Dimension Reduction

David M. Blei

April 23, 2012

1 Basic idea

- Goal: Compute a reduced representation of data from p-dimensional to q-dimensional, where $q < p$.

$$\langle x_1, \ldots, x_p \rangle \rightarrow \langle z_1, \ldots, z_q \rangle \tag{1}$$

- We want to do this in such a way that we still have an idea of what the data are. (Making this more precise is what the various approaches to dimension reduction do.)

- In general, dimension reduction finds a set of **components** in p space and reduces each data point x_i to a q-dimensional vector of **weights** z_q which describe how data point i relates to each dimension.

- There are many reasons for performing dimension reduction
  - Store large data sets by storing a reduced representation
  - Explore the data, e.g., when $q = 2$ we can plot high dimensional data.
  - Generalize data, by computing regressions and predictors about the reduced representation. If p is correlated (e.g., text) than dimension reduction is an alternative to sparse prediction.
  - Generalize data by filling in missing values of the row x. For example, in collaborative filtering x represents the items a user liked. Reducing her dimension and representing components as weights for each item lets us predict what items she might like.

- Collaborative filtering is a (new) traditional example of dimension reduction.
  - Draw the matrix of users by items
  - Include ratings for some users
  - Describe components as weights for each item

  - Describe representing each user as a q dimensional vector
  - This lets us predict items

- We have seen this in a sense with k-means and mixture modeling
  - k-means: each data point represented with an integer
  - mixtures: each data point is a distribution over components

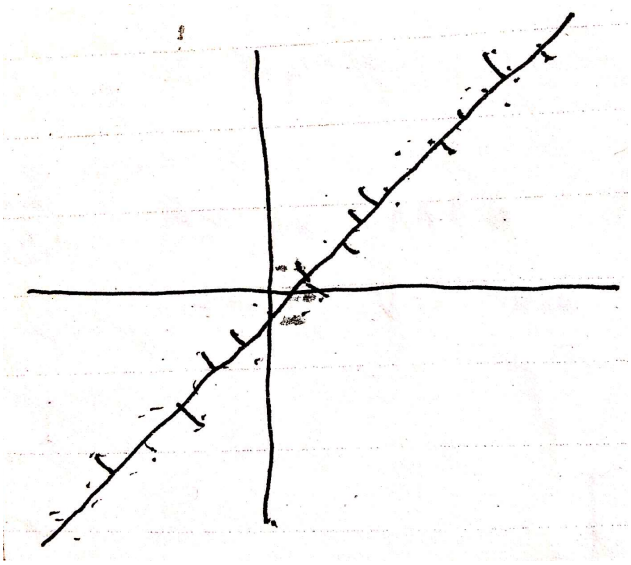
  Here we compress to the reals.

- (Show examples here.)

2 Principal component analysis

- A workhorse of dimension reduction methods. Invented multiple times in the last century: Pearson (1901), Hotelling (1933).

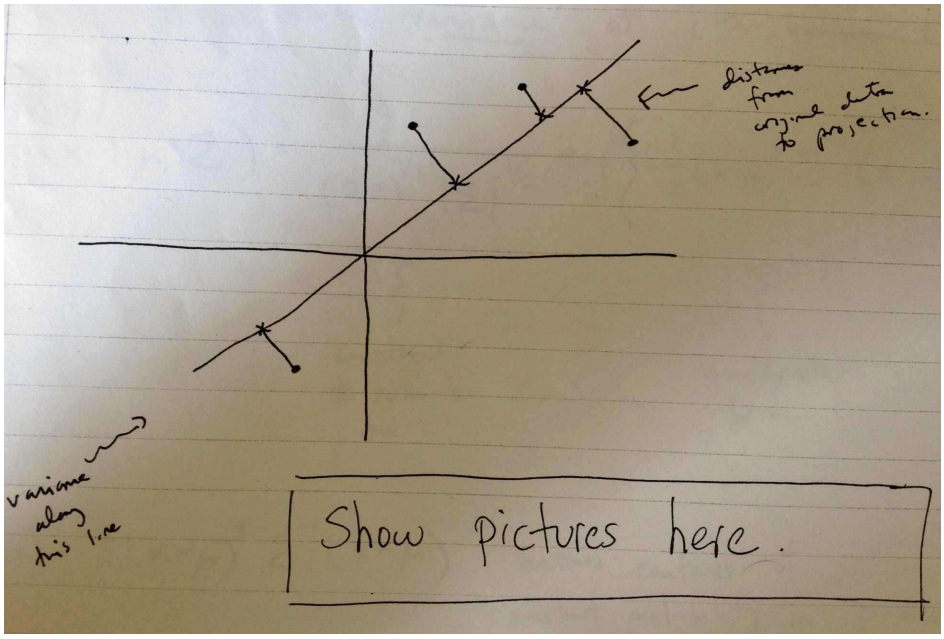
- Idea: Project data on a lower dimensional subspace in the original data space.



  - Draw a line (or manifold) through the subspace
  - Represent each data point as its projection on the line. (Notice: dimension reduction.)

- What is the free parameter? Where to draw the line.

- In PCA, there are three ways of thinking about defining the subspace.



  - Maximimize the variance of the projection along each of the q components. (Show this on the example graph. Confirm intuitively that tilting the line in any direction decreases the variance.) Hotelling (1933)
  - Minimize the **reconstruction error**, i.e., the distance between the original data and its "estimate" in the low dimensional subspace. Pearson (1901)
  - As the MLE of a parameter in a latent variable probabilistic model. Bishop (1996); Roweis (1998); Schapire (2001)

- We will focus on the third perspective, because it is illuminating for thinking about how PCA generalizes. (It does not lead to the best algorithms, necessarily.)

- First we will discuss the multivariate Gaussian.

3 The multivariate Gaussian

- A Gaussian for p dimensional vectors. The parameters are:
  - μ: The mean, a $p \times 1$ vector.
  - Σ: The covariance matrix, a $p \times p$ positive definite symmetric matrix.

- Each component of Σ is a covariance,

$$\sigma_{ij} = \mathrm{E}[X_i X_j] - \mathrm{E}[X_i]\mathrm{E}[X_j]. \tag{2}$$

  And note that the variances of each component lie along the diagonal

- The density is

$$p(x\,|\,\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right\}. \tag{3}$$

  - The first term is a constant to ensure that it integrates to one
  - The second term is a quadratic form

- The function $f(x) = (1/2)(x-\mu)^\top \Sigma^{-1}(x-\mu)$ defines contours of equal probability.

- (Show examples here.)

3.1 MLE of a multivariate Gaussian

The data are $\{x_1, \ldots, x_N\}$, where x_i is a p vector observation. The MLEs are

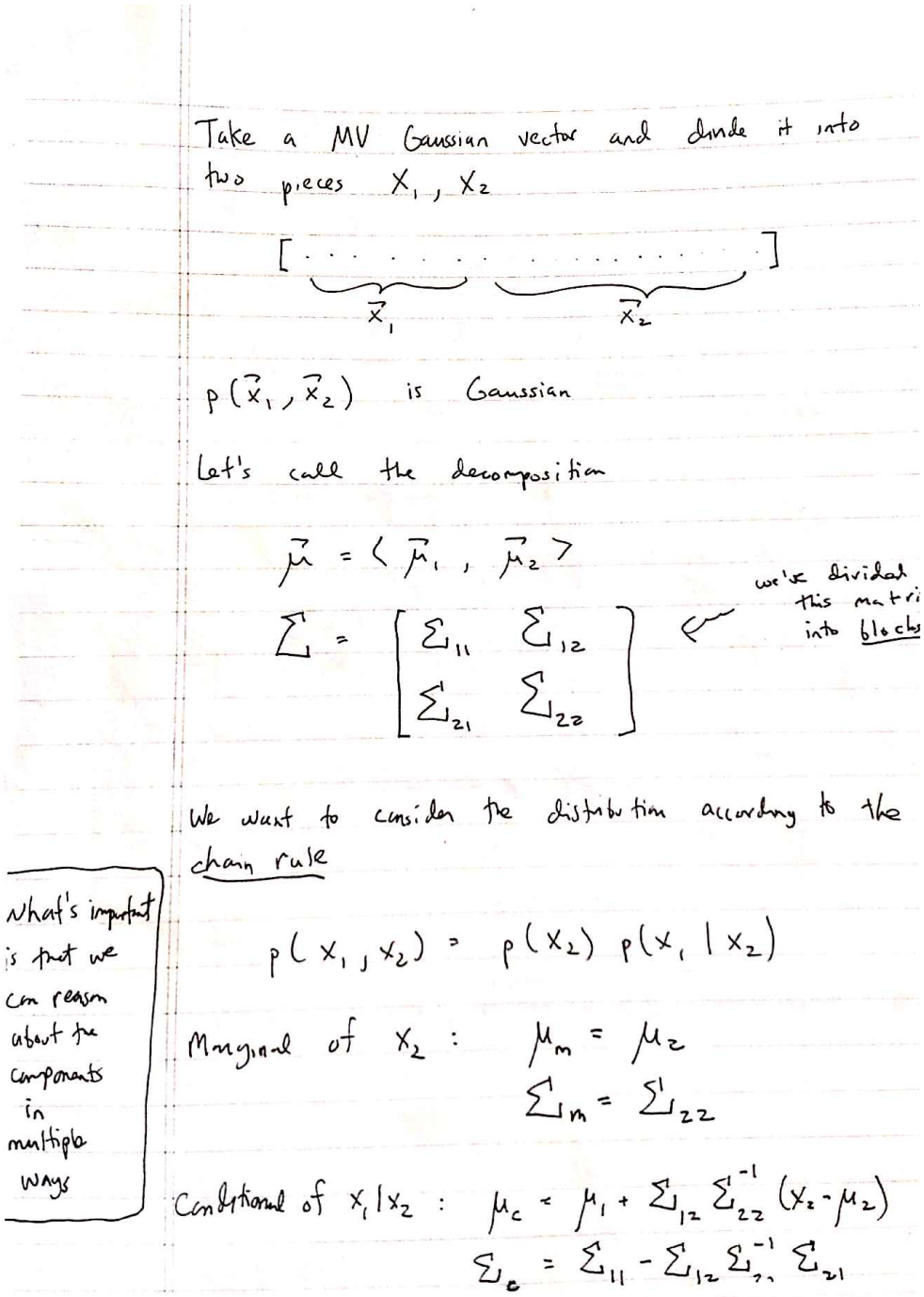
$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{4}$$

and

$$\hat{\Sigma} = \frac{1}{N}\sum_{i=1}^{N} (x_i - \hat{\mu})(x_i - \hat{\mu})^\top. \tag{5}$$

These generalize the 1-dimensional case.

3.2 Marginals and conditionals



4 Probabilistic PCA and factor analysis

- For the variance and reconstruction error approaches, we find a solution to the PCA problem via taking derivatives and setting them to zero. The solution relates to the eigenvalues

(and vectors) of the empirical covariance matrix.

- We will focus on the probabilistic interpretation, which relates especially to the reconstruction error approach. (As we've seen before, mean squared error based metrics can often be reinterpreted as with Gaussian distributions.)

- In probabilistic PCA,

$$\vec{z}_i \sim \mathcal{N}_q(0, I) \tag{6}$$
$$\vec{x}_i \sim \mathcal{N}_p(\mu + \Lambda z_i, I\psi) \tag{7}$$

  Notes:

  - μ is a p-vector
  - Λ is a $p \times q$ matrix
  - ψ is a scalar

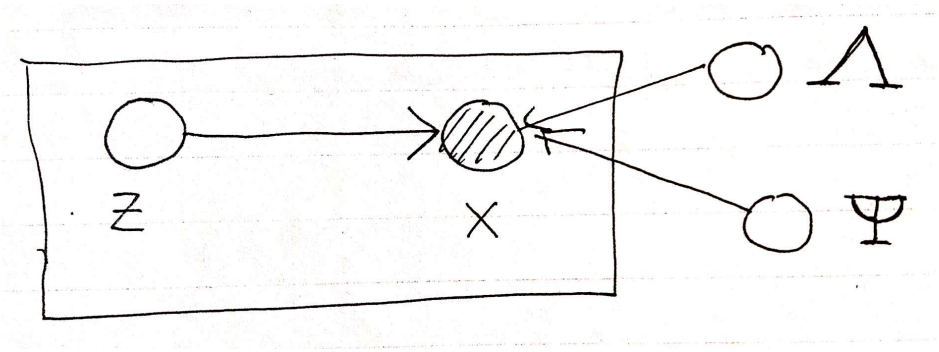
- As usual, assume that $\mu = 0$ (i.e., center the data). Then

$$\vec{z}_i \sim \mathcal{N}_q(0, I) \tag{8}$$
$$\vec{x}_i \sim \mathcal{N}_p(\Lambda\vec{z}_i, I\psi) \tag{9}$$

- In **factor analysis**, the only difference is that the data-level covariance is diagnoal with possibly difference variances along the components,

$$\vec{z}_i \sim \mathcal{N}_q(0, I) \tag{10}$$
$$\vec{x}_i \sim \mathcal{N}_p(\Lambda\vec{z}_i, \Psi) \tag{11}$$

- Graphical model:



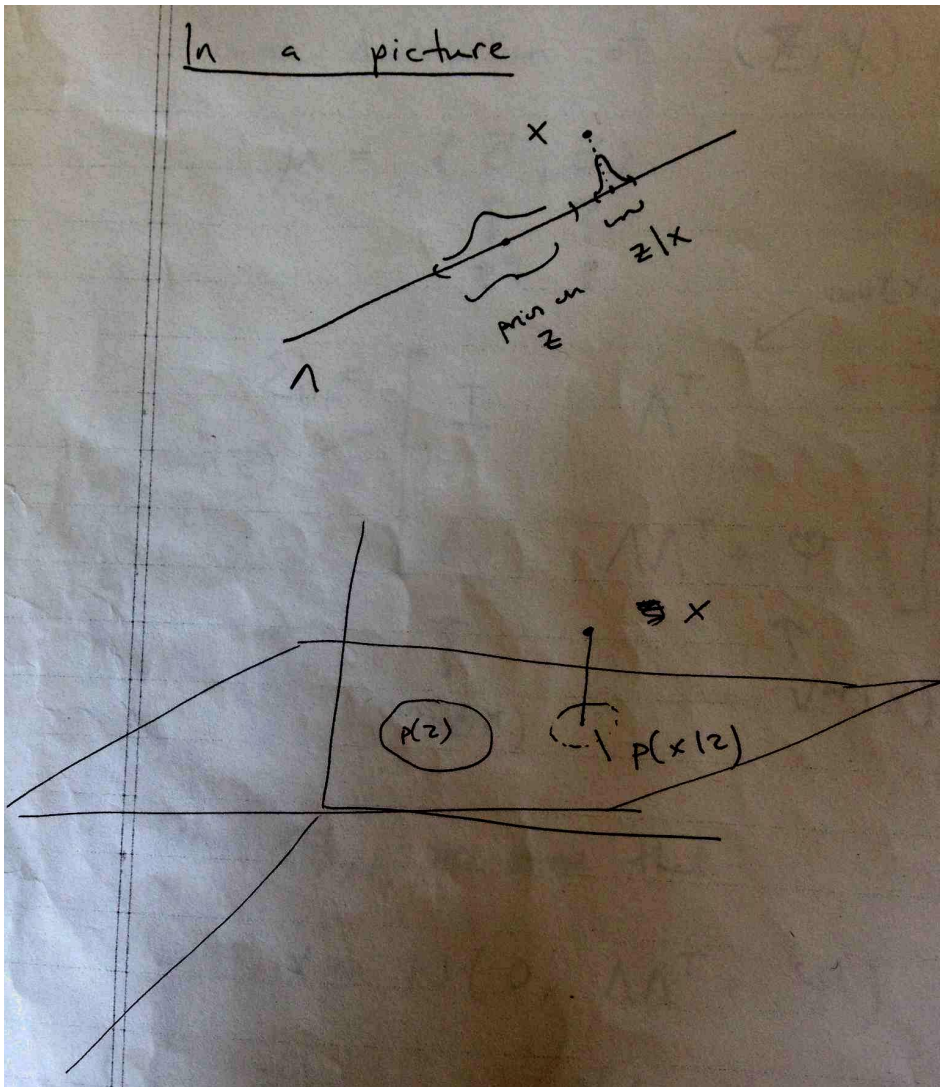
- In these models z_i is a "representation" of x_i

$$\mathrm{E}[\vec{X}_i \,|\, \vec{z}_i] = z_{i1}\vec{\lambda}_1 + z_{i2}\vec{\lambda}_2 + \ldots + z_{iq}\vec{\lambda}_q \tag{12}$$

  where $\vec{\lambda}_k$ is a column of Λ.

- The idea is that the $\vec{z}_i$ reflects common sources of variation amongst the datad, accounting for correlation between the data. The uncorrelated noise around the Gaussian picks up the variation that is not accounted for by the latent variable.

- Show intuition pictures



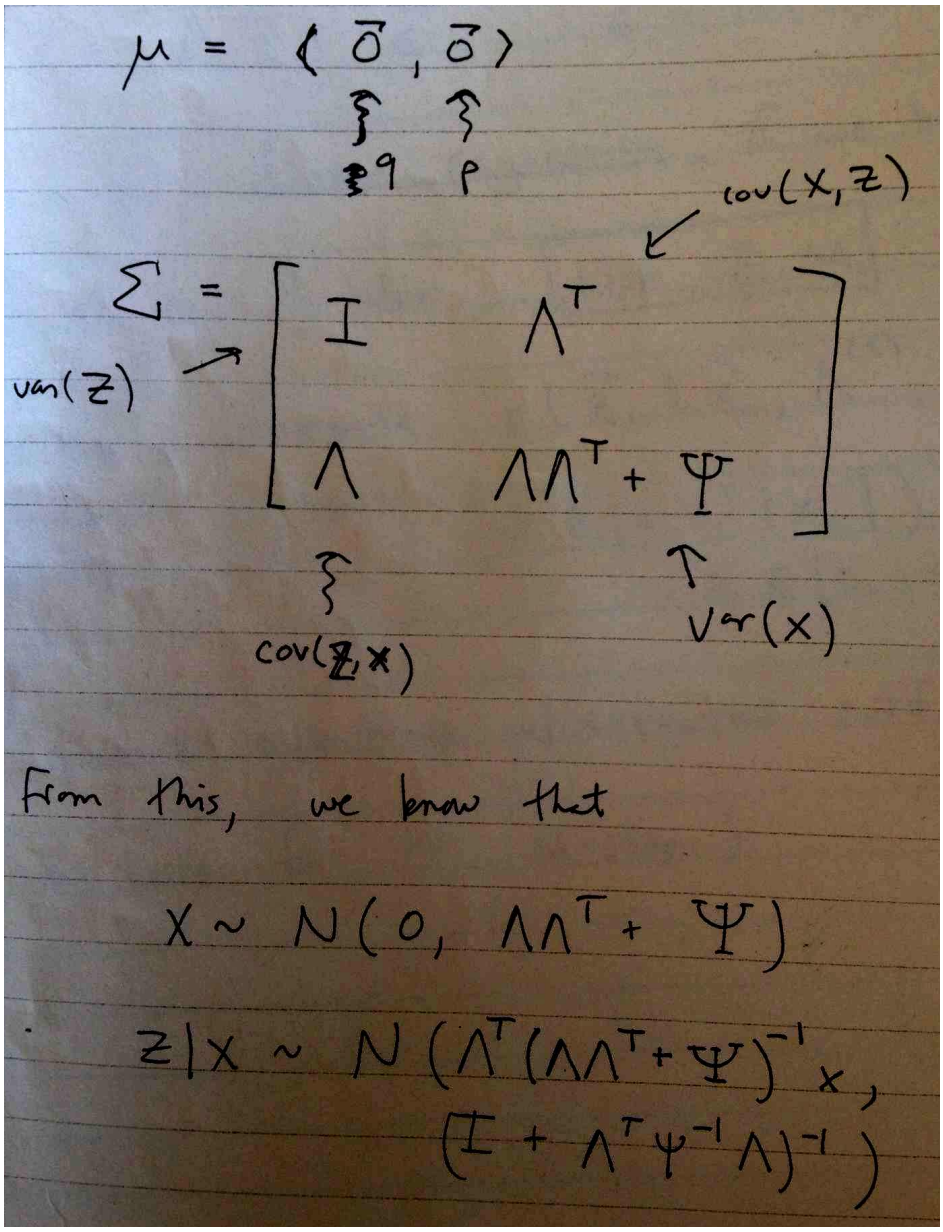
  - The plane is the parameter
  - The posterior expectation gives a low dimensional representation

5 Fitting probabilistic PCA/FA with EM

- We can fit these models with EM.
  - In the E-step we compute the posterior weights, $p(\vec{z}_i \,|\, \vec{x}_i, \Lambda, \Psi)$.

- In the M-step we re-estimate the components.

- Key insight for deriving EM: The joint distribution of (Z, X) is a $p+q$ dimensional Gaussian



  - The mean $\vec{\mu} = \langle \vec{0}, \vec{0} \rangle$.
  - The covariance Σ is a matrix
  - This gives us the E-step.

- For the M-step, notice that

$$\vec{x}_i = \Lambda \vec{z}_i + \vec{\epsilon}_i, \tag{13}$$

  where $\epsilon \sim \mathcal{N}_p(0, \Psi)$

- This is a **linear regression** where
  - The response is $\vec{x}$
  - The covariates are $\vec{z}$
  - The coefficients are Λ

- So, the M-step is like a posterior expected solution to linear regression

$$\Lambda^{(t+1)} = \left(\sum_{n=1}^{N} \mathrm{E}[\vec{Z}_n \vec{Z}_n{}^\top \mid \vec{x}_n] \right)^{-1} \left(\sum_{n=1}^{N} \mathrm{E}[\vec{Z}_n \mid \vec{x}_n]^\top \vec{x}_n \right). \tag{14}$$

- These are like the normal equations with expected sufficient statistics.

6 Kalman filter

- Application area: Tracking
  - (Draw a picture)
  - You are tracking items on a radar
  - You observe noisy estimates of the location of objects
  - You want to predict where they are going.

- More generally
  - You observe a number of measurements about an object
  - You know how those measurements relate to its position (ideally)
  - You want to track the object through space

- The Kalman filter—published in 1960 but probably secretly developed before that—solves this problem.

- The idea is that
  - We have a sequence of continuous observations $x_1, \ldots, x_t$.
  - Each is a noisy estimate of a latent position $z_1, \ldots, z_t$.
  - The latent position at time t depends on the latent position at time $t-1$.

- Two components

  - How the object moves through time
  - How the objects position relates to your measurements about it

- Old and new applications
  - Missiles (e.g., Tomahawk missile)
  - Autopilot (e.g., in submarines, oil tankers, space shuttle)
  - Financial time series
  - Touch screens, e.g., for character recognition
  - GPS and routing
  - Ratings over time (different from "position")

- (Notice: same algorithms for Tomahawk missile are used to predict movies.)

- The graphical model is: (draw graphical model)

- The generative process is:
  - Draw $z_t \mid z_{t-1} \sim \mathcal{N}(Az_t, I\sigma^2)$
  - Draw $x_t \mid z_t \sim \mathcal{N}(\Lambda z_t, \Psi)$

- Notice the similarities to HMMs
  - It has the same graphical model structure. What is the difference?
  - The z_t are continuous vectors instead of "states" (i.e., indicator vectors)

- Notice the similarity to factor analysis
  - Conditioned on z_t, x_t is drawn as in an FA model
  - The difference is that the weights are sequential
  - Note: immediate application to collaborative filtering
  - HMMs to mixture models is like factor analysis to Kalman filters
  - (Don't take that for granted—these models were all developed independently!)

- For those of you in ORFE
  - The latent vectors follow an AR model
  - All the innovations in AR modeling can be applied

- The parameters are
  - How the weights drift: variance and mean (or AR parameters)
  - The components: subspace and variances

- The filtering problem is $p(z_t \mid x_1, \ldots, x_{t-1})$—where is the duck going to be next?

- The smoothing problem is $p(z_1, \ldots, z_t \mid x_1, \ldots, x_t)$—what is the duck's trajectory?

- These are both solveable using algorithms related to HMM algorithms. Further, we can fit the components and time series parameters with EM.

7 Nonnegative matrix factorization

- The idea is to do matrix factorization, but to constrain the factors and/or weights to be non-negative

- This gives a "parts-based" representation. The idea is that negative values cannot posit factors. Run on faces, this recovers things like eyes and noses.

- Contrast with "holistic" approaches—a data point can exhibit a negative factor and a factor can exhibit negative features.

- Other applications
  - Gene profiles
  - Graphics ("Light cannot be negative" –Adam F.)
  - Computer vision
  - Natural language processing
  - Collaborative filtering

- Rant: The whole biological plausibility argument is fantasy—ignore it. (And you can probably ignore most biological plausibility arguments that you find, unless they are backed up with experimental data that tries to verify the low level mechanical workings.)

- The simple idea is to optimize

$$F = \sum_{i=1}^{N} \sum_{j=1}^{p} x_{ij} \log(z_i^\top \lambda_j) - z_i \lambda_j \tag{15}$$

where z_i are the weights for data point i and λ_j are the q component values at the dimension j. (Draw a matrix picture.)

- This is like replacing the Gaussian likelihood with a Poisson likelihood (appropriate for integers) and Poisson parameters. The weights z_i are constrained to be positive; the components Λ are also positive.

- The algorithm is nearly the same as EM.