Machine Learning for Signal Processing
Applications of Linear Gaussian Models

Class 15. 12 Nov 2015

Instructor: Bhiksha Raj
Recap: MAP Estimators

• MAP (Maximum A Posteriori): Find a “best guess” for $y$ (statistically), given known $x$

$$y = \text{argmax}_y P(Y|x)$$
Conditional Probability of $y \mid x$

$$P(y \mid x) = N(\mu_y + C_{yx} C_{xx}^{-1} (x - \mu_x), C_{yy} - C_{yx}^T C_{xx}^{-1} C_{xy})$$

$$E_{y \mid x}[y] = \mu_{y \mid x} = \mu_y + C_{yx} C_{xx}^{-1} (x - \mu_x)$$

$$\text{Var}(y \mid x) = C_{yy} - C_{xy}^T C_{xx}^{-1} C_{xy}$$

- The conditional probability of $y$ given $x$ is also Gaussian
  - The slice in the figure is Gaussian
- The mean of this Gaussian is a function of $x$
- The variance of $y$ reduces if $x$ is known
  - Uncertainty is reduced
Gaussians and more Gaussians...

• Linear Gaussian Models..

• PCA to develop the idea of LGM
A Brief Recap

• Principal component analysis: Find the $K$ bases that best explain the given data

• Find $\mathbf{B}$ and $\mathbf{C}$ such that the difference between $\mathbf{D}$ and $\mathbf{BC}$ is minimum
  – While constraining that the columns of $\mathbf{B}$ are orthonormal
Learning PCA

• For the given data: find the K-dimensional subspace such that it captures most of the variance in the data
  – Variance in remaining subspace is minimal
A Statistical Formulation of PCA

\[ x = Vw + e \]

\[ w \sim N(0, B) \]

\[ e \sim N(0, E) \]

- \( x \) is a random variable generated according to a linear relation
- \( w \) is drawn from an K-dimensional Gaussian with diagonal covariance
- \( e \) is drawn from a 0-mean (D-K)-rank D-dimensional Gaussian
- Estimate \( V \) (and \( B \)) given examples of \( x \)

Error is at 90° to the eigenface
Linear Gaussian Models!!

\[ x = Vw + e \]

\[ w \sim N(0, B) \]

\[ e \sim N(0, E) \]

• \( x \) is a random variable generated according to a linear relation
• \( w \) is drawn from a Gaussian
• \( e \) is drawn from a 0-mean Gaussian
• Estimate \( V \) given examples of \( x \)
  - In the process also estimate \( B \) and \( E \)
Estimating the variables of the model

\[ x = \mu + Vw + e \]

\[ w \sim N(0, I) \]
\[ e \sim N(0, E) \]

\[ x \sim \mathcal{N}(\mu, VV^T + E) \]

- Estimating the variables of the LGM is equivalent to estimating \( P(x) \)
- The variables are \( \mu, V, \) and \( E \)
The Maximum Likelihood Estimate

\[ x \sim N(\mu, VV^T + E) \]

• Given training set \( x_1, x_2, .. x_N \), find \( \mu, V, E \)

• The ML estimate of \( \mu \) does not depend on the covariance of the Gaussian

\[ \mu = \frac{1}{N} \sum_i x_i \]
Estimating the variables of the LGM is equivalent to estimating $P(x)$. The variables are $V$, and $E$. 

$$x = Vw + e$$

$$w \sim N(0, I)$$

$$e \sim N(0, E)$$

$$x \sim N(0, VV^T + E)$$
LGM: The complete EM algorithm

- Initialize $V$ and $E$
- E step:
  \[ E_{w|x_i}[w] = V^T (VV^T + E)^{-1} x_i \]
  \[ E_{w|x_i}[ww^T] = I - V^T (VV^T + E)^{-1} V + E_{w|x_i}[w]E_{w|x_i}[w]^T \]
- M step:
  \[ V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1} \]
  \[ E = \frac{1}{N} \sum_i x_i x_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w] x_i^T \]
So what have we achieved

- Employed a complicated EM algorithm to learn a *Gaussian* PDF for a variable x
- What have we gained???

- Example uses:
  - PCA
    - Sensible PCA
    - EM algorithms for PCA
  - Factor Analysis
    - FA for feature extraction
LGMs: Application 1

Learning principal components

\[ x = Vw + e \]

\[ w \sim N(0, I) \]

\[ e \sim N(0, E) \]

- Find directions that capture most of the variation in the data
- Error is orthogonal to principal directions
  - \( V^T e = 0 \); \( e^T V = 0 \)
Some Observations: 1

\[ x = Vw + e \]

\[ e \sim N(0, E) \]

\[ E = E[e e^T] \]

\[ V^T E = E[V^T e e^T] = E[0 e^T] = 0 \]

- The covariance \( E \) of \( e \) is orthogonal to \( V \)
Observation 2

\[ V^T E = 0 \]

\[ V^T (VV^T + E)^{-1} = (V^T V)^{-1} V^T \]

**Proof**

\[ V^T (VV^T + E)^{-1} (VV^T + E) = (V^T V)^{-1} V^T (VV^T + E) \]

\[ V^T = (V^T V)^{-1} V^T VV^T + (V^T V)^{-1} V^T E \]

\[ V^T = IV^T + (V^T V)^{-1} 0 \]

\[ V^T = V^T \]
Observation 3

\[ V^T E = 0 \]

\[ V^T (VV^T + E)^{-1} = (V^T V)^{-1} V^T \]

\[ = \text{pinv}(V) \]
LGM: The complete EM algorithm

\[ x = Vw + e \quad X \approx VW \]

- Initialize \( V \) and \( E \)
- E step:
  \[ E_{w|x_i}[w] = V^T (VV^T + E)^{-1} x_i \]
  \[ E_{w|x_i}[ww^T] = I - V^T (VV^T + E)^{-1} V + E_{w|x_i}[w]E_{w|x_i}[w]^T \]

- M step:
  \[ V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1} \]
  \[ E = \frac{1}{N} \sum_i x_ix_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T \]
LGM: The complete EM algorithm

\[ x = Vw + e \quad X \approx VW \]

- Initialize \( V \) and \( E \)
- E step:
  \[ E_{w|x_i}[w] = V^T(VV^T + E)^{-1}x_i \]
  \[ E_{w|x_i}[ww^T] = I - V^T(VV^T + E)^{-1}V + E_{w|x_i}[w]E_{w|x_i}[w]^T \]
- M step:
  \[ V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1} \]
  \[ E = \frac{1}{N} \sum_i x_i x_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T \]
LGM: The complete EM algorithm

\[ x = Vw + e \quad X \approx VW \]

- Initialize \( V \) and \( E \)
- E step:
  \[ w_i = V^T (VV^T + E)^{-1} x_i = \text{pinv}(V)x_i \]

\[ E_{w|x_i}[ww^T] = I - V^T (VV^T + E)^{-1} V + E_{w|x_i}[w]E_{w|x_i}[w]^T \]

- M step:
  \[ V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1} \]

\[ E = \frac{1}{N} \sum_i x_i x_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T \]
LGM: The complete EM algorithm

\[ x = Vw + e \quad X \approx VW \]

- Initialize \( V \) and \( E \)
- E step:
  \[ w_i = \text{pinv}(V)x_i \]

\[
E_{w|x_i}[ww^T] = I - V^T(VV^T + E)^{-1}V + E_{w|x_i}[w]E_{w|x_i}[w]^T
\]

- M step:
  \[
V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1}
\]

\[
E = \frac{1}{N} \sum_i x_ix_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T
\]
LGM: The complete EM algorithm

- Initialize $V$ and $E$
- E step:
  \[ w_i = \text{pinv}(V)x_i \]
  \[ W = \text{pinv}(V)X \]
- M step:
  \[ E_{w|x_i}[ww^T] = I - V^T(VV^T + E)^{-1}V + E_{w|x_i}[w]E_{w|x_i}[w]^T \]
  \[ V = \left( \sum_i x_i E_{w|x_i}[w^T] \right)^{-1} \left( \sum_i E_{w|x_i}[ww^T] \right) \]
  \[ E = \frac{1}{N} \sum_i x_i x_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T \]

\[ X \approx VW \]
LGM: The complete EM algorithm

- Initialize $V$ and $E$
- E step:
  \[ w_i = \text{pinv}(V)x_i \]
  \[ W = \text{pinv}(V)X \]
  \[
  E_{w|x_i}[ww^T] = I - V^T (VV^T + E)^{-1} V + E_{w|x_i}[w]E_{w|x_i}[w]^T
  \]
- M step:
  \[
  V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1}
  \]
  \[
  E = \frac{1}{N} \sum_i x_i x_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T
  \]
EM for PCA

• Initialize $V$ and $E$

• E step:

$w_i = \text{pinv}(V)x_i$

$W = \text{pinv}(V)X$

$E_{w|x_i}[ww^T] = I - V^T(VV^T + E)^{-1}V + E_{w|x_i}[w]E_{w|x_i}[w]^T$

• M step:

$V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1}$

$E = \frac{1}{N} \sum_i x_ix_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T$

$X \approx VW$
EM for PCA

- Initialize \( V \) and \( E \)
- E step:
  \[
  w_i = \text{pinv}(V)x_i
  \]
  \[
  W = \text{pinv}(V)X
  \]
- M step:
  \[
  E_{w|x_i}[ww^T] = I - V^T(VV^T + E)^{-1}V + E_{w|x_i}[w]E_{w|x_i}[w]^T
  \]
  \[
  V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1}
  \]
  \[
  E = \frac{1}{N} \sum_i x_ix_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T
  \]
EM for PCA

- Initialize $V$ and $E$

- E step:
  \[ w_i = \text{pinv}(V)x_i \]
  \[ W = \text{pinv}(V)X \]

- M step:
  \[ E_{w|x_i}[ww^T] = I - V^T(VV^T + E)^{-1}V + E_{w|x_i}[w]E_{w|x_i}[w]^T \]
  \[ V = \left( \sum_i x_iE_{w|x_i}[w^T] \right)\left( \sum_i E_{w|x_i}[ww^T] \right)^{-1} = XW^T(WW^T)^{-1} \]
  \[ E = \frac{1}{N} \sum_i x_ix_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T \]
EM for PCA

• Initialize $V$ and $E$

• E step:

$$w_i = \text{pinv}(V)x_i$$

$$W = \text{pinv}(V)X$$

$$E_{w|x_i}[ww^T] = I - V^T (VV^T + E)^{-1} V + E_{w|x_i}[w]E_{w|x_i}[w]^T$$

• M step:

$$V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1} = XW^T (WW^T)^{-1} = X\text{pinv}(W)$$

$$E = \frac{1}{N} \sum_i x_i x_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T$$
EM for PCA

- Initialize $V$ and $E$
- E step:
  \[
  w_i = \text{pinv}(V)x_i \quad \text{and} \quad W = \text{pinv}(V)X
  \]

- \[E_{w|x_i}[ww^T] = I - V^T(VV^T + E)^{-1}V + E_{w|x_i}[w]E_{w|x_i}[w]^T\]

- M step:
  \[V = X\text{pinv}(W)\]

\[
E = \frac{1}{N}\sum_i x_ix_i^T - \frac{1}{N}V\sum_i E_{w|x_i}[w]x_i^T
\]
EM for PCA

• Initialize $V$ and $E$

• E step:

$$W = \text{pinv}(V)X$$

$$E_{w|x_i}[ww^T] = I - V^T (VV^T + E)^{-1} V + E_{w|x_i}[w]E_{w|x_i}[w]^T$$

• M step:

$$V = X \text{pinv}(W)$$

$$E = \frac{1}{N} \sum_i x_ix_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T$$
EM for PCA

• Initialize $\mathbf{V}$ and $\mathbf{E}$

• E step:

$$\mathbf{W} = \text{pinv}(\mathbf{V}) \mathbf{X}$$

$$E_{w|x_i}[\mathbf{ww}^T] = I - \mathbf{V}^T (\mathbf{VV}^T + \mathbf{E})^{-1} \mathbf{V} - E_{w|x_i}[\mathbf{w}]E_{w|x_i}[\mathbf{w}]^T$$

• M step:

$$\mathbf{V} = \mathbf{X} \text{pinv}(\mathbf{W})$$

$$E = \frac{1}{N} \sum_i x_i^T x_i - \frac{1}{N} \mathbf{V} \sum_i E_{w|x_i}[\mathbf{w}] x_i^T$$

irrelevant
EM for PCA

• Initialize $V$

• Iterate

\[
W = \text{pinv}(V)X
\]

\[
V = X \text{pinv}(W)
\]

• Note: $V$ will not be actual eigenvectors, but a set of bases in space spanned by principal eigenvectors
  – Additional decorrelation within PC space may be needed
Why EM PCA?

- Example: Computing eigenfaces
- Each face is 100x100 : 10000 dimensional
- But only 300 examples
  - \( X \) is 10000 x 300
- What is the size of the covariance matrix?
- What is its rank?

\[
XX^T \quad 10000 \times 10000
\]
PCA on illconditioned data

• Few instances of high-dimensional data
  – No. instances < dimensionality

• Covariance matrix is very large
  – Eigen decomposition is expensive
  – E.g. 1000000-dimensional data: Covariance has $10^{12}$ elements

• But the rank of the covariance is low
  – Only the no. of instances of data
Why EM PCA?

- Consequence of low rank $X$
  - The actual number of bases is limited to the rank of $X$

- Note actual size of $V$
  - Max number of columns = min(dimension, no. data points)
  - No. of columns = rank of $(XX^T)$

- Note size of $W$
  - Max number of rows = min(dimension, no. of data points)
Why EM PCA?

• If $X$ is high dimensional
  – Particularly if the number of vectors in $X$ is smaller than the dimensionality

• $\text{Pinv}(V)$ and $\text{pinv}(W)$ are efficient to compute
  – $V$ will have a max of 300 columns in the example
  – $W$ will have a max of 300 rows
PCA as an instance of LGM

• Viewing PCA as an instance of linear Gaussian models leads to EM solution
• Very effective in dealing with high-dimensional and/or data poor situations

• An aside: Another simpler solution for the same situation..
An Aside: The GRAM trick

- The number of non-zero Eigen values is no more than the length of the smallest “edge” of $X$
  - 300 in this case
- This leads to the “gram” trick..

- Assumption $XX^T$ is invertible: the instances are linearly independent
An Aside: The GRAM trick

- \(XX^T\) is large but \(X^TX\) is not

If \(X\) is 10000 x 300, 
\[XX^T = 10000 \times 10000\]

- Difficult to compute Eigen vectors of \(XX^T\)
- But easy to compute Eigen vectors of \(X^TX\)
The Gram Trick

• To compute principal vectors we Eigendecompose $XX^T$

$$\left(XX^T\right)\hat{E} = \hat{E}\Lambda$$

• Let us find the Eigen vectors of $X^TX$ instead

$$\left(X^TX\right)\hat{E} = \hat{E}\hat{\Lambda}$$

• Manipulating it slightly

Note that for a diagonal matrix:

$$\Lambda\Lambda^{-0.5} = \Lambda^{-0.5}\Lambda$$

$$X^TX\hat{E}\hat{\Lambda}^{-0.5} = \hat{E}\hat{\Lambda}^{-0.5}\hat{\Lambda}$$
The Gram Trick

• Eigendecompose $X^TX$ instead of $XX^T$

\[
(X^TX)E = \hat{E}\Lambda
\]

\[
X^TX\hat{E}\Lambda^{-0.5} = \hat{E}\Lambda^{-0.5}\Lambda
\]

\[
(XX^T)(\hat{X}\hat{E}\Lambda^{-0.5}) = (\hat{X}\hat{E}\Lambda^{-0.5})\Lambda
\]

• Letting: $\hat{X}\hat{E}\Lambda^{-0.5} = E$

\[
(XX^T)E = E\Lambda
\]

• $E$ is the matrix of Eigenvectors of $XX^T$!!!
The Gram Trick

• When X is low rank or $XX^T$ is too large:

  • Compute $X^TX$ instead
    – Will be manageable size
  
  • Perform Eigen Decomposition of $X^TX$

    $$(X^TX)^\hat{E} = \hat{E}\hat{\Lambda}$$

  • Compute Eigenvectors of $XX^T$ as

    $$X\hat{E}\hat{\Lambda}^{-0.5} = E$$

• These are the principal components of X
Why EM PCA

• Dimensionality / Rank has alternate potential solution
  – Gram Trick

• Other uses?
  – Noise
  – Incomplete data
PCA with noisy data

\[ x = Vw + e + n \]

\[ \mathbf{w} \sim \mathcal{N}(0, I) \]
\[ \mathbf{e} \sim \mathcal{N}(0, E) \]
\[ \mathbf{n} \sim \mathcal{N}(0, B) \]

- Error is orthogonal to principal directions
  - \( V^T e = 0; \ e^T V = 0 \)
- Noise is isotropic
  - \( B \) is diagonal
  - Noise is not orthogonal to either \( V \) or \( e \)
LGM: The complete EM algorithm

• Initialize $V$ and $E$

• E step:

\[
E_{w|x_i}[w] = V^T (VV^T + E)^{-1} x_i \]

\[
E_{w|x_i}[ww^T] = I - V^T (VV^T + E)^{-1} V + E_{w|x_i}[w]E_{w|x_i}[w]^T
\]

• M step:

\[
V = \left( \sum_i x_i E_{w|x_i}[w^T] \right) \left( \sum_i E_{w|x_i}[ww^T] \right)^{-1}
\]

\[
E = \frac{1}{N} \sum_i x_i x_i^T - \frac{1}{N} V \sum_i E_{w|x_i}[w]x_i^T
\]
PCA with Noisy Data

- Initialize $V$ and $B$
- E step: $\beta = V^T (VV^T + B)^{-1}$ $\quad W = \beta X$

$$C = NI - N\beta V + WW^T$$

- M step: $V = XW^T C^{-1}$

$$B = \frac{1}{N} \text{diag} \left( XX^T - VWX^T \right)$$
PCA with *Incomplete* Data

- How to compute principal directions when some components in your training data are missing?

- Eigen decomposition is not possible
  - Cannot compute correlation matrix with missing data
PCA with missing data

• How it goes

• Given : $X = \{X_c, X_m\}$
  – $X_m$ are missing components

1. Initialize: Initialize $X_m$

2. Build “complete” data $X = \{X_c, X_m\}$

3. PCA ($X = VW$): Estimate $V$
  – $V$ must have fewer bases than dimensions of $X$

4. $W = V^TX$

5. $\hat{X} = VW$

6. Select $X_m$ from $\hat{X}$

7. Return to 2
LGM for PCA

- Obviously many uses:
  - Ill-conditioned data
  - Noise
  - Missing data
  - Any combination of the above..
LGMs : Application 2
Learning with insufficient data

- The full covariance matrix of a Gaussian has $D^2$ terms
- Fully captures the relationships between variables
- Problem: Needs a lot of data to estimate robustly
• Assume the covariance is diagonal
  – Gaussian is aligned to axes: no correlation between dimensions
  – Covariance has only $D$ terms
• **Needs less data**
• **Problem: Model loses all information about correlation between dimensions**
Is There an Intermediate

• Capture the most important correlations
• But require less data

• Solution: Find the key subspaces in the data
  – Capture the complete correlations in these subspaces
  – Assume data is otherwise uncorrelated
Factor Analysis

\[ x = Vw + e \]

\[ w \sim N(0, I) \]
\[ e \sim N(0, E) \]

\[ x \sim N(0, VV^T + E) \]

- \( E \) is a full rank diagonal matrix
- \( V \) has \( K \) columns: \( K \)-dimensional subspace
  - We will capture all the correlations in the subspace represented by \( V \)
- Estimated covariance: Diagonal covariance \( E \) plus the covariance between dimensions in \( V \)
Factor Analysis

- Initialize $V$ and $E$

- **E step:**

  \[
  E_{wi|x_i}[w] = V^T (VV^T + E)^{-1} x_i
  \]

  \[
  E_{wi|x_i}[ww^T] = I - V^T (VV^T + E)^{-1} V + E_{wi|x_i}[w]E_{wi|x_i}[w]^T
  \]

- **M step:**

  \[
  V = \left( \sum_i x_i E_{wi|x_i}[w^T] \right) \left( \sum_i E_{wi|x_i}[ww^T] \right)^{-1}
  \]

  \[
  E = \frac{1}{N} \text{diag} \left( \sum_i x_i x_i^T - \frac{1}{N} V \sum_i E_{wi|x_i}[w] x_i^T \right)
  \]
FA Gaussian

- Will get a full covariance matrix
- But only estimate DK terms
- Data insufficiency less of a problem
The Factor Analysis Model

\[ \mathbf{x} = \mathbf{Vw} + \mathbf{e} \]

- Often used to learn distribution of data when we have insufficient data
- Often used in psychometrics
  - Underlying model: The actual systematic variations in the data are totally explained by a small number of “factors”
  - FA uncovers these factors

\[ \mathbf{w} \sim N(0, I) \]
\[ \mathbf{e} \sim N(0, \mathbf{E}) \]
FA: Example

• Hypothesis: there are two kinds of intelligence, "verbal" and "mathematical",
  – neither is directly observed.
  – Evidence sought from examination scores from each of 10 different academic fields of 1000 students.

• Solution: Find out if distribution is well explained by two factors
  – Hack: Attempt to relate factors to verbal and math IQ
FA, PCA etc.

\[ x = Vw + e \]

\[ w \sim N(0, I) \]
\[ e \sim N(0, E) \]

• Note: distinction between PCA and FA is only in the assumptions about \( e \)
• FA looks a lot like PCA with noise
• FA can also be performed with incomplete data
FA, PCA etc.

- **PCA**: Error is always at 90 degrees to the bases in \( \mathbf{V} \)
- **FA**: Error may be at any angle
- **PCA** used mainly to find *principal* directions that capture most of the variance
  - Bases in \( \mathbf{V} \) will be orthogonal to one another
- **FA** tries to capture most of the covariance
FA: A very successful use

• Voice biometrics: Speaker identification

• Given: Only a small amount of training data from a speaker, learn model for speaker
  – Use to verify speaker later

• Problem: Immense variation in ways people can speak
  – 15 minutes of training data totally insufficient!
Speaker Verification

- A model represents distribution of cepstral vectors for the speaker
- A second model represents everyone else (potential imposters)
- The cepstra computed from a test recording are “scored” against both models
  - Accept the speaker if the speaker model scores higher
Speaker Verification

• Problem: One typically has only a few seconds or minutes of training data from the speaker
• Hard to estimate speaker model
• Test data may be spoken differently, or come over a different channel, or in noise
  – Won't really match
Hypothesis

• Variations between different instances of the utterance spoken by the same speaker related to only a few factors
• Factors are common to all speakers
• Solution: Learn factors by analyzing many speakers
  – Use learned factors to predict variations for a given speaker
  – Can provide robust models for a speaker from very little data
Representing the Data: “super vectors”

• Convert recordings to a sequence of feature vectors
  – Cepstra
• Compute the probability distribution for the data
  – Modeled as a Gaussian mixture
• The data are represented by the parameters of the distribution
Representing the Data: “super vectors”

\[ P(X) = \sum_k w_k N(X; \mu_k, \Theta_k) \]

This “supervector” is the feature that represents the recording.
• Supervectors are obtained for each training speaker by adapting a “Universal background model” trained from large amounts of data.
Training the Factor Analyzer

\[ x = Vw + e \]

\[ w \sim N(0, I) \quad e \sim N(0, E) \]

- The supervectors are assumed to be the output of a linear Gaussian process.
- Use FA to estimate \( V \)
  - \( V \) are the factors that cause variations.
  - The *real* information is in the factor \( w \).
Training models for a speaker

- From training data: estimate the means for the speaker to conform to the factor analysis
  - Constrained estimation: requires much less data
- Use the estimated means as the distribution for the speaker
  - Solves data insufficiency problem
  - Also solves the problem of variations

\[ x = Vw_s + e \]
\[ w \sim N(0, I) \quad e \sim N(0, E) \]
Many other applications..

- Exploratory FA
- Confirmatory FA..