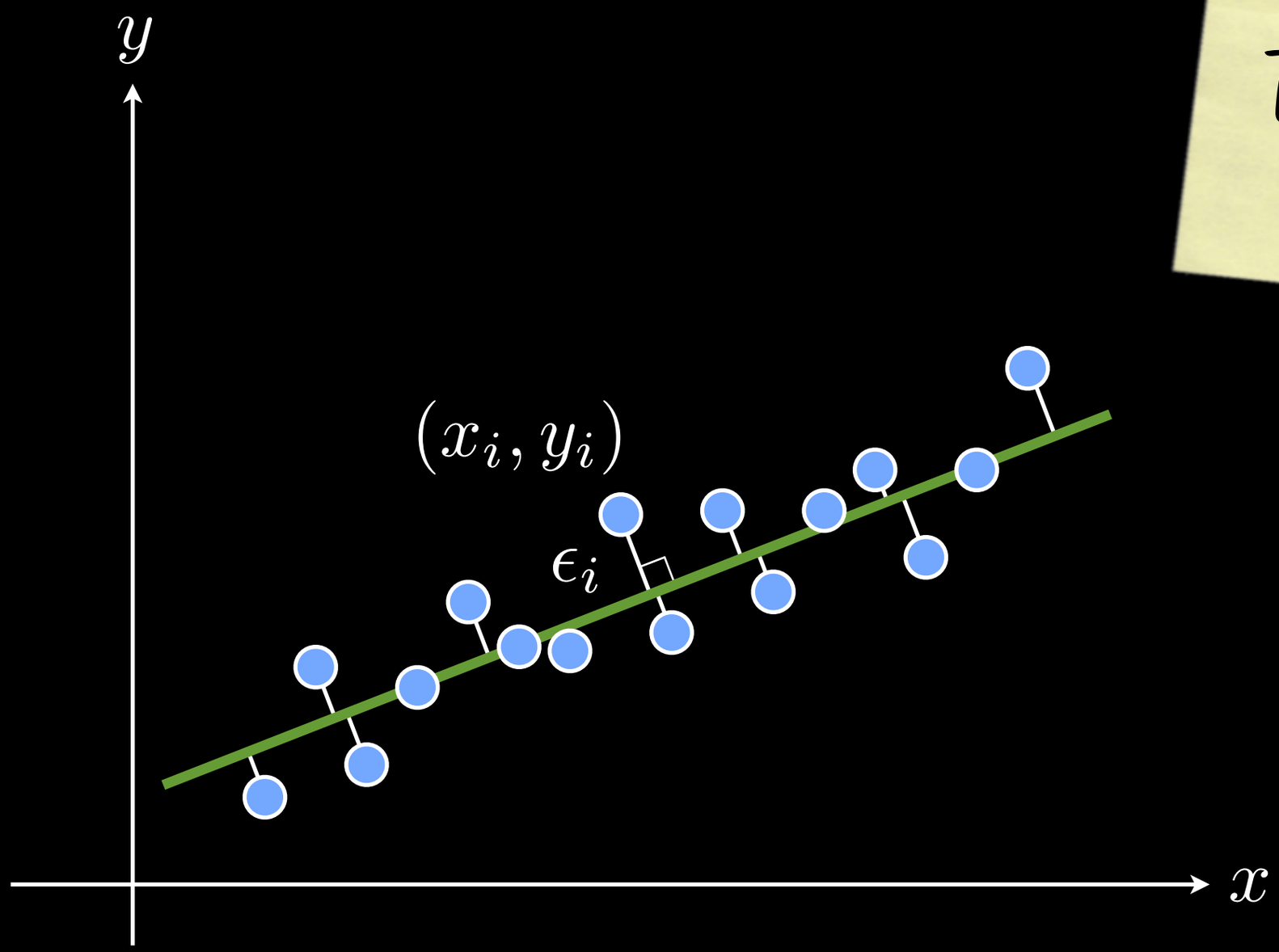
Not rotation invariant

Fails completely for vertical lines

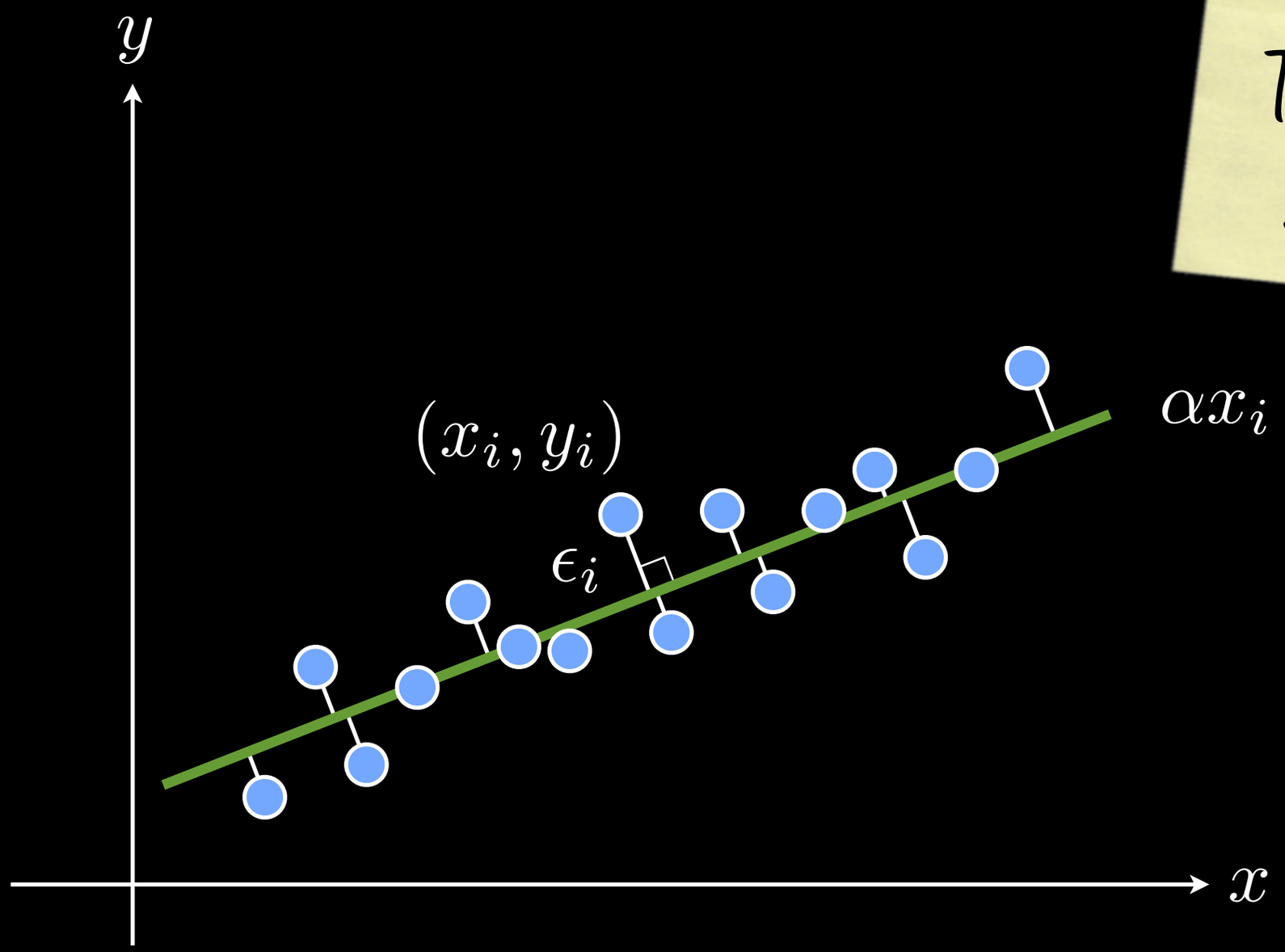
Total-least Squares



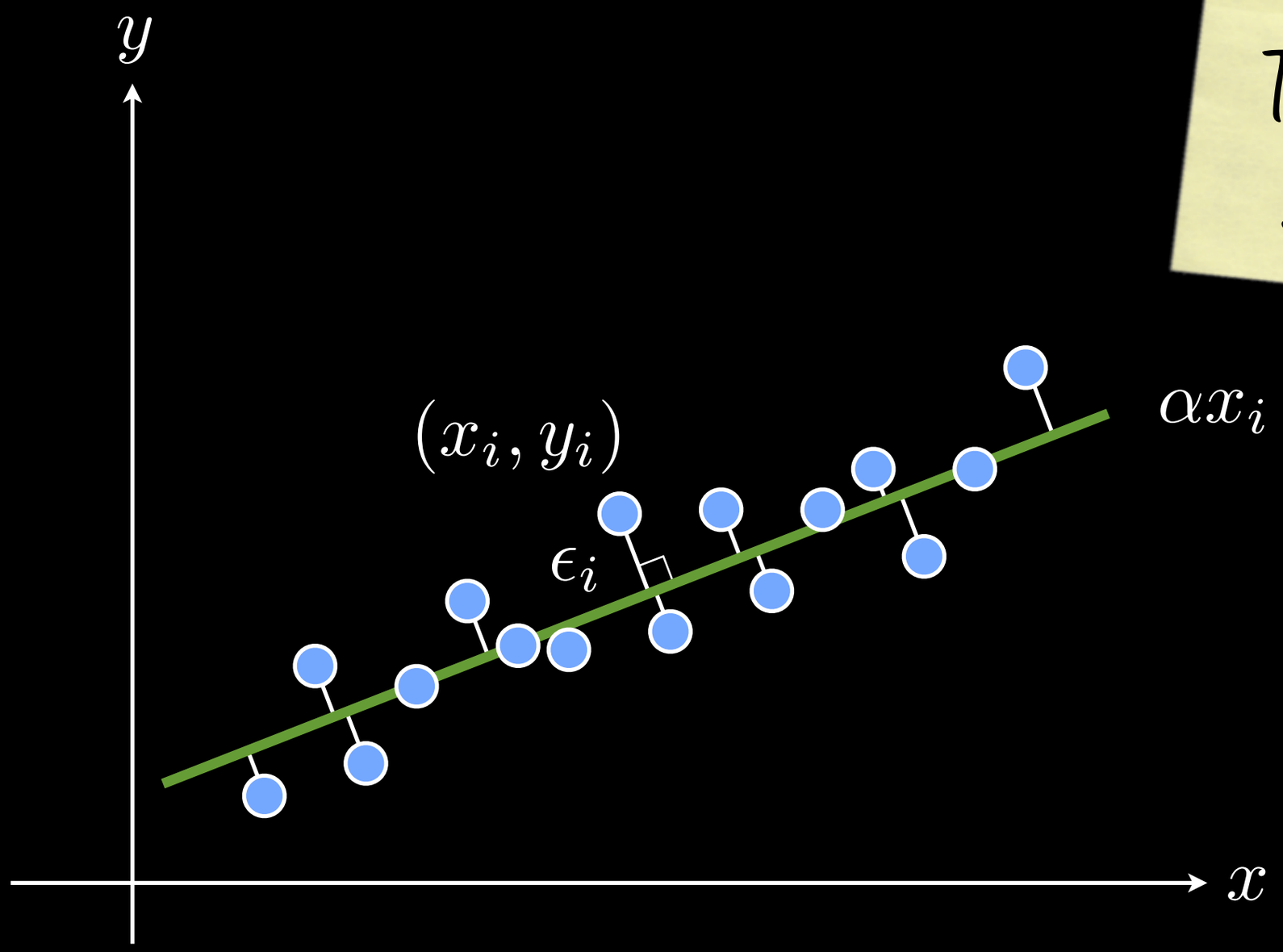
Total-least Squares



Total-least Squares



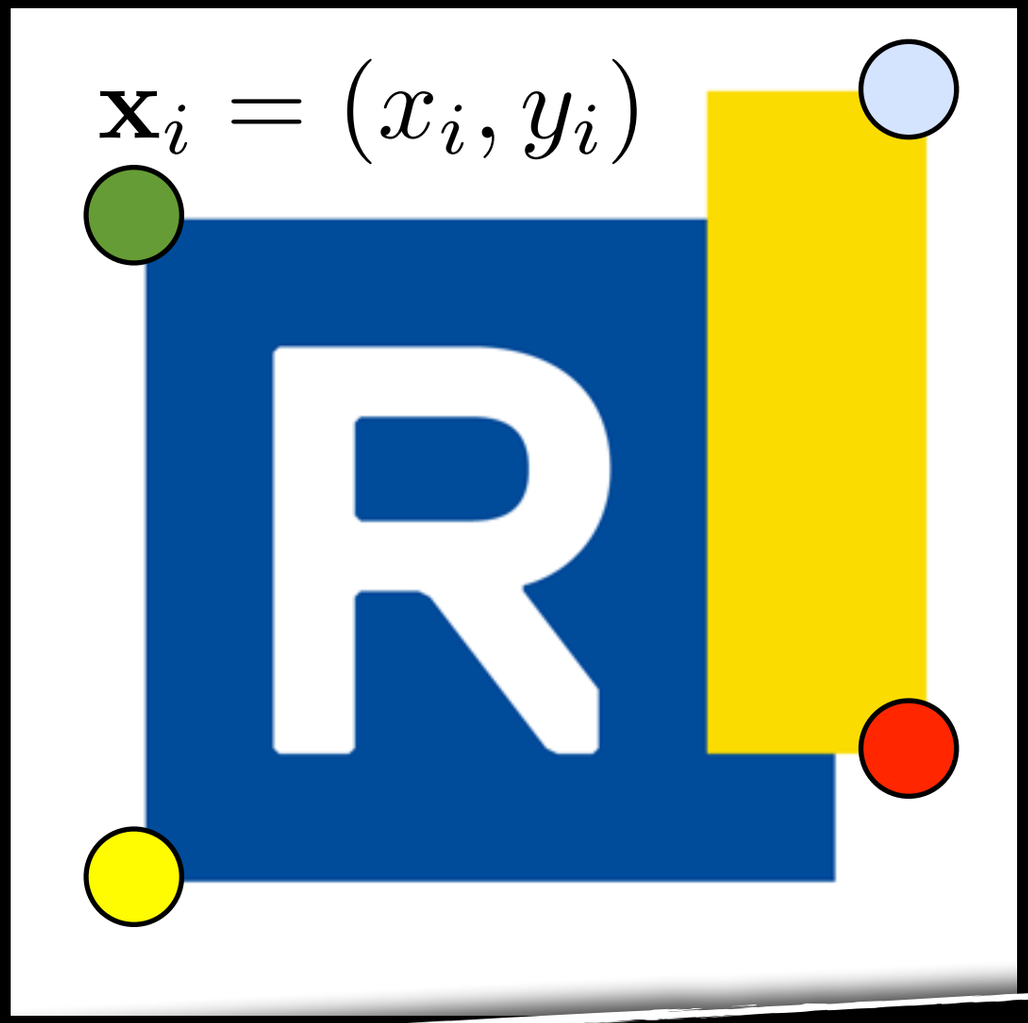
Total-least Squares



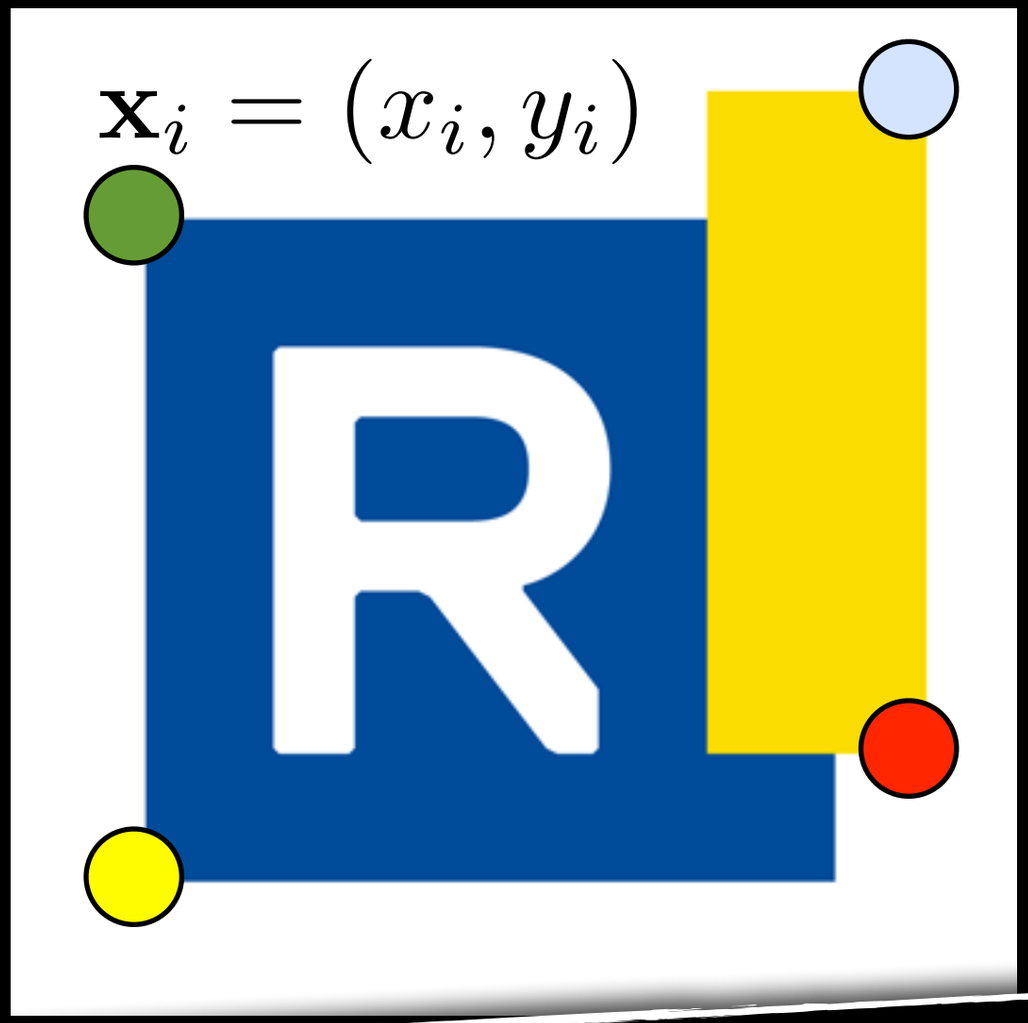
$$\arg\min_{(\alpha,\beta,\gamma)} \sum_{i=1}^{N} (\alpha x_i + \beta y_i - \gamma)^2 \quad \text{subject to} \quad \|(\alpha, \beta)^\top\| = 1$$

What about estimating geometric transformations?

Given correspondences, how do we estimate mapping?

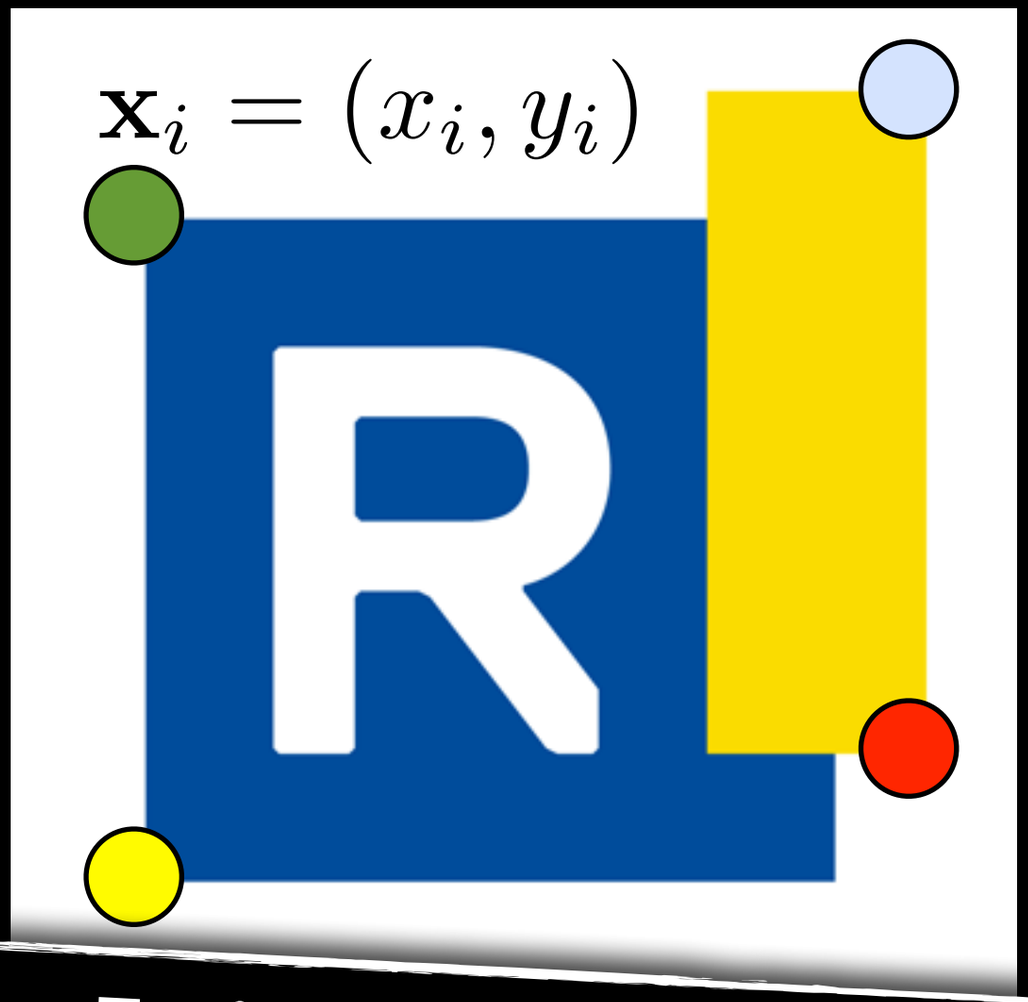


Given correspondences, how do we estimate mapping?



$\mathbf{x}'_i = \mathbf{T}\mathbf{x}_i + \mathbf{t}$

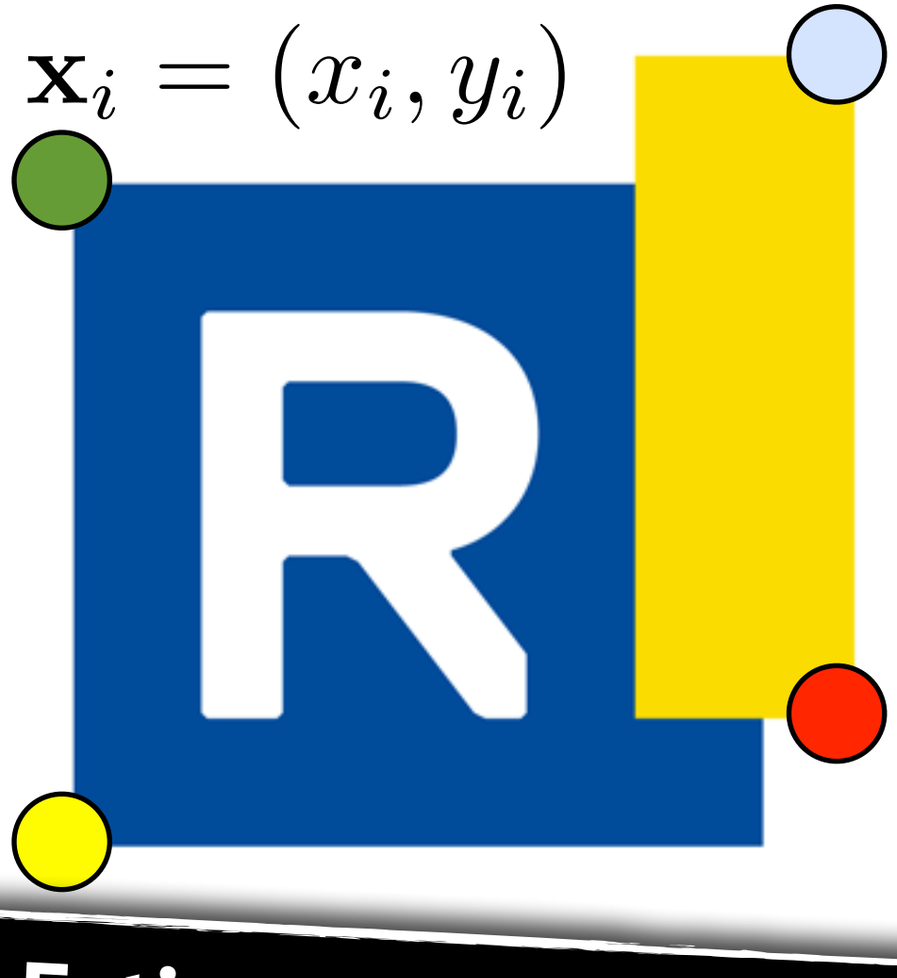
Given correspondences, how do we estimate mapping?



$\mathbf{x}'_i = \mathbf{T}\mathbf{x}_i + \mathbf{t}$

Estimate the mapping using least-squares fitting

$$\mathbf{x}_i = (x_i, y_i)$$



$$\mathbf{x}'_i = \mathbf{T}\mathbf{x}_i + \mathbf{t}$$

Estimate the mapping using least-squares fitting

$$\mathbf{T}^*, \mathbf{t}^* = \arg\min_{\mathbf{T},\mathbf{t}} \sum_{i=1}^{N} \|\mathbf{x}'_i - \mathbf{T}\mathbf{x}_i - \mathbf{t}\|^2$$

$\mathbf{x}_i = (x_i, y_i)$

$\mathbf{x}'_i = \mathbf{T}\mathbf{x}_i + \mathbf{t}$

Estimate the mapping using least-squares fitting

$$\mathbf{T}^*, \mathbf{t}^* = \arg\min_{\mathbf{T},\mathbf{t}} \sum_{i=1}^{N} \|\mathbf{x}'_i - \mathbf{T}\mathbf{x}_i - \mathbf{t}\|^2$$

Set up normal equation and solve for unknowns

Least-squares with translation

$$\begin{pmatrix} x_i' \\ y_i' \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Least-squares with translation

$$\begin{pmatrix} x_i' \\ y_i' \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Rewrite as a linear transformation of the unknowns

Least-squares with translation

$$\begin{pmatrix} x'_i \\ y'_i \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Rewrite as a linear transformation of the unknowns

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ & \vdots \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \begin{pmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ \vdots \\ x'_N - x_N \\ y'_N - y_N \end{pmatrix}$$

Least-squares with translation

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ & \vdots \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \begin{pmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ \vdots \\ x'_N - x_N \\ y'_N - y_N \end{pmatrix}$$

Least-squares with translation

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ & \vdots \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \begin{pmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ \vdots \\ x'_N - x_N \\ y'_N - y_N \end{pmatrix}$$

What is the minimum number of points to solve?

Least-squares with translation

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ \vdots & \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \begin{pmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ \vdots \\ x'_N - x_N \\ y'_N - y_N \end{pmatrix}$$

Set up normal equation and solve for unknowns

Least-squares with translation

$$\underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ \vdots & \\ 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{A}} \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \begin{pmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ \vdots \\ x'_N - x_N \\ y'_N - y_N \end{pmatrix}$$

Set up normal equation and solve for unknowns

Least-squares with translation

$$\underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ \vdots & \\ 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} t_x \\ t_y \end{pmatrix}}_{\mathbf{x}} = \begin{pmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ \vdots \\ x'_N - x_N \\ y'_N - y_N \end{pmatrix}$$

Set up normal equation and solve for unknowns

Least-squares with translation

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ & \vdots \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \begin{pmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ \vdots \\ x'_N - x_N \\ y'_N - y_N \end{pmatrix}$$

$$\mathbf{A} \qquad \mathbf{x} \qquad \mathbf{b}$$

Set up normal equation and solve for unknowns

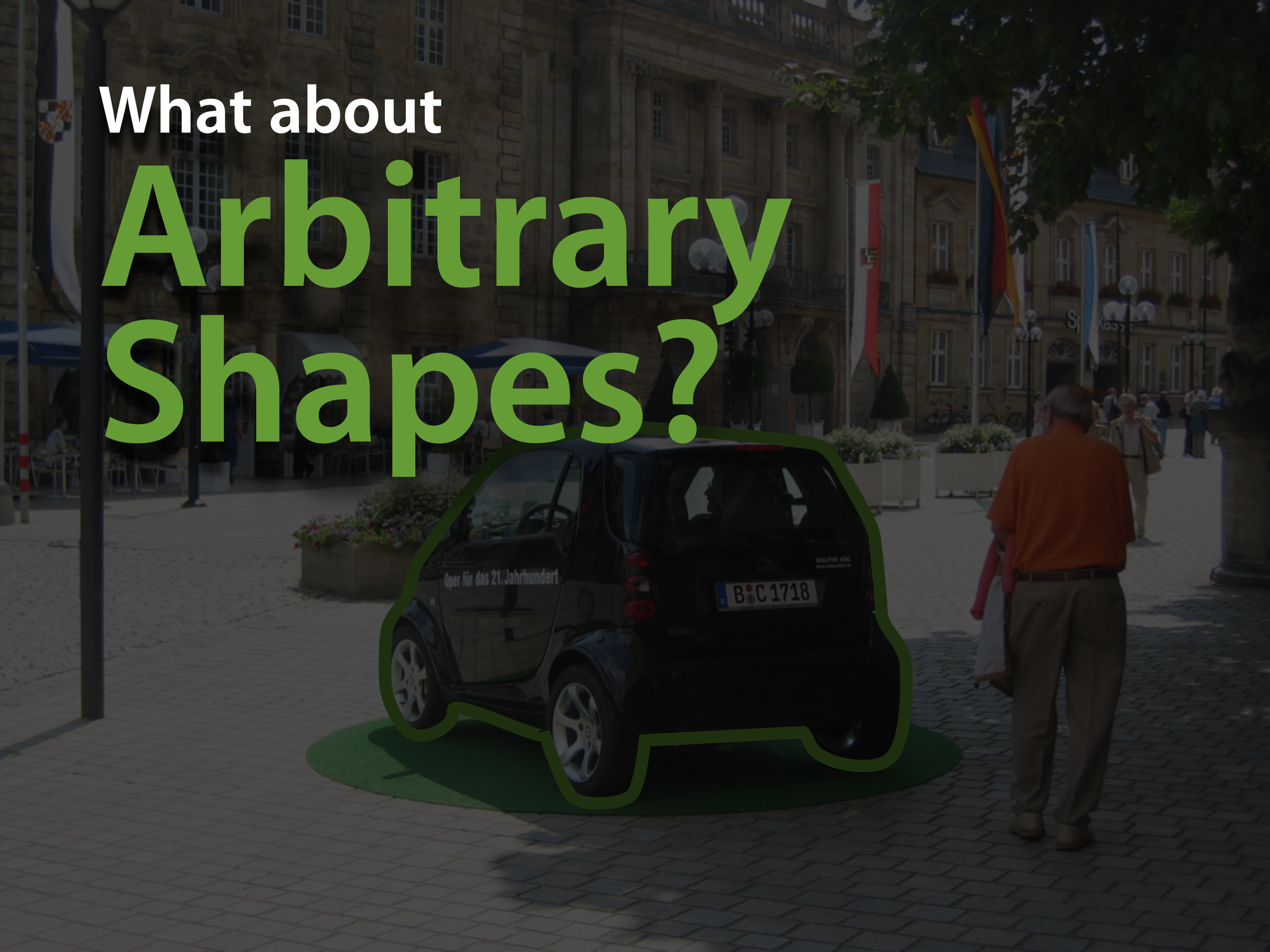
Least-squares with translation

$$\underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ \vdots & \\ 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} t_x \\ t_y \end{pmatrix}}_{\mathbf{x}} = \underbrace{\begin{pmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ \vdots \\ x'_N - x_N \\ y'_N - y_N \end{pmatrix}}_{\mathbf{b}}$$

Set up normal equation and solve for unknowns

$$\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$$

What about Arbitrary Shapes?



Pattern Recognition Vol. 13, No. 2, pp. 111–122, 1981.
Printed in Great Britain.

0031-3203/81/020111-12 $02.00/0
Pergamon Press Ltd.

GENERALIZING THE HOUGH TRANSFORM TO DETECT ARBITRARY SHAPES*

D. H. BALLARD
Computer Science Department, University of Rochester, Rochester, NY 14627, U.S.A.

(*Received* 10 *October* 1979; *in revised form* 9 *September* 1980; *received for publication* 23 *September* 1980)

Abstract– The Hough transform is a method for detecting curves by exploiting the duality between points on a curve and parameters of that curve. The initial work showed how to detect both analytic curves[1,2] and non-analytic curves,[3] but these methods were restricted to binary edge images. This work was generalized to the detection of some analytic curves in grey level images, specifically lines,[4] circles[5] and parabolas.[6] The line detection case is the best known of these and has been ingeniously exploited in several applications.[7,8,9]

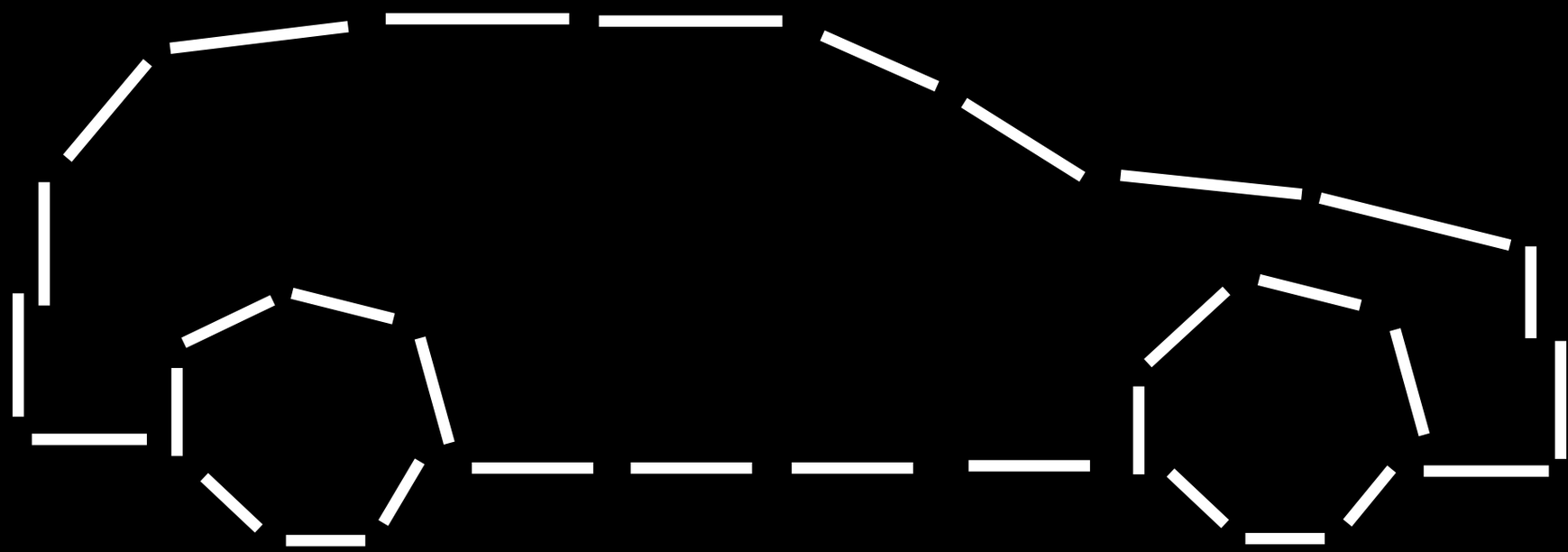
We show how the boundaries of an *arbitrary* non-analytic shape can be used to construct a mapping between image space and Hough transform space. Such a mapping can be exploited to detect instances of that particular shape in an image. Furthermore, variations in the shape such as rotations, scale changes or figure–ground reversals correspond to straightforward transformations of this mapping. However, the most remarkable property is that such mappings can be composed to build mappings for complex shapes from the mappings of simpler component shapes. This makes the generalized Hough transform a kind of universal transform which can be used to find arbitrarily complex shapes.

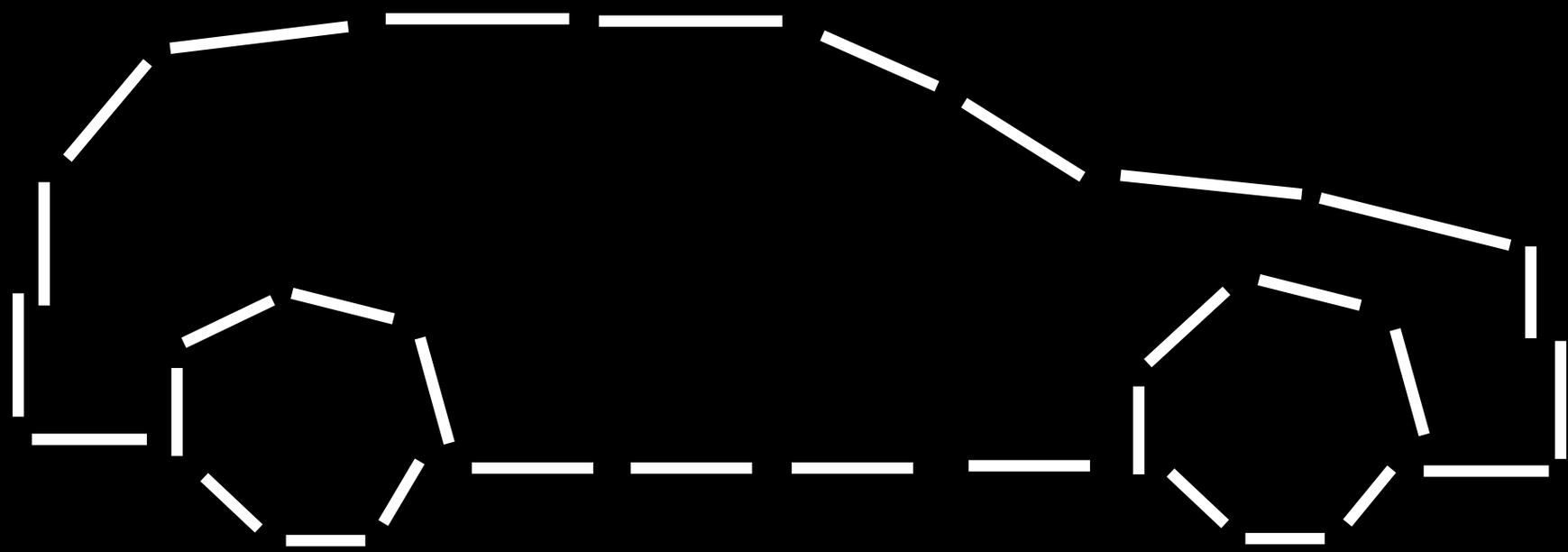
2

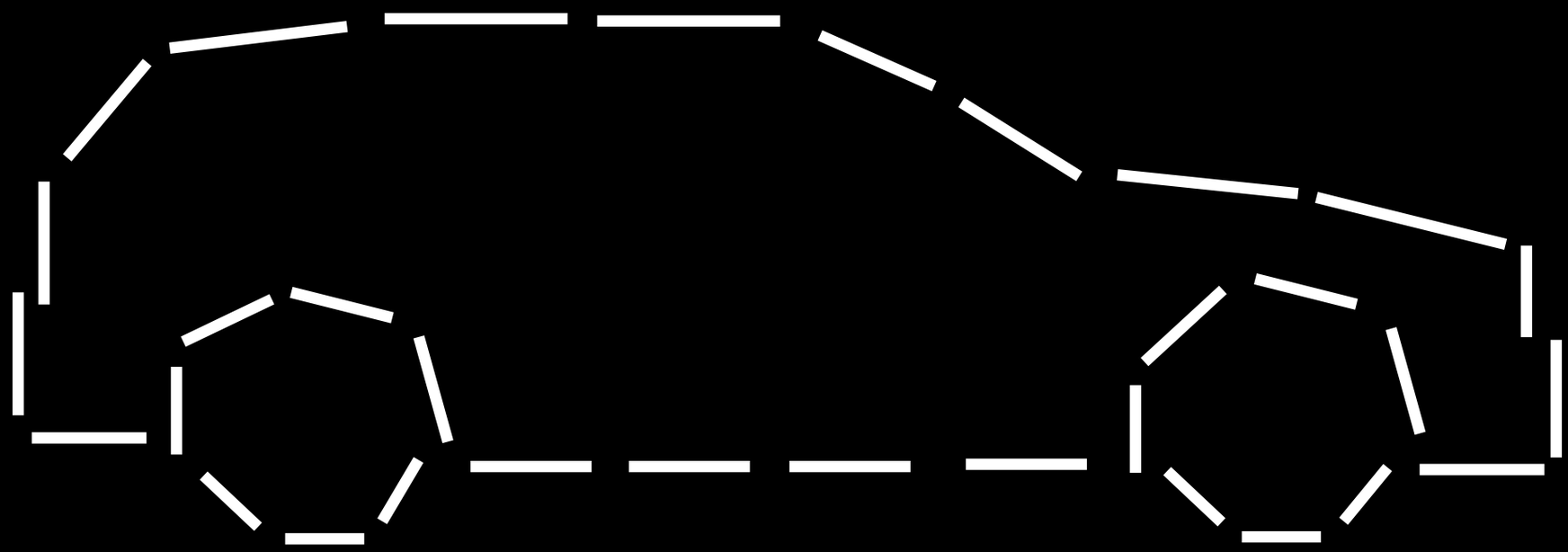
major steps

Step 1

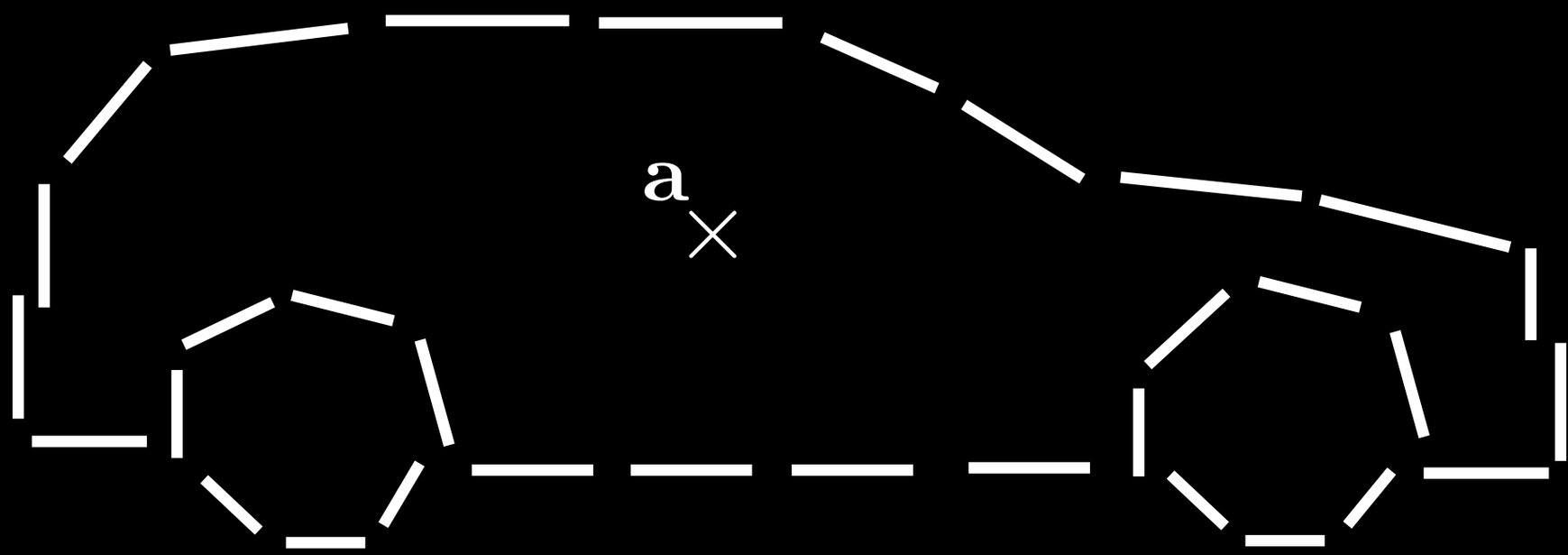
Build shape model



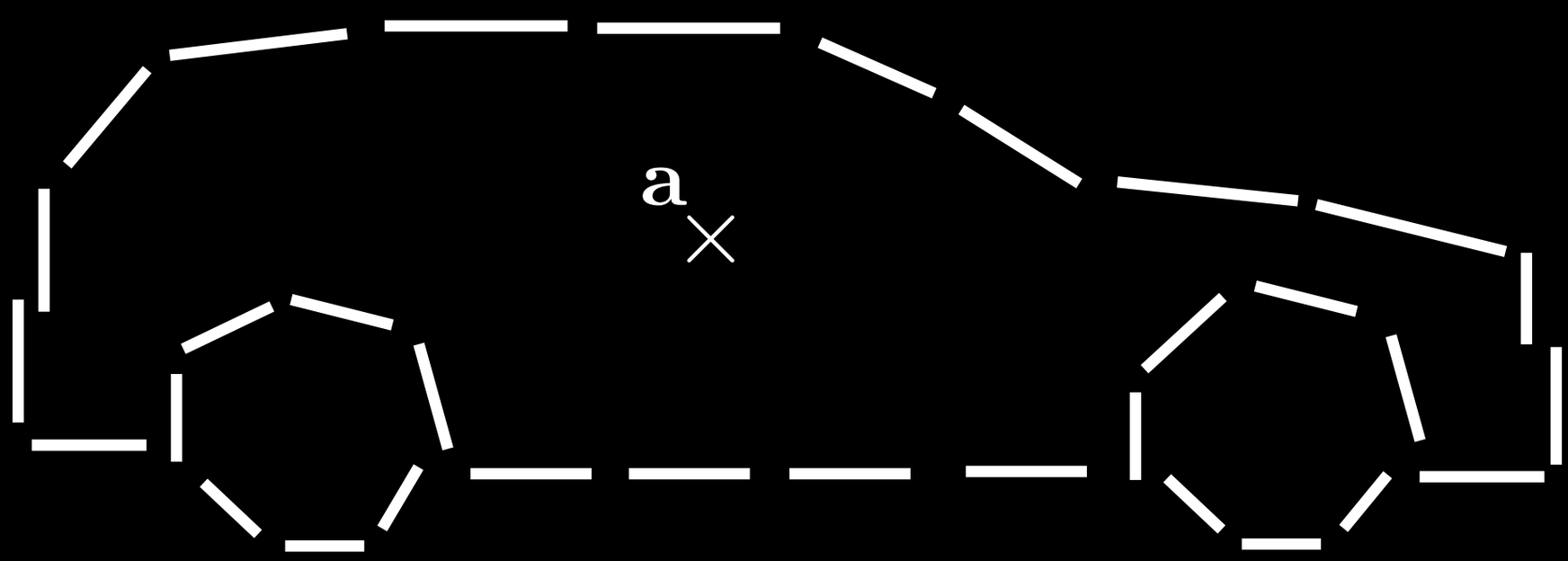




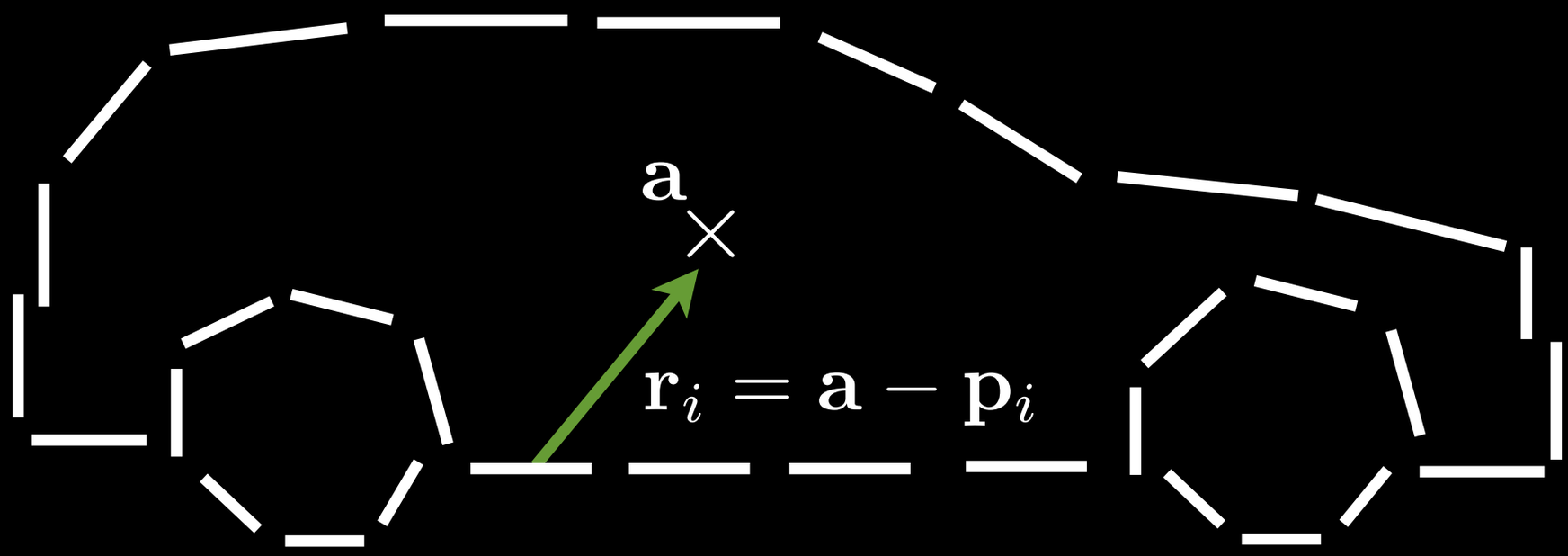
Define a reference point, a



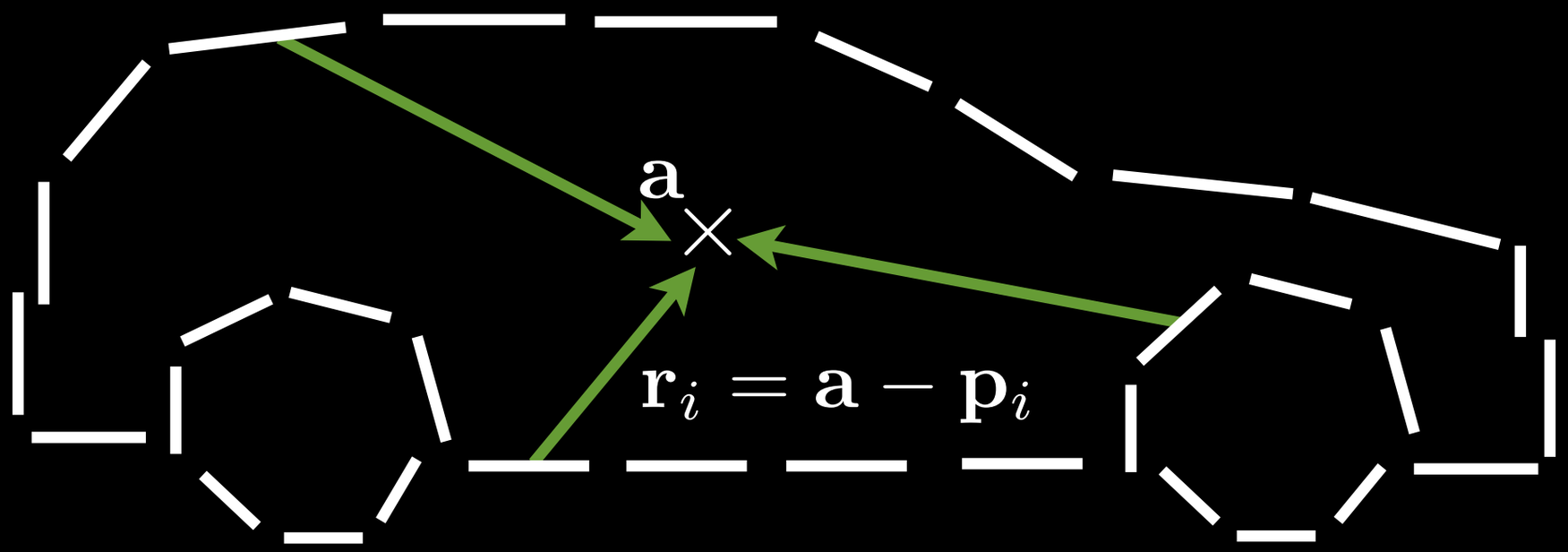
Define a reference point, a



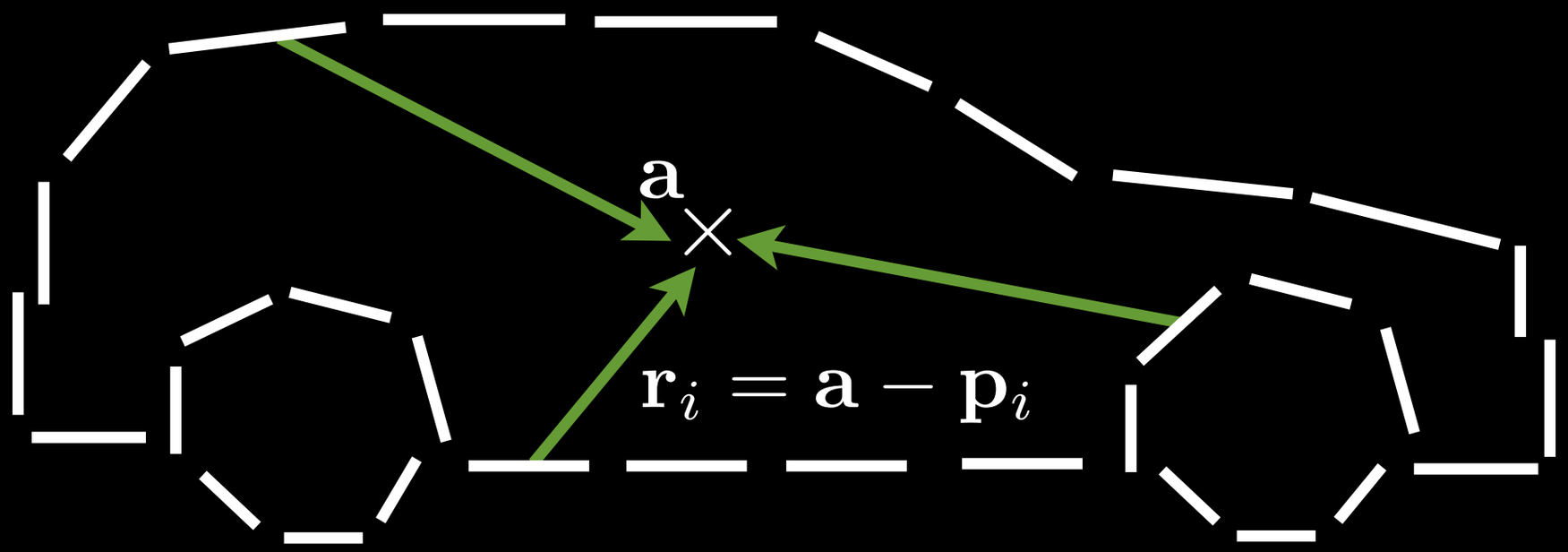
At each boundary point, compute the displacement vector $\mathrm{r}_i = \mathrm{a} - \mathrm{p}_i$



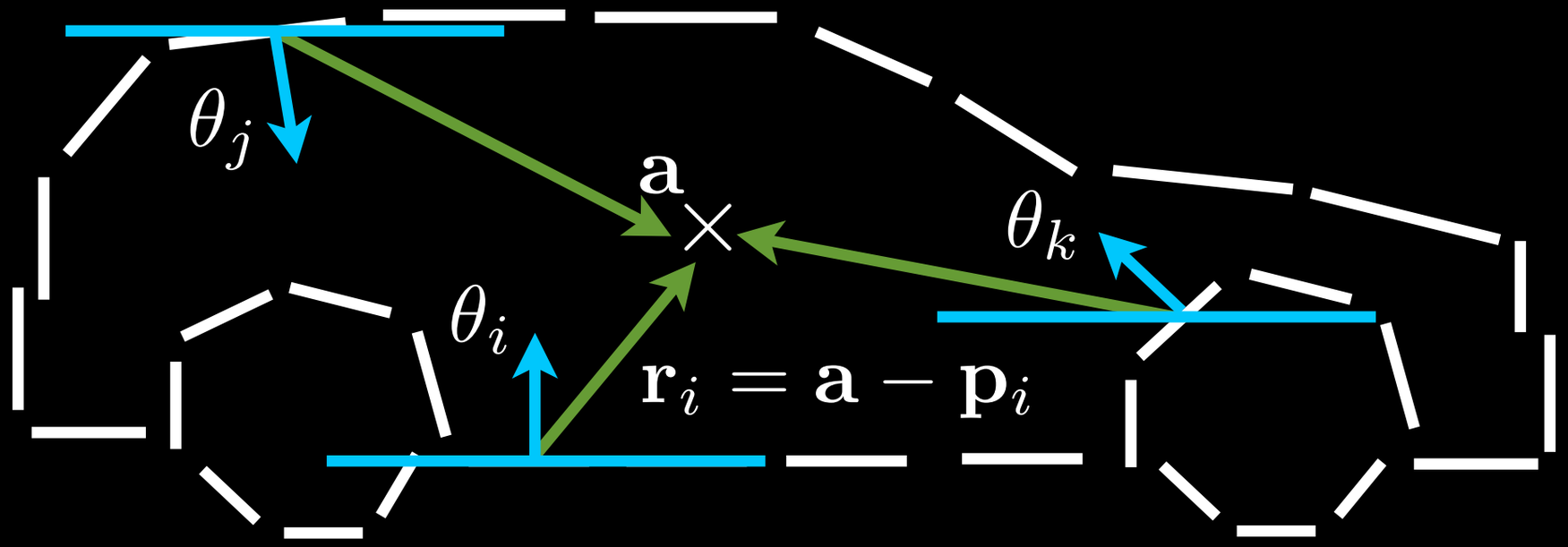
At each boundary point, compute the displacement vector $\mathbf{r}_i = \mathbf{a} - \mathbf{p}_i$



At each boundary point, compute the displacement vector $\mathbf{r}_i = \mathbf{a} - \mathbf{p}_i$



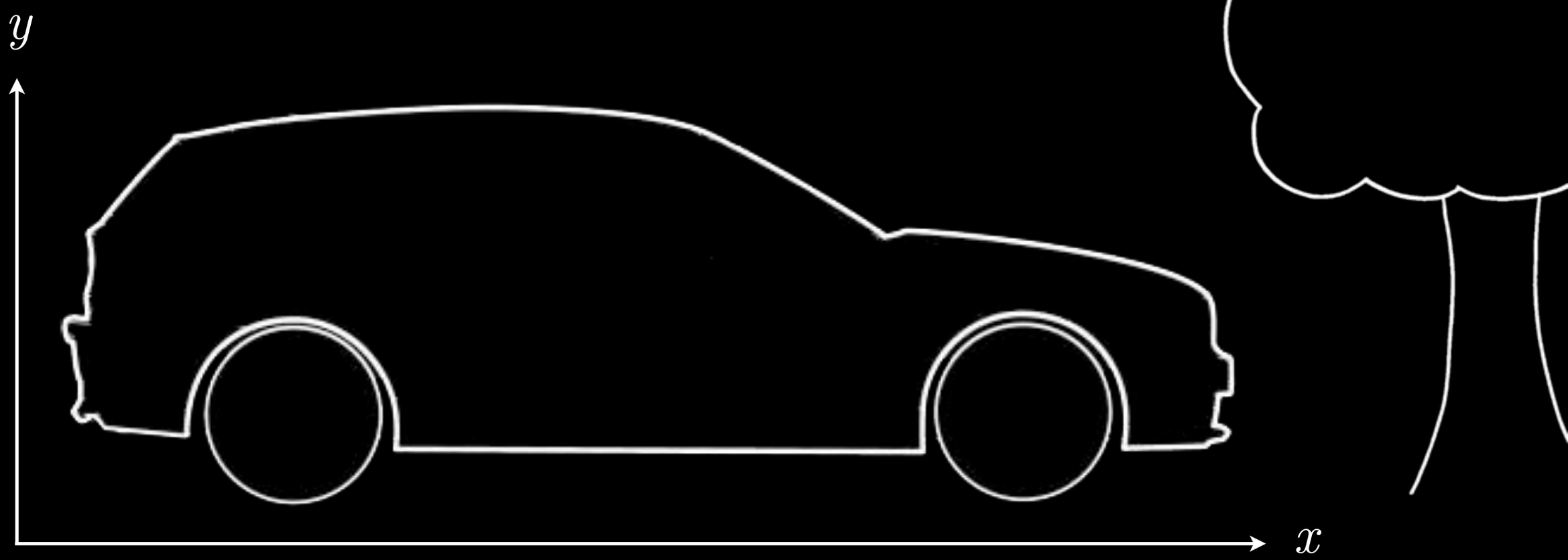
Store displacement vectors in a table indexed by the gradient orientation, θ

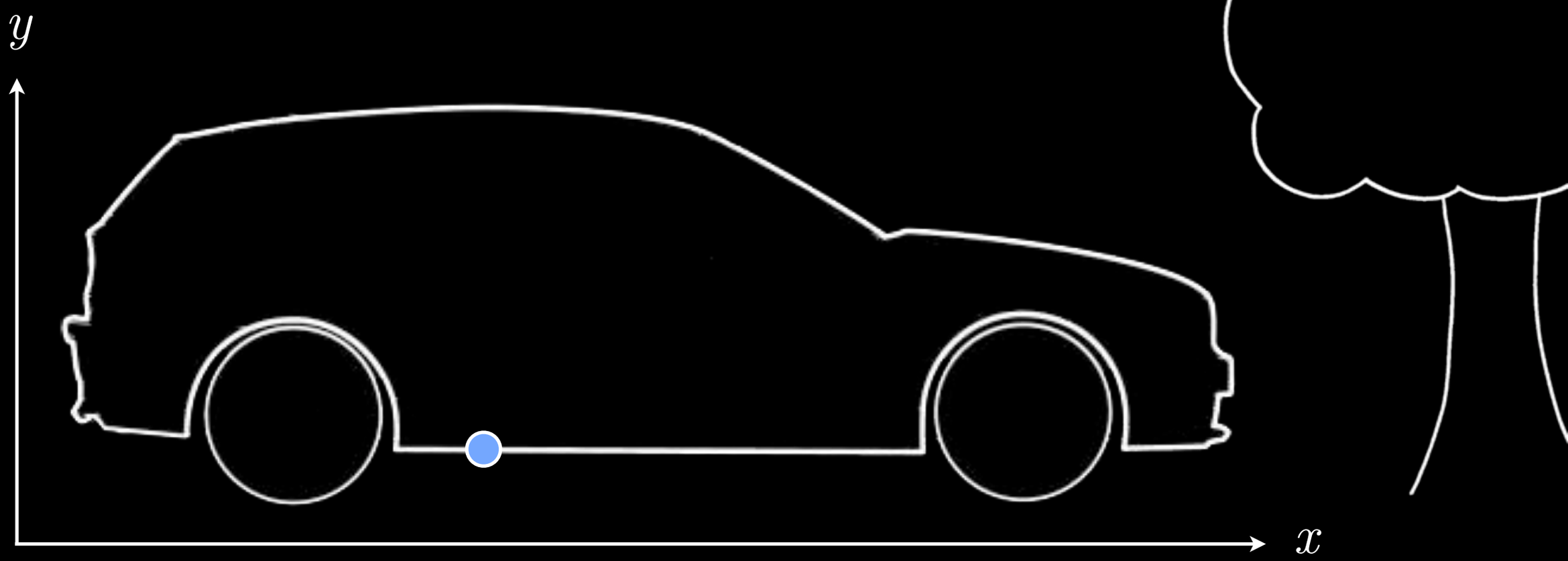


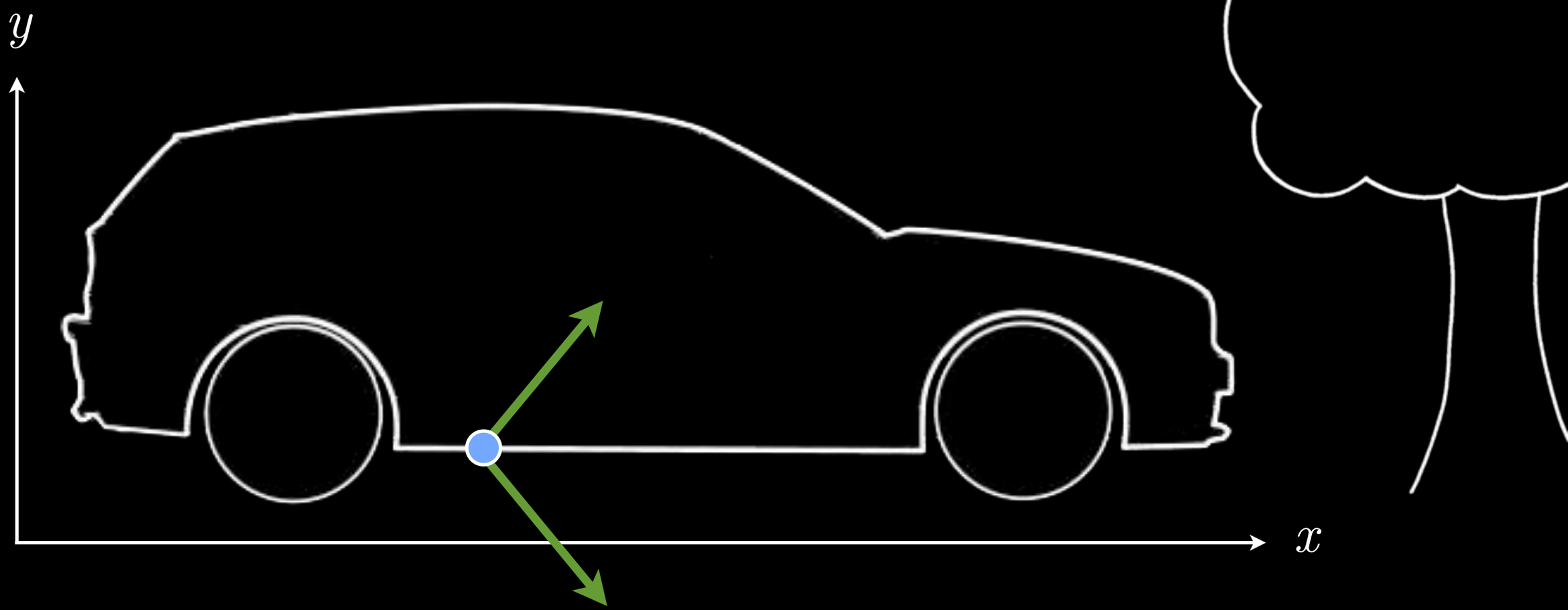
Store displacement vectors in a table indexed by the gradient orientation, θ

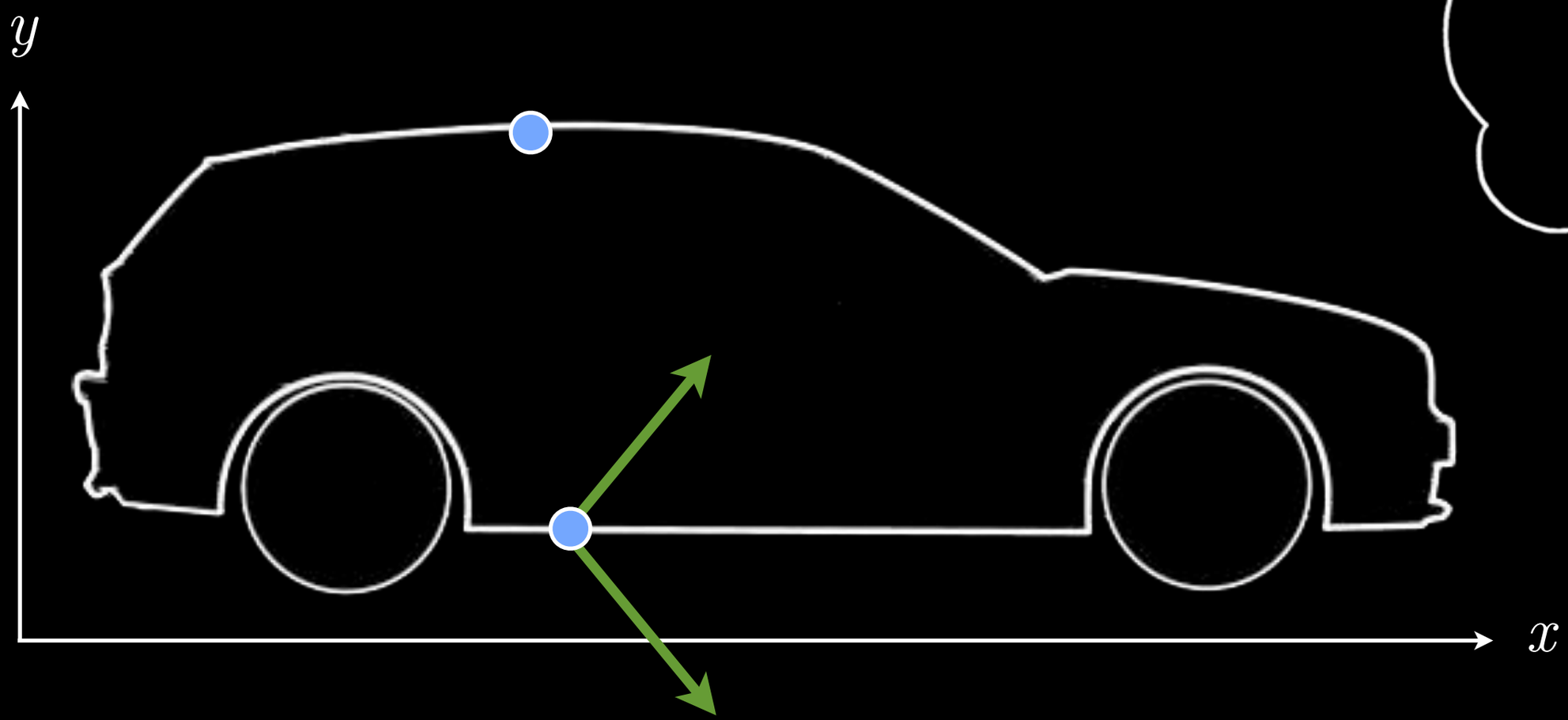
Step 2

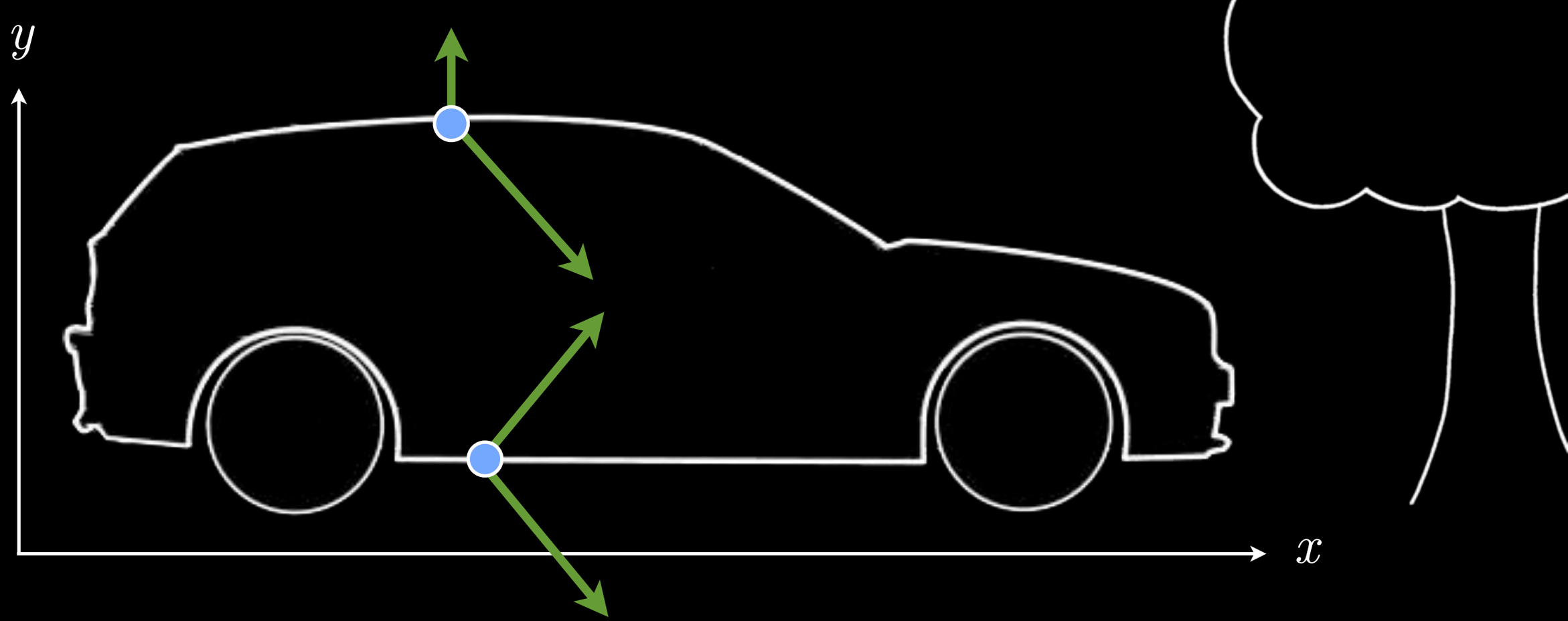
Shape detection

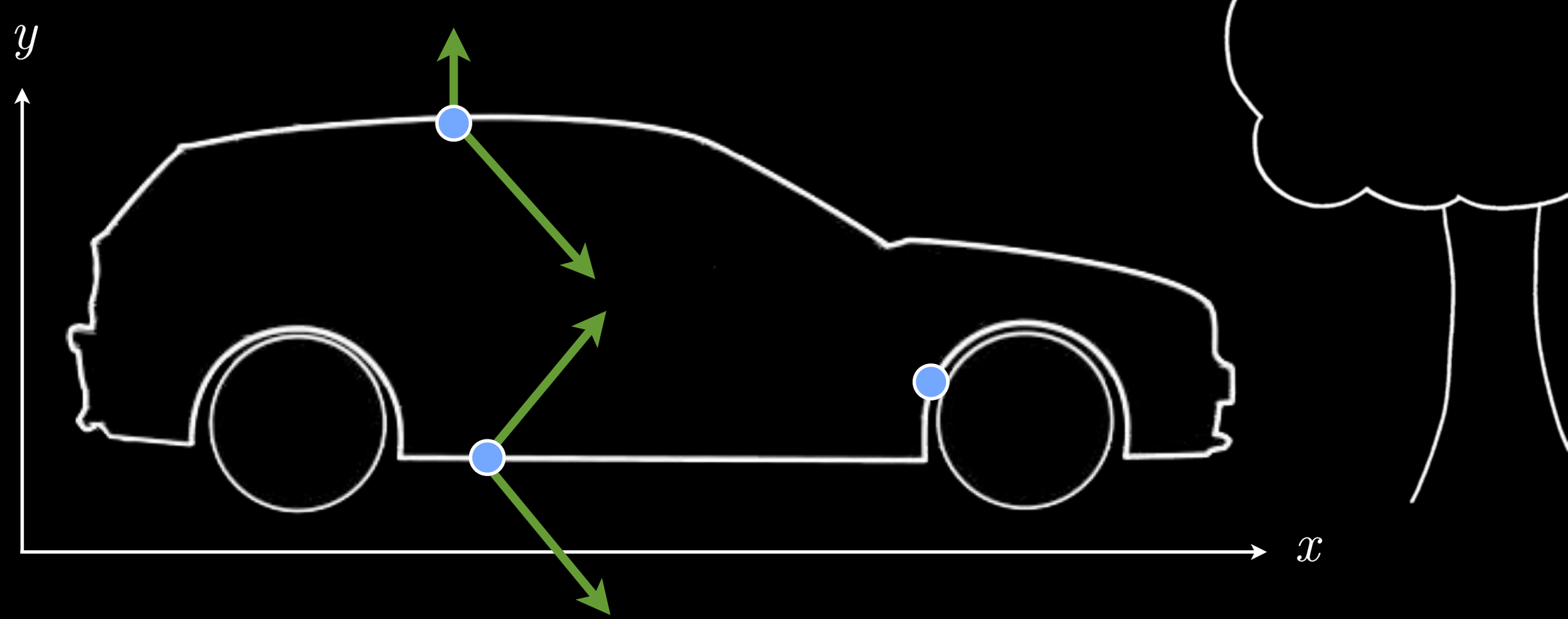

y
x

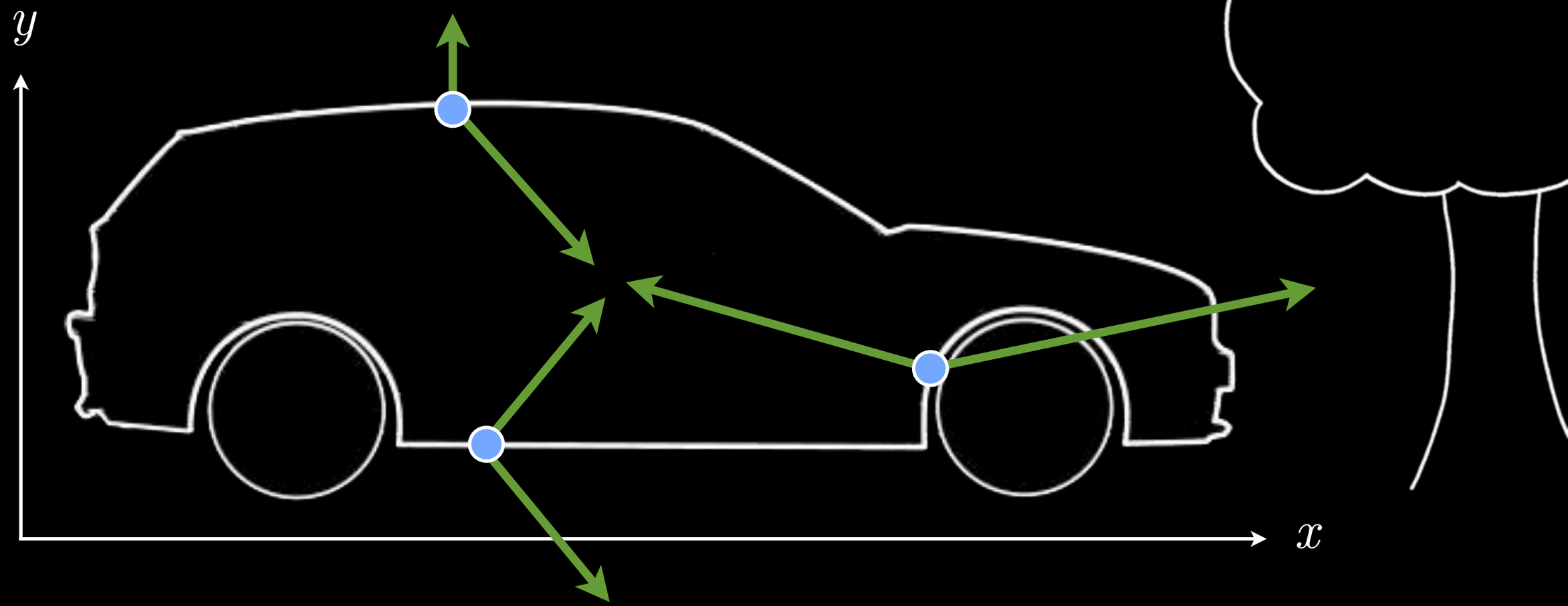

y
x

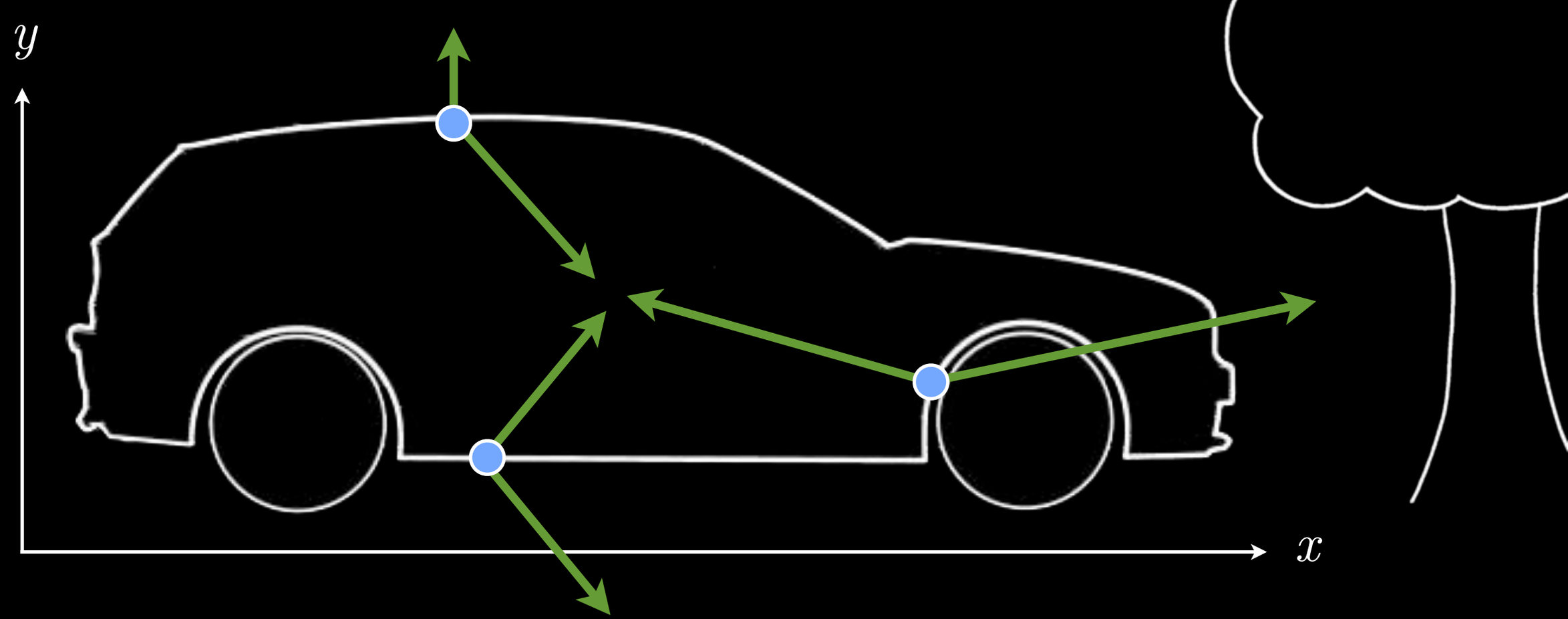

y
x

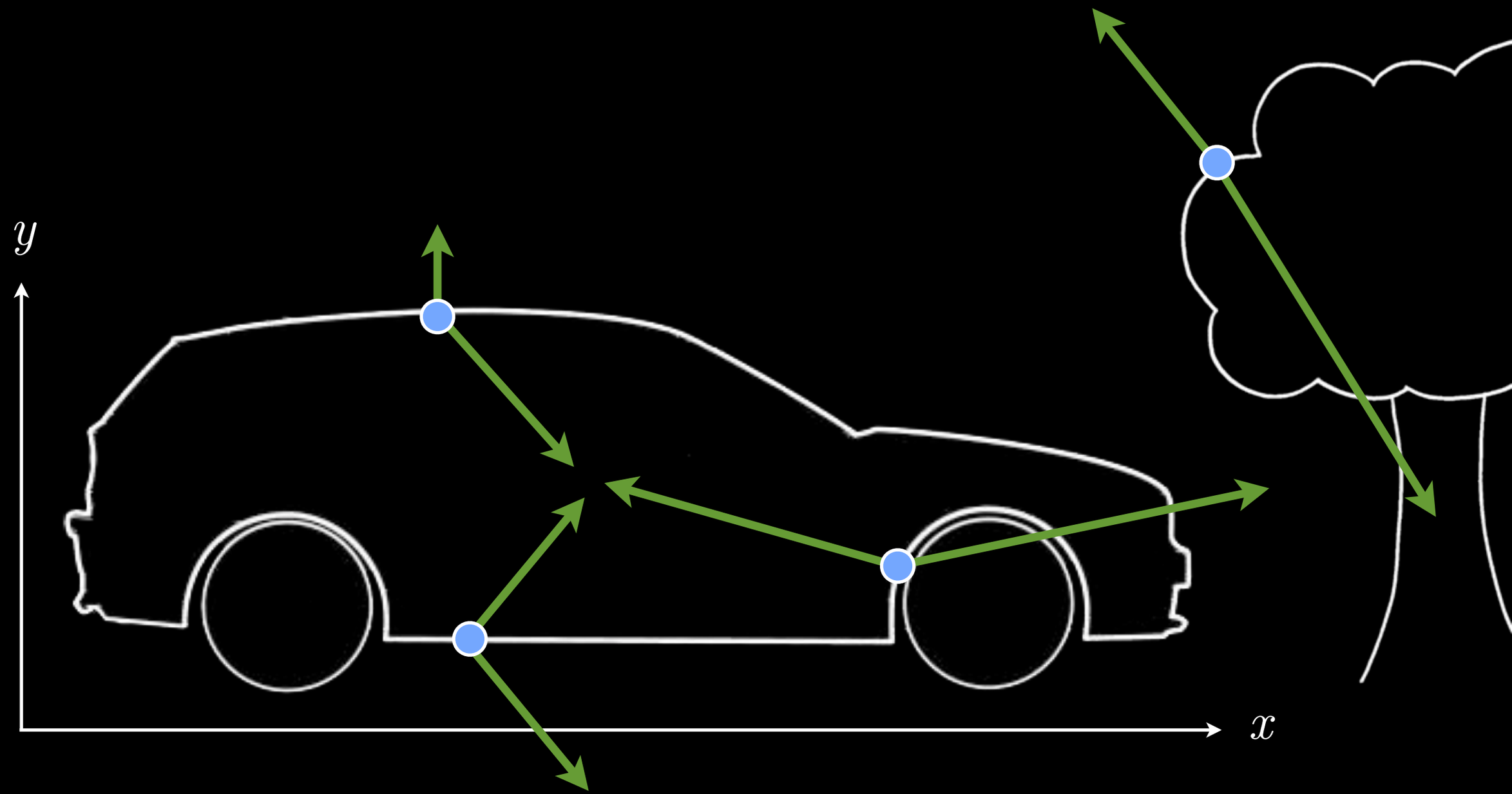

y
x

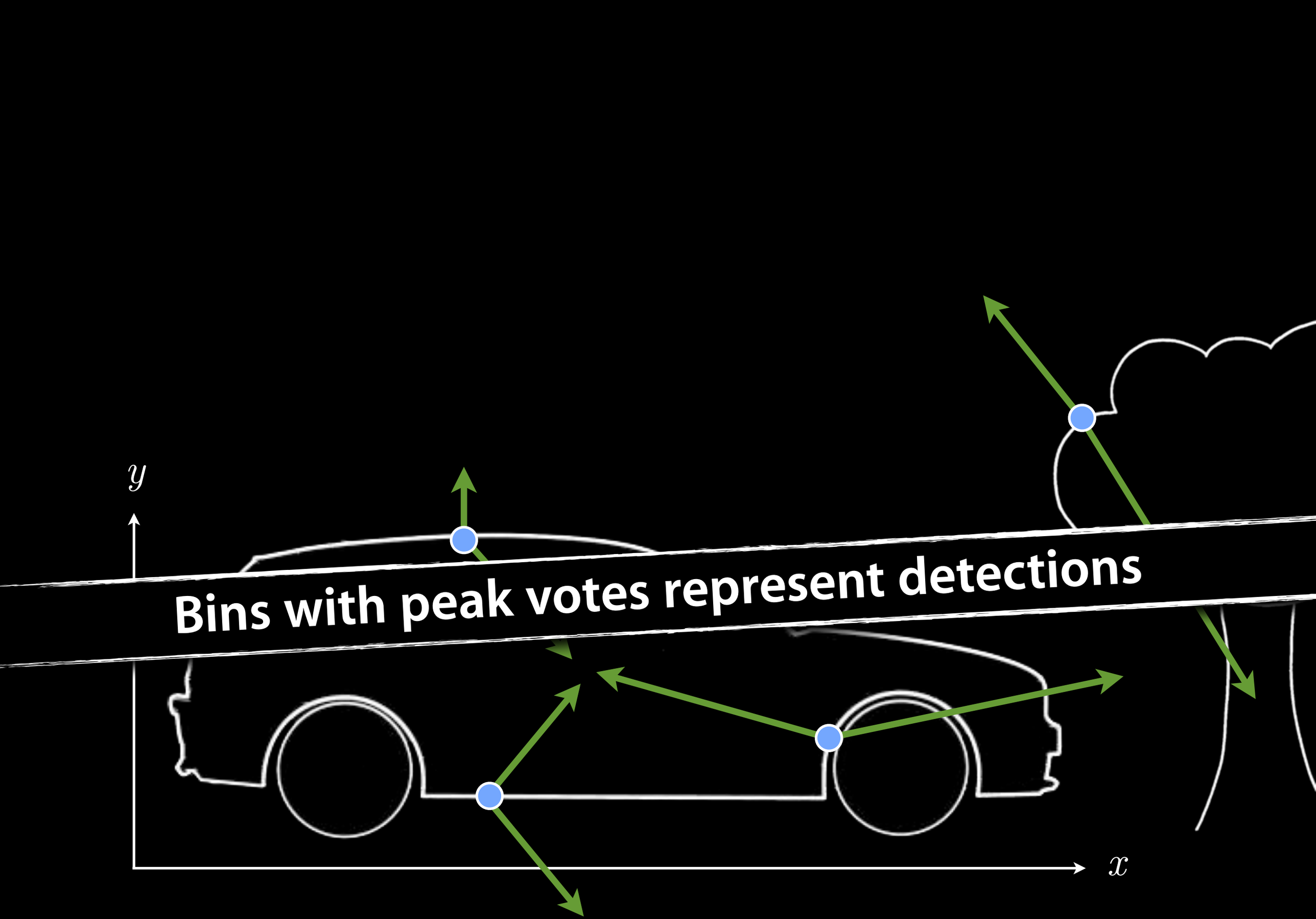

y
x

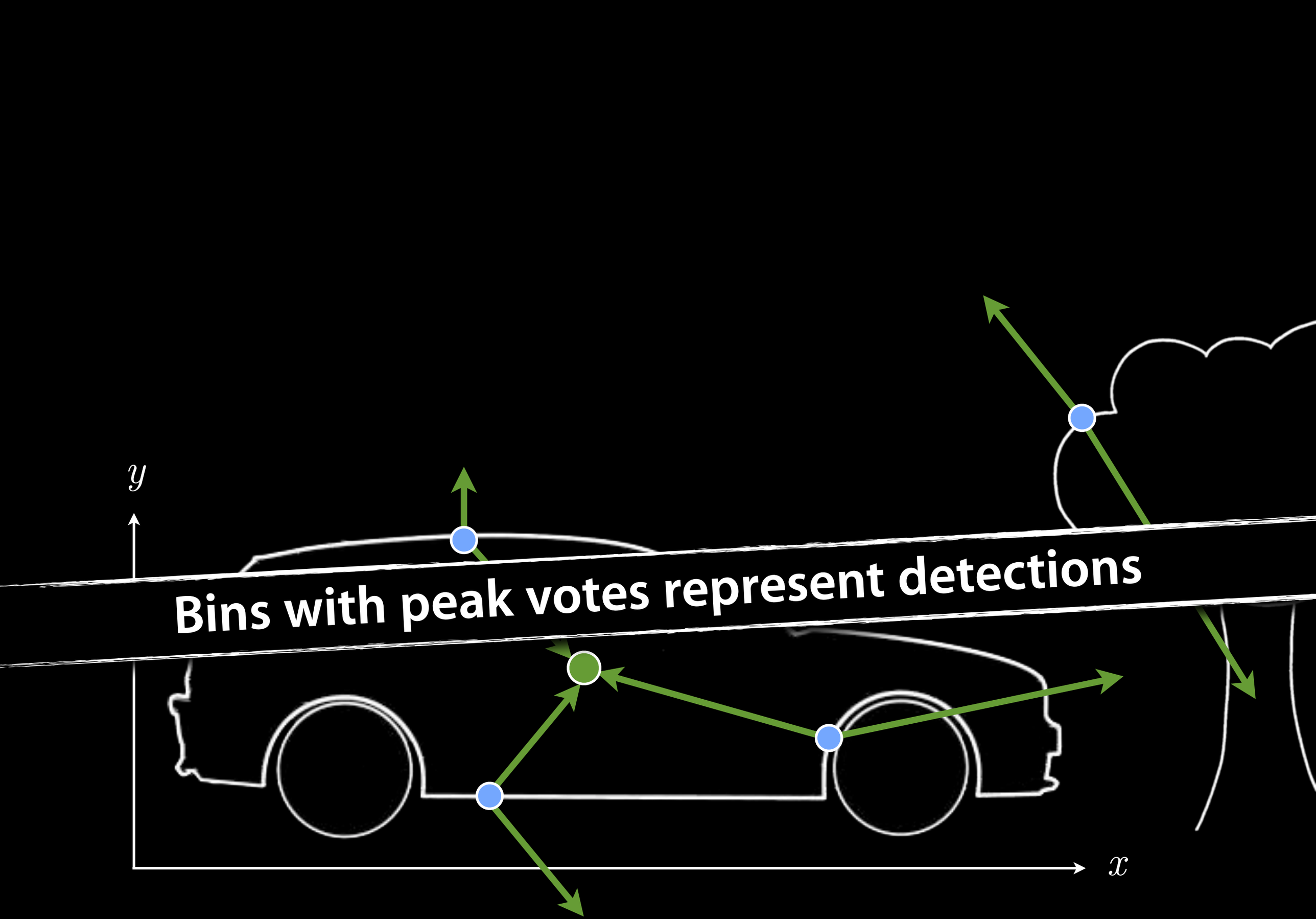

y
x


y
x


y
x


y
x


y
x
Bins with peak votes represent detections


y
x
Bins with peak votes represent detections

visual codeword

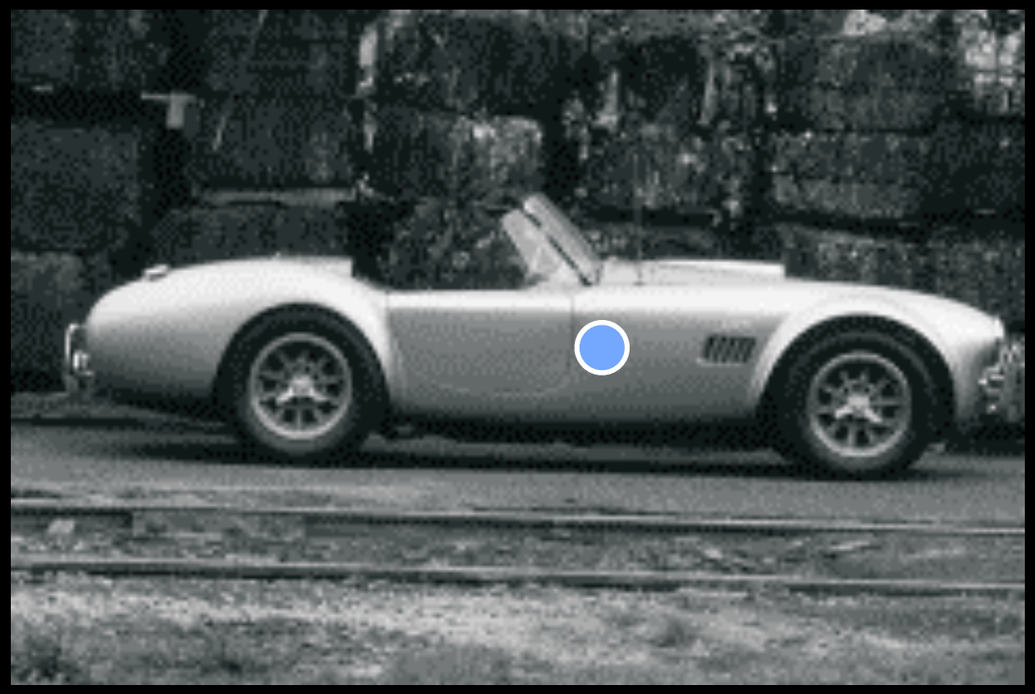
visual codeword

Instead of indexing displacements by gradient orientation index by “visual codeword”

training image

visual codeword

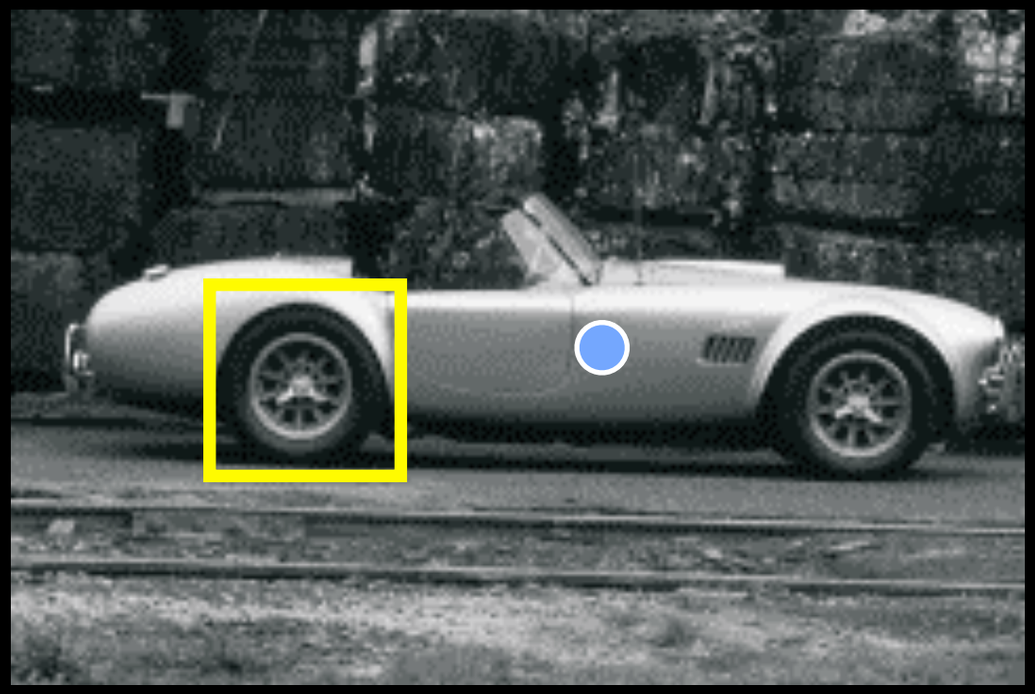
Instead of indexing displacements by gradient orientation index by “visual codeword”



training image

visual codeword

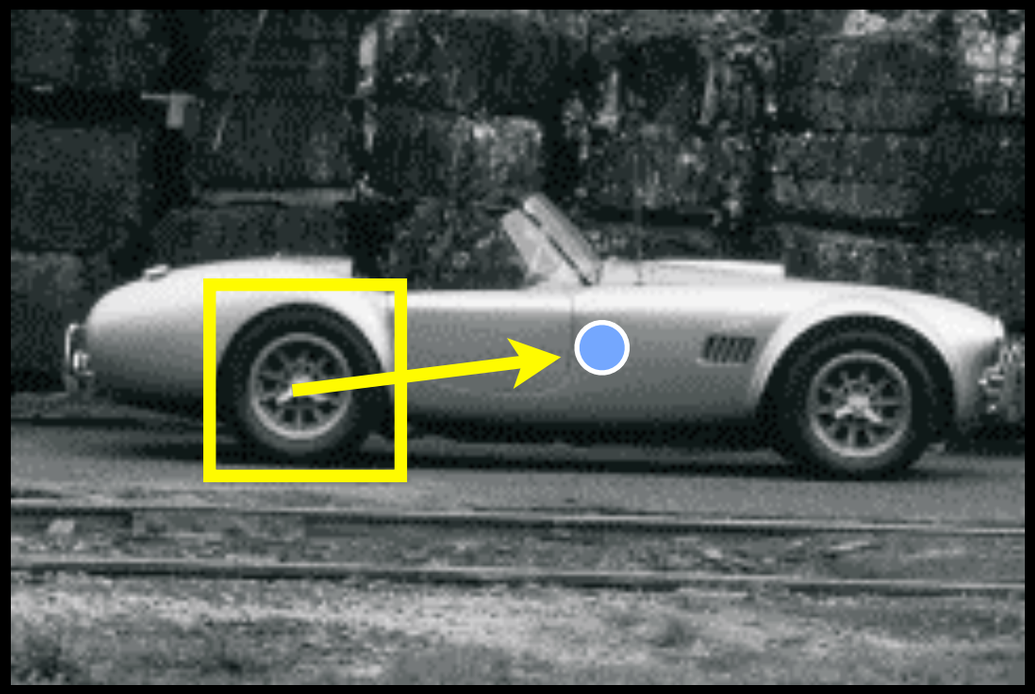
Instead of indexing displacements by gradient orientation index by “visual codeword”



training image

visual codeword

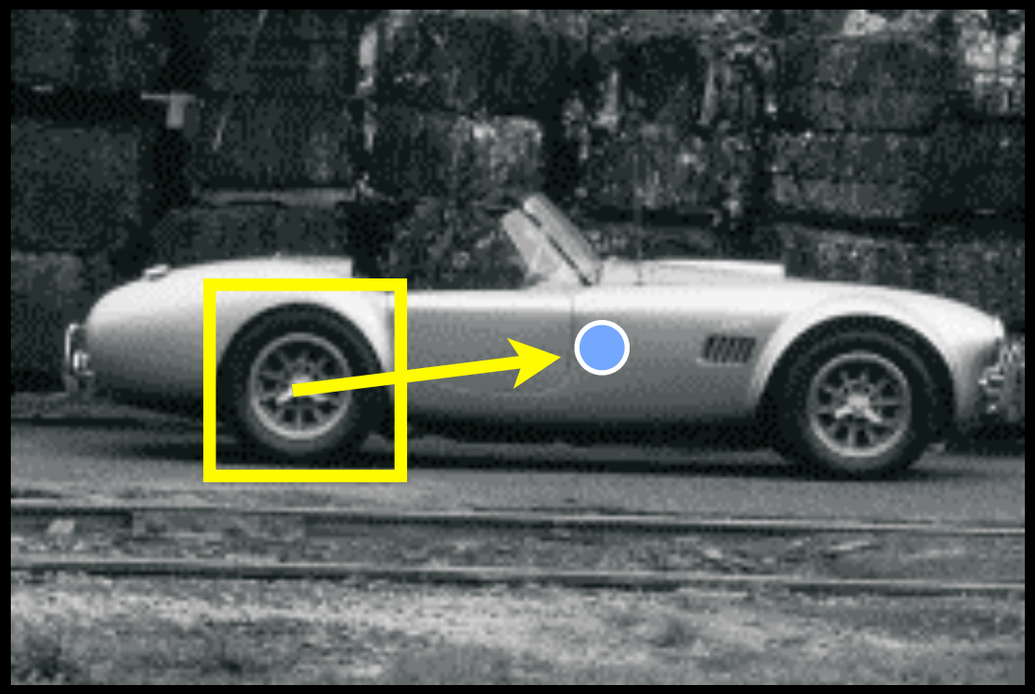
Instead of indexing displacements by gradient orientation index by “visual codeword”



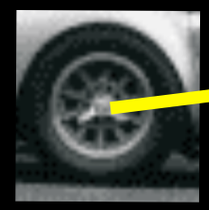
training image

visual codeword

Instead of indexing displacements by gradient orientation index by “visual codeword”

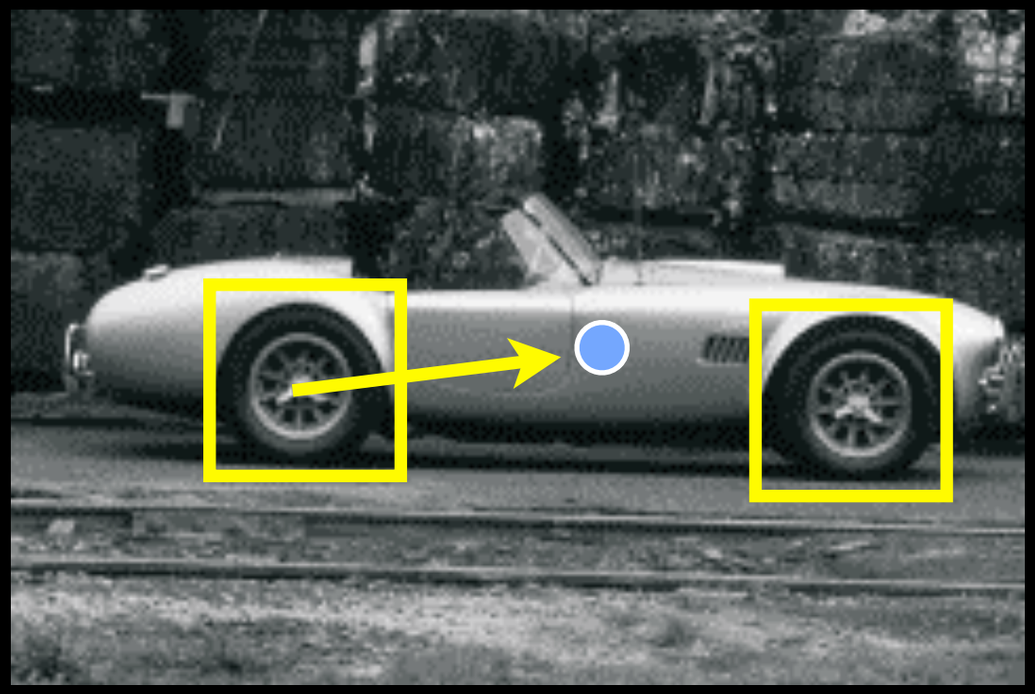


training image

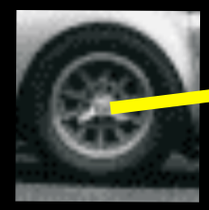


visual codeword

Instead of indexing displacements by gradient orientation index by “visual codeword”

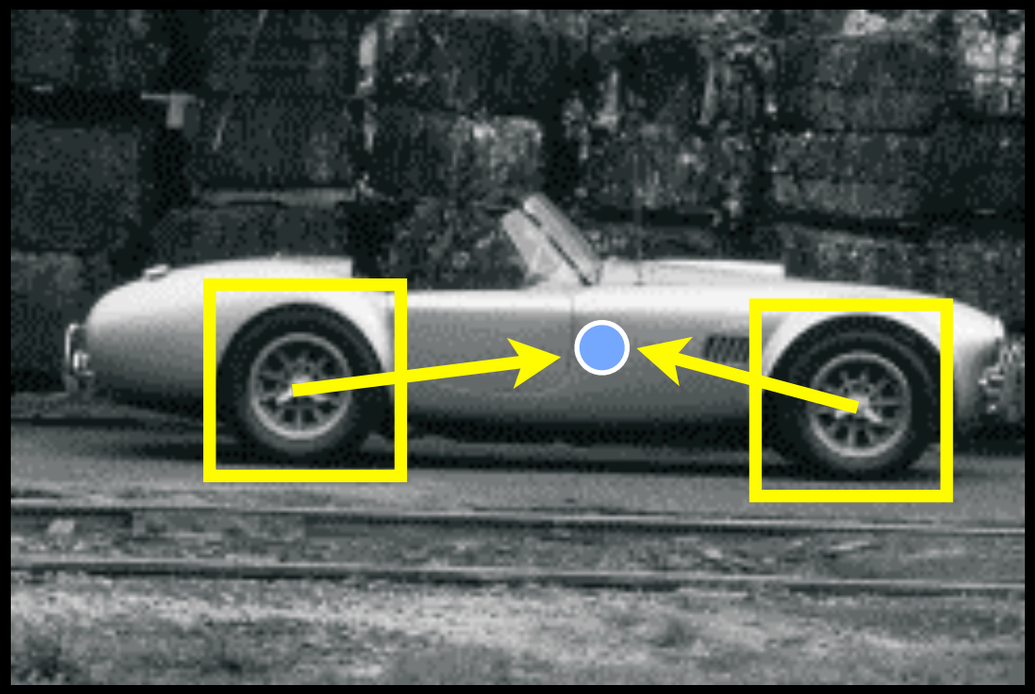


training image

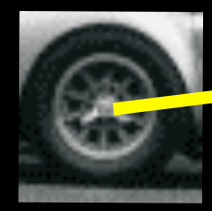


visual codeword

Instead of indexing displacements by gradient orientation index by “visual codeword”

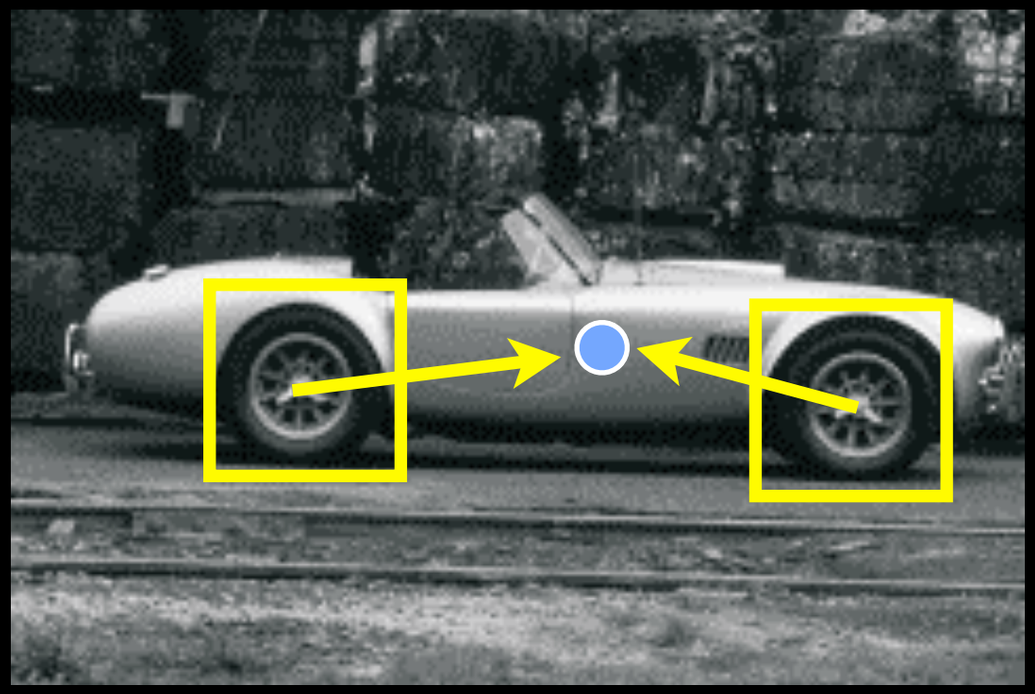


training image

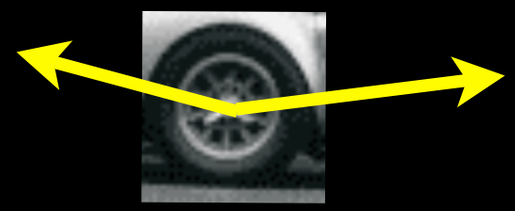


visual codeword

Instead of indexing displacements by gradient orientation index by “visual codeword”



training image

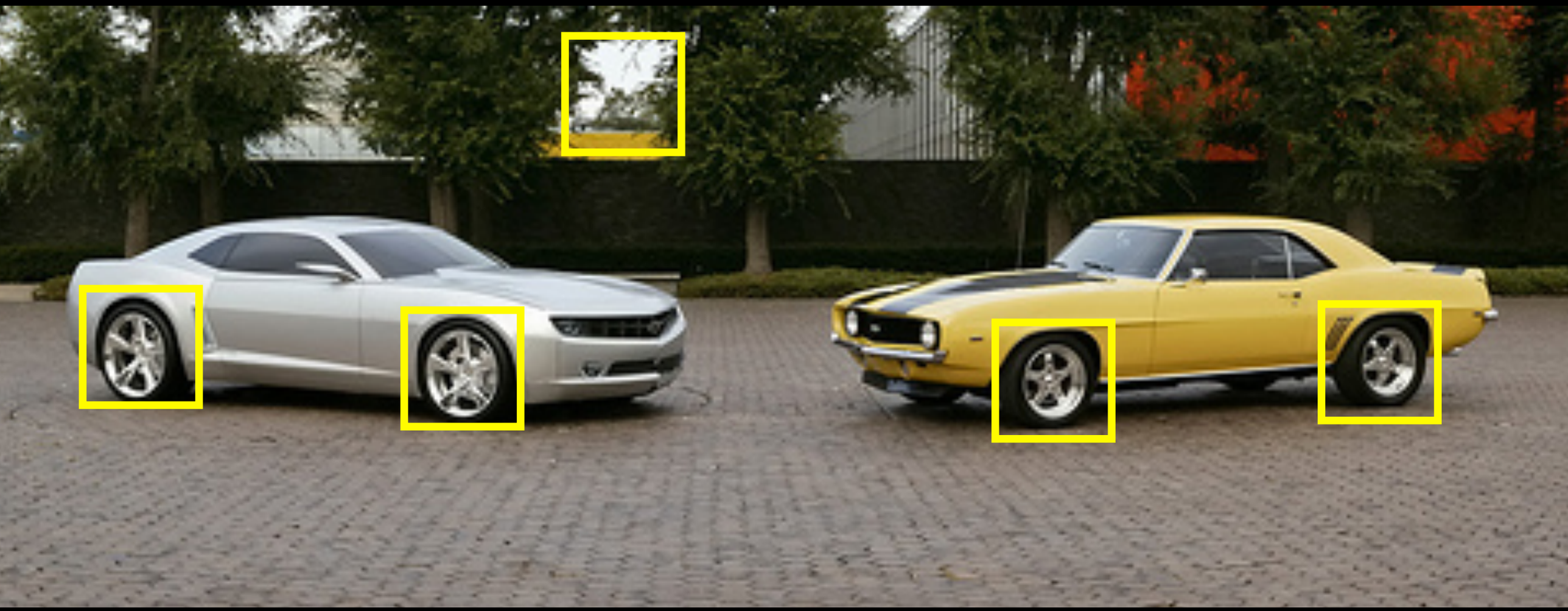


visual codeword

Instead of indexing displacements by gradient orientation index by “visual codeword”

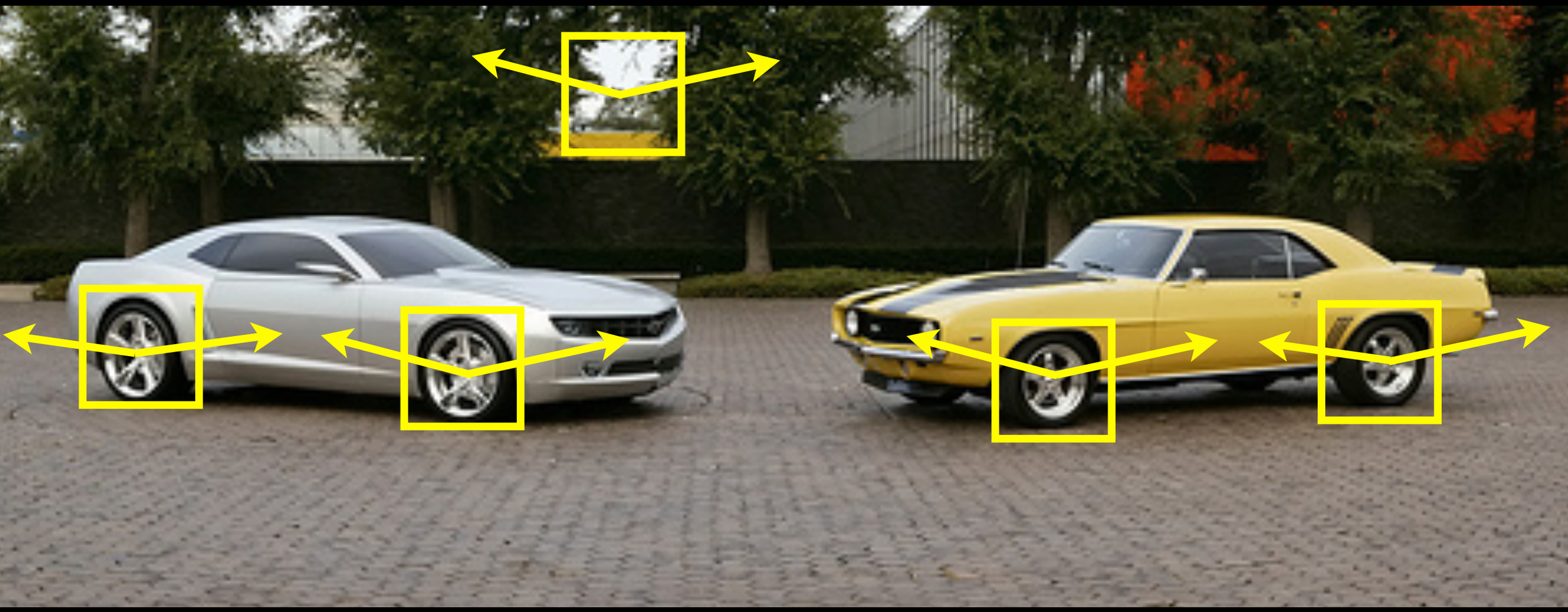
test image

Leibe, Leonardis and Schiele, “Combined Object Categorization and Segmentation with Implicit Shape Model”, 2004



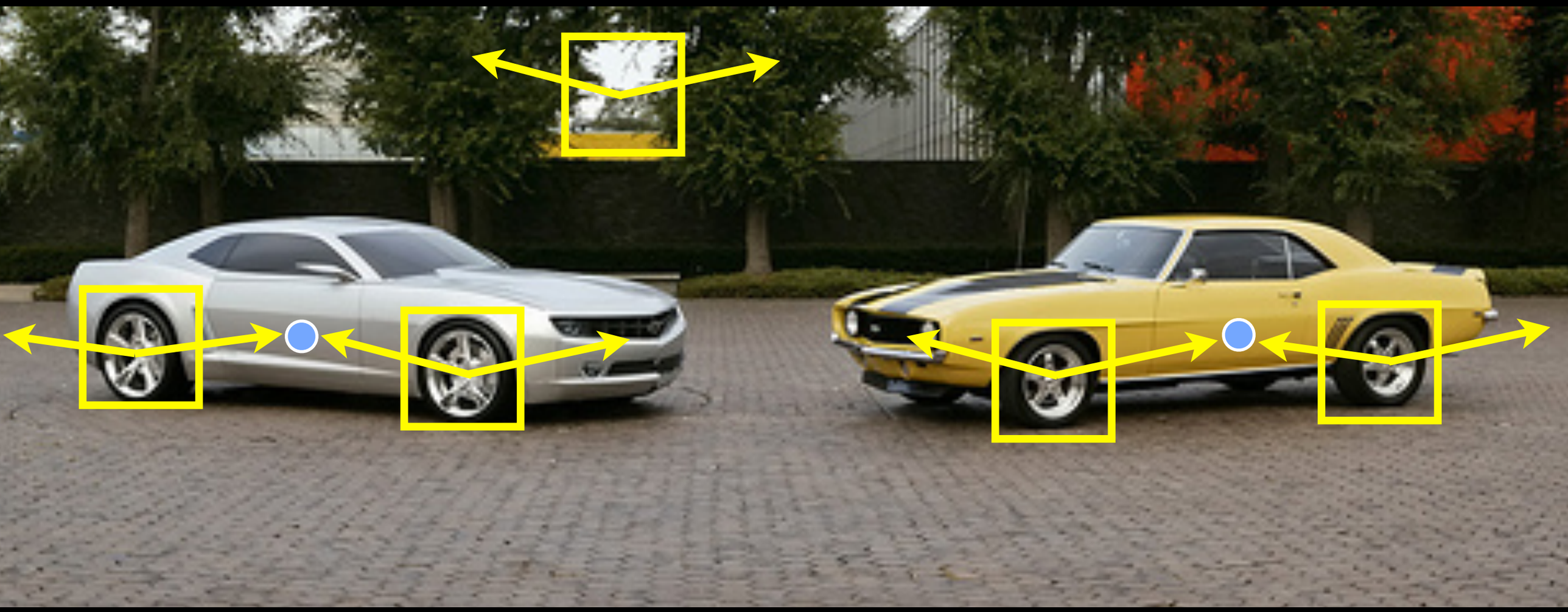
test image

Leibe, Leonardis and Schiele, “Combined Object Categorization and Segmentation with Implicit Shape Model”, 2004



test image

Leibe, Leonardis and Schiele, “Combined Object Categorization and Segmentation with Implicit Shape Model”, 2004



test image

Leibe, Leonardis and Schiele, “Combined Object Categorization and Segmentation with Implicit Shape Model”, 2004

RANSAC

RANdom **SA**mple **C**onsensus

Graphics and
Image Processing

J. D. Foley
Editor

Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography

Martin A. Fischler and Robert C. Bolles
SRI International

A new paradigm, Random Sample Consensus (RANSAC), for fitting a model to experimental data is introduced. RANSAC is capable of interpreting/smoothing data containing a significant percentage of gross errors, and is thus ideally suited for applications in automated image analysis where interpretation is based on the data provided by error-prone feature detectors. A major portion of this paper describes the application of RANSAC to the Location Determination Problem (LDP): Given an image depicting a set of landmarks with known locations, determine that point in space from which the image was obtained. In response to a RANSAC requirement, new results are derived on the minimum number of landmarks needed to obtain a solution, and algorithms are presented for

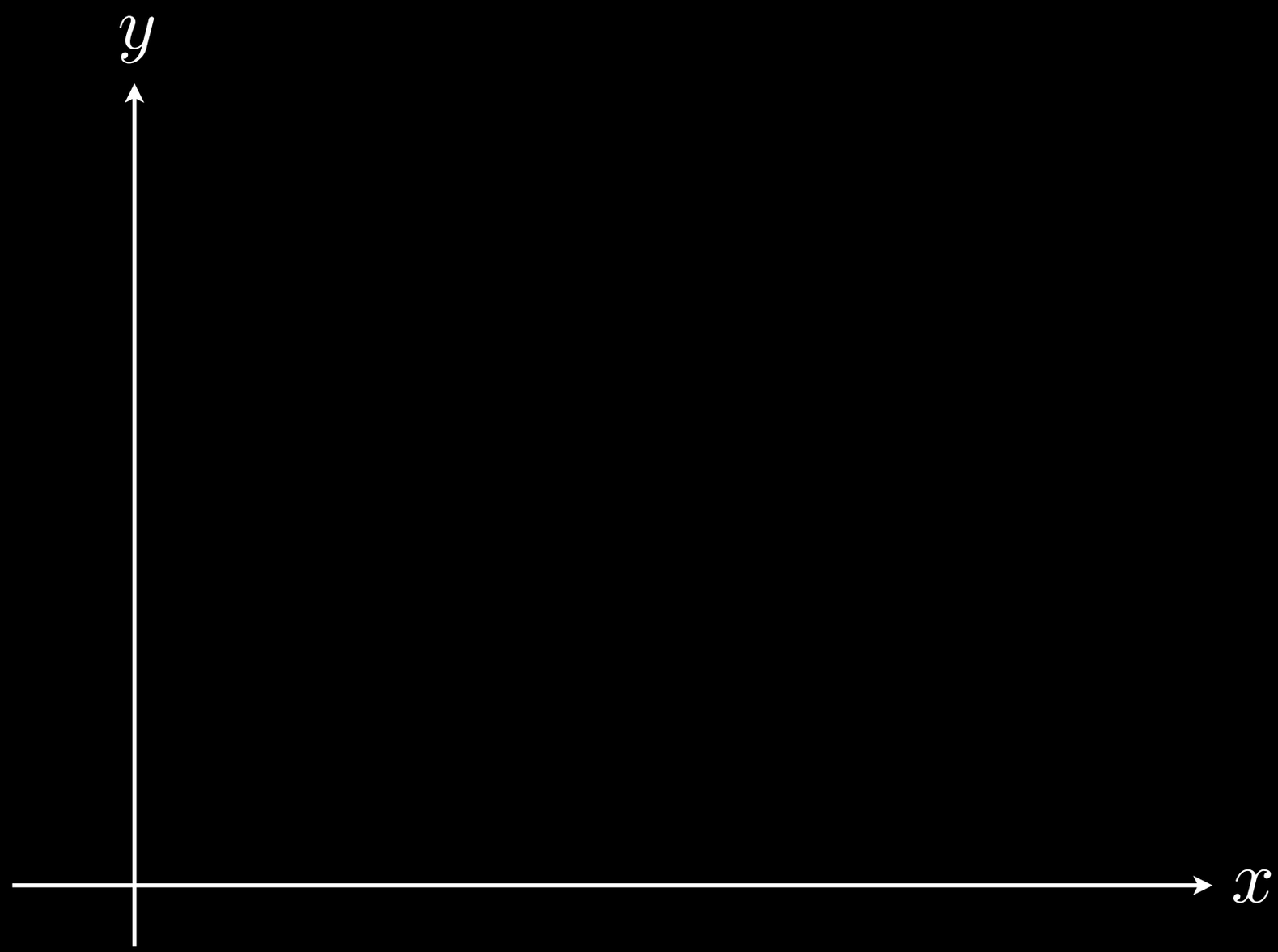
I. Introduction

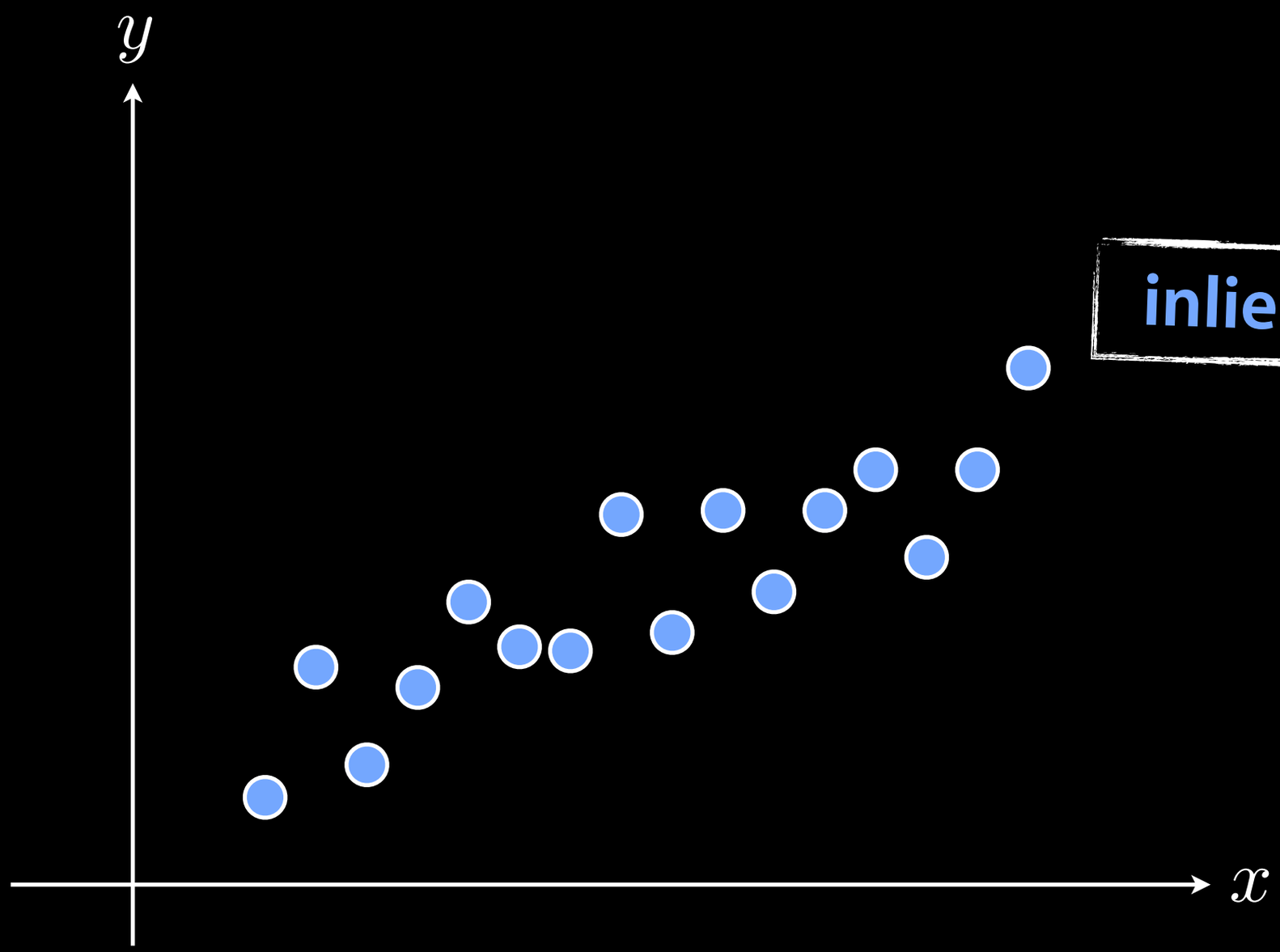
We introduce a new paradigm, Random Sample Consensus (RANSAC), for fitting a model to experimental data; and illustrate its use in scene analysis and automated cartography. The application discussed, the location determination problem (LDP), is treated at a level beyond that of a mere example of the use of the RANSAC paradigm; new basic findings concerning the conditions under which the LDP can be solved are presented and a comprehensive approach to the solution of this problem that we anticipate will have near-term practical applications is described.

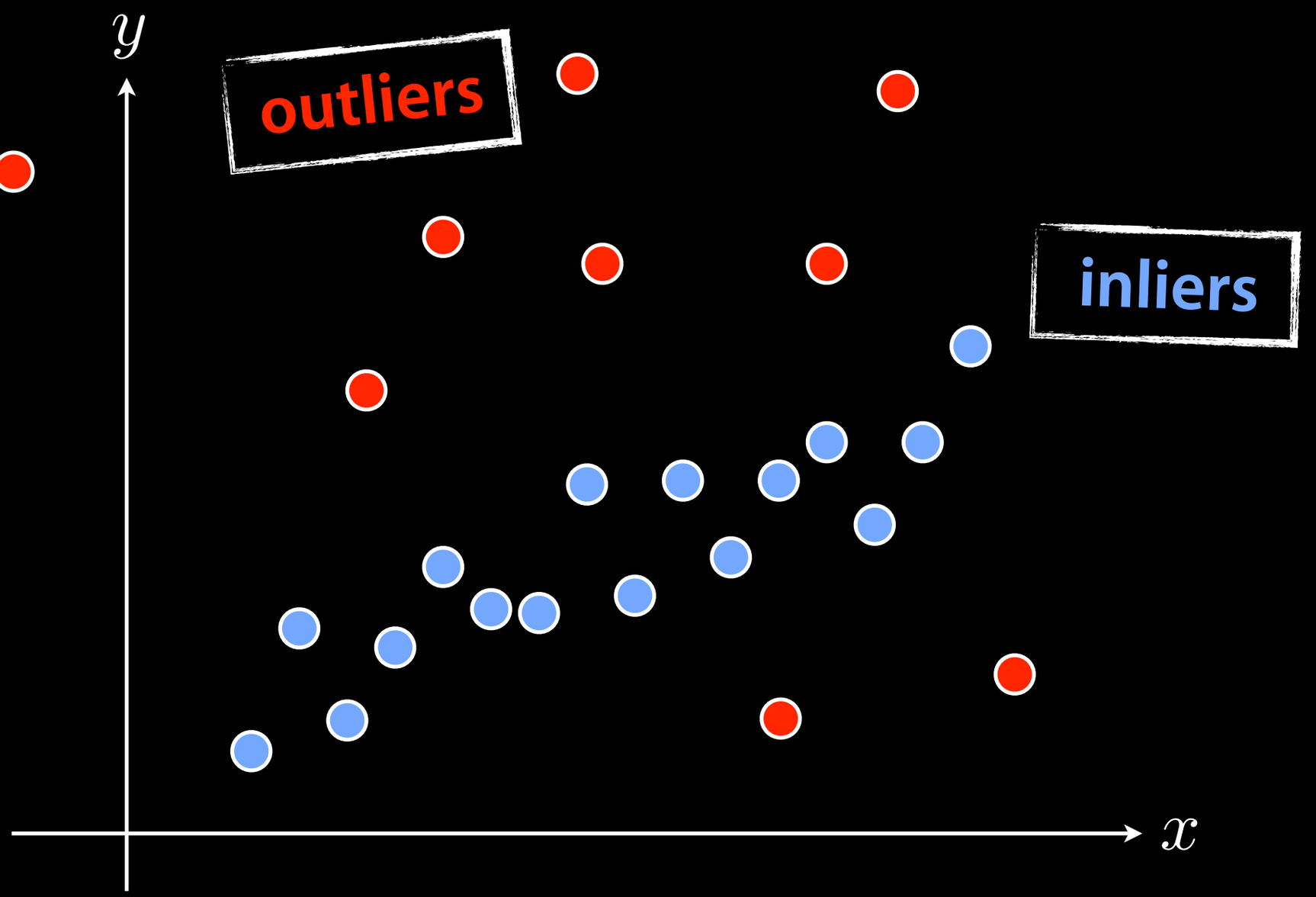
To a large extent, scene analysis (and, in fact, science in general) is concerned with the interpretation of sensed data in terms of a set of predefined models. Conceptually, interpretation involves two distinct activities: First, there is the problem of finding the best match between the data and one of the available models (the classification problem); Second, there is the problem of computing the best values for the free parameters of the selected model (the parameter estimation problem). In practice, these two problems are not independent—a solution to the parameter estimation problem is often required to solve the classification problem.

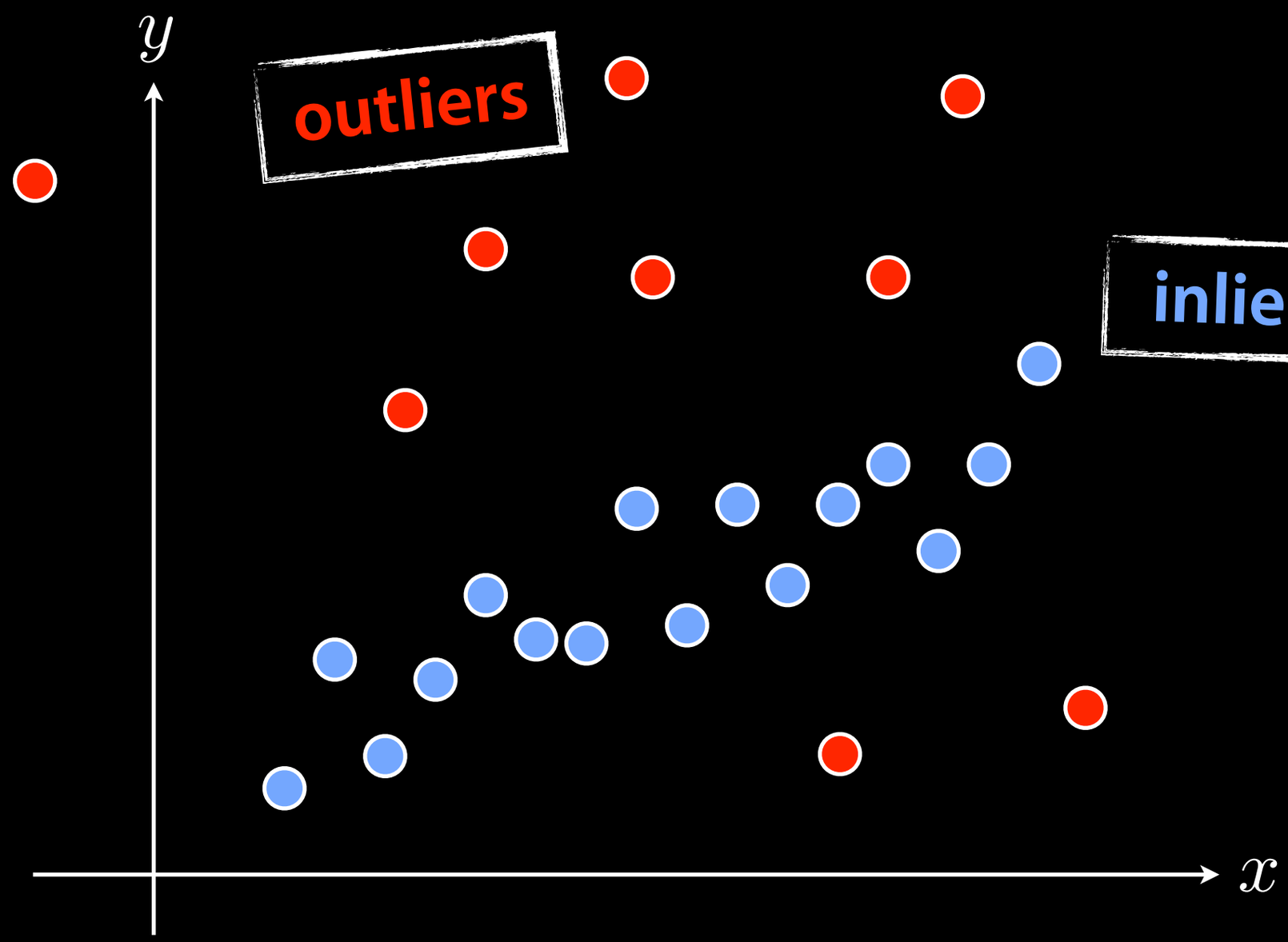
Classical techniques for parameter estimation, such as least squares, optimize (according to a specified objective function) the fit of a functional description (model) to *all* of the presented data. These techniques have no internal mechanisms for detecting and rejecting gross errors. They are averaging techniques that rely on the assumption (that is, hoping, assuming) that the maximum expected deviation of any datum from the assumed model is a direct function of the size of the data set, and thus regardless of the size of the data set, there

General procedure for fitting a parametric model to data containing outliers

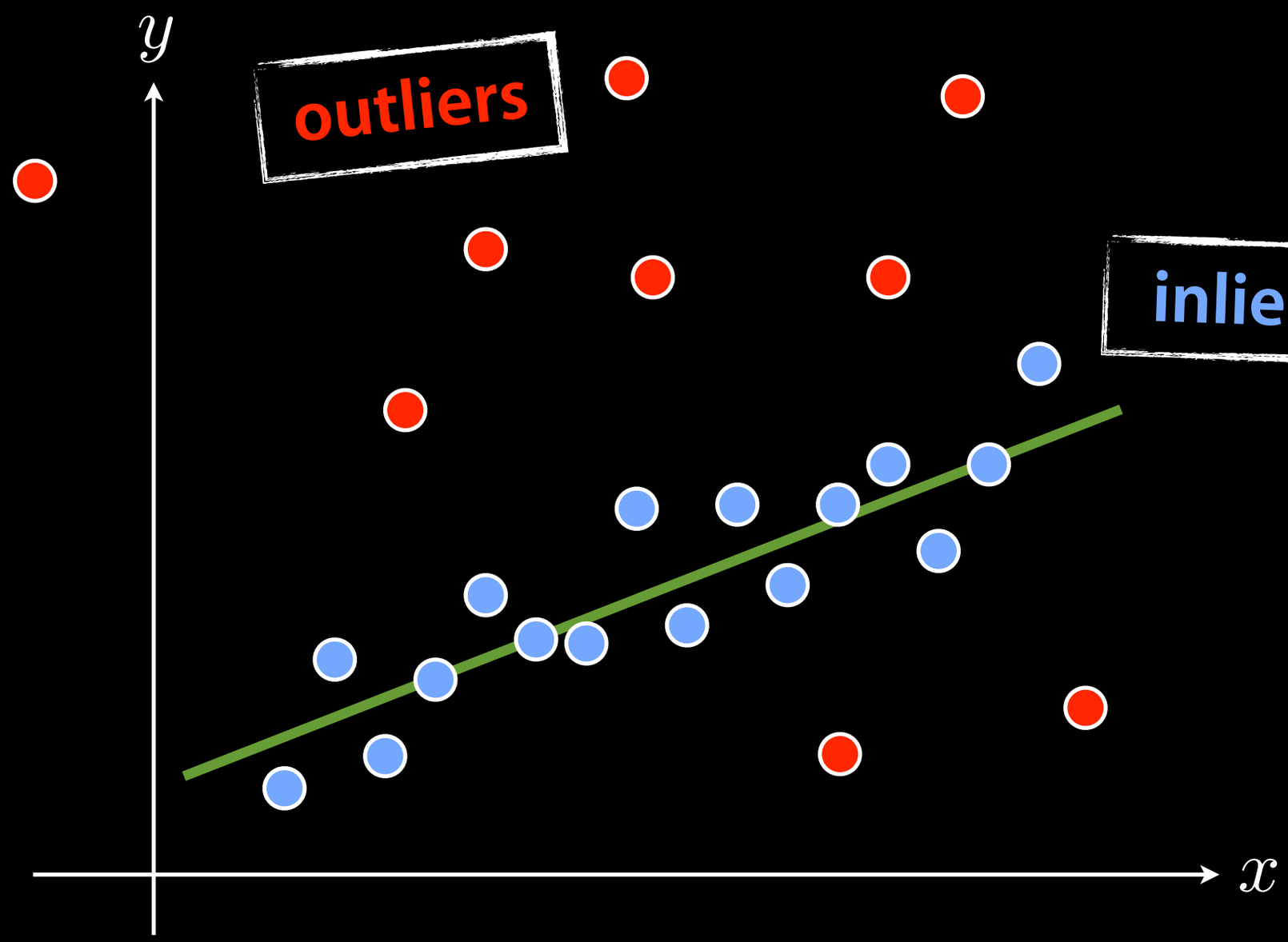

y
x


y
inliers
x

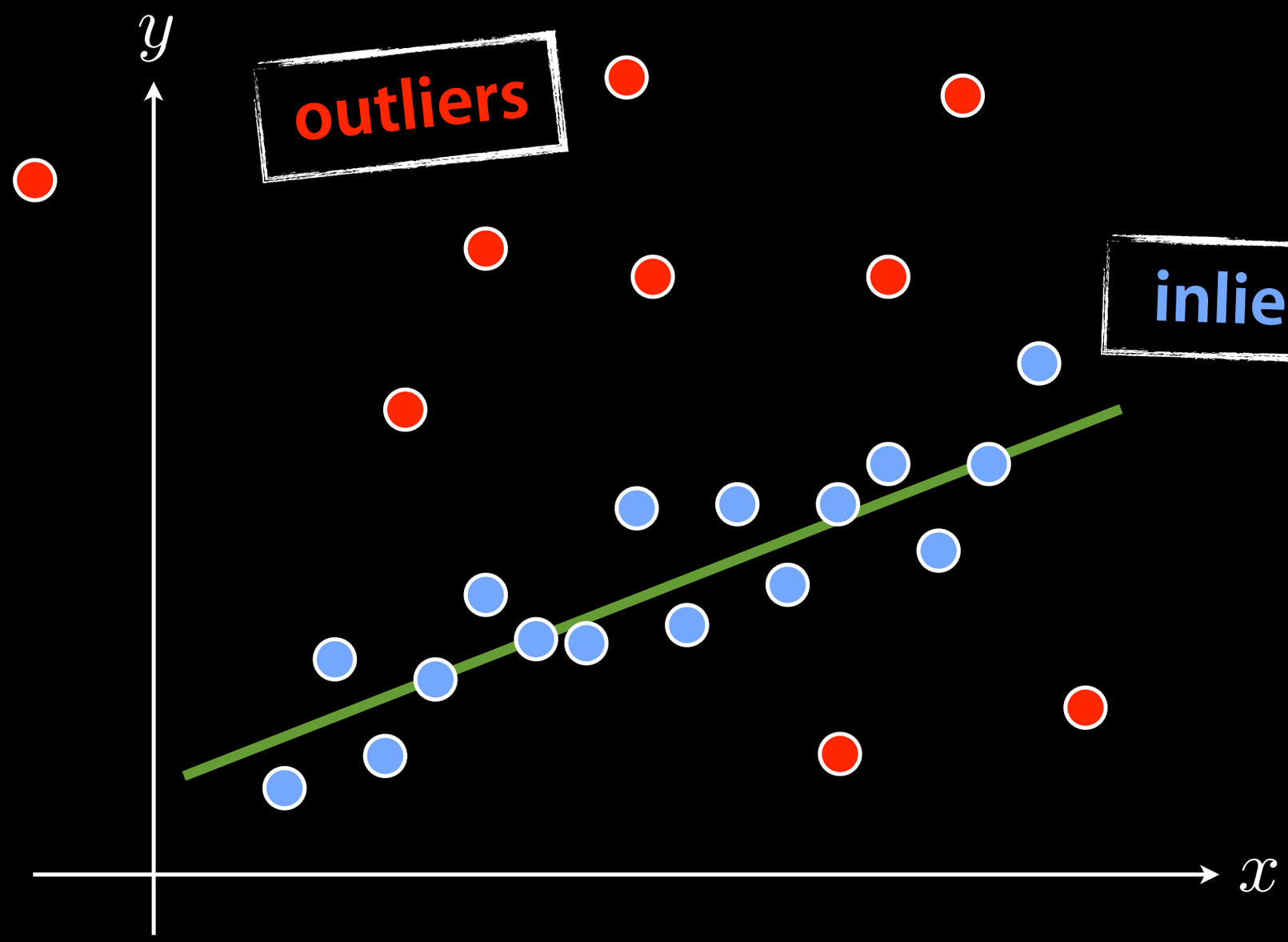

y
outliers
inliers
x



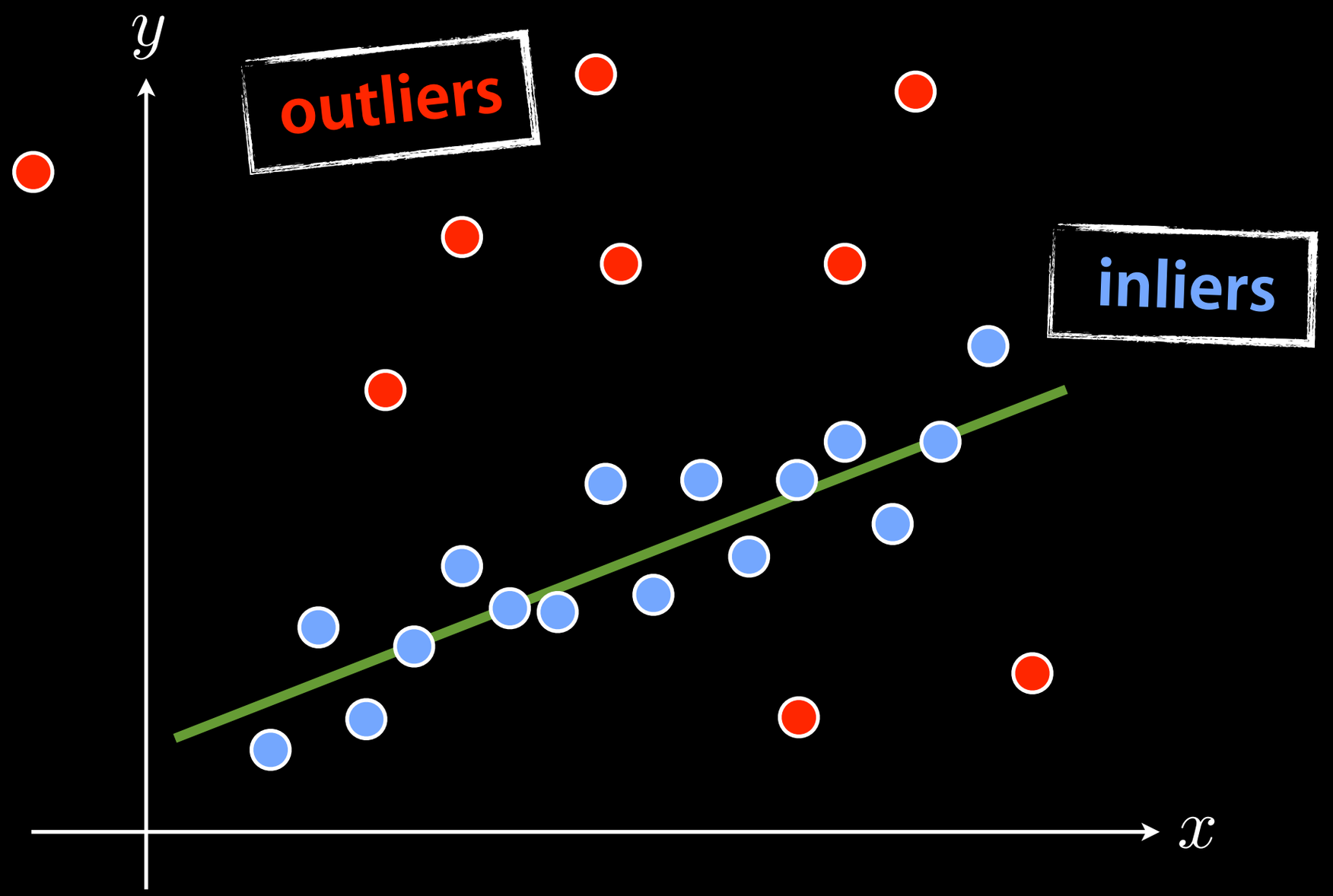
Goal: Fit the model to the data with **outliers**



Goal: Fit the model to the data with **outliers**

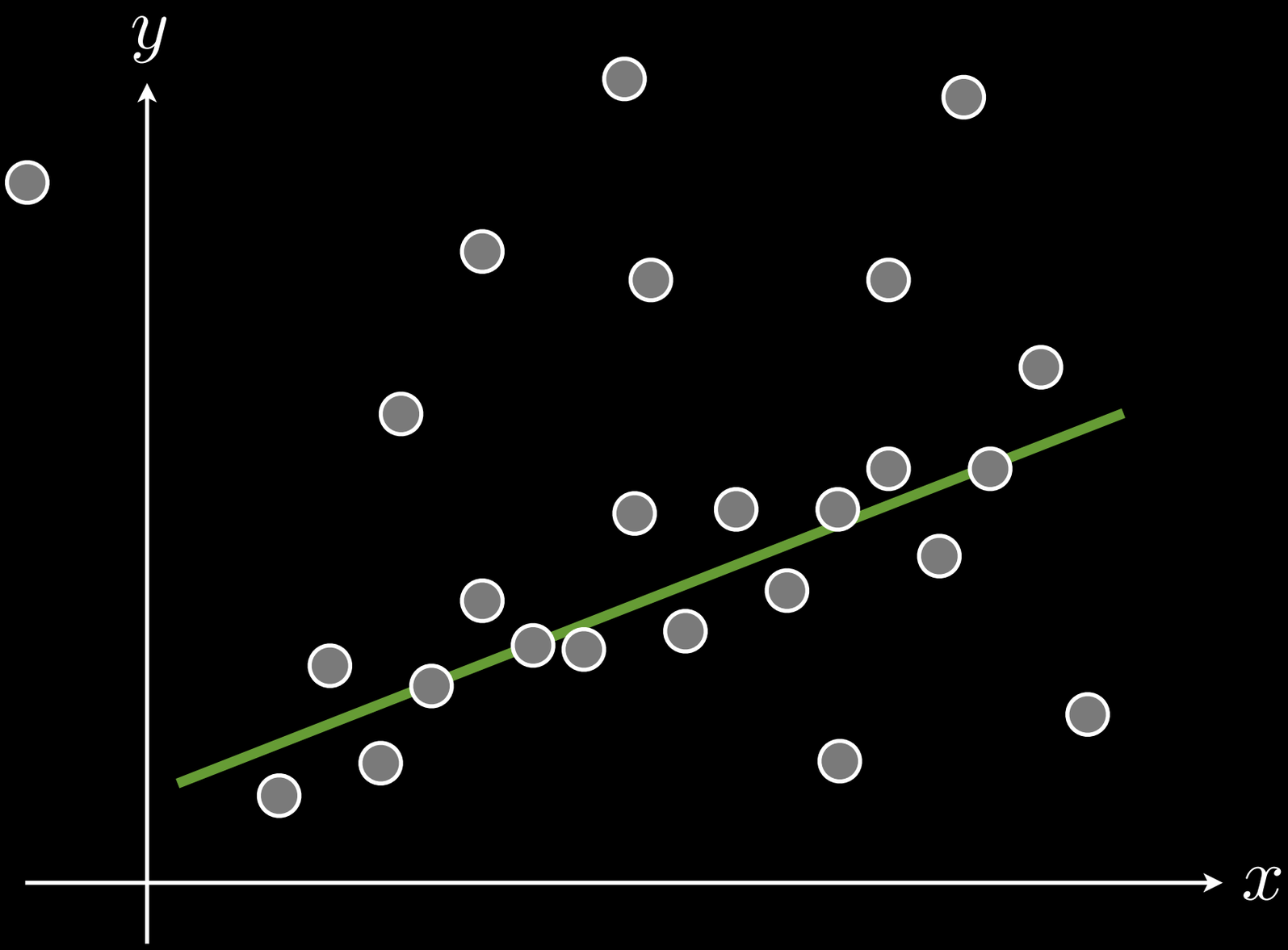


Goal: Fit the model to the data with outliers



Goal: Fit the model to the data with outliers

unknown



unknown

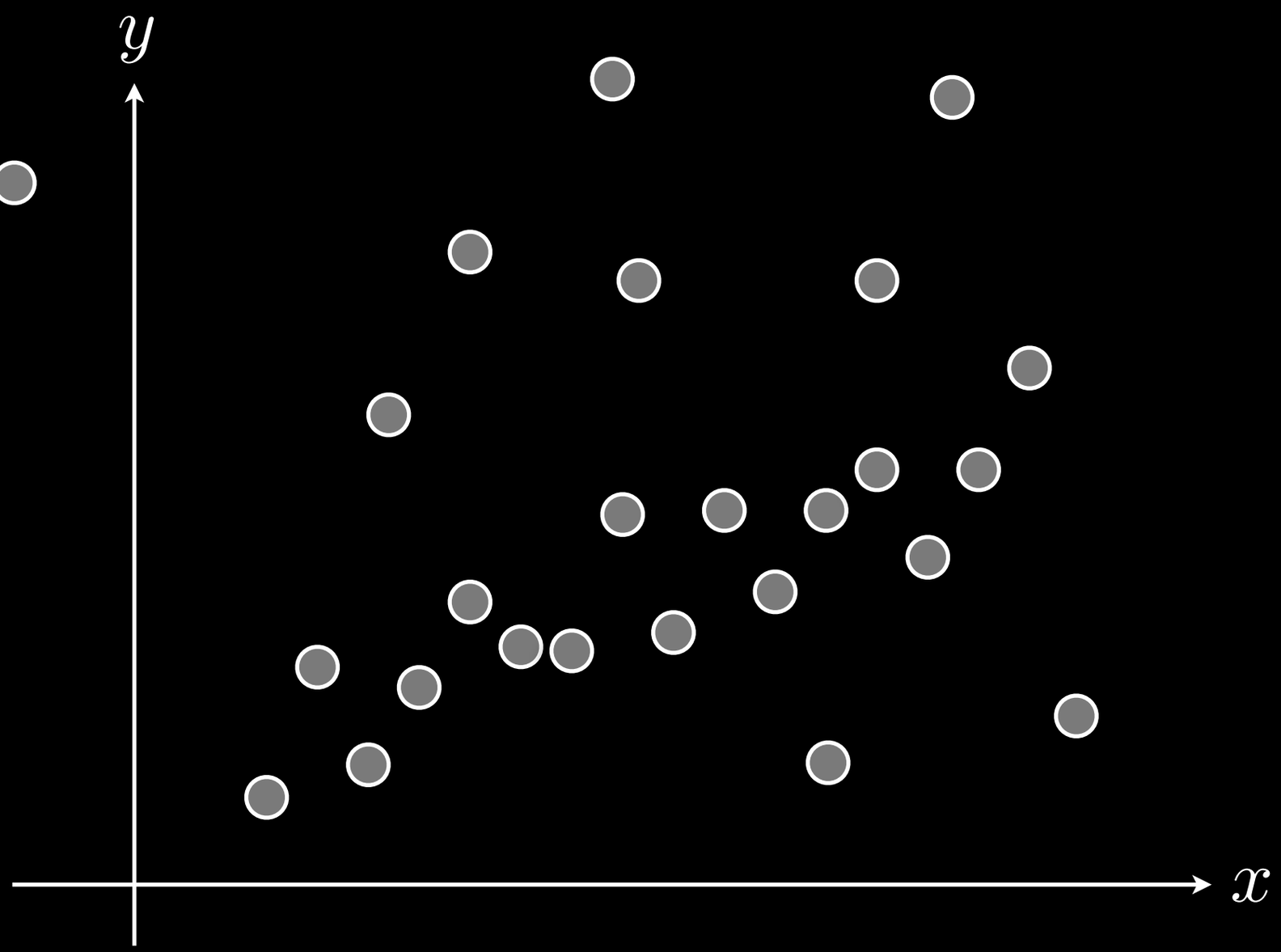
Goal: Fit the model to the data with **outliers**

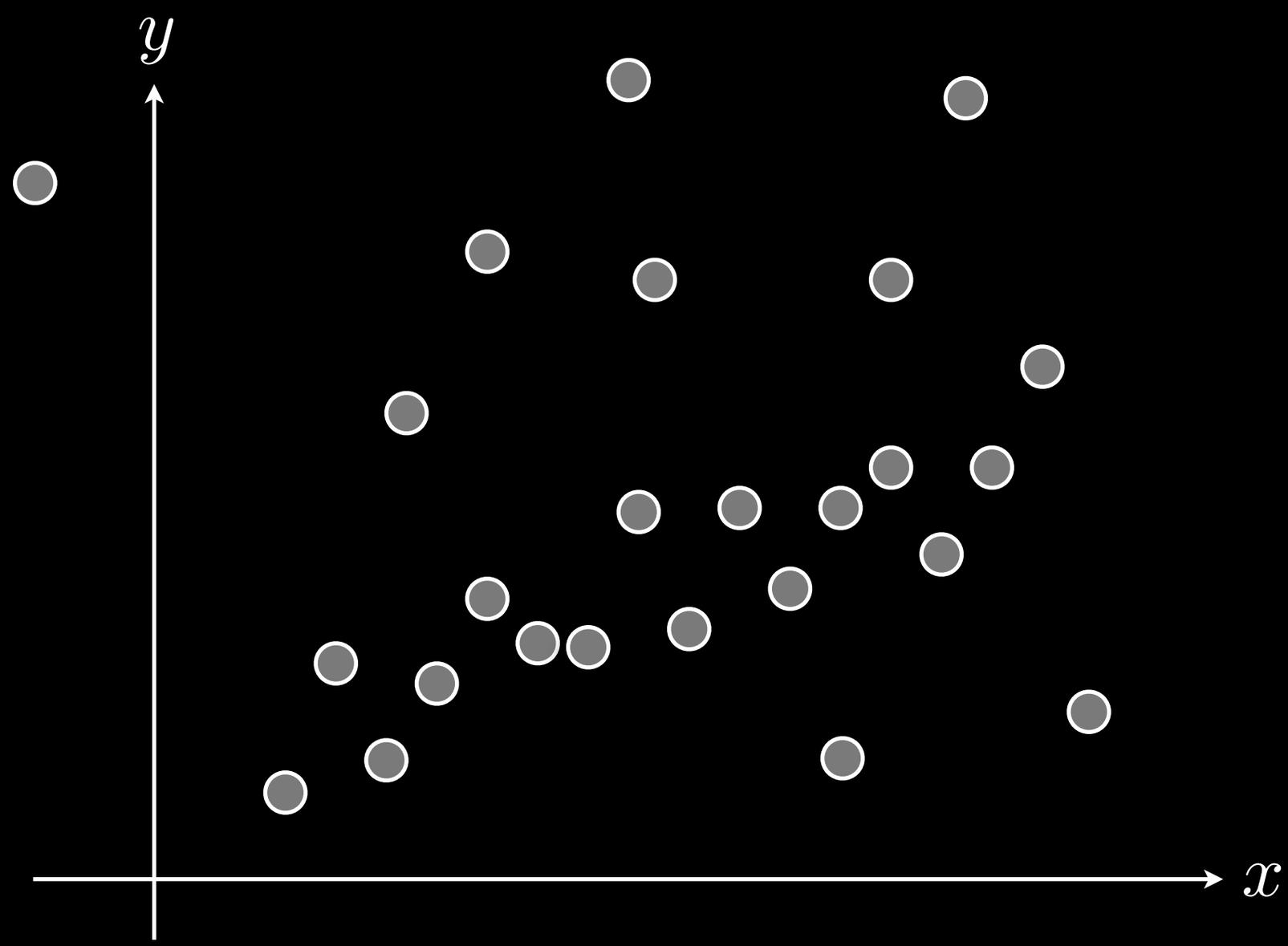
4

major steps

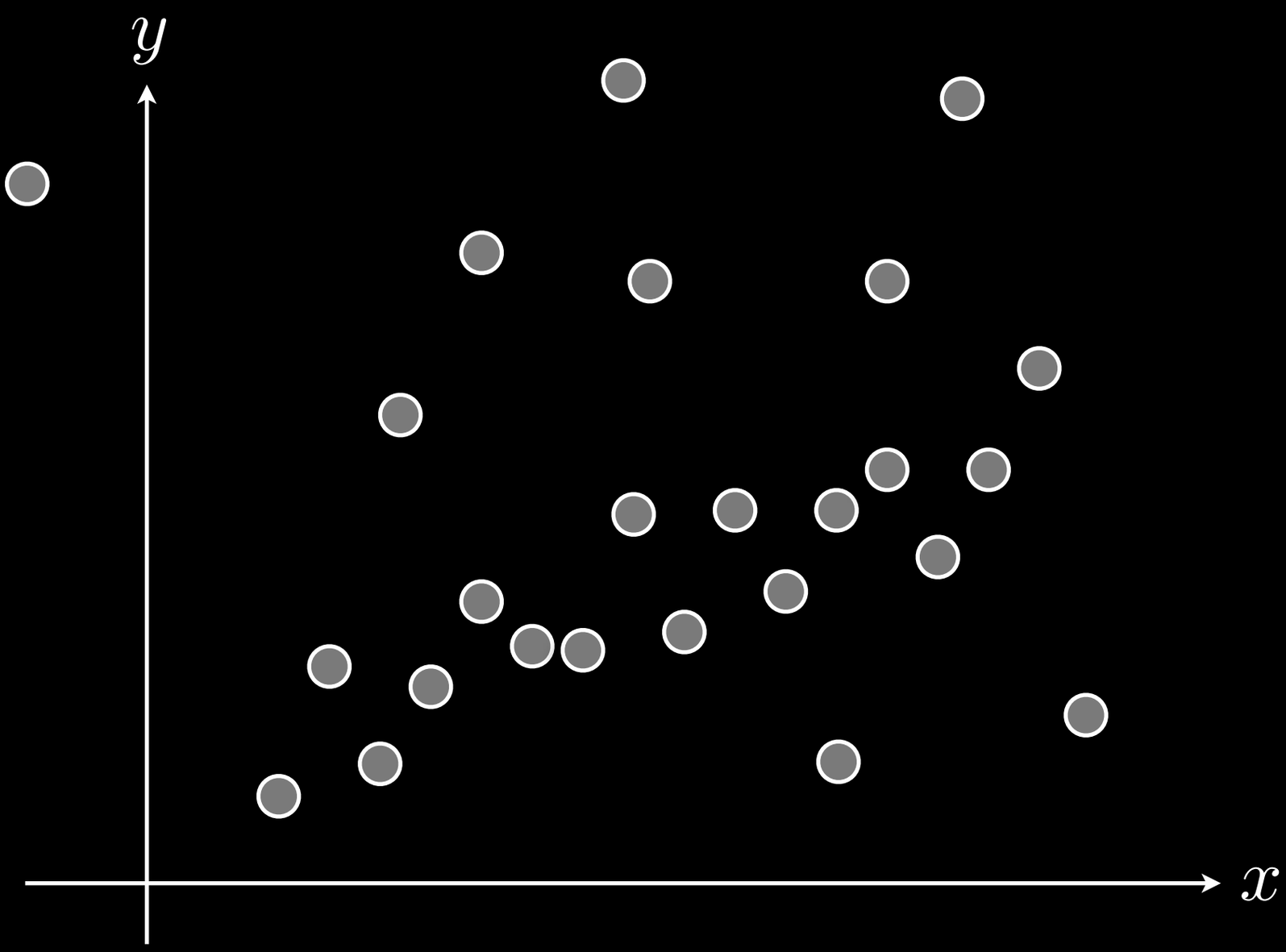
Step 1

Select minimum point set

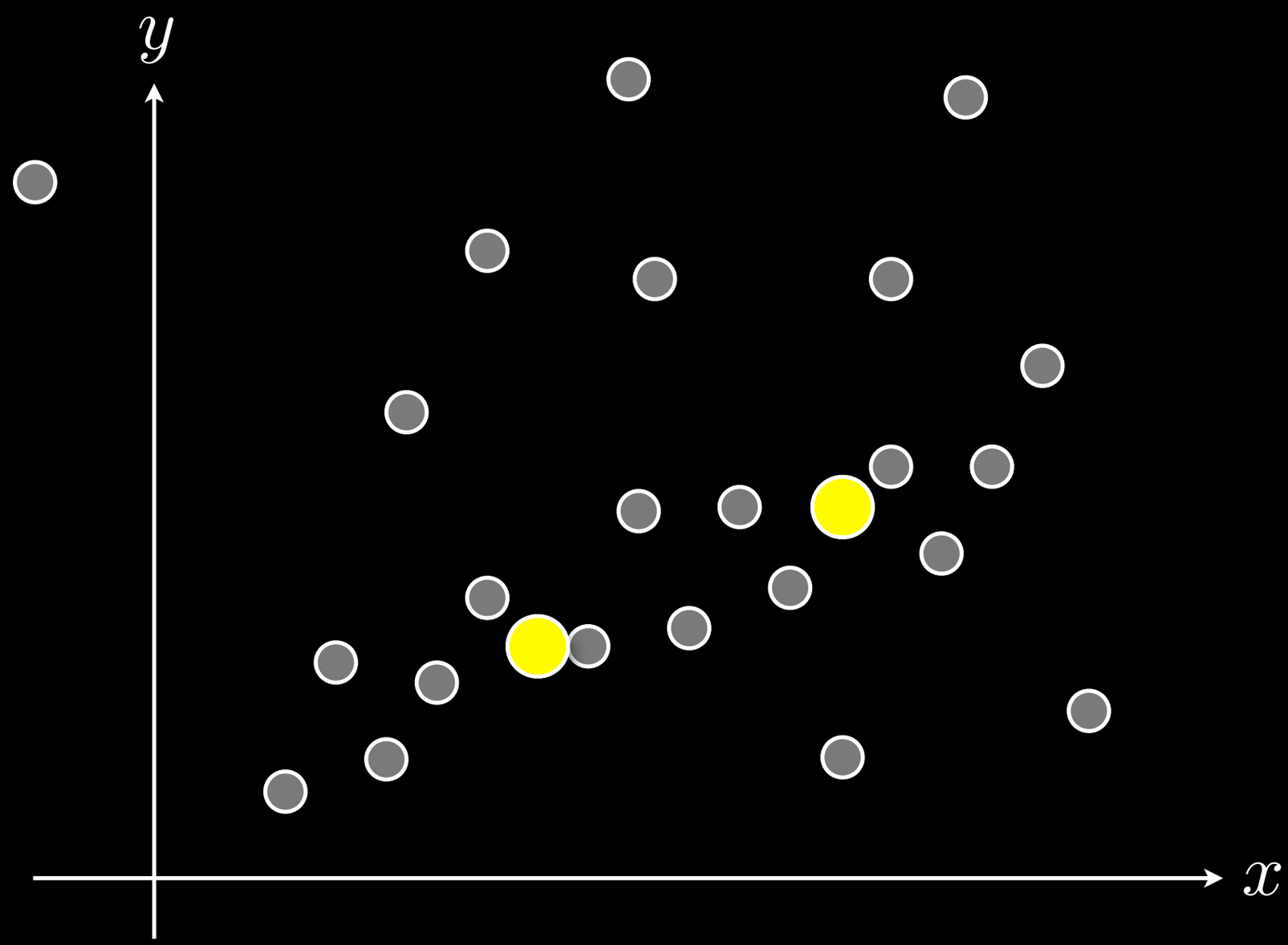

y
x



Select random **minimum** set of points to fit the model



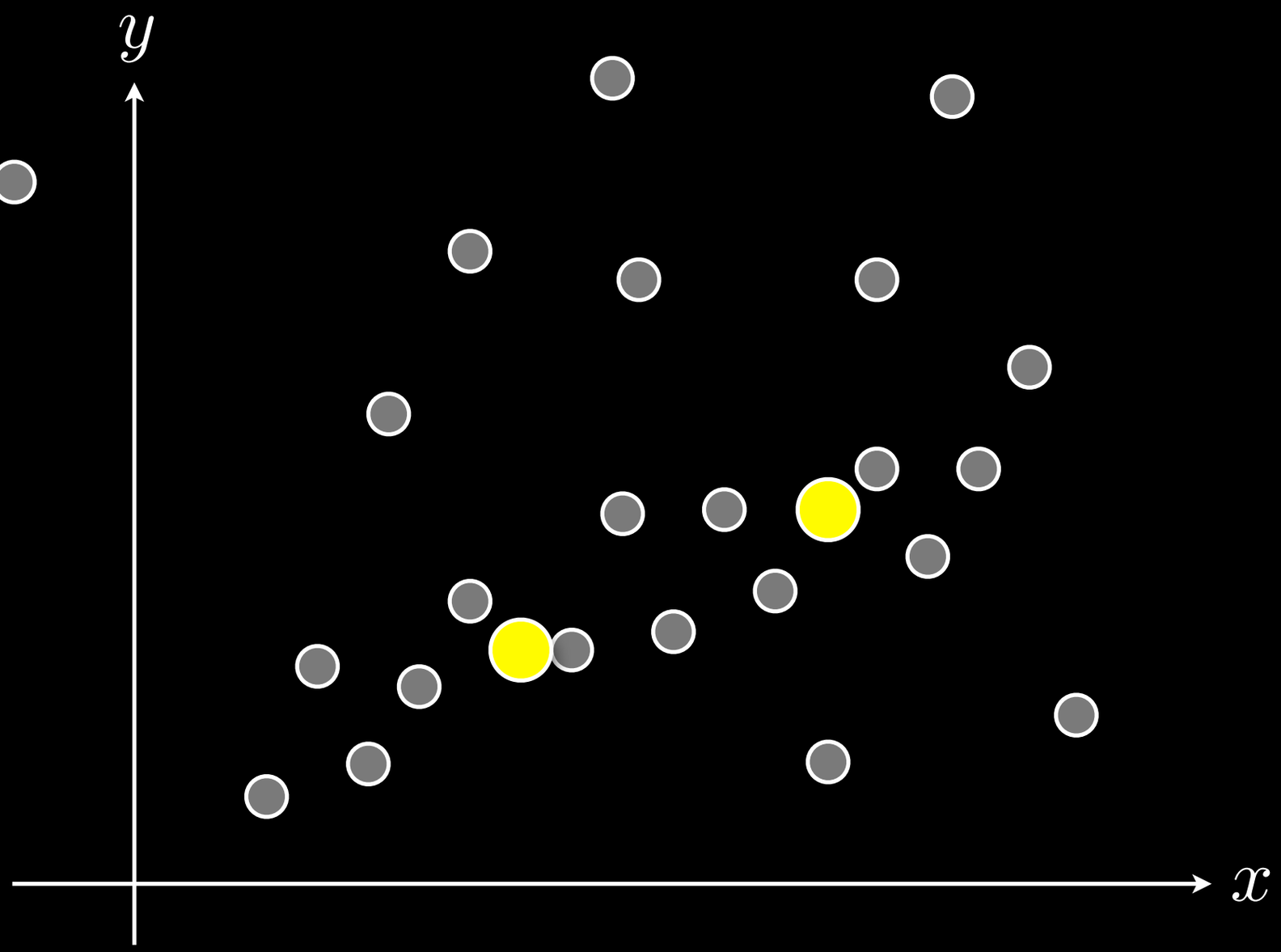
What is the size of the minimum set?

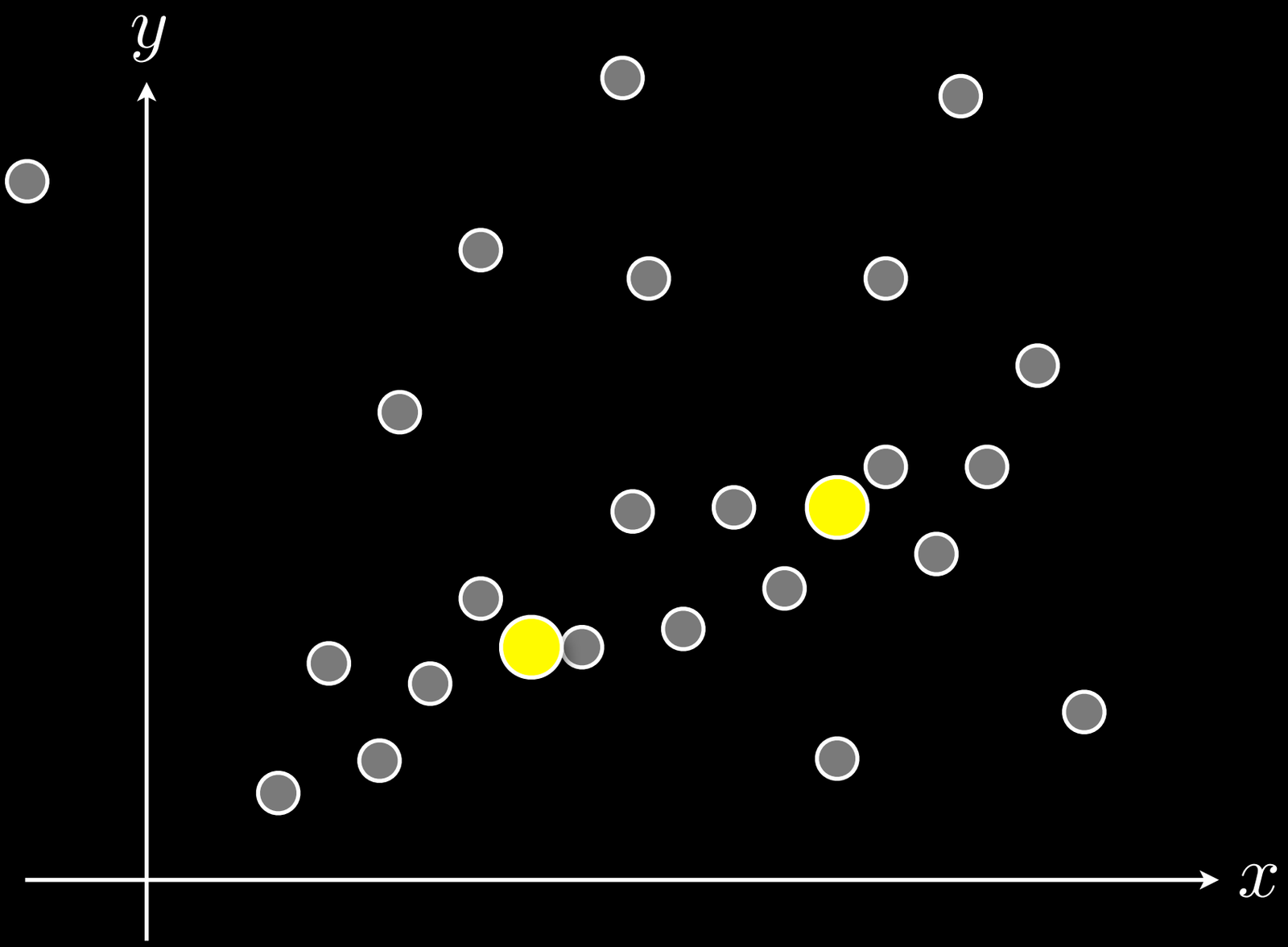


What is the size of the minimum set?

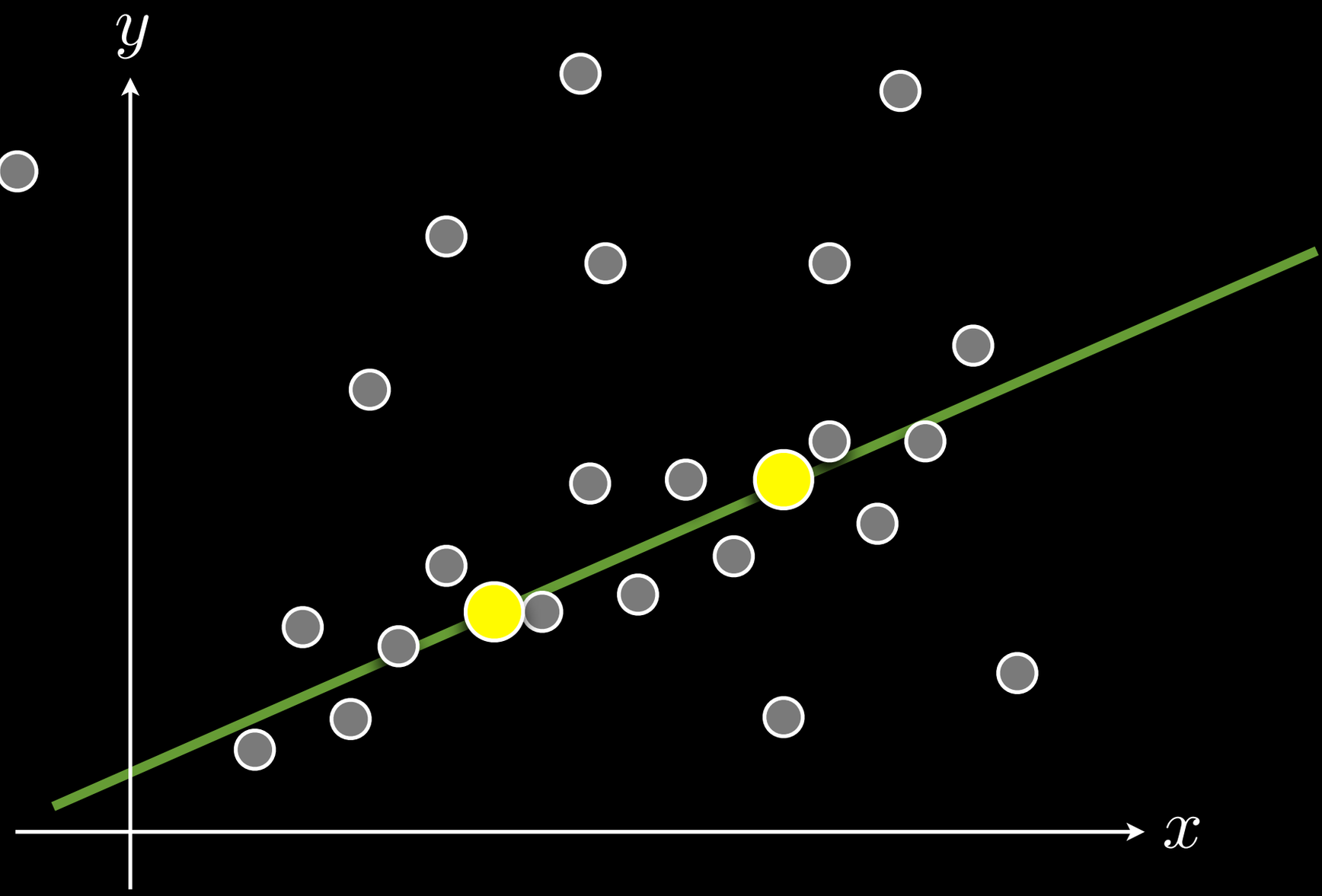
Step 2

Find best fitting line


y
x



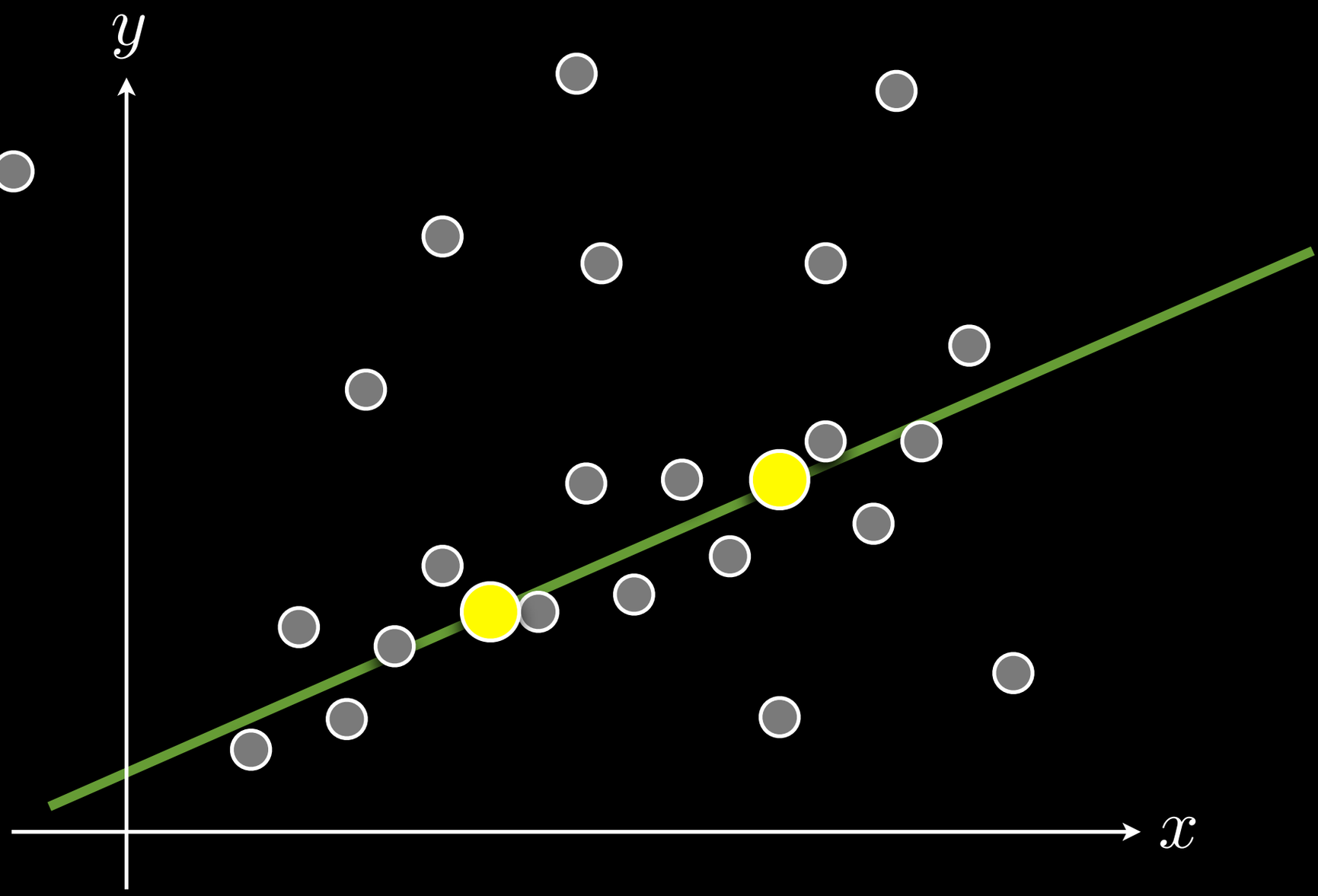
Compute the best fitting line

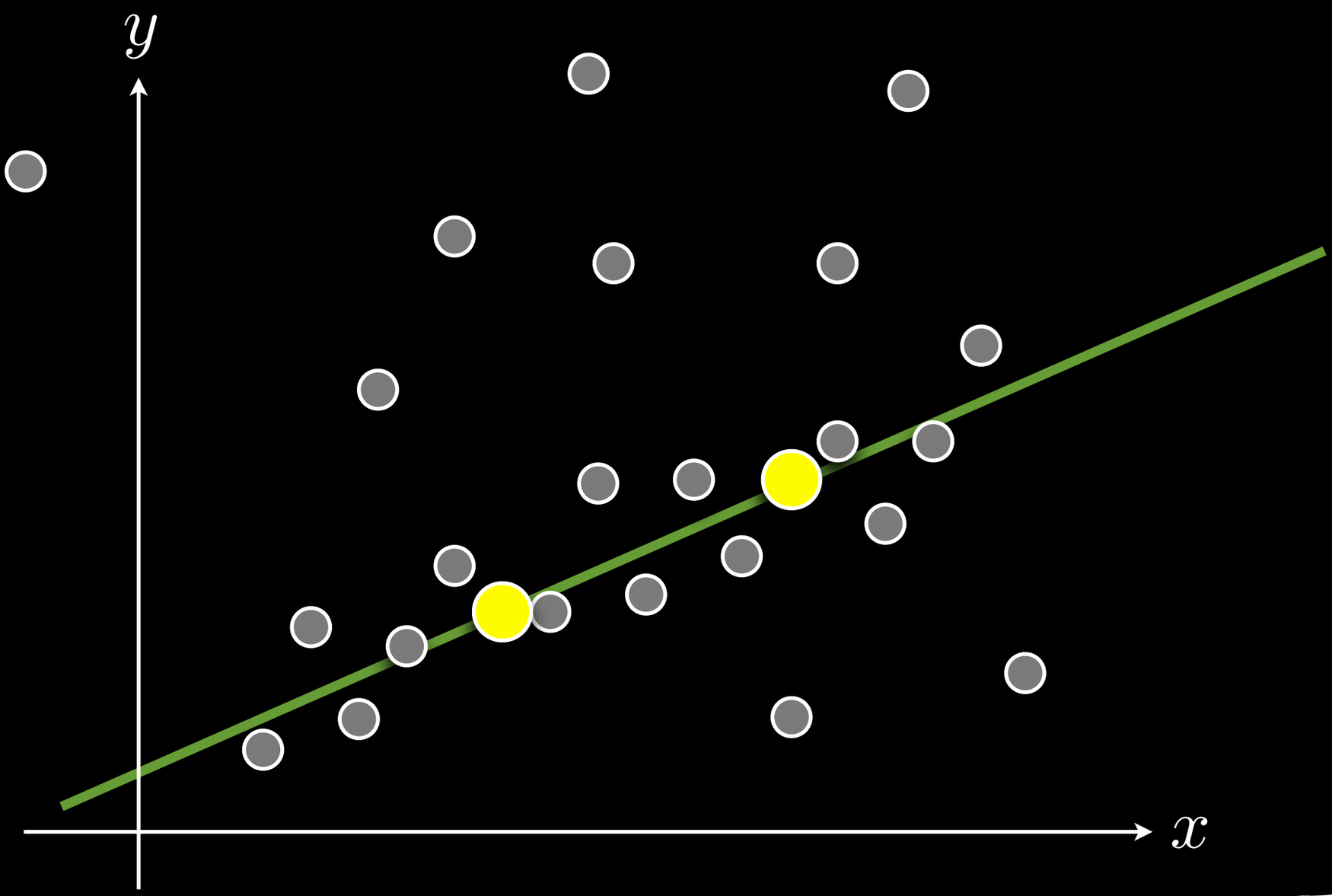


Compute the best fitting line

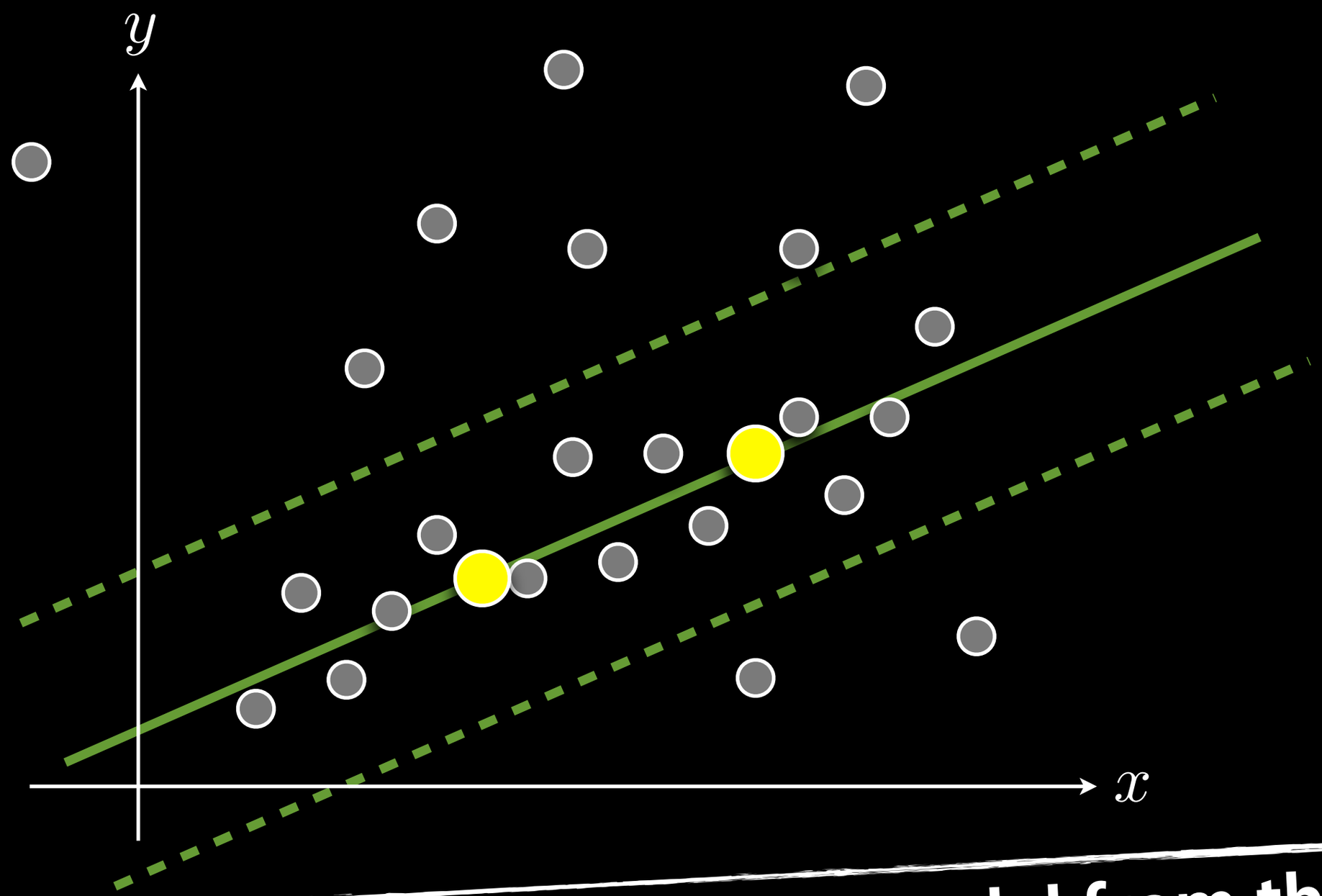
Step 3

Determine inliers

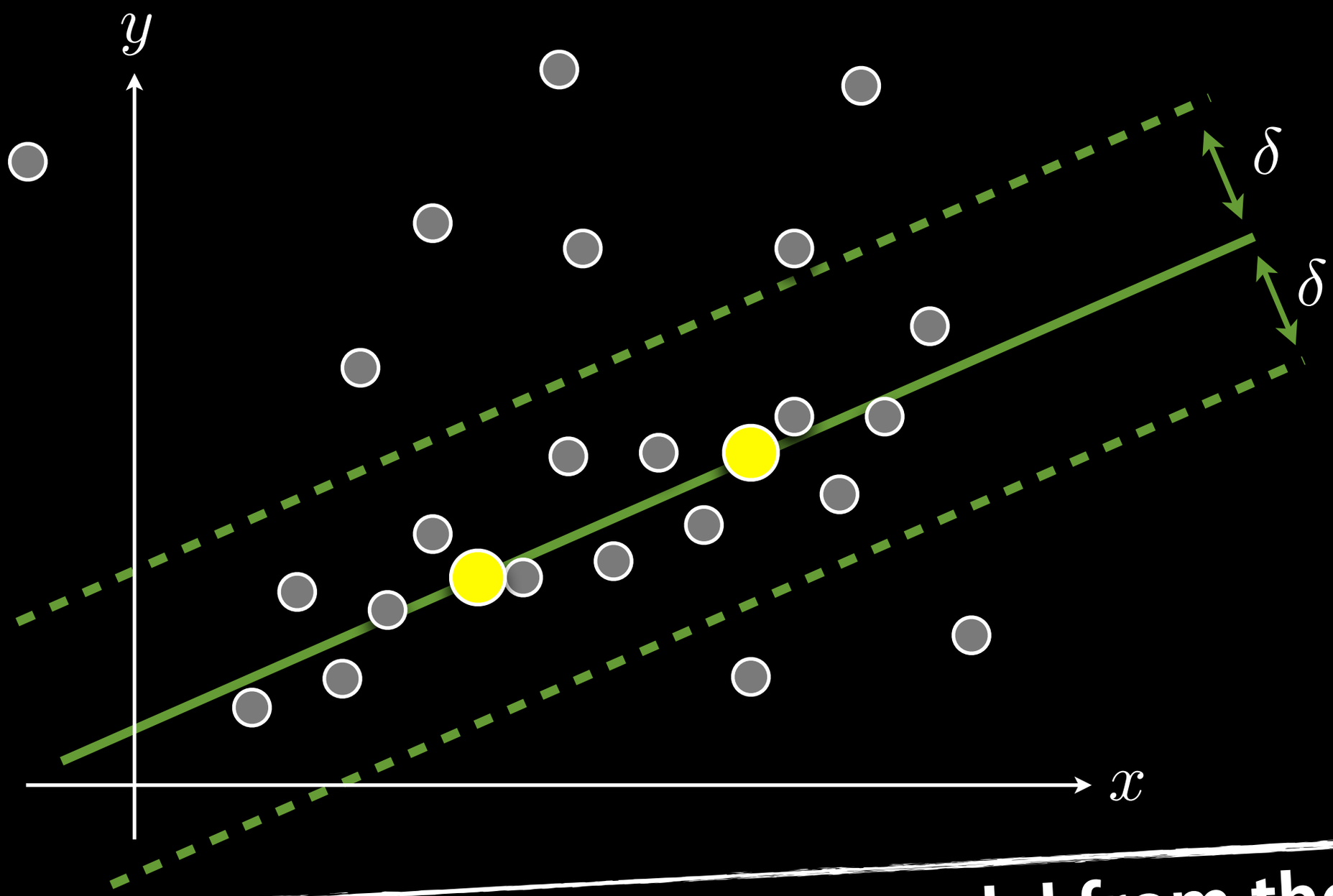

y
x



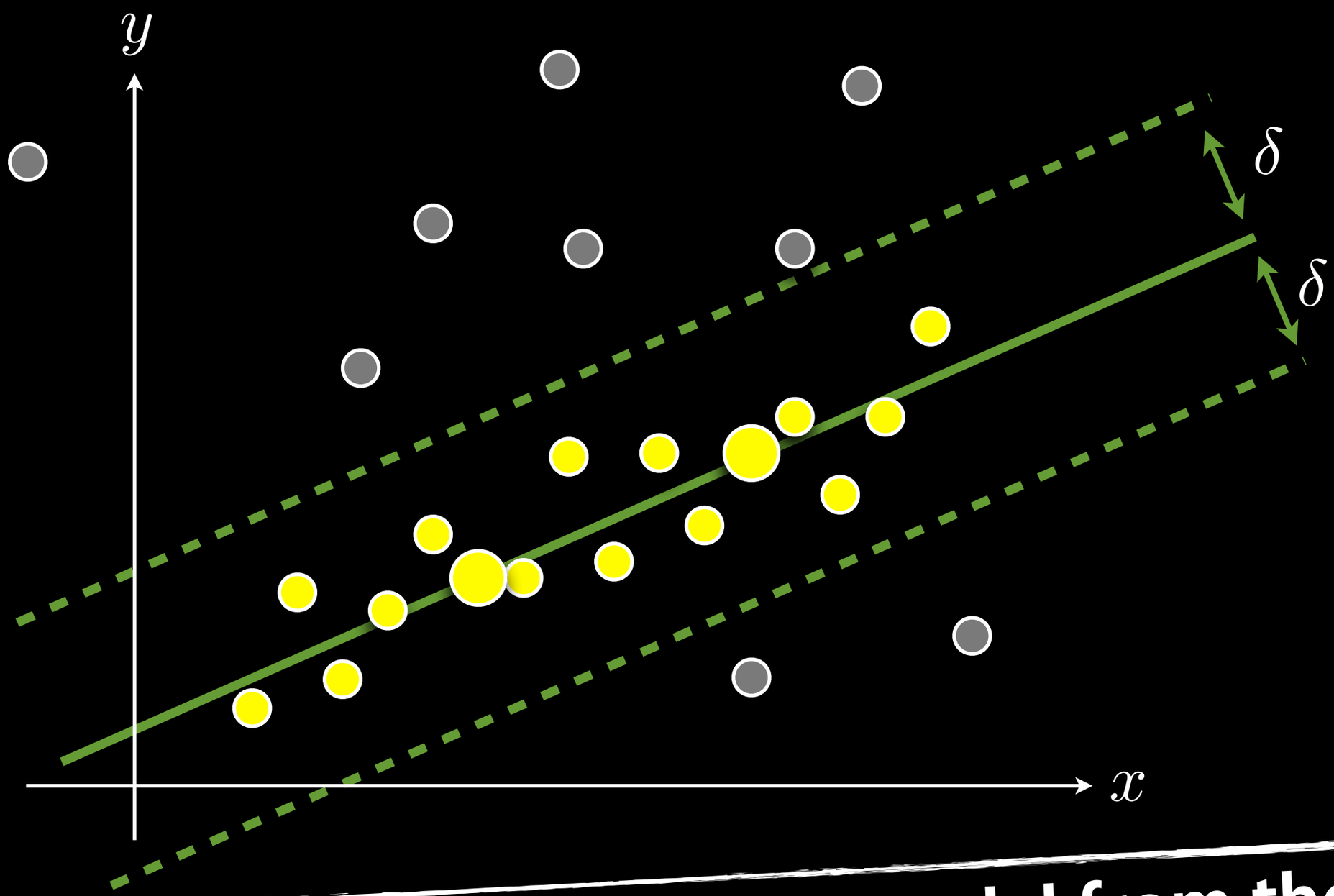
Compute the set of inliers to model from the dataset based on threshold



Compute the set of inliers to model from the dataset based on threshold



Compute the set of inliers to model from the dataset based on threshold



Compute the set of inliers to model from the dataset based on threshold

Step 4

Repeat steps 1-3 for N steps and estimate the final model using the sample with the most inliers

How many samples?

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

Confused?

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$e$$

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$1 - e$$

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$1 - e$$

Probability that choosing one point yields an inlier

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$(1 - e)^s$$

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$(1 - e)^s$$

Probability of choosing s inliers in a row

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$(1 - e)^s$$

Sample only contains inliers

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$1 - (1 - e)^s$$

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$1 - (1 - e)^s$$

Probability that one or more sample points are outliers

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$1 - (1 - e)^s$$

Sample is contaminated

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$(1-(1-e)^s)^N$$

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$(1-(1-e)^s)^N$$

Probability that N samples were contaminated

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$1-(1-(1-e)^s)^N$$

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$1-(1-(1-e)^s)^N$$

Probability at least one sample of s points contains only inliers

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$1-(1-(1-e)^s)^N$$

At least one sample was not contaminated

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$p = 1 - (1 - (1 - e)^s)^N$$

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$p = 1 - (1 - (1 - e)^s)^N$$

How to solve for N ?

$N =$ **number of samples**

$e =$ **probability that a point is an outlier**

$s =$ **number of points in a sample**

$p =$ **desired probability that we get a good sample**

$$p = 1 - (1 - (1 - e)^s)^N$$

$$N = \frac{\log(1 - p)}{\log(1 - (1 - e)^s)}$$

$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

Choose N such that with probability $p = 0.99$ at least one random sample is free from outliers

$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

Choose N such that with probability $p = 0.99$ at least one random sample is free from outliers

	proportion of outliers, e (%)						
s	5	10	20	25	30	40	50
2							
3							
4							
5							
6							
7							
8							

$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

Choose N such that with probability $p = 0.99$ at least one random sample is free from outliers

	proportion of outliers, e (%)						
s	5	10	20	25	30	40	50
2	2	3	5	6	7	11	17
3							
4							
5							
6							
7							
8							

$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

Choose N such that with probability $p = 0.99$ at least one random sample is free from outliers

	proportion of outliers, e (%)						
s	5	10	20	25	30	40	50
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4							
5							
6							
7							
8							

$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

Choose N such that with probability $p = 0.99$ at least one random sample is free from outliers

	proportion of outliers, e (%)						
s	5	10	20	25	30	40	50
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

PROS

PROS

Simple to implement

PROS

Simple to implement

Applicable in many diverse contexts

CONS

- Many parameters to tune

CONS

Many parameters to tune

Cannot be used if ratio of inliers/outliers is too small