EECS 553 Lecture 20
Machine Learning (ECE)
[class version]

Jeff Fessler

University of Michigan
©Laura Balzano and Jeff Fessler, UM, November 10, 2026, 13:11

F26

Logistics

© 2026 Fessler

- HW 7 due Wed. Nov. 11
- Exam 2: Mon. Nov. 16, 6-8PM; 1013 Dow; 2 sheets
  - W26 practice exam solutions on solution server
  - Two Claude-generated practice exams on Canvas
  - Student MC practice problems curated by Claude on Canvas
  - Exam "review" slides on gdrive and my web site
  - Prof. Scott's excellent course notes on Canvas also have problems at end of each topic.

Recap / Outline

© 2026 Fessler

- Recap
  - Embedding, dimensionality reduction, t-SNE
- Today
  - Clustering
    - Lloyd's algorithm
  - Gaussian mixture model (GMM)
  - Latent variable models
  - Expectation Maximization (EM)

Lloyd's algorithm for k-means with Euclidean distance

© 2026

Fessler

Input: $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, k

Initialize $\boldsymbol{m}_1, \ldots . \boldsymbol{m}_k \in \mathbb{R}^d$

Repeat

  For $i = 1, \ldots, n$

    $C(i) = \arg\min_{\ell} \|\boldsymbol{x}_i - \boldsymbol{m}_\ell\|$

  End

  For $\ell = 1, \ldots, k$

    $\boldsymbol{m}_\ell = \frac{1}{|\{i : C(i) = \ell|} \sum_{i:C(i)=\ell} \boldsymbol{x}_i$

  End

Until the clusters don't change.

Output: C

The cost function for the k-**means method** for (unsupervised) clustering is the **within cluster sum of squares**:

$$\arg\min_{\mathcal{C},\,\mathcal{M}} \mathrm{WCSS}(\mathcal{C},\mathcal{M}), \quad \mathrm{WCSS}(\mathcal{C},\mathcal{M}) \triangleq \sum_{j=1}^{k} \sum_{i\,:\,\boldsymbol{x}_i \in \mathcal{C}_j} \|\boldsymbol{x}_i - \boldsymbol{m}_j\|_2^2 .$$

- WCSS "monotonically decreases" (non-increasing)
- Wikipedia says "k-means always converges" - too imprecise!
- Must break ties in a consistent and deterministic way.
- Must have non-empty clusters or freeze centroids of empty clusters.
- With appropriate handling of those edge cases, the $(\mathcal{C}, \mathcal{M})$ iterates stabilize in finitely many iterations.
- At convergence, if there are no ties and all clusters are nonempty, then the fixed point is a **strict local minimum**.
- **k-means++** has smart initialization and some performance guarantee.

Illustration: Old Faithful exploration

© 2026 Fessler

- **Old Faithful** is a geyser in **Yellowstone National Park** that erupts spectacularly for several minutes, every 35-120 minutes.
- It is only quasi-periodic.



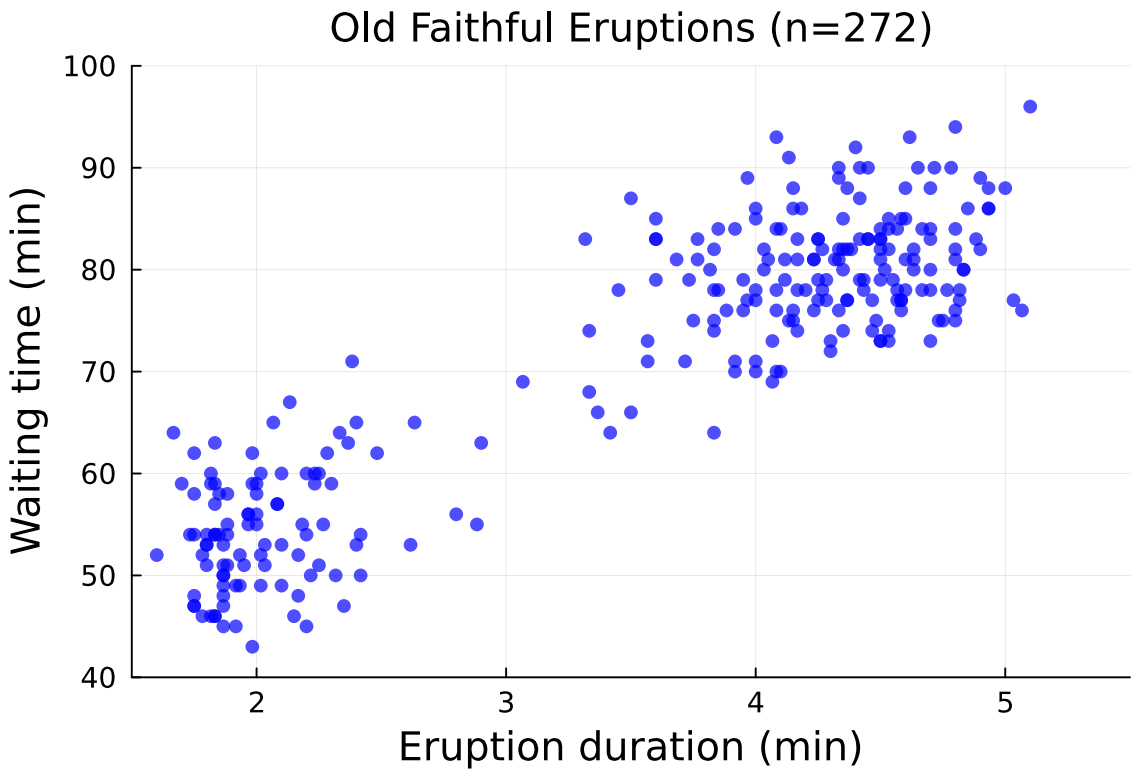
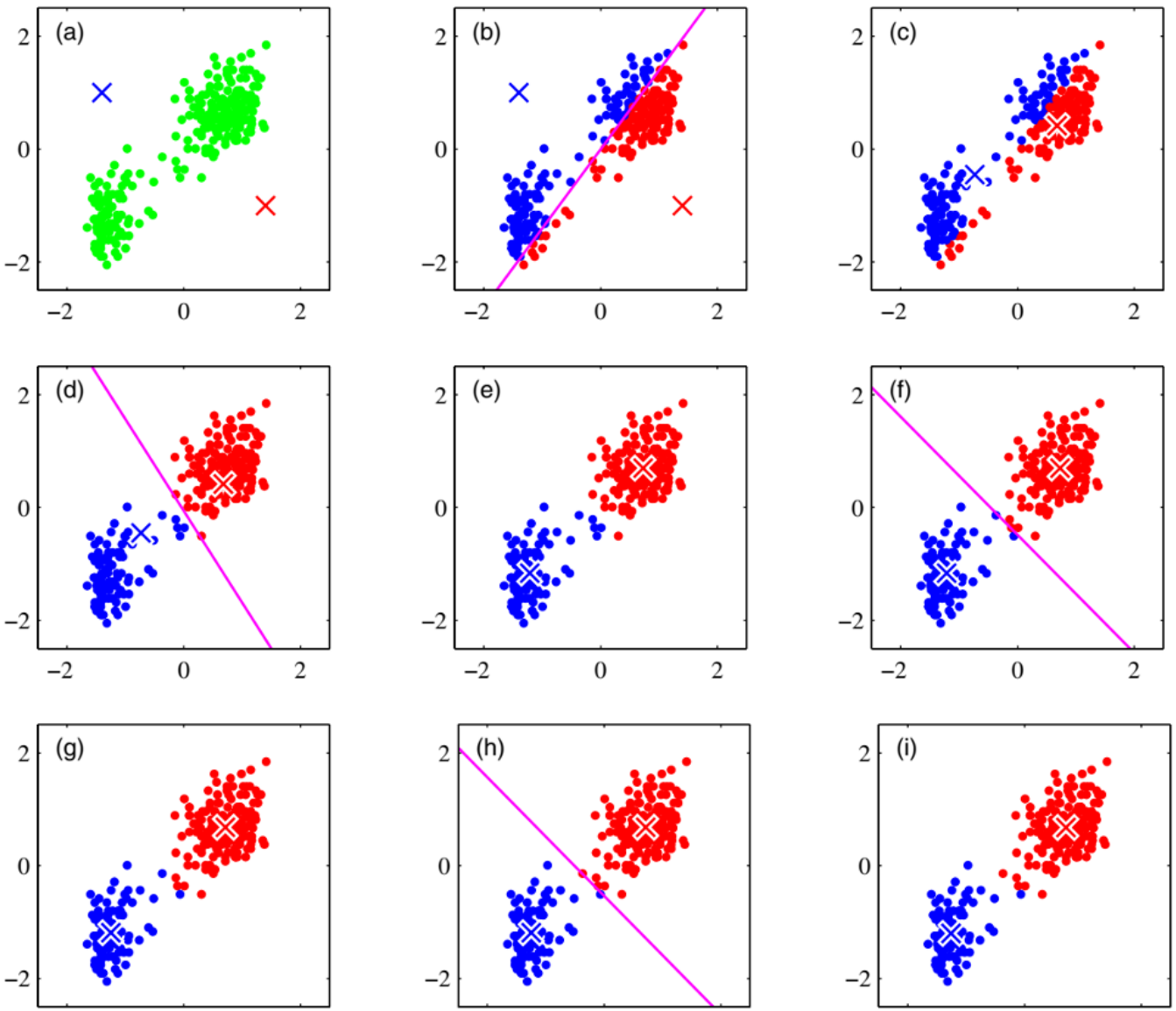


Illustration: Old Faithful results

© 2026 Fessler



Example from [7].

CQ: clustering

© 2026 Fessler

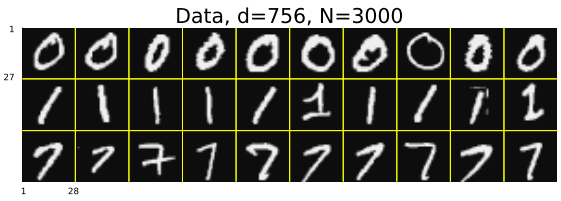
Recall clustering goal:
"Want intra-class similarities higher than inter-class similarities."

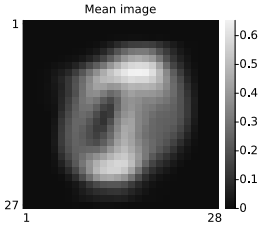
Q20.1 Did k-means method applied to Old Faithful data accomplish this goal?

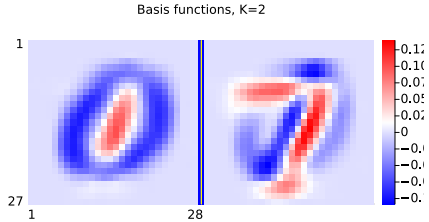
A: Yes B: Mostly yes C: Maybe not D: No

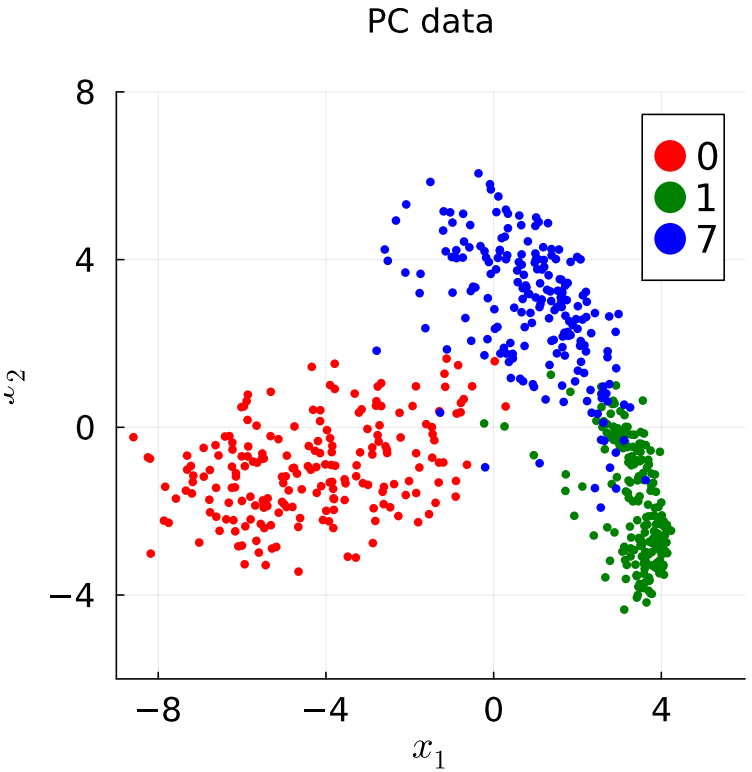
Even though k-means is a very classic clustering method, it has been used in modern ML for learnable sparse attenation [1, p. 531].

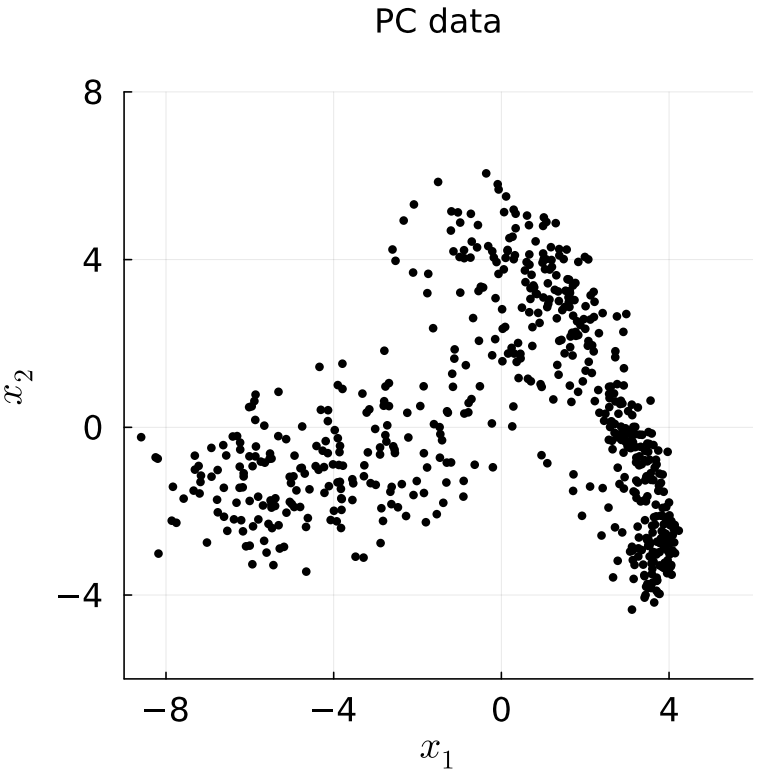
Demo







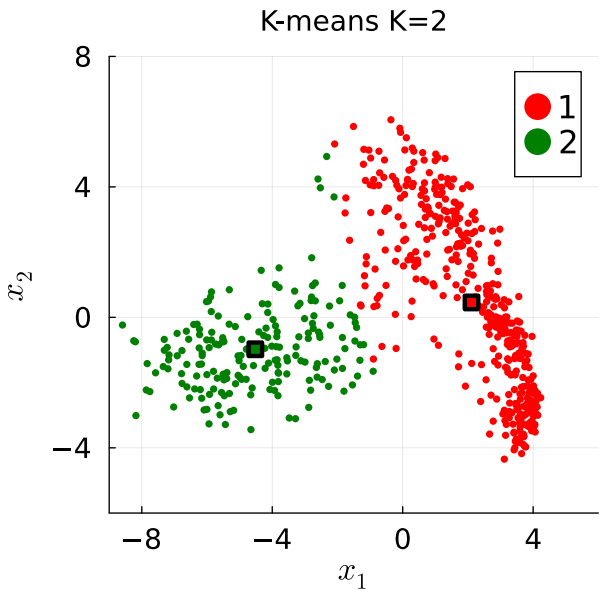


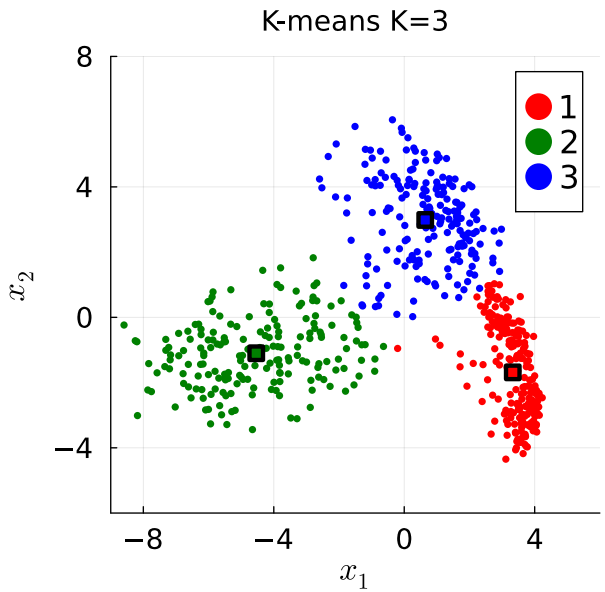


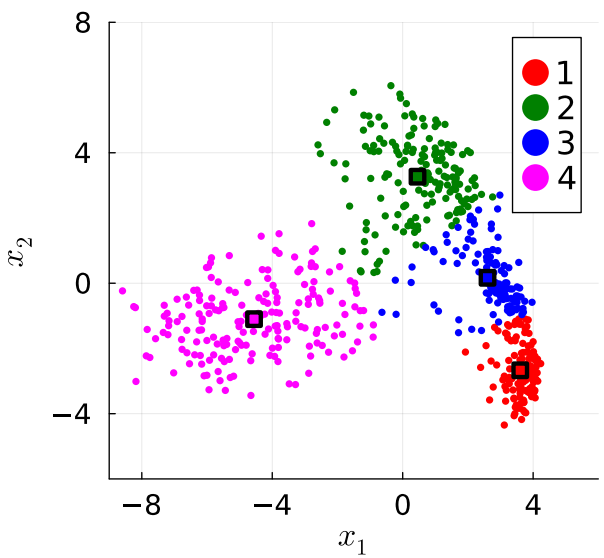
In real-world clustering applications, the data is unlabeled.

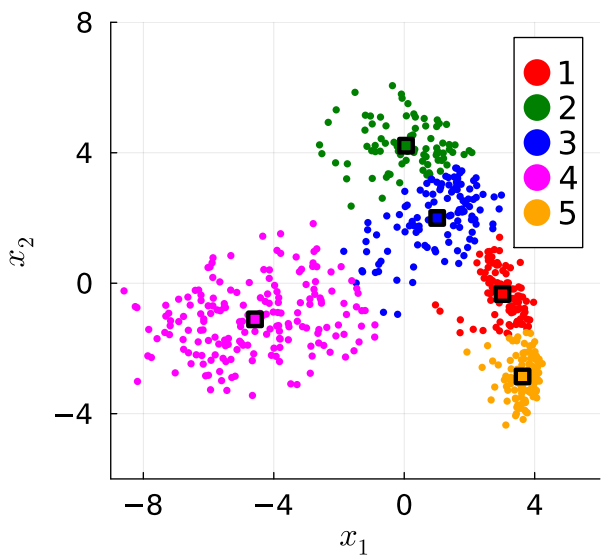
k-means: Demo results

© 2026 Fessler

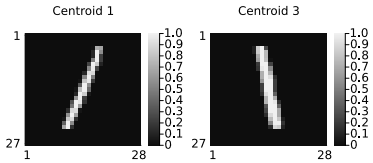








Many methods for choosing k, the **number of clusters**.



k-means is a centroid-based algorithm.

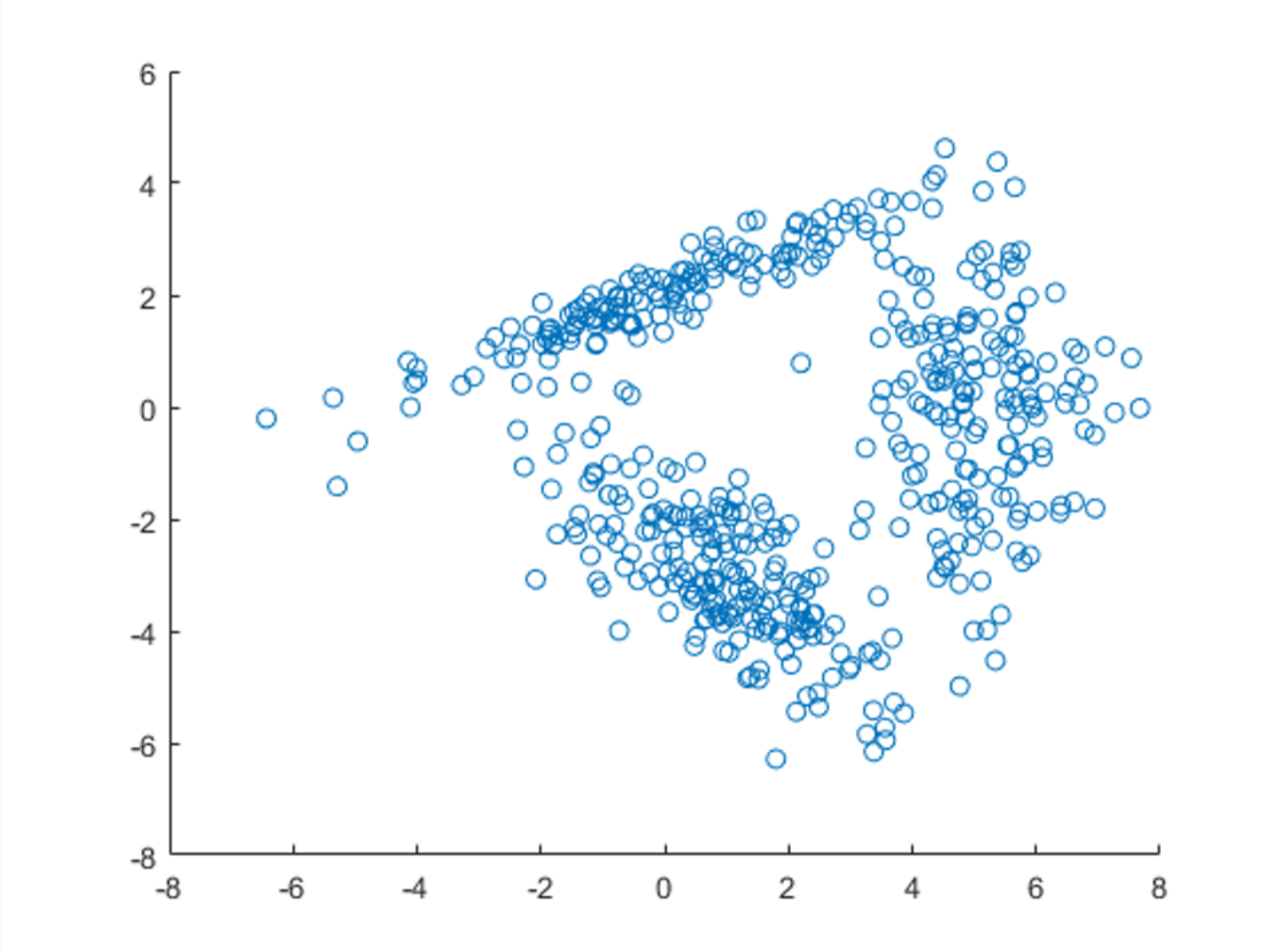


Figure courtesy Clay Scott.

Another approach is to assume a distributional (generative) model.

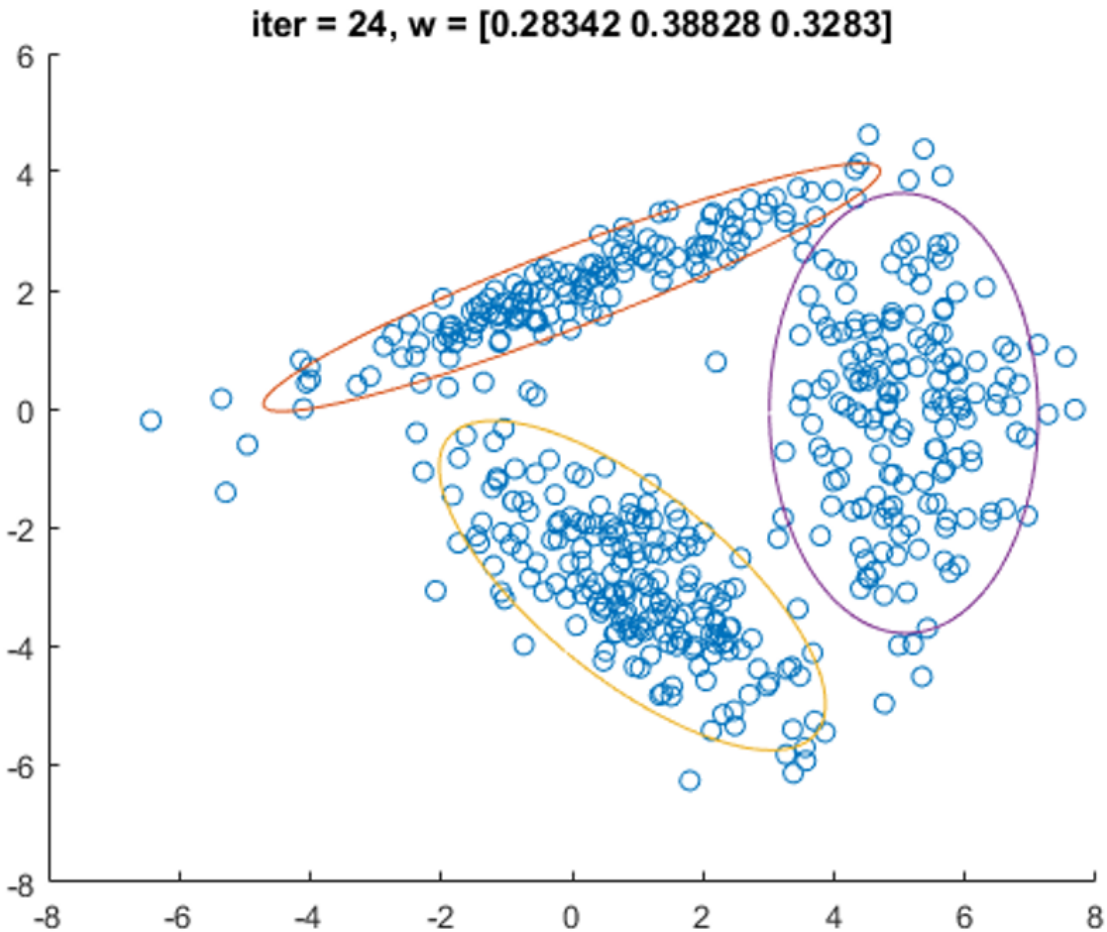


Figure courtesy Clay Scott.

Gaussian mixture model (GMM): Formulation

© 2026 Fessler

- Recall the **multivariate Gaussian density** for $\boldsymbol{x}, \boldsymbol{\mu} \in \mathbb{R}^d$ and covariance matrix $\boldsymbol{\Sigma} \succ \mathbf{0} \in \mathbb{R}^{d \times d}$:

$$\phi(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = |2\pi\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right).$$

- A random variable $\boldsymbol{X}$ follows a **Gaussian mixture model** (**GMM**) iff its probability density function has the form

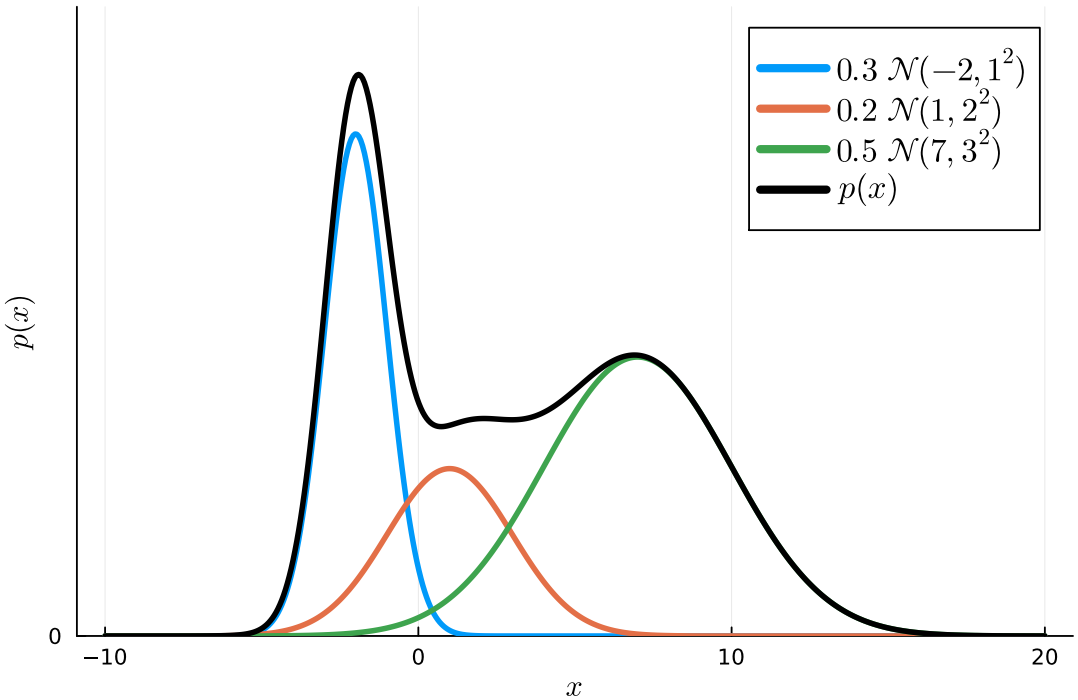
$$\mathrm{p}(\boldsymbol{x}) =$$

where

and for all k, $\boldsymbol{\mu}_k \in \mathbb{R}^d$ and $\boldsymbol{\Sigma}_k \in \mathbb{R}^{d \times d}, \boldsymbol{\Sigma}_k \succ \mathbf{0}$.

© 2026 Fessler

Example:
$d = 1,\ K = 3$



Q20.2 A mixture of Gaussian densities is (always, never) a Gaussian density.

A: (T,T) B: (T,F) C: (F,T) D: (F,F) **??**

GMM generative model

© 2026 Fessler

- A key to understanding GMMs is to know how to simulate (generate) a realization from a known GMM.
- Suppose we know all GMM parameters:

$$\boldsymbol{\theta} = (w_1, \ldots, w_K, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_K).$$

- The following two-step procedure generates a sample (realization) drawn from the GMM with parameter vector $\boldsymbol{\theta}$.
  - Select $k \in \{1, \ldots, K\}$ at random, according to the **pmf** $w_1, \ldots, w_K$.
  - Draw a realization of $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$
- Example. The figure 3 slides ago was surely generated this way.

- Next we go in reverse: given (measured) data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, estimate model parameters $\boldsymbol{\theta}$ *assuming* a GMM.

GMM likelihood

© 2026 Fessler

- Observed data: $\mathcal{D} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$
- Assume IID from GMM with some
- For that assumption, the **likelihood** is

$$L(\boldsymbol{\theta}; \mathcal{D}) \triangleq \prod_{i=1}^{n} \mathrm{p}(\boldsymbol{x}_i; \boldsymbol{\theta}) =$$

and the **log-likelihood** is

$$\ell(\boldsymbol{\theta}; \mathcal{D}) \triangleq \sum_{i=1}^{n} \log$$

- When $K > 1$, there is no closed-form, maximum-likelihood estimate of $\boldsymbol{\theta}$.
- Numerical optimization is challenging because $-\ell(\boldsymbol{\theta}; \mathcal{D})$ is a function of $\boldsymbol{\theta}$.
- The constraint $\boldsymbol{\Sigma} \succ \mathbf{0}$ further complicates optimization.

- GMM is a special case of a **latent variable model**
- **EM algorithms** are well-suited to parameter estimation for such models

Latent variable models: Overview

© 2026 Fessler

- In a **latent variable model**, each observation $\boldsymbol{x}_i \in \mathbb{R}^d$ is explained (in part) by
- Examples:
  - Gaussian mixture model where
  - Any other mixture model
  - Missing features
  - Censored/truncated observations
  - Factor analysis
  - Probabilistic PCA
  - Hidden Markov models
  - Structural equation models
  - Variational Autoencoders
  - Diffusion models
  - Causal effect models
  - Combinations of the above

In all such data models, it is challenging to directly estimate

Latent variable models: Strategy

© 2026 Fessler

- Assume every observation $\boldsymbol{X}_i$ has an associated (hand-crafted!) **latent variable**
- Let $= (\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_n)$ denote all the **latent** variables.
- Let $= (\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n)$ denote all the *observed* variables.
- Assume $\underline{\boldsymbol{X}}$ and $\underline{\boldsymbol{Z}}$ are jointly distributed random variables.
- Generally, the latent variable is designed such that optimization of the model parameters $\boldsymbol{\theta}$ is easy when $\underline{\boldsymbol{Z}}$ is known.
- This property suggests an alternating strategy for model fitting:
  Fix
  Fix

- There are general strategies to perform maximum likelihood estimation in such models.
- We will now study the most well-known one: EM.

"The" **EM algorithm** is motivated by the fact that for many problems, calculation of the likelihood is greatly simplified if we had certain additional observations [8, p. 450].

- Gaussian mixture: if we observed $Z_i \in \{1, \dots, K\}_{i=1}^n$ that told us the mixture component for each data point, then we could easily compute ML estimates of means and variances of each mixture component.
- If we had observed the missing/censored/truncated values, we could perform standard full-feature processing like SVD/eigenvalues, *e.g.*, for PCA likelihood.
- If we knew underlying factors from a **factor model**, for any data point we could approximate it with the max likelihood mix of those factors using projection.

Expectation maximization: Formulation

© 2026 Fessler

- Let $\underline{\boldsymbol{Z}} = (\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_n)$ denote the latent (unobserved) variables.
- Assume $\underline{\boldsymbol{X}}$ and $\underline{\boldsymbol{Z}}$ are jointly distributed random variables.
- Let $\mathrm{p}(\underline{\boldsymbol{z}} \mid \underline{\boldsymbol{x}}; \boldsymbol{\theta})$ denote the conditional pdf/pmf of $\underline{\boldsymbol{Z}}$ given $\underline{\boldsymbol{X}} = \underline{\boldsymbol{x}}$.
- The **complete-data likelihood** is the (hypothetical!) likelihood with both the observed and unobserved variables:

$$\tilde{L}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) \triangleq \mathrm{p}(\underline{\boldsymbol{z}}, \underline{\boldsymbol{x}}; \boldsymbol{\theta}) =$$

- The **complete-data log-likelihood** is

$$\tilde{\ell}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) = \log \tilde{L}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) =$$

$$= \qquad (1)$$

Expectation maximization (EM): Strategy

© 2026 Fessler

- The basic idea behind the EM algorithm [9, 10] is to replace

$$\ell(\boldsymbol{\theta}; \underline{\boldsymbol{x}}),$$

  which is often difficult to maximize, with

- The latter is easier to compute for many problems of interest.
- Because $\boldsymbol{\theta}$ is unknown, we use estimate

- That approach seems natural, but what about **convergence**?
  - Does the log-likelihood sequence $\{\ell(\boldsymbol{\theta}_t; \underline{\boldsymbol{x}})\}$ converge?
  - Does the iterate sequence $\{\boldsymbol{\theta}_t\}$ converge?

CQ: Expectation maximization (EM)

© 2026 Fessler

- The EM algorithm starts at some initial guess $\boldsymbol{\theta}_0$ and then tries to constantly improve it by increasing the likelihood.
- From (1), the relationship between complete-data log-likelihood and observed-data log-likelihood is:

$$\tilde{\ell}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) = \ell(\boldsymbol{\theta}; \underline{\boldsymbol{x}}) + \log \mathsf{p}(\underline{\boldsymbol{z}} \mid \underline{\boldsymbol{x}}; \boldsymbol{\theta}).$$

Q20.3 Suppose our current guess is $\boldsymbol{\theta}_t$. Which of the following holds for the log-likelihood difference $\ell(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta}_t)$?

A: $\ell(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta}_t) = \tilde{\ell}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) - \tilde{\ell}(\boldsymbol{\theta}_t; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) + \log \frac{\mathsf{p}(\underline{\boldsymbol{z}}|\underline{\boldsymbol{x}};\boldsymbol{\theta}_t)}{\mathsf{p}(\underline{\boldsymbol{z}}|\underline{\boldsymbol{x}};\boldsymbol{\theta})}$

B: $\ell(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta}_t) = \tilde{\ell}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) - \tilde{\ell}(\boldsymbol{\theta}_t; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) - \log \frac{\mathsf{p}(\underline{\boldsymbol{z}},\underline{\boldsymbol{x}};\boldsymbol{\theta})}{\mathsf{p}(\underline{\boldsymbol{z}},\underline{\boldsymbol{x}};\boldsymbol{\theta}_t)}$

C: $\ell(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) - \ell(\boldsymbol{\theta}_t; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) = \tilde{\ell}(\boldsymbol{\theta}_t) - \tilde{\ell}(\boldsymbol{\theta}) + \log \frac{\mathsf{p}(\underline{\boldsymbol{z}}|\underline{\boldsymbol{x}};\boldsymbol{\theta})}{\mathsf{p}(\underline{\boldsymbol{z}}|\underline{\boldsymbol{x}};\boldsymbol{\theta}_t)}$

D: all of these

E: none of these

Expectation maximization (EM): Key inequality © 2026 Fessler

We have from (1):

$$\ell(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta}_t) = \tilde{\ell}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) - \tilde{\ell}(\boldsymbol{\theta}_t; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) +$$

- Taking expectation of both sides with respect to $\underline{\boldsymbol{Z}} \sim \mathsf{p}(\underline{\boldsymbol{z}} \mid \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t)$, we have

$$\ell(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta}_t) = \mathbb{E}[\tilde{\ell}(\boldsymbol{\theta})] - \mathbb{E}[\tilde{\ell}(\boldsymbol{\theta}_t)] +$$

- Because $D(p_0 \| p_1) \geq 0$, we have a key **minorization** inequality:

$$\ell(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta}_t) \geq \qquad (2)$$

- EM is a special case of **minorize-maximize** aka **majorize-minimize** (**MM**) optimization [12, 13].

Defining

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t) \triangleq$$

the following key inequality follows from (2):

$$\ell(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta}_t) \geq \tag{3}$$

EM chooses $\boldsymbol{\theta}_{t+1}$ to be a solution of the optimization problem

$$\boldsymbol{\theta}_{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t) \implies$$

Thus from (3):

$$\ell(\boldsymbol{\theta}_{t+1}) - \ell(\boldsymbol{\theta}_t) \geq Q(\boldsymbol{\theta}_{t+1}, \boldsymbol{\theta}_t) - Q(\boldsymbol{\theta}_t, \boldsymbol{\theta}_t) \geq 0$$

$$\implies \ell(\boldsymbol{\theta}_0) \leq \cdots \ell(\boldsymbol{\theta}_t) \leq \ell(\boldsymbol{\theta}_{t+1}) \leq \ell_{\max} =$$

- Because $\{\ell(\boldsymbol{\theta}_t; \underline{\boldsymbol{x}})\}$ is nondecreasing and bounded above, the *log-likelihood* sequence $\{\ell(\boldsymbol{\theta}_t; \underline{\boldsymbol{x}})\}$ converges monotonically!
- Convergence properties of the **iterate** sequence $\{\boldsymbol{\theta}_t\}$ generated by EM and variations thereof are studied in [14] [15].

Expectation maximization (EM): Algorithm

© 2026 Fessler

Initialize $\boldsymbol{\theta}_0$

$t \leftarrow 0$

Repeat

E-step: "Compute" the expectation:

$$Q(\boldsymbol{\theta}; \boldsymbol{\theta}_t) = \mathbb{E}_{\underline{\boldsymbol{Z}} \sim \mathsf{p}(\underline{\boldsymbol{z}} | \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t)} \left\{ \tilde{\ell}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) \right\}$$

M-step: Perform the maximization:

$$\boldsymbol{\theta}_{t+1} \leftarrow$$

$t \leftarrow t + 1$

Until convergence criterion satisfied.

Expectation maximization (EM): Monotonicity © 2026 Fessler

Graphical depiction of the likelihood and Q function and an EM algorithm update:

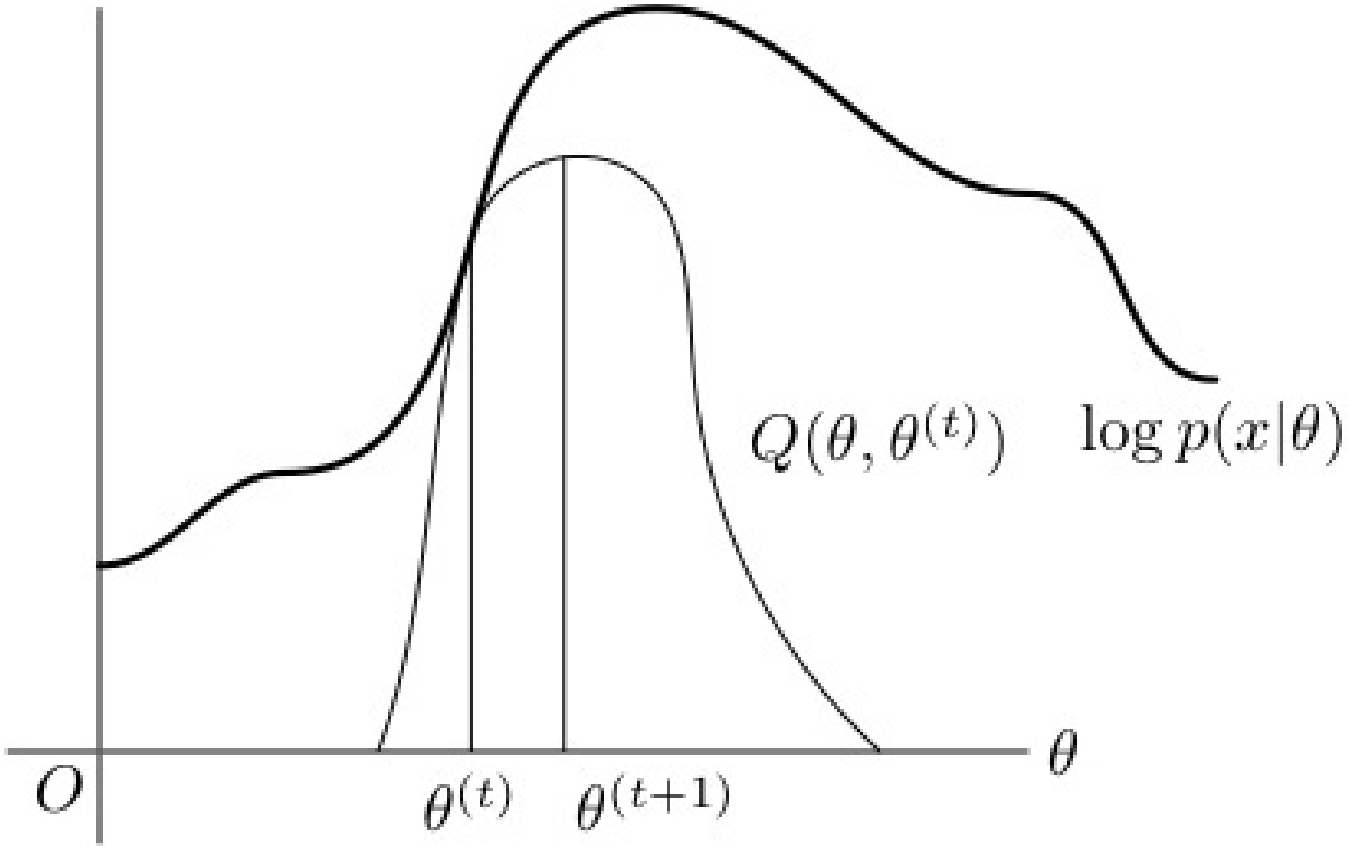


Figure courtesy Rob Nowak.

For $k = 1, \ldots, K$ and $i = 1, \ldots, n$, define the indicator variable:

$$\underbrace{\Delta_{i,k}}_{} = \begin{cases} 1, & \text{if } Z_i = k \\ 0, & \text{if } Z_i \neq k \end{cases} = \mathbb{I}_{\{Z_i = k\}}.$$

Q20.4 Then the **GMM** complete-data log-likelihood becomes:

$$\tilde{\ell}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) = \log\Big(\prod_{i=1}^{n} \mathsf{P}\{Z_i = z_i; \boldsymbol{\theta}\} \, \mathsf{p}(\boldsymbol{x}_i \mid z_i; \boldsymbol{\theta}) \Big)$$
$$= \sum_{i=1}^{n} \log(w_{z_i} \, \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})) = ?$$

A: $\sum_{i=1}^{n} \sum_{k=1}^{K} \Delta_{i,k} (\log w_k + \log \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))$

B: $\sum_{k=1}^{K} \Delta_{k,k} (\log w_k + \sum_{i=1}^{n} \log \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))$

C: $\sum_{i=1}^{n} \sum_{k=1}^{K} (\log w_k + \Delta_{i,k} \log \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))$

D: none of these

When $\Delta_{i,k} = \mathbb{I}_{\{Z_i = k\}}$, we have $\sum_{k=1}^{K} \Delta_{i,k} a_k = a_{Z_i}$ so the complete-data log-likelihood becomes:

$$\ell(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) = \log\Big(\prod_{i=1}^{n} \mathsf{P}\{Z_i = z_i; \boldsymbol{\theta}\} \, \mathsf{p}(\boldsymbol{x}_i \mid z_i; \boldsymbol{\theta}) \Big)$$

$$= \log\left(\prod_{i=1}^{n} w_{z_i} \, \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}) \right) = \sum_{i=1}^{n} \log\big(\underbrace{w_{z_i} \, \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})}_{} \big)$$

$$= \qquad\qquad$$

$$= \qquad\qquad \tag{4}$$

because $f(b_{Z_i}) = f(\sum_k \Delta_{i,k} b_k) = \sum_k \Delta_{i,k} f(b_k)$.

- Denote the current iterate $\boldsymbol{\theta}_t = (w_1^{(t)}, \ldots, w_K^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_K^{(t)}, \boldsymbol{\Sigma}_1^{(t)}, \ldots, \boldsymbol{\Sigma}_K^{(t)})$
- The only random variable in (4) is $\Delta_{i,k} = \Delta_{i,k}(Z_i)$.
- So the E-step requires calculating these "soft class assignments:"

$$\gamma_{i,k}^{(t)} \triangleq \mathbb{E}_{\underline{\boldsymbol{Z}} \sim \mathrm{p}(\underline{\boldsymbol{z}} | \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t)} \{\Delta_{i,k}\} = \mathbb{E}\{\Delta_{i,k} \mid \underline{\boldsymbol{X}} = \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t\}$$

$$= \mathrm{P}\{\Delta_{i,k} = 1 \mid \underline{\boldsymbol{X}} = \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t\} = \mathrm{P}\{Z_i = k \mid \underline{\boldsymbol{X}} = \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t\}$$

$$= \quad \text{(blank)}$$

$$= \frac{w_k^{(t)} \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{j=1}^K w_j^{(t)} \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)})} \in [0, 1]. \text{ "responsibility"}$$

- The M-step for GMM requires maximization:

$$\boldsymbol{\theta}_{t+1} = \arg\max_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t)$$

where from (4) and the E-step:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{i,k}^{(t)}$$

- The w_k update involves the **unit simplex**:

$$w_k^{(t+1)} = \arg\max_{w_k \geq 0 \,:\, \sum_k w_k = 1} \sum_{i=1}^{n} \sum_{k=1}^{K}$$

- The $\boldsymbol{\mu}_k$ update involves a quadratic objective function:

$$\boldsymbol{\mu}_k^{(t+1)} = \arg\max_{\boldsymbol{\mu}_k} -\frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{i,k}^{(t)} \left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)^\top \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)$$

$$\implies \nabla_{\boldsymbol{\mu}_k} Q(\cdot; \boldsymbol{\theta}_t) = \sum_{i=1}^{n}$$

$$\implies \boldsymbol{\mu}_k^{(t+1)} =$$

  which is a responsibility-weighted average (convex combination) of the data points.

- This is a "soft assignment" version of the centroid calculation in the k-means algorithm.

- The $\boldsymbol{\Sigma}_k$ update involves the determinant (cf HW):

$$\boldsymbol{\Sigma}_k^{(t+1)} = \arg\max_{\boldsymbol{\Sigma}_k} -\frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{i,k}^{(t)} \left[\log |\boldsymbol{\Sigma}_k| + (\boldsymbol{x}_i - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k) \right]$$

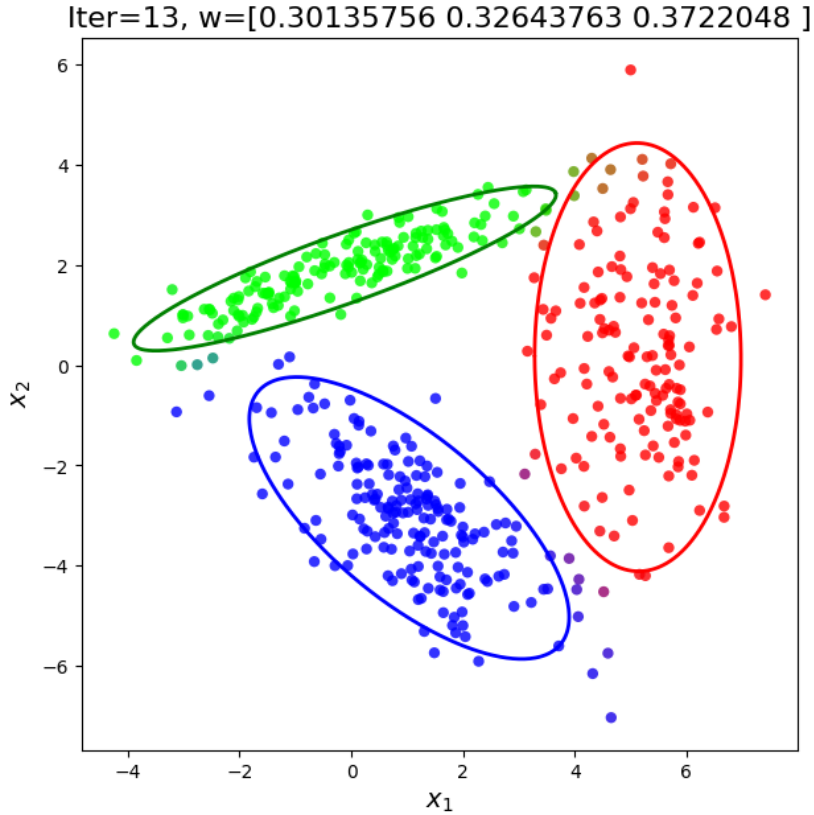
$$= \cdots$$

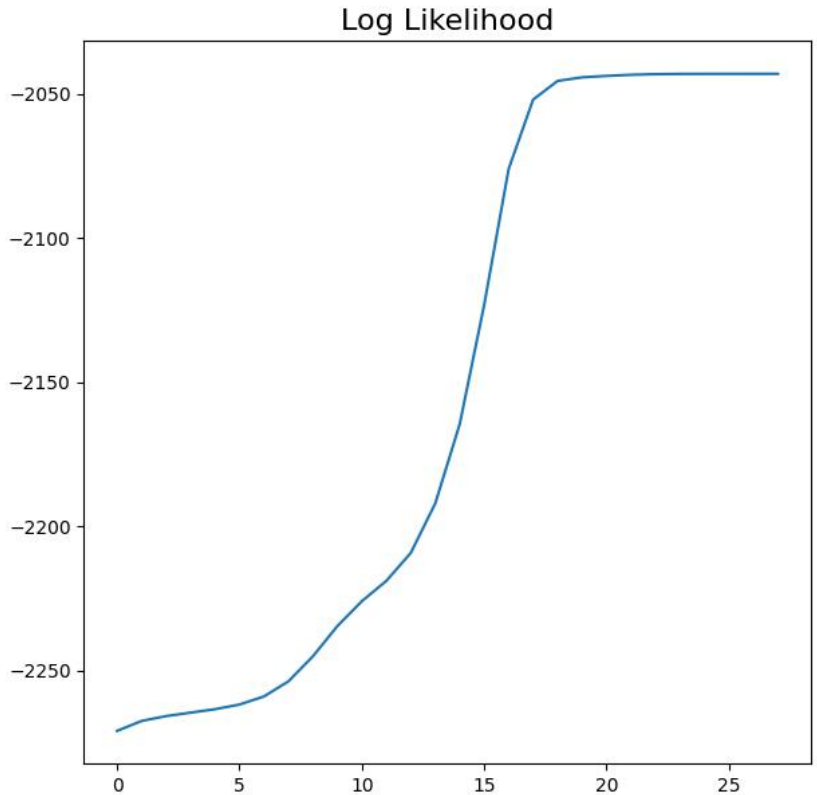
$$=$$

using some matrix derivatives from the matrix cookbook.

- This is a responsibility-weighted sample covariance.
- It uses the updated mean vectors $\{\boldsymbol{\mu}_k\}$.

Python version





- RGB coloring of
- log-likelihood increases and eventually converges
- Unlike k-means, here we have "soft assignments"
- Like k-means, this involves a nonconvex optimization problem

Alternatives to EM for GMM?

© 2026 Fessler

Could we apply GD instead of EM for GMM parameter estimation?

-
-

CQ: Does EM algorithm above guarantee that each $\boldsymbol{\Sigma}_k$ remains invertible?

Clustering is a huge field

© 2026 Fessler

- centroid based (*e.g.*, k-means) and distribution based (*e.g.*, GMM)
- hierarchical clustering
- density-based clustering
- graph clustering
- clustering non-real-valued data
- model selection
- clustering quality metrics

References I

[1] K. P. Murphy.
Probabilistic machine learning: An introduction.
MIT Press, 2022.

[2] Dongnan Liu, Donghao Zhang, Yang Song, Fan Zhang, Lauren O'Donnell, Heng Huang, Mei Chen, and Weidong Cai.
Unsupervised instance segmentation in microscopy images via panoptic domain adaptation and task re-weighting.
In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4243–4252, 2020.

[3] John Lipor and Laura Balzano.
Clustering quality metrics for subspace clustering.
Pattern Recognition, 104:107328, 2020.

[4] Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat.
Np-hardness of euclidean sum-of-squares clustering.
Machine learning, 75(2):245–248, 2009.

[5] Sanjoy Dasgupta.
The hardness of k-means clustering.
Technical report, University of California, San Diego, 2008.

[6] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan.
The planar k-means problem is np-hard.
Theoretical computer science, 442:13–21, 2012.

[7] Christopher M Bishop and Nasser M Nasrabadi.
Pattern recognition and machine learning, volume 4.
Springer, 2006.

References II

[8] C. Bishop.
Pattern recognition and machine learning.
Springer, 2006.

[9] Donald B Rubin.
Em and beyond.
Psychometrika, 56(2):241–254, 1991.

[10] Zoubin Ghahramani and Michael I Jordan.
Mixture models for learning from incomplete data.
Computational learning theory and natural learning systems, 4:67–85, 1997.

[11] R. O. Duda, P. E. Hart, and D. G. Stork.
Pattern classification.
Wiley, New York, 2001.

[12] K. Lange.
MM optimization algorithms.
Soc. Indust. Appl. Math., 2016.

[13] M. W. Jacobson and J. A. Fessler.
An expanded theoretical treatment of iteration-dependent majorize-minimize algorithms.
IEEE Trans. Im. Proc., 16(10):2411–22, October 2007.

[14] CF Jeff Wu.
On the convergence properties of the em algorithm.
The Annals of statistics, pages 95–103, 1983.

[15] J. A. Fessler and A. O. Hero.
Space-alternating generalized expectation-maximization algorithm.
IEEE Trans. Sig. Proc., 42(10):2664–77, October 1994.