Latent Variable Models and Signal Separation

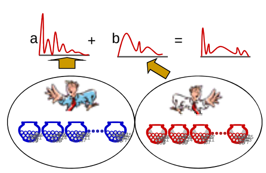
Class 11. 6 Oct 2011

Signal Separation from Monaural Recordings

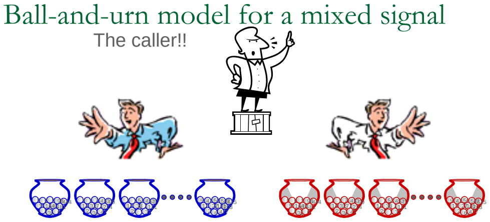
- The problem:
  - Multiple sources are producing sound simultaneously
  - The combined signals are recorded over a single microphone
  - The goal is to selectively separate out the signal for a target source in the mixture
    - Or at least to enhance the signals from a selected source

Problem Specification

- The mixed signal contains components from multiple sources
- Each source has its own “bases”
- In each frame
  - Each source draws from its own collection of bases to compose a spectrum
    - Bases are selected with a frame specific mixture weight
  - The overall spectrum is a mixture of the spectra of individual sources
    - I.e. a histogram combining draws from both sources
- Underlying model: Spectra are histograms over frequencies



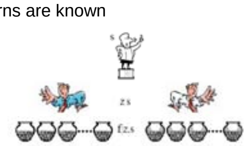
Ball-and-urn model for a mixed signal



- Each sound source is represented by its own picker and urns
  - Urns represent the distinctive spectral structures for that source
  - **Assumed to be known beforehand** (learned from some separate training data)
- The caller selects a picker at random
  - The picker selects an urn randomly and draws a ball
  - The caller calls out the frequency on the ball
- A spectrum is a histogram of frequencies called out
  - The total number of draws of any frequency includes contributions from *both* sources

Separating the sources

- Goal: Estimate number of draws from each source
  - The probability distribution for the mixed signal is a linear combination of the distribution of the individual sources
  - The individual distributions are mixture multinomials
  - And the urns are known

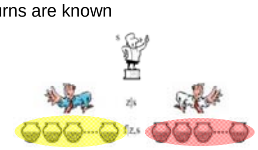


$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

$$\boxed{P_t(f) = P_t(s_1)\sum_z P_t(z \mid s_1)P(f \mid z, s_1) + P_t(s_2)\sum_z P_t(z \mid s_1)P(f \mid z, s_2)}$$

Separating the sources

- Goal: Estimate number of draws from each source
  - The probability distribution for the mixed signal is a linear combination of the distribution of the individual sources
  - The individual distributions are mixture multinomials
  - And the urns are known

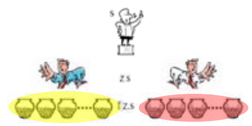


$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

$$\boxed{P_t(f) = P_t(s_1)\sum_z P_t(z \mid s_1)P(f \mid z, s_1) + P_t(s_2)\sum_z P_t(z \mid s_1)P(f \mid z, s_2)}$$

Separating the sources

- Goal: Estimate number of draws from each source
  - The probability distribution for the mixed signal is a linear combination of the distribution of the individual sources
  - The individual distributions are mixture multinomials
  - And the urns are known
  - **Estimate remaining terms using EM**



$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

$$P_t(f) = P_t(s_1)\sum_z P_t(z \mid s_1)P(f \mid z, s_1) + P_t(s_2)\sum_z P_t(z \mid s_1)P(f \mid z, s_2)$$

Algorithm

- For each frame:
  - Initialize $P_t(s)$
    - The fraction of balls obtained from source s
    - Alternately, the fraction of energy in that frame from source s
  - Initialize $P_t(z|s)$
    - The mixture weights of the urns in frame *t* for source s
  - Reestimate the above two iteratively
- Note: P(f|z,s) is not frame dependent
  - It is also not re-estimated
  - Since it is assumed to have been learned from separately obtained unmixed training data for the source

Iterative algorithm

- Iterative process:
  - Compute a posteriori probability of the combination of speaker s and the z[th] urn for each speaker for each f

$$P_t(s,z \mid f) = \frac{P_t(s)P_t(z \mid s)P(f \mid z,s)}{\sum_{s'} P_t(s')\sum_{z'} P_t(z' \mid s')P(f \mid z',s')}$$

  - Compute the a priori weight of speaker s

$$P_t(s) = \frac{\sum_z \sum_f P_t(s,z \mid f)S_t(f)}{\sum_{s'}\sum_{z'}\sum_f P_t(s',z' \mid f)S_t(f)}$$

  - Compute mixture weight of z[th] urn for speaker s

$$P_t(z \mid s) = \frac{\sum_f P_t(s,z \mid f)S_t(f)}{\sum_{z'}\sum_f P_t(s',z' \mid f)S_t(f)}$$

What is $P_t(s,z \mid f)$

- Compute how each ball (frequency) is split between the urns of the various sources
- The ball is first split between the sources

$$P_t(s \mid f) = \frac{P_t(s)}{\sum_{s'} P_t(s')}$$

- The fraction of the ball attributed to any source s is split between its urns:

$$P_t(z \mid s, f) = \frac{P_t(z \mid s)P(f \mid z,s)}{\sum_{z'} P_t(z' \mid s)P(f \mid z',s)}$$

- The portion attributed to any urn of any source is a product of the two

$$P_t(s,z \mid f) = \frac{P_t(s)P_t(z \mid s)P(f \mid z,s)}{\sum_{s'} P_t(s')\sum_{z'} P_t(z' \mid s')P(f \mid z',s')}$$

Reestimation

- The reestimate of source weights is simply the proportion of all balls that was attributed to the sources

$$P_t(s) = \frac{\sum_z \sum_f P_t(s,z \mid f)S_t(f)}{\sum_{s'}\sum_{z'}\sum_f P_t(s',z' \mid f)S_t(f)}$$

- The reestimate of mixture weights is the proportion of all balls attributed to each urn

$$P_t(z \mid s) = \frac{\sum_f P_t(s,z \mid f)S_t(f)}{\sum_{z'}\sum_f P_t(s',z' \mid f)S_t(f)}$$

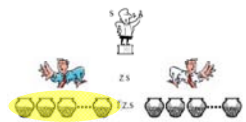
Separating the Sources

- For each frame:
- Given
  - $S_t(f)$ – The spectrum at frequency f of the mixed signal
- Estimate
  - $S_{t,i}(f)$ – The spectrum of the separated signal for the i-th source at frequency f
- A simple maximum a posteriori estimator

$$\hat{S}_{t,i}(f) = S_t(f)\sum_z P_t(z,s \mid f)$$

If we have only have bases for one source?

- Only the bases for one of the two sources is given
  - Or, more generally, for N-1 of N sources

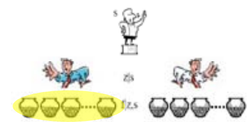


$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

$$\boxed{P_t(f) = P_t(s_1)\sum_z P_t(z \mid s_1)P(f \mid z, s_1) + P_t(s_2)\sum_z P_t(z \mid s_1)P(f \mid z, s_2)}$$

If we have only have bases for one source?

- Only the bases for one of the two sources is given
  - Or, more generally, for N-1 of N sources
  - The unknown bases for the remaining source must also be estimated!



$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

$$\boxed{P_t(f) = P_t(s_1)\sum_z P_t(z \mid s_1)P(f \mid z, s_1) + P_t(s_2)\sum_z P_t(z \mid s_1)P(f \mid z, s_2)}$$

Partial information: bases for one source unknown

- P(f|z,s) must be initialized for the additional source
- Estimation procedure now estimates bases along with mixture weights and source probabilities
  - From the ***mixed signal itself***
- The final separation is done as before

Iterative algorithm

- Iterative process:
  - Compute a posteriori probability of the combination of speaker s and the z^{th} urn for the speaker for each f

$$\boxed{P_t(s,z \mid f) = \frac{P_t(s)P_t(z \mid s)P(f \mid z,s)}{\sum_{s'} P_t(s')\sum_{z'} P_t(z' \mid s')P(f \mid z',s')}}$$

  - Compute the a priori weight of speaker s and mixture

$$\boxed{P_t(s) = \frac{\sum_z \sum_f P_t(s,z \mid f)S_t(f)}{\sum_{s'}\sum_{z'}\sum_f P_t(s',z' \mid f)S_t(f)}} \quad \boxed{P_t(z \mid s) = \frac{\sum_f P_t(s,z \mid f)S_t(f)}{\sum_{z'}\sum_f P_t(s',z' \mid f)S_t(f)}}$$

  - Compute unknown bases

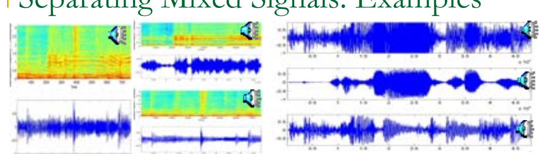
$$\boxed{P(f \mid z,s) = \frac{\sum_t P_t(s,z \mid f)S_t(f)}{\sum_{f'}\sum_t P_t(s,z \mid f')S_t(f')}}$$

Partial information: bases for one source unknown

- P(f|z,s) must be initialized for the additional source
- Estimation procedure now estimates bases along with mixture weights and source probabilities
  - From the ***mixed signal itself***
- The final separation is done as before

$$\boxed{\hat{S}_{t,i}(f) = S_t(f)\sum_z P_t(z,s \mid f)}$$

Separating Mixed Signals: Examples

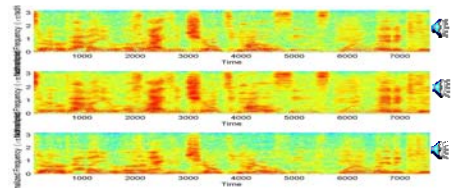


- "Raise my rent" by David Gilmour
- Background music "bases" learnt from 5-seconds of music-only segments within the song
- Lead guitar "bases" bases learnt from the rest of the song
- Norah Jones singing "Sunrise"
- A more difficult problem:
  - Original audio clipped!
- Background music bases learnt from 5 seconds of music-only segments

Where it works

- When the spectral structures of the two sound sources are distinct
  - Don't look much like one another
  - E.g. Vocals and music
  - E.g. Lead guitar and music
- Not as effective when the sources are similar
  - Voice on voice

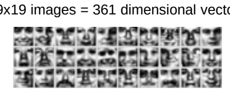
Separate overlapping speech

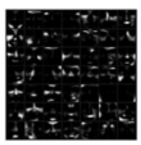


- Bases for both speakers learnt from 5 second recordings of individual speakers
- Shows improvement of about 5dB in Speaker-to-Speaker ratio for both speakers
  - Improvements are worse for same-gender mixtures

How about non-speech data

19x19 images = 361 dimensional vectors





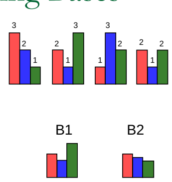
- We can use the same model to represent other data
- Images:
  - Every face in a collection is a histogram
  - Each histogram is composed from a mixture of a fixed number of multinomials
    - All faces are composed from the same multinomials, but the manner in which the multinomials are selected differs from face to face
  - Each component multinomial is also an image
    - And can be learned from a collection of faces
- Component multinomials are observed to be *parts of faces*

How many bases can we learn

- The *number* of bases that must be learned is a fundamental question
  - How do we know how many bases to learn
  - How many bases can we actually learn computationally
- A key computational problem in learning bases:
  - The number of bases we can learn correctly is restricted by the dimension of the data
  - I.e., if the spectrum has *F* frequencies, we cannot estimate more than *F-1* component multinomials reliably
    - Why?

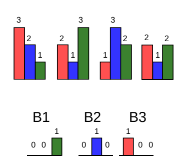
Indeterminacy in Learning Bases

- Consider the four histograms to the right
- All of them are mixtures of the same K component multinomials
- For K < 3, a single global solution may exist
  - I.e there may be a unique set of component multinomials that explain all the multinomials
    - With error – model will not be perfect
- For K = 3 a trivial solution exists



Indeterminacy

- Multiple solutions for K = 3..
  - We cannot *learn* a non-trivial set of "optimal" bases from the histograms
  - The component multinomials we do learn tell us nothing about the data
- For K > 3, the problem only gets worse
  - An inifinite set of solutions are possible
    - E.g. the trivial solution plus a random basis



Indeterminacy in signal representations

- Spectra:
  - If our spectra have D frequencies (no. of unique indices in the DFT) then..
  - We cannot learn D or more meaningful component multinomials to represent them
    - The trivial solution will give us D components, each of which has probability 1.0 for one frequency and 0 for all others
    - This does not capture the innate spectral structures for the source
- Images: Not possible to learn more than P-1 meaningful component multinomials from a collection of P-pixel images

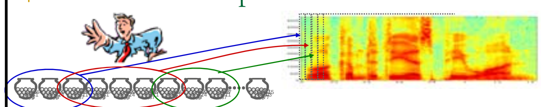
How many bases to represent sounds/images?

- In each case, the bases represent “typical unit structures”
  - Notes
  - Phonemes
  - Facial features..
- How many notes in music
  - Several octaves
  - Several instruments
- The typical sounds in speech –
  - Many phonemes, many variations, can number in the thousands
- Images:
  - Millions of units that can compose an image – trees, dogs, walls, sky, etc. etc. etc…
- To model the data well, *all of these must be represented*
  - More bases than dimensions

Overcomplete Representations

- Representations where there are more bases than dimensions are called *Overcomplete*
  - E.g. more multinomial components than dimensions
  - Overcomplete representations are required to represent the world adequately
    - The complexity of the world is not restricted by the dimensionality of our representations!
- Overcomplete representations are difficult to compute
  - Straight-forward computation results in indeterminate solutions
- Additional constraints must be imposed in the learning process to learn more components than dimensions
- We will require our solutions to be ***sparse***

SPARSE Decompositions



- Allow any arbitrary number of bases (urns)
  - Overcomplete
- Specify that for any *specific* frame only a small number of bases may be used
  - Although there are many spectral structures, any given frame only has a few of these
- In other words, the mixture weights with which the bases are combined must be sparse
  - Have non-zero value for only a small number of bases
  - Alternately, be of the form that only a small number of bases contribute significantly

The history of sparsity

- The search for “sparse” decompositions has a long history
  - Even outside the scope of overcomplete representations
- A landmark paper: Sparse Coding of Natural Images Produces Localized, Oriented, Bandpass Receptive Fields, by Olshausen and Fields
  - “*The images we typically view, or natural scenes, constitute a minuscule fraction of the space of all possible images. It seems reasonable that the visual cortex, which has evolved and developed to effectively cope with these images, has discovered efficient coding strategies for representing their structure. Here, we explore the hypothesis that the coding strategy employed at the earliest stage of the mammalian visual cortex maximizes the sparseness of the representation. We show that a learning algorithm that attempts to find linear sparse codes for natural scenes will develop receptive fields that are localized, oriented, and bandpass, much like those in the visual system.*”
  - Images can be described in terms of a small number of descriptors from a large set
    - E.g. a scene is “a grapevine plus grapes plus a fox plus sky”
- Other studies indicate that human perception may be based on sparse compositions of a large number of “icons”
- The number of sensors (rods/cones in the eye, hair cells in the ear) is much smaller than the number of visual / auditory objects in the world around us
  - The internal representation of images must be overcomplete

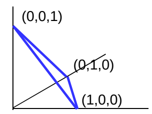
Estimating Mixture Weights given Multinomials

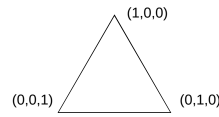
- Basic estimation: Maximum likelihood
  - $\text{Argmax}_W \; \log P(X \,;\, B,W) = \text{Argmax}_W \; \Sigma_f X(f)\log(\Sigma_i w_i B_i(f))$
- Modified estimation: Maximum *a posteriori*
  - Denote W = [w1 w2 ..] (in vector form)
  - $\text{Argmax}_W \; \Sigma_f X(f)\log(\Sigma_i w_i B_i(f)) + \beta \log P(W)$
- Sparsity obtained by enforcing an *a priori* probability distribution P(W) over the mixture weights that favors sparse mixture weights
- The algorithm for estimating weights must be modified to account for the priors

The *a priori* distribution

- A variety of *a priori* probability distributions all provide a bias towards "sparse" solutions
- The Dirichlet prior:
  - $P(W) = Z^* \Pi_i w_i^{\alpha-1}$
- The entropic prior:
  - $P(W) = Z^*\exp(-\alpha H(W))$
    - $H(W)$ = entropy of W = $-\Sigma_i w_i \log(w_i)$

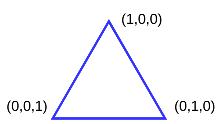
A simplex view of the world





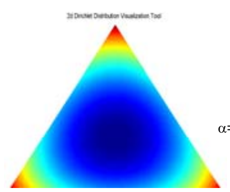
- The mixture weights are a probability distribution
  - $\Sigma_i w_i = 1.0$
- They can be viewed as a vector
  - $W = [w_0\ w_1\ w_2\ w_3\ w_4\ \ldots]$
  - The vector components are positive and sum to 1.0
- All probability vectors lie on a *simplex*
  - A convex region of a linear subspace in which all vectors sum to 1.0

Probability Simplex



- The sparsest probability vectors lie on the vertices of the simplex
- The edges of the simplex are progressively less sparse
  - Two-dimensional edges have 2 non-zero elements
  - Three-dimensional edges have 3 non-zero elements
  - Etc.

Sparse Priors: Dirichlet

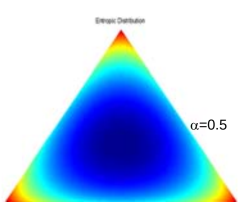


$P(W) = Z^* \Pi_i w_i^{\alpha-1}$

$\alpha=0.5$

- For alpha < 1, sparse probability vectors are more likely than dense ones

Sparse Priors: The entropic prior



$P(W) = Z^*\exp(-\alpha H(W))$

$\alpha=0.5$

- Vectors (probability distributions) with low entropy are more probable than those with high entropy
  - Low-entropy distributions are sparse!

Optimization with the entropic prior

- The objective function

  $\text{Argmax}_W\ \Sigma_X X(f)\log(\Sigma_i w_i B_i(f)) - \alpha H(W)$
- By estimating W such that the above equation is maximized, we can derive minimum entropy solutions
  - Jointly optimize W for predicting the data while minimizing its entropy

The Expectation Maximization Algorithm

- The parameters are actually learned using the *Expectation Maximization* (EM) algorithm
- The EM algorithm actually optimizes the following objective function
  - $Q = \Sigma_X P(Z \mid f) X(f)\log(P(Z) P(f|Z)) - \alpha H(\{P(Z)\})$
    - $P(Z) = w_z$, $\{P(Z)\} = W$
- The second term here is derived from the entropic prior
- Optimization of the above needs a solution to the following

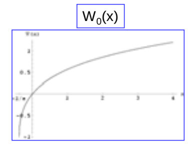
$$\frac{\sum_f S(t,f) P_t(z \mid f)}{P_t(z)} + \alpha(1+\log P_t(z)) + \lambda = 0$$

- The solution requires a new function:
  - The lambert W function

Lambert's W Function

- Lambert's W function is the solution to:
  W + log(W) = X
  - Where W = F(X) is the Lambert function
- Alternately, the *inverse* function of
  - **X = W exp(W)**
- In general, a multi-valued function
- If X is real, W is real for X > -1/e
  - Still multi-valued
- If we impose the restriction W > -1 and W == real we get the zeroth branch of the W function
  - Single valued
- For W < -1 and W == real we get the -1th branch of the W function
  - Single valued

$W_0(x)$



Estimating $W_0(z)$

- An iterative solution
  - Newton's Method

$$w_{j+1} = w_j - \frac{w_j e^{w_j} - z}{e^{w_j} + w_j e^{w_j}}$$

  - Halley Iterations

$$w_{j+1} = w_j - \frac{w_j e^{w_j} - z}{e^{w_j}(w_j+1) - \frac{(w_j+2)(w_j e^{w_j} - z)}{2w_j+2}}$$

  - Code for Lambert's W function is available on wikipedia

Solutions with entropic prior

$$P_t(z) = \frac{-\gamma/\alpha}{W(-\gamma e^{1+\lambda/\alpha}/\alpha)}; \qquad \gamma = \sum_f S_t(f) P_t(z \mid f)$$

$$\lambda = -\left(\frac{\gamma}{P_t(z)} + \alpha\left(1 + \log\left(P_t(z)\right)\right)\right)$$

- The update rules are the same as before, with one minor modification
- To estimate the mixture weights, the above two equations must be iterated
  - To convergence
  - Or just for a few iterations
- Alpha is the sparsity factor
- $P_t(z)$ must be initialized randomly

Learning Rules for Overcomplete Basis Set

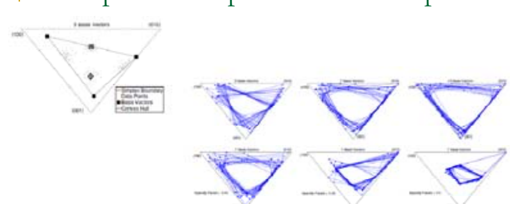
- Exactly the same as earlier, with the modification that $P_t(z)$ is now estimated to be sparse
  - Initialize $P_t(z)$ for all t and P(f|z)
  - Iterate

$$P_t(z \mid f) = \frac{P_t(z) P(f \mid z)}{\sum_{z'} P_t(z') P(f \mid z')} \qquad P(f \mid z) = \frac{\sum_t P_t(z \mid f) S_t(f)}{\sum_{f'} \sum_t P_t(z \mid f') S_t(f')}$$

$$P_t(z) = \frac{-\gamma/\alpha}{W(-\gamma e^{1+\lambda/\alpha}/\alpha)}; \qquad \gamma = \sum_f S_t(f) P_t(z \mid f)$$

$$\lambda = -\left(\frac{\gamma}{P_t(z)} + \alpha\left(1 + \log\left(P_t(z)\right)\right)\right)$$

A Simplex Example for Overcompleteness

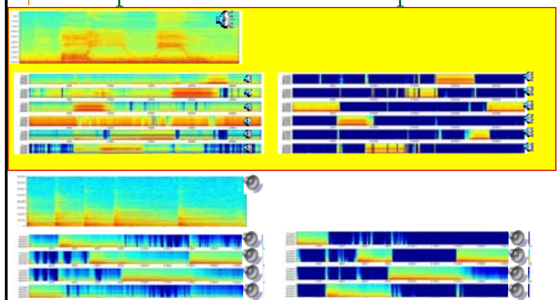


- Synthetic data: Four clusters of data within the probability simplex
- Regular learning with 3 bases learns an enclosing triangle
- Overcomplete solutions without sparsity restults in meaningless solutions
- Sparse overcomplete model captures the distribution of the data

Sparsity can be employed *without* overcompleteness

- Overcompleteness requires sparsity
- Sparsity does *not* require overcompleteness
  - Sparsity only imposes the constraint that the data are composed from a mixture of *as few multinomial components as possible*
  - This makes no assumption about overcompleteness

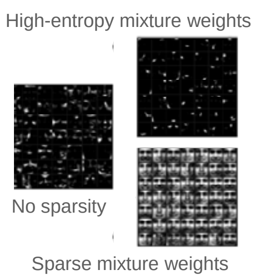
Examples without overcompleteness



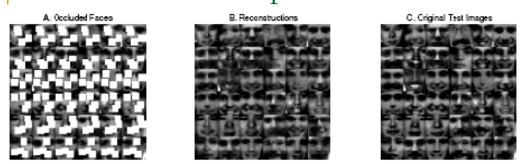
- Left panel, Regular learning: most bases have significant energy in all frames
- Right panel, Sparse learning: Fewer bases active within any frame
  - Sparse decomposiions result in more localized activation of bases
  - Bases, too, are better defined in their structure

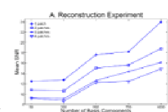
Face Data: The effect of sparsity

- As solutions get more sparse, bases become more informative
  - In the limit, each basis is a complete face by itself.
  - Mixture weights simply select face
- Solution also allows for mixture weights to have *maximum* entropy
  - *Maximally dense,* i.e. *minimally sparse*
  - The bases become much more localized components
- The sparsity factor allows us to tune the bases we learn



Benefit of overcompleteness





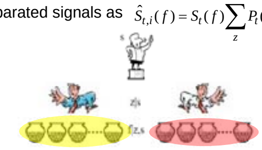
- 19x19 pixel images (361 pixels)
- Up to1000 bases trained from 2000 faces
- SNR of reconstruction from overcomplete basis set more than 10dB better than reconstruction from corresponding “compact” (regular) basis set

Signal Processing: How

- Exactly as before
- Learn an overcomplete set of bases
- For each new data vector to be processed, compute the optimal mixture weights
  - Constrainting the mixture weights to be sparse now
- Use the estimated mixture weights and the bases to perform additional processing

Signal Separation with Overcomplete Bases

- Learn overcomplete bases for each source
- For each frame of the mixed signal
  - Estimate prior probability of source and mixture weights for each source
    - Constraint: Use *sparse* learning for mixture weights
- Estimate separated signals as $\hat{S}_{t,i}(f) = S_t(f)\sum_z P_t(z,s \mid f)$

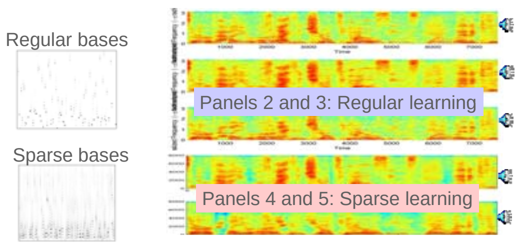


$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

$$P_t(f) = P_t(s_1)\sum_z P_t(z \mid s_1)P(f \mid z, s_1) + P_t(s_2)\sum_z P_t(z \mid s_1)P(f \mid z, s_2)$$

Sparse Overcomplete Bases: Separation

- 3000 bases for each of the speakers
  - The speaker-to-speaker ratio typically doubles (in dB) w.r.t "compact" bases



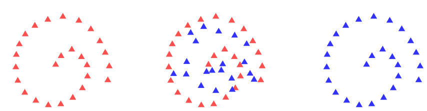
The Limits of Overcompleteness

- How many bases can we learn?
- The limit is: as many bases as the number of vectors in the training data
  - Or rather, the number of distinct histograms in the training data
    - Since we treat each vector as a histogram
- It is not possible to learn more than this number regardless of sparsity
  - The arithmetic supports it, but the results will be meaningless

Working at the limits of overcompleteness: The "Example-Based" Model

- *Every training vector is a basis*
  - Normalized to be a distribution
- Let S(t,f) be the t^{th} training vector
- Let T be the total number of training vectors
- The total number of bases is T
- The k^{th} basis is given by
  - $B(k,f) = S(k,f) / \Sigma_f S(k,f) = S(k,f) / |S(k,f)|_1$
- Learning bases requires no additional learning steps besides simply collecting (and computing spectra from) training data

The example based model – an illustration

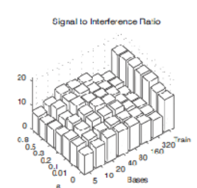


- In the above example all training data lie on the curve shown (Left Panel)
  - Each of them is a vector that sums to 1.0
- The learning procedure for bases learns multinomial components that are linear combinations of the data (Middle Panel)
  - These can lie anywhere within the area enclosed by the data
  - The layout of the components hides the actual structure of the layout of the data
- The example based representation captures the layout of the data perfectly (right panel)
  - Since the data *are the bases*

Signal Processing with the Example Based Model

- All previously defined operations can be performed using the example based model exactly as before
  - For each data vector, estimate the optimal mixture weights to combine the bases
    - Mixture weights MUST be estimated to be sparse
- The example based representation is simply a special case of an overcomplete basis set

Speaker Separation Example



- Speaker-to-interference ratio of separated speakers
  - State-of-the-art separation results

Example-based model: *All* the training data?

- In principle, no need to use *all* training data as the model
  - A well-selected subset will do
  - E.g. – ignore spectral vectors from all pauses and non-speech regions of speech samples
  - E.g. – eliminate spectral vectors that are nearly identical
- The problem of *selecting* the optimal set of training examples remains open, however

Summary So Far

- PLCA:
  - The basic mixture-multinomial model for audio (and other data)
- Sparse Decomposition:
  - The notion of sparsity and how it can be imposed on learning
- Sparse Overcomplete Decomposition:
  - The notion of *overcomplete* basis set
- Example-based representations
  - Using the training data itself as our representation