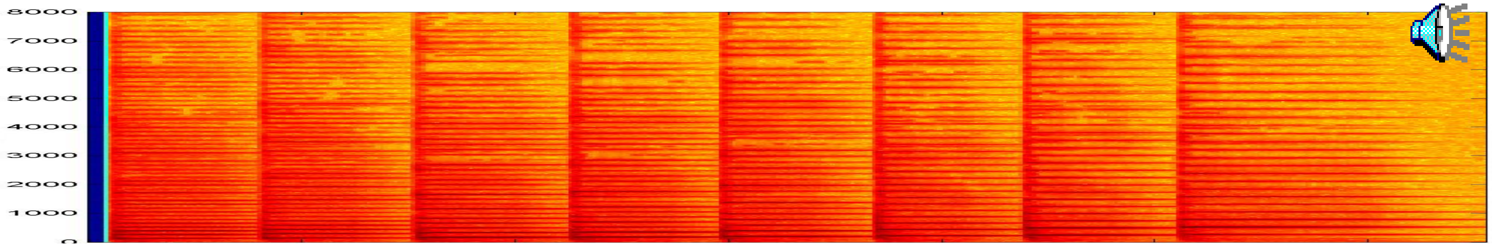
Latent Variable Models and Signal Separation

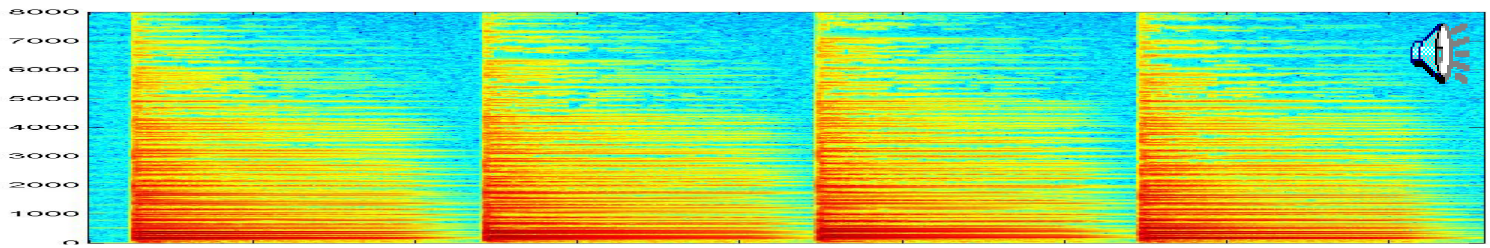
Class 21. 02 Nov 2010

Sounds – an example

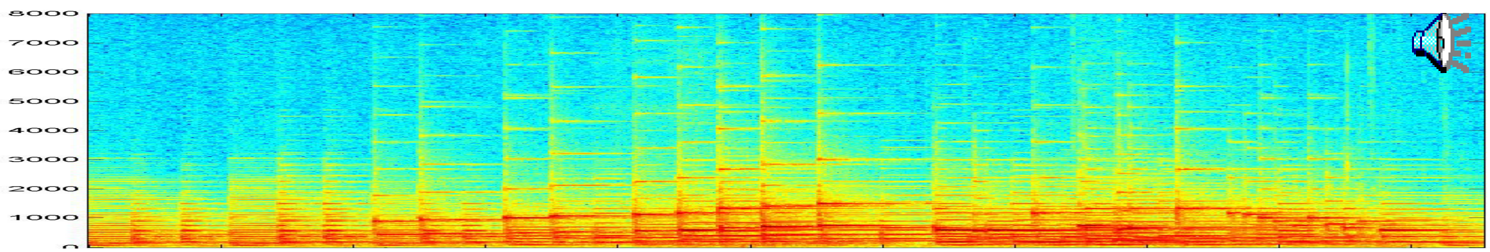
- A sequence of notes



- Chords from the same notes

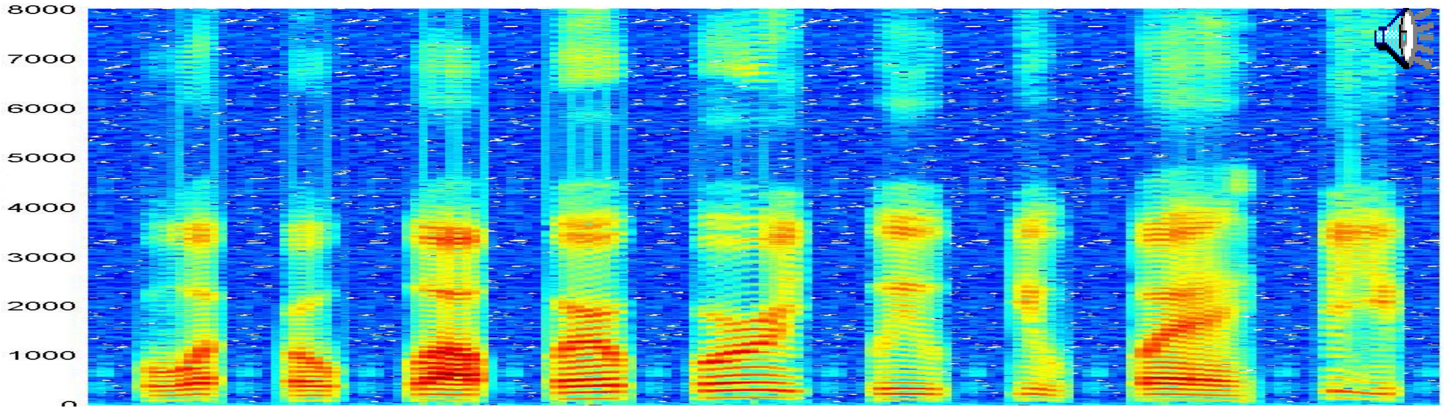


- A piece of music from the same (and a few additional) notes

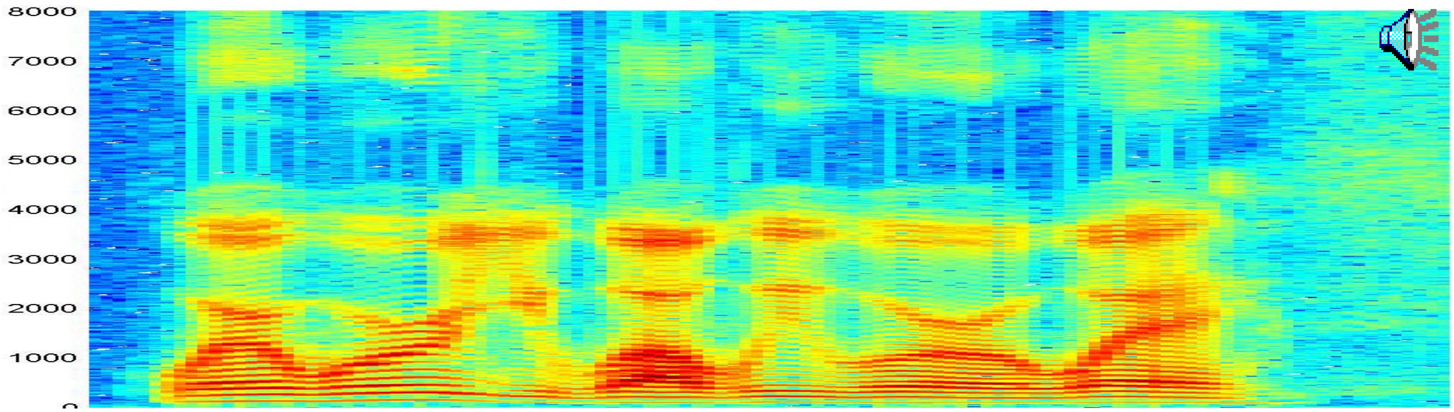


Sounds – an example

- A sequence of sounds



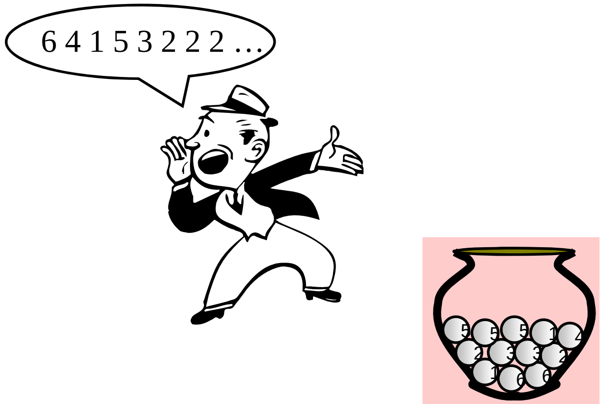
- A proper speech utterance from the same sounds



Template Sounds Combine to Form a Signal

- The individual component sounds "combine" to form the final complex sounds that we perceive
  - Notes form music
  - Phoneme-like structures combine in utterances
  - Component sounds – notes, phonemes – too are complex
- Sound in general is composed of such "building blocks" or themes
  - Our definition of a building block: the entire structure occurs repeatedly in the process of forming the signal
- Goal: To learn these building blocks automatically, from analysis of data

Urns and balls



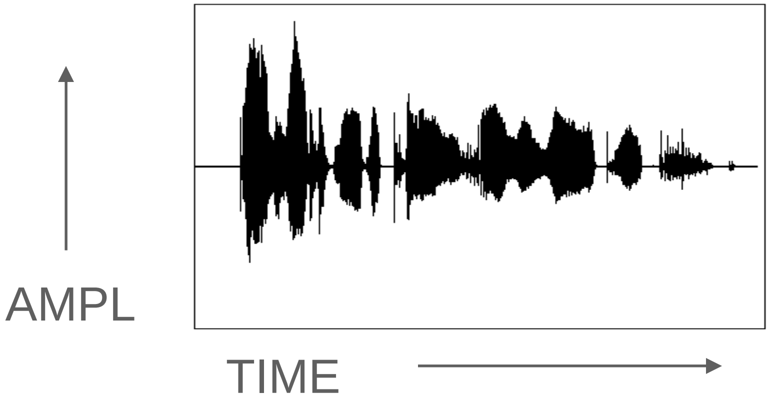
- An urn has many balls
- Each ball has a number marked on it
  - Multiple balls may have the same number
- A "picker" draws balls at random..
- This is a multinomial

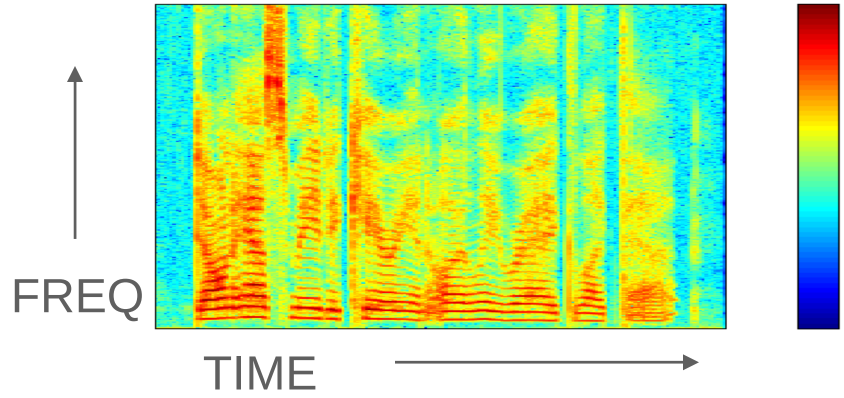
Signal Separation with the Urn model

- What does the probability of drawing balls from Urns have to do with sounds?
  - Or Images?

- We shall see..

The representation





- We represent signals spectrographically
  - Sequence of magnitude spectral vectors estimated from (overlapping) segments of signal
  - Computed using the short-time Fourier transform

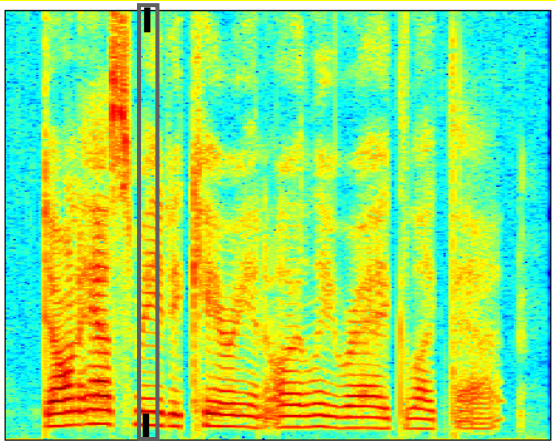
  - Note: Only retaining the magnitude of the STFT for our operations

  - We will, however need the phase later for conversion to a signal

A Multinomial Model for Spectra

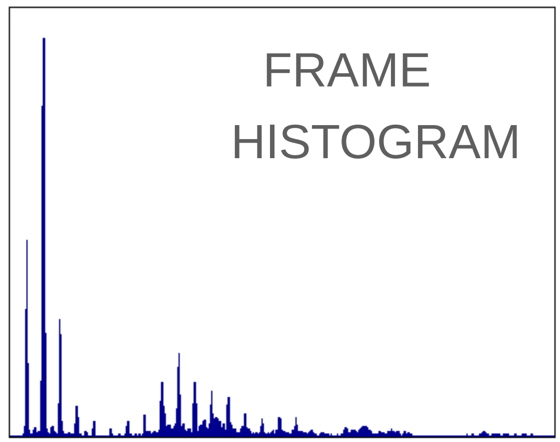
- A magnitude spectral vector obtained from a DFT represents spectral magnitude against discrete frequencies
  - This may be viewed as a histogram of draws from a multinomial

f



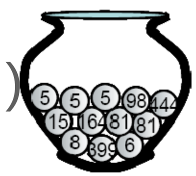
t

FRAME



f

Power spectrum of frame t



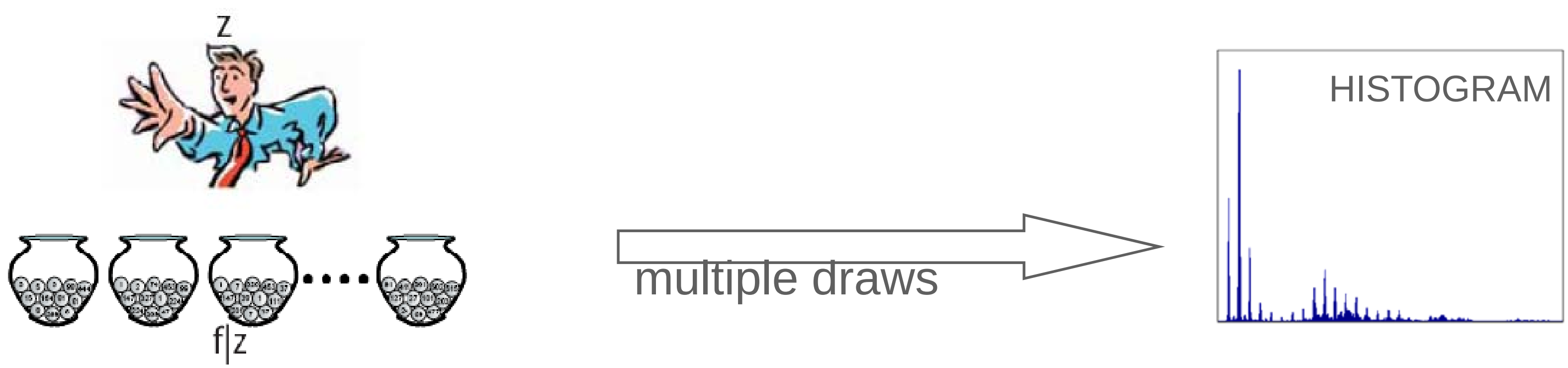
$P_t(f)$

The balls are marked with discrete frequency indices from the DFT

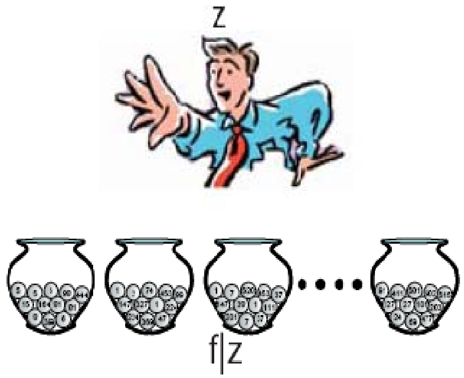
Probability distribution underlying the t-th spectral vector

A more complex model

- A "picker" has multiple urns
- In each draw he first selects an urn, and then a ball from the urn
  - Overall probability of drawing *f* is a *mixture multinomial*
    - Since several multinomials (urns) are combined
  - Two aspects – the probability with which he selects any urn, and the probability of frequencies with the urns

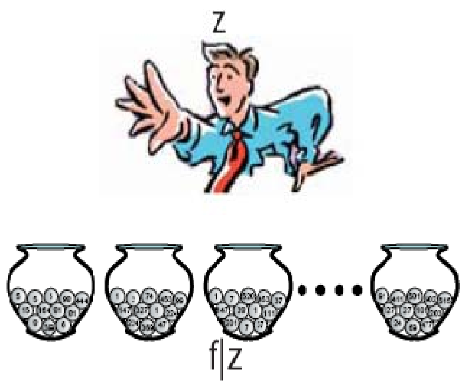


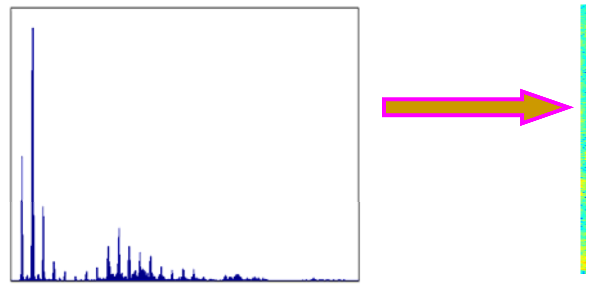
The Picker Generates a Spectrogram



- The picker has a fixed set of Urns
  - Each urn has a different probability distribution over *f*
- He draws the spectrum for the first frame
  - In which he selects urns according to some probability $P_0(z)$
- Then draws the spectrum for the second frame
  - In which he selects urns according to some probability $P_1(z)$
- And so on, until he has constructed the entire spectrogram

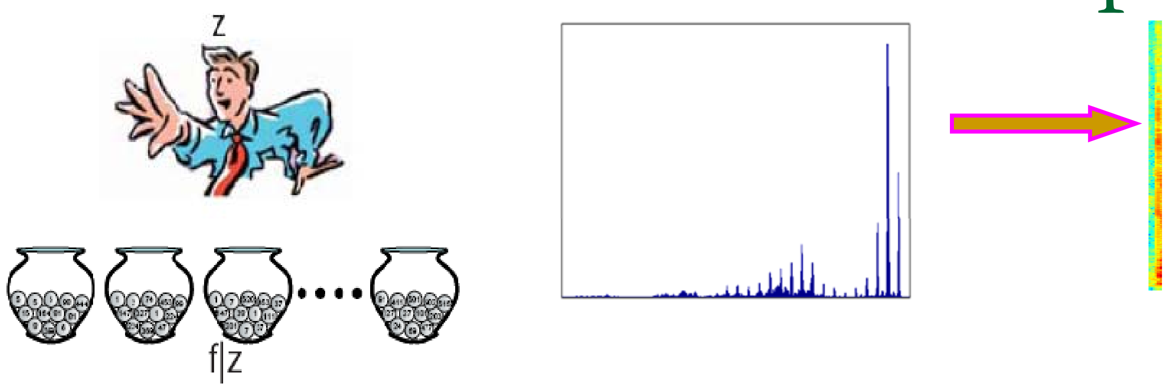
The Picker Generates a Spectrogram





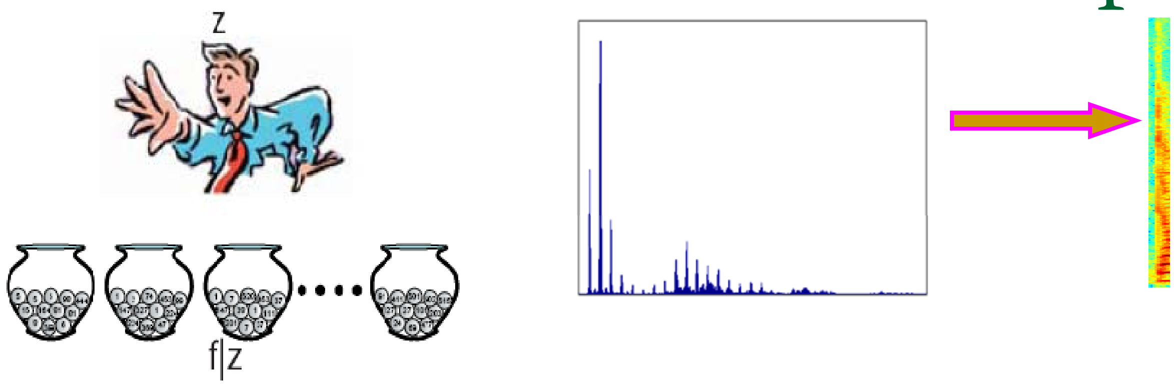
- The picker has a fixed set of Urns
  - Each urn has a different probability distribution over *f*
- He draws the spectrum for the first frame
  - In which he selects urns according to some probability $P_0(z)$
- Then draws the spectrum for the second frame
  - In which he selects urns according to some probability $P_1(z)$
- And so on, until he has constructed the entire spectrogram

The Picker Generates a Spectrogram



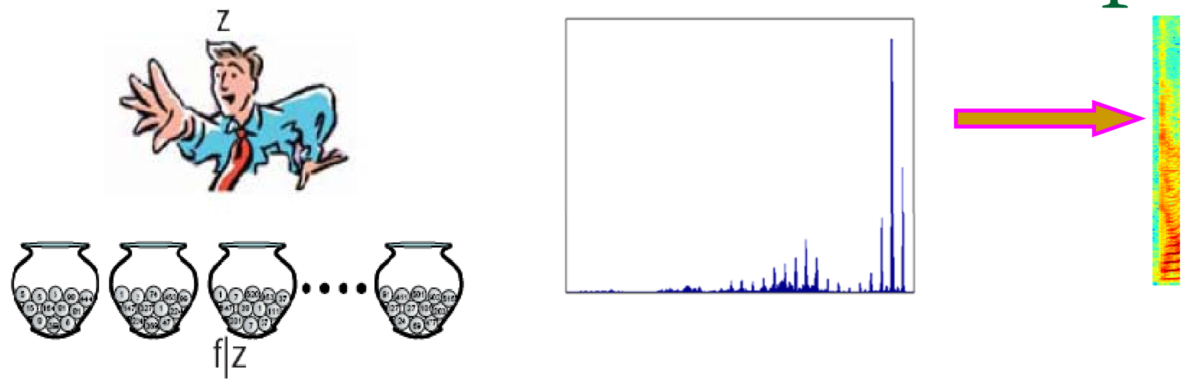
- The picker has a fixed set of Urns
  - Each urn has a different probability distribution over *f*
- He draws the spectrum for the first frame
  - In which he selects urns according to some probability $P_0(z)$
- Then draws the spectrum for the second frame
  - In which he selects urns according to some probability $P_1(z)$
- And so on, until he has constructed the entire spectrogram

The Picker Generates a Spectrogram



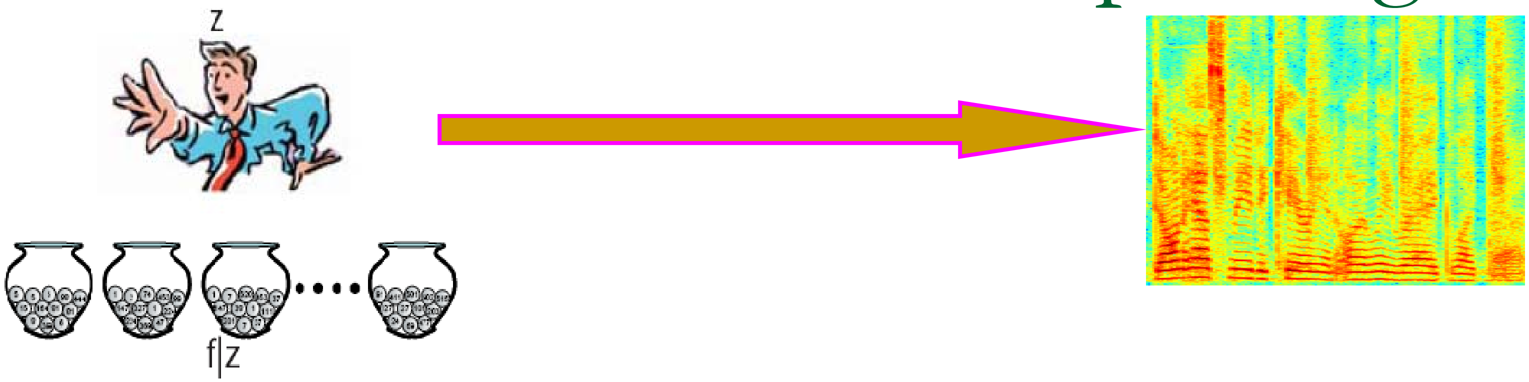
- The picker has a fixed set of Urns
  - Each urn has a different probability distribution over *f*
- He draws the spectrum for the first frame
  - In which he selects urns according to some probability $P_0(z)$
- Then draws the spectrum for the second frame
  - In which he selects urns according to some probability $P_1(z)$
- And so on, until he has constructed the entire spectrogram

The Picker Generates a Spectrogram



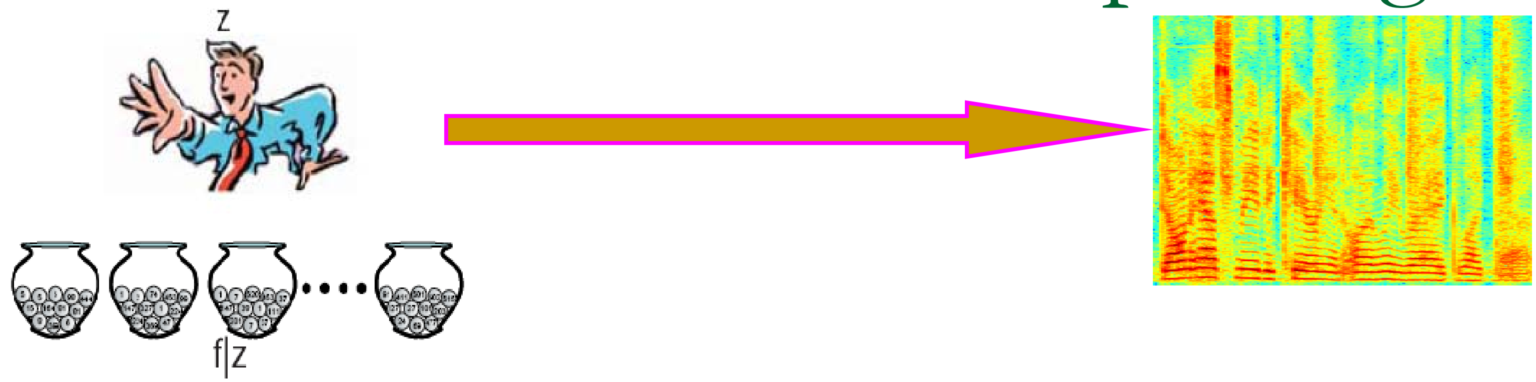
- The picker has a fixed set of Urns
  - Each urn has a different probability distribution over *f*
- He draws the spectrum for the first frame
  - In which he selects urns according to some probability $P_0(z)$
- Then draws the spectrum for the second frame
  - In which he selects urns according to some probability $P_1(z)$
- And so on, until he has constructed the entire spectrogram

The Picker Generates a Spectrogram



- The picker has a fixed set of Urns
  - Each urn has a different probability distribution over f
- He draws the spectrum for the first frame
  - In which he selects urns according to some probability $P_0(z)$
- Then draws the spectrum for the second frame
  - In which he selects urns according to some probability $P_1(z)$
- And so on, until he has constructed the entire spectrogram
  - The number of draws in each frame represents the rms energy in that frame

The Picker Generates a Spectrogram



- The URNS are the same for every frame
  - These are the ***component multinomials*** or ***bases*** for the source that generated the signal
- The only difference between frames is the probability with which he selects the urns

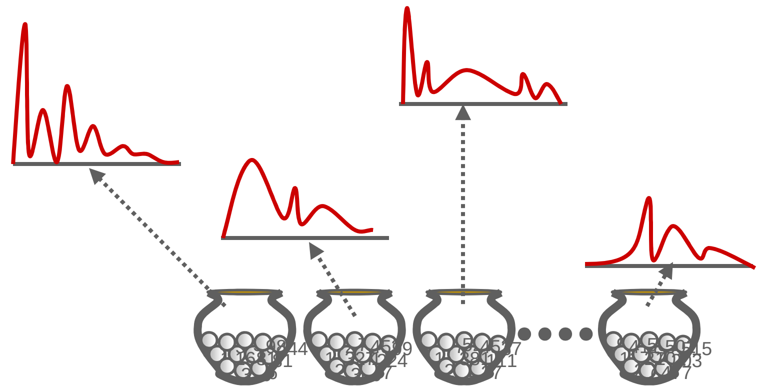
$$P_t(f) = \sum_z P_t(z) P(f \mid z)$$

Frame-specific spectral distribution

SOURCE specific bases

Frame(time) specific mixture weight

Spectral View of *Component* Multinomials

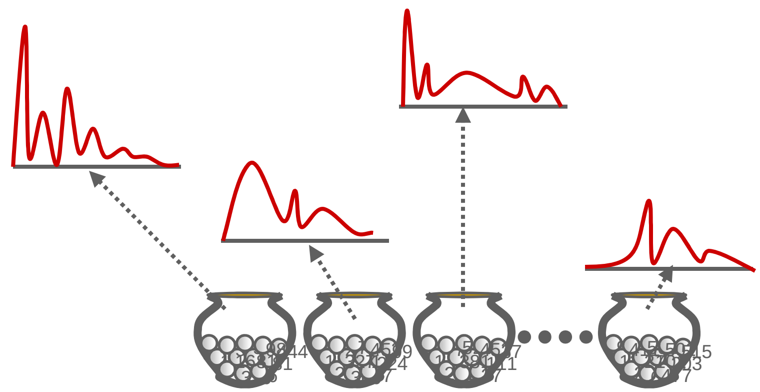


- Each component multinomial (urn) is actually a normalized histogram over frequencies $P(f\,|z)$
  - I.e. a spectrum

- Component multinomials represent latent spectral structures (bases) for the given sound source

- The spectrum for *every* analysis frame is explained as an additive combination of these latent spectral structures

Spectral View of *Component* Multinomials

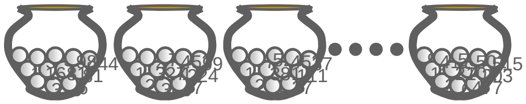


- By "learning" the mixture multinomial model for any sound source we "discover" these latent spectral structures for the source

- The model can be learnt from spectrograms of a small amount of audio from the source using the EM algorithm

EM learning of bases

- Initialize bases
  - P(f|z) for all z, for all f
    - Must decide on the number of urns



- For each frame
  - Initialize $P_t(z)$

EM Update Equations

- Iterative process:
  - Compute a posteriori probability of the z^{th} urn for the source for each f

$$P_t(z|f) = \frac{P_t(z)P(f|z)}{\sum_{z'} P_t(z')P(f|z')}$$

  - Compute mixture weight of z^{th} urn

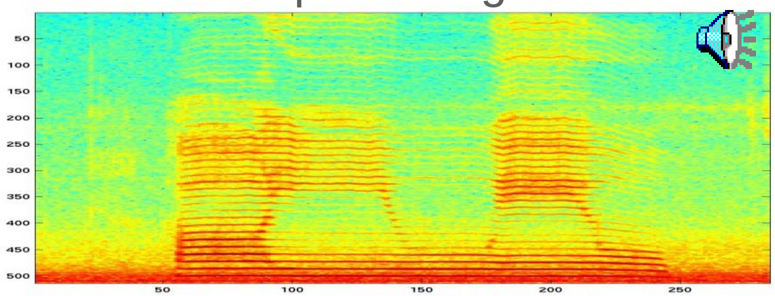
$$P_t(z) = \frac{\sum_f P_t(z|f)S_t(f)}{\sum_{z'}\sum_f P_t(z'|f)S_t(f)}$$

  - Compute the probabilities of the frequencies for the z^{th} urn

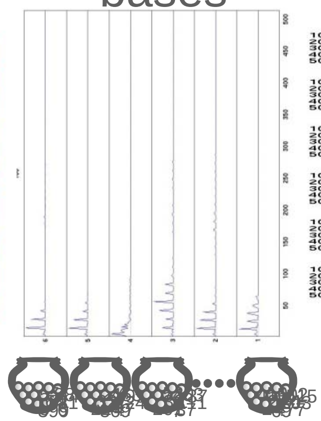
$$P(f|z) = \frac{\sum_t P_t(z|f)S_t(f)}{\sum_{f'}\sum_t P_t(z|f')S_t(f')}$$

Learning Structures

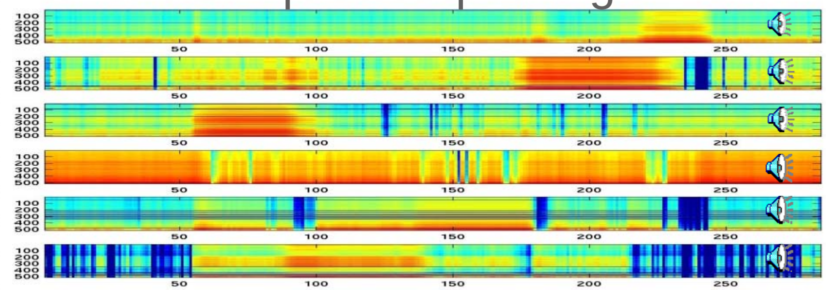
Speech Signal



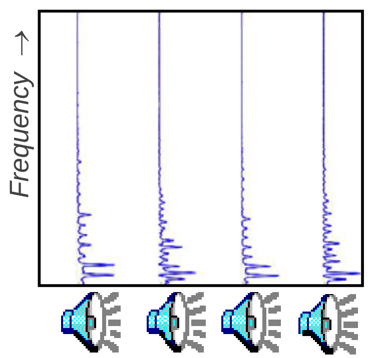
bases



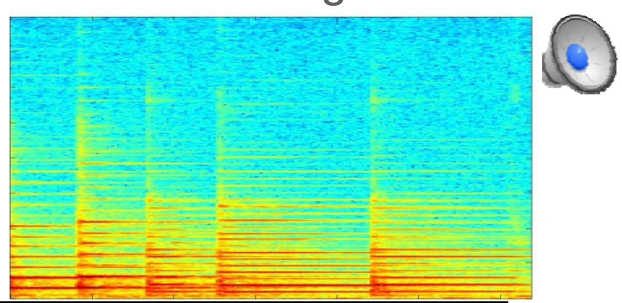
Basis-specific spectrograms

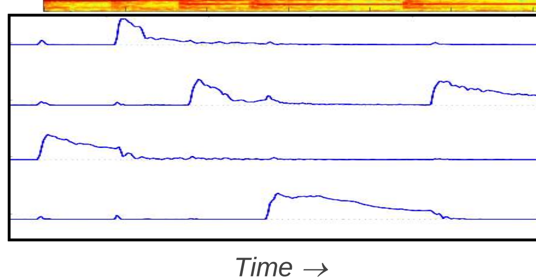


P(f|z)



From Bach's Fugue in Gm





$P_t(z)$

How meaningful are these structures

- If bases capture data structure they must
  - Allow prediction of data
    - **Hearing only the low-frequency components of a note, we can still know the note**
    - **Which means we can predict its higher frequencies**
  - Be resolvable in complex sounds
    - Must be able to pull them out of complex mixtures
      - **Denoising**
      - **Signal Separation from Monaural Recordings**

The musician vs. the signal processor

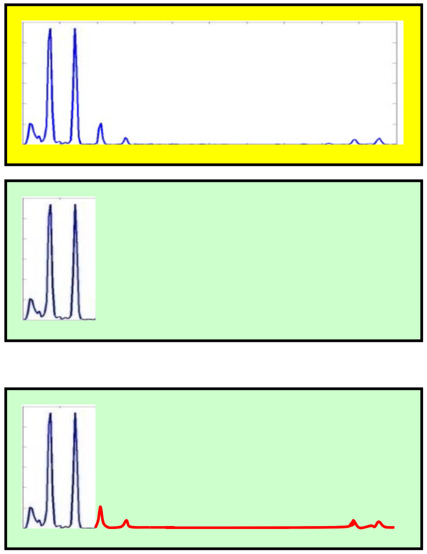
- Some badly damaged music is given to a signal processing whiz and a musician
  - They must “repair” it. What do they do?

- Signal processing :
  - Invents many complex algorithms
  - Writes proposals for government grants
  - Spends $1000,000
  - Develops an algorithm that results in less scratchy sounding music

- Musician:
  - Listens to the music and transcribes it
  - Plays it out on his keyboard/piano

Prediction

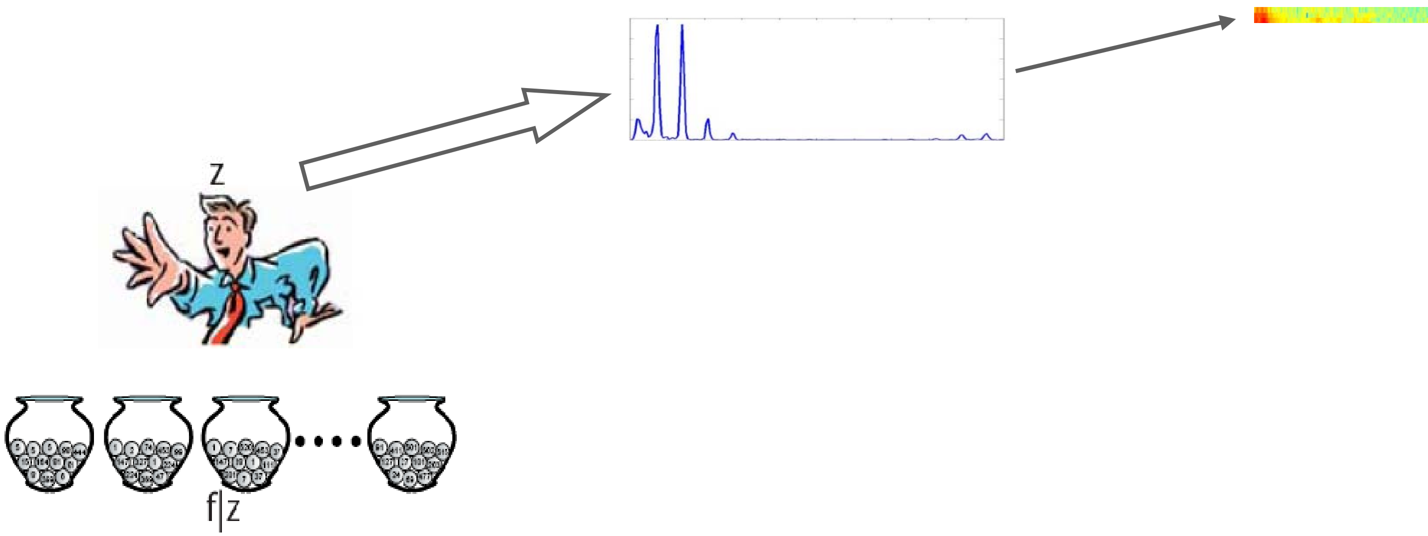
- **Bandwidth Expansion**
  - Problem: A given speech signal only has frequencies in the 300Hz-3.5Khz range
    - Telephone quality speech
  - Can we estimate the rest of the frequencies



- The full basis is known
- The presence of the basis is identified from the observation of a part of it
- The obscured remaining spectral pattern can be guessed

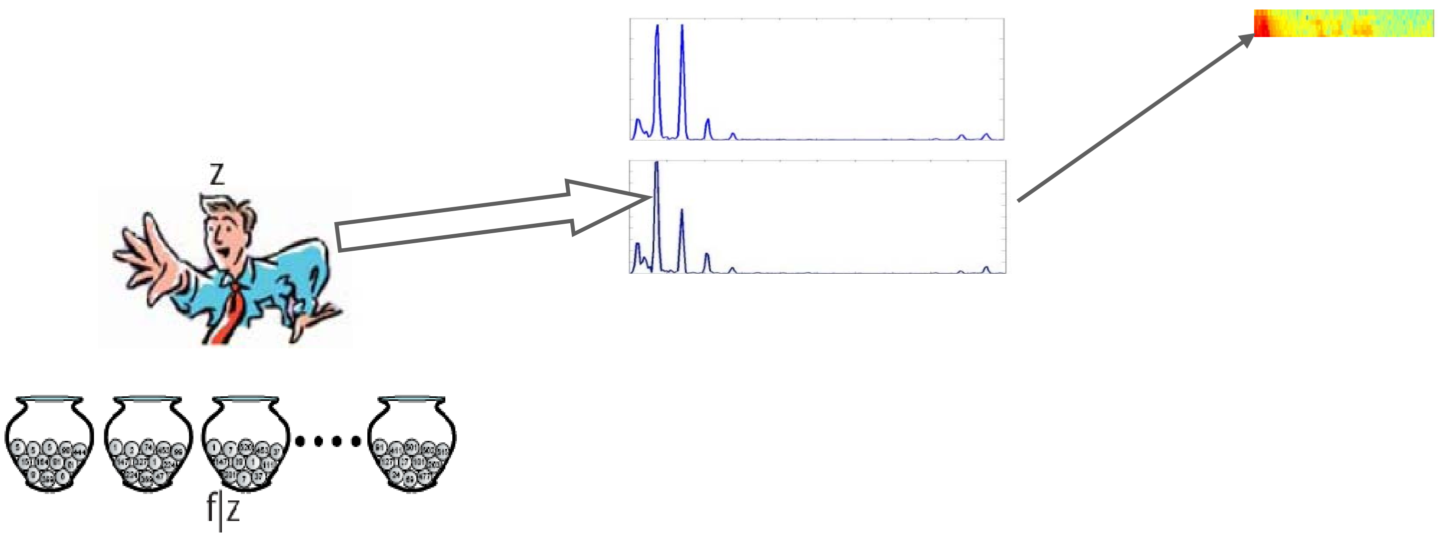
Bandwidth Expansion

- The picker has drawn the histograms for every frame in the signal



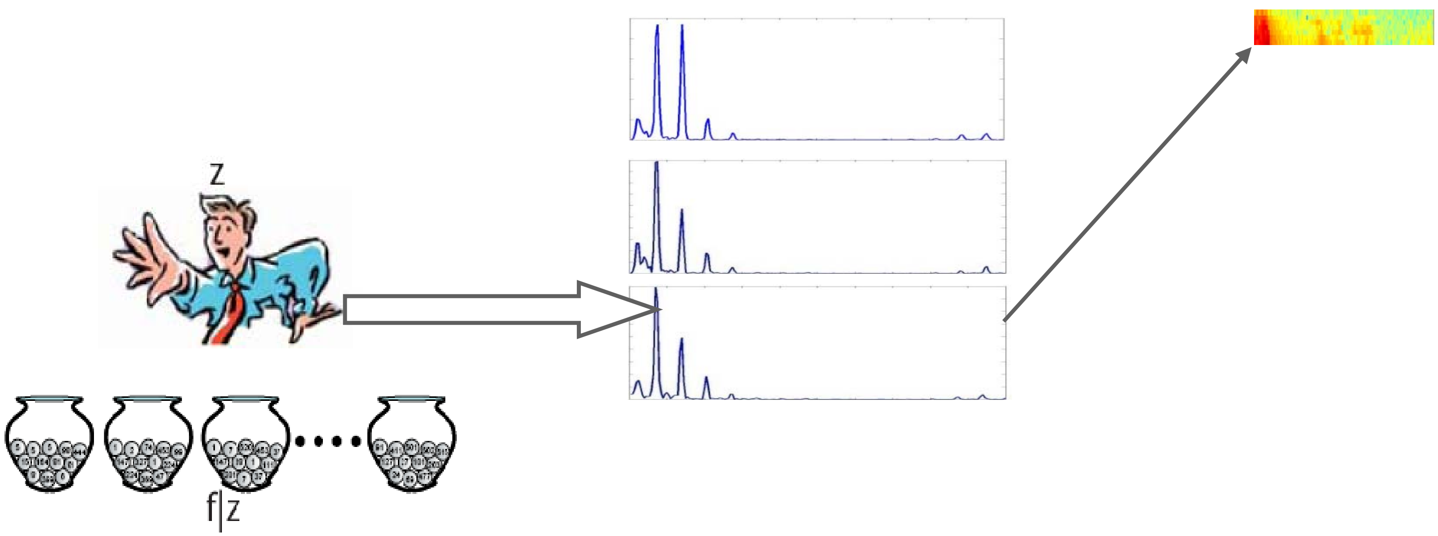
Bandwidth Expansion

- The picker has drawn the histograms for every frame in the signal



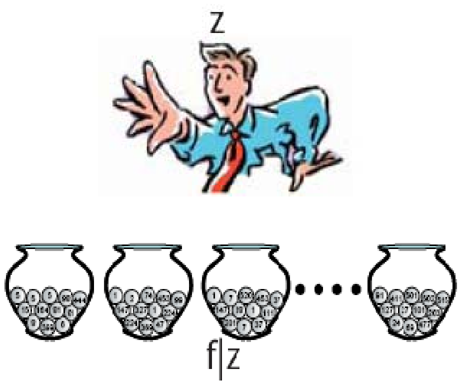
Bandwidth Expansion

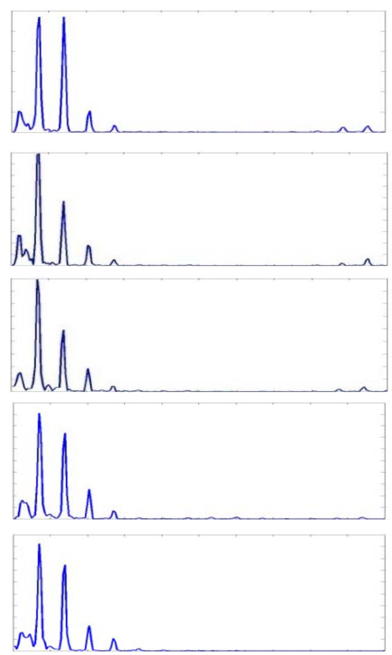
- The picker has drawn the histograms for every frame in the signal

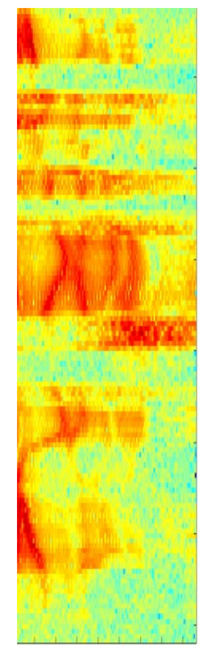


Bandwidth Expansion

- The picker has drawn the histograms for every frame in the signal

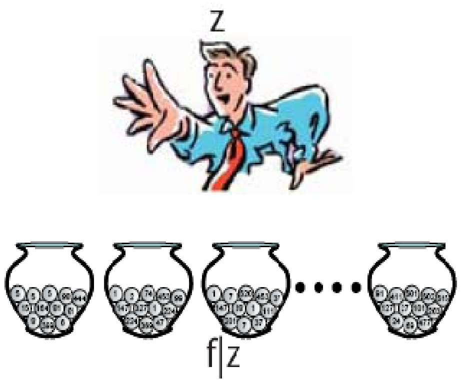


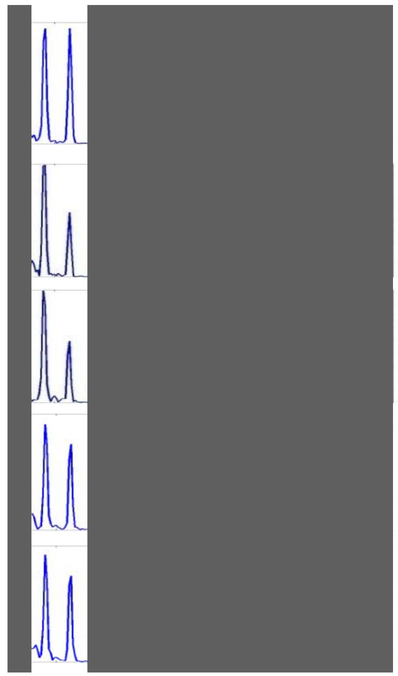


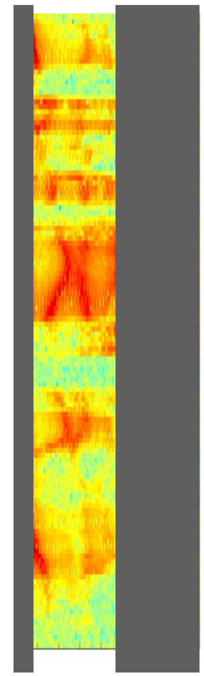


Bandwidth Expansion

- The picker has drawn the histograms for every frame in the signal

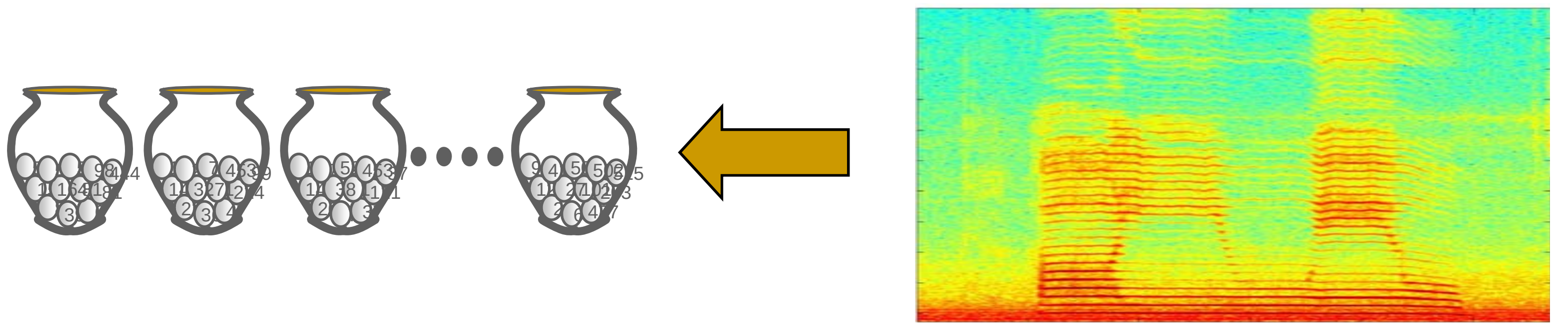






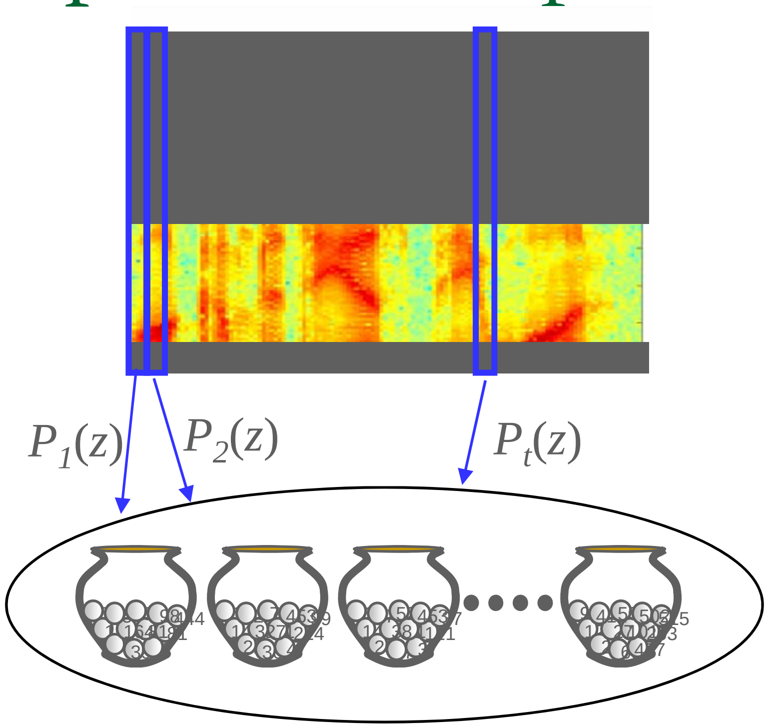
- However, we are only able to observe the number of draws of some frequencies and not the others
- We must estimate the number of draws of the unseen frequencies

Bandwidth Expansion: Step 1 – Learning



- From a collection of ***full-bandwidth*** training data that are similar to the bandwidth-reduced data, learn spectral bases
  - Using the procedure described earlier

Bandwidth Expansion: Step 2 – Estimation



- Using *only the observed frequencies* in the bandwidth-reduced data, estimate mixture weights for the bases learned in step 1.

Step 2

- Iterative process:
  - Compute a posteriori probability of the zth urn for the speaker for each *f*

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

  - Compute mixture weight of zth urn for each frame *t*

$$P_t(z) = \frac{\sum_{f \in (\text{observed frequencies})} P_t(z \mid f) S_t(f)}{\sum_{z'} \sum_{f \in (\text{observed frequencies})} P_t(z' \mid f) S_t(f)}$$

  - P(f|z) was obtained from training data and will not be reestimated

Step 3 and Step 4

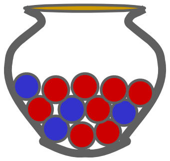
- Compose the complete probability distribution for each frame, using the mixture weights estimated in Step 2

$$P_t(f) = \sum_z P_t(z) P(f \mid z)$$

- Note that we are using mixture weights estimated from the reduced set of observed frequencies
    - This also gives us estimates of the probabilities of the *unobserved* frequencies

- Use the complete probability distribution $P_t(f)$ to predict the unobserved frequencies!

Predicting from $P_t(f)$: Simplified Example



- A single Urn with only red and blue balls
- Given that out an unknown number of draws, exactly *m* were red, how many were blue?
- One Simple solution:
  - Total number of draws N = m / P(red)
  - The number of tails drawn = N*P(blue)
  - Actual multinomial solution is only slightly more complex

Estimating unobserved frequencies

- Expected value of the number of draws:

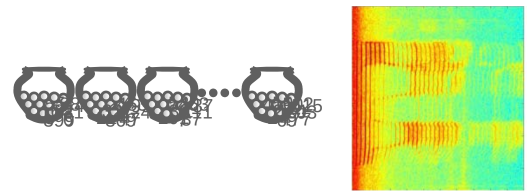
$$\hat{N}_t = \frac{\sum_{f \in (\text{observed frequencies})} S_t(f)}{\sum_{f \in (\text{observed frequencies})} P_t(f)}$$

- Estimated spectrum in unobserved frequencies

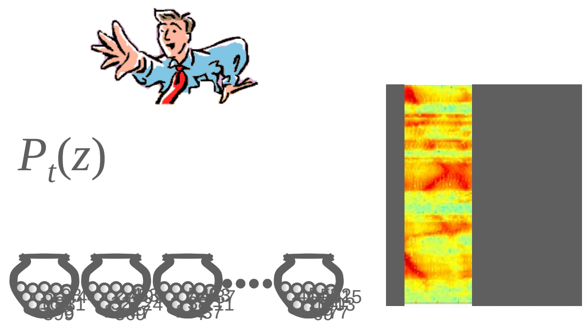
$$\hat{S}_t(f) = \hat{N}_t P_t(f)$$

Overall Solution

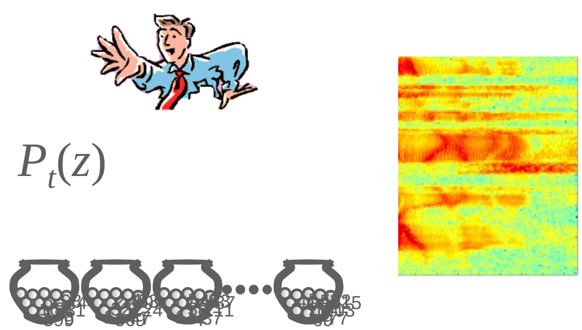
- Learn the “urns” for the signal source from broadband training data



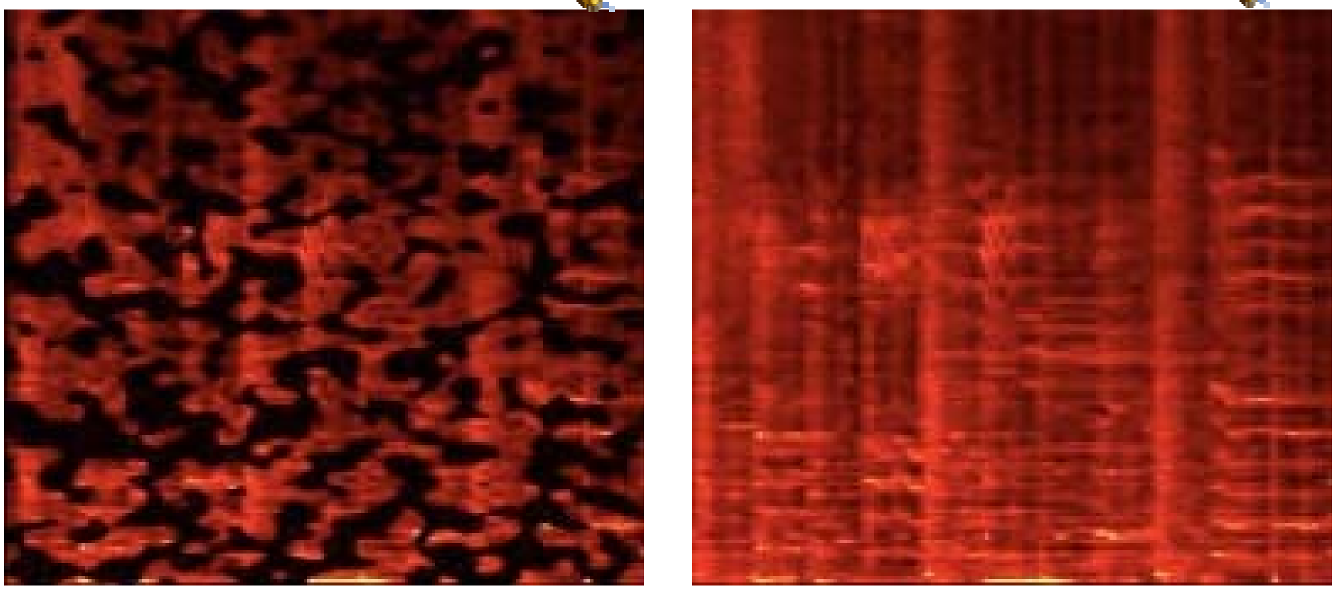
- For each frame of the reduced bandwidth test utterance, find mixture weights for the urns
  - Ignore (marginalize) the unseen frequencies



- Given the complete mixture multinomial distribution for each frame, estimate spectrum (histogram) at unseen frequencies



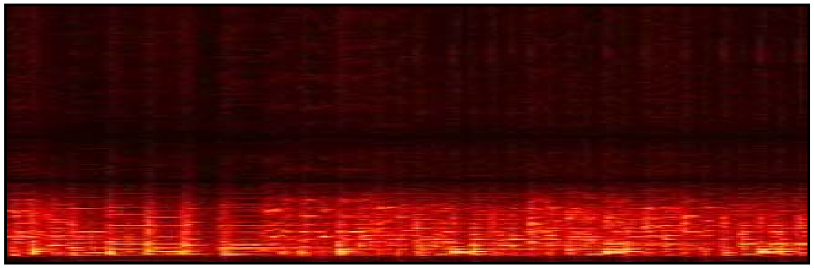
Prediction of Audio



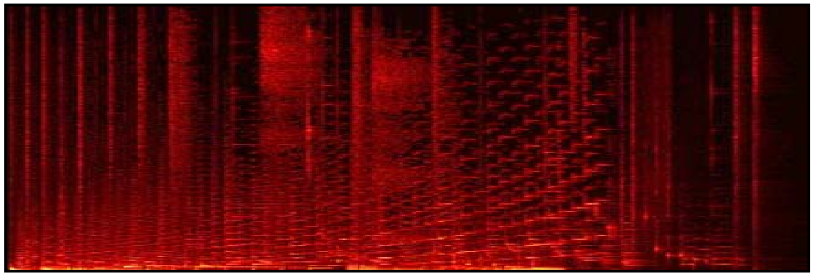
- Some frequency components are missing (left panel)
- We know the bases P(f|z)
  - But not the mixture weights for any particular spectral frame
- We must "fill in" the hole in the image
  - To obtain the one to the right
  - Easy to do – as explained

A more fun example

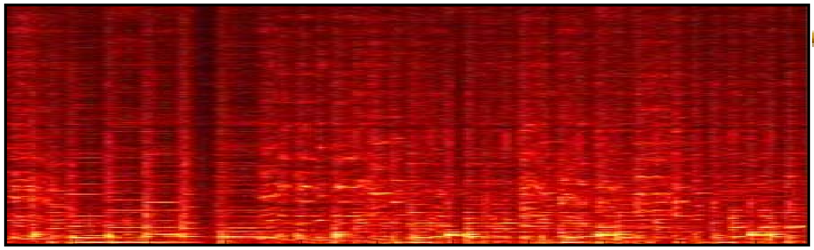
- Reduced BW data



- Bases learned from this



- Bandwidth expanded version

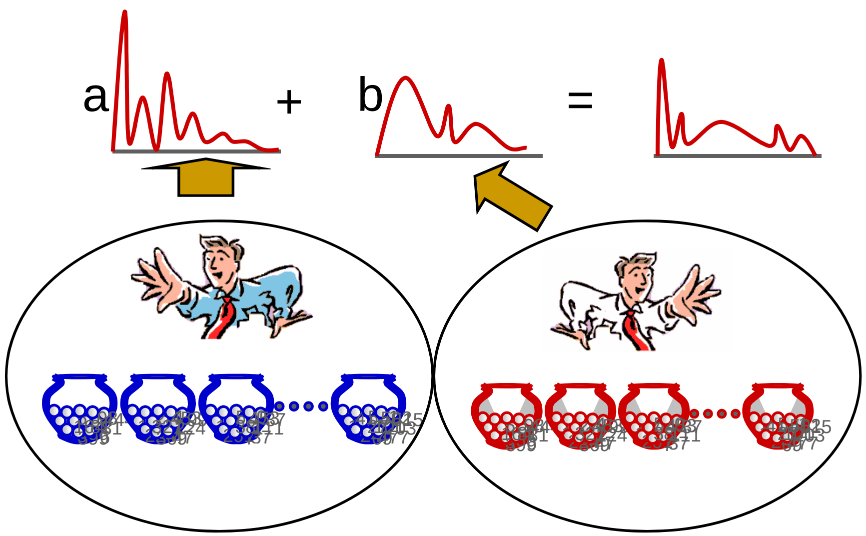


Signal Separation from Monaural Recordings

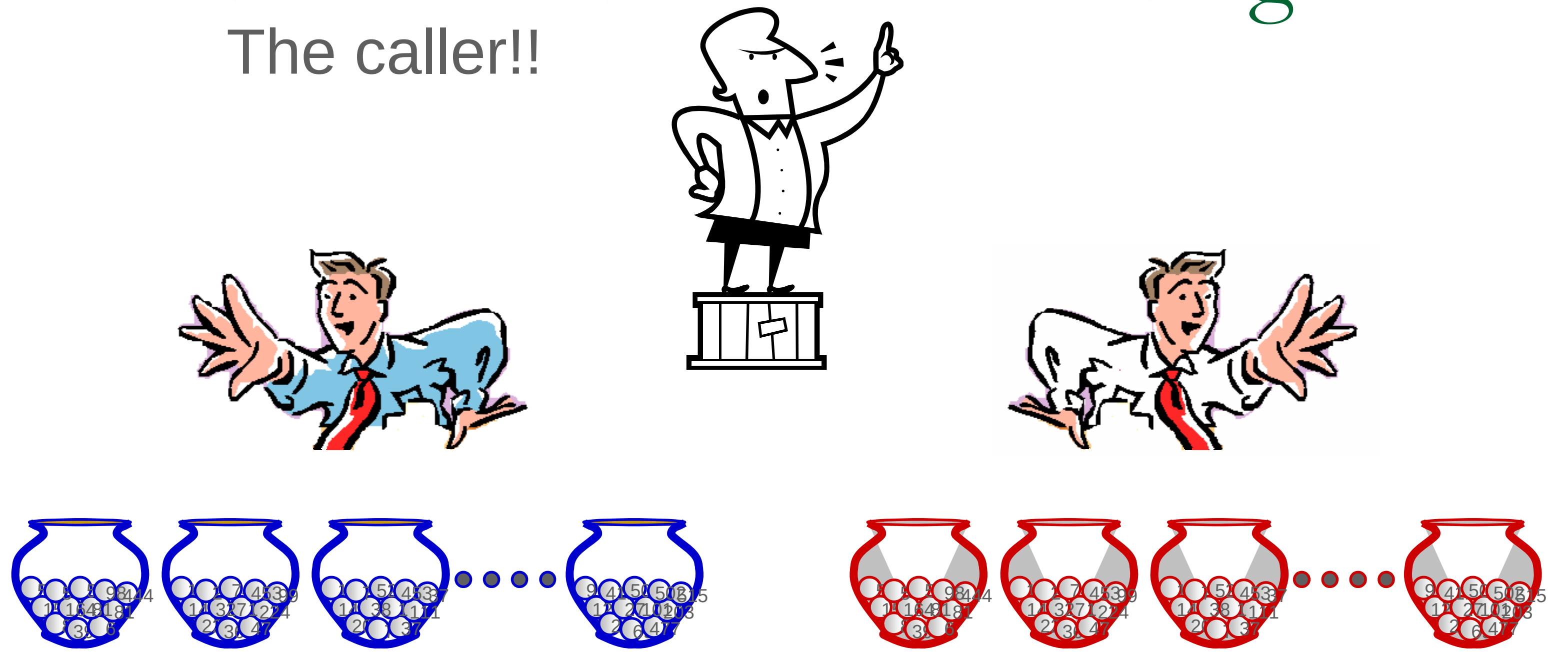
- The problem:
  - Multiple sources are producing sound simultaneously
  - The combined signals are recorded over a single microphone
  - The goal is to selectively separate out the signal for a target source in the mixture
    - Or at least to enhance the signals from a selected source

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



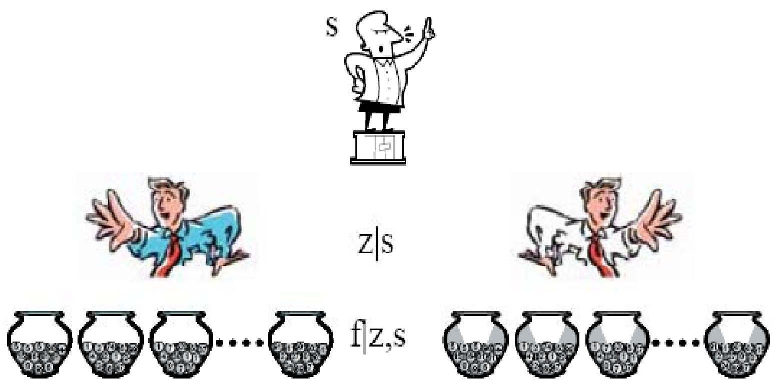
Ball-and-urn model for a mixed signal



- 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

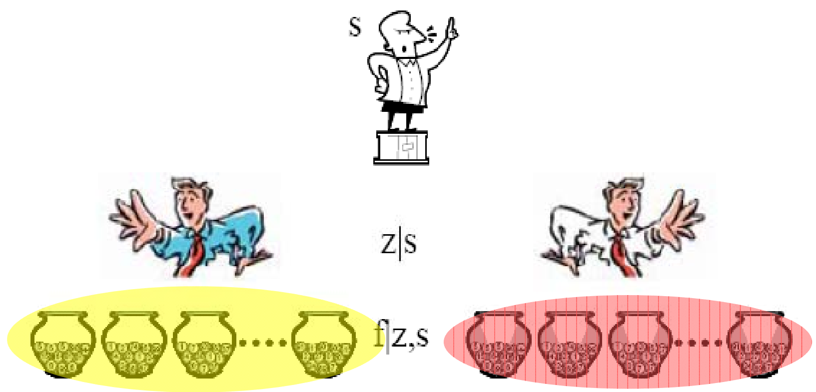


$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

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

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

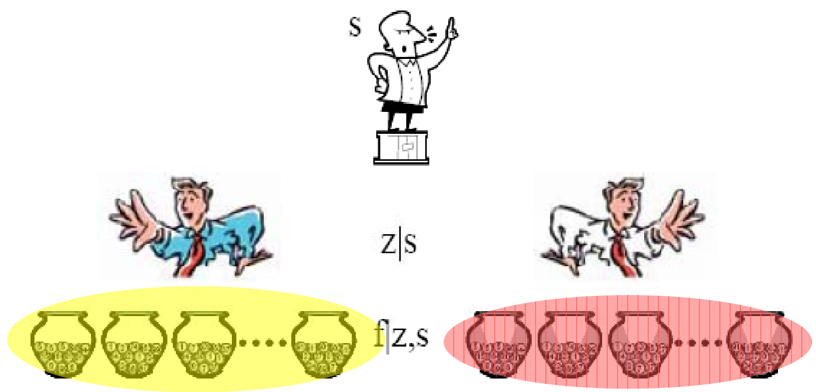


$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

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

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**



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

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

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

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

  - Compute the a priori weight of speaker s

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

  - Compute mixture weight of z^{th} urn for speaker s

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

What is $P_t(s,z|f)$

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

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

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

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

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

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

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

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

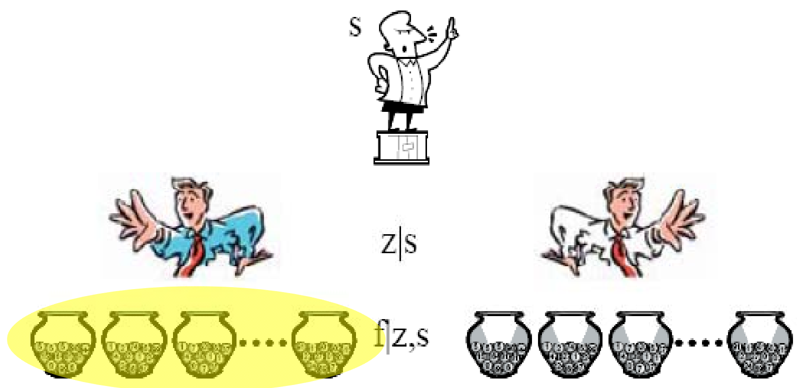
Separating the Sources

- 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

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

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

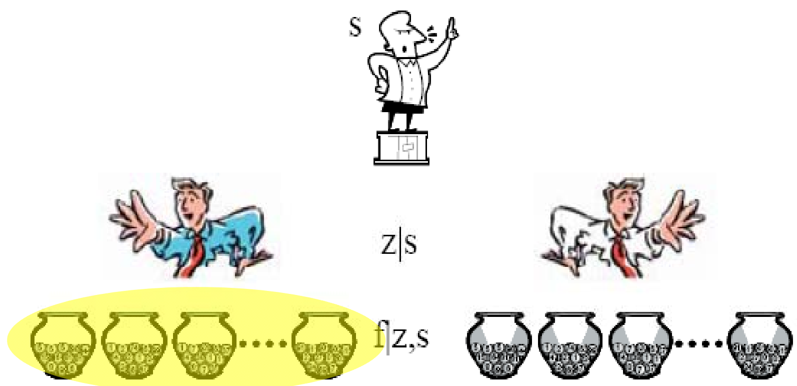


$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

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

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!



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

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

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

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

  - Compute the a priori weight of speaker s and mixture

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

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

  - Compute unknown bases

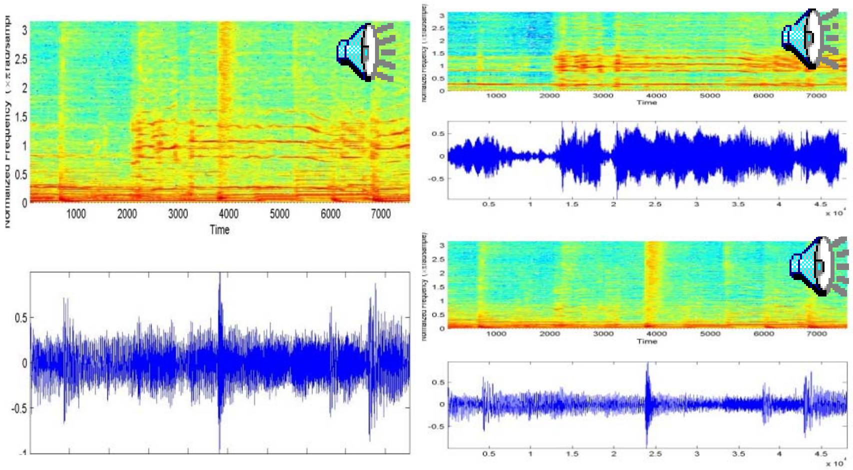
$$P(f|z,s) = \frac{\sum_t P_t(s,z|f)S_t(f)}{\sum_{f'} \sum_t P_t(s,z|f')S_t(f')}$$

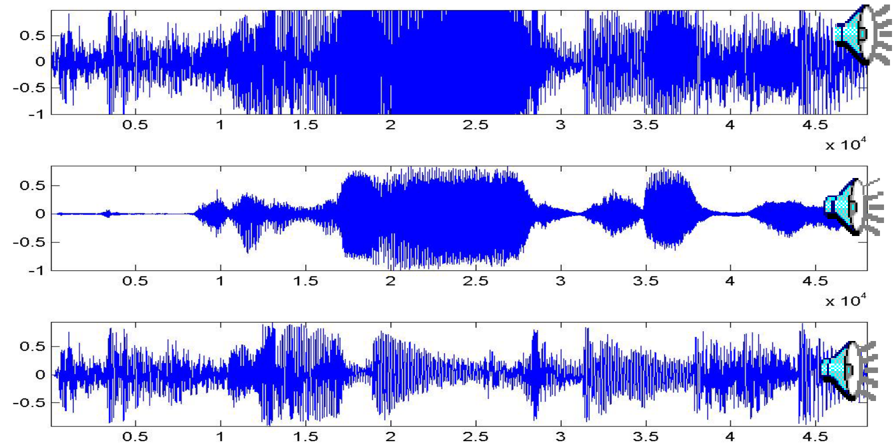
Partial information: bases for one source unknown

- 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





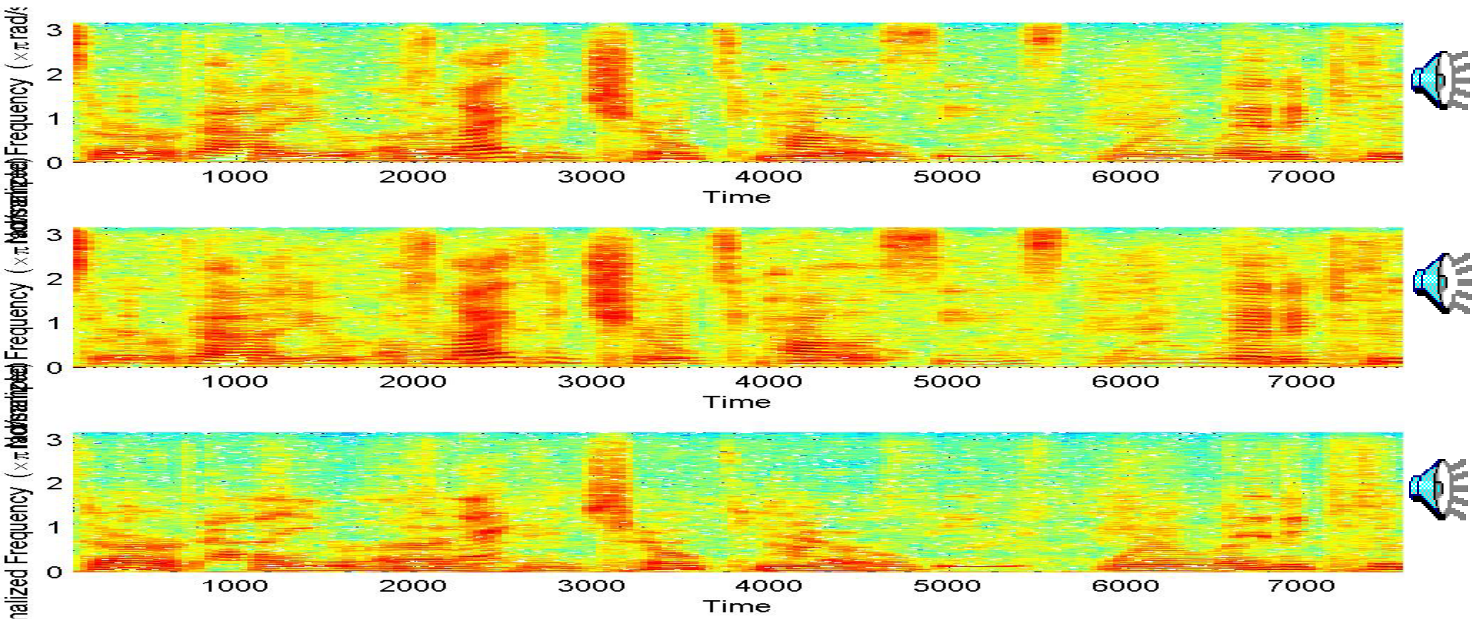
- "Raise my rent" by David Gilmour
- Background music "bases" learnt from 5-seconds of music-only segments within the song
- Lead guitar "bases" bases learnt from the rest of the song

- 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

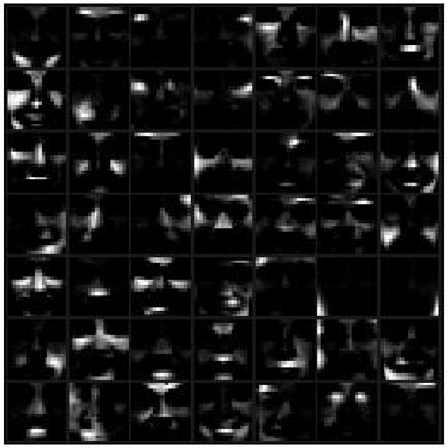
Separate overlapping speech



- 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



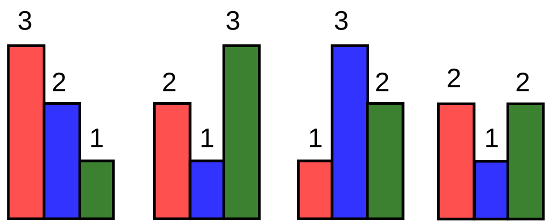
- We can use the same model to represent other data
- Images:
  - Every face in a collection is a histogram
  - Each histogram is composed from a mixture of a fixed number of multinomials
    - All faces are composed from the same multinomials, but the manner in which the multinomials are selected differs 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*

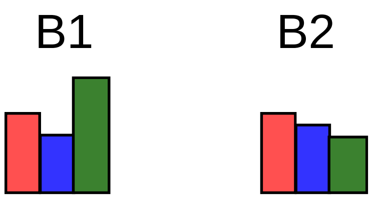
How many bases can we learn

- 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

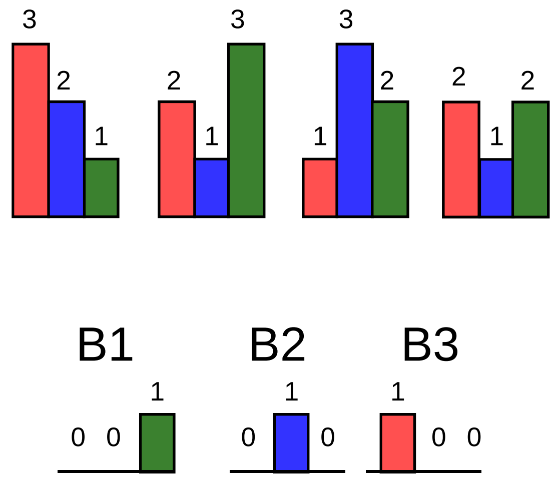
- Consider the four histograms to the right
- All of them are mixtures of the same K component multinomials
- For K < 3, a single global solution may exist
  - I.e there may be a unique set of component multinomials that explain all the multinomials
    - With error – model will not be perfect
- For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

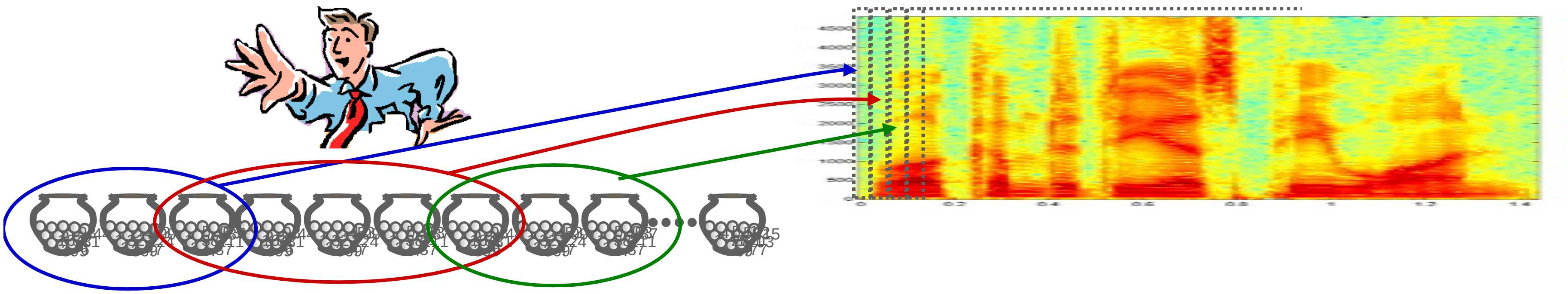
- In each case, the bases represent "typical unit structures"
  - Notes
  - Phonemes
  - Facial features..
- How many notes in music
  - Several octaves
  - Several instruments
- The typical sounds in speech –
  - Many phonemes, many variations, can number in the thousands
- Images:
  - Millions of units 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

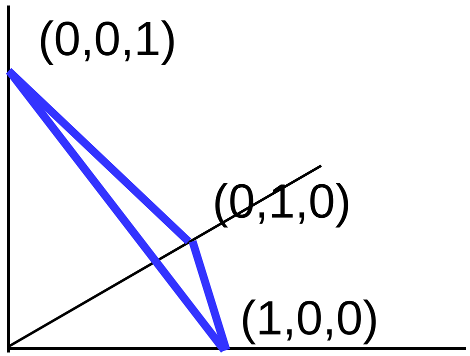
The *a priori* distribution

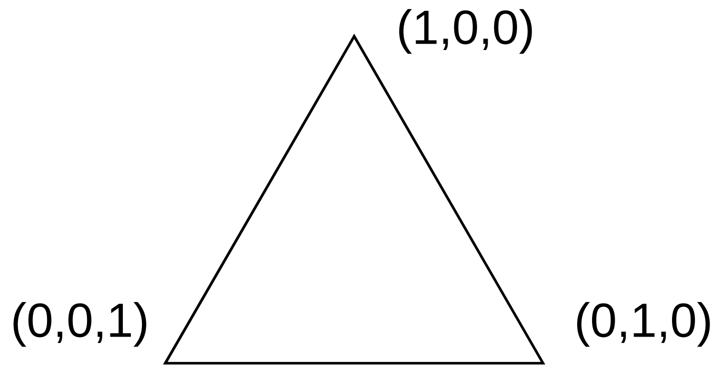
- A variety of *a priori* probability distributions all provide a bias towards "sparse" solutions

- The Dirichlet prior:
  - $P(W) = Z^* \prod_i w_i^{\alpha-1}$

- The entropic prior:
  - $P(W) = Z^*\exp(-\alpha H(W))$
    - $H(W)$ = entropy of $W = -\Sigma_i w_i \log(w_i)$

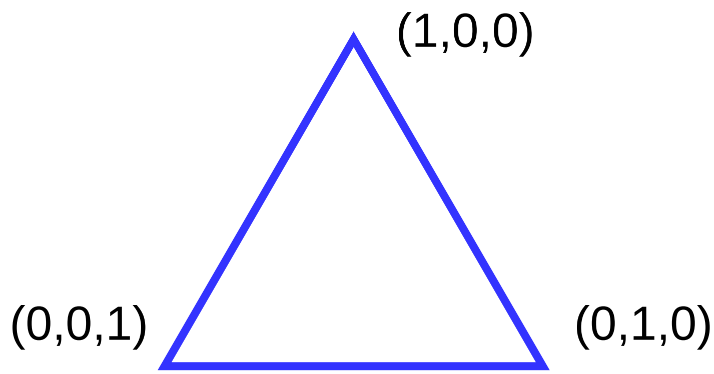
A simplex view of the world





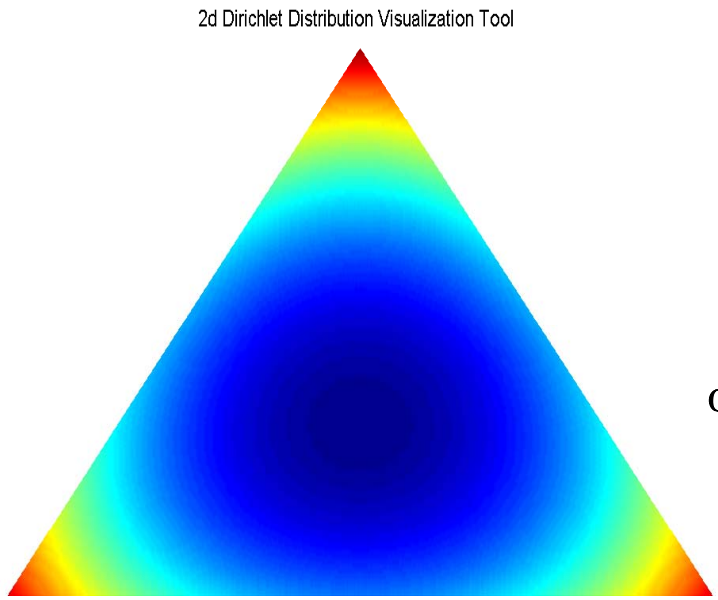
- The mixture weights are a probability distribution
  - $\Sigma_i\ w_i = 1.0$
- They can be viewed as a vector
  - $W = [w_0\ w_1\ w_2\ w_3\ w_4\ \ldots]$
  - The vector components are positive and sum to 1.0
- All probability vectors lie on a *simplex*
  - A convex region of a linear subspace in which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

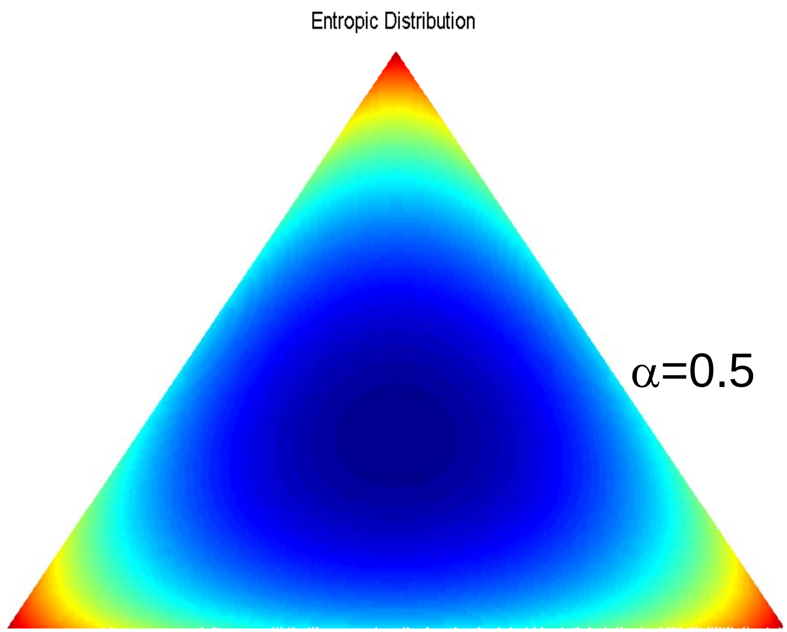


$P(W) = Z^* \prod_i w_i^{\alpha-1}$

α=0.5

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

Sparse Priors: The entropic prior



P(W) = Z*exp(-αH(W))

- 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

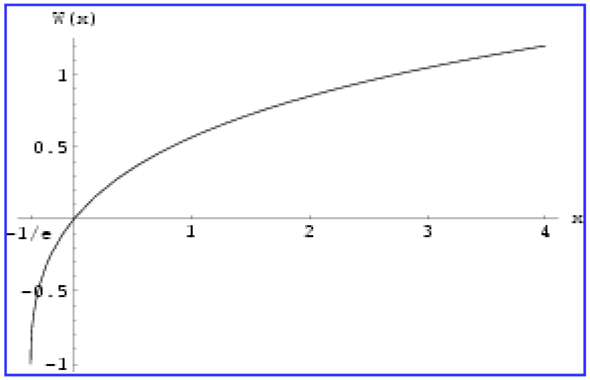
Lambert's W Function

- 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

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

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

- 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

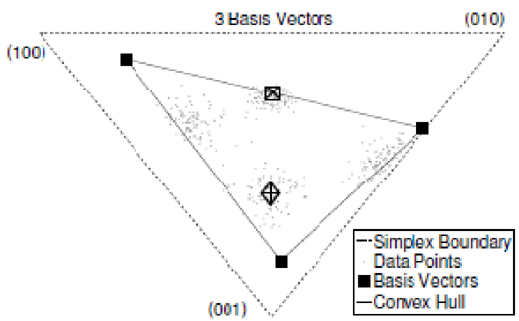
$$P_t(z \mid f) = \frac{P_t(z)P(f \mid z)}{\sum_{z'} P_t(z')P(f \mid z')}$$

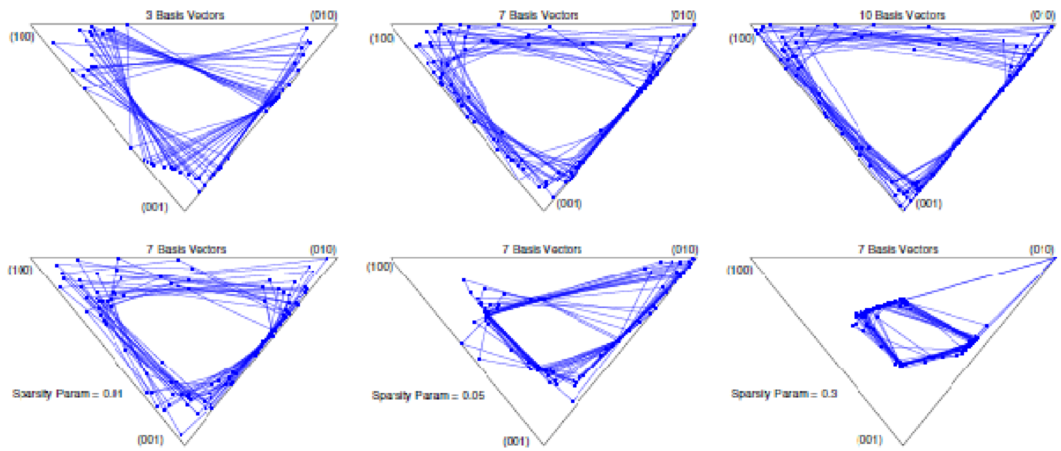
$$P(f \mid z) = \frac{\sum_t P_t(z \mid f)S_t(f)}{\sum_{f'}\sum_t P_t(z \mid f')S_t(f')}$$

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

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

A Simplex Example for Overcompleteness





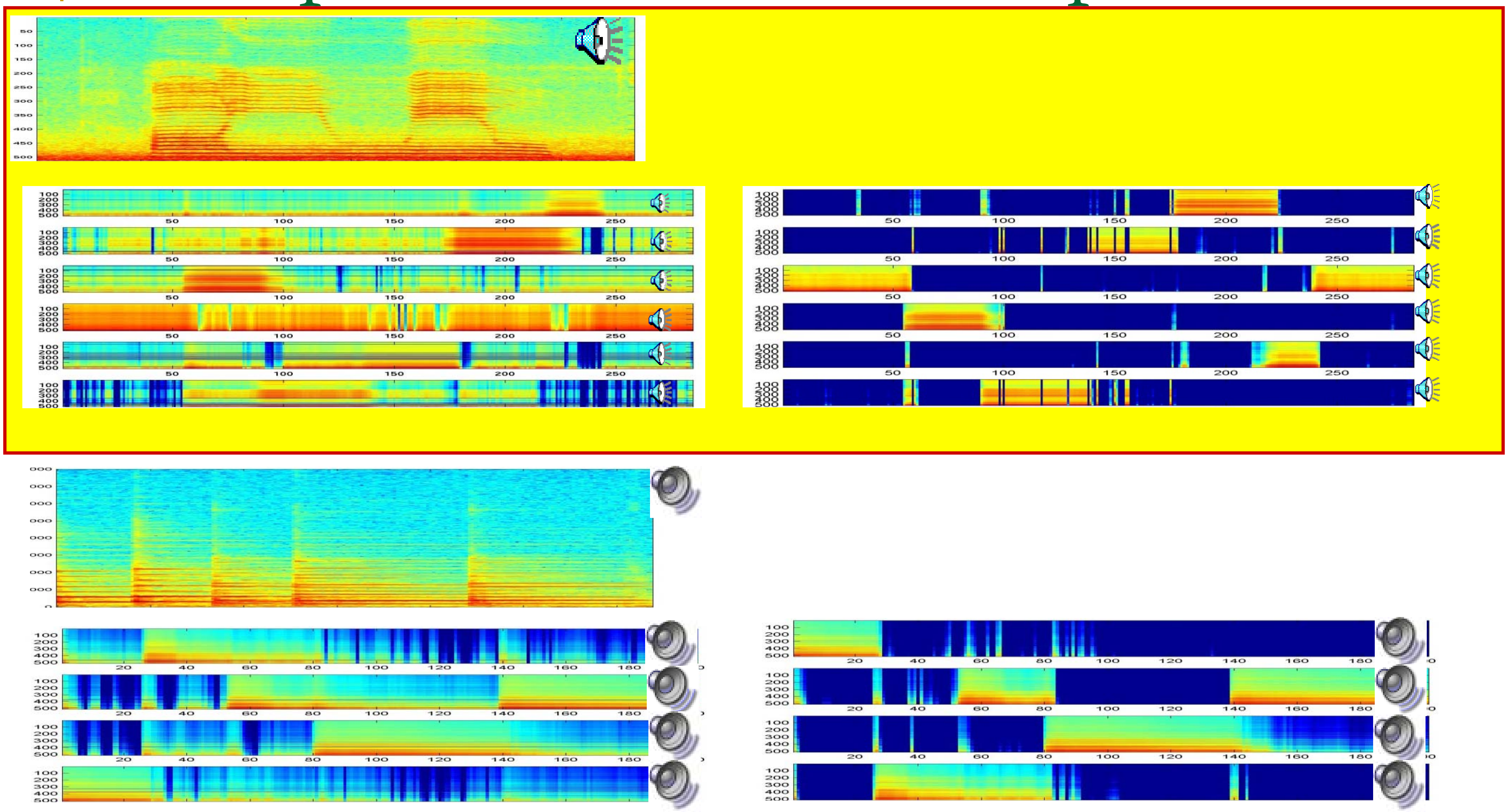
- Synthetic data: Four clusters of data within the probability simplex
- Regular learning with 3 bases learns an enclosing triangle
- Overcomplete solutions without sparsity restults in meaningless solutions
- Sparse overcomplete model captures the distribution of the data

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

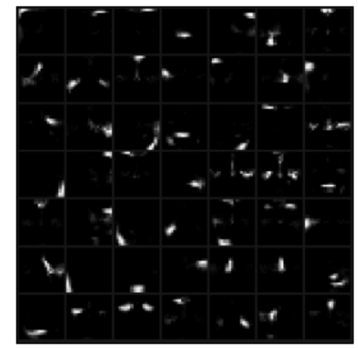
Face Data: The effect of sparsity

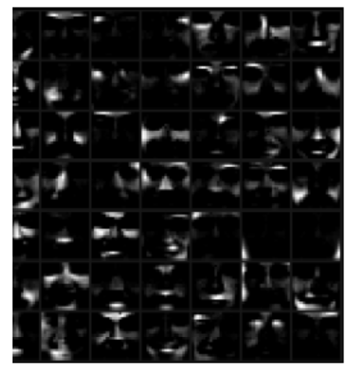
- As solutions get more sparse, bases become more informative
  - In the limit, each basis is a complete face by itself.
  - Mixture weights simply select face

- 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

High-entropy mixture weights

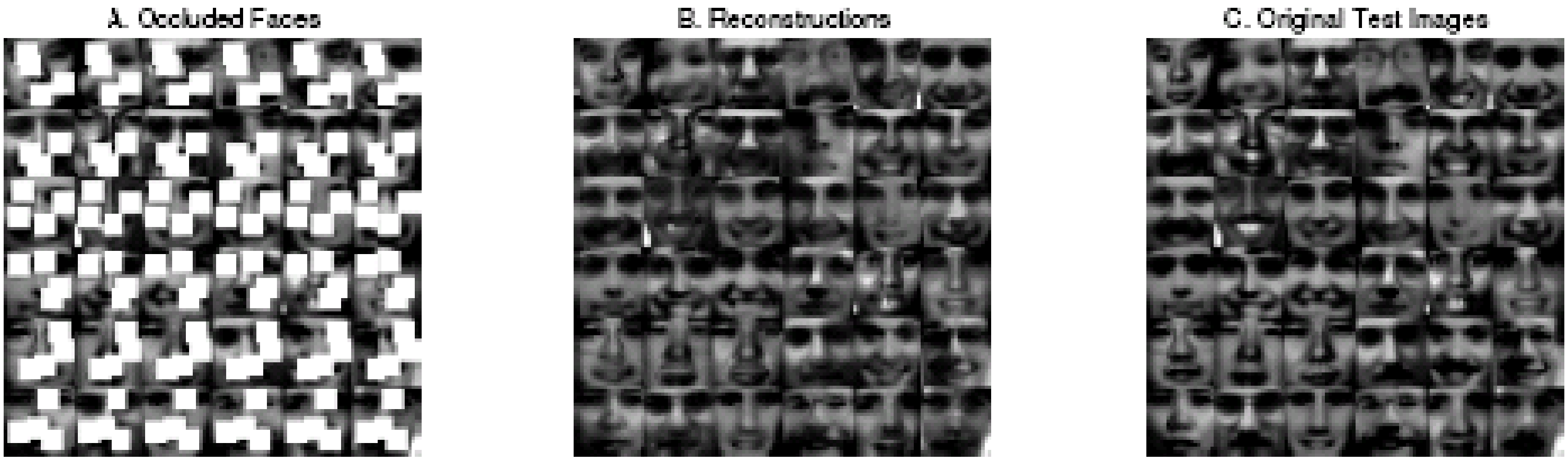


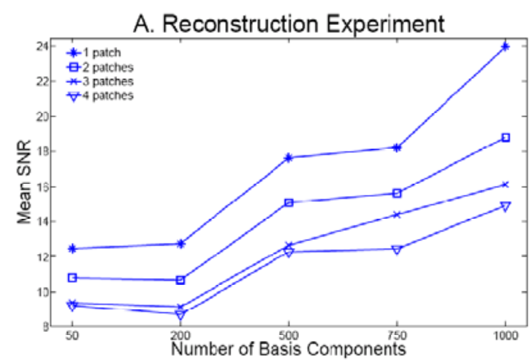


No sparsity

Sparse mixture weights

Benefit of overcompleteness





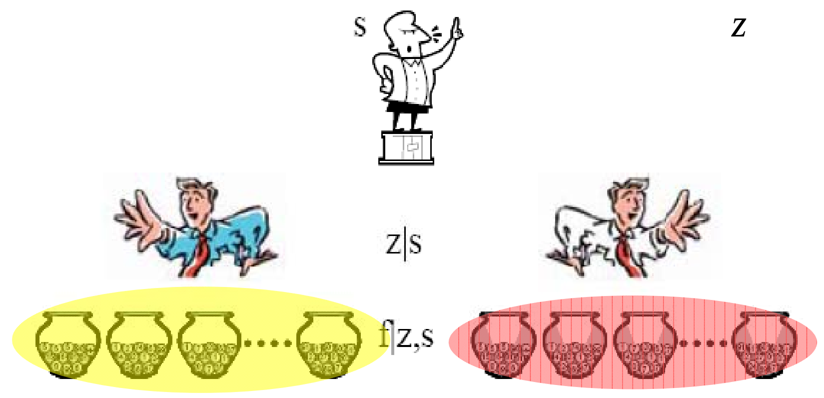
- 19x19 pixel images (361 pixels)
- Up to1000 bases trained from 2000 faces
- SNR of reconstruction from overcomplete basis set more than 10dB better than reconstruction from corresponding "compact" (regular) basis set

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



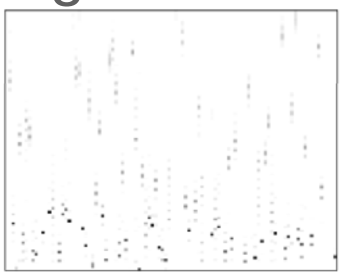
$$P_t(f) = P_t(s_1)P_t(f \mid s_1) + P_t(s_2)P_t(f \mid s_2)$$

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

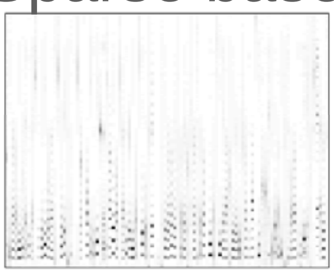
Sparse Overcomplete Bases: Separation

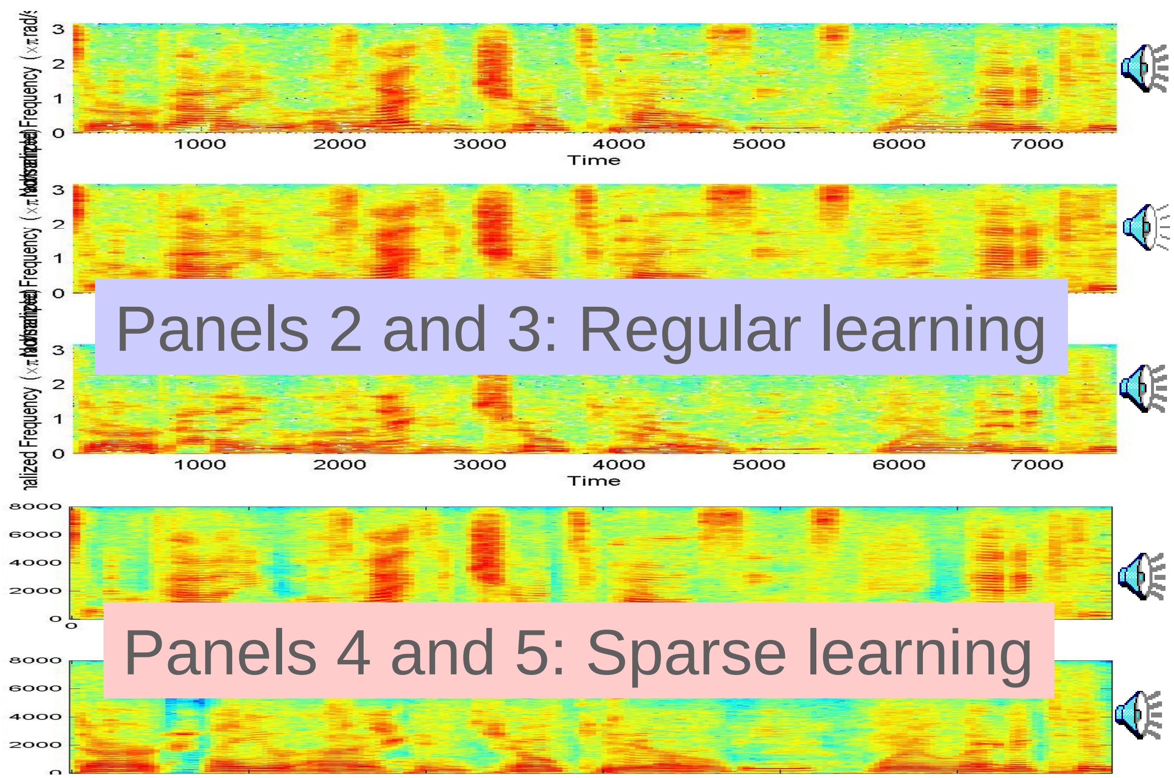
- 3000 bases for each of the speakers
  - The speaker-to-speaker ratio typically doubles (in dB) w.r.t "compact" bases

Regular bases



Sparse bases





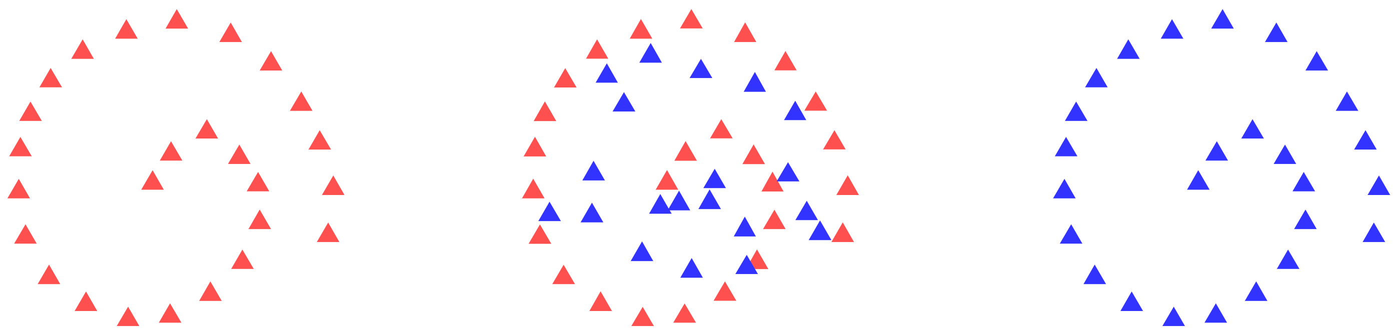
The Limits of Overcompleteness

- 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



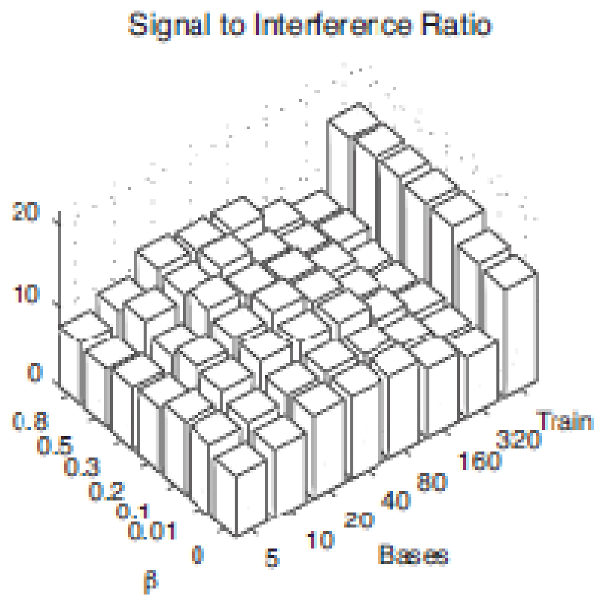
- In the above example all training data lie on the curve shown (Left Panel)
  - Each of them is a vector that sums to 1.0
- The learning procedure for bases learns multinomial components that are linear combinations of the data (Middle Panel)
  - These can lie anywhere within the area enclosed by the 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

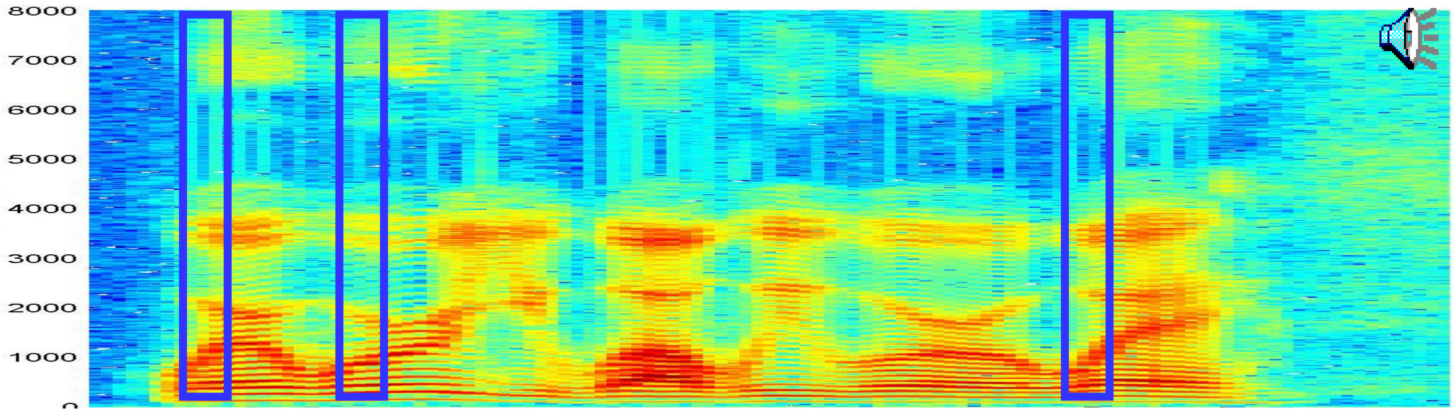
Example-based model: *All* the training data?

- 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

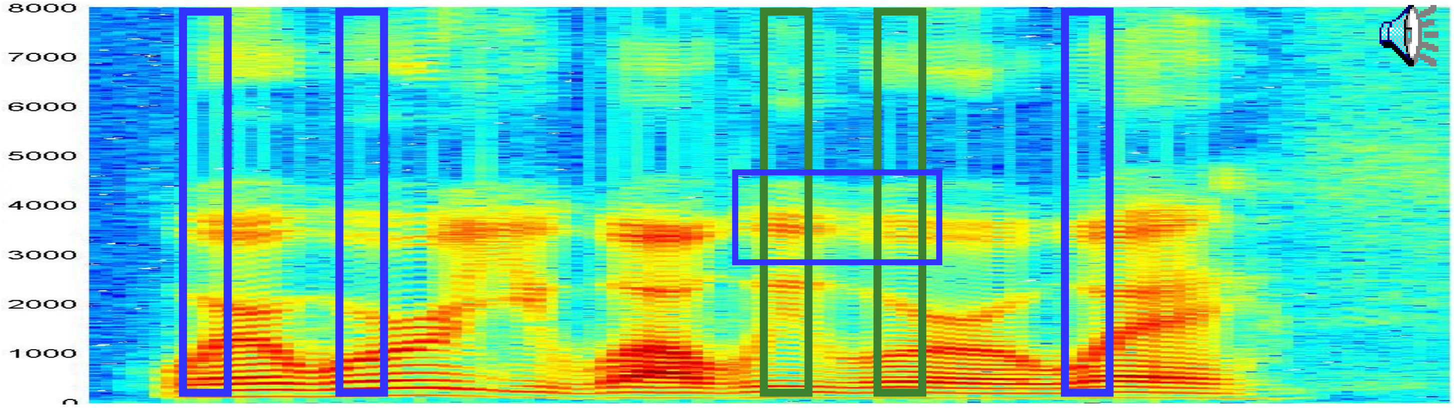
- PLCA:
  - The basic mixture-multinomial model for audio (and other data)
- Sparse Decomposition:
  - The notion of sparsity and how it can be imposed on learning
- Sparse Overcomplete Decomposition:
  - The notion of *overcomplete* basis set
- Example-based representations
  - Using the training data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

Next up: Shift/Transform Invariance



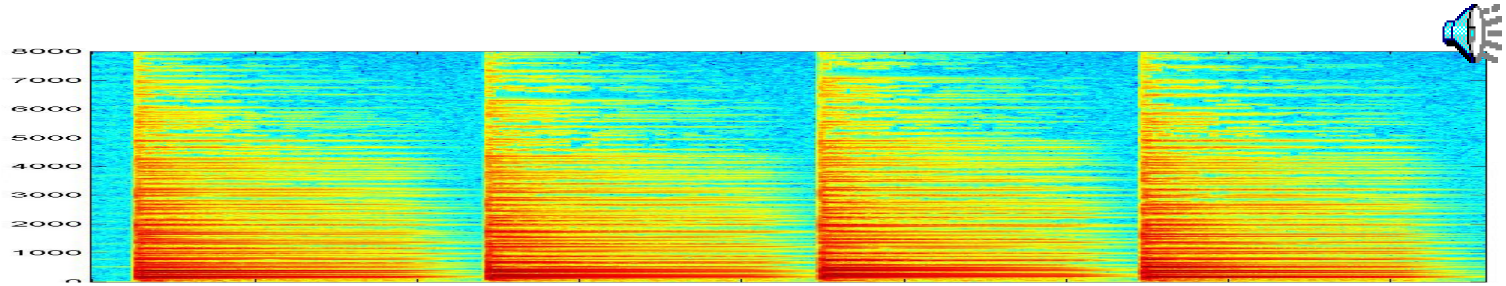
- Sometimes the "typical" structures that compose a sound are wider than one spectral frame
  - E.g. in the above example we note multiple examples of a pattern that spans several frames

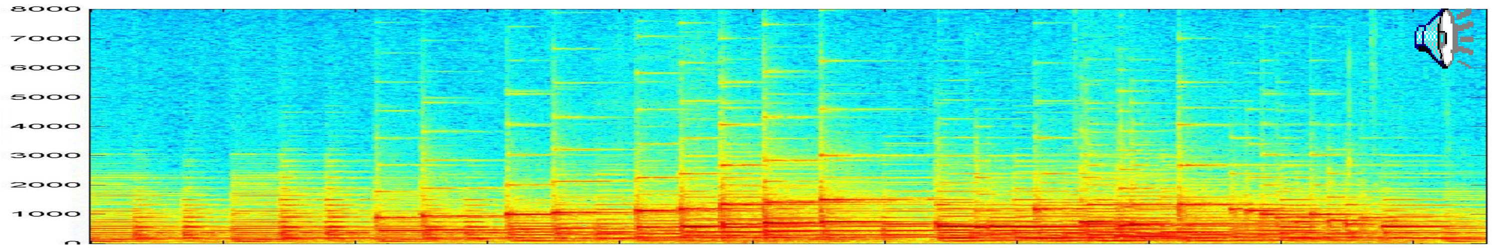
Next up: Shift/Transform Invariance



- Sometimes the "typical" structures that compose a sound are wider than one spectral frame
  - E.g. in the above example we note multiple examples of a pattern that spans several frames
- Multiframe patterns may also be local in frequency
  - E.g. the two green patches are similar only in the region enclosed by the blue box

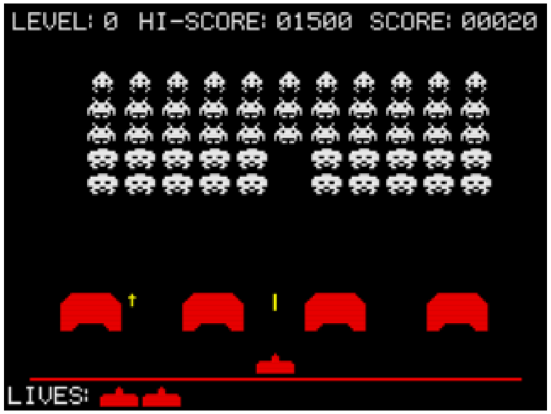
Patches are more representative than frames

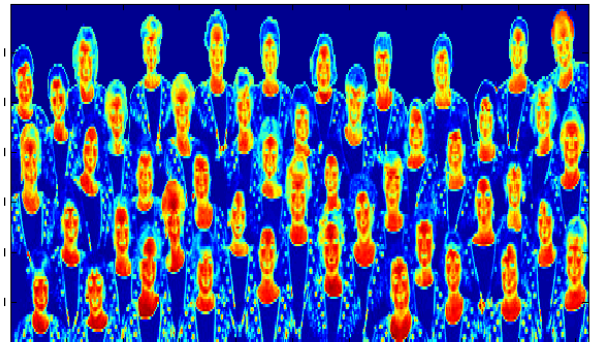




- Four bars from a music example
- The spectral patterns are actually patches
  - Not all frequencies fall off in time at the same rate
- The basic unit is a spectral patch, not a spectrum

Images: Patches often form the image



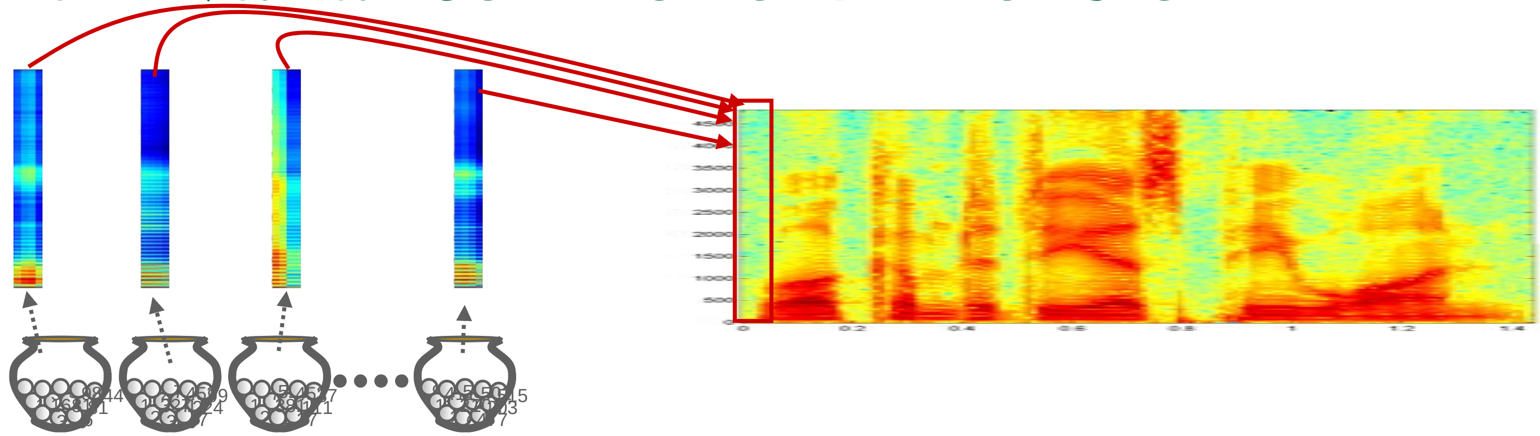


- A typical image component may be viewed as a patch
  - The alien invaders
  - Face like patches
  - A car like patch
    - overlaid on itself many times..

Shift-invariant modelling

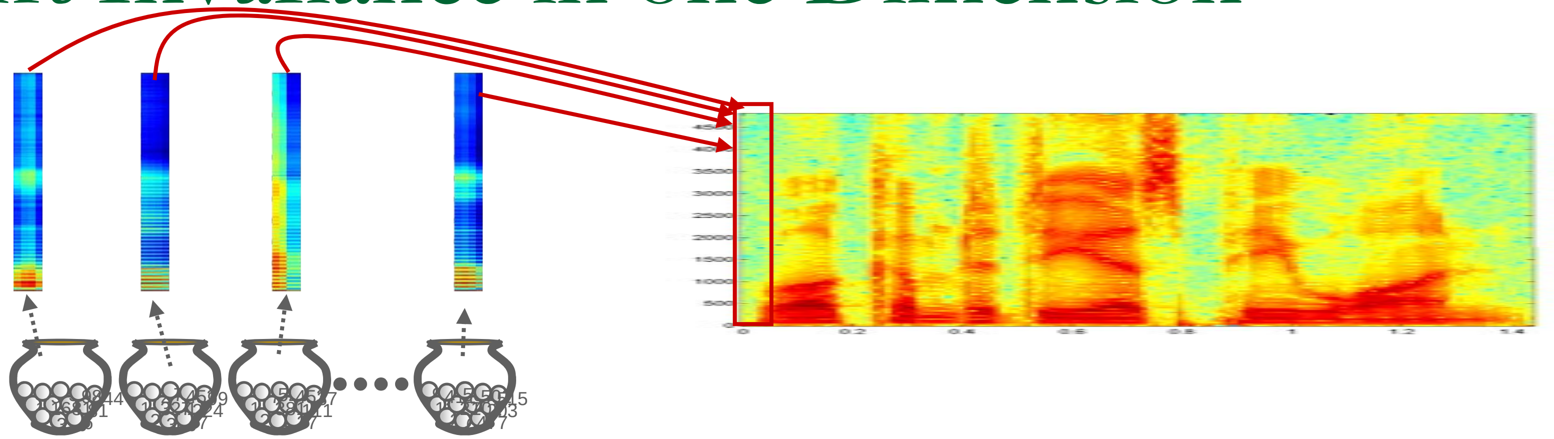
- A shift-invariant model permits individual bases to be *patches*
- Each patch composes the entire image.
- The data is a sum of the compositions from individual patches

Shift Invariance in one Dimension



- Our bases are now "patches"
  - Typical *spectro-temporal* structures
- The urns now represent patches
  - Each draw results in a (t,f) pair, rather than only f
  - *Also associated with each urn: A shift probability distribution P(T|z)*
- The overall drawing process is slightly more complex
- Repeat the following process:
  - Select an urn Z with a probability P(Z)
  - Draw a value T from P(t|Z)
  - Draw (t,f) pair from the urn
  - Add to the histogram at (t+T, f)

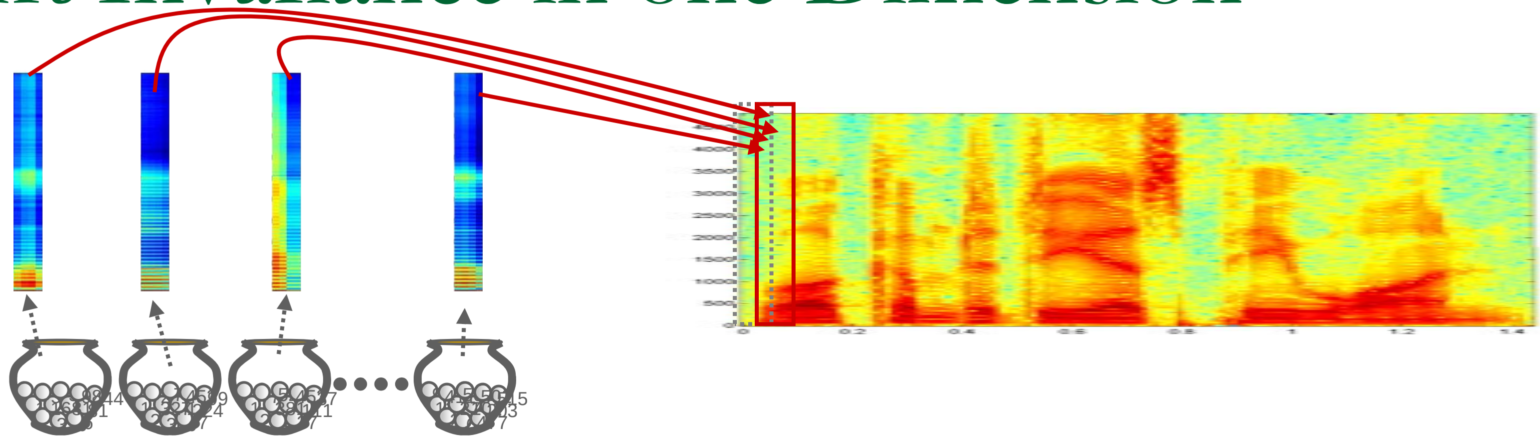
Shift Invariance in one Dimension



- The process is *shift-invariant* because the probability of drawing a shift P(T|Z) does not affect the probability of selecting urn Z

- Every location in the spectrogram has contributions from every urn patch

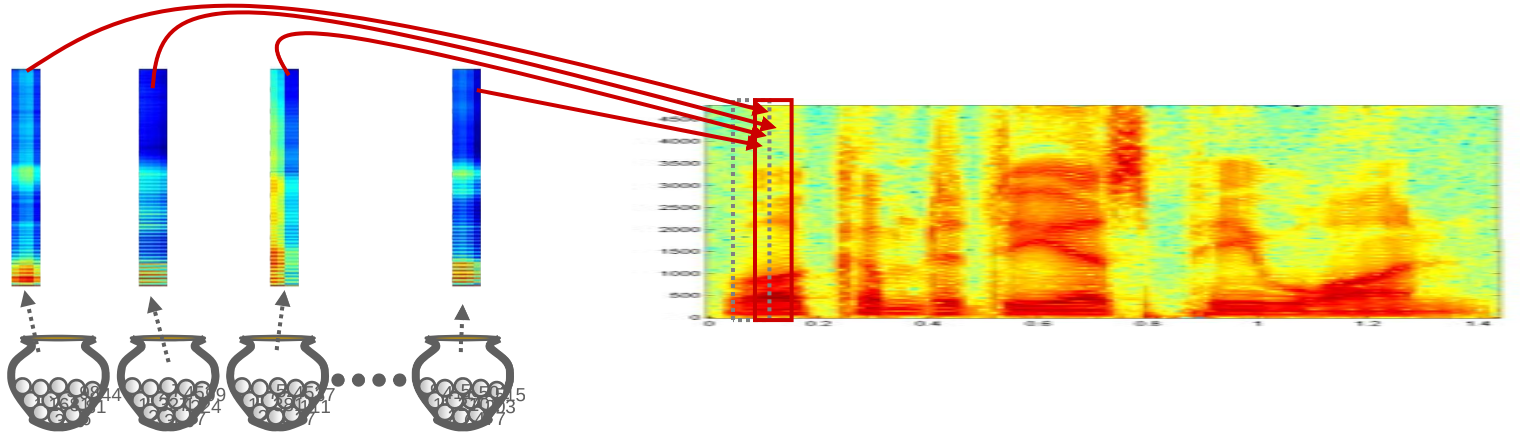
Shift Invariance in one Dimension



- The process is *shift-invariant* because the probability of drawing a shift P(T|Z) does not affect the probability of selecting urn Z

- Every location in the spectrogram has contributions from every urn patch

Shift Invariance in one Dimension



- The process is *shift-invariant* because the probability of drawing a shift P(T|Z) does not affect the probability of selecting urn Z
- Every location in the spectrogram has contributions from every urn patch

Probability of drawing a particular (t,f) combination

$$P(t,f) = \sum_z P(z) \sum_\tau P(\tau \mid z) P(t-\tau, f \mid z)$$

- The parameters of the model:
  - P(t,f|z) – the urns
  - P(T|z) – the *urn-specific* shift distribution
  - P(z) – probability of selecting an urn
- The ways in which (t,f) can be drawn:
  - Select any urn z
  - Draw T from the urn-specific shift distribution
  - Draw (t-T,f) from the urn
- The actual probability sums this over all shifts and urns

Learning the Model

- The parameters of the model are learned analogously to the manner in which mixture multinomials are learned

- Given observation of (t,f), it we knew which urn it came from and the shift, we could compute all probabilities by counting!
  - If shift is T and urn is Z
    - Count(Z) = Count(Z) + 1
    - For shift probability: Count(T|Z) = Count(T|Z)+1
    - For urn: Count(t-T,f | Z) = Count(t-T,f|Z) + 1
      - Since the value drawn from the urn was t-T,f

  - After all observations are counted:
    - Normalize Count(Z) to get P(Z)
    - Normalize Count(T|Z) to get P(T|Z)
    - Normalize Count(t,f|Z) to get P(t,f|Z)

- Problem: When learning the urns and shift distributions from a histogram, the urn (Z) and shift (T) for any draw of (t,f) is not known
  - These are unseen variables

Learning the Model

- Urn Z and shift T are unknown
  - So (t,f) contributes partial counts to *every* value of T and Z
  - Contributions are proportional to the *a posteriori* probability of Z and T,Z

$$P(t,f,Z) = P(Z)\sum_T P(T \mid Z)P(t-T,f \mid Z) \qquad P(T,t,f \mid Z) = P(T \mid Z)P(t-T,f \mid Z)$$

$$P(Z \mid t,f) = \frac{P(t,f,Z)}{\sum_{Z'} P(t,f,Z')} \qquad P(T \mid Z,t,f) = \frac{P(T,t-T,f \mid Z)}{\sum_{T'} P(T',t-T',f \mid Z)}$$

- Each observation of (t,f)
  - P(z|t,f) to the count of the total number of draws from the urn
    - Count(Z) = Count(Z) + P(z | t,f)
  - P(z|t,f)P(T | z,t,f) to the count of the shift T for the shift distribution
    - Count(T | Z) = Count(T | Z) + P(z|t,f)P(T | Z, t, f)
  - P(z|t,f)P(T | z,t,f) to the count of (t-T, f) for the urn
    - Count(t-T,f | Z) = Count(t-T,f | Z) + P(z|t,f)P(T | z,t,f)

Shift invariant model: Update Rules

- Given data (spectrogram) S(t,f)
- Initialize P(Z), P(T|Z), P(t,f | Z)
- Iterate

$$P(t,f,Z) = P(Z)\sum_T P(T|Z)P(t-T,f|Z) \qquad P(T,t,f|Z) = P(T|Z)P(t-T,f|Z)$$

$$P(Z|t,f) = \frac{P(t,f,Z)}{\sum_{Z'} P(t,f,Z')} \qquad P(T|Z,t,f) = \frac{P(T,t-T,f|Z)}{\sum_{T'} P(T',t-T',f|Z)}$$

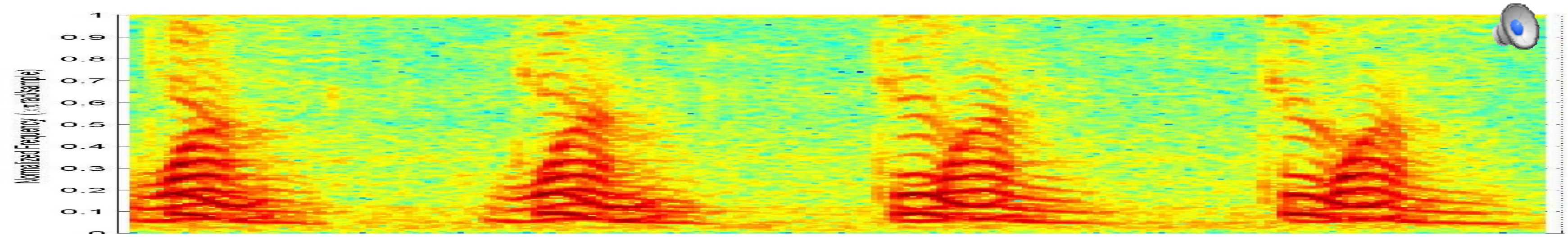
$$P(Z) = \frac{\sum_t\sum_f P(Z|t,f)S(t,f)}{\sum_{Z'}\sum_t\sum_f P(Z'|t,f)S(t,f)} \qquad P(T|Z) = \frac{\sum_t\sum_f P(Z|t,f)P(T|Z,t,f)S(t,f)}{\sum_{T'}\sum_t\sum_f P(Z|t,f)P(T'|Z,t,f)S(t,f)}$$

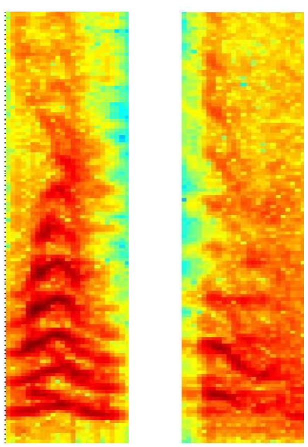
$$P(t,f|Z) = \frac{\sum_T P(Z|T,f)P(T-t|Z,T,f)S(T,f)}{\sum_{t'}\sum_T P(Z|T,f)P(T-t'|Z,T,f)S(T,f)}$$

Shift-invariance in one time: example

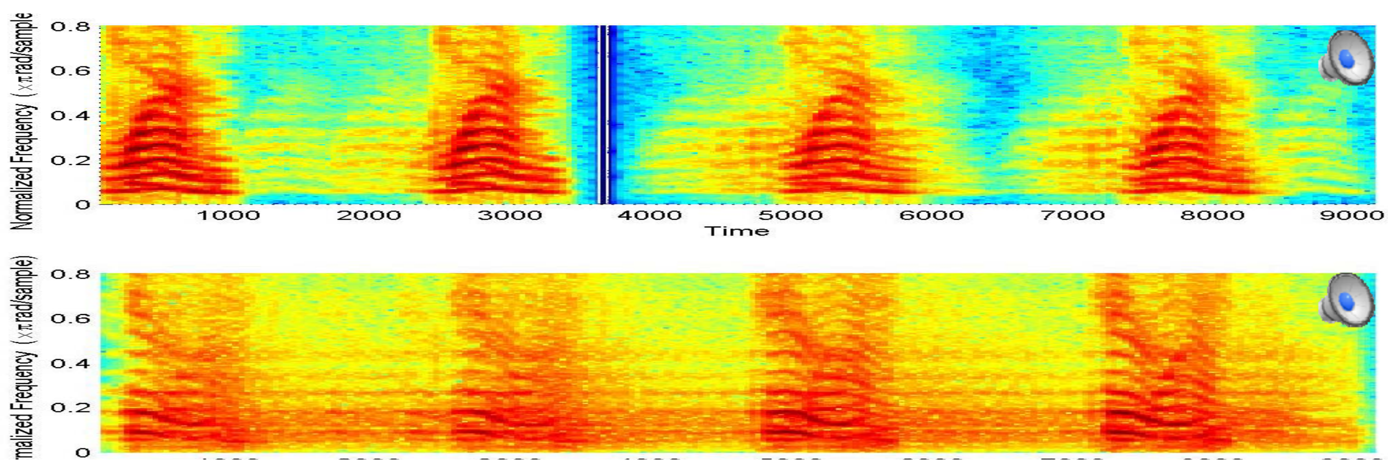
- An Example: Two distinct sounds occuring with different repetition rates within a signal
  - Modelled as being composed from two time-frequency bases
  - NOTE: Width of patches must be specified

INPUT SPECTROGRAM



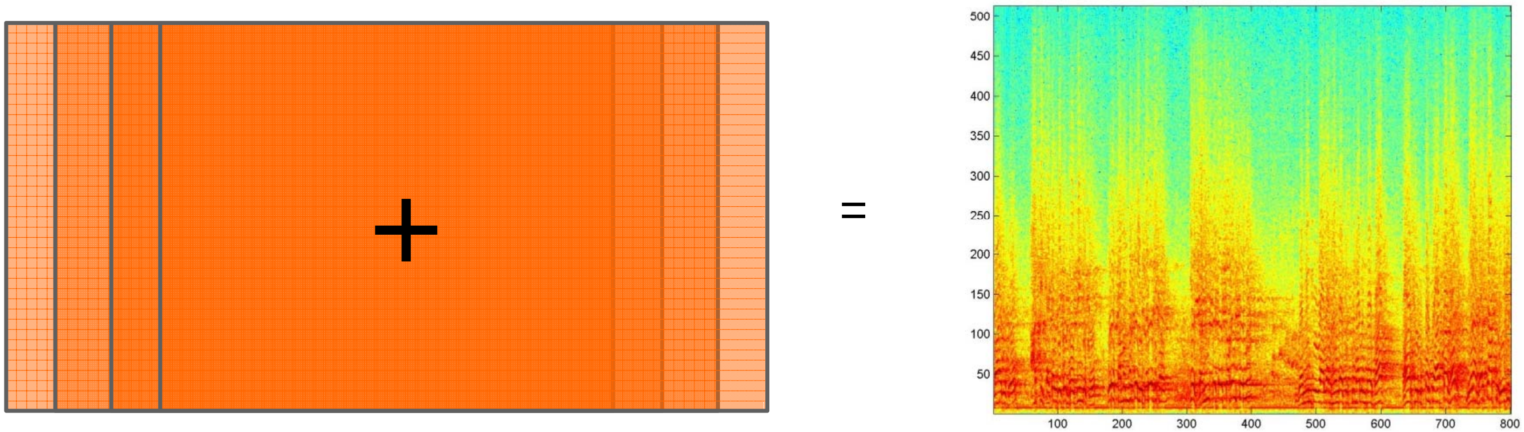


Discovered time-frequency “patch” bases (urns)



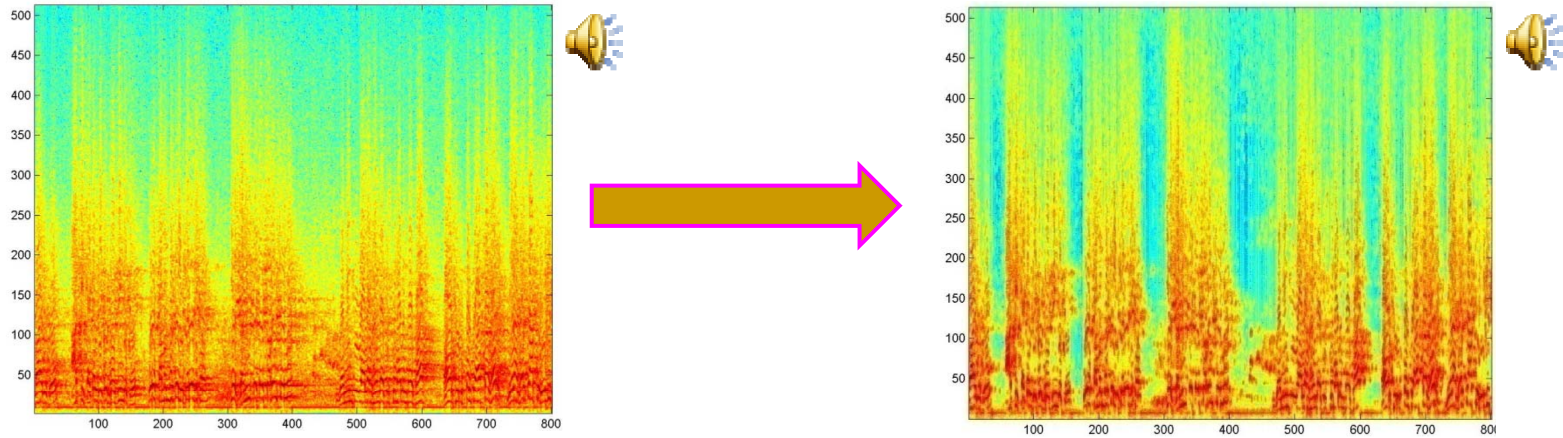
Contribution of individual bases to the recording

Shift Invariance in Time: Dereverberation



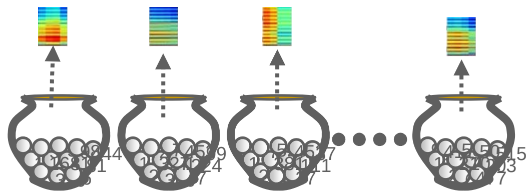
- Reverberation – a simple model
  - The Spectrogram of the reverberated signal is a sum of the spectrogram of the clean signal and several shifted and scaled versions of itself
  - A convolution of the spectrogram and a room response

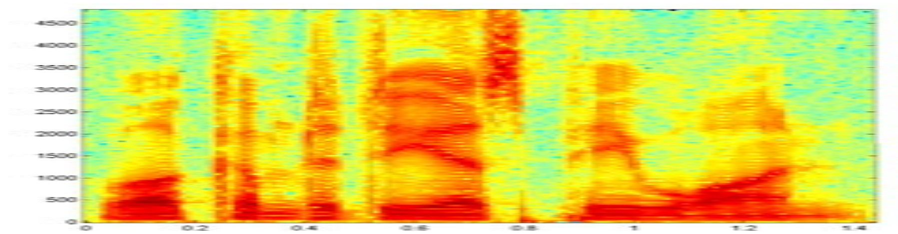
Dereverberation



- Given the spectrogram of the reverberated signal:
  - Learn a shift-invariant model with a single patch basis
    - Sparsity must be enforced on the basis
  - The “basis” represents the clean speech!

Shift Invariance in Two Dimensions





- We now have urn-specific shifts along both T and F
- The Drawing Process
  - Select an urn Z with a probability P(Z)
  - Draw SHIFT values (T,F) from $P_s(T,F|Z)$
  - Draw (t,f) pair from the urn
  - Add to the histogram at (t+T, f+F)

- This is a two-dimensional shift-invariant model
  - We have shifts in both time and frequency
    - Or, more generically, along both axes

Learning the Model

- Learning is analogous to the 1-D case

- Given observation of (t,f), it we knew which urn it came from and the shift, we could compute all probabilities by counting!
  - If shift is T,F and urn is Z
    - Count(Z) = Count(Z) + 1
    - For shift probability: ShiftCount(T,F|Z) = ShiftCount(T,F|Z)+1
    - For urn: Count(t-T,f-F | Z) = Count(t-T,f-F|Z) + 1
      - Since the value drawn from the urn was t-T,f-F

  - After all observations are counted:
    - Normalize Count(Z) to get P(Z)
    - Normalize ShiftCount(T,F|Z) to get P_s(T,F|Z)
    - Normalize Count(t,f|Z) to get P(t,f|Z)

- Problem: Shift and Urn are unknown

Learning the Model

- Urn Z and shift T,F are unknown
  - So (t,f) contributes partial counts to *every* value of T,F and Z
  - Contributions are proportional to the *a posteriori* probability of Z and T,F|Z

$$P(t,f,Z) = P(Z)\sum_{T,F} P(T,F \mid Z)P(t-T,f-F \mid Z) \qquad P(T,F,t,f \mid Z) = P(T,F \mid Z)P(t-T,f-F \mid Z)$$

$$P(Z \mid t,f) = \frac{P(t,f,Z)}{\sum_{Z'} P(t,f,Z')} \qquad P(T,F \mid Z,t,f) = \frac{P(T,F,t-T,f-F \mid Z)}{\sum_{T',F'} P(T',F',t-T',f-F' \mid Z)}$$

- Each observation of (t,f)
  - P(z|t,f) to the count of the total number of draws from the urn
    - Count(Z) = Count(Z) + P(z | t,f)
  - P(z|t,f)P(T,F | z,t,f) to the count of the shift T,F for the shift distribution
    - ShiftCount(T,F | Z) = ShiftCount(T,F | Z) + P(z|t,f)P(T | Z, t, f)
  - P(T | z,t,f) to the count of (t-T, f-F) for the urn
    - Count(t-T,f-F | Z) = Count(t-T,f-F | Z) + P(z|t,f)P(t-T,f-F | z,t,f)

Shift invariant model: Update Rules

- Given data (spectrogram) S(t,f)
- Initialize P(Z), P_s(T,F|Z), P(t,f | Z)
- Iterate

$$P(t,f,Z) = P(Z)\sum_{T,F} P(T,F\,|\,Z)P(t-T, f-F\,|\,Z) \qquad P(T,F,t,f\,|\,Z) = P(T,F\,|\,Z)P(t-T,f-F\,|\,Z)$$

$$P(Z\,|\,t,f) = \frac{P(t,f,Z)}{\sum_{Z'} P(t,f,Z')} \qquad P(T,F\,|\,Z,t,f) = \frac{P(T,F,t-T,f-F\,|\,Z)}{\sum_{T',F'} P(T',F',t-T',f-F'\,|\,Z)}$$

$$P(Z) = \frac{\sum_t \sum_f P(Z\,|\,t,f)S(t,f)}{\sum_{Z'}\sum_t\sum_f P(Z'\,|\,t,f)S(t,f)} \qquad P(T,F\,|\,Z) = \frac{\sum_t\sum_f P(Z\,|\,t,f)P(T,F\,|\,Z,t,f)S(t,f)}{\sum_{T'}\sum_{F'}\sum_t\sum_f P(Z\,|\,t,f)P(T',F'\,|\,Z,t,f)S(t,f)}$$

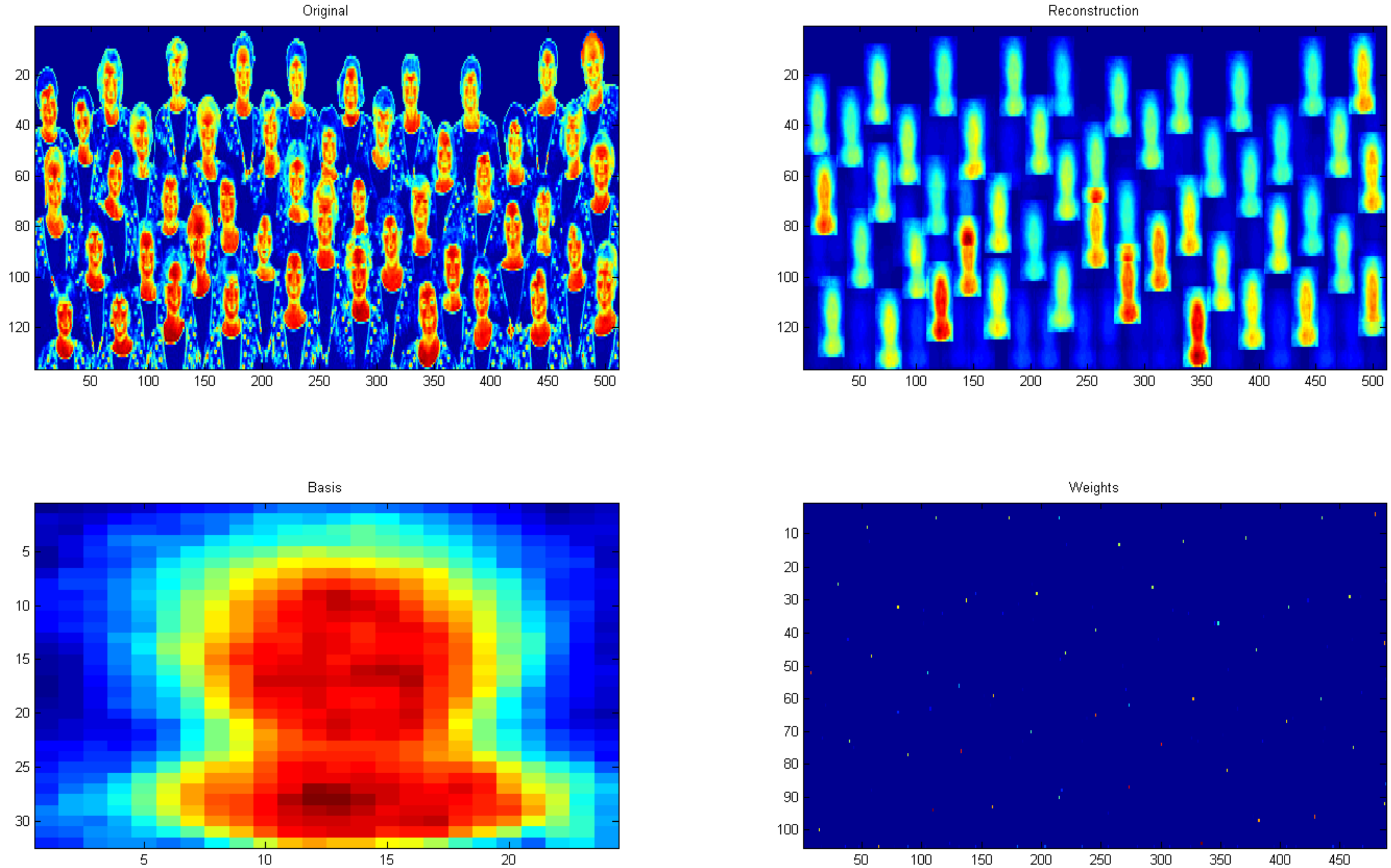
$$P(t,f\,|\,Z) = \frac{\sum_{T,F} P(Z\,|\,T,F)P(T-t,F-f\,|\,Z,T,F)S(T,F)}{\sum_{t',f'}\sum_{T,F} P(Z\,|\,T,F)P(T-t',F-f'\,|\,Z,T,F)S(T,F)}$$

2D Shift Invariance: The problem of indeterminacy

- $P(t,f|Z)$ and $P_s(T,F|Z)$ are analogous
  - Difficult to specify which will be the "urn" and which the "shift"

- Additional constraints required to ensure that one of them is clearly the shift and the other the urn
- Typical solution: Enforce sparsity on $P_s(T,F|Z)$
  - The patch represented by the urn occurs only in a few locations in the data

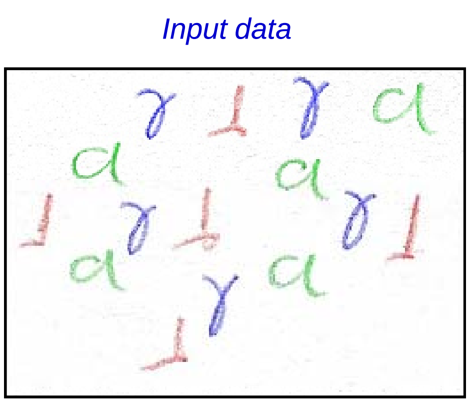
Example: 2-D shift invariance

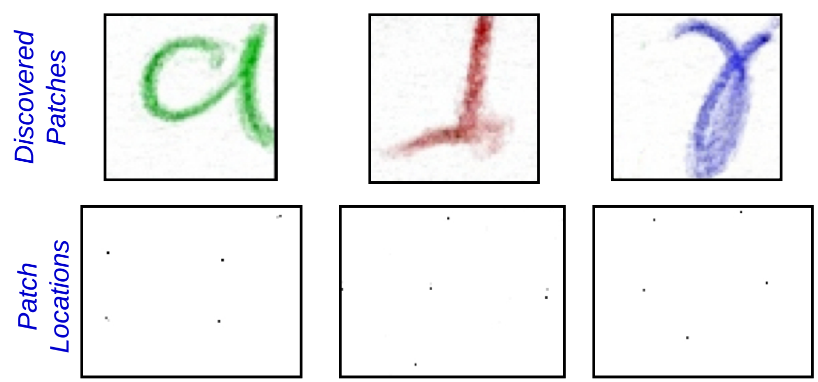


- Only one “patch” used to model the image (i.e. a single urn)
  - The learnt urn is an “average” face, the learned shifts show the locations of faces

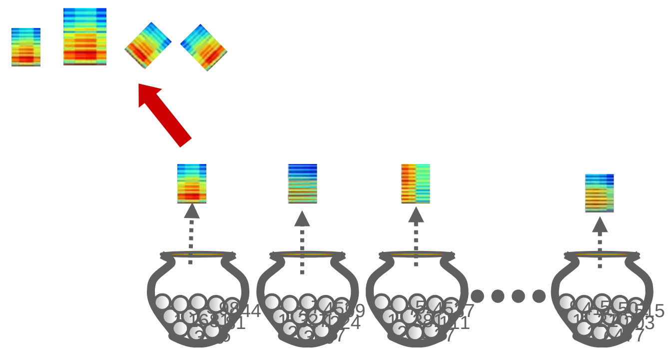
Example: 2-D shift invarince

- The original figure has multiple handwritten renderings of three characters
  - In different colours
- The algorithm learns the three characters and identifies their locations in the figure





Beyond shift-invariance: transform invariance



- The draws from the urns may not only be shifted, but also transformed
- The arithmetic remains very similar to the shift-invariant model
  - We must now impose one of an enumerated set of transforms to (t,f), after shifting them by (T,F)
  - In the estimation, the precise transform applied is an unseen variable

Transform invariance: Generation

- The set of transforms is enumerable
  - E.g. scaling by 0.9, scaling by 1.1, rotation right by 90degrees, rotation left by 90 degrees, rotation by 180 degrees, reflection
  - Transformations can be chosen by draws from a distribution over transforms
    - E.g. P(rotation by 90 degrees) = 0.2..
    - Distributions are URN SPECIFIC

- The drawing process:
  - Select an urn Z (patch)
  - Select a shift (T,F) from P_s(T, F| Z)
  - Select a transform from P(txfm | Z)
  - Select a (t,f) pair from P(t,f | Z)
  - *Transform* (t,f) to txfm(t,f)
  - Increment the histogram at txfm(t,f) + (T,F)

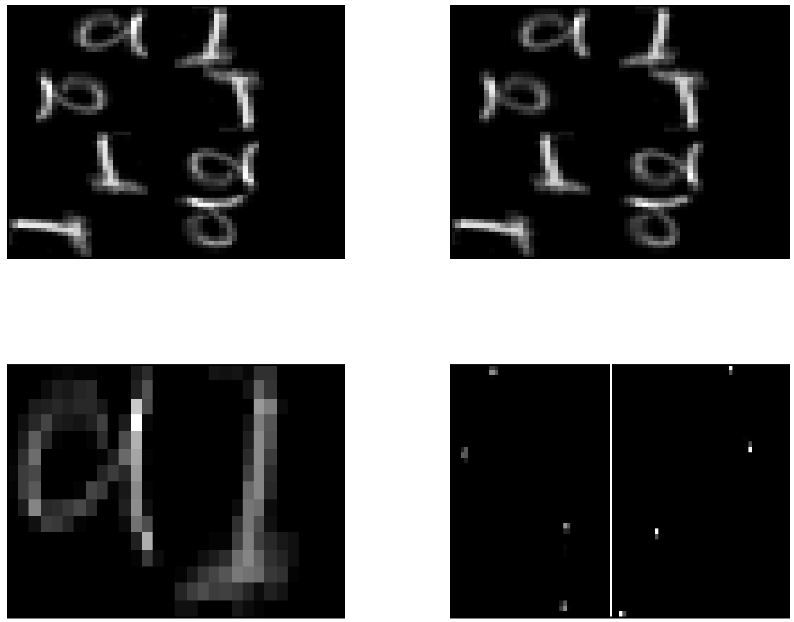
Transform invariance

- The learning algorithm must now estimate
  - P(Z) – probability of selecting urn/patch in any draw
  - P(t,f|Z) – the urns / patches
  - P(txfm | Z) – the urn specific distribution over transforms
  - P_s(T,F|Z) – the urn-specific shift distribution

- Essentially determines what the basic shapes are, where they occur in the data and how they are transformed
- The mathematics for learning are similar to the maths for shift invariance
  - With the addition that each instance of a draw must be fractured into urns, shifts AND transforms

- Details of learning are left as an exercise
  - Alternately, refer to Madhusudana Shashanka's PhD thesis at BU

Example: Transform Invariance



- Top left: Original figure
- Bottom left – the two bases discovered
- Bottom right –
  - Left panel, positions of “a”
  - Right panel, positions of “l”
- Top right: estimated distribution underlying original figure

Transform invariance: model limitations and extensions

- The current model only allows *one* transform to be applied at any draw
  - E.g. a basis may be rotated or scaled, but not scaled *and* rotated
- An obvious extension is to permit combinations of transformations
  - Model must be extended to draw the combination from some distribution
- Data dimensionality: All examples so far assume only *two* dimensions (e.g. in spectrogram or image)
- The models are trivially extended to higher-dimensional data

Transform Invariance: Uses and Limitations

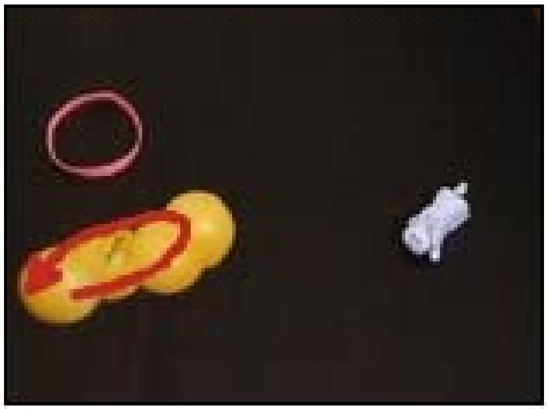
- Not very useful to analyze audio
- May be used to analyze images and video

- Main restriction: Computational complexity
  - Requires unreasonable amounts of memory and CPU
  - Efficient implementation an open issue

Example: Higher dimensional data

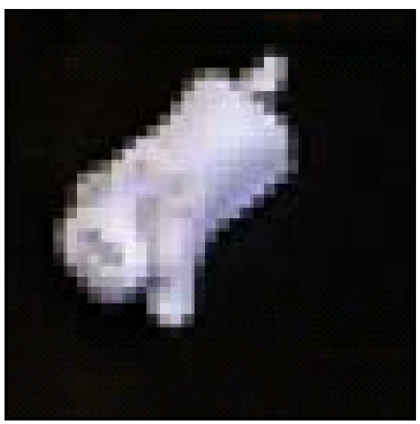
- Video example

Description of Input



Kernel 1

Kernel 2



Kernel 3