Dimension reduction and manifold learning

Deep embeddings, clustering, and label information

Eddie Aamari
Département de mathématiques et applications
CNRS, ENS PSL

Master IASD & MATH — Dauphine PSL

Auto-encoders

Back to coding and decoding

Given a dataset $x_1, \ldots, x_n$, assume the existence of latent variables $z \in \mathbb{R}^d$ of low dimension $d \ll p$ such that $x_i \simeq \mathrm{dec}^*(z_i)$.

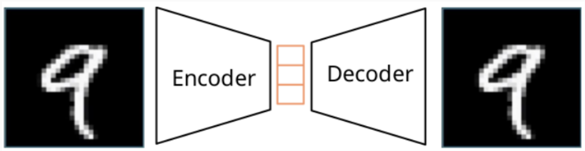
Given a *coder* and $\mathrm{cod} : \mathbb{R}^p \to \mathbb{R}^d$, a *decoder* $\mathrm{dec} : \mathbb{R}^d \to \mathbb{R}^p$

$$\mathbb{R}^p \xrightarrow{\mathrm{cod}} \mathbb{R}^d \xrightarrow{\mathrm{dec}} \mathbb{R}^p.$$

and a loss function $\ell : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$, one can consider the generic reconstruction error

$$\mathcal{E}_{\mathrm{codec}}(\mathrm{cod}, \mathrm{dec}) := \mathbb{E}_x \left[\ell\big(x, \mathrm{dec}(\mathrm{cod}(x))\big) \right].$$

The smallest dimension along the way is called the bottleneck.



PCA considered loss $\ell(x, x') := \|x - x'\|^2$, with linear $\mathrm{cod} = A \in \mathbb{R}^{d \times p}$, $\mathrm{dec} = B \in \mathbb{R}^{p \times d}$

Back to coding and decoding

Given a *coder* architecture $\mathrm{cod}_\theta : \mathbb{R}^p \to \mathbb{R}^d$ and a *decoder* architecture $\mathrm{dec}_\phi : \mathbb{R}^d \to \mathbb{R}^p$, define the population reconstruction risk

$$\mathcal{E}(\theta, \phi) := \mathbb{E}\big[\ell\big(x,\ \mathrm{dec}_\phi(\mathrm{cod}_\theta(x))\big)\big] \, .$$

for $\theta \in \Theta$ and $\phi \in \Xi$.

Training problem (ERM). Given samples $x_1, \ldots, x_n$,

$$(\hat{\theta}, \hat{\phi}) \in \operatorname*{arg\,min}_{\theta,\phi} \ \frac{1}{n} \sum_{i=1}^{n} \ell\big(x_i, \mathrm{dec}_\phi(\mathrm{cod}_\theta(x_i))\big).$$

Squared loss and basic calculus

Write the reconstruction map

$$r_{\theta,\phi}(x) := \operatorname{dec}_\phi(\operatorname{cod}_\theta(x)) \in \mathbb{R}^p.$$

If $\ell(x, \hat{x}) = \|x - \hat{x}\|^2$, then

$$\mathcal{E}(\theta, \phi) = \mathbb{E}\big[\|x - r_{\theta,\phi}(x)\|^2\big].$$

Squared loss and basic calculus

Write the reconstruction map

$$r_{\theta,\phi}(x) := \mathrm{dec}_\phi(\mathrm{cod}_\theta(x)) \in \mathbb{R}^p.$$

If $\ell(x,\hat{x}) = \|x-\hat{x}\|^2$, then

$$\mathcal{E}(\theta,\phi) = \mathbb{E}\big[\|x - r_{\theta,\phi}(x)\|^2\big].$$

For differentiable $\mathrm{cod}_\theta, \mathrm{dec}_\phi$ and any perturbation δr,

$$\delta\mathcal{E} = 2\,\mathbb{E}\big[\langle r(x) - x,\ \delta r(x)\rangle\big].$$

Hence stationarity is governed by the orthogonality condition

$$\mathbb{E}\big[(x - r(x))\,\delta r(x)^\top\big] = 0 \quad \text{for all admissible perturbations } \delta r.$$

Squared loss and basic calculus

Write the reconstruction map

$$r_{\theta,\phi}(x) := \mathrm{dec}_\phi(\mathrm{cod}_\theta(x)) \in \mathbb{R}^p.$$

If $\ell(x, \hat{x}) = \|x - \hat{x}\|^2$, then

$$\mathcal{E}(\theta, \phi) = \mathbb{E}\left[\|x - r_{\theta,\phi}(x)\|^2\right].$$

For differentiable $\mathrm{cod}_\theta, \mathrm{dec}_\phi$ and any perturbation δr,

$$\delta\mathcal{E} = 2\,\mathbb{E}\left[\langle r(x) - x,\ \delta r(x)\rangle\right].$$

Hence stationarity is governed by the orthogonality condition

$$\mathbb{E}\left[(x - r(x))\,\delta r(x)^\top\right] = 0 \quad \text{for all admissible perturbations } \delta r.$$

$\hookrightarrow$ This becomes explicit for linear auto-encoders.

Linear auto-encoder, aka PCA

Assuming centered data $\mathbb{E}[x] = 0$, consider a linear coder/decoder:

$$\mathrm{cod}(x) = Ax, \quad A \in \mathbb{R}^{d\times p}, \qquad \mathrm{dec}(z) = Bz, \quad B \in \mathbb{R}^{p\times d},$$

so that $r(x) = BAx$.

The population risk becomes

$$\mathcal{E}(A,B) = \mathbb{E}\big[\|x - BAx\|^2\big] = \mathrm{Tr}(\Sigma) - 2\,\mathrm{Tr}(BA\Sigma) + \mathrm{Tr}(BA\Sigma A^\top B^\top),$$

where $\Sigma := \mathbb{E}[xx^\top]$.

For fixed A, the objective is quadratic in B. Differentiating w.r.t. B gives the normal equation

$$-2A\Sigma + 2BA\Sigma A^\top = 0 \quad \Longrightarrow \quad B^\star(A) = \Sigma A^\top (A\Sigma A^\top)^{-1} \quad \text{(if } A\Sigma A^\top \text{ invertible).}$$

Plugging back yields a function of the subspace $\mathrm{Im}(A^\top)$ only:

$$\begin{aligned}\mathcal{E}(A, B^\star(A)) &= \mathrm{Tr}(\Sigma) - \mathrm{Tr}\big(\Sigma A^\top (A\Sigma A^\top)^{-1} A\Sigma\big)\\ &= \mathrm{Tr}(\Sigma) - \mathrm{Tr}\big(\mathrm{proj}_{\Sigma^{1/2}\mathrm{Im}(A^\top)}\Sigma\big).\end{aligned}$$

Non-identifiability and normalization constraints

Even at a global optimum, (A, B) is not unique.

If $Q \in \mathbb{R}^{d\times d}$ is invertible, then

$$Ax \mapsto QAx, \qquad Bz \mapsto BQ^{-1}z$$

keeps the reconstruction BAx unchanged.

Common normalizations (to fix a gauge).

- Tied weights: $B = A^\top$ (which we used for PCA earlier in the course).
- Whitening of codes: $\mathbb{E}[zz^\top] = A\Sigma A^\top = I_d$ (or empirically $Z^\top Z = I_d$).
- Orthogonality: $AA^\top = I_d$ (restricts A to an orthogonal projection).

Non-identifiability and normalization constraints

Even at a global optimum, (A, B) is not unique.

If $Q \in \mathbb{R}^{d\times d}$ is invertible, then

$$Ax \mapsto QAx, \qquad Bz \mapsto BQ^{-1}z$$

keeps the reconstruction BAx unchanged.

Common normalizations (to fix a gauge).

- Tied weights: $B = A^\top$ (which we used for PCA earlier in the course).
- Whitening of codes: $\mathbb{E}[zz^\top] = A\Sigma A^\top = I_d$ (or empirically $Z^\top Z = I_d$).
- Orthogonality: $AA^\top = I_d$ (restricts A to an orthogonal projection).

$\hookrightarrow$ For nonlinear AEs, such constraints become implicit via regularization terms / architecture choice).

The simplest non-linear auto-encoder (historical perspective)

A first non-linear extension of PCA is an undercomplete (i.e. $d \ll p$) 1-hidden-layer network (**bourlard1988autoassociation**)

$$x \in \mathbb{R}^p \xrightarrow{\mathrm{cod}_\theta} z = \sigma(Wx + b) \in \mathbb{R}^d \xrightarrow{\mathrm{dec}_\phi} \hat{x} = \tau(Vz + c) \in \mathbb{R}^p.$$

with activations σ, τ (e.g. sigmoid / $\tanh$ / ReLU) and parameters $\theta = (W, b)$, $\phi = (V, c)$.

The simplest non-linear auto-encoder (historical perspective)

A first non-linear extension of PCA is an undercomplete (i.e. $d \ll p$) 1-hidden-layer network (**bourlard1988autoassociation**)

$$x \in \mathbb{R}^p \xrightarrow{\mathrm{cod}_\theta} z = \sigma(Wx + b) \in \mathbb{R}^d \xrightarrow{\mathrm{dec}_\phi} \hat{x} = \tau(Vz + c) \in \mathbb{R}^p.$$

with activations σ, τ (e.g. sigmoid / $\tanh$ / ReLU) and parameters $\theta = (W, b)$, $\phi = (V, c)$.

With squared loss, the population objective is

$$\mathcal{E}(\theta, \phi) = \mathbb{E}\big[\|x - \tau(V\sigma(Wx + b) + c)\|^2\big].$$

The simplest non-linear auto-encoder (historical perspective)

A first non-linear extension of PCA is an undercomplete (i.e. $d \ll p$) 1-hidden-layer network (**bourlard1988autoassociation**)

$$x \in \mathbb{R}^p \xrightarrow{\mathrm{cod}_\theta} z = \sigma(Wx + b) \in \mathbb{R}^d \xrightarrow{\mathrm{dec}_\phi} \hat{x} = \tau(Vz + c) \in \mathbb{R}^p.$$

with activations σ, τ (e.g. sigmoid / $\tanh$ / ReLU) and parameters $\theta = (W, b)$, $\phi = (V, c)$.

With squared loss, the population objective is

$$\mathcal{E}(\theta, \phi) = \mathbb{E}\big[\|x - \tau(V\sigma(Wx + b) + c)\|^2\big].$$

Key point. Unlike the linear case, even with $d \ll p$, nonlinearity can adapt the reconstruction map $r(x) = \mathrm{dec}_\phi(\mathrm{cod}_\theta(x))$ to *curved* low-dimensional structure.

Architectures and losses

Architectures. (Goodfellow, Bengio, and Courville 2016)

- MultiLayer Perceptron: compositions of affine maps and pointwise nonlinearities.
- Convolutional: replace affine maps by convolutions (translation equivariance).
- Residual / U-Net: skip connections to improve optimization and preserve fine-scale details. (He et al. 2016; Ronneberger, Fischer, and Brox 2015)

Architectures and losses

Architectures. (Goodfellow, Bengio, and Courville 2016)

- MultiLayer Perceptron: compositions of affine maps and pointwise nonlinearities.
- Convolutional: replace affine maps by convolutions (translation equivariance).
- Residual / U-Net: skip connections to improve optimization and preserve fine-scale details. (He et al. 2016; Ronneberger, Fischer, and Brox 2015)

Losses.

- Square loss: $\ell(x, \hat{x}) = \|x - \hat{x}\|^2$.
- Bernoulli model (binary data): $\ell(x, \hat{x}) = -\sum_{j=1}^{p} \left(x_j \log \hat{x}_j + (1 - x_j) \log(1 - \hat{x}_j)\right)$ with $x, \hat{x} \in (0,1)^p$ (e.g. $\tau = \mathrm{sigmoid}$).
- Robust losses: Huber / ℓ_1 for heavy-tailed corruption (coordinate-wise).

Architectures and losses

Architectures. (Goodfellow, Bengio, and Courville 2016)

- MultiLayer Perceptron: compositions of affine maps and pointwise nonlinearities.
- Convolutional: replace affine maps by convolutions (translation equivariance).
- Residual / U-Net: skip connections to improve optimization and preserve fine-scale details. (He et al. 2016; Ronneberger, Fischer, and Brox 2015)

Losses.

- Square loss: $\ell(x, \hat{x}) = \|x - \hat{x}\|^2$.
- Bernoulli model (binary data): $\ell(x, \hat{x}) = -\sum_{j=1}^{p} \big(x_j \log \hat{x}_j + (1 - x_j) \log(1 - \hat{x}_j) \big)$ with $x, \hat{x} \in (0,1)^p$ (e.g. $\tau = \mathrm{sigmoid}$).
- Robust losses: Huber / ℓ_1 for heavy-tailed corruption (coordinate-wise).

Optimization.

- Training uses gradient methods (SGD/Adam): backprop computes $\nabla_\theta \ell(x_i, r_{\theta,\phi}(x_i))$ and $\nabla_\phi \ell(x_i, r_{\theta,\phi}(x_i))$ by the chain rule, where $r_{\theta,\phi}(x) := \mathrm{dec}_\phi(\mathrm{cod}_\theta(x))$.

Why regularization is not optional (nonlinear case)

If $\mathrm{cod}, \mathrm{dec}$ are too expressive, the trivial solution $r(x) \approx x$ on the training set may occur: memorization (think of a Hilbert space-filling curve).

Why regularization is not optional (nonlinear case)

If $\mathrm{cod}, \mathrm{dec}$ are too expressive, the trivial solution $r(x) \approx x$ on the training set may occur: memorization (think of a Hilbert space-filling curve).

A standard approach is to penalize complexity and/or enforce structure:

$$\min_{\theta,\phi} \ \frac{1}{n}\sum_{i=1}^{n} \ell\big(x_i, r_{\theta,\phi}(x_i)\big) \ + \ \lambda\, \mathrm{pen}(\theta,\phi).$$

Why regularization is not optional (nonlinear case)

If $\mathrm{cod}, \mathrm{dec}$ are too expressive, the trivial solution $r(x) \approx x$ on the training set may occur: memorization (think of a Hilbert space-filling curve).

A standard approach is to penalize complexity and/or enforce structure:

$$\min_{\theta,\phi} \ \frac{1}{n}\sum_{i=1}^{n} \ell\big(x_i, r_{\theta,\phi}(x_i)\big) \ + \ \lambda \, \mathrm{pen}(\theta, \phi).$$

Typical choices of pen: (Goodfellow, Bengio, and Courville 2016, Chapter 14)

- Weight decay: $\mathrm{pen} = \|\theta\|_2^2 + \|\phi\|_2^2$
- Sparse codes: add $\lambda_s \, \mathbb{E}[\|z(x)\|_1]$ or a KL penalty toward a target sparsity if $z \in (0,1)^d$
- Contractive penalty (stability of the encoder): $\lambda_c \, \mathbb{E}[\|\nabla_x z(x)\|_{\mathrm{F}}^2]$ (Rifai et al. 2011)

Deep auto-encoders and layerwise training

Deep AEs use multiple layers:

$$\mathrm{cod}_\theta = f_L \circ \cdots \circ f_1, \qquad \mathrm{dec}_\phi = g_1 \circ \cdots \circ g_L,$$

with $f_\ell(x) = \sigma(W_\ell x + b_\ell)$, etc.

Deep auto-encoders and layerwise training

Deep AEs use multiple layers:

$$\mathrm{cod}_\theta = f_L \circ \cdots \circ f_1, \qquad \mathrm{dec}_\phi = g_1 \circ \cdots \circ g_L,$$

with $f_\ell(x) = \sigma(W_\ell x + b_\ell)$, etc.

A historically important practical difficulty: optimization (vanishing gradients in early deep nets). A classical workaround was layerwise pretraining: (Bengio et al. 2007; Hinton and Salakhutdinov 2006)

- Train a shallow AE on x to produce $z^{(1)}$.
- Train a second AE on $z^{(1)}$ to produce $z^{(2)}$.
- Stack the encoders and decoders, then fine-tune the full objective by backprop.

Deep auto-encoders and layerwise training

Deep AEs use multiple layers:

$$\mathrm{cod}_\theta = f_L \circ \cdots \circ f_1, \qquad \mathrm{dec}_\phi = g_1 \circ \cdots \circ g_L,$$

with $f_\ell(x) = \sigma(W_\ell x + b_\ell)$, etc.

A historically important practical difficulty: optimization (vanishing gradients in early deep nets). A classical workaround was layerwise pretraining: (Bengio et al. 2007; Hinton and Salakhutdinov 2006)

- Train a shallow AE on x to produce $z^{(1)}$.
- Train a second AE on $z^{(1)}$ to produce $z^{(2)}$.
- Stack the encoders and decoders, then fine-tune the full objective by backprop.

Insight. Pretraining provides an initialization that already approximately preserves information layer by layer. Then, fine-tuning optimizes the end-to-end reconstruction.

More recent AE methods

Modern AEs are often used as representation learners rather than pure compressors.

More recent AE methods

Modern AEs are often used as representation learners rather than pure compressors.

Three common directions.

- Geometric regularization: combine reconstruction with Jacobian penalties / Lipschitz control to obtain stable embeddings.
- Latent prior matching: encourage the empirical latent code distribution to match a target prior while still reconstructing. (Gretton et al. 2012; Makhzani et al. 2016)
- Task-aware regularization: add a supervised term $\ell_{\text{sup}}(h(z), y)$ to align z with downstream prediction task.

More recent AE methods

Modern AEs are often used as representation learners rather than pure compressors.

Three common directions.

- Geometric regularization: combine reconstruction with Jacobian penalties / Lipschitz control to obtain stable embeddings.
- Latent prior matching: encourage the empirical latent code distribution to match a target prior while still reconstructing. (Gretton et al. 2012; Makhzani et al. 2016)
- Task-aware regularization: add a supervised term $\ell_{\text{sup}}(h(z), y)$ to align z with downstream prediction task.

These modifications can be written in a unified way:

$$\min_{\theta,\phi} \; \mathbb{E}[\ell_{\text{rec}}(x, r(x))] \; + \; \lambda_1 \, \mathcal{R}_{\text{geom}}(\theta) \; + \; \lambda_2 \, \mathcal{R}_{\text{latent}}(z_\theta(x)) \; + \; \lambda_3 \, \mathcal{R}_{\text{sup}}(z_\theta(x), y).$$

Implicit regularization via noise injection

To regularize implicitly, a Denoising Auto-Encoder (Vincent et al. 2008) replaces the clean input x by a *intentionally* corrupted one $\tilde{x} = x + \varepsilon$ and simply trains

$$\mathcal{E}_\sigma(r) = \mathbb{E}\big[\|x - r(x+\varepsilon)\|^2\big], \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2 I_p).$$

Implicit regularization via noise injection

To regularize implicitly, a Denoising Auto-Encoder (Vincent et al. 2008) replaces the clean input x by a *intentionally* corrupted one $\tilde{x} = x + \varepsilon$ and simply trains

$$\mathcal{E}_\sigma(r) = \mathbb{E}\big[\|x - r(x+\varepsilon)\|^2\big], \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2 I_p).$$

Heuristic: Assuming r is C^2 and expanding around x,

$$\|x - r(x+\varepsilon)\|^2 = \|x - r(x)\|^2 - 2\langle x - r(x), \nabla r(x)\varepsilon\rangle + \|\nabla r(x)\varepsilon\|^2 + O(\|\varepsilon\|^3).$$

Implicit regularization via noise injection

To regularize implicitly, a Denoising Auto-Encoder (Vincent et al. 2008) replaces the clean input x by a *intentionally* corrupted one $\tilde{x} = x + \varepsilon$ and simply trains

$$\mathcal{E}_\sigma(r) = \mathbb{E}\big[\|x - r(x+\varepsilon)\|^2\big], \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2 I_p).$$

Heuristic: Assuming r is C^2 and expanding around x,

$$\|x - r(x+\varepsilon)\|^2 = \|x - r(x)\|^2 - 2\langle x - r(x), \nabla r(x)\varepsilon\rangle + \|\nabla r(x)\varepsilon\|^2 + O(\|\varepsilon\|^3).$$

Taking expectation over ε conditionally on x gives

$$\mathbb{E}_\varepsilon[\langle x - r(x), \nabla r(x)\varepsilon\rangle \mid x] = 0, \qquad \mathbb{E}_\varepsilon[\|\nabla r(x)\varepsilon\|^2 \mid x] = \sigma^2 \|\nabla r(x)\|_{\mathrm{F}}^2.$$

Therefore, for small σ,

$$\mathcal{E}_\sigma(r) = \mathbb{E}\big[\|x - r(x)\|^2\big] + \sigma^2\, \mathbb{E}\big[\|\nabla r(x)\|_{\mathrm{F}}^2\big] + O(\sigma^3),$$

so denoising acts like a contractive regularization on the reconstruction map. (Bishop 1995)

$\hookrightarrow$ Next: the exact optimality statement for DAEs (posterior mean).

Denoising auto-encoders: objective and first-order optimality

An optimal denoising auto-encoder (DAE) learns to reconstruct x from a corrupted version $\tilde{x}$

$$\tilde{x} := x + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2 I_p) \text{ independent of } x,$$

by minimizing $\mathcal{E}_\sigma(r) := \mathbb{E}\big[\|x - r(\tilde{x})\|^2\big]$ over $r : \mathbb{R}^p \to \mathbb{R}^p$ measurable.

Denoising auto-encoders: objective and first-order optimality

An optimal denoising auto-encoder (DAE) learns to reconstruct x from a corrupted version $\tilde{x}$

$$\tilde{x} := x + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2 I_p) \text{ independent of } x,$$

by minimizing $\mathcal{E}_\sigma(r) := \mathbb{E}\big[\|x - r(\tilde{x})\|^2\big]$ over $r : \mathbb{R}^p \to \mathbb{R}^p$ measurable.

Conditioning on $\tilde{x}$:

$$\mathcal{E}_\sigma(r) = \mathbb{E}\Big[\mathbb{E}\big[\|x - r(\tilde{x})\|^2 \mid \tilde{x}\big]\Big].$$

For each $\tilde{x}$, the inner expectation is minimized by $r^\star_\sigma(\tilde{x}) := \mathbb{E}[x \mid \tilde{x}]$.

Denoising auto-encoders: objective and first-order optimality

An optimal denoising auto-encoder (DAE) learns to reconstruct x from a corrupted version $\tilde{x}$

$$\tilde{x} := x + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2 I_p) \text{ independent of } x,$$

by minimizing $\mathcal{E}_\sigma(r) := \mathbb{E}\big[\|x - r(\tilde{x})\|^2\big]$ over $r : \mathbb{R}^p \to \mathbb{R}^p$ measurable.

Conditioning on $\tilde{x}$:

$$\mathcal{E}_\sigma(r) = \mathbb{E}\Big[\mathbb{E}\big[\|x - r(\tilde{x})\|^2 \mid \tilde{x}\big]\Big].$$

For each $\tilde{x}$, the inner expectation is minimized by $r^\star_\sigma(\tilde{x}) := \mathbb{E}[x \mid \tilde{x}]$.

Thus, an ideal DAE estimates the posterior mean $\mathbb{E}[x \mid \tilde{x}]$ ($\sim$ regression problem).

DAE & score-like vector field

Theorem ((Alain and Bengio 2014), informal)

Let p be the density of x, and $p_\sigma := p * \mathcal{N}(0, \sigma^2 I)$ the density of $\tilde{x} = x + \varepsilon$. For squared loss, the optimal denoiser satisfies

$$r^\star_\sigma(\tilde{x}) - \tilde{x} = \sigma^2 \nabla \log p_\sigma(\tilde{x}).$$

In particular, if p is smooth and σ is small,

$$r^\star_\sigma(x) \approx x + \sigma^2 \nabla \log p(x).$$

DAE & score-like vector field

Theorem ((Alain and Bengio 2014), informal)

Let p be the density of x, and $p_\sigma := p * \mathcal{N}(0, \sigma^2 I)$ the density of $\tilde{x} = x + \varepsilon$. For squared loss, the optimal denoiser satisfies

$$r_\sigma^\star(\tilde{x}) - \tilde{x} = \sigma^2 \nabla \log p_\sigma(\tilde{x}).$$

In particular, if p is smooth and σ is small,

$$r_\sigma^\star(x) \approx x + \sigma^2 \nabla \log p(x).$$

Idea of proof. Since $r_\sigma^\star(\tilde{x}) = \mathbb{E}[x \mid \tilde{x}]$, it is enough to relate the posterior mean shift to $\nabla \log p_\sigma$. Write

$$p_\sigma(\tilde{x}) = \int p(x)\, \mathcal{N}(\tilde{x}; x, \sigma^2 I)\, dx, \qquad \nabla_{\tilde{x}} \mathcal{N}(\tilde{x}; x, \sigma^2 I) = -\frac{\tilde{x} - x}{\sigma^2} \mathcal{N}(\tilde{x}; x, \sigma^2 I).$$

Differentiate under the integral:

$$\nabla \log p_\sigma(\tilde{x}) = \frac{\nabla p_\sigma(\tilde{x})}{p_\sigma(\tilde{x})} = -\frac{1}{\sigma^2} \frac{\int (\tilde{x} - x)\, p(x) \mathcal{N}(\tilde{x}; x, \sigma^2 I)\, dx}{\int p(x) \mathcal{N}(\tilde{x}; x, \sigma^2 I)\, dx} = \frac{1}{\sigma^2}\, \mathbb{E}[x - \tilde{x} \mid \tilde{x}].$$

Contractive regularization: a geometric reading

A contractive AE penalizes sensitivity of the code to input perturbations explicitly.

Let $z(x) := \mathrm{cod}_\theta(x) \in \mathbb{R}^d$, and write $J_z(x) := \nabla_x z(x) \in \mathbb{R}^{d\times p}$.

A common objective is

$$\min_{\theta,\phi} \ \mathbb{E}\big[\|x - r_{\theta,\phi}(x)\|^2\big] + \lambda\, \mathbb{E}\big[\|J_z(x)\|_{\mathrm{F}}^2\big].$$

Contractive regularization: a geometric reading

A contractive AE penalizes sensitivity of the code to input perturbations explicitly.

Let $z(x) := \mathrm{cod}_\theta(x) \in \mathbb{R}^d$, and write $J_z(x) := \nabla_x z(x) \in \mathbb{R}^{d\times p}$.

A common objective is

$$\min_{\theta,\phi} \ \mathbb{E}\big[\|x - r_{\theta,\phi}(x)\|^2\big] + \lambda\,\mathbb{E}\big[\|J_z(x)\|_{\mathrm{F}}^2\big].$$

Qualitative effect. If x lies near a d-dimensional manifold $M \subset \mathbb{R}^p$, the penalty $\|J_z(x)\|_{\mathrm{F}}^2$ encourages z to be (locally) insensitive to directions orthogonal to M: small changes in x that do not stay on M should not change the representation.

$\hookrightarrow$ Connecting with manifold learning.

Supervised regularization: multi-objective learning

Given labels y (classification or regression), add a supervised head $h_\psi : \mathbb{R}^d \to \mathcal{Y}$ and optimize

$$\min_{\theta,\phi,\psi} \mathbb{E}\big[\ell_{\text{rec}}(x, r(x))\big] + \lambda\,\mathbb{E}\big[\ell_{\text{sup}}(h_\psi(z(x)), y)\big].$$

Supervised regularization: multi-objective learning

Given labels y (classification or regression), add a supervised head $h_\psi : \mathbb{R}^d \to \mathcal{Y}$ and optimize

$$\min_{\theta,\phi,\psi} \mathbb{E}\big[\ell_{\mathrm{rec}}(x, r(x))\big] + \lambda\,\mathbb{E}\big[\ell_{\mathrm{sup}}(h_\psi(z(x)), y)\big].$$

Limiting regimes.

- $\lambda \to 0$: code z is optimized mainly for reconstruction (unsupervised geometry).
- $\lambda \to \infty$: code z is optimized mainly for prediction (may discard nuisance information).

Supervised regularization: multi-objective learning

Given labels y (classification or regression), add a supervised head $h_\psi : \mathbb{R}^d \to \mathcal{Y}$ and optimize

$$\min_{\theta,\phi,\psi} \mathbb{E}\big[\ell_{\text{rec}}(x, r(x))\big] + \lambda\, \mathbb{E}\big[\ell_{\text{sup}}(h_\psi(z(x)), y)\big].$$

Limiting regimes.

- $\lambda \to 0$: code z is optimized mainly for reconstruction (unsupervised geometry).
- $\lambda \to \infty$: code z is optimized mainly for prediction (may discard nuisance information).

$\hookrightarrow$ For representation learning, λ is chosen by validating downstream performance of z.

Practice: diagnostics and failure modes

Key hyperparameters:

- bottleneck d,
- architecture depth/width,
- noise level σ (DAE),
- Jacobian weight λ (CAE),
- supervised weight λ.

Diagnostics:

- Reconstruction: $\frac{1}{n}\sum_i \ell(x_i, \hat{x}_i)$ and its generalization gap.
- Latent covariance: $\widehat{\Sigma}_z := \frac{1}{n}\sum_i (z_i - \bar{z})(z_i - \bar{z})^\top$ (collapse $\Leftrightarrow$ low rank).
- Local stability: $\|J_z(x_i)\|_{\mathrm{F}}$ (contractive effect), or $\|r(x_i + \delta) - r(x_i)\|$ for small δ.

Typical failure modes:

- overcomplete memorization (too large d / too expressive decoder),
- representation collapse (too strong regularization),
- label leakage (too large supervised weight).

Variational auto-encoders

From reconstructing points to modeling distributions

A classical auto-encoder learns a deterministic reconstruction map

$$x \in \mathbb{R}^p \xrightarrow{\mathrm{cod}_\phi} z \in \mathbb{R}^d \xrightarrow{\mathrm{dec}_\theta} \hat{x} \in \mathbb{R}^p, \qquad \hat{x} := r_{\theta,\phi}(x) = \mathrm{dec}_\theta(\mathrm{cod}_\phi(x)),$$

typically by minimizing a reconstruction loss $\mathbb{E}[\ell(x, \hat{x})]$.

Question. Beyond reconstruction, can we *model the population* that generated the data, and *sample new observations*?

From reconstructing points to modeling distributions

A classical auto-encoder learns a deterministic reconstruction map

$$x \in \mathbb{R}^p \xrightarrow{\mathrm{cod}_\phi} z \in \mathbb{R}^d \xrightarrow{\mathrm{dec}_\theta} \hat{x} \in \mathbb{R}^p, \qquad \hat{x} := r_{\theta,\phi}(x) = \mathrm{dec}_\theta(\mathrm{cod}_\phi(x)),$$

typically by minimizing a reconstruction loss $\mathbb{E}[\ell(x, \hat{x})]$.

Question. Beyond reconstruction, can we *model the population* that generated the data, and *sample new observations*?

$\hookrightarrow$ A Variational Auto-Encoder answers this by making the decoder a *likelihood model*.

Jointly introduced by (Kingma and Welling 2014; Rezende, Mohamed, and Wierstra 2014)

VAE: latent variable model and decoder as likelihood

Assume a latent variable model

$$Z \sim p(z) \quad \text{(prior on latent space)}, \qquad X \mid (Z = z) \sim p_\theta(x \mid z) \quad \text{(decoder / likelihood)}.$$

The joint density is

$$p_\theta(x, z) = p_\theta(x \mid z)\, p(z), \qquad \text{hence} \qquad p_\theta(x) = \int_{\mathbb{R}^d} p_\theta(x \mid z)\, p(z)\, \mathrm{d}z.$$

VAE: latent variable model and decoder as likelihood

Assume a latent variable model

$$Z \sim p(z) \quad \text{(prior on latent space)}, \qquad X \mid (Z = z) \sim p_\theta(x \mid z) \quad \text{(decoder / likelihood)}.$$

The joint density is

$$p_\theta(x, z) = p_\theta(x \mid z)\, p(z), \qquad \text{hence} \qquad p_\theta(x) = \int_{\mathbb{R}^d} p_\theta(x \mid z)\, p(z)\, \mathrm{d}z.$$

Typical choices.

- $p(z) = \mathcal{N}(0, I_d)$,
- $p_\theta(x \mid z) = \mathcal{N}(\mu_\theta(z), \sigma_x^2 I_p)$ (Gaussian decoder),
- or $p_\theta(x \mid z) = \prod_{j=1}^{p} \mathrm{Bernoulli}(\pi_{\theta,j}(z))$ (binary data).

VAE: latent variable model and decoder as likelihood

Assume a latent variable model

$$Z \sim p(z) \quad \text{(prior on latent space)}, \qquad X \mid (Z = z) \sim p_\theta(x \mid z) \quad \text{(decoder / likelihood)}.$$

The joint density is

$$p_\theta(x, z) = p_\theta(x \mid z)\, p(z), \qquad \text{hence} \qquad p_\theta(x) = \int_{\mathbb{R}^d} p_\theta(x \mid z)\, p(z)\, \mathrm{d}z.$$

Typical choices.

- $p(z) = \mathcal{N}(0, I_d)$,
- $p_\theta(x \mid z) = \mathcal{N}(\mu_\theta(z), \sigma_x^2 I_p)$ (Gaussian decoder),
- or $p_\theta(x \mid z) = \prod_{j=1}^{p} \mathrm{Bernoulli}(\pi_{\theta,j}(z))$ (binary data).

Objective. Fit θ by maximum likelihood: $\max_\theta \sum_{i=1}^{n} \log p_\theta(x_i)$,
Problem. $\log p_\theta(x)$ involves an intractable integral in general.

Variational inference: Jensen lower bound

Introduce an auxiliary density $q(z \mid x)\mathrm{d}z$. Then

$$\begin{aligned}
\log p_\theta(x) &= \log \int p_\theta(x, z)\, \mathrm{d}z \\
&= \log \int q(z \mid x)\, \frac{p_\theta(x, z)}{q(z \mid x)}\, \mathrm{d}z \\
&= \log \mathbb{E}_{z \sim q(\cdot \mid x)} \left[\frac{p_\theta(x, z)}{q(z \mid x)} \right] \\
&\geq \mathbb{E}_{z \sim q(\cdot \mid x)} \left[\log \left(\frac{p_\theta(x, z)}{q(z \mid x)} \right) \right] \\
&= \mathbb{E}_{z \sim q(\cdot \mid x)} [\log p_\theta(x, z) - \log q(z \mid x)]\,,
\end{aligned}$$

by Jensen's inequality.

Definition (Evidence Lower BOund – ELBO (Kingma and Welling 2014))

For any $q(z \mid x)\mathrm{d}z$, define

$$\mathrm{ELBO}_\theta(q; x) := \mathbb{E}_{z \sim q(\cdot \mid x)}\left[\log p_\theta(x, z) - \log q(z \mid x)\right].$$

We have $\mathrm{ELBO}_\theta(q; x) \leq \log p_\theta(x)$. (Evidence is just Bayesian name for the marginal likelihood $p_\theta(x)$)

Definition (Evidence Lower BOund – ELBO (Kingma and Welling 2014))

For any $q(z \mid x)\mathrm{d}z$, define

$$\mathrm{ELBO}_\theta(q; x) := \mathbb{E}_{z \sim q(\cdot \mid x)}[\log p_\theta(x, z) - \log q(z \mid x)] .$$

We have $\mathrm{ELBO}_\theta(q; x) \le \log p_\theta(x)$. (Evidence is just Bayesian name for the marginal likelihood $p_\theta(x)$)

Object	Name
$p(z)$	Prior
$p_\theta(x \mid z)$	Likelihood / Decoder distribution
$p_\theta(x, z)$	Joint model
$p_\theta(x)$	Marginal likelihood / Evidence
$p_\theta(z \mid x)$	True posterior
$q(z \mid x)$	Variational posterior / Approximate posterior
$q_\phi(z \mid x)$	Encoder distribution
$\mathrm{ELBO}_\theta(q; x)$	Evidence Lower Bound (ELBO)

ELBO decomposition and the two terms

Using $p_\theta(x, z) = p_\theta(x \mid z)p(z)$,

$$\mathrm{ELBO}_\theta(q; x) = \mathbb{E}_{q(z|x)}[\log p_\theta(x \mid z)] - \mathrm{KL}\big(q(z \mid x) \,\|\, p(z)\big).$$

ELBO decomposition and the two terms

Using $p_\theta(x,z) = p_\theta(x \mid z)p(z)$,

$$\mathrm{ELBO}_\theta(q;x) = \mathbb{E}_{q(z|x)}[\log p_\theta(x \mid z)] - \mathrm{KL}\big(q(z \mid x) \,\|\, p(z)\big).$$

Moreover, the gap to the log-likelihood is an explicit KL divergence:

$$\log p_\theta(x) - \mathrm{ELBO}_\theta(q;x) = \mathrm{KL}\big(q(z \mid x) \,\|\, p_\theta(z \mid x)\big) \;\geq\; 0,$$

so maximizing the ELBO pushes $q(z \mid x)$ towards the true posterior $p_\theta(z \mid x)$.

ELBO decomposition and the two terms

Using $p_\theta(x,z) = p_\theta(x \mid z)p(z)$,

$$\mathrm{ELBO}_\theta(q;x) = \mathbb{E}_{q(z|x)}[\log p_\theta(x \mid z)] - \mathrm{KL}\big(q(z \mid x) \,\|\, p(z)\big).$$

Moreover, the gap to the log-likelihood is an explicit KL divergence:

$$\log p_\theta(x) - \mathrm{ELBO}_\theta(q;x) = \mathrm{KL}\big(q(z \mid x) \,\|\, p_\theta(z \mid x)\big) \;\geq\; 0,$$

so maximizing the ELBO pushes $q(z \mid x)$ towards the true posterior $p_\theta(z \mid x)$.

Interpretation.

- $\mathbb{E}_q[\log p_\theta(x \mid z)]$ = reconstruction / data-fit term,
- $\mathrm{KL}(q(z \mid x) \| p(z))$ = regularization enforcing a structured latent space (matching the prior).

VAE training objective (amortized variational family)

In practice, choose a parametric family $q_\phi(z \mid x)$ (the encoder), and optimize

$$\max_{\theta,\phi} \frac{1}{n}\sum_{i=1}^{n} \mathrm{ELBO}_\theta(\phi; x_i), \qquad \mathrm{ELBO}_\theta(\phi; x) = \mathbb{E}_{z\sim q_\phi(\cdot\mid x)}[\log p_\theta(x \mid z)] - \mathrm{KL}\big(q_\phi(z \mid x) \,\|\, p(z)\big).$$

VAE training objective (amortized variational family)

In practice, choose a parametric family $q_\phi(z \mid x)$ (the encoder), and optimize

$$\max_{\theta,\phi} \frac{1}{n}\sum_{i=1}^{n} \mathrm{ELBO}_\theta(\phi; x_i), \qquad \mathrm{ELBO}_\theta(\phi; x) = \mathbb{E}_{z\sim q_\phi(\cdot\mid x)}[\log p_\theta(x \mid z)] - \mathrm{KL}\big(q_\phi(z \mid x) \,\|\, p(z)\big).$$

Gaussian encoder (standard choice).

$$q_\phi(z \mid x) = \mathcal{N}\big(\mu_\phi(x), \mathrm{diag}(\sigma_\phi(x)^2)\big).$$

The KL term to $p(z) = \mathcal{N}(0, I_d)$ is available in closed-form:

$$\mathrm{KL}\big(q_\phi(z \mid x) \,\|\, \mathcal{N}(0, I)\big) = \frac{1}{2}\sum_{k=1}^{d} \big(\mu_{\phi,k}(x)^2 + \sigma_{\phi,k}(x)^2 - \log \sigma_{\phi,k}(x)^2 - 1\big)\,.$$

Reparameterization trick (enabling backprop through sampling)

The reconstruction term involves sampling $z \sim q_\phi(z \mid x)$, which naively breaks backprop.

Reparameterization trick (enabling backprop through sampling)

The reconstruction term involves sampling $z \sim q_\phi(z \mid x)$, which naively breaks backprop.

For the Gaussian encoder,

$$z \sim \mathcal{N}(\mu_\phi(x), \mathrm{diag}(\sigma_\phi(x)^2)) \quad \Longleftrightarrow \quad z = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I_d),$$

where $\odot$ is the coordinatewise product.

Reparameterization trick (enabling backprop through sampling)

The reconstruction term involves sampling $z \sim q_\phi(z \mid x)$, which naively breaks backprop.

For the Gaussian encoder,

$$z \sim \mathcal{N}(\mu_\phi(x), \operatorname{diag}(\sigma_\phi(x)^2)) \quad \Longleftrightarrow \quad z = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I_d),$$

where $\odot$ is the coordinatewise product.

Then, for a Monte-Carlo draw ε,

$$\mathbb{E}_{z \sim q_\phi(\cdot \mid x)}[\log p_\theta(x \mid z)] = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,I)} \left[\log p_\theta(x \mid \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon) \right],$$

which is differentiable w.r.t. (θ, ϕ) and amenable to SGD.

How VAEs relate to classical AEs

Assume a Gaussian decoder

$$p_\theta(x \mid z) = \mathcal{N}(\mu_\theta(z), \sigma_x^2 I_p).$$

Then (up to an additive constant)

$$-\log p_\theta(x \mid z) = \frac{1}{2\sigma_x^2}\|x - \mu_\theta(z)\|^2.$$

How VAEs relate to classical AEs

Assume a Gaussian decoder

$$p_\theta(x \mid z) = \mathcal{N}(\mu_\theta(z), \sigma_x^2 I_p).$$

Then (up to an additive constant)

$$-\log p_\theta(x \mid z) = \frac{1}{2\sigma_x^2}\|x - \mu_\theta(z)\|^2.$$

Thus, maximizing the ELBO is equivalent to minimizing (up to constants)

$$\underbrace{\mathbb{E}_{z\sim q_\phi(\cdot|x)}\left[\|x - \mu_\theta(z)\|^2\right]}_{\text{reconstruction}} + \underbrace{2\sigma_x^2\,\mathrm{KL}\big(q_\phi(z \mid x)\,\|\,p(z)\big)}_{\text{latent regularization}}.$$

How VAEs relate to classical AEs

Assume a Gaussian decoder

$$p_\theta(x \mid z) = \mathcal{N}(\mu_\theta(z), \sigma_x^2 I_p).$$

Then (up to an additive constant)

$$-\log p_\theta(x \mid z) = \frac{1}{2\sigma_x^2}\|x - \mu_\theta(z)\|^2.$$

Thus, maximizing the ELBO is equivalent to minimizing (up to constants)

$$\underbrace{\mathbb{E}_{z\sim q_\phi(\cdot|x)}\left[\|x - \mu_\theta(z)\|^2\right]}_{\text{reconstruction}} + \underbrace{2\sigma_x^2\,\mathrm{KL}\big(q_\phi(z \mid x)\,\|\,p(z)\big)}_{\text{latent regularization}}.$$

Heuristic. If $q_\phi(z \mid x)$ becomes highly concentrated, the first term resembles a classical AE loss, but the KL forces the encoder's output distribution to stay close to the prior.

Generation: what a VAE gives that an AE does not

Once trained, a VAE defines a full generative model:

$$z \sim p(z), \qquad x \sim p_\theta(x \mid z).$$

Generation: what a VAE gives that an AE does not

Once trained, a VAE defines a full generative model:

$$z \sim p(z), \qquad x \sim p_\theta(x \mid z).$$

In the common Gaussian decoder, one often visualizes

$$x \approx \mu_\theta(z) \quad \text{with} \quad z \sim \mathcal{N}(0, I_d),$$

and latent interpolations are meaningful because the training enforces $q_\phi(z \mid x)$ to match the prior distribution.

Generation: what a VAE gives that an AE does not

Once trained, a VAE defines a full generative model:

$$z \sim p(z), \qquad x \sim p_\theta(x \mid z).$$

In the common Gaussian decoder, one often visualizes

$$x \approx \mu_\theta(z) \quad \text{with} \quad z \sim \mathcal{N}(0, I_d),$$

and latent interpolations are meaningful because the training enforces $q_\phi(z \mid x)$ to match the prior distribution.

Take-home.
$\hookrightarrow$ A classical AE learns *a reconstruction map*.
$\hookrightarrow$ A VAE learns *a probability model* of the population, enabling principled sampling.

VAE vs DAE: two drastically different ways to use noise

VAE noise (latent). Randomness is introduced in $z \sim q_\phi(z \mid x)$ to optimize a variational bound, and the KL term structures the latent space.

$\hookrightarrow$ *models* $p_\theta(x)$ via latent-variable likelihoods and ELBOs.

VAE vs DAE: two drastically different ways to use noise

VAE noise (latent). Randomness is introduced in $z \sim q_\phi(z \mid x)$ to optimize a variational bound, and the KL term structures the latent space.

$\hookrightarrow$ *models* $p_\theta(x)$ via latent-variable likelihoods and ELBOs.

DAE noise (input). Randomness is introduced at the input $\tilde{x} = x + \varepsilon$, and the network is trained to *denoise*:

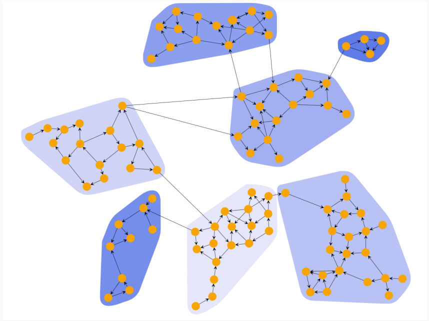
$$\min_r \ \mathbb{E}\left[\|x - r(\tilde{x})\|^2\right].$$

$\hookrightarrow$ *learns* a reconstruction/denoising operator.

Dimension reduction & clustering

Graph clustering

Given an unlabeled undirected weighted graph $\mathcal{G} = (V, W)$ with $|V| = n$ nodes and weight matrix $W \in \mathbb{R}^{n \times n}$, we would like to *decompose* it “as best as possible” into K clusters.



Here, W is a similarity matrix, i.e. $w_{i,j}$ is $\begin{cases} \text{large} & \text{if } i \text{ and } j \text{ are close / well-connected,} \\ \text{small} & \text{otherwise.} \end{cases}$

$\hookrightarrow$ For instance, one may take a binary adjacency matrix $w_{i,j} = \mathbf{1}_{i \sim j}$.

Graph cut

Idea: find clusters that are internally dense and separated by few cross-edges.

Given a candidate partition $V_1, \ldots, V_K$ of V, its cut value is

$$\mathrm{Cut}_W(V^{(1)}, \ldots, V^{(K)}) := \frac{1}{2} \sum_{k=1}^{K} \mathrm{Cut}_W(V^{(k)}), \quad \text{where } \mathrm{Cut}_W(V^{(k)}) := \sum_{\substack{i \in V^{(k)} \\ j \in (V^{(k)})^c}} w_{i,j}.$$

The graph cut counts each cross-edge once.

Graph cut

Idea: find clusters that are internally dense and separated by few cross-edges.

Given a candidate partition $V_1, \ldots, V_K$ of V, its cut value is

$$\mathrm{Cut}_W(V^{(1)}, \ldots, V^{(K)}) := \frac{1}{2} \sum_{k=1}^{K} \mathrm{Cut}_W(V^{(k)}), \quad \text{where } \mathrm{Cut}_W(V^{(k)}) := \sum_{\substack{i \in V^{(k)} \\ j \in (V^{(k)})^c}} w_{i,j}.$$

The graph cut counts each cross-edge once.

Unfortunately, a partition minimizing $\mathrm{Cut}_W(V^{(1)}, \ldots, V^{(K)})$ typically is very unbalanced:

$\hookrightarrow$ $(K-1)$ singletons $V^{(k)} = \{v_{i_k}\}$ of lowest degree $d_{i_k} := \sum_{j \in V} w_{i_k,j}$,

$\hookrightarrow$ $V^{(K)}$ being the remaining $n-K$ nodes.

Normalized graph cut

To *penalize* uneven clusters, consider the normalized cut

$$\mathrm{NCut}_W(V^{(1)}, \dots, V^{(K)}) := \frac{1}{2} \sum_{k=1}^{K} \frac{\mathrm{Cut}_W(V^{(k)})}{\mathrm{Vol}(V^{(k)})}, \quad \text{where } \mathrm{Vol}(V^{(k)}) := \sum_{i \in V^{(k)}} d_i.$$

This loss leads to the minimization problem

$$\operatorname*{arg\,min}_{(V^{(1)}, \dots, V^{(K)})} \mathrm{NCut}_W(V^{(1)}, \dots, V^{(K)}).$$

This problem called `NCUT` is unfortunately NP-complete (Shi and Malik 2000).

However, we can try and find a computationally simpler convex relaxation of it.

Convexifying the normalized graph cut

To convexify the problem, we shall replace each partition element $V^{(k)}$ by its (normalized) indicator function.

Namely, define the normalized cluster indicator $v^{(k)} \in \mathbb{R}^n$ and degree matrix $D \in \mathbb{R}^{n\times n}$ by

$$v_i^{(k)} := \begin{cases} 1/\sqrt{\mathrm{Vol}(V^{(k)})} & \text{if } i \in V^{(k)}, \\ 0 & \text{otherwise.} \end{cases}, \quad \text{and} \quad D := \mathrm{diag}(d_1, \ldots, d_n) \in \mathbb{R}^{n\times n}$$

Convexifying the normalized graph cut

To convexify the problem, we shall replace each partition element $V^{(k)}$ by its (normalized) indicator function.

Namely, define the normalized cluster indicator $v^{(k)} \in \mathbb{R}^n$ and degree matrix $D \in \mathbb{R}^{n\times n}$ by

$$v_i^{(k)} := \begin{cases} 1/\sqrt{\mathrm{Vol}(V^{(k)})} & \text{if } i \in V^{(k)}, \\ 0 & \text{otherwise.} \end{cases}, \quad \text{and} \quad D := \mathrm{diag}(d_1, \dots, d_n) \in \mathbb{R}^{n\times n}$$

– In $\mathbb{R}^n$, for all $k \neq k' \leq K$, we have:

  – $\langle v^{(k)}, Dv^{(k)} \rangle = 1$,
  – $\langle v^{(k)}, Dv^{(k')} \rangle = 0$, since $V^{(k)} \cap V^{(k')} = \emptyset$.

– The normalized cut can be rewritten as

$$\mathrm{NCut}_W(v^{(1)}, \dots, v^{(K)}) = \frac{1}{2} \sum_{k=1}^{K} \sum_{\substack{i \in V^{(k)} \\ j \in (V^{(k)})^c}} \frac{w_{i,j}}{\mathrm{Vol}(V^{(k)})} = \frac{1}{2} \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{i,j} (v_i^{(k)} - v_j^{(k)})^2.$$

Graph Laplacian

On the other hand, since $\mathcal{G}$ is undirected, $w_{i,j} = w_{j,i}$.

Therefore, for all $v \in \mathbb{R}^n$,

$$\begin{aligned}
\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j}(v_i - v_j)^2 &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j}(v_i^2 + v_j^2 - 2v_iv_j) \\
&= \sum_{i=1}^{n} d_i v_i^2 - \sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j} v_i v_j \\
&= v^\top (D - W)v.
\end{aligned}$$

Graph Laplacian

On the other hand, since $\mathcal{G}$ is undirected, $w_{i,j} = w_{j,i}$.

Therefore, for all $v \in \mathbb{R}^n$,

$$\begin{aligned}
\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j}(v_i - v_j)^2 &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j}(v_i^2 + v_j^2 - 2v_iv_j) \\
&= \sum_{i=1}^{n} d_i v_i^2 - \sum_{i=1}^{n}\sum_{j=1}^{n} w_{i,j}v_iv_j \\
&= v^\top (D - W)v.
\end{aligned}$$

The matrix $L := D - W$ is called the Laplacian matrix of the graph $\mathcal{G} = (V, W)$.

Spectral clustering

Coming back to the initial problem, we have relaxed the problem to

$$\underset{\substack{v^{(1)},\dots,v^{(K)}\\ \langle v^{(k)},Dv^{(k')}\rangle=\mathbf{1}_{k=k'}}}{\arg\min} \sum_{k=1}^{K}(v^{(k)})^\top L v^{(k)} = \underset{\substack{Y\in\mathbb{R}^{n\times K}\\ Y^\top DY=I_K}}{\arg\min} \operatorname{Tr}(Y^\top LY).$$

This is exactly the K smallest generalized eigenstructure $(y^{(1)}|\cdots|y^{(K)}) \in \mathbb{R}^{n\times K}$ of (L, D).

Spectral clustering

Coming back to the initial problem, we have relaxed the problem to

$$\underset{\substack{v^{(1)},\dots,v^{(K)}\\ \langle v^{(k)}, Dv^{(k')}\rangle = \mathbf{1}_{k=k'}}}{\arg\min} \sum_{k=1}^{K} (v^{(k)})^\top L v^{(k)} = \underset{\substack{Y\in\mathbb{R}^{n\times K}\\ Y^\top DY = I_K}}{\arg\min} \operatorname{Tr}(Y^\top LY).$$

This is exactly the K smallest generalized eigenstructure $(y^{(1)}|\cdots|y^{(K)}) \in \mathbb{R}^{n\times K}$ of (L, D).

– This algorithm called Spectral Clustering is widely used in clustering (Giulini 2016; Von Luxburg 2007).

– Because we solve a relaxation, the output $Y \in \mathbb{R}^{n\times K}$ is no longer a matrix of *hard* indicators (rows are not one-hot vectors). It should rather be interpreted as a *soft assignment*.

To recover a *discrete* partition, one applies a rounding step, such as

- **Thresholding:** $\hat{k}(i) \in \arg\max_{1\le k\le K} y_i^{(k)}$, and set $i \in \widehat{V}^{(\hat{k}(i))}$.
- **Euclidean clustering:** Apply k-means on the rows $\{y_i\}_{i=1}^{n}$ (often after normalization)

Laplacian Eigenmaps VS Spectral Clustering

$$\underset{\substack{Y \in \mathbb{R}^{n \times K} \\ Y^\top D Y = I_K}}{\arg\min} \ \operatorname{Tr}(Y^\top L Y).$$

For localized kernel graphs $w_{i,j} = k(\|x_i - x_j\|)$, this is Laplacian Eigenmaps with $K = d + 1$.

Laplacian Eigenmaps VS Spectral Clustering

$$\underset{\substack{Y \in \mathbb{R}^{n\times K} \\ Y^\top D Y = I_K}}{\arg\min} \; \mathrm{Tr}(Y^\top L Y).$$

For localized kernel graphs $w_{i,j} = k(\|x_i - x_j\|)$, this is Laplacian Eigenmaps with $K = d + 1$.

The difference is the post-processing / interpretation:

- Spectral clustering does not drop the smallest eigenvalues, and you have to discretize the vectors (e.g. via k-means) to recover a partition.
- Laplacian eigenmaps uses eigenvectors as continuous coordinates.

Two regimes, one computation:

- If the kernel graph is *nearly disconnected* into K weakly linked regions, the first eigenvectors are *almost piecewise-constant* $\Rightarrow$ clustering.
- If the graph is *well connected*, they vary *smoothly* across vertices and capture only *global, low-frequency* structure rather than separating clusters. $\Rightarrow$ manifold coordinates.

Supervised dimension reduction

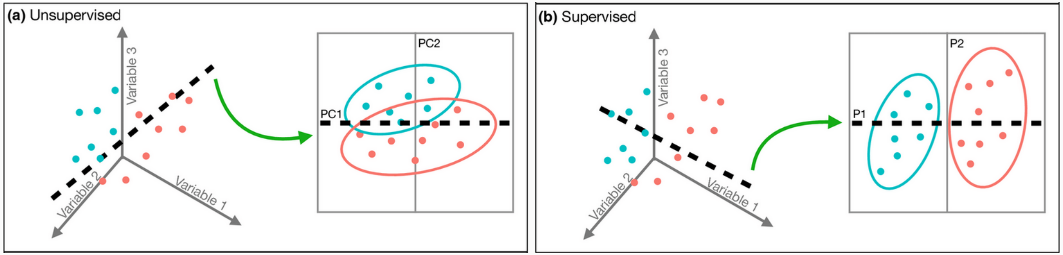
Discriminant Analysis

So far, we have addressed unsupervised data: all the coordinates of $x \in \mathbb{R}^p$ were considered in the same way.

Discriminant Analysis

So far, we have addressed unsupervised data: all the coordinates of $x \in \mathbb{R}^p$ were considered in the same way.

In supervised learning where groups of data points are known, one may wonder which variables contribute most to the geometric separation of some available labels.



Linear / Fisher discriminant analysis

Linear Discriminant Analysis, also called Fisher Discriminant Analysis, dates back to (Fisher 1936) on the `Iris` dataset.

Quoting Fisher, the method is meant to find the best

> *"linear functions of the measurements by which the populations are best discriminated"*

Geometric intuition: a good feature is one for which points in the same class are simultaneously near each other and far from points in the other classes.

Linear / Fisher discriminant analysis

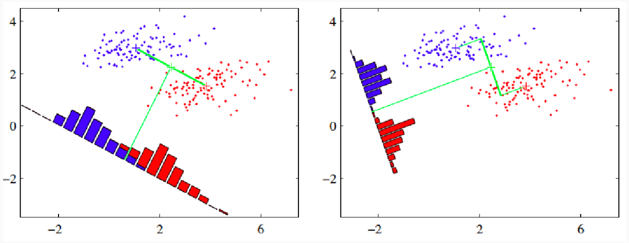
Linear Discriminant Analysis, also called Fisher Discriminant Analysis, dates back to (Fisher 1936) on the `Iris` dataset.

Quoting Fisher, the method is meant to find the best

> *"linear functions of the measurements by which the populations are best discriminated"*

Geometric intuition: a good feature is one for which points in the same class are simultaneously near each other and far from points in the other classes.

Idea: project while maximizing between-class variance *relative* to the within-class variance.



Total / Within-class / and Between-class variance

Write $(x, c) \in \mathbb{R}^p \times \{1, \ldots, J\}$ for the labeled data with J classes.

- The dataset has mean $\mu_T := \mathbb{E}_x[x]$ and total covariance

$$\mathbb{R}^{p\times p} \ni \Sigma_T := \mathbb{E}_x[(x-\mu_T)(x-\mu_T)^\top] = \frac{1}{n}\sum_{j=1}^{J}\sum_{i=1}^{n^{(j)}}(x_i^{(j)}-\mu_T)(x_i^{(j)}-\mu_T)^\top.$$

- Each class $j \in \{1, \ldots J\}$ has mean $\mu^{(j)} := \mathbb{E}_x[x \mid c = j]$ and within-class covariances

$$\mathbb{R}^{p\times p} \ni \Sigma_W^{(j)} := \mathbb{E}_x\left[(x-\mu^{(j)})(x-\mu^{(j)})^\top \mid c = j\right] = \frac{1}{n^{(j)}}\sum_{i=1}^{n^{(j)}}(x_i^{(j)}-\mu^{(j)})(x_i^{(j)}-\mu^{(j)})^\top.$$

- Across the classes, the between-class covariance is

$$\mathbb{R}^{p\times p} \ni \Sigma_B := \mathbb{E}_x\left[\left[(\mathbb{E}(x \mid c)-\mathbb{E}(x))(\mathbb{E}(x \mid c)-\mathbb{E}(x))^\top\right] = \frac{1}{n}\sum_{j=1}^{J} n^{(j)}(\mu^{(j)}-\mu_T)(\mu^{(j)}-\mu_T)^\top.$$

Linking the variances

Since

$$\mathbb{E}_x\left[(x - \mathbb{E}_x(x \mid c))(\mathbb{E}_x(x \mid c) - \mathbb{E}_x(x))^\top\right] = 0,$$

we get the decomposition

$$\begin{aligned}
\Sigma_T &= \mathbb{E}_x[(x - \mathbb{E}_x(x))(x - \mathbb{E}_x(x))^\top] \\
&= \mathbb{E}[(x - \mathbb{E}(x \mid c))(x - \mathbb{E}(x \mid c))^\top] + \mathbb{E}[(\mathbb{E}_x(x \mid c) - \mathbb{E}_x(x))(\mathbb{E}_x(x \mid c) - \mathbb{E}_x(x))^\top] \\
&= \Sigma_W + \Sigma_B,
\end{aligned}$$

where

$$\Sigma_W := \sum_{j=1}^{J} \frac{n^{(j)}}{n} \Sigma_W^{(j)} \qquad \text{and} \qquad \Sigma_B := \sum_{j=1}^{J} \frac{n^{(j)}}{n} (\mu^{(j)} - \mu_T)(\mu^{(j)} - \mu_T)^\top.$$

Linear Discriminant Analysis

Take a linear transformation (encoder) $A \in \mathbb{R}^{d \times p}$, and set $y_i := Ax_i$. Equivalently $Y := XA^\top$.

Linear Discriminant Analysis

Take a linear transformation (encoder) $A \in \mathbb{R}^{d \times p}$, and set $y_i := Ax_i$. Equivalently $Y := XA^\top$.

Formally, we would like to enhance visualization through class separation. At order two, this can translate to:

- maximize the inertia of between-class distance $\mathrm{Cov}_B(Y) = A\Sigma_B A^\top$;
- while keeping the within-class distances normalized $\mathrm{Cov}_W(Y) = A\Sigma_W A^\top$.

Linear Discriminant Analysis

Take a linear transformation (encoder) $A \in \mathbb{R}^{d\times p}$, and set $y_i := Ax_i$. Equivalently $Y := XA^\top$.

Formally, we would like to enhance visualization through class separation. At order two, this can translate to:

- maximize the inertia of between-class distance $\mathrm{Cov}_B(Y) = A\Sigma_B A^\top$;
- while keeping the within-class distances normalized $\mathrm{Cov}_W(Y) = A\Sigma_W A^\top$.

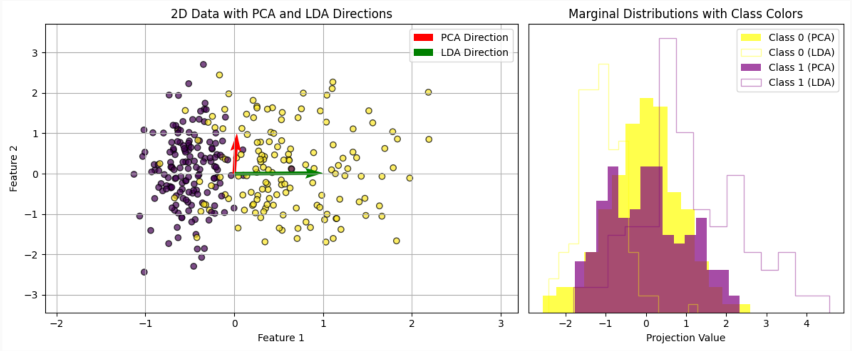
This leads to Linear Discriminant Analysis.

$$A_{\mathrm{LDA}} := \operatorname*{arg\,max}_{A\in\mathbb{R}^{d\times p}} \frac{\mathrm{Tr}(A\Sigma_B A^\top)}{\mathrm{Tr}(A\Sigma_W A^\top)},$$

with associated reduced point cloud $Y_{\mathrm{LDA}} := XA_{\mathrm{LDA}}^\top$.

PCA vs LDA

$$A_{\mathrm{PCA}} := \underset{A \in \mathbb{R}^{d \times p}}{\arg\max} \frac{\operatorname{Tr}(A \Sigma_T A^\top)}{\operatorname{Tr}(A A^\top)}, \qquad A_{\mathrm{LDA}} := \underset{A \in \mathbb{R}^{d \times p}}{\arg\max} \frac{\operatorname{Tr}(A \Sigma_B A^\top)}{\operatorname{Tr}(A \Sigma_W A^\top)}.$$



Compared to PCA, LDA increases the between-scatter and decreases the within-scatter.

Remarks

– Often presented with scatter matrices $S_\square = n\Sigma_\square$ instead of covariances.

Remarks

– Often presented with scatter matrices $S_\square = n\Sigma_\square$ instead of covariances.

– Since $\Sigma_T = \Sigma_B + \Sigma_W$,

$$A_{\mathrm{LDA}} = \underset{A \in \mathbb{R}^{d\times p}}{\arg\max} \frac{\mathrm{Tr}(A(\Sigma_T - \Sigma_W)A^\top)}{\mathrm{Tr}(A\Sigma_W A^\top)} = \underset{A \in \mathbb{R}^{d\times p}}{\arg\max} \frac{\mathrm{Tr}(A\Sigma_T A^\top)}{\mathrm{Tr}(A\Sigma_W A^\top)}.$$

Remarks

– Often presented with scatter matrices $S_\square = n\Sigma_\square$ instead of covariances.

– Since $\Sigma_T = \Sigma_B + \Sigma_W$,

$$A_{\mathrm{LDA}} = \underset{A \in \mathbb{R}^{d \times p}}{\arg\max} \frac{\mathrm{Tr}(A(\Sigma_T - \Sigma_W)A^\top)}{\mathrm{Tr}(A\Sigma_W A^\top)} = \underset{A \in \mathbb{R}^{d \times p}}{\arg\max} \frac{\mathrm{Tr}(A\Sigma_T A^\top)}{\mathrm{Tr}(A\Sigma_W A^\top)}.$$

– Many variants exist (Ghojogh et al. 2023).

– Can be seen as a classification algorithm with likelihood maximization (Kim, Magnani, and Boyd 2005).

– If Σ_W is (too close to being) singular, one may modify it and replace its smallest eigenvalues by a fixed user-defined value to robustify it (Deng et al. 2007; Guo and Wang 2015).

Fifty shades of linear discriminant component analysis

To account for unbalanced classes, one may also consider the same generalized eigenvalue problem with different scatter/covariance matrices.

Fifty shades of linear discriminant component analysis

To account for unbalanced classes, one may also consider the same generalized eigenvalue problem with different scatter/covariance matrices.

In Discriminative component analysis, (Hoi et al. 2006) take instead

$$\widetilde{\Sigma}'_B := \frac{1}{J(J-1)} \sum_{\ell=1}^{J} \sum_{j=1}^{J} (\mu^{(\ell)} - \mu^{(j)})(\mu^{(\ell)} - \mu^{(j)})^\top,$$

or

$$\widetilde{\Sigma}'_B := \frac{1}{J} \sum_{\ell=1}^{J} (\mu^{(\ell)} - \mu_T)(\mu^{(\ell)} - \mu_T)^\top ?$$

In Relevant Component analysis, (Shental et al. 2002) take instead

$$\widetilde{\Sigma}'_W = \frac{1}{nJ} \sum_{j=1}^{J} \sum_{i=1}^{n^{(j)}} (x_i^{(j)} - \mu^{(j)})(x_i^{(j)} - \mu^{(j)})^\top$$

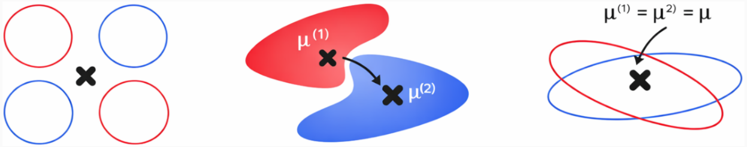
Limitations of LDA

Dimensionality bound. With J classes, LDA returns $J-1$ discriminant directions: If a classifier seems to require more than $J-1$ informative features, one must complement LDA with another representation.

Modeling assumption. LDA is somehow optimal under the simplistic generative picture

$$x \mid (Y=j) \;\approx\; \mathcal{N}(\mu^{(j)}, \Sigma) \quad \text{(one Gaussian per class, common covariance).}$$

When class-conditional distributions are far from Gaussian (multimodal, heavy-tailed, curved), the LDA projection can wash out this finer geometry.



LDA will also fail when the discriminatory information is not in the mean but rather in the variance of the data.

Featurized Component Analysis

Instead of minimizing a non-convex objective over a complicated hypothesis space, we may first apply a feature map

$$\Phi : \mathbb{R}^p \to \mathcal{H}$$

and then run LDA in feature space. Write $\widetilde{x}_i := \Phi(x_i) \in \mathcal{H}$ for the transformed variables.

Featurized Component Analysis

Instead of minimizing a non-convex objective over a complicated hypothesis space, we may first apply a feature map

$$\Phi : \mathbb{R}^p \to \mathcal{H}$$

and then run LDA in feature space. Write $\widetilde{x}_i := \Phi(x_i) \in \mathcal{H}$ for the transformed variables.

Denote by $\widetilde{\mu}^{(j)} := \mathbb{E}[\widetilde{x} \mid c = j], \widetilde{\mu}_T := \mathbb{E}[\widetilde{x}]$ the class and global means in $\mathcal{H}$.

Define the (feature-space) between/within covariances as linear operators $\mathcal{H} \to \mathcal{H}$:

$$\widetilde{\Sigma}_B := \sum_{j=1}^{J} \frac{n^{(j)}}{n} (\widetilde{\mu}^{(j)} - \widetilde{\mu}_T)^{\otimes 2}, \quad \widetilde{\Sigma}_W := \sum_{j=1}^{J} \frac{n^{(j)}}{n} \widetilde{\Sigma}_W^{(j)}, \quad \widetilde{\Sigma}_W^{(j)} := \mathbb{E}\left[(\widetilde{x} - \widetilde{\mu}^{(j)})^{\otimes 2} \mid c = j\right].$$

Featurized Component Analysis

Instead of minimizing a non-convex objective over a complicated hypothesis space, we may first apply a feature map

$$\Phi : \mathbb{R}^p \to \mathcal{H}$$

and then run LDA in feature space. Write $\widetilde{x}_i := \Phi(x_i) \in \mathcal{H}$ for the transformed variables.

Denote by $\widetilde{\mu}^{(j)} := \mathbb{E}[\widetilde{x} \mid c = j], \widetilde{\mu}_T := \mathbb{E}[\widetilde{x}]$ the class and global means in $\mathcal{H}$.

Define the (feature-space) between/within covariances as linear operators $\mathcal{H} \to \mathcal{H}$:

$$\widetilde{\Sigma}_B := \sum_{j=1}^{J} \frac{n^{(j)}}{n} (\widetilde{\mu}^{(j)} - \widetilde{\mu}_T)^{\otimes 2}, \quad \widetilde{\Sigma}_W := \sum_{j=1}^{J} \frac{n^{(j)}}{n} \widetilde{\Sigma}_W^{(j)}, \quad \widetilde{\Sigma}_W^{(j)} := \mathbb{E}\left[(\widetilde{x} - \widetilde{\mu}^{(j)})^{\otimes 2} \mid c = j\right].$$

Rayleigh quotient for feature-space LDA: for $a \in \mathcal{H} \setminus \{0\}$,

$$\mathcal{R}(a) := \frac{\langle a, \widetilde{\Sigma}_B a \rangle_{\mathcal{H}}}{\langle a, \widetilde{\Sigma}_W a \rangle_{\mathcal{H}}}.$$

Representer theorem: reduce the search to an n-dimensional span

We seek directions $a \in \mathcal{H}$ maximizing $\mathcal{R}(a)$. In finite sample, the relevant information lives in

$$\mathrm{span}\{\widetilde{x}_1, \ldots, \widetilde{x}_n\} \subset \mathcal{H}.$$

Representer theorem: reduce the search to an n-dimensional span

We seek directions $a \in \mathcal{H}$ maximizing $\mathcal{R}(a)$. In finite sample, the relevant information lives in

$$\operatorname{span}\{\widetilde{x}_1, \dots, \widetilde{x}_n\} \subset \mathcal{H}.$$

Representer principle (finite-sample). For Fisher-type criteria built from empirical means/covariances, an optimizer can be chosen in the form

$$a = \sum_{i=1}^{n} \alpha_i \, \widetilde{x}_i, \qquad \alpha \in \mathbb{R}^n.$$

Representer theorem: reduce the search to an n-dimensional span

We seek directions $a \in \mathcal{H}$ maximizing $\mathcal{R}(a)$. In finite sample, the relevant information lives in

$$\operatorname{span}\{\widetilde{x}_1, \dots, \widetilde{x}_n\} \subset \mathcal{H}.$$

Representer principle (finite-sample). For Fisher-type criteria built from empirical means/covariances, an optimizer can be chosen in the form

$$a = \sum_{i=1}^{n} \alpha_i \, \widetilde{x}_i, \qquad \alpha \in \mathbb{R}^n.$$

Then the feature-space coordinate of point x_j becomes

$$y_j = \langle a, \widetilde{x}_j \rangle_{\mathcal{H}} = \sum_{i=1}^{n} \alpha_i \langle \widetilde{x}_i, \widetilde{x}_j \rangle_{\mathcal{H}} = (\mathbb{K}\alpha)_j, \qquad \mathbb{K}_{ij} := K(x_i, x_j).$$

Centering in feature space via Gram matrices

Most works assume centered features. In $\mathcal{H}$, centering amounts to replacing $\widetilde{x}_i$ by

$$\widetilde{x}_i^c := \widetilde{x}_i - \widetilde{\mu}_T.$$

Centering in feature space via Gram matrices

Most works assume centered features. In $\mathcal{H}$, centering amounts to replacing $\widetilde{x}_i$ by

$$\widetilde{x}_i^c := \widetilde{x}_i - \widetilde{\mu}_T.$$

Define the centering matrix

$$H := I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top.$$

Then the centered Gram matrix is

$$G := H\mathbb{K}H, \qquad G_{ij} = \langle \widetilde{x}_i^c, \widetilde{x}_j^c \rangle_{\mathcal{H}}.$$

Centering in feature space via Gram matrices

Most works assume centered features. In $\mathcal{H}$, centering amounts to replacing $\widetilde{x}_i$ by

$$\widetilde{x}_i^c := \widetilde{x}_i - \widetilde{\mu}_T.$$

Define the centering matrix

$$H := I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top.$$

Then the centered Gram matrix is

$$G := H\mathbb{K}H, \qquad G_{ij} = \langle \widetilde{x}_i^c, \widetilde{x}_j^c \rangle_{\mathcal{H}}.$$

As for classical scaling, all inner products between centered linear combinations of $(\widetilde{x}_i)$ can be computed with G only.

Kernel LDA: class averaging operator and two Gram-only scatters

Let $c_i \in \{1, \dots, J\}$ be the label of x_i and set $n^{(j)} := |\{i : c_i = j\}|$.

Define the class-averaging matrix $M \in \mathbb{R}^{n \times n}$ by

$$M_{ij} := \begin{cases} \frac{1}{n^{(c_i)}} & \text{if } c_i = c_j, \\ 0 & \text{otherwise.} \end{cases} \qquad \Longrightarrow \qquad (Mv)_i = \frac{1}{n^{(c_i)}} \sum_{j : c_j = c_i} v_j .$$

Kernel LDA: class averaging operator and two Gram-only scatters

Let $c_i \in \{1, \dots, J\}$ be the label of x_i and set $n^{(j)} := |\{i : c_i = j\}|$.

Define the class-averaging matrix $M \in \mathbb{R}^{n \times n}$ by

$$M_{ij} := \begin{cases} \frac{1}{n^{(c_i)}} & \text{if } c_i = c_j, \\ 0 & \text{otherwise.} \end{cases} \qquad \Longrightarrow \qquad (Mv)_i = \frac{1}{n^{(c_i)}} \sum_{j : c_j = c_i} v_j .$$

For a coefficient vector $\alpha \in \mathbb{R}^n$ and $y := G\alpha$ (centered), one checks the identities

$$\text{(between)} \quad \alpha^\top G(M - \tfrac{1}{n}\mathbf{1}\mathbf{1}^\top)G\,\alpha = \sum_{j=1}^{J} \frac{n^{(j)}}{n} \left(\bar{y}^{(j)} - \bar{y}_T\right)^{\otimes 2},$$

$$\text{(within)} \quad \alpha^\top G(I - M)G\,\alpha = \frac{1}{n} \sum_{j=1}^{J} \sum_{i : c_i = j} \left(y_i - \bar{y}^{(j)}\right)^{\otimes 2},$$

where $\bar{y}^{(j)}$ is the average within class j and $\bar{y}_T$ the global average.

Kernel LDA as a Rayleigh quotient in $\mathbb{R}^n$

We end up with the Gram-only Rayleigh quotient defined for all $\alpha \in \mathbb{R}^n$ by

$$\mathcal{R}(\alpha) := \frac{\alpha^\top G(M - \frac{1}{n}\mathbf{1}\mathbf{1}^\top)G\,\alpha}{\alpha^\top G(I - M)G\,\alpha}.$$

Kernel LDA as a Rayleigh quotient in $\mathbb{R}^n$

We end up with the Gram-only Rayleigh quotient defined for all $\alpha \in \mathbb{R}^n$ by

$$\mathcal{R}(\alpha) := \frac{\alpha^\top G(M - \frac{1}{n}\mathbf{1}\mathbf{1}^\top)G\,\alpha}{\alpha^\top G(I - M)G\,\alpha}.$$

A standard ridge-regularized kernel LDA direction solves

$$\alpha^\star \in \arg\max_{\alpha \neq 0} \frac{\alpha^\top G(M - \frac{1}{n}\mathbf{1}\mathbf{1}^\top)G\,\alpha}{\alpha^\top \big(G(I - M)G + \varepsilon I\big)\,\alpha}, \qquad \varepsilon > 0.$$

Equivalently, $\alpha^\star$ is a top generalized eigenvector of

$$G(M - \tfrac{1}{n}\mathbf{1}\mathbf{1}^\top)G\,\alpha = \lambda\big(G(I - M)G + \varepsilon I\big)\alpha.$$

Kernel LDA as a Rayleigh quotient in $\mathbb{R}^n$

We end up with the Gram-only Rayleigh quotient defined for all $\alpha \in \mathbb{R}^n$ by

$$\mathcal{R}(\alpha) := \frac{\alpha^\top G(M - \frac{1}{n}\mathbf{1}\mathbf{1}^\top)G\,\alpha}{\alpha^\top G(I - M)G\,\alpha}.$$

A standard ridge-regularized kernel LDA direction solves

$$\alpha^\star \in \arg\max_{\alpha \neq 0} \frac{\alpha^\top G(M - \frac{1}{n}\mathbf{1}\mathbf{1}^\top)G\,\alpha}{\alpha^\top \big(G(I - M)G + \varepsilon I\big)\,\alpha}, \qquad \varepsilon > 0.$$

Equivalently, $\alpha^\star$ is a top generalized eigenvector of

$$G(M - \tfrac{1}{n}\mathbf{1}\mathbf{1}^\top)G\,\alpha = \lambda\big(G(I - M)G + \varepsilon I\big)\alpha.$$

The embedded coordinate is $y = G\alpha^\star$, and multiple components come from multiple orthogonal eigenvectors. (as now usual at this point of the class!)

Multi-class dimension bound: still at most $J-1$ directions

Even in feature space, the between-class operator has rank at most $J-1$:

$$\operatorname{rank}(\widetilde{\Sigma}_B) \leq J - 1.$$

Multi-class dimension bound: still at most $J-1$ directions

Even in feature space, the between-class operator has rank at most $J-1$:

$$\operatorname{rank}(\widetilde{\Sigma}_B) \leq J-1.$$

Indeed, $\widetilde{\Sigma}_B$ is a sum of J rank-one operators built from $\widetilde{\mu}^{(j)} - \widetilde{\mu}_T$, but these J vectors satisfy

$$\sum_{j=1}^{J} \frac{n^{(j)}}{n} (\widetilde{\mu}^{(j)} - \widetilde{\mu}_T) = 0,$$

hence they live in a $(J-1)$-dimensional affine subspace of $\mathcal{H}$.

Multi-class dimension bound: still at most $J-1$ directions

Even in feature space, the between-class operator has rank at most $J-1$:

$$\operatorname{rank}(\widetilde{\Sigma}_B) \le J-1.$$

Indeed, $\widetilde{\Sigma}_B$ is a sum of J rank-one operators built from $\widetilde{\mu}^{(j)} - \widetilde{\mu}_T$, but these J vectors satisfy

$$\sum_{j=1}^{J} \frac{n^{(j)}}{n} (\widetilde{\mu}^{(j)} - \widetilde{\mu}_T) = 0,$$

hence they live in a $(J-1)$-dimensional affine subspace of $\mathcal{H}$.

Conclusion: Kernel LDA can yield at most $J-1$ non-trivial discriminant components (same limitation as LDA).

Supervised PCA: from HSIC to a Rayleigh quotient

Supervised PCA (Barshan et al. 2011) selects directions that *maximally correlate* to labels.

The so-called Hilbert-Schmidt Independence Criterion is $\mathrm{HSIC} := \frac{1}{(n-1)^2}\mathrm{Tr}\big(K_X H K_C H\big)$, with K_X and K_C Gram matrices on samples and H the centering matrix.

The idea of HSIC is to measure the dependence of two random variables by calculating the correlation of their values mapped to the Hilbert space.

Supervised PCA: from HSIC to a Rayleigh quotient

Supervised PCA (Barshan et al. 2011) selects directions that *maximally correlate* to labels.

The so-called Hilbert-Schmidt Independence Criterion is $\mathrm{HSIC} := \frac{1}{(n-1)^2}\mathrm{Tr}\big(K_X H K_C H\big)$, with K_X and K_C Gram matrices on samples and H the centering matrix.

The idea of HSIC is to measure the dependence of two random variables by calculating the correlation of their values mapped to the Hilbert space.

If we restrict to *linear* projections $z = Xa$ (one component), take $K_X(a) := zz^\top = Xaa^\top X^\top$ and a fixed label-kernel K_C. Then

$$\mathrm{HSIC}(Xa, C) \propto \mathrm{Tr}\big((Xaa^\top X^\top)HK_CH\big) = a^\top X^\top HK_CHX\, a.$$

Supervised PCA: from HSIC to a Rayleigh quotient

Supervised PCA (Barshan et al. 2011) selects directions that *maximally correlate* to labels.

The so-called Hilbert-Schmidt Independence Criterion is $\mathrm{HSIC} := \frac{1}{(n-1)^2}\mathrm{Tr}\big(K_X H K_C H\big)$, with K_X and K_C Gram matrices on samples and H the centering matrix.

The idea of HSIC is to measure the dependence of two random variables by calculating the correlation of their values mapped to the Hilbert space.

If we restrict to *linear* projections $z = Xa$ (one component), take $K_X(a) := zz^\top = Xaa^\top X^\top$ and a fixed label-kernel K_C. Then

$$\mathrm{HSIC}(Xa, C) \propto \mathrm{Tr}\big((Xaa^\top X^\top)HK_CH\big) = a^\top X^\top HK_CHX\, a.$$

The Rayleigh quotient for Supervised PCA is defined for all $a \in \mathbb{R}^p$ by

$$\mathcal{R}_{\mathrm{SPCA}}(a) := \frac{a^\top\big(X^\top HK_CHX\big)a}{a^\top(X^\top X)\,a}.$$

Supervised PCA: generalized eigenvectors

For multiple components $A \in \mathbb{R}^{d \times p}$, the natural trace form is

$$\arg\max_{A} \frac{\operatorname{Tr}\big(A\,X^\top H K_C H X\,A^\top\big)}{\operatorname{Tr}\big(A\,X^\top X\,A^\top\big)}.$$

Supervised PCA: generalized eigenvectors

For multiple components $A \in \mathbb{R}^{d \times p}$, the natural trace form is

$$\arg\max_{A} \frac{\operatorname{Tr}\left(A\, X^\top H K_C H X\, A^\top\right)}{\operatorname{Tr}\left(A\, X^\top X\, A^\top\right)}.$$

Imposing the standard whitening constraint $A X^\top X A^\top = I_d$ gives

$$\arg\max_{A:\ A(X^\top X)A^\top = I_d} \operatorname{Tr}\left(A\, X^\top H K_C H X\, A^\top\right),$$

whose solutions are the top generalized eigenvectors of

$$X^\top H K_C H X\, a = \lambda\, (X^\top X)\, a.$$

Supervised PCA: generalized eigenvectors

For multiple components $A \in \mathbb{R}^{d\times p}$, the natural trace form is

$$\arg\max_{A} \frac{\operatorname{Tr}\big(A\,X^\top H K_C H X\,A^\top\big)}{\operatorname{Tr}\big(A\,X^\top X\,A^\top\big)}.$$

Imposing the standard whitening constraint $AX^\top X A^\top = I_d$ gives

$$\arg\max_{A:\ A(X^\top X)A^\top = I_d} \operatorname{Tr}\big(A\,X^\top H K_C H X\,A^\top\big),$$

whose solutions are the top generalized eigenvectors of

$$X^\top H K_C H X\,a = \lambda\,(X^\top X)\,a.$$

$\hookrightarrow$ Compare with PCA: $X^\top X a = \lambda a$.

Overall correlation: fix notation and connect to HSIC

Given centered random variables $U \in \mathbb{R}^u$ and $V \in \mathbb{R}^v$, define their cross-covariance

$$\Sigma_{UV} := \mathbb{E}[UV^\top] \in \mathbb{R}^{u \times v}.$$

(Here $UV^\top$ is $u \times v$; not $U^\top V$.)

Overall correlation: fix notation and connect to HSIC

Given centered random variables $U \in \mathbb{R}^u$ and $V \in \mathbb{R}^v$, define their cross-covariance

$$\Sigma_{UV} := \mathbb{E}[UV^\top] \in \mathbb{R}^{u\times v}.$$

(Here $UV^\top$ is $u \times v$; not $U^\top V$.)

The Hilbert–Schmidt norm (a.k.a. Frobenius norm) summarizes second-order dependence:

$$\|\Sigma_{UV}\|_{\mathrm{HS}}^2 := \mathrm{Tr}\big(\Sigma_{UV}^\top \Sigma_{UV}\big).$$

A normalized scalar is

$$\rho_{UV} := \frac{\|\Sigma_{UV}\|_{\mathrm{HS}}^2}{\|\Sigma_{UU}\|_{\mathrm{HS}}\ \|\Sigma_{VV}\|_{\mathrm{HS}}}.$$

Overall correlation: fix notation and connect to HSIC

Given centered random variables $U \in \mathbb{R}^u$ and $V \in \mathbb{R}^v$, define their cross-covariance

$$\Sigma_{UV} := \mathbb{E}[UV^\top] \in \mathbb{R}^{u\times v}.$$

(Here $UV^\top$ is $u \times v$; not $U^\top V$.)

The Hilbert–Schmidt norm (a.k.a. Frobenius norm) summarizes second-order dependence:

$$\|\Sigma_{UV}\|_{\mathrm{HS}}^2 := \mathrm{Tr}\big(\Sigma_{UV}^\top \Sigma_{UV}\big).$$

A normalized scalar is

$$\rho_{UV} := \frac{\|\Sigma_{UV}\|_{\mathrm{HS}}^2}{\|\Sigma_{UU}\|_{\mathrm{HS}}\ \|\Sigma_{VV}\|_{\mathrm{HS}}}.$$

HSIC is the nonlinear analogue: it replaces (U, V) by RKHS features and replaces covariances by RKHS covariance operators, whose Hilbert–Schmidt norm is computed from Gram matrices.

Robust Fisher Discriminant Analysis: worst-case Rayleigh quotient

(Kim, Magnani, and Boyd 2005) proposes a robust variant of Fisher LDA: instead of trusting empirical $(\mu^{(j)}, \Sigma_W)$, assume they live in an uncertainty set $\mathcal{U}$.

Robust Fisher Discriminant Analysis: worst-case Rayleigh quotient

(Kim, Magnani, and Boyd 2005) proposes a robust variant of Fisher LDA: instead of trusting empirical $(\mu^{(j)}, \Sigma_W)$, assume they live in an uncertainty set $\mathcal{U}$.

Robust Fisher ratio (one direction):

$$\max_{a \neq 0} \min_{(\mu,\Sigma) \in \mathcal{U}} \frac{a^\top \Sigma_B(\mu)\, a}{a^\top \Sigma_W(\Sigma)\, a}.$$

Robust Fisher Discriminant Analysis: worst-case Rayleigh quotient

(Kim, Magnani, and Boyd 2005) proposes a robust variant of Fisher LDA: instead of trusting empirical $(\mu^{(j)}, \Sigma_W)$, assume they live in an uncertainty set $\mathcal{U}$.

Robust Fisher ratio (one direction):

$$\max_{a \neq 0} \min_{(\mu,\Sigma)\in\mathcal{U}} \frac{a^\top \Sigma_B(\mu)\, a}{a^\top \Sigma_W(\Sigma)\, a}.$$

$\hookrightarrow$ Under convex uncertainty models, this becomes a tractable convex problem (often a regularized eigenproblem).

Quadratic Discriminant Analysis: generative model and decision rule

Quadratic Discriminant Analysis (QDA) uses a Gaussian model with class-dependent covariances:

$$x \mid (Y = j) \sim \mathcal{N}(\mu^{(j)}, \Sigma^{(j)}), \qquad j \in \{1, \dots, J\},$$

with class priors π_j.

Quadratic Discriminant Analysis: generative model and decision rule

Quadratic Discriminant Analysis (QDA) uses a Gaussian model with class-dependent covariances:

$$x \mid (Y = j) \sim \mathcal{N}(\mu^{(j)}, \Sigma^{(j)}), \qquad j \in \{1, \dots, J\},$$

with class priors π_j.

The Bayes rule selects the class maximizing the log-posterior score

$$\delta_j(x) := -\frac{1}{2}(x - \mu^{(j)})^\top (\Sigma^{(j)})^{-1}(x - \mu^{(j)}) - \frac{1}{2}\log\det\Sigma^{(j)} + \log\pi_j.$$

Quadratic Discriminant Analysis: generative model and decision rule

Quadratic Discriminant Analysis (QDA) uses a Gaussian model with class-dependent covariances:

$$x \mid (Y = j) \sim \mathcal{N}(\mu^{(j)}, \Sigma^{(j)}), \qquad j \in \{1, \dots, J\},$$

with class priors π_j.

The Bayes rule selects the class maximizing the log-posterior score

$$\delta_j(x) := -\frac{1}{2}(x - \mu^{(j)})^\top (\Sigma^{(j)})^{-1}(x - \mu^{(j)}) - \frac{1}{2}\log\det\Sigma^{(j)} + \log\pi_j.$$

If all $\Sigma^{(j)}$ are equal, $\delta_j(x)$ is affine in x and we recover LDA. If not, the boundary $\delta_j(x) = \delta_{j'}(x)$ is quadratic in x.

Supervised dimension reduction with RKHS: the SDR principle

(Fukumizu, Bach, and Jordan 2004) studies sufficient dimension reduction (SDR): find $Z = \psi(X) \in \mathbb{R}^d$ such that

$$Y \perp\!\!\!\perp X \,\big|\, Z.$$

Supervised dimension reduction with RKHS: the SDR principle

(Fukumizu, Bach, and Jordan 2004) studies sufficient dimension reduction (SDR): find $Z = \psi(X) \in \mathbb{R}^d$ such that

$$Y \perp\!\!\!\perp X \,\big|\, Z.$$

Interpretation: Z keeps exactly the information in X that is useful to predict Y.

Supervised dimension reduction with RKHS: the SDR principle

(Fukumizu, Bach, and Jordan 2004) studies sufficient dimension reduction (SDR): find $Z = \psi(X) \in \mathbb{R}^d$ such that

$$Y \perp\!\!\!\perp X \,\big|\, Z.$$

Interpretation: Z keeps exactly the information in X that is useful to predict Y.

A convenient proxy is to minimize a conditional dependence measure:

$$Z \text{ is good} \quad \Longrightarrow \quad \text{“}\Sigma_{YY|Z} \text{ is small”}$$

where $\Sigma_{YY|Z}$ is a conditional covariance operator in an RKHS on Y.

RKHS view: conditional covariance operator and a Gram estimator

Let $\mathcal{H}_Y$ be an RKHS on $\mathcal{Y}$ and $\Phi_Y(Y) \in \mathcal{H}_Y$. Define the covariance operators

$$\Sigma_{YY} := \mathbb{E}\left[(\Phi_Y(Y) - \mu_Y) \otimes (\Phi_Y(Y) - \mu_Y)\right], \qquad \Sigma_{YZ} := \mathbb{E}\left[(\Phi_Y(Y) - \mu_Y) \otimes (\Phi_Z(Z) - \mu_Z)\right],$$

and similarly Σ_{ZZ}.

RKHS view: conditional covariance operator and a Gram estimator

Let $\mathcal{H}_Y$ be an RKHS on $\mathcal{Y}$ and $\Phi_Y(Y) \in \mathcal{H}_Y$. Define the covariance operators

$$\Sigma_{YY} := \mathbb{E}\left[(\Phi_Y(Y) - \mu_Y) \otimes (\Phi_Y(Y) - \mu_Y)\right], \qquad \Sigma_{YZ} := \mathbb{E}\left[(\Phi_Y(Y) - \mu_Y) \otimes (\Phi_Z(Z) - \mu_Z)\right],$$

and similarly Σ_{ZZ}.

The conditional covariance operator is

$$\Sigma_{YY|Z} := \Sigma_{YY} - \Sigma_{YZ}\Sigma_{ZZ}^{-1}\Sigma_{ZY}.$$

RKHS view: conditional covariance operator and a Gram estimator

Let $\mathcal{H}_Y$ be an RKHS on $\mathcal{Y}$ and $\Phi_Y(Y) \in \mathcal{H}_Y$. Define the covariance operators

$$\Sigma_{YY} := \mathbb{E}\left[(\Phi_Y(Y) - \mu_Y) \otimes (\Phi_Y(Y) - \mu_Y)\right], \qquad \Sigma_{YZ} := \mathbb{E}\left[(\Phi_Y(Y) - \mu_Y) \otimes (\Phi_Z(Z) - \mu_Z)\right],$$

and similarly Σ_{ZZ}.

The conditional covariance operator is

$$\Sigma_{YY|Z} := \Sigma_{YY} - \Sigma_{YZ}\Sigma_{ZZ}^{-1}\Sigma_{ZY}.$$

In finite sample, $\|\Sigma_{YY|Z}\|_{\mathrm{HS}}^2$ admits a Gram-matrix estimator. This yields an optimization over ψ (or over linear $Z = XB$) that becomes an eigenproblem when ψ is linear.

RKHS view: conditional covariance operator and a Gram estimator

Let $\mathcal{H}_Y$ be an RKHS on $\mathcal{Y}$ and $\Phi_Y(Y) \in \mathcal{H}_Y$. Define the covariance operators

$$\Sigma_{YY} := \mathbb{E}\left[(\Phi_Y(Y) - \mu_Y) \otimes (\Phi_Y(Y) - \mu_Y)\right], \qquad \Sigma_{YZ} := \mathbb{E}\left[(\Phi_Y(Y) - \mu_Y) \otimes (\Phi_Z(Z) - \mu_Z)\right],$$

and similarly Σ_{ZZ}.

The conditional covariance operator is

$$\Sigma_{YY|Z} := \Sigma_{YY} - \Sigma_{YZ}\Sigma_{ZZ}^{-1}\Sigma_{ZY}.$$

In finite sample, $\|\Sigma_{YY|Z}\|_{\mathrm{HS}}^2$ admits a Gram-matrix estimator. This yields an optimization over ψ (or over linear $Z = XB$) that becomes an eigenproblem when ψ is linear.

See (Fukumizu, Bach, and Jordan 2004) for the precise estimators and assumptions.

Neighborhood Components Analysis: a probabilistic kNN objective

Neighborhood Components Analysis (NCA) (Goldberger et al. 2004) learns a representation

$$z_i := Ax_i \in \mathbb{R}^d, \qquad A \in \mathbb{R}^{d \times p},$$

so that a softened kNN classifier performs well.

Neighborhood Components Analysis: a probabilistic kNN objective

Neighborhood Components Analysis (NCA) (Goldberger et al. 2004) learns a representation

$$z_i := Ax_i \in \mathbb{R}^d, \qquad A \in \mathbb{R}^{d \times p},$$

so that a softened kNN classifier performs well.

For $i \neq j$, define

$$p_{ij}(A) := \frac{\exp\big(-\|Ax_i - Ax_j\|^2\big)}{\sum_{\ell \neq i} \exp\big(-\|Ax_i - Ax_\ell\|^2\big)}, \qquad p_{ii} := 0.$$

Neighborhood Components Analysis: a probabilistic kNN objective

Neighborhood Components Analysis (NCA) (Goldberger et al. 2004) learns a representation

$$z_i := Ax_i \in \mathbb{R}^d, \qquad A \in \mathbb{R}^{d\times p},$$

so that a softened kNN classifier performs well.

For $i \neq j$, define

$$p_{ij}(A) := \frac{\exp\big(-\|Ax_i - Ax_j\|^2\big)}{\sum_{\ell\neq i}\exp\big(-\|Ax_i - Ax_\ell\|^2\big)}, \qquad p_{ii} := 0.$$

Then the probability that x_i is correctly classified is

$$p_i(A) := \sum_{j:\ c_j = c_i} p_{ij}(A).$$

NCA maximizes

$$F(A) := \sum_{i=1}^{n} p_i(A) \qquad \text{or} \qquad G(A) := \sum_{i=1}^{n} \log p_i(A).$$

NCA: comparison with Rayleigh-quotient methods

Unlike PCA / LDA / kernel-LDA / supervised PCA, NCA is not a Rayleigh quotient.

NCA: comparison with Rayleigh-quotient methods

Unlike PCA / LDA / kernel-LDA / supervised PCA, NCA is not a Rayleigh quotient.

- PCA / (kernel) LDA / supervised PCA $\Rightarrow$ eigenvectors of a symmetric (generalized) eigenproblem.
- NCA $\Rightarrow$ smooth but non-convex objective; optimized by gradient methods.

NCA: comparison with Rayleigh-quotient methods

Unlike PCA / LDA / kernel-LDA / supervised PCA, NCA is not a Rayleigh quotient.

- PCA / (kernel) LDA / supervised PCA $\Rightarrow$ eigenvectors of a symmetric (generalized) eigenproblem.
- NCA $\Rightarrow$ smooth but non-convex objective; optimized by gradient methods.

Heuristic: Rayleigh quotients optimize *global second-order structure*. NCA optimizes a *local classification* criterion (soft kNN).

Roweis Discriminant Analysis: a unifying Rayleigh template

Roweis Discriminant Analysis (RDA) (Ghojogh et al. 2023; Hoi et al. 2006) organizes many supervised DR methods as instances of

$$\operatorname*{arg\,max}_{A\in\mathbb{R}^{d\times p}} \frac{\mathrm{Tr}(A\,\Sigma_{\mathrm{num}}\,A^\top)}{\mathrm{Tr}(A\,\Sigma_{\mathrm{den}}\,A^\top)}.$$

Roweis Discriminant Analysis: a unifying Rayleigh template

Roweis Discriminant Analysis (RDA) (Ghojogh et al. 2023; Hoi et al. 2006) organizes many supervised DR methods as instances of

$$\underset{A\in\mathbb{R}^{d\times p}}{\arg\max}\,\frac{\mathrm{Tr}(A\,\Sigma_{\mathrm{num}}\,A^\top)}{\mathrm{Tr}(A\,\Sigma_{\mathrm{den}}\,A^\top)}.$$

Typical building blocks are the same three covariances

$$\Sigma_T, \qquad \Sigma_B, \qquad \Sigma_W,$$

combined with weights to interpolate between reconstruction and discrimination.

Roweis Discriminant Analysis: a unifying Rayleigh template

Roweis Discriminant Analysis (RDA) (Ghojogh et al. 2023; Hoi et al. 2006) organizes many supervised DR methods as instances of

$$\arg\max_{A\in\mathbb{R}^{d\times p}} \frac{\mathrm{Tr}(A\,\Sigma_{\mathrm{num}}\,A^\top)}{\mathrm{Tr}(A\,\Sigma_{\mathrm{den}}\,A^\top)}.$$

Typical building blocks are the same three covariances

$$\Sigma_T, \qquad \Sigma_B, \qquad \Sigma_W,$$

combined with weights to interpolate between reconstruction and discrimination.

With a whitening constraint $A\Sigma_{\mathrm{den}}A^\top = I_d$, the solution is given by the top generalized eigenvectors of

$$\Sigma_{\mathrm{num}}a = \lambda\,\Sigma_{\mathrm{den}}a.$$

Double Supervised Discriminant Analysis: two uses of supervision

In the Roweis template, an extreme regime is to use supervision both in the numerator and in the normalization.

Double Supervised Discriminant Analysis: two uses of supervision

In the Roweis template, an extreme regime is to use supervision both in the numerator and in the normalization.

Schematic form:

$$\arg\max_{A} \frac{\mathrm{Tr}(A\,\Sigma_B^{(\mathrm{sup})}\,A^\top)}{\mathrm{Tr}(A\,\Sigma_{\mathrm{den}}^{(\mathrm{sup})}\,A^\top)}.$$

Double Supervised Discriminant Analysis: two uses of supervision

In the Roweis template, an extreme regime is to use supervision both in the numerator and in the normalization.

Schematic form:

$$\arg\max_{A} \frac{\mathrm{Tr}(A\,\Sigma_B^{(\mathrm{sup})}\,A^\top)}{\mathrm{Tr}(A\,\Sigma_{\mathrm{den}}^{(\mathrm{sup})}\,A^\top)}.$$

Take-home: stronger supervision can yield sharper class separation, at the price of increased sensitivity to label noise / overfitting.

Large Margin Nearest Neighbor (LMNN): metric learning

A classical alternative to Fisher ratios is to learn a Mahalanobis metric

$$d_M(x, x')^2 := (x - x')^\top M (x - x'), \qquad M \succeq 0,$$

directly optimized for kNN classification performance.

Large Margin Nearest Neighbor (LMNN): metric learning

A classical alternative to Fisher ratios is to learn a Mahalanobis metric

$$d_M(x, x')^2 := (x - x')^\top M(x - x'), \qquad M \succeq 0,$$

directly optimized for kNN classification performance.

LMNN (Weinberger and Saul 2009) selects for each i a set of target neighbors $\mathcal{N}_i$ (same class), and enforces a margin against impostors (different class).

Large Margin Nearest Neighbor (LMNN): metric learning

A classical alternative to Fisher ratios is to learn a Mahalanobis metric

$$d_M(x,x')^2 := (x-x')^\top M(x-x'), \qquad M \succeq 0,$$

directly optimized for kNN classification performance.

LMNN (Weinberger and Saul 2009) selects for each i a set of target neighbors $\mathcal{N}_i$ (same class), and enforces a margin against impostors (different class).

LMNN solves a convex problem in M:

$$\min_{M\succeq 0} \sum_i \sum_{j\in\mathcal{N}_i} d_M(x_i,x_j)^2 + C \sum_i \sum_{j\in\mathcal{N}_i} \sum_{\ell:\ c_\ell\neq c_i} \left[1 + d_M(x_i,x_j)^2 - d_M(x_i,x_\ell)^2\right]_+ .$$

References

Alain, Guillaume and Yoshua Bengio (2014). **"What regularized auto-encoders learn from the data-generating distribution".** In: *Journal of machine learning research* 15.1, pp. 3563–3593.

Barshan, Elnaz, Ali Ghodsi, Zohreh Azimifar, and Mansoor Zolghadri Jahromi (2011). **"Supervised principal component analysis: visualization, classification and regression on subspaces and submanifolds".** In: *Pattern recognition* 44.7, pp. 1357–1371.

Bengio, Yoshua, Pascal Lamblin, Dan Popovici, and Hugo Larochelle (2007). **"Greedy layer-wise training of deep networks".** In: *Advances in neural information processing systems*. Vol. 19.

Bishop, Christopher M. (1995). **"Training with noise is equivalent to Tikhonov regularization".** In: *Neural computation* 7.1, pp. 108–116. DOI: 10.1162/neco.1995.7.1.108.

Deng, Weihong, Jiani Hu, Jun Guo, and Honggang Zhang (2007). **"Robust discriminant analysis of gabor feature for face recognition".** In: *Fourth international conference on fuzzy systems and knowledge discovery (fskd 2007)*. Vol. 3. IEEE, pp. 248–252.

Fisher, Ronald A (1936). **"The use of multiple measurements in taxonomic problems".** In: *Annals of eugenics* 7.2, pp. 179–188.

Fukumizu, Kenji, Francis R Bach, and Michael I Jordan (2004). **"Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces".** In: *Journal of machine learning research* 5.Jan, pp. 73–99.

Ghojogh, Benyamin, Mark Crowley, Fakhri Karray, and Ali Ghodsi (2023). **Elements of dimensionality reduction and manifold learning.** Springer.

Giulini, Ilaria (2016). **"Kernel spectral clustering".** In: *Arxiv preprint arxiv:1606.06519*.

Goldberger, Jacob, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov (2004). **"Neighbourhood components analysis".** In: *Advances in neural information processing systems* 17.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). **Deep learning.** Available at `https://www.deeplearningbook.org/`. MIT Press.

Gretton, Arthur, Karsten M. Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alexander J. Smola (2012). **"A kernel two-sample test".** In: *Journal of machine learning research* 13.Mar, pp. 723–773.

Guo, Ming and Zhelong Wang (2015). **“A feature extraction method for human action recognition using body-worn inertial sensors”.** In: *2015 ieee 19th international conference on computer supported cooperative work in design (cscwd)*. IEEE, pp. 576–581.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). **“Deep residual learning for image recognition”.** In: *Proceedings of the ieee conference on computer vision and pattern recognition (cvpr)*, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

Hinton, Geoffrey E. and Ruslan R. Salakhutdinov (2006). **“Reducing the dimensionality of data with neural networks”.** In: *Science* 313.5786, pp. 504–507. DOI: 10.1126/science.1127647.

Hoi, Steven CH, Wei Liu, Michael R Lyu, and Wei-Ying Ma (2006). **“Learning distance metrics with contextual constraints for image retrieval”.** In: *2006 ieee computer society conference on computer vision and pattern recognition (cvpr’06)*. Vol. 2. IEEE, pp. 2072–2078.

Kim, Seung-Jean, Alessandro Magnani, and Stephen Boyd (2005). **“Robust fisher discriminant analysis”.** In: *Advances in neural information processing systems* 18.

Kingma, Diederik P. and Max Welling (2014). **“Auto-encoding variational Bayes”.** In: *Arxiv preprint arxiv:1312.6114*. arXiv: 1312.6114 [stat.ML].

Makhzani, Alireza, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow (2016). **“Adversarial autoencoders”.** In: *International conference on learning representations (iclr)*. arXiv:1511.05644.

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra (2014). **"Stochastic backpropagation and approximate inference in deep generative models".** In: *Proceedings of the 31st international conference on machine learning (icml)*. Vol. 32. Proceedings of Machine Learning Research. PMLR, pp. 1278–1286.

Rifai, Salah, Grégoire Mesnil, Pascal Vincent, Xavier Muller, Yoshua Bengio, Yann Dauphin, and Xavier Glorot (2011). **"Contractive auto-encoders: explicit invariance during feature extraction".** In: *Proceedings of the 28th international conference on machine learning (icml)*, pp. 833–840.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). **"U-net: convolutional networks for biomedical image segmentation".** In: *Medical image computing and computer-assisted intervention (miccai)*. Vol. 9351. Lecture Notes in Computer Science. Springer, pp. 234–241. DOI: 10.1007/978-3-319-24574-4_28.

Shental, Noam, Tomer Hertz, Daphna Weinshall, and Misha Pavel (2002). **"Adjustment learning and relevant component analysis".** In: *Computer vision—eccv 2002: 7th european conference on computer vision copenhagen, denmark, may 28–31, 2002 proceedings, part iv 7*. Springer, pp. 776–790.

Shi, Jianbo and Jitendra Malik (2000). **"Normalized cuts and image segmentation".** In: *Ieee transactions on pattern analysis and machine intelligence* 22.8, pp. 888–905.

Vincent, Pascal, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol (2008). **“Extracting and composing robust features with denoising autoencoders”.** In: *Proceedings of the 25th international conference on machine learning (icml)*, pp. 1096–1103. DOI: 10.1145/1390156.1390294.

Von Luxburg, Ulrike (2007). **“A tutorial on spectral clustering”.** In: *Statistics and computing* 17, pp. 395–416.

Weinberger, Kilian Q and Lawrence K Saul (2009). **“Distance metric learning for large margin nearest neighbor classification.”.** In: *Journal of machine learning research* 10.2.