

WELCOME TO EECS 189 DISC 2

yipeee yipeeee 

EXTRA:

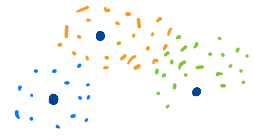
(more "mathy")

For a more in-depth[^] video version of problem 1, see my
sp26 recording: tinyurl.com/sp26-kmeans

Problem 1

K-Means Algorithm (Conceptual Overview)

Given we want to partition points into k clusters:

1. Initialize centroids (μ_k) 
2. Assign each data pt (x_i) to closest centroid
3. Update^{each} centroid position by taking MEAN of all data pts assigned to that cluster
4. Repeat steps 2 & 3 until convergence

Discussion 2



fig: a pelican on a bicycle
drawn by Qwen 3.8 27B

This week's resources:

- Matrix differentiation guide (written by Randal Barnes)
- Mathematics for Machine Learning
- 3Blue1Brown: Essence of Linear Algebra
- Simon Willison's pelican on a bicycle benchmark for LLMs

[link](#)
[link](#)
[playlist](#)
[link](#)

1 Proof of K-means Convergence

Consider the K-means algorithm applied to data points $\{x_n\}_{n=1}^N \subset \mathbb{R}^D$ with K clusters. Define binary indicator variables $r_{nk} \in \{0, 1\}$ with $\sum_{k=1}^K r_{nk} = 1$ for each n , and cluster centers $\{\mu_k\}_{k=1}^K$. The K-means objective is

$$J(\{r_{nk}\}, \{\mu_k\}) = \sum_{n=1}^N \sum_{k=1}^K r_{nk} \|x_n - \mu_k\|^2.$$

Handwritten notes:
 - Above the equation: "for every pt n" (green), "for every pt k" (green), "Euclidean distance" (green), "Euclidean distance" (green).
 - To the right: $r_{nk} = \begin{cases} 1 & \text{if pt } x_n \text{ belongs to cluster } k \\ 0 & \text{if pt } x_n \text{ does NOT belong to cluster } k \end{cases}$ (blue).
 - Above the sum: "Ex: $0 + 1 + \dots + 0 = 1$ " (red), " $\sum_{k=1}^K r_{nk} = r_{n1} + r_{n2} + \dots + r_{nK} = 1$ " (red).
 - To the right of the equation: $x_1, x_2, \dots, x_N$ (red).
 - Below the equation: "sum of pt values in cluster k" (green), "# pts in cluster k" (green).

The K-means algorithm alternates between

for every pt n

1. **Update assignments:** Set $r_{nk} = 1$ for the k that minimizes $\|x_n - \mu_k\|^2$ and $r_{nj} = 0$ for all $j \neq k$.

2. **Update means:** Set the cluster centers to $\mu_k = \frac{\sum_{n=1}^N r_{nk} x_n}{\sum_{n=1}^N r_{nk}}$ (for clusters with $\sum_{n=1}^N r_{nk} > 0$).
- Handwritten notes:
 - "for every cluster k" (green) next to the update means step.
 - "sum of pt values in cluster k" (green) pointing to the numerator.
 - "# pts in cluster k" (green) pointing to the denominator.

Prove that the K-means algorithm converges after a finite number of iterations. (Hint: It is sufficient to show that the objective is monotonically decreasing, and that there are a finite number of iterations).

① Show objective J is monotonically decreasing

(a) Show assignment update is monotonically decreasing

1. μ_k 's all stay constant ☺

For each x_n , we're essentially assigning x_n to the the closest cluster... aka for every pt x_n , we choose an $r_{nk}=1$ s.t. Eucl. distance $\|x_n - \mu_k\|^2$ is minimized... $r_{nk}\|x_n - \mu_k\|^2$ has to $\downarrow$; $\therefore$ our objective J HAS to decrease or stay the same.

(b) 2. x_n 's all stay constant ☺

show that mean update is monotonically decreasing

We set the cluster centers to the MEAN of pts in that cluster; as shown in lec, the center that minimizes the sum of distances IS the mean (Want a more in-depth explanation... check out sp26 vid ☺) $\rightarrow$ we're essentially $\underset{\mu_k}{\operatorname{argmin}} J_k(\mu) = \sum_{n:r_{nk}=1} \|x_n - \mu_k\|^2$

uniquely minimized by mean of assigned pts...

$\therefore$ Updating means NEVER

$\uparrow J$

$$\mu_k^* = \frac{\sum_{n=1}^N r_{nk} x_n}{\sum_{n=1}^N r_{nk}}$$

$\therefore$ Bc we have shown (a) AND (b), objective must be monotonically $\downarrow$ ☺

② Show finite # iterations in k-means algo;

Restate: At some pt, are assignments will stop changing...

Each pt n has k choices of clusters... $k \cdot k \cdot k \dots k = k^n$ assignments

$\therefore$ Bc J monotonically decreases during every pt assignment + mean update, and there are only finite # assignments, algo will reach a pt in which nothing changes $\rightarrow$ k-means converge after finite # iterations.

Q: Does K-means guarantee GLOBAL convergence?

NO!!! K-means, are HIGHLY dependent on where the centers were initially ^{algo's final assignments} initiated!

Problem 2 Notes:

1) Symmetric Matrix: $S = S^T$;

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}^T \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ symmetric!}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}^T \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{bmatrix} \text{ NOT symmetric!!}$$

2) Properties of symmetric matrices

- Real eigenvalues
- Eigenvectors corresponding to ^{distinct} eigenvalues are orthogonal (i.e. perpendicular)
- They're square!
- They're always diagonalizable

3) What's an **eigenvector**? An **eigenvalue**? A **singular val**?

$$Ax = \lambda x$$

$$\sigma = \sqrt{\lambda}$$

- **Eigenvector**: vector that's only scaled in a linear transformation (hence the constant) by a scalar value

- **Eigenvalue**: the scalar (λ) that represents how much a vector has been stretched or shrunk

- **Singular val**: square root of **eigenval**

4) Why does the above matter LOL?

- **PCA/Dimension Reduction**: Which direction does data vary most / is most important?

$$SVD: M = U \Sigma V^T$$

rotation scale (singular vals) rotation

(OR: which direction in feature space has greatest variance?)
← 2nd order derivs

- Eigenvalues of the Hessian matrix $\rightarrow$ Curvature of loss func (how steep change in direction of eigenvectors)

5) How to compute eigenvals? (And hence eigenvectors thru $Ax = \lambda x$, singular vals thru $\sigma = \sqrt{\lambda}$)

i) given B , find $A = B^T B$

ii) solve $Ax = \lambda x$ by setting $\det(A - \lambda I) = 0$

iii) Solve for λ_i

6) How to do PCA, what is significance? We shall see in this problem har har har 😊

2 PC-yay!

(a) A warehouse robot records the following four two-dimensional displacement vectors:

$$\mathbf{x}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} -1 \\ -2 \end{bmatrix}, \quad \mathbf{x}_4 = \begin{bmatrix} -2 \\ -1 \end{bmatrix}.$$

Each coordinate measures displacement in meters. We will use PCA to find the direction in which the robot's motion varies the most.

(i) Compute the sample mean

$$\mu = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i. \quad N=4$$

$$\mu = \frac{1}{4} (\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 + \mathbf{x}_4) = \frac{1}{4} \left(\begin{bmatrix} 2 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} -1 \\ -2 \end{bmatrix} + \begin{bmatrix} -2 \\ -1 \end{bmatrix} \right)$$

$$= \frac{1}{4} \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

(ii) Using the normalization $1/N$, compute the empirical covariance matrix

$$C = \frac{1}{N} \sum_{i=1}^N (\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^\top.$$

$$C = \frac{1}{4} \sum_{i=1}^N (\mathbf{x}_i)(\mathbf{x}_i)^\top = \frac{1}{4} [\mathbf{x}_1 \mathbf{x}_1^\top + \mathbf{x}_2 \mathbf{x}_2^\top + \mathbf{x}_3 \mathbf{x}_3^\top + \mathbf{x}_4 \mathbf{x}_4^\top]$$

$$= \frac{1}{4} \left(\begin{bmatrix} 2 \\ 1 \end{bmatrix} \begin{bmatrix} 2 & 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 2 \end{bmatrix} + \begin{bmatrix} -1 \\ -2 \end{bmatrix} \begin{bmatrix} -1 & -2 \end{bmatrix} + \begin{bmatrix} -2 \\ -1 \end{bmatrix} \begin{bmatrix} -2 & -1 \end{bmatrix} \right)$$

$$= \frac{1}{4} \left(\begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} + \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} + \begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} \right) = \begin{bmatrix} 5/2 & 2 \\ 2 & 5/2 \end{bmatrix}$$

(iii) Find the eigenvalues of C and a corresponding unit eigenvector for each eigenvalue.

$$(C - \lambda I) \mathbf{v} = \mathbf{0}$$

$$C \mathbf{v} = \lambda \mathbf{v}$$

$$\det(C - \lambda I) = 0$$

$$\lambda I = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}; \quad C - \lambda I$$

$$\begin{vmatrix} 5/2 - \lambda & 2 \\ 2 & 5/2 - \lambda \end{vmatrix} = (5/2 - \lambda)^2 - 4 = \lambda^2 - 10/2 \lambda + 9/4 = 0$$

$$(\lambda - 9/2)(\lambda - 1/2) = 0$$

$$\lambda_1 = 9/2$$

$$\lambda_2 = 1/2$$

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

(iv) The plot below shows the four robot displacements and several contours of equal covariance-scaled distance from the mean. Starting at the mean, draw and label the two eigenvector directions $\mathbf{v}_1$ and $\mathbf{v}_2$. Make the length of each arrow proportional to $\sqrt{\lambda_1}$ and $\sqrt{\lambda_2}$, respectively, using the same scale factor for both arrows. Which eigenvector is the first principal component? Describe its physical meaning for the robot.

2a)iii Cont'd): $Cv = \lambda v \Rightarrow (C - \lambda I)v = 0$

$\lambda_1 = 9/2$
 $(C - 9/2 I) = 0 \Rightarrow \begin{bmatrix} 5/2 - 9/2 & 2 \\ 2 & 5/2 - 9/2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = 0$

$\begin{bmatrix} -2 & 2 \\ 2 & -2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = 0$ $-2a + 2b = 0$
 $2a - 2b = 0$

$2a = 2b$
 $a = b = 1$

$v = \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Rightarrow v_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$\sqrt{1^2 + 1^2} = \sqrt{2}$

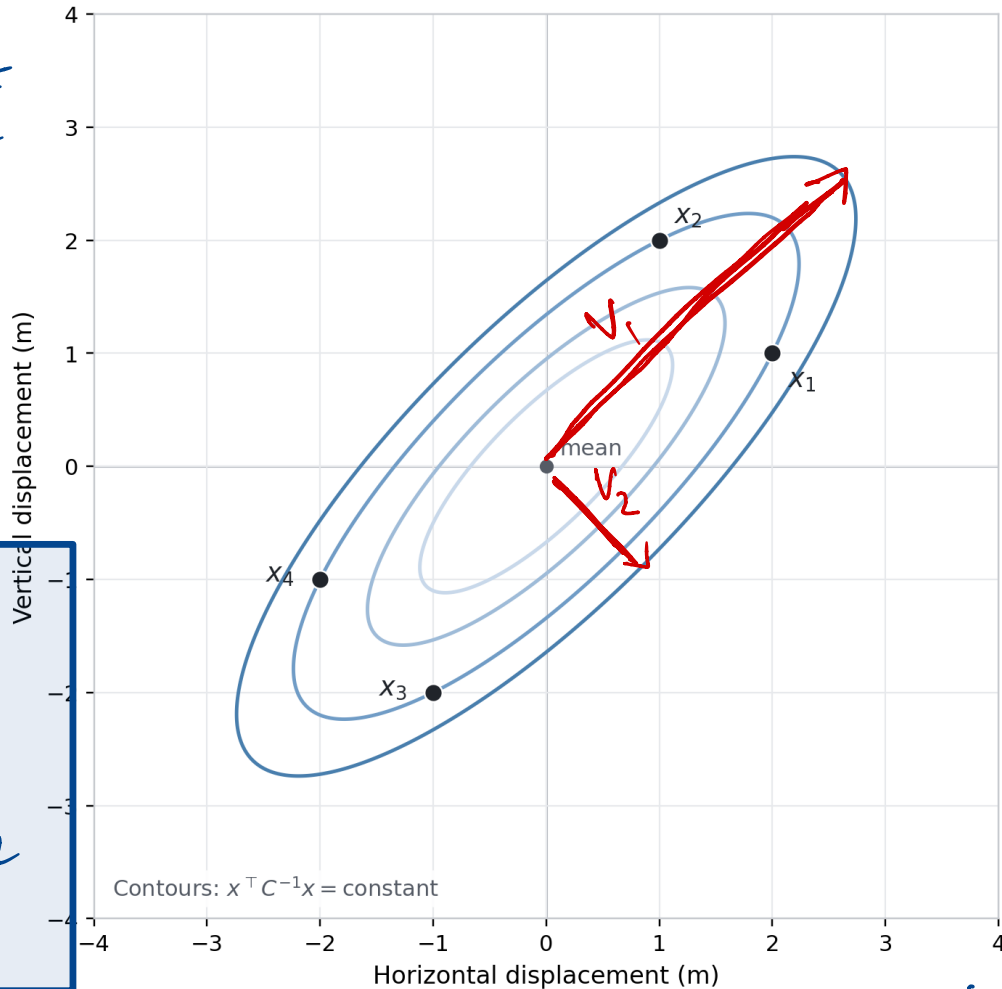
For $\lambda_1 = 9/2$,
 $v_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

INTEREST OF TIME: Try eigenvec of $\lambda_2 = 1/2$ yourself!
(Same procedure as above... solve for $(C - \lambda I)v = 0$)

•
•
•

For $\lambda = 1/2$, $v_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

Robot Displacements and Covariance Contours



$$\lambda_1 = 9/2; \sqrt{\lambda_1} = \frac{3}{\sqrt{2}}$$

$$\lambda_2 = 1/2; \sqrt{\lambda_2} = \frac{1}{\sqrt{2}}$$

$$\frac{\sqrt{\lambda_1}}{\sqrt{\lambda_2}} = \frac{3/\sqrt{2}}{1/\sqrt{2}} = 3$$

$\therefore v_1$ vector has $\uparrow$ variance, and v_1 is 3 times the size of v_2

v_1 is the first principal component $\Rightarrow$ our robot motion varies most in diagonal direction where hztl + vtl displacement change together!

(v) Project each centered displacement onto the first principal component by computing $z_i = \mathbf{v}_1^T (\mathbf{x}_i - \boldsymbol{\mu})$.

$$z_1 = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \frac{3}{\sqrt{2}}$$

$$z_2 = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \frac{3}{\sqrt{2}}$$

$$v_1 = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} \quad z_3 = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} -1 \\ -2 \end{bmatrix} = -\frac{3}{\sqrt{2}}$$

$$z_4 = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} -2 \\ -1 \end{bmatrix} = -\frac{3}{\sqrt{2}}$$

(vi) Suppose the robot stores only the one-dimensional values z_i . What information about its motion is lost? Give a specific example from the four recorded displacements.

Lost motion in direction of 2nd principal component v_2 . For ex: x_1 and x_2 (which used to be different) both became $3/\sqrt{2}$ (z_1 and z_2).

(vii) The robot later records a new displacement

$$\mathbf{x}_{\text{new}} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

$$\mathbf{v}_1 = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

Project this displacement onto the first principal component, then reconstruct its two-dimensional approximation using

$$\hat{\mathbf{x}}_{\text{new}} = \boldsymbol{\mu} + \left(\mathbf{v}_1^\top (\mathbf{x}_{\text{new}} - \boldsymbol{\mu}) \right) \mathbf{v}_1.$$

In a safety-critical robot, why might replacing $\mathbf{x}_{\text{new}}$ with this approximation be problematic?

$$\mathbf{v}_1^\top \mathbf{x}_{\text{new}} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 3 \\ 1 \end{bmatrix} = \frac{4 \cdot \sqrt{2}}{\sqrt{2} \cdot \sqrt{2}} = \underline{2\sqrt{2}}$$

$$\hat{\mathbf{x}}_{\text{new}} = \mathbf{0} + \left(\mathbf{v}_1^\top \mathbf{x}_{\text{new}} \right) \mathbf{v}_1 = \underline{2\sqrt{2}} \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

Discarded displacement: $\mathbf{x}_{\text{new}} - \hat{\mathbf{x}}_{\text{new}} = \begin{bmatrix} 3 \\ 1 \end{bmatrix} - \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$
 Lost important info (i.e. the entire direction alongside $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$... robot could misjudge position relative to an obstacle.

Problem 3 Notes: ☺

1. Partial Derivatives: $f(x, y) = 2x + 3y$

$\frac{\partial f}{\partial x} = 2$
 $\frac{\partial f}{\partial y} = 3$

2. Chain Rule: $\frac{\partial}{\partial x} f(g(x) + h(x)) = f'(g(x) + h(x)) (g'(x) + h'(x))$

$$= \frac{\partial f}{\partial g} \frac{dg}{dx} + \frac{\partial f}{\partial h} \frac{dh}{dx}$$

3. Derivative Rules (multiplication, quotient etc.)

$$(fg)' = f'g + fg'$$

$$\left(\frac{f}{g}\right)' = \frac{f'g - fg'}{g^2}$$

4. If we have $f(\vec{x})$ where x is a vector, taking gradient of f w/respect to $\vec{x}$ is equivalent to taking the partial derivative w/respect to every component $x_1 \dots x_n$

$$\text{If } \vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \nabla_x f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

3 Calculus

(a) Consider the function

$$f(x, y) = \sigma(ax + by),$$

where $a, b \in \mathbb{R}$ and $\sigma(t) = \frac{1}{1 + e^{-t}}$ for $t \in \mathbb{R}$.

(i) Show that $\frac{d\sigma}{dt} = \sigma(t)(1 - \sigma(t))$

$$\frac{d\sigma}{dt} = -\frac{1}{(1+e^{-t})^2} (e^{-t})(-1) = \frac{e^{-t}}{(1+e^{-t})^2} = \frac{1}{(1+e^{-t})} \cdot \frac{e^{-t}}{(1+e^{-t})} = \underbrace{\sigma(t)}_{\sigma(t)} \underbrace{(1 - \sigma(t))}_{1 - \sigma(t)}$$

(ii) Using the result you showed in part (i) and the chain rule, compute $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$.

$$a) \frac{\partial f}{\partial x} = \underbrace{\sigma'(ax+by)}_{\sigma'(ax+by)} (a) = \underbrace{a \sigma(ax+by)(1 - \sigma(ax+by))}_{a \sigma(ax+by)(1 - \sigma(ax+by))}$$

$$b) \frac{\partial f}{\partial y} = \underbrace{\sigma'(ax+by)}_{\sigma'(ax+by)} (b) = \underbrace{b \sigma(ax+by)(1 - \sigma(ax+by))}_{b \sigma(ax+by)(1 - \sigma(ax+by))}$$

(b) For $\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$, define

$$g(\mathbf{x}) = \sum_{j=1}^n x_j^2.$$

Compute the partial derivative $\frac{\partial g}{\partial x_i}$ for a generic coordinate $i \in \{1, \dots, n\}$.

$$\frac{\partial g}{\partial x_i} = \frac{d}{dx_i} (\cancel{x_1^2} + \cancel{x_2^2} + \dots + \underbrace{x_i^2}_{\text{red wavy}} + \dots + \cancel{x_n^2}) = \boxed{2x_i}$$

(c) Let $\mathbf{w} \in \mathbb{R}^n$ be a constant vector and define the scalar function

$$h(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} = \sum_{j=1}^n w_j x_j.$$

Compute $\frac{\partial h}{\partial x_i}$ for a generic coordinate $i \in \{1, \dots, n\}$.

$$\frac{\partial h}{\partial x_i} = \frac{d}{dx_i} (\cancel{w_1 x_1} + \cancel{w_2 x_2} + \dots + \underbrace{w_i x_i}_{\text{red wavy}} + \dots + \cancel{w_n x_n}) = \boxed{w_i}$$

(d) Let $\mathbf{w} \in \mathbb{R}^n$ be a constant vector and define the scalar function

$$\ell(\mathbf{x}) = \frac{1}{2} \sum_{j=1}^n x_j^2 - \mathbf{w}^\top \mathbf{x} = \frac{1}{2} g(\mathbf{x}) - h(\mathbf{x}).$$

In single-variable calculus, we can minimize/maximize a function $f(x)$ using its derivatives. Here ℓ takes a whole vector $\mathbf{x}$ as input instead of a single number, but we can reuse that same idea one coordinate at a time.

(i) Using your answers to parts (b) and (c), compute $\frac{\partial \ell}{\partial x_i}$ for a generic coordinate $i \in \{1, \dots, n\}$. Stacking these n partial derivatives into a vector gives the *gradient*

$$\nabla \ell(\mathbf{x}) = \begin{bmatrix} \frac{\partial \ell}{\partial x_1} \\ \dots \\ \frac{\partial \ell}{\partial x_n} \end{bmatrix}.$$

$$\frac{\partial \ell}{\partial x_i} = \frac{d}{dx_i} \left(\frac{1}{2} g(x_i) - h(x_i) \right) = \frac{1}{2} (2x_i) - w_i = \boxed{x_i - w_i}$$

$$\nabla \ell(\mathbf{x}) = \begin{bmatrix} \frac{\partial \ell}{\partial x_1} \\ \frac{\partial \ell}{\partial x_2} \\ \vdots \\ \frac{\partial \ell}{\partial x_n} \end{bmatrix} = \begin{bmatrix} x_1 - w_1 \\ x_2 - w_2 \\ \vdots \\ x_n - w_n \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} - \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \boxed{\vec{x} - \vec{w}}$$

- (ii) Recall from single-variable calculus that we find values of x that minimize/maximize $f(x)$ by solving $\frac{df}{dx} = 0$; the solutions are called *critical points*. In vector-calculus, a critical point of ℓ is a vector $\mathbf{x}$ where the partials $\frac{\partial \ell}{\partial x_i} = 0$ for all i components of $\mathbf{x}$ (equivalently $\nabla \ell(\mathbf{x}) = \mathbf{0}$).

Set all n partial derivatives equal to zero and solve for the minimizer $\mathbf{x}^*$. (Hint: the i -th component of $\mathbf{x}^*$ is the value of x_i that makes $\frac{\partial \ell}{\partial x_i} = 0$; solve for one coordinate and then stack each component into the vector $\mathbf{x}^*$)

$$\frac{\partial \ell}{\partial x_i} = x_i - w_i = 0; \quad x_i = w_i$$

$$\nabla \ell(\mathbf{x}) = \vec{x} - \vec{w} = \mathbf{0}; \quad \boxed{\vec{x}^* = \vec{w}}$$

- (iii) In single-variable calculus, once we have a critical point we check the sign of the second derivative: $f''(x) > 0$ means the function f is concave up, so the point is a minimum. Use this same second-derivative idea to argue that $\mathbf{x}^*$ is a minimum, and not a maximum or a saddle point.

$$\frac{\partial^2 \ell}{\partial x_i^2} = \frac{\partial}{\partial x_i} \left[\frac{\partial \ell}{\partial x_i} \right] = \frac{\partial}{\partial x_i} (x_i - w_i) = 1$$

$$\nabla^2 \ell = \frac{\partial}{\partial \mathbf{x}} \left(\frac{\partial \ell}{\partial \mathbf{x}} \right) = \begin{bmatrix} \frac{\partial}{\partial x_1} (x_1 - w_1) \\ \frac{\partial}{\partial x_2} (x_2 - w_2) \\ \vdots \\ \frac{\partial}{\partial x_n} (x_n - w_n) \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \vec{1} > \vec{0}$$

$\therefore$ every term's $\frac{\partial^2 \ell}{\partial x_i^2} > 0$, so $\mathbf{x}^*$ is a minimum.