Canonical pre-processing

- It is common to perform the following pre-processing steps before doing any sort of learning on image patches:

1. DC centering

\[ I(x, y) \leftarrow I(x, y) - \frac{1}{MN} \sum_{x', y'} I(x', y') \]

2. Contrast normalization

\[ I(x, y) \leftarrow \frac{I(x, y)}{\sqrt{\sum_{x,y} I(x, y)^2} + \epsilon} \]

3. Whitening (today's lecture)

Don’t confuse DC-centering/contrast normalization with:

1. Mean-centering each pixel.
2. Setting the standard deviation of each pixel to 1.

But one may do this in addition.

The difference between these sets of operations is that they work along a different dimension of your image data array.

The “vision equation”

- The purpose of vision: Infer world properties (or hidden “causes”), \( z \), from an image, \( x \).

- We can express this with an analysis, inference, encoder or backward equation:

\[ z = g(x) \]

- Learning amounts to estimating the parameters of \( g \) from data.
Latent variables and generative models

▶ In practice, it is often easier to write down how images got formed given the causes.
▶ This leads to the synthesis, or decoder, or forward equation:

\[ x = f(z) \]

▶ It describes how images depend on the state of the world.
▶ \( z \) is called “latent variable” or “hidden variable”, because unlike the image, \( x \), we do not observe it.

Manifold learning

▶ When the dimensionality of the latent variables is smaller than the dimensionality of the data, then we can think of the data as being distributed along some lower-dimensional manifold in the dataspace.
▶ Learning the manifold is known as dimensionality reduction.

Principal Components Analysis (PCA)

▶ If we assume the manifold to be linear, learning is easy and can done in closed form.
▶ It amounts to finding the latent subspace.
▶ Inference amounts to projecting data into that subspace.
Principal Components Analysis

- Learning the linear manifold is known as Principal Components Analysis (PCA).
- There are a lot of equivalent learning criteria leading to PCA.
- Two of the most well-known are
  1. find the subspace in which the projection of the training data has maximal variance
  2. maximize the average distance between the projections and the original points.

Roland Memisevic Machine learning for vision

Principal Components Analysis

- The variance along the manifold is large.
- The average projection error is small.

Roland Memisevic Machine learning for vision

Principal Components Analysis

- To learn a subspace means we need to work under the assumption that the data is mean-centered:
  \[ \frac{1}{N} \sum_{n=1}^{N} x_n = 0 \]
- To derive PCA, we define an orthonormal basis for the subspace, consisting of vectors
  \[ u_1, \ldots, u_M \]

where \( M \) is smaller than the dimensionality of the data.
- PCA amounts to learning this basis.
Principal Components Analysis
▶ It is convenient to stack the basis-vectors column-wise in matrix $U$.
▶ Assuming we had already learned the optimal basis, we can write the forward and backward mappings as:

Projecting data (backward mapping)
▶ The optimal coefficients that approximate $x$ within the subspace are given by
$$z = U^T x$$

Reconstructing data (forward mapping)
▶ The approximation $\tilde{x}$ of $x$ is given by
$$\tilde{x} = Uz = UU^T x$$

Optimizing quadratic forms
▶ The maximizer of
$$\text{Tr}(U^T A U)$$
subject to
$$U^T U = I$$
(where $U$ is $D \times M$) is given by the matrix whose columns are the eigenvectors of $A$ corresponding to the $M$ largest eigenvalues.

So to find principal components perform an eigen-decomposition of the data covariance matrix.

Principal Components Analysis
▶ One way to learn the subspace: minimize reconstruction error
$$E(U) = \sum_n \|x_n - UU^T x_n\|^2$$
under the constraint $U^T U = I$
▶ To solve the problem, we stack the data row-wise in matrix $X$ and rewrite the objective function as a quadratic form in $U$:
$$E(U) = \|X - UU^T X\|_F^2$$
$$= \text{Tr}((X^T - UU^T X)^T(X^T - UU^T X))$$
$$= \text{Tr}(XX^T) - \text{Tr}(U^T X^T X U)$$
$$= -\text{Tr}(U^T X^T X U) + \text{const}$$
▶ Optimizing a quadratic form under an orthonormality constraint is a common exercise in linear algebra:

Summary: Computing principal components
1. Mean-center the data.
2. Compute the covariance matrix $C = \frac{1}{N} X^T X$.
3. Perform an eigen-decomposition of $C$.
4. Sort the eigen-vectors according to the size of their eigenvalues.
5. Stack the leading $M$ eigen-vectors in matrix $U$. 
Principal Components Analysis

- A two-dimensional dataset and the two principal components.
- Projections onto the leading eigenvectors preserve most of the variability in the data. So PCA performs *lossy compression.*

Translation invariance 1d

- The covariance between natural image pixels does not depend (much) on their absolute position.

Translation invariance 2d

PCA and Fourier analysis 1d

- A (covariance) matrix whose entries are translation-invariant has phasors as eigenvectors:
  
  \[
  \sum_{t'} \text{cov}(t, t') e^{i\omega t'} = \sum_{t'} c(t - t') e^{i\omega t'} = \sum_{z} c(z) e^{i\omega t} e^{-i\omega z} = \left[ \sum_{z} c(z) e^{-i\omega z} \right] e^{i\omega t}
  \]

- In fact, the covariance matrix defines a convolution.
PCA and Fourier analysis 1d

▶ Covariance matrices are symmetric \((c(z) = c(T - z))\)
▶ So the eigenvalues are real:

\[
\left[ \sum_z c(z) e^{i\omega z} \right] e^{i\omega t} = [c(0) + \sum_{z=1}^{T-1} c(z)(e^{i\omega z} + e^{-i\omega z})] e^{i\omega t}
\]

\[
= [c(0) + 2 \sum_{z=1}^{T-1} c(z) \cos(\omega z)] e^{i\omega t}
\]

PCA example (first 96 EVs)

from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)

PCA and Fourier analysis 2d

▶ An image covariance matrix whose entries are translation-invariant has 2d waves as eigenvectors:

\[
\sum_{x', y'} \text{COV}((x, y), (x', y')) e^{i(\omega_1 x' + \omega_2 y')}
\]

\[
= \sum_{x', y'} c((x - x')^2 + (y - y')^2) e^{i(\omega_1 x' + \omega_2 y')}
\]

\[
= \sum_{\xi, \eta} c(\xi, \eta) e^{i(\omega_1 x - \omega_1 \xi + \omega_2 y - \omega_2 \eta)}
\]

\[
= \left[ \sum_{\xi, \eta} c(\xi, \eta) e^{-i(\omega_1 \xi + \omega_2 \eta)} \right] e^{i(\omega_1 x + \omega_2 y)}
\]

Dimensionality reduction and anti-aliasing

▶ Thus, PCA dimensionality reduction is low-pass filtering.
▶ Low-pass filtering can be a good idea because:
   1. In rectangular images, oblique frequencies are overrepresented as compared to vertical or horizontal frequencies.
   2. Phase becomes meaningless at the highest representable frequencies.

from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)
The components of the features, $Z$, are uncorrelated (that is, $Z$ has a diagonal covariance matrix):

\[
\frac{1}{N} \sum_n z_n z_n^T = \frac{1}{N} \sum_n U^T x_n x_n^T U
= U^T \left( \frac{1}{N} \sum_n x_n x_n^T \right) U
= U^T C U
= L
\]

where the diagonal matrix $L$ contains the eigenvalues of $C$ on its diagonal.

(The last step follows from the eigenvalue definition: $C u_i = \lambda_i u_i$)

We can obtain the identity as the covariance matrix for $Z$, if we define the forward mapping as

\[
V = L^{-\frac{1}{2}} U^T
\]

Data with identity covariance matrix is known as white; multiplying data by $V$ as whitening.

Whitening may be performed without reducing the dimensionality.

This amounts to just rotating the coordinate system of the data, followed by independently “stretching” or “squeezing” the dimensions to obtain unit variance in each.
ZCA Whitening

- Multiplying whitened data with an orthonormal matrix leaves the data white (exercise).
- Thus, the whitening matrix $V = L^{-\frac{1}{2}}U^T$ is not the only whitening matrix. Any matrix $AV$ with orthonormal $A$ will be, too.
- One way to define a canonical whitening matrix is to choose the symmetric one: Get the symmetric whitening matrix $W$ by pre-multiplying $V$ with the orthonormal matrix $U$:

$$W := UV = UL^{-\frac{1}{2}}U^T$$

- Transforming data with this matrix is known as ZCA (zero-phase components analysis).
PCA whitened data

ZCA whitened data

ZCA whitened data and original data

ZCA example (all columns of $W$)

from: Natural Image Statistics (Hyvarinen, Hurri, Hoyer; 2009)
De-whitening

- How to get the original images back from the whitened images?

(Pseudo-)invert the whitening matrix:

\[ x = V^{-1}z = UL^\frac{1}{2}z \] (inverse PCA whitening)

\[ x = W^{-1}z = UL^\frac{1}{2}U^Tz \] (inverse ZCA whitening)

where \( L^\frac{1}{2} \) is diagonal with \( (L^\frac{1}{2})_{ii} = \sqrt{\lambda_i} \)

Probabilistic PCA

- One can define PCA also as a probabilistic latent variable model (Tipping, Bishop; 1999).
- Assume a Gaussian prior distribution over latent variables

\[ p(z) = \mathcal{N}(z|0, I) \]

and a Gaussian conditional distribution over images \( x \)

\[ p(x|z) = \mathcal{N}(x|Wz + \mu, \sigma^2 I) \]

This defines the following generative model: First draw from a low-dimensional Gaussian in the latent space, and then draw the observation from a \( D \)-dimensional conditional Gaussian whose mean depends on the latent variable.

from: Bishop 2009
Probabilistic PCA

- The backward mapping now follows from Bayes’ rule:
  \[ p(z|x) = \frac{p(x|z)p(z)}{\int_z p(x|z)p(z) \, dz} \]
- Plugging in the Gaussian yields
  \[ p(z|x) = \mathcal{N}(z|M^{-1}W^T(x - \mu), \sigma^{-2}M) \]
  with \( M = W^T W + \sigma^2 I \)
- The marginals turn out to be Gaussian, too:
  \[ p(x) = \int_z p(x|z)p(z) \, dz = \mathcal{N}(x|\mu, WW^T + \sigma^2 I) \]
- This is simply a Gaussian, whose covariance matrix is the outer product of two low-rank matrices (plus noise).

Factor Analysis

- Probabilistic PCA is closely related to a statistical model known as Factor Analysis:
- For factor analysis the conditional Gaussian
  \[ p(x|z) = \mathcal{N}(x|Wz + \mu, \Psi) \]
  has a diagonal (not necessarily spherical) covariance matrix \( \Psi \).
- Learning is similar to probabilistic PCA.