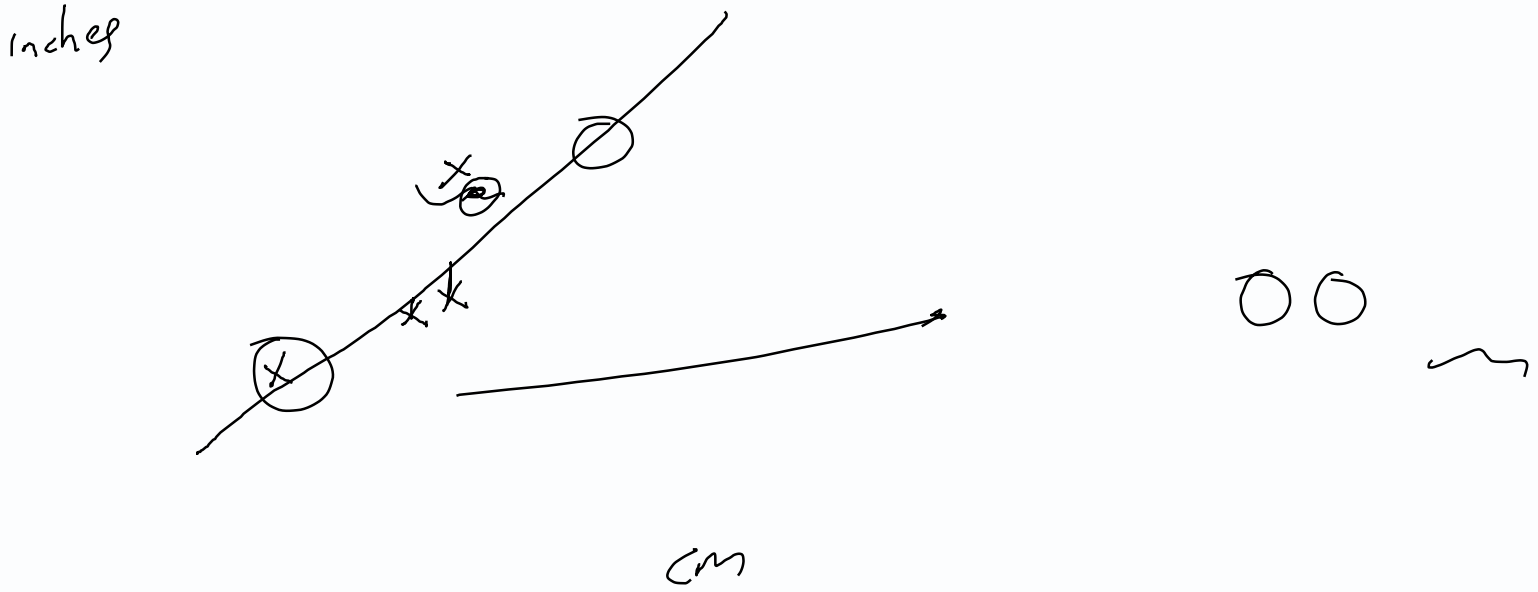
Dimensionality Reduction

By Principal Component Analysis

Ioannis Patras

Dimensionality Reduction - motivation

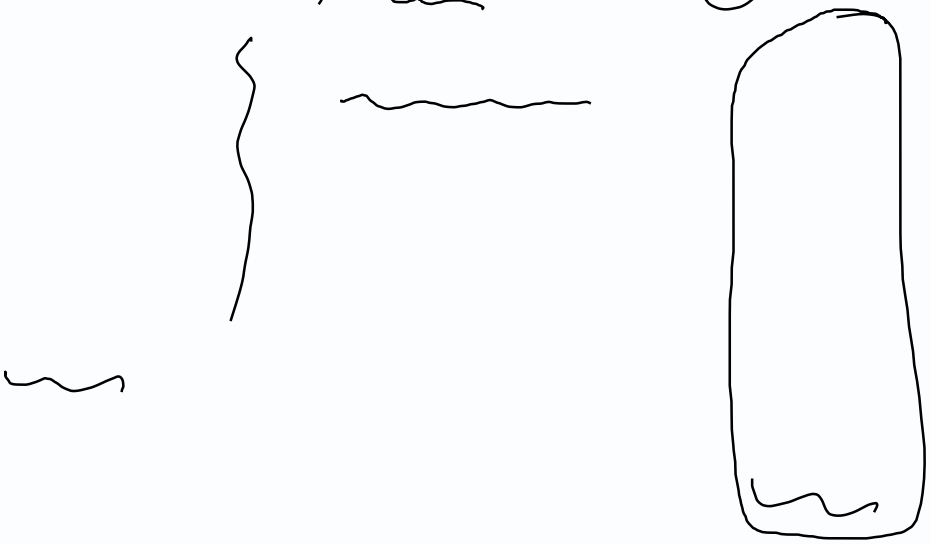


Reduce the dimensionality of each data sample retaining as much as possible information.

Linear Dimensionality Reduction

New feature space is obtained by a linear transform, that is

$$z = U^T x, \quad \text{where } U^T \in R^{k \times m}$$

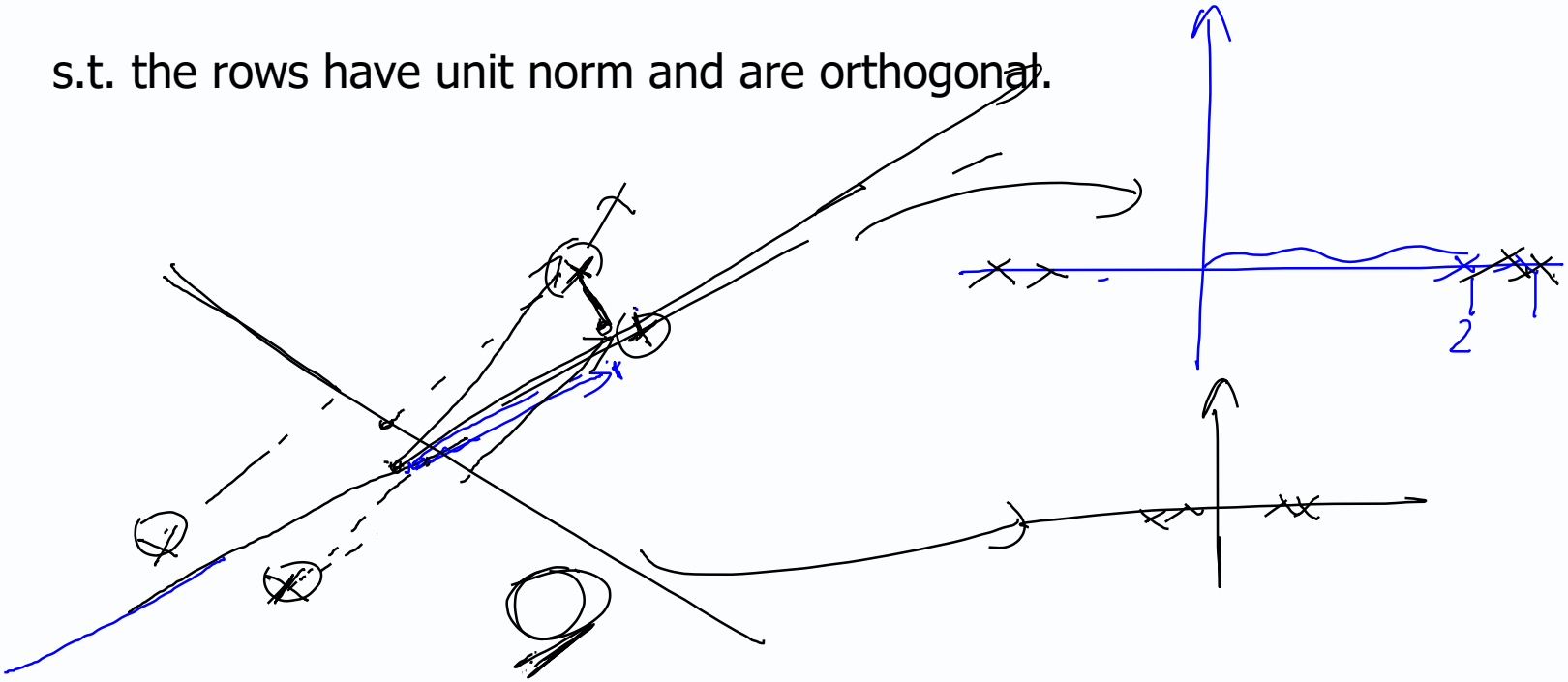


Projection in the space spanned by the rows of $U^T \in R^{k \times m}$

Principal Component Analysis

Find the projection matrix that minimizes the projection error

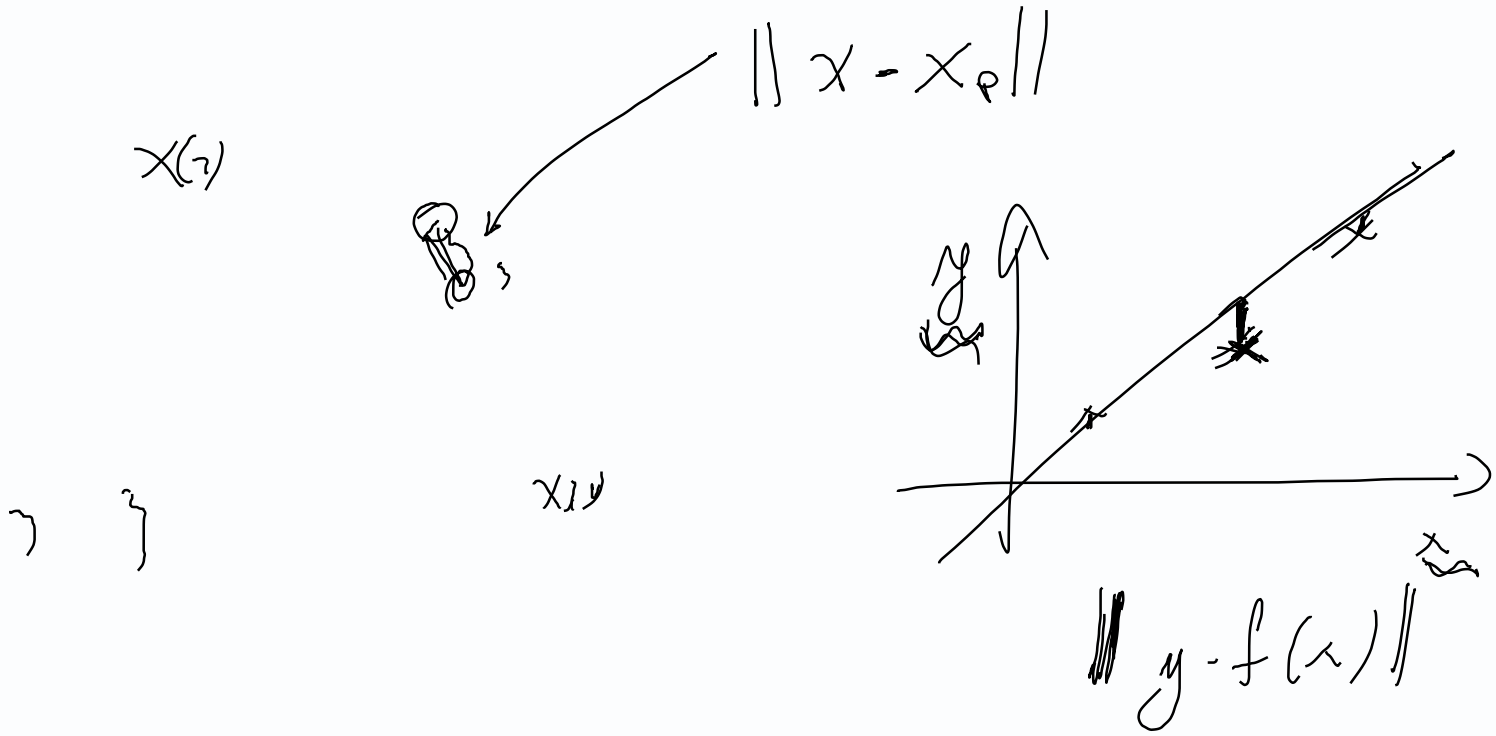
s.t. the rows have unit norm and are orthogonal.



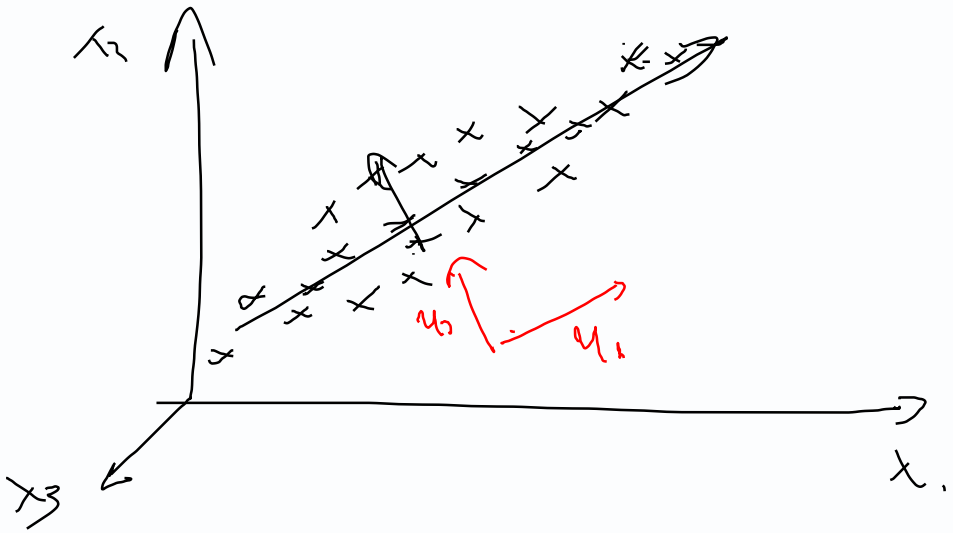
$$z = U^T x, \quad \text{where } U^T \in R^{k \times m}$$

Principal Component Analysis

Find the projection matrix that maximizes the variance along the projection s.t. the rows have unit norm and are orthogonal.



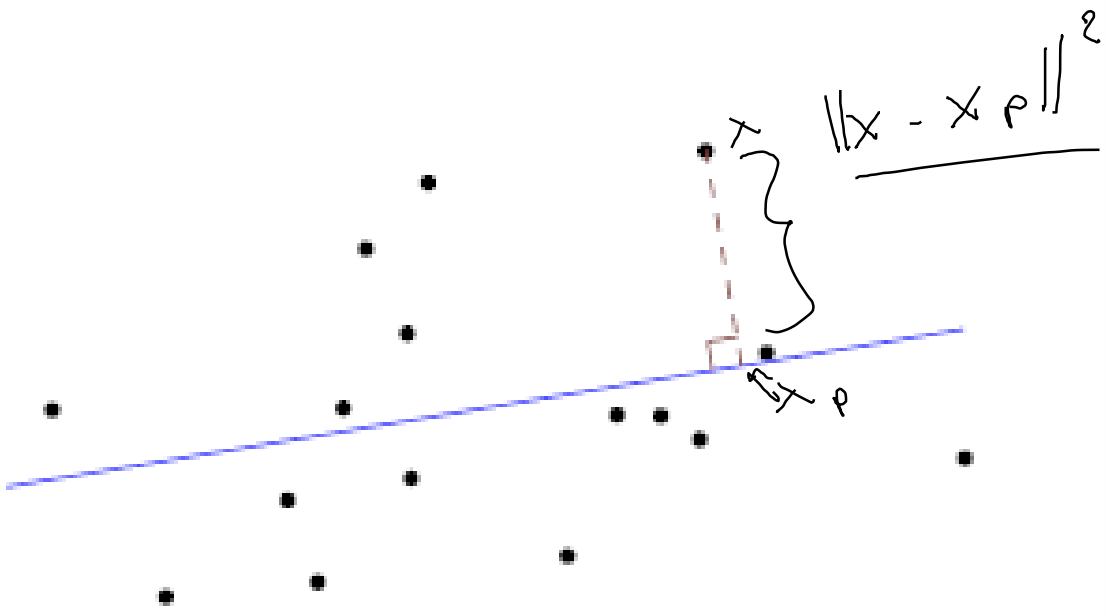
$$z = U^T x, \quad \text{where } U^T \in R^{k \times m}$$


x_2
x_3
x_1
u_2
u_1

Principal Subspace Projections

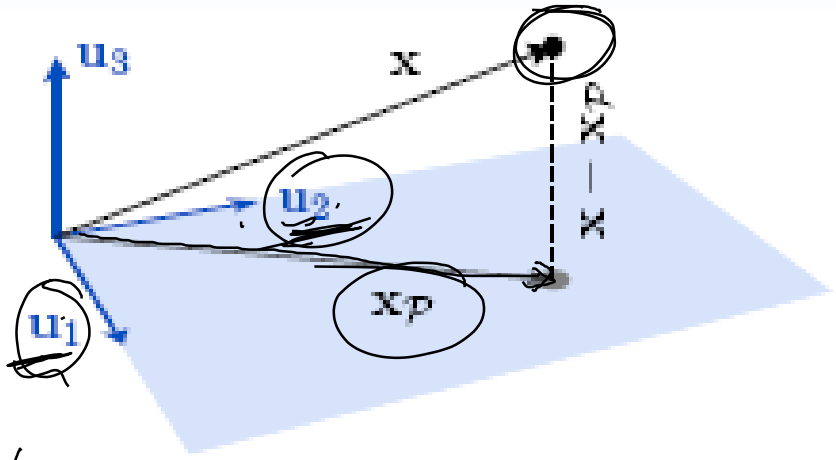
Introduced by Pearson in 1901 in "On lines and planes of closest fit to a system of points in space" [4]. Also known as the Karhunen-Loéve transform (KLT), or the Hotelling transform [3].

Aim is to find the affine **subspace** of a given dimensionality (a line, plane, or hyperplane, not necessarily passing through origin) that best represents a set of points.



Goodness of fit is measured as **mean-square** distance from each to point its orthogonal projection into the affine subspace.

Now consider projection down to an m-dimensional linear subspace $\mathcal{P}$. Set up an orthonormal basis $(\mathbf{u}_1, \ldots, \mathbf{u}_m)$ spanning $\mathcal{P}$. Another $n-m$ basis vectors $(\mathbf{u}_{m+1}, \ldots, \mathbf{u}_n)$ are chosen orthogonal to $\mathcal{P}$ to span the rest of the data space.



Let projection of $\mathbf{x}$ into $\mathcal{P}$ be $\mathbf{x}_\mathcal{P} = \sum_{i=1}^{m} \xi_i \mathbf{u}_i$, the ξ_i are coordinates of $\mathbf{x}_\mathcal{P}$ in $\mathcal{P}$. Given $\mathbf{x}$ and the $\mathbf{u}_i$, we choose the ξ_i to minimise norm of residual $\|\mathbf{x} - \mathbf{x}_\mathcal{P}\|^2$. Setting derivatives w.r.t. ξ_j to zero gives (for $1 \le j \le m$)

$$\frac{\partial}{\partial \xi_j} \frac{1}{2} \|\mathbf{x} - \mathbf{x}_\mathcal{P}\|^2 = -(\mathbf{x} - \mathbf{x}_\mathcal{P}) \cdot \mathbf{u}_j = 0 \quad \Longrightarrow \quad \mathbf{x} \cdot \mathbf{u}_j = \sum_{i=1}^{m} \xi_i \mathbf{u}_i \cdot \mathbf{u}_j$$

By construction, $\mathbf{u}_i \cdot \mathbf{u}_j = \delta_{ij}$, so $\xi_j = \mathbf{u}_j \cdot \mathbf{x}$ or $\xi_j = \mathbf{u}_j^T \mathbf{x}$ using matrix notation. Incidentally, since $(\mathbf{x} - \mathbf{x}_\mathcal{P}) \cdot \mathbf{u}_j = 0 \quad \forall \quad 1 \le j \le m$, this is an orthogonal

To find the principal subspace, we must choose the $\mathbf{u}_i$ to minimise the mean-square residual:

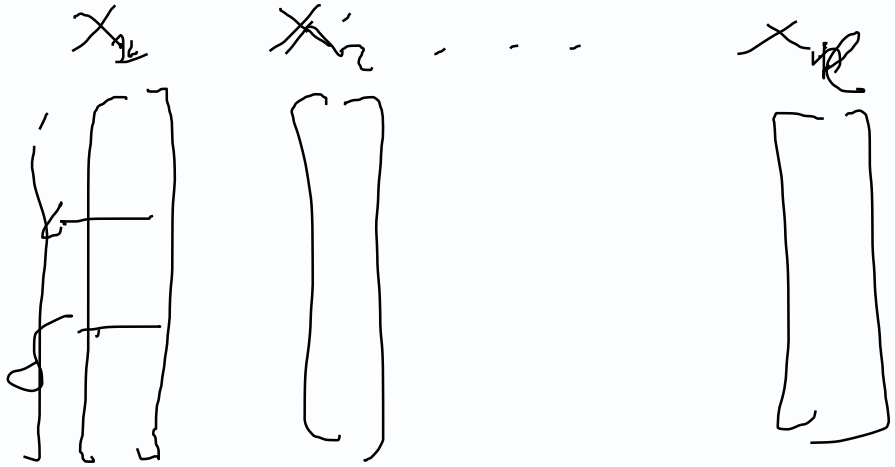
$$J_{\mathcal{P}} = \left\langle \tfrac{1}{2} \|\mathbf{x} - \mathbf{x}_{\mathcal{P}}\|^2 \right\rangle.$$

We already know that $\mathbf{x}_{\mathcal{P}} = \sum_{i=1}^{m} \xi_i \mathbf{u}_i$, and $\xi_i = \mathbf{u}_i \cdot \mathbf{x}$, so $\|\mathbf{x} - \mathbf{x}_{\mathcal{P}}\|^2 = \|\mathbf{x}\|^2 - \sum_{i=1}^{m} \xi_i^2$. For a given data set, $\langle \|\mathbf{x}\|^2 \rangle$ is fixed, so minimising squared error is equivalent to *maximising* the variance of the projected coordinates. Recall that $(\mathbf{u}_1, \ldots, \mathbf{u}_n)$ forms a *full* basis of the data space; this means that $\|\mathbf{x}\|^2 = \sum_{i=1}^{n} \xi_i^2$, so $\|\mathbf{x}_{\mathcal{P}} - \mathbf{x}\|^2 = \sum_{i=m+1}^{n} \xi_i^2$, that is, the variance 'lost' in projection.

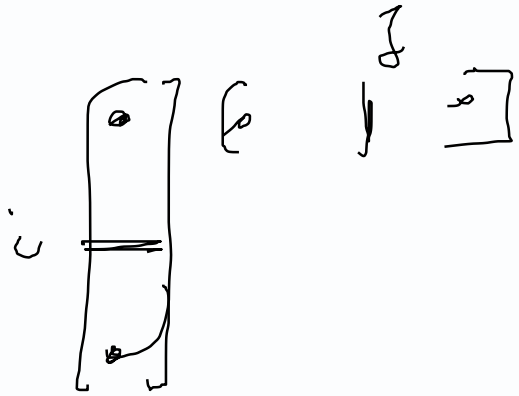
Summary: To minimise $J_{\mathcal{P}}$, we must maximise

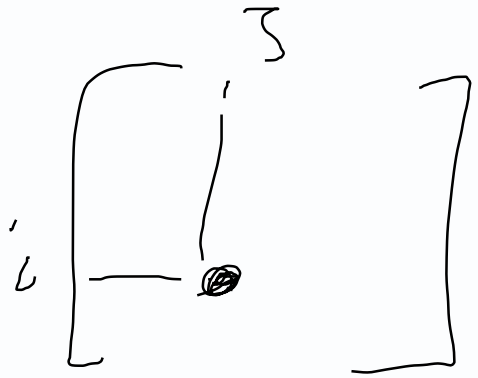
$$\sum_{i=1}^{m} \langle \xi_i^2 \rangle = \sum_{i=1}^{m} \langle (\mathbf{u}_i^T \mathbf{x})(\mathbf{x}^T \mathbf{u}_i) \rangle = \sum_{i=1}^{m} \mathbf{u}_i^T \langle \mathbf{x}\mathbf{x}^T \rangle \mathbf{u}_i$$

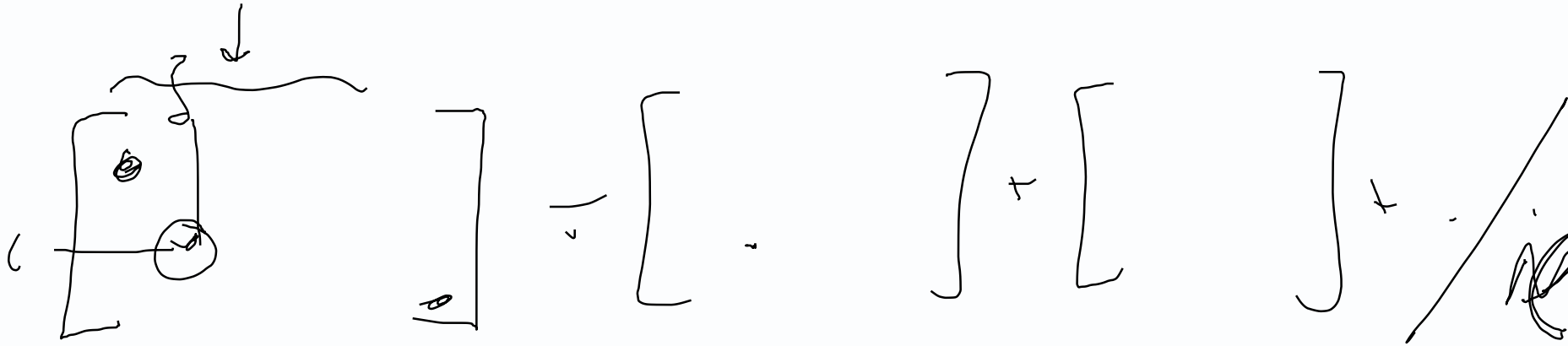
under the constraint that $\mathbf{u}_i$ form orthonormal basis, i.e. $\mathbf{u}_i^T \mathbf{u}_j = \delta_{ij}$. This can be done using Lagrange multipliers. Note that $\langle \mathbf{x}\mathbf{x}^T \rangle$ is the sample covariance matrix. Note also that *rotation* of the $(\mathbf{u}_1, \ldots, \mathbf{u}_m)$ within their own subspace has no effect on $J_{\mathcal{P}}$.



$$C_{ij} = \langle x(i) \cdot x(j) \rangle$$







For example: For $m = 1$, i.e. projection onto a line, only one basis vector $\mathbf{u}$ is required to span the principal subspace. We must find the stationary points of

$$L = \frac{1}{2}\mathbf{u}^T\mathbf{C}\mathbf{u} - \frac{1}{2}\lambda(\mathbf{u}^T\mathbf{u} - 1)$$

where $\mathbf{C} = \langle \mathbf{x}\mathbf{x}^T \rangle$, the covariance matrix, and λ is a Lagrange multiplier. The stationary points are given by the solutions of

$$\frac{\partial L}{\partial \mathbf{u}} = \mathbf{C}\mathbf{u} - \lambda\mathbf{u} = \mathbf{0}.$$

So, $\mathbf{u}$ is defined by $\mathbf{C}\mathbf{u} = \lambda\mathbf{u}$. This is just the defining condition for an eigenvector of $\mathbf{C}$, with eigenvalue λ. (See next slide.) To maximise $\frac{1}{2}\mathbf{u}^T\mathbf{C}\mathbf{u}$, we choose the eigenvector with the *largest* eigenvalue.

In general, for $m > 1$, the principal subspace is spanned by the m eigenvectors associated with the m largest eigenvalues of the covariance matrix $\mathbf{C}$.

Eigenvalue decomposition of a matrix

For an $n \times n$ matrix $\mathbf{A}$, any vector $\mathbf{u}$ that satisfies $\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$ is called an eigenvector of $\mathbf{A}$ of eigenvalue λ. The λ are the roots of the polynomial

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0$$

and hence there are n of them, though they may be complex. If $\mathbf{A}$ is real and symmetric (or complex and Hermitian) then the eigenvalues are all real, and the eigenvectors form an orthonormal basis. If several eigenvalues are the same, then there is a rotational indeterminacy in their eigenvectors.

Given the n eigenvalues λ_i and eigenvectors $\mathbf{u}_i$, the matrix $\mathbf{A}$ can be written

$$\mathbf{A} = \sum_{i=1}^{n} \lambda_i \mathbf{u}_i \mathbf{u}_i^T = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$$

where $\mathbf{U}$ is a matrix whose columns are the $\mathbf{u}_i$ and $\mathbf{\Lambda}$ is a diagonal matrix with the λ_i down the main diagonal. The matrices $\mathbf{U}$ and $\mathbf{\Lambda}$ can be found in MATLAB using the `eig` or `svd` functions.

Principal component analysis

Step 1: Subtract the mean

$$x_n \leftarrow x_n - \frac{1}{N}\sum_{n'} x_{n'}$$

Step 2: Normalise the features to have unit variance

$$x_n(l) \leftarrow x_n(l) \Big/ \sqrt{\frac{1}{N}\sum_{n'} \left(x_{n'}(l)\right)^2}$$

Step 3: Calculate the covariance

$$C \leftarrow \frac{1}{N}\sum_{n} x_n x_n^T$$

Step 4: Solve for the eigenvalues and eigenvectors

$$Cu_j - \lambda_j u_j = 0$$

Step 5: Set the projection matrix

$$U = [u_1, \ldots, u_k]$$

Projection:

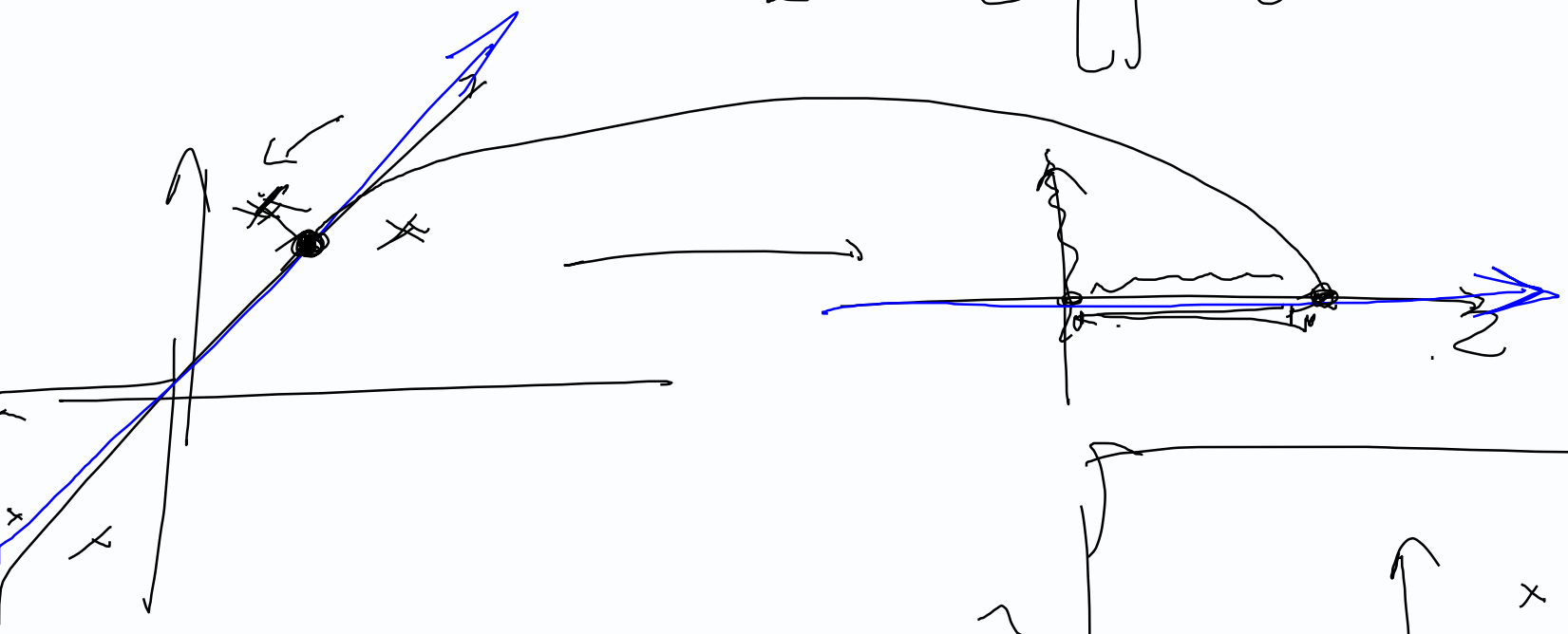
$$z = U^T x$$

Reconstruction:

$$\hat{x} = Uz$$

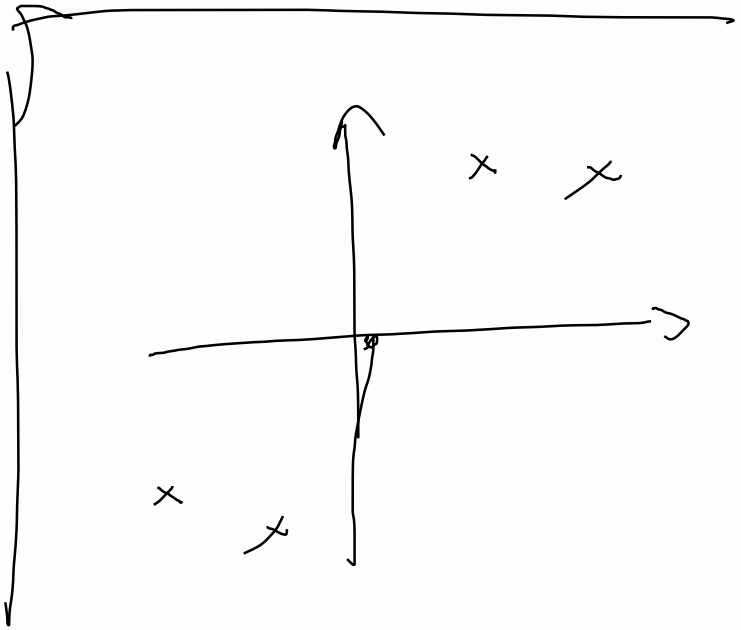
$z = M^{-1} x$

$$\begin{bmatrix} u_1^T \\ u_2^T \end{bmatrix} \begin{bmatrix} \\ x \\ \\ \end{bmatrix} = \begin{bmatrix} \\ z \\ \\ \end{bmatrix}$$



$$U \quad z = \hat{x}$$

$$\begin{bmatrix} \\ S \\ \\ \end{bmatrix} \begin{bmatrix} \\ \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \hat{x} \\ \\ \end{bmatrix}$$



PCA in MATLAB using `eig`

We'll assume our data lives in an N-dimensional vector space. We have L samples in our data set, arranged as column vectors in a $N \times L$ matrix `z`. If we cannot assume the data is zero mean, then the first step is to subtract the sample mean:

```
x = z - mean(z,2)
```

Next, we need to estimate the covariance of the data and compute the eigenvectors:

```
C = x*x'/size(x,2)
[U,D] = eig(C)
```

The matrix `D` contains the eigenvalues down the diagonal. The eigenvector `U(:,i)` has the eigenvalue `D(i,i)`, so it should be a simple matter to sort the eigenvalues and find the columns of `U` which correspond to the largest M eigenvalues. For example, if these turn out to be columns 4 to 7, then the dimension-reduced data is `U(:,4:7)*x`.

PCA in MATLAB using `svd`

An alternative is to use the Singular Value Decomposition of the zero-mean data `x`:

```
[U,D,V]=svd(x,'econ')
```

The return values are such that `U*D*V' = x`, where `U` is an $N \times N$ orthogonal matrix (i.e. `U*U' = U'*U = eye(N)`), `D` is an $N \times N$ *diagonal* matrix, and `V` is an $L \times N$ such that `V'*V = eye(N)`.

It can be shown that the columns of `U` are actually the principal components (in the right order too), and the first M rows of `D*V'` contain the M-dimensional projection of the data. The values down the diagonal of the matrix `D` give the standard deviation of the original data set in each of the principal directions.

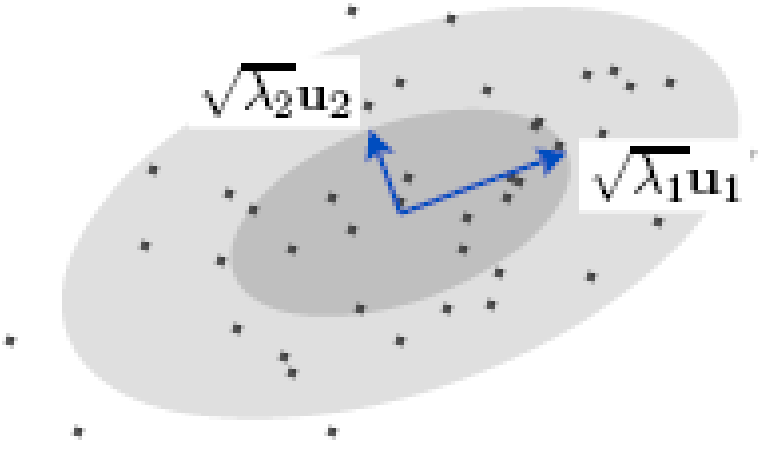
Variance and covariance: a geometric view

Variance is easy to understand as the 'spread' of a distribution. But what is a covariance matrix?

- A square array of numbers: not very meaningful on its own.
- A **directional variance**: a recipe for mapping a unit vector $\hat{\mathbf{a}}$ to the variance of $\mathbf{x}$ in the direction of $\hat{\mathbf{a}}$:

$$\left\langle (\mathbf{x}\cdot\hat{\mathbf{a}})^2 \right\rangle = \left\langle (\hat{\mathbf{a}}^T\mathbf{x})(\mathbf{x}^T\hat{\mathbf{a}}) \right\rangle = \hat{\mathbf{a}}^T \left\langle \mathbf{x}\mathbf{x}^T \right\rangle \hat{\mathbf{a}} = \hat{\mathbf{a}}^T \mathbf{C} \hat{\mathbf{a}}$$

- An object describing the shape and orientation of an **ellipsoid** approximating the distribution. Surface of ellipsoid defined by $\mathbf{x}^T\mathbf{C}^{-1}\mathbf{x} = 1$. Lengths of each axis is the standard deviation of data in that direction. Directions given by eigenvectors of $\mathbf{C}$.



Summary: PCA and the covariance matrix

- The covariance matrix $\langle \mathbf{x}\mathbf{x}^T \rangle$ is symmetric and positive semi-definite, i.e., its eigenvalues are all non-negtive. This means that...
- ...the covariance matrix defines a 'generalised ellipsoid' in the n-dimensional data space.
- The directions of these axes are given by the n eigenvectors $\mathbf{u}_i$ of the covariance matrix. These vectors, ordered by eigenvalue, are also called the **principal components** of the data set, e.g. *first PC*, *the second PC*.
- The length of the ellipsoid in the direction of eigenvector $\mathbf{u}_i$ is proportional to $\sqrt{\lambda_i}$, where λ_i is the relevant eigenvalue: it is the standard deviation of the data in that direction.
- The m-dimensional **principal subspace** is obtained from the m longest axes of the ellipsoid, i.e. by 'flattening' the $n - m$ directions of least variance. Each datum $\mathbf{x}$ can then be encoded using the m coordinates $\xi_i = \mathbf{u}_i \cdot \mathbf{x}$.

Applications

Visualisation

Data compression

Avoid over-fitting (not the best idea)

Manifold learning

Choosing the number of eigenvalues

Rule of thumb: the projection explains 99% of the data variance

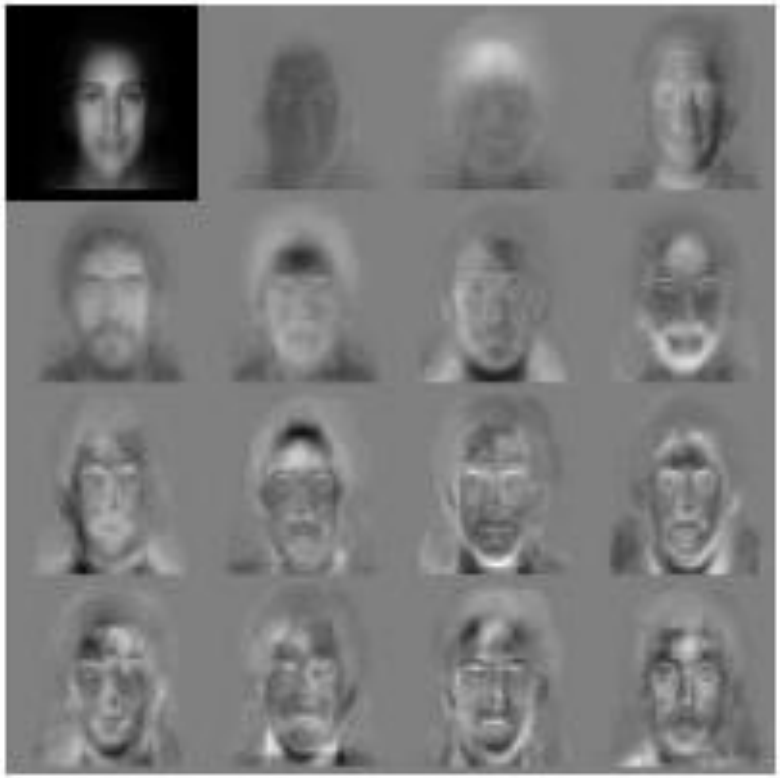
$$\sum_{i=1}^{k} \xi_i \; / \sum_{i=1}^{m} \xi_i > 0.99$$

The main use of PCA is for dimensionality reduction: it can help both humans and computers make sense of a high dimensional data set in terms of a few directions of variability.

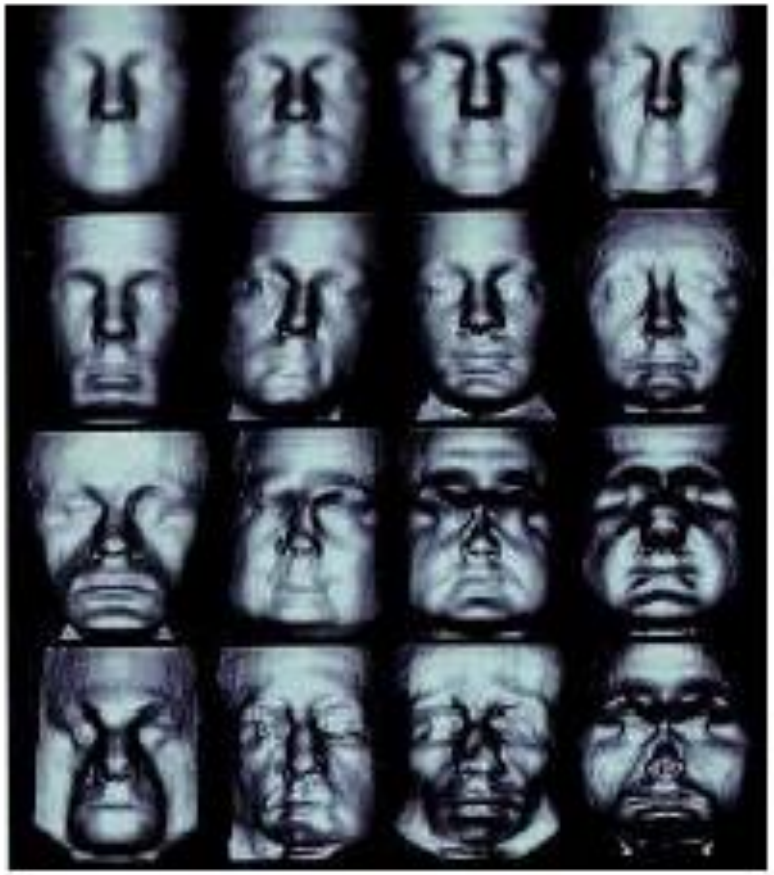
For example, these are some *eigenfaces* computed using PCA from different sets of face photos (gleaned from various websites). An $W \times H$ image can be treated as a vector in a $W \times H$ dimensional space. PCA can be used to capture the variability in the data set in a few variables, e.g. to simplify clustering or recognition.

First 8 principal components computed from 201 faces (by Vince Scheib, http://scheib.net/school/uncfaces/). They represent the main axes of intensity variation in the set of images. The first is the 'average' face. The second models for overall brightness. The next two account for left/right variations in lighting.

These are two sets of eigenfaces from the MIT Media Lab.



These are some 3D eigenfaces; sadly, no information about how they were computed and by whom was posted on the web page.



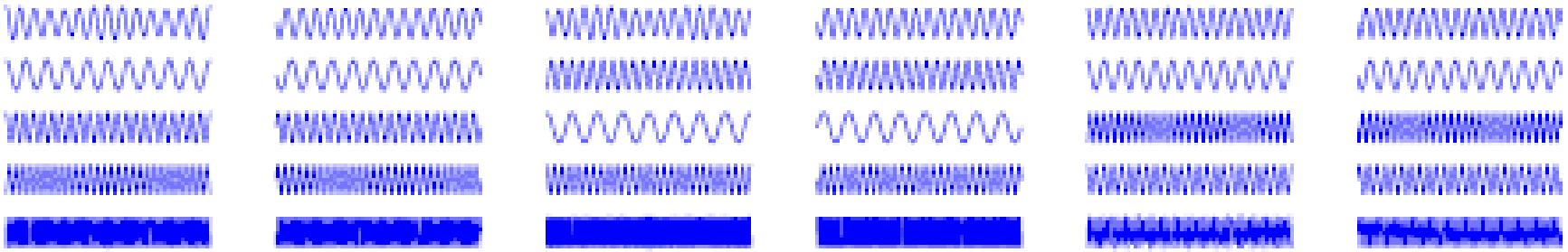
Audio and Image Processing Applications

- In 1 and 2-D signal processing, it is common practice to work with *frames* of data. Blocks of n audio samples or $L \times L$ patches of pixel values ($n = L^2$) are packed into n-dimensional vectors.

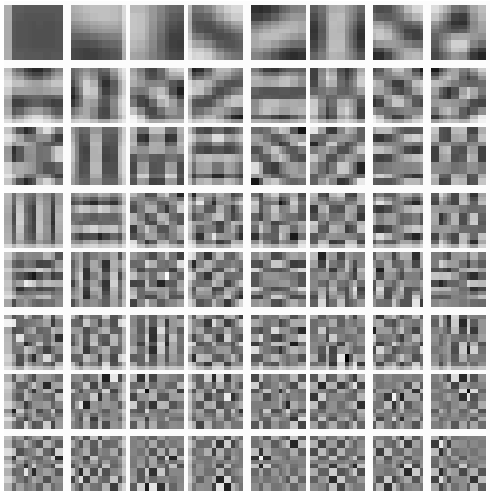
- Clearly, n could be very large (in the hundreds or thousands) so very often, PCA is used to reduce the dimensionality of the data to something more manageable.

- PCA is entirely driven by the data covariance. For audio frames and image patches, *shift invariance* means the covariance matrix has a special structure. This will be easier to visualise for audio data (1-D signal) but analogous arguments hold for image patches...

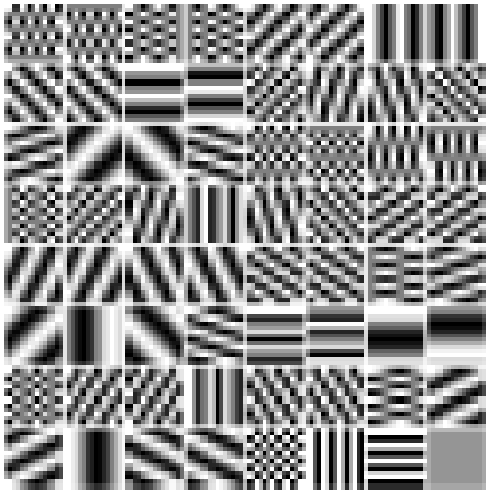
First 32 PCs deriveded from a short extract of music (arranged left to right):



Shift invariant image patches have an analogous covariance structure and hence PCs are similar to a 2D Fourier basis made up of plane waves.



PCA basis



Fourier basis

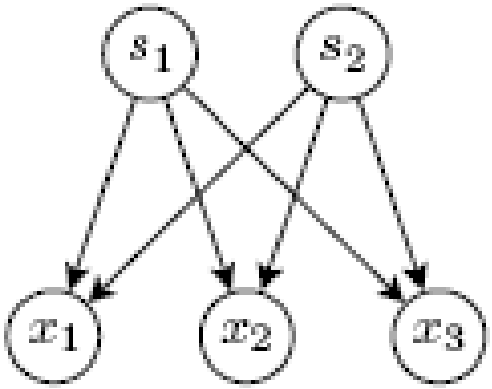
First few PCs are essentially the frequencies with the highest energy content, which tends to be the lower frequencies. Hence, for most sound and image ensembles, projection into the principal subspace is similar to low-pass filtering.

Probabilistic PCA: A Gaussian density model

So far, PCA defined as an analysis of variance/covariance structure, not as an explicit model of a probability density function. A probabilistic interpretation was put forward by Tipping and Bishop [5].

Characterisation by variance best suited to Gaussian data: no information about distribution is lost by keeping only 2nd order stats (these are *sufficient statistics* for Gaussian random variables.)

Hence, probabilistic PCA can be defined using a Gaussian **generative model**: $\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{n}$ where $\mathbf{s}$ is spherical Gaussian and of lower dimension than $\mathbf{x}$, and $\mathbf{n}$ is spherical Gaussian noise of unknown variance σ^2. Is a **latent variable model** because the s_i are not observed directly, but have to be estimated along with the matrix $\mathbf{A}$.



The parameters $\mathbf{A}$ and σ can be estimated using the maximum-likelihood method.

Maximum-likelihood estimation in probabilistic PCA

All the variables are Gaussian: $\mathbf{s} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, so we can write down all the relevant probability densities quite easily:

$$p(\mathbf{s}) = \frac{\exp \frac{1}{2}\|\mathbf{s}\|^2}{\sqrt{(2\pi)^M}} \qquad p(\mathbf{n}) = \frac{\exp \|\mathbf{n}\|^2/2\sigma^2}{\sigma^N \sqrt{(2\pi)^N}}$$

The observations $\mathbf{x}$ are also Gaussian, of zero-mean, with covariance

$$\begin{aligned} \langle \mathbf{x}\mathbf{x}^T \rangle &= \mathbf{A}\langle \mathbf{s}\mathbf{s}^T \rangle \mathbf{A}^T + \mathbf{A}\langle \mathbf{s}\mathbf{n}^T \rangle + \langle \mathbf{n}\mathbf{s}^T \rangle \mathbf{A}^T + \langle \mathbf{n}\mathbf{n}^T \rangle \\ &= \mathbf{A}\mathbf{A}^T + \sigma^2\mathbf{I} = \mathbf{C}. \end{aligned}$$

So, $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$, and average log-likelihood of a data set, $\log p([\mathbf{x}_1, \ldots, \mathbf{x}_K]; \mathbf{A})$ is

$$\mathcal{L} = -\tfrac{1}{2}\left\{ N \log 2\pi + \log \det \mathbf{C} + \tfrac{1}{K}\textstyle\sum_{k=1}^{K} \mathbf{x}_k^T \mathbf{C}^{-1} \mathbf{x}_k \right\}$$

Optimisation of this yields $\mathbf{A}$ = *a basis of the principle subspace*, (not necessarily orthogonal.) See [5] for more details.

Limitations of PCA (1)

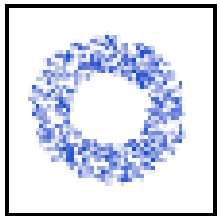
PCA is not scale invariant: if the input axes are differentially rescaled, the directions of greates variance will change and PCA will produce a different result.

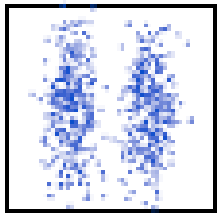
This is not too much of a problem when the input axes represent the same kind of quantity and have a natural relative scaling, e.g. *length of arm* vs. *length of leg*. But in some cases, there is no obvious natural scaling, e.g. *arm length* vs. *lifespan*.

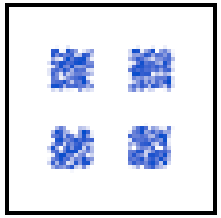
One solution is to normalise each input to unit variance, but little theoretical justification for this.

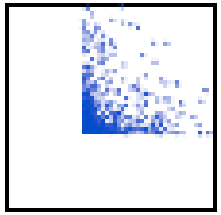
Generative model in probabilistic PCA provides an interpretation for scaling problem: pPCA assumes noise is spherical Gaussian, so we should scale the data so that *noise* has same variance in all directions.

Data summarised by *second order* statistics. Ok for Gaussian data, since these are completely determined by means and covariances. However, in the non-Gaussian world, very different data distributions can have the same covariance structure, for example the datasets to the right (after Fig. 10.6 in [2]) all have $\langle \mathbf{x}\mathbf{x}^T \rangle = \mathbf{I}$, and therefore have *no unique eigenvectors*, even though 3 of them clearly have 'special' diections. PCA is blind to these differences.









PCA measures 'interestingness' of projection by variance, which is not necessarily appropriate: small variance directions may carry useful information.

PCA is a linear transformation of the data: may not be suitable for dimensionality reduction when data lies in a curved manifold. In classification tasks, PCA cannot make linearly inseparable problems linearly separable.

What happens if we drop Gaussianity assumption? *See next lecture on ICA...*

References

[1] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, 1995.

[2] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley and Sons, New York, second edition, 2001.

[3] H. Hotelling. Analysis of complex statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[4] K. Pearson. On lines and planes of closest fit to a system of points in space. *Philosophical Magazine*, 2:559–572, 1901.

[5] M. Tipping and C. M. Bishop. Probabilistic principal component analysis. Technical Report NCRG/97/010, Neural Computing Research Group, Aston University, Birmingham, UK, September 1997.

See Sections 3.8.1 and 10.13.1 in Duda et al. [2]. See Appendix E in Bishop .