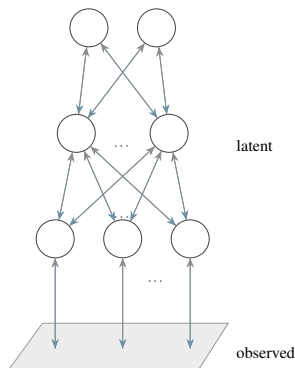


LATENT VARIABLE MODELS

LATENT VARIABLE MODELS

Explain correlations in $\mathbf{x}$ by assuming dependence on latent variables $\mathbf{z}$



$$\mathbf{z} \sim \mathcal{P}[\theta_z]$$

$$\mathbf{x} \mid \mathbf{z} \sim \mathcal{P}[\theta_x]$$

$$p(\mathbf{x}, \mathbf{z}; \theta_x, \theta_z) = p(\mathbf{x} \mid \mathbf{z}; \theta_x) p(\mathbf{z}; \theta_z)$$

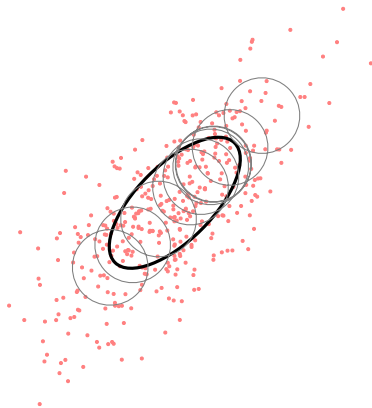
$$p(\mathbf{x}; \theta_x, \theta_z) = \int d\mathbf{z} p(\mathbf{x} \mid \mathbf{z}; \theta_x) p(\mathbf{z}; \theta_z)$$

LATENT VARIABLE MODELS

- Describe structured distributions.
 - Correlations in high-dimensional $\mathbf{x}$ may be captured by fewer parameters.
- Capture an underlying generative process.
 - $\mathbf{z}$ may describe *causes* of $\mathbf{x}$.
 - help to separate signal from noise.
- Combine exponential family distributions into richer, more flexible forms.
 - $P(\mathbf{z})$, $P(\mathbf{x}|\mathbf{z})$ and even $P(\mathbf{x}, \mathbf{z})$ may be in the exponential family
 - $P(\mathbf{x})$ rarely is. (Exception: Linear Gaussian models).

LATENT VARIABLES AND GAUSSIANS

Gaussian correlation can be composed from latent components and uncorrelated noise.



$$\mathbf{x} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix}\right) \quad \Leftrightarrow \quad z \sim \mathcal{N}(0, 1) \quad \mathbf{x} \sim \mathcal{N}\left(\sqrt{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix} z, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

PROBABILISTIC PRINCIPAL COMPONENTS ANALYSIS

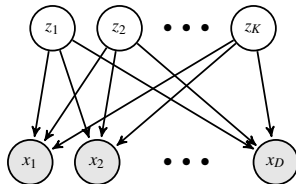
If the uncorrelated noise is assumed to be isotropic, this model is called **Probabilistic Principal Components Analysis**, or **PPCAB**.

Data: $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}; \mathbf{x}_i \in \mathbb{R}^d$

Latents: $\mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n\}; \mathbf{z}_i \in \mathbb{R}^K$

Linear generative model: $x_j = \sum_{k=1}^K \Lambda_{jk} z_k + \epsilon_j$

- z_k are independent $\mathcal{N}(0, 1)$ Gaussian **factors**
- ϵ_j are independent $\mathcal{N}(0, \psi)$ Gaussian **noise**
- $K < d$



Model for observations $\mathbf{x}$ is a correlated Gaussian:

$$p(\mathbf{z}) = \mathcal{N}(0, I)$$

$$\text{Note: } \mathbb{E}_{\mathbf{x}} [f(\mathbf{x})] = \mathbb{E}_{\mathbf{z}} [\mathbb{E}_{\mathbf{x}|\mathbf{z}} [f(\mathbf{x})]]$$

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\Lambda\mathbf{z}, \psi I)$$

$$\mathbb{V}_{\mathbf{x}} [x] = \mathbb{E}_{\mathbf{z}} [\mathbb{V} [\mathbf{x}|\mathbf{z}]] + \mathbb{V}_{\mathbf{z}} [\mathbb{E} [\mathbf{x}|\mathbf{z}]]$$

$$p(\mathbf{x}) = \int p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z} = \mathcal{N}\left(\mathbb{E}_{\mathbf{z}} [\Lambda\mathbf{z}], \mathbb{E}_{\mathbf{z}} [\Lambda\mathbf{z}\mathbf{z}^T\Lambda^T] + \psi I\right) = \mathcal{N}\left(0, \Lambda\Lambda^T + \psi I\right)$$

where Λ is a $d \times K$ matrix.

MULTIVARIATE GAUSSIANS AND LATENT VARIABLES

Two models:

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \Sigma)$$

- **Descriptive** density model: correlations are captured by off-diagonal elements of Σ .
- Σ has $\frac{D(d+1)}{2}$ free parameters.
- Only constrained to be positive definite.
- Simple ML estimate.

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, I)$$

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\Lambda\mathbf{z}, \psi I)$$

$$\Rightarrow p(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \Lambda\Lambda^\top + \psi I)$$

- **Interpretable** causal model: correlations captured by common influence of latent variable.
- $\Lambda\Lambda^\top + \psi I$ has $dK + 1$ free parameters.
- For $K < d$ covariance structure is constrained (“blurry pancake”)
- ML estimation is more complex.

PPCA LIKELIHOOD

The marginal distribution on $\mathbf{x}$ gives us the PPCA likelihood:

$$\log p(\mathcal{D}|\Lambda, \psi) = -\frac{n}{2} \log \left| 2\pi(\Lambda\Lambda^\top + \psi I) \right| - \frac{1}{2} \text{Tr} \left[(\Lambda\Lambda^\top + \psi I)^{-1} \underbrace{\sum_i \mathbf{x}\mathbf{x}^\top}_{nS} \right]$$

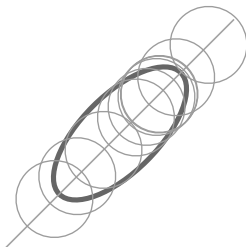
To find the ML values of (Λ, ψ) we could optimize numerically (gradient ascent / Newton's method), or we could use a different iterative algorithm called EM which we'll introduce soon.

In fact, however, ML for PPCA is more straightforward in principle, as we will see by first considering the limit $\psi \rightarrow 0$.

[Note: We may also add a constant mean μ to the output, so as to model data that are not distributed around 0. In this case, the ML estimate $\hat{\mu} = \frac{1}{n} \sum_i \mathbf{x}_i$ and we can define $S = \frac{1}{n} \sum_i (\mathbf{x}_i - \hat{\mu})(\mathbf{x}_i - \hat{\mu})^\top$ in the likelihood above.]

THE $\psi \rightarrow 0$ LIMIT

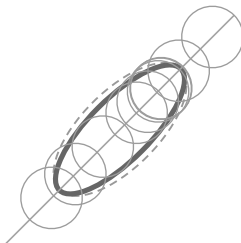
As $\psi \rightarrow 0$, the latent model can only capture K dimensions of variance.



In a Gaussian model, the ML parameters will find the K -dimensional space of **most** variance.

THE $\psi \rightarrow 0$ LIMIT

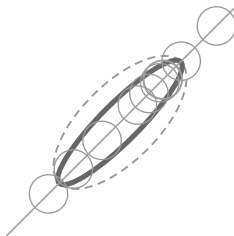
As $\psi \rightarrow 0$, the latent model can only capture K dimensions of variance.



In a Gaussian model, the ML parameters will find the K -dimensional space of **most** variance.

THE $\psi \rightarrow 0$ LIMIT

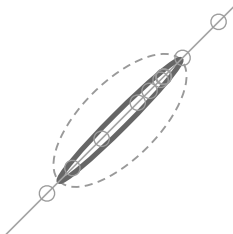
As $\psi \rightarrow 0$, the latent model can only capture K dimensions of variance.



In a Gaussian model, the ML parameters will find the K -dimensional space of **most** variance.

THE $\psi \rightarrow 0$ LIMIT

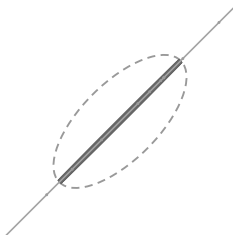
As $\psi \rightarrow 0$, the latent model can only capture K dimensions of variance.



In a Gaussian model, the ML parameters will find the K -dimensional space of **most** variance.

THE $\psi \rightarrow 0$ LIMIT

As $\psi \rightarrow 0$, the latent model can only capture K dimensions of variance.



In a Gaussian model, the ML parameters will find the K -dimensional space of **most** variance.

PRINCIPAL COMPONENTS ANALYSIS

This leads us to an (old) algorithm called Principal Components Analysis (PCA).

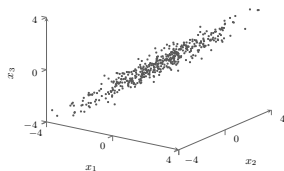
Assume data $\mathcal{D} = \{\mathbf{x}_i\}$ have zero mean (if not, subtract it).

- Find direction of greatest variance – $\lambda_{(1)}$.

$$\lambda_{(1)} = \arg \max_{\|\mathbf{v}\|=1} \sum_i (\mathbf{x}_i^T \mathbf{v})^2$$

- Find direction orthogonal to $\lambda_{(1)}$ with greatest variance – $\lambda_{(2)}$
- $\vdots$

- Find direction orthogonal to $\{\lambda_{(1)}, \lambda_{(2)}, \dots, \lambda_{(k-1)}\}$ with greatest variance – $\lambda_{(k)}$.
- Terminate when remaining variance drops below a threshold.



EIGENDECOMPOSITION OF A COVARIANCE MATRIX

The eigendecomposition of a covariance matrix makes finding the PCs easy.

Recall that $\mathbf{u}$ is an **eigenvector**, with scalar **eigenvalue** ω , of a matrix S if

$$S\mathbf{u} = \omega\mathbf{u}$$

$\mathbf{u}$ can have any norm, but we will define it to be unity (i.e., $\mathbf{u}^T\mathbf{u} = 1$).

For a covariance matrix $S = \langle \mathbf{x}\mathbf{x}^T \rangle$ (which is $d \times d$, symmetric, positive semi-definite):

- In general there are d eigenvector-eigenvalue pairs $(\mathbf{u}_{(i)}, \omega_{(i)})$, except if two or more eigenvectors share the same eigenvalue (in which case the eigenvectors are degenerate — any linear combination is also an eigenvector).
- The d eigenvectors are orthogonal (or orthogonalizable, if $\omega_{(i)} = \omega_{(j)}$). Thus, they form an **orthonormal basis**. $\sum_i \mathbf{u}_{(i)}\mathbf{u}_{(i)}^T = I$.
- Any vector $\mathbf{v}$ can be written as

$$\mathbf{v} = \left(\sum_i \mathbf{u}_{(i)}\mathbf{u}_{(i)}^T \right) \mathbf{v} = \sum_i (\mathbf{u}_{(i)}^T \mathbf{v}) \mathbf{u}_{(i)} = \sum_i v_{(i)} \mathbf{u}_{(i)}$$

- The original matrix S can be written:

$$S = \sum_i \omega_{(i)} \mathbf{u}_{(i)} \mathbf{u}_{(i)}^T = U W U^T$$

where $U = [\mathbf{u}_{(1)}, \mathbf{u}_{(2)}, \dots, \mathbf{u}_{(D)}]$ collects the eigenvectors and $W = \text{diag}[(\omega_{(1)}, \omega_{(2)}, \dots, \omega_{(D)})]$.

PCA AND EIGENVECTORS

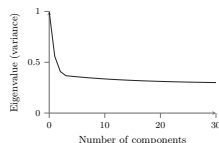
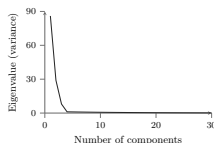
- The variance in direction $\mathbf{u}_{(i)}$ is

$$\langle (\mathbf{x}^T \mathbf{u}_{(i)})^2 \rangle = \langle \mathbf{u}_{(i)}^T \mathbf{x} \mathbf{x}^T \mathbf{u}_{(i)} \rangle = \mathbf{u}_{(i)}^T S \mathbf{u}_{(i)} = \mathbf{u}_{(i)}^T \omega_{(i)} \mathbf{u}_{(i)} = \omega_{(i)}$$

- The variance in an arbitrary direction $\mathbf{v}$ is

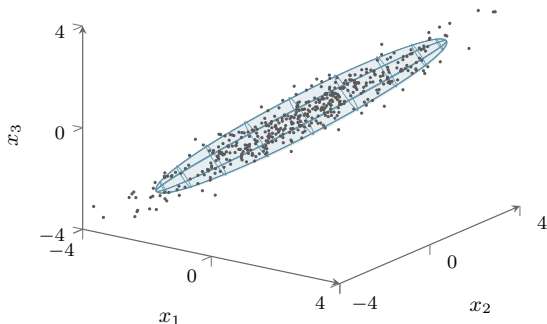
$$\begin{aligned} \langle (\mathbf{x}^T \mathbf{v})^2 \rangle &= \left\langle \left(\mathbf{x}^T \left(\sum_i v_{(i)} \mathbf{u}_{(i)} \right) \right)^2 \right\rangle = \sum_{ij} v_{(i)} \mathbf{u}_{(i)}^T S \mathbf{u}_{(j)} v_{(j)} \\ &= \sum_{ij} v_{(i)} \omega_{(j)} v_{(j)} \mathbf{u}_{(i)}^T \mathbf{u}_{(j)} = \sum_i v_{(i)}^2 \omega_{(i)} \end{aligned}$$

- If $\mathbf{v}^T \mathbf{v} = 1$, then $\sum_i v_{(i)}^2 = 1$ and so $\arg \max_{\|\mathbf{v}\|=1} \langle (\mathbf{x}^T \mathbf{v})^2 \rangle = \mathbf{u}_{(\max)}$
The direction of greatest variance is the eigenvector the largest eigenvalue.
- In general, the PCs are exactly the eigenvectors of the empirical covariance matrix, ordered by decreasing eigenvalue.
- The **eigenspectrum** shows how the variance is distributed across dimensions; can identify transitions that might separate signal from noise, or the number of PCs that capture a pre-determined fraction of variance.



PCA SUBSPACE

The K principle components define the K -dimensional subspace of greatest variance.

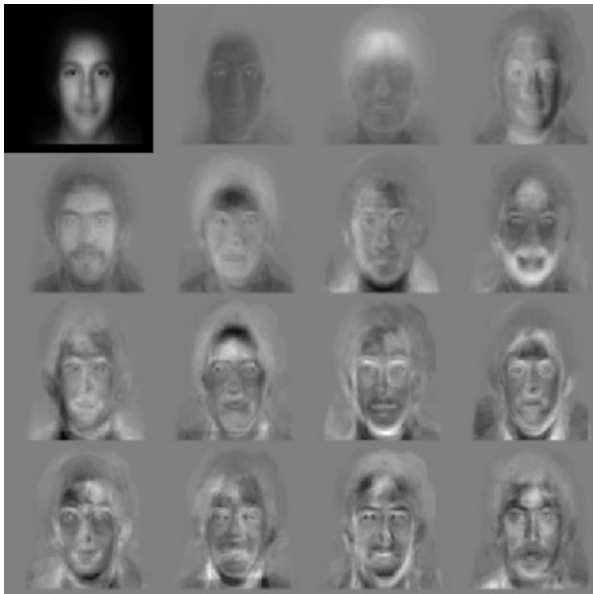


- Each data point $\mathbf{x}_i$ is associated with a **projection** $\hat{\mathbf{x}}_i$ into the principle subspace.

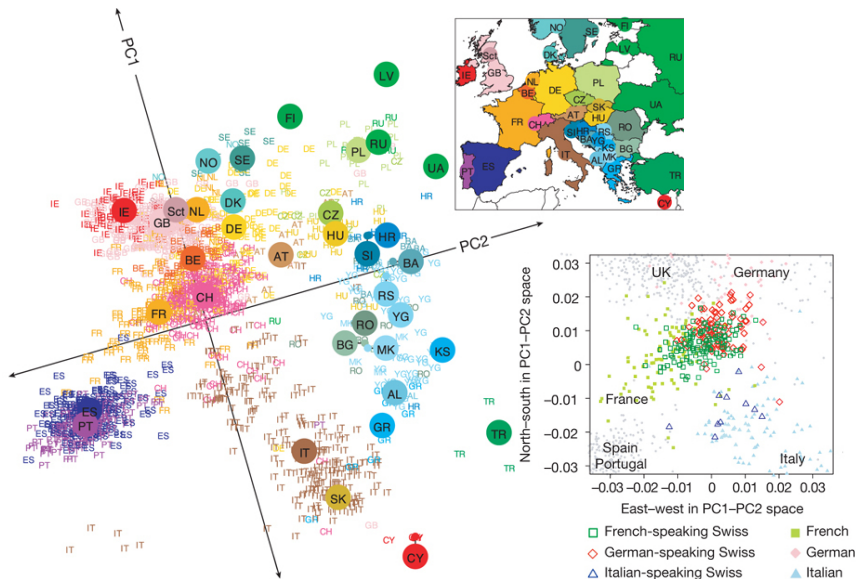
$$\hat{\mathbf{x}}_i = \sum_{k=1}^K x_{i(k)} \lambda_{(k)}$$

- This can be used for lossy compression, denoising, recognition, ...

EXAMPLE OF PCA: EIGENFACES



EXAMPLE OF PCA: GENETIC VARIATION WITHIN EUROPE



EQUIVALENT DEFINITIONS OF PCA

- Find K directions of greatest variance in data.
- Find K -dimensional orthogonal projection that *preserves* greatest variance.
- Find K -dimensional vectors $\mathbf{z}_i$ and matrix Λ so that $\hat{\mathbf{x}}_i = \Lambda \mathbf{z}_i$ is as close as possible (in squared distance) to $\mathbf{x}_i$.
- ... (many others)

ML LEARNING FOR PPCA

$$\ell = -\frac{n}{2} \log |2\pi C| - \frac{n}{2} \text{Tr} [C^{-1}S] \quad \text{where } C = \Lambda\Lambda^T + \psi I$$

$$\frac{\partial \ell}{\partial \Lambda} = \frac{n}{2} \left(-\frac{\partial}{\partial \Lambda} \log |C| - \frac{\partial}{\partial \Lambda} \text{Tr} [C^{-1}S] \right) = n \left(-C^{-1}\Lambda + C^{-1}SC^{-1}\Lambda \right)$$

So at the stationary points we have $SC^{-1}\Lambda = \Lambda$. This implies either:

- $\Lambda = 0$, which turns out to be a minimum (if ψ is smaller than all eigenvalues of S), or a saddle point (otherwise).
- $C = S \Rightarrow \Lambda\Lambda^T = S - \psi I$. Now $\text{rank}(\Lambda\Lambda^T) \leq K \Rightarrow \text{rank}(S - \psi I) \leq K \Rightarrow S$ has $d - K$ eigenvalues $= \psi$ and Λ aligns with space of remaining eigenvectors.
- or, taking the SVD: $\Lambda = ULV^T$:

$$\begin{aligned} S(ULV^T VLU^T + \psi I)^{-1} ULV^T &= ULV^T && \times VL^{-1} \\ \Rightarrow S(UL^2U^T + \psi I)^{-1} U &= U && U(L^2 + \psi I) = (UL^2U^T + \psi I)U \\ &&& \Rightarrow (UL^2U^T + \psi I)^{-1} U = U(L^2 + \psi I)^{-1} \\ \Rightarrow SU(L^2 + \psi I)^{-1} &= U && \times (L^2 + \psi I) \\ \Rightarrow SU &= \underbrace{U(L^2 + \psi I)}_{\text{diagonal}} \end{aligned}$$

$\Rightarrow$ columns of U are eigenvectors of S with eigenvalues given by $l_i^2 + \psi$.

Thus, $\Lambda = ULV^T$ spans a space defined by K eigenvectors of S ; and the lengths of the column vectors of L are $l_i = \sqrt{\omega_{(i)} - \psi}$ (V selects an arbitrary basis in the latent space).

It remains to show—we won't, but it's intuitively reasonable—that the global ML solution is attained when Λ aligns with the K leading eigenvectors.

- In PCA the “noise” is orthogonal to the subspace, and we can project $\mathbf{x}_i \rightarrow \hat{\mathbf{x}}_i$ trivially.
- In PPCA, the noise is more sensible (equal in all directions). But what is the projection?

Find the expected value $\bar{\mathbf{z}}_i = \mathbb{E}[\mathbf{z}_i|\mathbf{x}_i]$ and then take $\hat{\mathbf{x}}_i = \Lambda\bar{\mathbf{z}}_i$.

- **Tactic:** First consider a general Ψ . Write $p(\mathbf{z}_i, \mathbf{x}_i|\theta)$, consider $\mathbf{x}_i$ to be fixed.

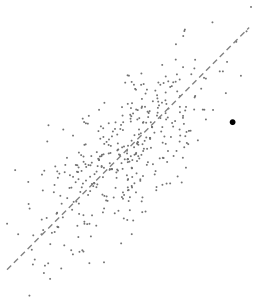
What is this as a function of $\mathbf{z}_i$?

$$\begin{aligned} p(\mathbf{z}_i, \mathbf{x}_i) &= p(\mathbf{z}_i)p(\mathbf{x}_i|\mathbf{z}_i) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{z}_i^\top \mathbf{z}_i\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_i - \Lambda\mathbf{z}_i)^\top \Psi^{-1}(\mathbf{x}_i - \Lambda\mathbf{z}_i)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{z}_i^\top \mathbf{z}_i + (\mathbf{x}_i - \Lambda\mathbf{z}_i)^\top \Psi^{-1}(\mathbf{x}_i - \Lambda\mathbf{z}_i)]\right\} \\ &= c' \times \exp\left\{-\frac{1}{2}[\mathbf{z}_i^\top (I + \Lambda^\top \Psi^{-1} \Lambda) \mathbf{z}_i - 2\mathbf{z}_i^\top \Lambda^\top \Psi^{-1} \mathbf{x}_i]\right\} \\ &= c'' \times \exp\left\{-\frac{1}{2}[\mathbf{z}_i^\top \Sigma^{-1} \mathbf{z}_i - 2\mathbf{z}_i^\top \Sigma^{-1} \mu + \mu^\top \Sigma^{-1} \mu]\right\} \end{aligned}$$

So $\Sigma = (I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} = I - \beta \Lambda$ and $\mu = \Sigma \Lambda^\top \Psi^{-1} \mathbf{x}_i = \beta \mathbf{x}_i$. Where $\beta = \Sigma \Lambda^\top \Psi^{-1}$.

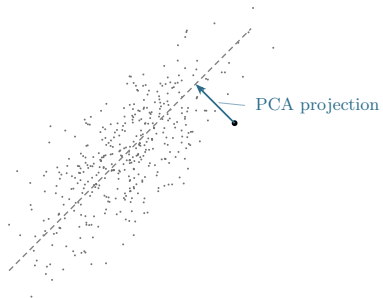
- Thus, $\hat{\mathbf{x}}_i = \Lambda(I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} \Lambda^\top \Psi^{-1} \mathbf{x}_i = \mathbf{x}_i - \Psi(\Lambda \Lambda^\top + \Psi)^{-1} \mathbf{x}_i$
- This is not the same projection. PPCA takes into account noise in the principal subspace.
- Now suppose $\Psi = \psi I$. As $\psi \rightarrow 0$, the PPCA estimate $\rightarrow$ the PCA value.

PPCA LATENTS



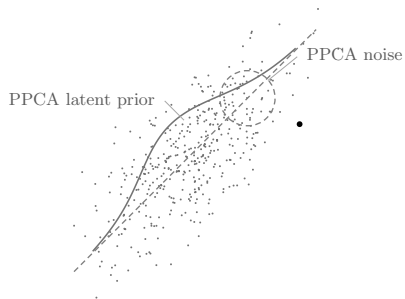
dashed line = principal subspace

PPCA LATENTS



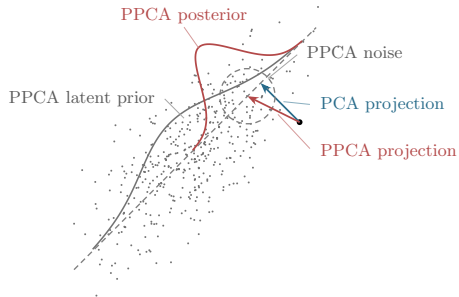
dashed line = principal subspace

PPCA LATENTS



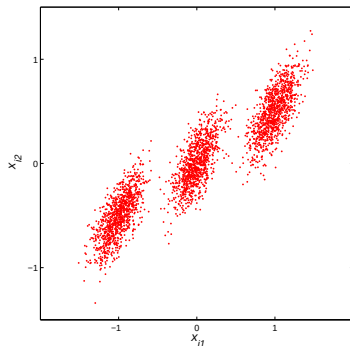
dashed line = principal subspace

PPCA LATENTS



dashed line = principal subspace

MIXTURE DISTRIBUTIONS



A (finite) mixture distribution has a single discrete latent variable:

$$s_i \stackrel{\text{iid}}{\sim} \text{Discrete}[\pi]$$

$$\mathbf{x}_i \mid s_i \sim \mathcal{P}_{s_i}[\theta_{s_i}]$$

Mixtures arise naturally when observations from different sources have been collated. They can also be used to *approximate* arbitrary distributions.

THE MIXTURE LIKELIHOOD

The mixture model is

$$\begin{aligned}s_i &\stackrel{\text{iid}}{\sim} \text{Discrete}[\pi] \\ \mathbf{x}_i \mid s_i &\sim \mathcal{P}_{s_i}[\theta_{s_i}]\end{aligned}$$

Under the discrete distribution

$$P(s_i = k) = \pi_k; \quad \pi_k \geq 0, \quad \sum_{k=1}^K \pi_k = 1$$

Thus, the probability (density) at a single data point $\mathbf{x}_i$ is

$$\begin{aligned}P(\mathbf{x}_i) &= \sum_{k=1}^K P(\mathbf{x}_i \mid s_i = k)P(s_i = k) \\ &= \sum_{k=1}^K \pi_k P_k(\mathbf{x}_i; \theta_k)\end{aligned}$$

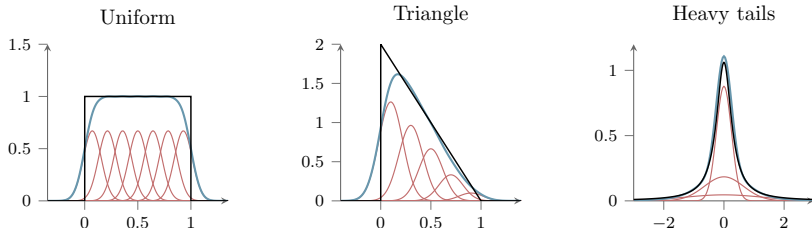
The mixture distribution (density) is a convex combination (or *weighted average*) of the component distributions (densities).

APPROXIMATION WITH A MIXTURE OF GAUSSIANS

We now consider a **mixture of Gaussians (MoG)**, ie each component distribution is Gaussian.

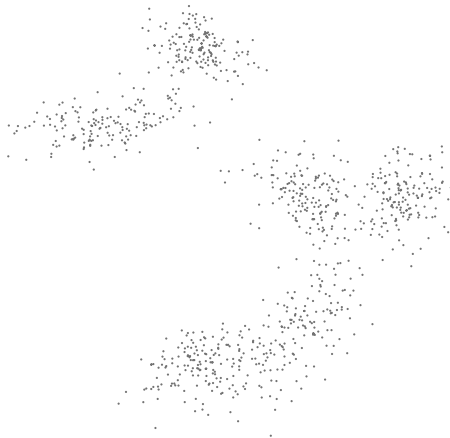
The component densities may be viewed as elements of a “basis” which can be combined to approximate arbitrary distributions.

Here are examples where non-Gaussian densities are modelled (approximated) as a mixture of Gaussians. The red curves show the (weighted) Gaussians, and the blue curve the resulting density.

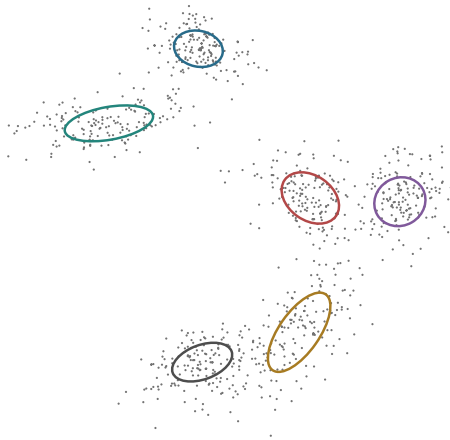


Given enough mixture components we can model (almost) any density (as accurately as desired), but still only need to work with the well-known Gaussian form.

CLUSTERING WITH A MoG



CLUSTERING WITH A MoG



CLUSTERING WITH A MoG

In clustering applications, the latent variable s_i represents the (unknown) identity of the cluster to which the i th observation belongs.

Thus, the latent distribution gives the *prior* probability of a data point coming from each cluster.

$$P(s_i = k \mid \pi) = \pi_k$$

Data from the k th cluster are distributed according to the k th component:

$$P(\mathbf{x}_i \mid s_i = k) = P_k(\mathbf{x}_i)$$

Once we observe a data point, the *posterior* probability distribution for the cluster it belongs to is

$$P(s_i = k \mid \mathbf{x}_i) = \frac{P_k(\mathbf{x}_i)\pi_k}{\sum_k P_k(\mathbf{x}_i)\pi_k}$$

This is often called the **responsibility** of the k th cluster for the i th data point.

THE MOG LIKELIHOOD

Each component of a MoG is a Gaussian, with mean μ_k and covariance matrix Σ_k . Thus, the probability density evaluated at a set of n iid observations, $\mathcal{D} = \{\mathbf{x}_1 \dots \mathbf{x}_n\}$ (i.e. the likelihood) is

$$\begin{aligned} p(\mathcal{D} \mid \{\mu_k\}, \{\Sigma_k\}, \pi) &= \prod_{i=1}^n \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_i \mid \mu_k, \Sigma_k) \\ &= \prod_{i=1}^n \sum_{k=1}^K \pi_k \frac{1}{\sqrt{|2\pi\Sigma_k|}} e^{-\frac{1}{2}(\mathbf{x}_i - \mu_k)^\top \Sigma_k^{-1}(\mathbf{x}_i - \mu_k)} \end{aligned}$$

The log of the likelihood is

$$\log p(\mathcal{D} \mid \{\mu_k\}, \{\Sigma_k\}, \pi) = \sum_{i=1}^n \log \sum_{k=1}^K \pi_k \frac{1}{\sqrt{|2\pi\Sigma_k|}} e^{-\frac{1}{2}(\mathbf{x}_i - \mu_k)^\top \Sigma_k^{-1}(\mathbf{x}_i - \mu_k)}$$

Note that the logarithm fails to simplify the component density terms. A mixture distribution does not lie in the exponential family. Direct optimization is not easy.

MAXIMUM LIKELIHOOD FOR A MIXTURE MODEL

The log likelihood is:

$$\mathcal{L} = \sum_{i=1}^n \log \sum_{k=1}^K \pi_k P_k(\mathbf{x}_i; \theta_k)$$

Its partial derivative wrt θ_k is

$$\frac{\partial \mathcal{L}}{\partial \theta_k} = \sum_{i=1}^n \frac{\pi_k}{\sum_{k=1}^K \pi_k P_k(\mathbf{x}_i; \theta_k)} \frac{\partial P_k(\mathbf{x}_i; \theta_k)}{\partial \theta_k}$$

or, using $\partial P / \partial \theta = P \times \partial \log P / \partial \theta$,

$$\begin{aligned} &= \sum_{i=1}^n \underbrace{\frac{\pi_k P_k(\mathbf{x}_i; \theta_k)}{\sum_{k=1}^K \pi_k P_k(\mathbf{x}_i; \theta_k)}}_{r_{ik}} \frac{\partial \log P_k(\mathbf{x}_i; \theta_k)}{\partial \theta_k} \\ &= \sum_{i=1}^n r_{ik} \frac{\partial \log P_k(\mathbf{x}_i; \theta_k)}{\partial \theta_k} \end{aligned}$$

And its partial derivative wrt π_k is

$$\frac{\partial \mathcal{L}}{\partial \pi_k} = \sum_{i=1}^n \frac{P_k(\mathbf{x}_i; \theta_k)}{\sum_{k=1}^K \pi_k P_k(\mathbf{x}_i; \theta_k)} = \sum_{i=1}^n \frac{r_{ik}}{\pi_k}$$

For a MoG, with $\theta_k = \{\mu_k, \Sigma_k\}$ we get

$$\frac{\partial \mathcal{L}}{\partial \mu_k} = \sum_{i=1}^n r_{ik} \Sigma_k^{-1} (\mathbf{x}_i - \mu_k)$$
$$\frac{\partial \mathcal{L}}{\partial \Sigma_k^{-1}} = \frac{1}{2} \sum_{i=1}^n r_{ik} \left(\Sigma_k - (\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^\top \right)$$

These equations can be used (along with Lagrangian derivatives wrt π_k that enforce normalization) for gradient based learning; e.g., taking small steps in the direction of the gradient (or using conjugate gradients).

THE K-MEANS ALGORITHM

The K-means algorithm is a limiting case of the mixture of Gaussians (c.f. PCA).

Take $\pi_k = 1/K$ and $\Sigma_k = \sigma^2 I$, with $\sigma^2 \rightarrow 0$. Then the responsibilities become binary

$$r_{ik} \rightarrow \delta(k, \arg \min_l \|\mathbf{x}_i - \mu_l\|^2)$$

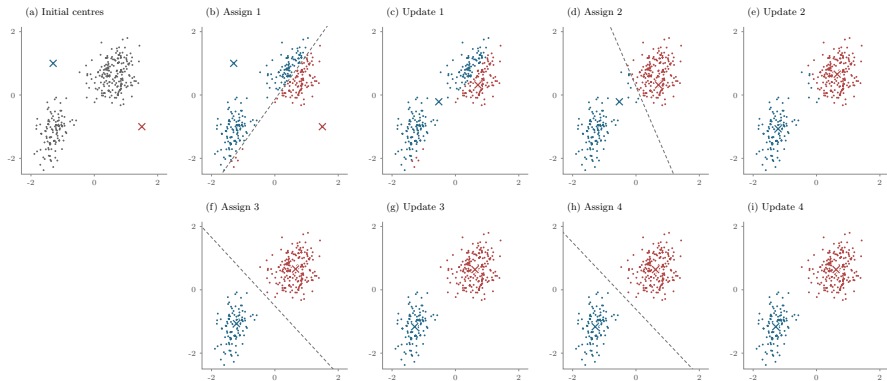
with 1 for the component with the closest mean and 0 for all other components. We can then solve directly for the means by setting the gradient to 0.

The **k-means algorithm** iterates these two steps:

- assign each point to its closest mean $\left(\text{set } r_{ik} = \delta(k, \arg \min_l \|\mathbf{x}_i - \mu_l\|^2) \right)$
- update the means to the average of their assigned points $\left(\text{set } \mu_k = \frac{\sum_i r_{ik} \mathbf{x}_i}{\sum_i r_{ik}} \right)$

This usually converges within a few iterations, although the fixed point depends on the initial values chosen for μ_k . The algorithm has no learning rate, but the assumptions are quite limiting.

ILLUSTRATION: k -MEANS



A PREVIEW OF THE EM ALGORITHM

We wrote the k-means algorithm in terms of binary responsibilities. Suppose, instead, we used the fractional responsibilities from the full (non-limiting) MoG, but still neglected the dependence of the responsibilities on the parameters. We could then solve for both μ_k and Σ_k .

The **EM algorithm** for MoGs iterates these two steps:

- Evaluate the responsibilities for each point given the current parameters.
- Optimize the parameters assuming the responsibilities stay fixed:

$$\mu_k = \frac{\sum_i r_{ik} \mathbf{x}_i}{\sum_i r_{ik}} \quad \text{and} \quad \Sigma_k = \frac{\sum_i r_{ik} (\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^\top}{\sum_i r_{ik}}$$

Although this appears *ad hoc*, we will see (later) that it is a special case of a general algorithm, and is actually guaranteed to not decrease the likelihood at each iteration.

There are several problems with these algorithms:

- slow convergence for the gradient based method
- gradient based method may develop invalid covariance matrices
- local optima; the end configuration may depend on the starting state
- how do you adjust K ? Using the likelihood alone is no good.
- singularities; components with a single data point will have their covariance going to zero and the likelihood will tend to infinity.

We will attempt to address many of these as the course goes on.