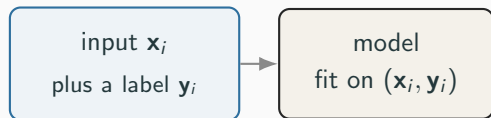


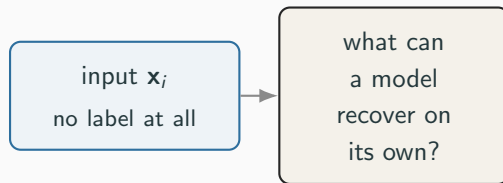
Unsupervised Learning

What can we learn with no labels at all?

Every model so far needed labels; today we drop them



Linear/logistic regression, the Perceptron, SVM: every model this semester needed a human to attach $\mathbf{y}_i$ to every $\mathbf{x}_i$.



All that remains today: unlabeled data $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$. Most of the data in the world looks like this.

"If intelligence is a cake, the bulk of the cake is unsupervised learning, the icing on the cake is supervised learning, and the cherry on the cake is reinforcement learning." — Yann LeCun, NIPS 2016 keynote

Two questions unlabeled data can answer, and today's plan

Can it be described more compactly? Fewer numbers per point, without losing what matters.
dimensionality reduction

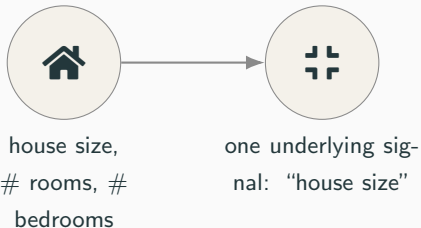
Does it have group structure? Distinct sources, with nothing in the data saying which point came from which.
clustering

1. **PCA:** Lecture 03's feature engineering run in reverse — project *down*, not up, to compress.
2. **K-means:** the simplest clustering algorithm, with K-means++ seeding.
3. **Expectation-Maximization:** the general algorithm underneath K-means: alternate between inferring hidden assignments and refitting parameters.
4. **EM for Gaussian Mixture Models:** EM on a model that gives clusters an explicit shape; K-means falls out as a restricted special case.

1. Principal Component Analysis

Compressing without losing what matters

Motivation: correlated features span a low-dimensional subspace



Three numbers that move together, all roughly measuring the same thing.

Ten related exam scores might really vary along one or two axes (“how strong is this student,” “theory vs. programming”), with the other eight dimensions mostly repeating that same signal plus noise.

Lecture 03’s feature engineering projected data *up*, into $\phi(\mathbf{x})$, on purpose, for expressivity. PCA projects *down*, on purpose, to compress.

PCA in one line: find the directions along which the data varies the **most** — automatically, no labels, no hand-picked redundancy. When features are correlated, most of what varies lives in a lower-dimensional subspace.

Setup: center the data, then formalize “varies the most”

Data $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$. PCA first **centers**: let $\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$ and work with $\mathbf{x}_i - \bar{\mathbf{x}}$ throughout.

without centering, the direction of largest raw squared magnitude just points toward wherever the data happens to sit in $\mathbb{R}^d$ — nothing to do with the data's *shape*

Step 1. Find a unit vector $\mathbf{v} \in \mathbb{R}^d$ ($\|\mathbf{v}\| = 1$) so that the projection $\mathbf{v}^\top (\mathbf{x}_i - \bar{\mathbf{x}})$ preserves as much spread as possible:

$$\max_{\|\mathbf{v}\|=1} \text{Var}(\mathbf{v}^\top (\mathbf{x} - \bar{\mathbf{x}})) = \max_{\|\mathbf{v}\|=1} \frac{1}{n} \sum_{i=1}^n (\mathbf{v}^\top (\mathbf{x}_i - \bar{\mathbf{x}}))^2$$

Covariance matrix, Lagrange multipliers, eigenvector equation

Step 2. Expand the square and pull $\mathbf{v}$ out of the sum:

$$\frac{1}{n} \sum_{i=1}^n (\mathbf{v}^\top (\mathbf{x}_i - \bar{\mathbf{x}}))^2 = \mathbf{v}^\top \underbrace{\left(\frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top \right)}_{=:\Sigma} \mathbf{v}$$

$\Sigma \in \mathbb{R}^{d \times d}$ is the **covariance matrix**: symmetric and PSD (an average of outer products, each PSD).

Step 3. The problem is now $\max_{\|\mathbf{v}\|=1} \mathbf{v}^\top \Sigma \mathbf{v}$, a constrained optimization in $\mathbf{v}$ alone.
Lagrangian and stationarity:

$$\begin{aligned} \mathcal{L}(\mathbf{v}, \lambda) &= \mathbf{v}^\top \Sigma \mathbf{v} - \lambda(\mathbf{v}^\top \mathbf{v} - 1) \\ \nabla_{\mathbf{v}} \mathcal{L} &= 2\Sigma \mathbf{v} - 2\lambda \mathbf{v} = 0 \end{aligned}$$

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

The optimal direction is the top eigenvector; the rest follow

Step 4. Any optimal unit vector must be an eigenvector of Σ , and λ is its eigenvalue. At a stationary point,

$$\mathbf{v}^\top \Sigma \mathbf{v} = \mathbf{v}^\top (\lambda \mathbf{v}) = \lambda \|\mathbf{v}\|^2 = \lambda,$$

so the variance captured *equals* the eigenvalue. Σ symmetric $\Rightarrow d$ real eigenvalues $\lambda_1 \geq \dots \geq \lambda_d \geq 0$ with an orthonormal eigenbasis (spectral theorem). Maximizing λ over this set picks λ_1 : the **first principal component** $\mathbf{v}_1$.

Step 5. Repeat with the added constraint $\mathbf{v} \perp \mathbf{v}_1$: the same stationary-point argument again forces $\Sigma \mathbf{v} = \lambda \mathbf{v}$, now maximized by the *second*-largest eigenvalue λ_2 .

inducting on this argument recovers the full eigenbasis in decreasing eigenvalue order — the Courant–Fischer (Ky Fan) characterization, not re-derived in full generality here

Top- k PCA, and how to compute it

Top- k PCA. Given centered data with covariance Σ , the top- k principal components are the eigenvectors $\mathbf{v}_1, \dots, \mathbf{v}_k$ of Σ with the k largest eigenvalues $\lambda_1 \geq \dots \geq \lambda_k$.

$$\mathbf{x} \mapsto (\mathbf{v}_1^\top (\mathbf{x} - \bar{\mathbf{x}}), \dots, \mathbf{v}_k^\top (\mathbf{x} - \bar{\mathbf{x}})) \in \mathbb{R}^k$$

The linear projection into k dimensions that captures the most variance of any k -dimensional linear projection.

Computing it. Explicitly forming Σ and eigendecomposing it costs $O(nd^2 + d^3)$.

for large n, d : SVD of the centered data matrix directly, $X_c = UDV^\top$ — columns of V are the components, $\lambda_j = D_j^2/n$, without ever forming Σ ; usually more numerically stable, too

Explained variance and the scree plot: a 5D example

Total variance: $\text{tr}(\Sigma) = \sum_{j=1}^d \lambda_j$ (trace = sum of eigenvalues).

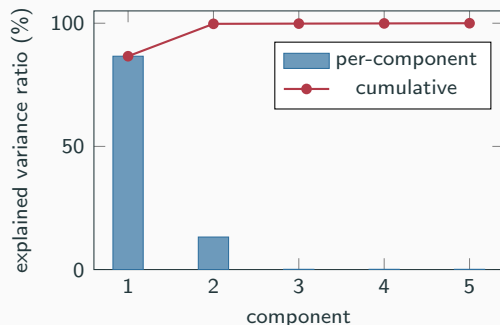
explained variance

ratio: $\lambda_j / \sum_l \lambda_l$

cumulative through

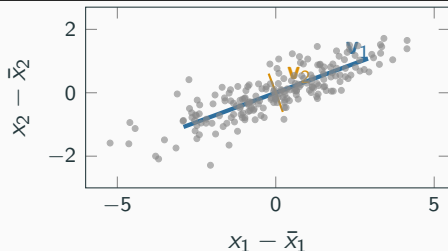
top k : $\sum_{j \leq k} \lambda_j / \sum_l \lambda_l$

Plotting explained variance ratio against component index gives a **scree plot**; it guides how many components to keep.



Data with 2 true latent directions in $\mathbb{R}^5$: eigenvalues 93.31, 14.22, 0.095, 0.090, 0.082 — sharp elbow after component 2, 99.75% cumulative.

Verified numerically: a 2D scatter, its axes, its eigenvalues



$$\Sigma = \begin{pmatrix} 3.3297 & 1.1726 \\ 1.1726 & 0.5964 \end{pmatrix}$$

$$\lambda_1 = 3.7639, \quad \lambda_2 = 0.1622$$

95.87%/4.13% explained

$k = 1$ reconstruction MSE
= 0.16223, matching the
dropped eigenvalue λ_2 (the
identity on the next slides)

$n = 200$ points with a dominant correlated direction (Colab notebook, seed fixed). $\mathbf{v}_1$: first principal component; $\mathbf{v}_2$: second, orthogonal, far less spread.

The other view: PCA also minimizes reconstruction error

Pearson (1901): closest-fit line/plane — minimize the sum of squared *perpendicular* distances from each point to the line. **Hotelling (1933)**: independently derived the eigenvector form just taught, the version used today.

these sound like two different problems, one about spread and one about fit, but they are *exactly* the same optimization

Fact. Reconstruct $\mathbf{x}_i - \bar{\mathbf{x}}$ from its top- k projection, $\widehat{\mathbf{x}_i - \bar{\mathbf{x}}} = \sum_{j=1}^k (\mathbf{v}_j^\top (\mathbf{x}_i - \bar{\mathbf{x}})) \mathbf{v}_j$. Then

$$\frac{1}{n} \sum_{i=1}^n \left\| (\mathbf{x}_i - \bar{\mathbf{x}}) - \widehat{\mathbf{x}_i - \bar{\mathbf{x}}} \right\|^2 = \sum_{j=k+1}^d \lambda_j$$

mean squared reconstruction error = exactly the variance not captured by the top k components

Proof: Pythagoras in the eigenbasis; the two framings coincide

Step 1. $\mathbf{v}_1, \dots, \mathbf{v}_d$ is an orthonormal basis of $\mathbb{R}^d$, so every centered point decomposes exactly, and subtracting the rank- k reconstruction leaves only the dropped terms:

$$\mathbf{x}_i - \bar{\mathbf{x}} = \sum_{j=1}^d (\mathbf{v}_j^\top (\mathbf{x}_i - \bar{\mathbf{x}})) \mathbf{v}_j, \quad (\mathbf{x}_i - \bar{\mathbf{x}}) - \widehat{\mathbf{x}_i - \bar{\mathbf{x}}} = \sum_{j=k+1}^d (\mathbf{v}_j^\top (\mathbf{x}_i - \bar{\mathbf{x}})) \mathbf{v}_j.$$

Step 2. Orthonormality kills every cross term (the Pythagorean theorem):

$$\left\| (\mathbf{x}_i - \bar{\mathbf{x}}) - \widehat{\mathbf{x}_i - \bar{\mathbf{x}}} \right\|^2 = \sum_{j=k+1}^d (\mathbf{v}_j^\top (\mathbf{x}_i - \bar{\mathbf{x}}))^2.$$

Step 3. Average over i ; each term is a projected variance (Step 2 of the derivation):

$$\frac{1}{n} \sum_{i=1}^n (\mathbf{v}_j^\top (\mathbf{x}_i - \bar{\mathbf{x}}))^2 = \mathbf{v}_j^\top \Sigma \mathbf{v}_j = \lambda_j. \text{ Summing over } j = k+1, \dots, d \text{ gives the identity.}$$

□

$\text{tr}(\Sigma) = \sum_j \lambda_j$ is fixed, independent of k or the directions chosen: captured variance + dropped variance = constant, so maximizing one is *identical* to minimizing the other. Pearson's "closest fit" and Hotelling's "maximum variance" are one problem; we teach Hotelling's form because it yields an eigendecomposition to compute.

Practical notes: scale matters, and PCA is linear

Scale. Σ 's entries depend on the units features are measured in.

a feature in millimeters has a numerically much larger variance than the identical information in meters — PCA (which only sees Σ) is dominated by whichever feature has larger raw scale, not whichever is more informative

Fix: standardize each feature to unit variance first (equivalently, run PCA on the *correlation* matrix).

Linearity.

PCA only ever finds a **linear** subspace — data along a curved structure is not well captured by any linear projection, however many components are kept

2. K-means

The simplest clustering algorithm

Sections 2–4 adapted from EPFL's ML_course, used with permission.

The K-means objective: NP-hard and non-convex

Section 1 asked whether the data can be described more *compactly*; the rest of the lecture asks whether it has **group structure**. K centroids $\mu_1, \dots, \mu_K \in \mathbb{R}^d$, and a hard assignment $z_{ik} \in \{0, 1\}$ ($\sum_k z_{ik} = 1$) minimizing total squared distance to the assigned centroid:

$$\min_{z, \mu} \sum_{i=1}^n \sum_{k=1}^K z_{ik} \|\mathbf{x}_i - \mu_k\|_2^2 \quad \Longleftrightarrow \quad \min_{\mu_1, \dots, \mu_K} \sum_{i=1}^n \min_k \|\mathbf{x}_i - \mu_k\|_2^2$$

(for fixed μ the best assignment is trivially the closest centroid). This looks like an easy optimization problem. It is not.

Aloise et al. (2009): minimizing this objective is **NP-hard**, even at $d = 2$. Also **non-convex**: the constraint set $z_{ik} \in \{0, 1\}$ alone is non-convex. What's used in practice is a fast local-search heuristic with no guarantee of global optimality.

Lloyd's algorithm, and what it guarantees

Lloyd (1982; building on unpublished 1957 work) gave the algorithm everyone means by “K-means” today. Alternate until nothing changes:

1. **Assignment step.** Fix μ ; assign each point to its nearest centroid:
$$z_i \leftarrow \arg \min_k \|\mathbf{x}_i - \mu_k\|^2.$$
2. **Update step.** Fix z ; move each centroid to the mean of its assigned points:
$$\mu_k \leftarrow \frac{1}{|\{i: z_i = k\}|} \sum_{i: z_i = k} \mathbf{x}_i.$$

Claim. Neither step ever increases the K-means objective, and the algorithm converges after finitely many iterations.

What this does *not* say: which local optimum, or how many iterations.

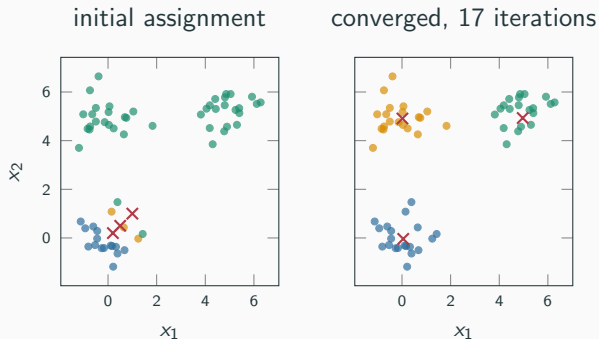
Proof: exact minimizers, finitely many assignments

Assignment step. With μ fixed, the objective $\sum_i \|\mathbf{x}_i - \mu_{z_i}\|^2$ is **separable across** i : each $\mathbf{x}_i$ contributes only $\|\mathbf{x}_i - \mu_{z_i}\|^2$, depending only on z_i . Setting $z_i = \arg \min_k \|\mathbf{x}_i - \mu_k\|^2$ minimizes each term individually, hence the sum.

Update step. With z fixed, the objective is **separable across** k : cluster k contributes $\sum_{i:z_i=k} \|\mathbf{x}_i - \mu_k\|^2$, a function of μ_k alone. Setting the gradient $-2 \sum_{i:z_i=k} (\mathbf{x}_i - \mu_k) = 0$ gives the mean: the update rule is the **exact minimizer**, not a heuristic.

Convergence. The objective is bounded below (by 0) and non-increasing after every step, and there are only finitely many hard assignments (K^n). A non-increasing sequence over a finite set of values must stop changing after finitely many steps. $\square$

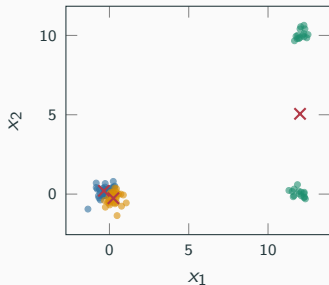
Verified numerically: Lloyd's algorithm converging



3 synthetic blobs, all 3 initial centroids placed deliberately close together near the origin (red $\times$).
Objective: $3973.7 \rightarrow 177.9$. Correctly separates all three — convergence to a local optimum, this time a good one.

Initialization decides which local optimum K-means reaches

bad local optimum (uniform-random init)



The simplest init picks K data points uniformly at random.

a large, loose cluster contains many more points than a small, tight one, so uniform-random selection easily places two initial centroids inside the large cluster and none near the small ones. After that, Lloyd's algorithm cannot recover: every step only lowers the objective *locally*.

Verified: one loose 200-point cluster + two tight, far, well-separated 15-point clusters. In the run shown, the big cluster is split in two (blue/amber) and both small clusters are merged (teal). Over 200 random-init trials, 175 (87.5%) ended more than 5% above a near-optimal objective.

K-means++: seed far apart, with a provable guarantee

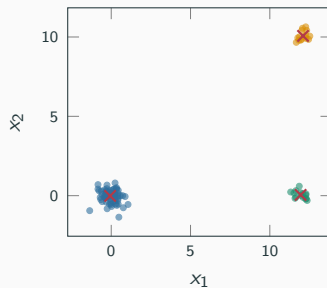
Arthur and Vassilvitskii (2007).

1. Pick μ_1 uniformly at random from the data.
2. For $k = 2, \dots, K$: sample μ_k with probability $\propto D(\mathbf{x})^2$, the squared distance to the *nearest already-chosen* centroid.

Points far from every existing centroid are far more likely to be picked next.

Guarantee: the expected final objective is $O(\log K)$ -competitive with the **optimal** clustering cost. No such guarantee exists for uniform-random initialization.

near-optimal (K-means++ init)



Same data as the previous slide. Bad local optima over 200 trials: K-means++ 17 (8.5%) vs. uniform-random 175 (87.5%).

Failure modes no re-run fixes: shape, outliers, no uncertainty

Beyond initialization, three failures are built into the objective itself.

- **Shape and size.** Minimizing squared distance to a single centroid assumes every cluster is roughly **spherical and similarly sized**; two elongated clusters at an angle are not well captured by “nearest centroid.” Section 4 builds the fix.
- **Sensitivity to outliers.** A single far-away point contributes a squared distance large enough to pull its centroid away from the rest of its cluster — squared error overweighting extreme points (Lecture 05’s regression-for-classification failure, once more).
- **No uncertainty.** A point equidistant between two centroids is still forced into exactly one cluster.

all three trace to one root cause: K-means has no model of what a cluster’s **shape** looks like, beyond “near this one point”

Choosing K : cross-validation does not work here

Adding another cluster can only ever help or do nothing: an extra centroid can always duplicate an existing one.

the objective decreases **monotonically** in K , on training *and* validation data alike — no held-out signal to catch overfitting

Standard fix: the **elbow method**. Plot objective vs. K and keep going only while it still pays off. Same diminishing-returns shape as the scree plot, applied to a different quantity.

3. Expectation-Maximization

The general algorithm underneath
K-means

K-means, abstracted: a latent-variable model fit by alternation

K-means alternated between two kinds of unknowns, a **hidden assignment** z_i per point and **parameters** μ shared by all points. Make that structure probabilistic:

a **latent-variable model** $p(X, Z \mid \theta)$: observed data $X = (\mathbf{x}_1, \dots, \mathbf{x}_n)$, latent variables $Z = (z_1, \dots, z_n)$ never observed, parameters θ . Goal: maximum likelihood,

$$\mathcal{L}(\theta) = \log p(X \mid \theta) = \log \sum_Z p(X, Z \mid \theta).$$

a **log-of-a-sum**: unlike Lecture 02's closed-form MLEs, setting the gradient to zero does not decouple into anything solvable in closed form. Had Z been observed, $\log p(X, Z \mid \theta)$ would usually be an easy “complete-data” MLE. Examples: mixture models (Section 4), hidden Markov models, topic models; nothing below is specific to any one of them.

EM (Dempster, Laird, Rubin 1977): infer Z , refit θ , repeat

Expectation-Maximization (EM): an iterative algorithm that provably **never decreases** the likelihood, alternating between inferring the latent variable and re-fitting the parameters (K-means' two-step skeleton, made probabilistic)

Hard EM.

E-step: $Z \leftarrow \arg \max_Z p(Z \mid X, \theta^{(t)})$ (MAP guess).

M-step: $\theta^{(t+1)} \leftarrow \arg \max_{\theta} \log p(X, Z \mid \theta)$, ordinary complete-data MLE.

Soft EM.

E-step: $q^{(t+1)}(Z) \leftarrow p(Z \mid X, \theta^{(t)})$, the full posterior.

M-step: $\theta^{(t+1)} \leftarrow \arg \max_{\theta} \mathbb{E}_{Z \sim q^{(t+1)}} [\log p(X, Z \mid \theta)]$.

Hard EM commits every point to one latent value even when the data does not support that; soft EM keeps the uncertainty. The next slides prove *why* soft EM works, using nothing model-specific.

The ELBO: a lower bound on $\mathcal{L}$, tight at the posterior

For *any* distribution $q(Z)$ over the latents:

$$\mathcal{L}(\theta) = \underbrace{\sum_Z q(Z) \log \frac{p(X, Z | \theta)}{q(Z)}}_{=: \tilde{\mathcal{L}}(q, \theta), \text{ the ELBO}} + \underbrace{\text{KL}(q(Z) \| p(Z | X, \theta))}_{\geq 0}$$

“Evidence Lower BOund”: always safe to maximize, since raising it can never overshoot $\mathcal{L}(\theta)$.

$\text{KL}(q \| p(Z | X, \theta)) \geq 0$ always (Gibbs' inequality), with equality iff $q(Z) = p(Z | X, \theta)$. So the ELBO is a lower bound on $\mathcal{L}(\theta)$ for *every* q , and **tight** exactly when q equals the true posterior over the latents.

Proof of the ELBO decomposition

Step 1. Combine the two terms:

$$\begin{aligned}\tilde{\mathcal{L}}(q, \theta) + \text{KL}(q \parallel p(Z \mid X, \theta)) &= \sum_Z q(Z) \log \frac{p(X, Z \mid \theta)}{q(Z)} + \sum_Z q(Z) \log \frac{q(Z)}{p(Z \mid X, \theta)} \\ &= \sum_Z q(Z) \log \frac{p(X, Z \mid \theta)}{p(Z \mid X, \theta)}\end{aligned}$$

Step 2. By the definition of conditional probability (Bayes' rule rearranged), for every value of Z :

$$\frac{p(X, Z \mid \theta)}{p(Z \mid X, \theta)} = p(X \mid \theta), \quad \text{independent of } Z.$$

Step 3. The ratio inside the sum is the constant $p(X \mid \theta)$, and $\sum_Z q(Z) = 1$:

$$\sum_Z q(Z) \log p(X \mid \theta) = \log p(X \mid \theta) \sum_Z q(Z) = \log p(X \mid \theta) = \mathcal{L}(\theta). \quad \square$$

EM is coordinate ascent on the ELBO

E-step. Fix $\theta = \theta^{(t)}$; maximize $\tilde{\mathcal{L}}(q, \theta^{(t)})$ over q . The bound is tight exactly at the posterior, so

$$q^{(t+1)}(Z) := p(Z \mid X, \theta^{(t)}), \quad \tilde{\mathcal{L}}(q^{(t+1)}, \theta^{(t)}) = \mathcal{L}(\theta^{(t)}).$$

M-step. Fix $q = q^{(t+1)}$; maximize $\tilde{\mathcal{L}}(q^{(t+1)}, \theta)$ over θ . Splitting the ELBO, $\tilde{\mathcal{L}}(q, \theta) = \mathbb{E}_q[\log p(X, Z \mid \theta)] + H(q)$, and $H(q)$ does not depend on θ , so this is exactly maximizing the **expected complete-data log-likelihood**:

$$\theta^{(t+1)} = \arg \max_{\theta} \mathbb{E}_{Z \sim q^{(t+1)}} [\log p(X, Z \mid \theta)]$$

These are precisely soft EM's two steps: the algorithm is coordinate ascent on $\tilde{\mathcal{L}}(q, \theta)$, alternating the q coordinate and the θ coordinate.

Monotonic ascent: the likelihood never decreases

$$\mathcal{L}(\theta^{(t+1)}) \geq \tilde{\mathcal{L}}(q^{(t+1)}, \theta^{(t+1)}) \geq \tilde{\mathcal{L}}(q^{(t+1)}, \theta^{(t)}) = \mathcal{L}(\theta^{(t)})$$

- **First inequality:** the ELBO lower-bounds $\mathcal{L}$ for every q .
- **Second inequality:** the M-step chose $\theta^{(t+1)}$ to maximize $\tilde{\mathcal{L}}(q^{(t+1)}, \cdot)$.
- **Final equality:** the E-step made the bound tight at $\theta^{(t)}$.

every EM iteration satisfies $\mathcal{L}(\theta^{(t+1)}) \geq \mathcal{L}(\theta^{(t)})$, for *any* latent-variable model — the same bounded-and-monotone structure that made Lloyd's algorithm converge, now for a likelihood

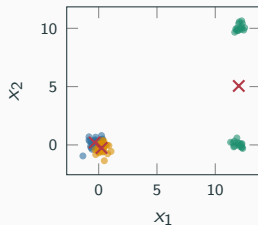
Non-convexity: why initialization decided K-means' optimum

Monotone ascent on a **non-convex** $\mathcal{L}(\theta)$ only reaches a **stationary point**, not the global maximum; which one depends on the starting point. (For a mixture, permuting the components gives $K!$ equivalent maxima, so $\mathcal{L}$ cannot be convex.)

This is Section 2's initialization failure, explained. K-means is EM's hard-assignment special case (Section 4), so Lloyd's algorithm is coordinate descent on a non-convex objective. The uniform-random init on the right started in the basin of a bad stationary point; every later step only improved the objective *locally*.

Same fix for both: random restarts (keep the best) and careful seeding (K-means++; K-means itself as EM's initializer, Section 4).

Section 2's bad local optimum



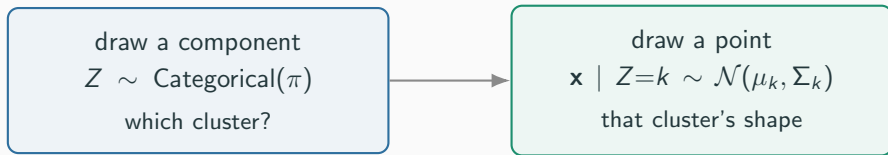
A stationary point of the K-means objective: no assignment or update step lowers it, yet it is far from optimal.

4. EM for Gaussian Mixture Models

Giving clusters an explicit shape

The Gaussian Mixture Model: a generative story with a latent Z

K-means' failures (Section 2) came from having no model of a cluster's shape. A **Gaussian Mixture Model (GMM)** replaces “near this one point” with an explicit generative story:



Each component gets its own mean *and* its own, possibly non-spherical, covariance. Parameters $\theta = (\pi, \mu, \Sigma)$; Z is **never observed**, a **latent variable** in exactly Section 3's sense. Marginalizing it out gives the density a GMM assigns to a point, and the log-likelihood EM must maximize:

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}; \mu_k, \Sigma_k), \quad \mathcal{L}(\theta) = \sum_{i=1}^n \log \left(\sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_i; \mu_k, \Sigma_k) \right).$$

E-step for a GMM: responsibilities r_{ik} are the posterior

Section 3's E-step sets $q(Z)$ to the posterior $p(Z | X, \theta^{(t)})$. For a GMM it factorizes over points; by Bayes' rule, the posterior probability that point i came from component k is

$$r_{ik} := \mathbb{P}(z_i = k | \mathbf{x}_i, \theta^{(t)}) = \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_i; \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_{j=1}^K \pi_j^{(t)} \mathcal{N}(\mathbf{x}_i; \mu_j^{(t)}, \Sigma_j^{(t)})}$$

$r_{ik} \in [0, 1]$: the **responsibility** component k takes for point i . $\sum_k r_{ik} = 1$ for every i .

A point roughly equidistant between two components' means gets $r_{ik} \approx \frac{1}{2}$ for each, instead of being forced into one; this is exactly the uncertainty K-means threw away.

M-step for a GMM: maximize $Q(\theta)$; the π_k update derived

$Z = (z_1, \dots, z_n)$, each z_i one-hot ($z_{ik} \in \{0, 1\}$), so

$\log p(X, Z \mid \theta) = \sum_i \sum_k z_{ik} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_i; \mu_k, \Sigma_k)]$. Taking the expectation under $q^{(t+1)}$ replaces z_{ik} with its expectation, the responsibility r_{ik} :

$$Q(\theta) := \mathbb{E}_{q^{(t+1)}} [\log p(X, Z \mid \theta)] = \sum_{i=1}^n \sum_{k=1}^K r_{ik} [\log \pi_k + \log \mathcal{N}(\mathbf{x}_i; \mu_k, \Sigma_k)]$$

The π_k update. Only the $\log \pi_k$ terms depend on π ; with $N_k := \sum_i r_{ik}$ (the effective count assigned to k) and the constraint $\sum_k \pi_k = 1$, the Lagrangian is $\sum_k N_k \log \pi_k - \nu (\sum_k \pi_k - 1)$.

$$\partial / \partial \pi_k = 0 \Rightarrow N_k / \pi_k = \nu \Rightarrow \pi_k \propto N_k.$$

$$\text{The constraint fixes } \nu = n: \quad \pi_k = N_k / n$$

The μ_k and Σ_k updates: weighted Gaussian MLE

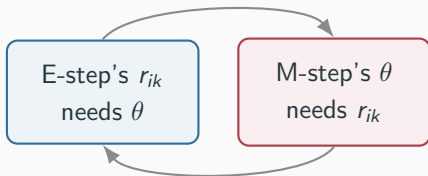
Only the Gaussian term in $Q(\theta)$ depends on μ_k . Setting its gradient to zero:

$$\sum_i r_{ik} \Sigma_k^{-1} (\mathbf{x}_i - \mu_k) = 0 \implies \mu_k = \frac{1}{N_k} \sum_i r_{ik} \mathbf{x}_i,$$

and Σ_k 's update is the analogous weighted second-moment MLE. The full M-step:

$$\pi_k = \frac{N_k}{n}, \quad \mu_k = \frac{1}{N_k} \sum_i r_{ik} \mathbf{x}_i, \quad \Sigma_k = \frac{1}{N_k} \sum_i r_{ik} (\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^\top$$

ordinary Gaussian MLE, each point weighted by r_{ik} , throughout



A chicken-and-egg structure, resolved as with K-means by **iterating** rather than solving simultaneously. Section 3's theorem guarantees $\mathcal{L}(\theta)$ never decreases along the way.

Hard EM on a restricted GMM is exactly K-means

Model side. Fix every covariance to the same spherical $\Sigma_k = \sigma^2 I$ and the mixing weights uniform, $\pi_k = 1/K$. Then

$$\log(\pi_k \mathcal{N}(\mathbf{x}; \mu_k, \sigma^2 I)) = \underbrace{\text{same for every } k} - \frac{1}{2\sigma^2} \|\mathbf{x} - \mu_k\|^2$$

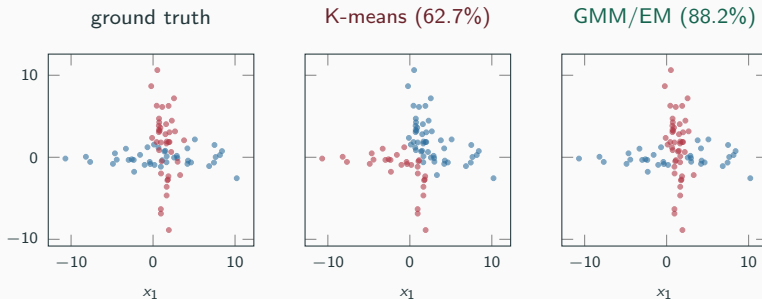
$$\text{so } \arg \max_k \pi_k \mathcal{N}(\mathbf{x}; \mu_k, \sigma^2 I) = \arg \min_k \|\mathbf{x} - \mu_k\|^2.$$

Assigning each point to its **most likely** (MAP) component under this restricted GMM is K-means' nearest-centroid rule.

Algorithm side. Hard EM for a GMM: E-step $z_i \leftarrow \arg \max_k \pi_k \mathcal{N}(\mathbf{x}_i; \mu_k, \Sigma_k)$; M-step = fully-observed MLE (π_k the fraction labeled k , μ_k the mean of points labeled k , Σ_k their sample covariance). Hold π and Σ fixed: the E-step is nearest-centroid and the M-step is K-means' mean update.

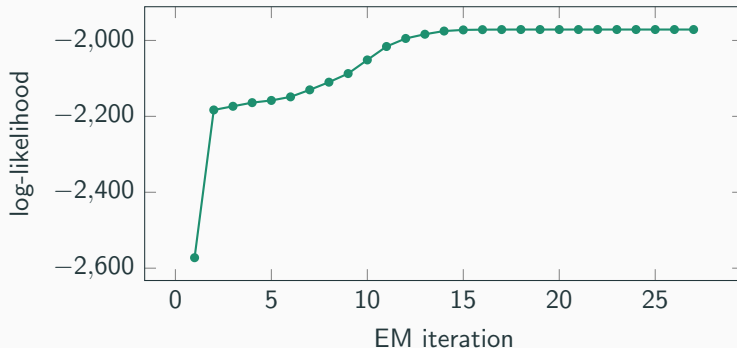
K-means is hard EM on a GMM with π uniform and Σ fixed spherical, held fixed instead of estimated. Soft EM on the full GMM generalizes it in two independent ways: soft assignments, and learned shapes.

Verified numerically: GMM/EM vs. K-means on crossed clusters



Two elongated, perpendicular clusters, crossing near the origin. K-means can only cut with straight bisecting lines (splits the cross left/right); GMM's per-component covariance recovers the true horizontal/vertical split. Accuracy vs. ground truth: K-means 62.7%, GMM/EM 88.2%. K-means objective = 7353.8; GMM/EM log-likelihood = -1971.3 after 27 iterations (best of 20 K-means++-seeded restarts each).

Verified numerically: the log-likelihood never decreases



Same crossed-cluster fit: $-2572.4 \rightarrow -1971.3$ over 27 iterations, checked non-decreasing at every step
— Section 3's monotonic-ascent theorem, observed rather than just asserted. The plateau after iteration 17 is a stationary point, the only kind of point EM promises.

Practical notes: choosing K , anomaly detection, initialization

- **Choosing K : cross-validation works here.** Training log-likelihood $\rightarrow \infty$ as a component's $\Sigma_k \rightarrow 0$ around one training point, genuine overfitting-of-likelihood. A held-out point is (almost surely) not exactly at that training point, so *test* likelihood degrades once K is too large, the same way test error does for any overfit model. Unlike K-means' monotonic-in- K objective, ordinary cross-validation applies.
- **Anomaly detection.** Once fit, $p(\mathbf{x}) = \sum_k \pi_k \mathcal{N}(\mathbf{x}; \mu_k, \Sigma_k)$ is an explicit density; thresholding it flags anomalous points, something K-means' hard partition never provides.
- **K-means as GMM initialization.** EM, like K-means, only reaches a stationary point (Section 3). Common recipe: run K-means first and use its centroids to initialize the GMM's means μ_k before running full soft EM, cheaper than random restarts of EM alone.

5. Wrap-up

One chain, tied together

The chain, tied together

Method	Assignment	Cluster shape
K-means	Hard (nearest centroid)	Fixed spherical, equal size
Hard EM (GMM)	Hard (MAP)	Estimated covariance
Soft EM (GMM)	Soft (r_{ik})	Estimated covariance

Each row generalizes the one above it; the same iterate-until-convergence skeleton, and the same non-convexity, throughout.

- **PCA**: variance-maximization $\equiv$ reconstruction-error-minimization, proved: the same optimization problem.
- **K-means**: coordinate descent on a non-convex objective, so initialization decides the optimum; K-means++ seeding gives an $O(\log K)$ guarantee.
- **EM**: coordinate ascent on the ELBO, provably monotonic, for any latent-variable model; K-means is its hard-assignment special case on a restricted GMM.

Exam 1 (Lectures 01–06), in two weeks
— classical supervised and unsupervised learning, no neural networks

Lecture 07 starts the deep-learning half of the course: neural network basics, backpropagation, and why optimizing deep networks is genuinely harder than anything built so far.

References

- Pearson. *On Lines and Planes of Closest Fit to Systems of Points in Space*. Philosophical Magazine, 2(11), 1901.
- Hotelling. *Analysis of a Complex of Statistical Variables into Principal Components*. Journal of Educational Psychology, 24, 1933.
- Lloyd. *Least Squares Quantization in PCM*. IEEE Trans. Inf. Theory, 28(2), 1982.
- MacQueen. *Some Methods for Classification and Analysis of Multivariate Observations*. Berkeley Symp., 1967.
- Arthur, Vassilvitskii. *k-means++: The Advantages of Careful Seeding*. SODA, 2007.
- Aloise, Deshpande, Hansen, Popat. *NP-hardness of Euclidean Sum-of-Squares Clustering*. Machine Learning, 75, 2009.
- Dempster, Laird, Rubin. *Maximum Likelihood from Incomplete Data via the EM Algorithm*. JRSS B, 39(1), 1977.

PCA content is original, written from Pearson and Hotelling (EPFL's course covers matrix factorization in an SVD/Netflix-Prize framing, not this derivation). Clustering/GMM/EM content adapted from EPFL's ML_course, used with permission. Full citation details:

`references/sources/lecture06-{pca,clustering-em}-citations.md`.