

Machine Learning for Signal Processing

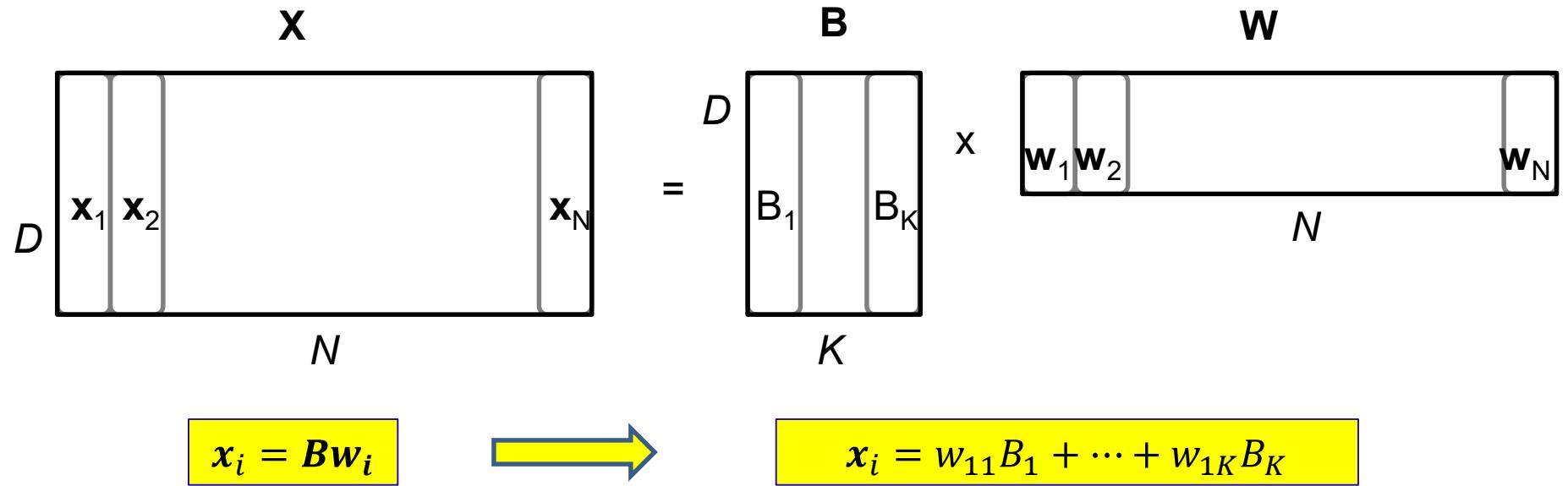
Non-negative Matrix Factorization

Instructor: Bhiksha Raj

11755/18797

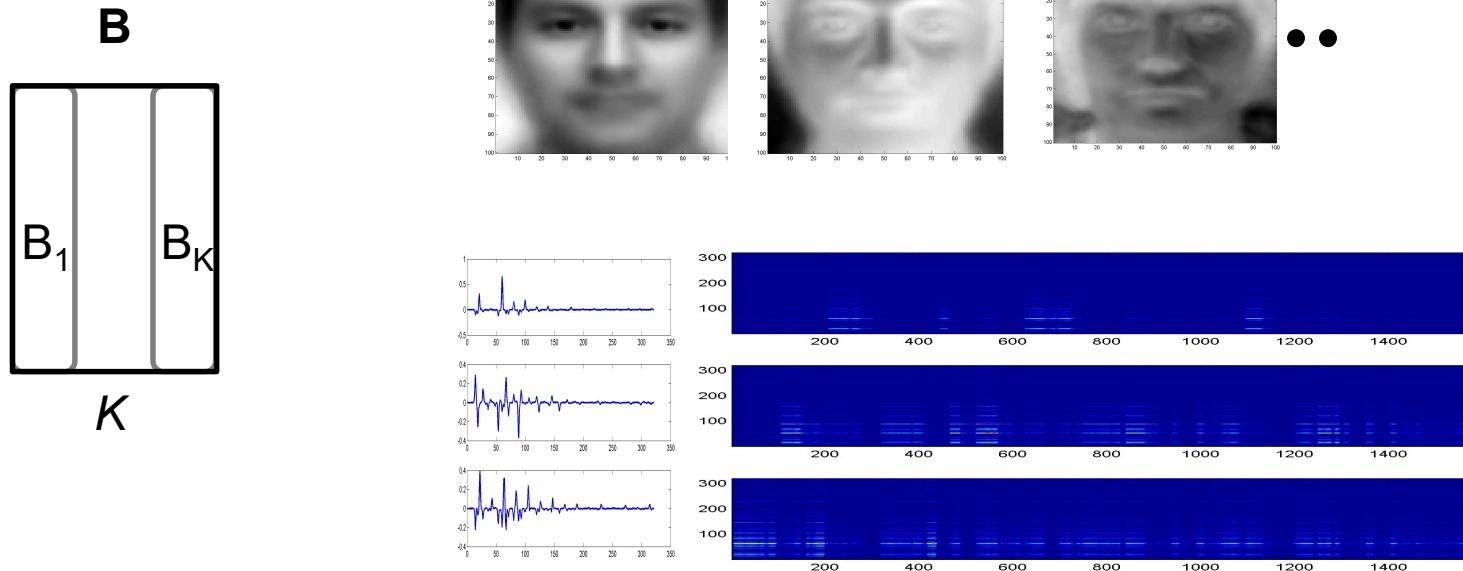
*With examples and
slides from
Paris Smaragdis*

A Quick Recap



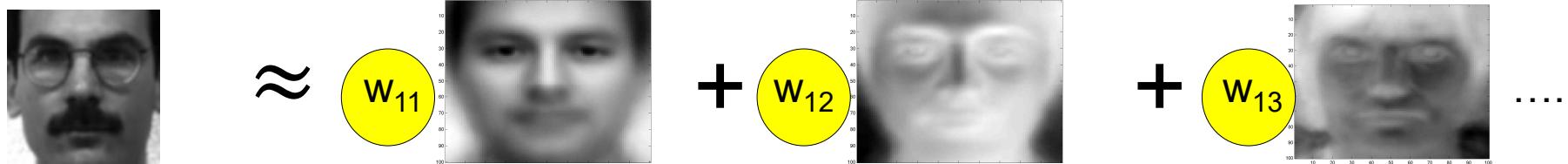
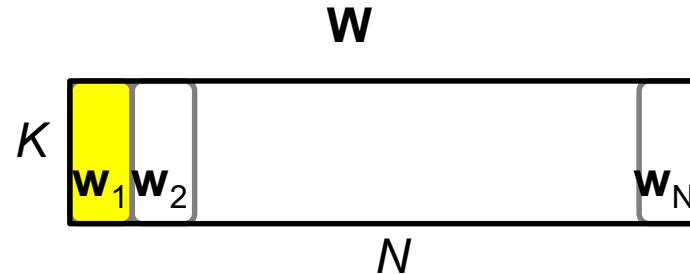
- **Problem:** Given a collection of data X , find a set of “bases” B , such that each vector x_i can be expressed as a weighted combination of the bases

A Quick Recap: Subproblem 1



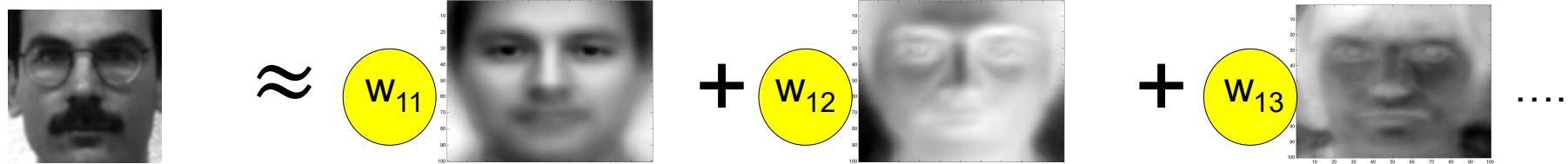
- **Problem 1: Finding bases**
 - Finding typical faces
 - Finding “notes” like structures

A Quick Recap: Subproblem 2



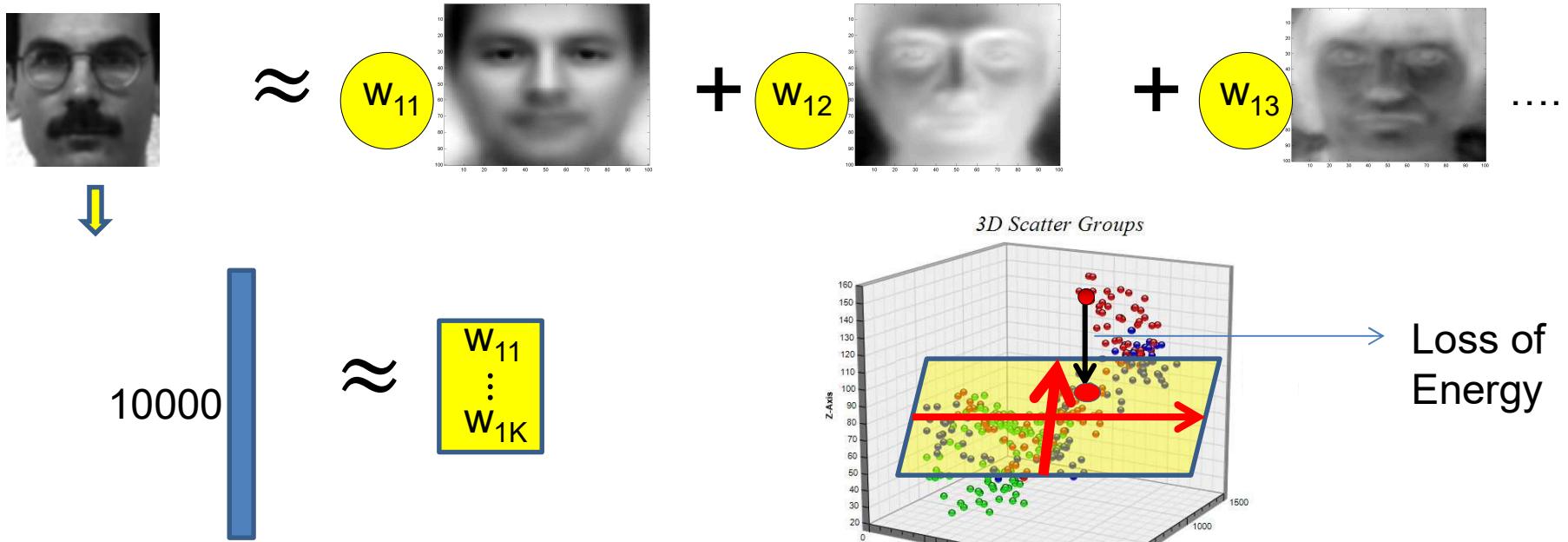
- **Problem 2:** Expressing instances in terms of these bases
 - Finding weights of typical faces
 - Finding weights of notes

A Quick Recap: WHY? 1.



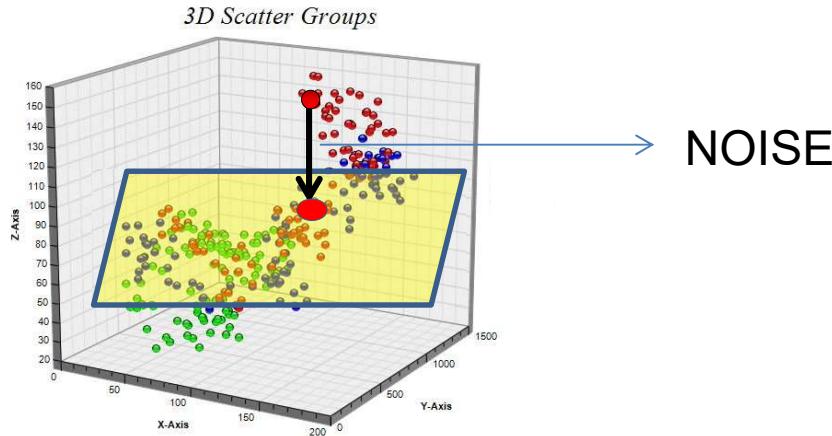
- ***Better Representation:*** The weights $\{w_{ij}\}$ represent the vectors in a *meaningful* way
 - Better suited to semantically motivated operation
 - Better suited for specific statistical models

A Quick Recap: WHY? 2.



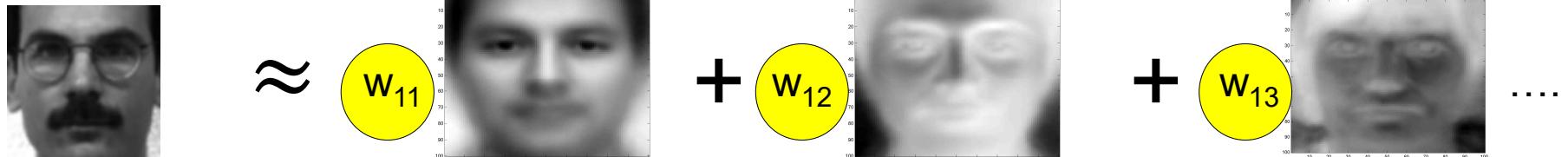
- **Dimensionality Reduction:** The number of Bases may be fewer than the dimensions of the vectors
 - Represent each Vector using fewer numbers
 - Expresses each vector within a *subspace*
 - Loses information / energy
 - **Objective:** Lose *least* energy

A Quick Recap: WHY? 3.

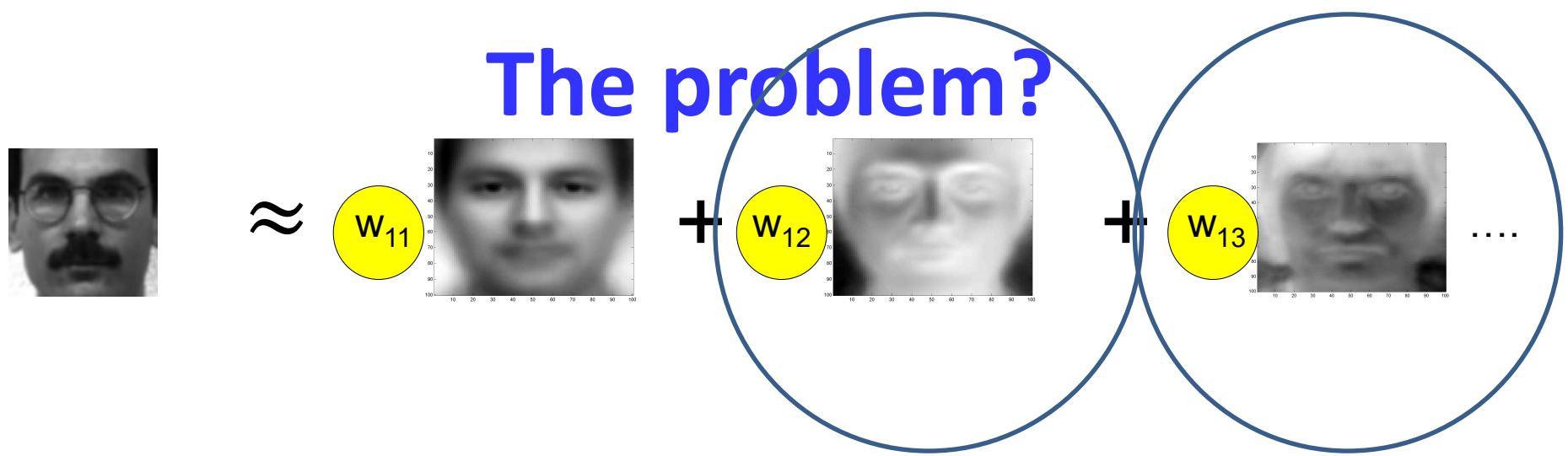


- **Denoising:** Reduced dimensional representation eliminates dimensions
- Can often eliminate *noise* dimensions
 - Signal-to-Noise ratio worst in dimensions where the signal has least energy/information
 - Removing them eliminates noise

A Quick Recap: HOW? KLT/PCA



- *Find Eigenvectors of Correlation matrix*
 - These are our “Eigen” bases
 - Capture information compactly and satisfy most of our requirements
- **MOST??**



- ***What is a negative face?***
 - And what does it mean to subtract one face from the other?
- Problem more obvious when applied to music
 - You would like bases to be notes
 - Weights to be scores
 - What is a negative note? What is a negative score?

Poll 1

Summary

- Orthogonality and energy maximization are statistically meaningful operations
- But may not be *physically* meaningful
- Next: A physically meaningful constraint
 - Non-negativity

The Engineer and the Musician

Once upon a time a rich potentate discovered a previously unknown recording of a beautiful piece of music. Unfortunately it was badly damaged.



He greatly wanted to find out what it would sound like if it were not.



So he hired an engineer and a musician to solve the problem..

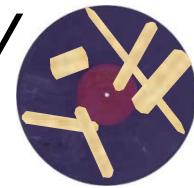
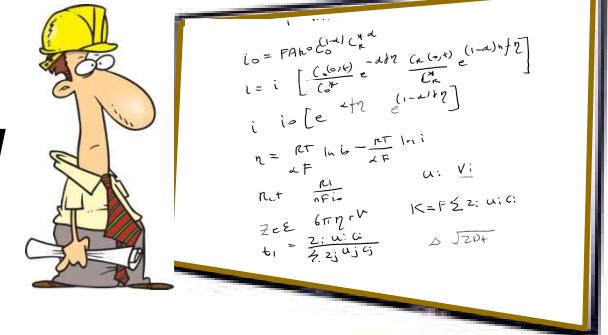


The Engineer and the Musician

The engineer worked for many years. He spent much money and published many papers.



Finally he had a somewhat scratchy restoration of the music..



The musician listened to the music carefully for a day, transcribed it, broke out his trusty keyboard and replicated the music.

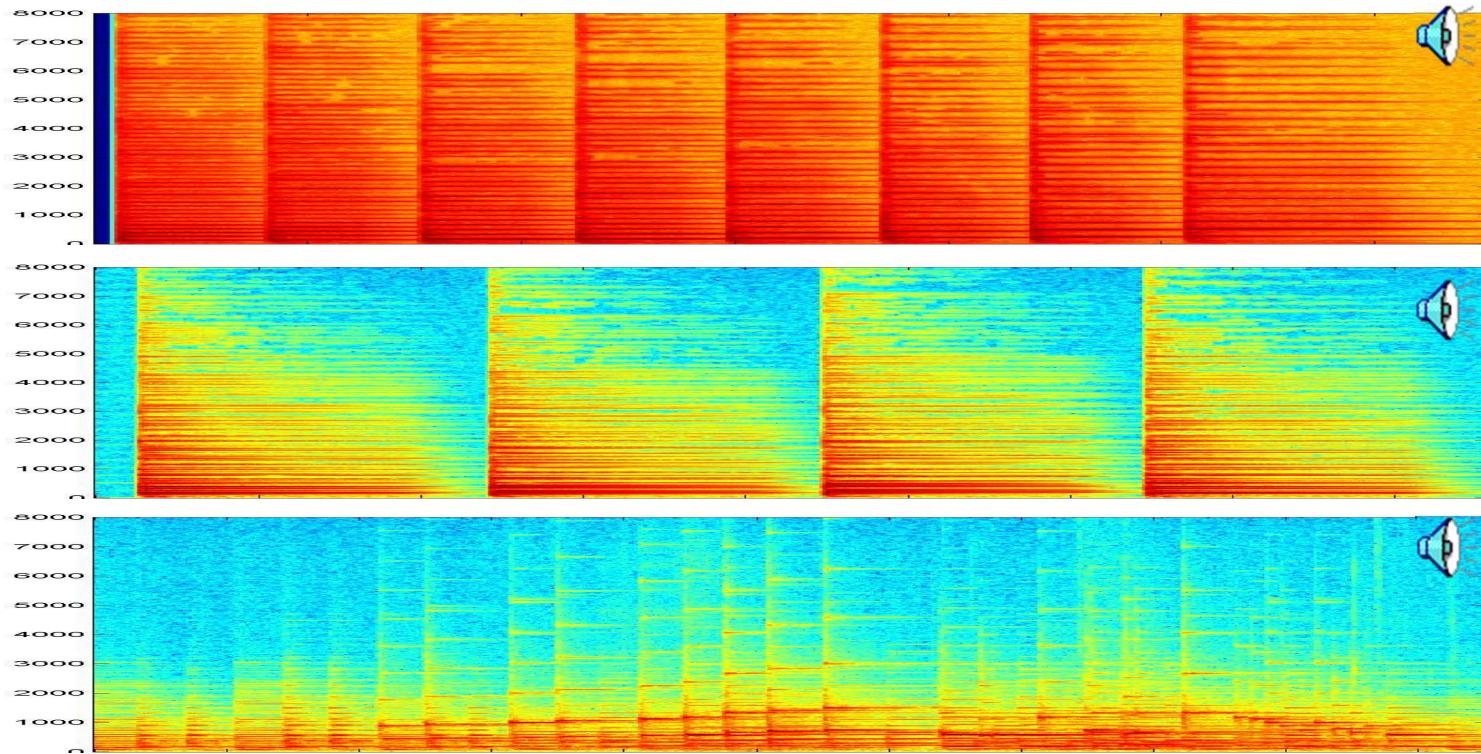


The Prize

Who do you think won the princess?



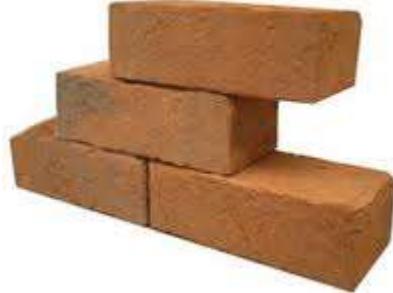
The search for building blocks



- What composes an audio signal?
 - E.g. notes compose music

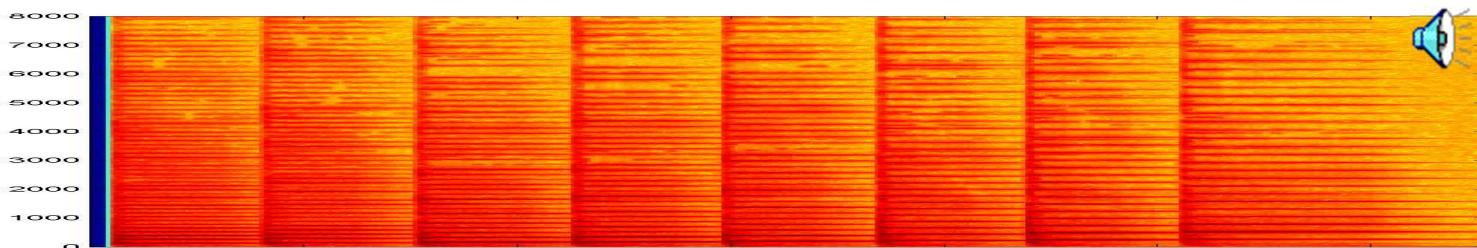
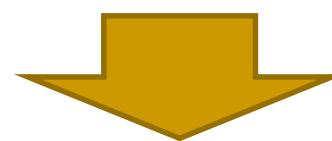
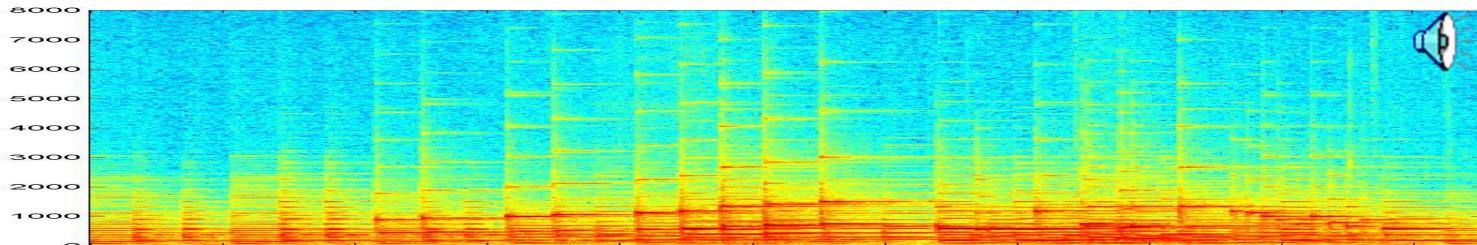
The properties of building blocks

- Constructive composition
 - A second note does not diminish a first note



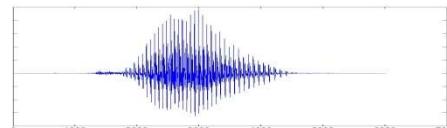
- Linearity of composition
 - Notes do not distort one another

Looking for building blocks in sound

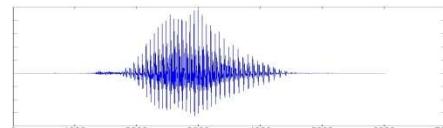


- Can we compute the building blocks from sound itself
 - *Can we learn the notes from the music?*

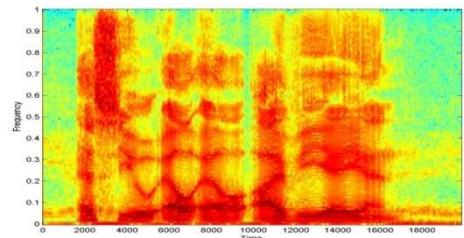
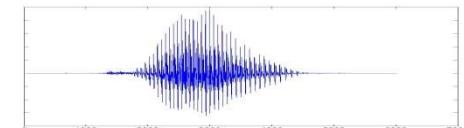
A property of power spectra



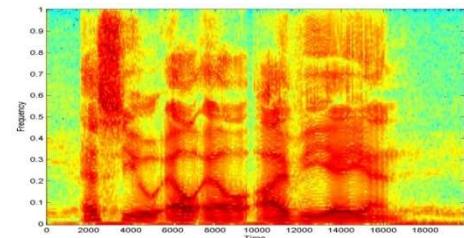
+



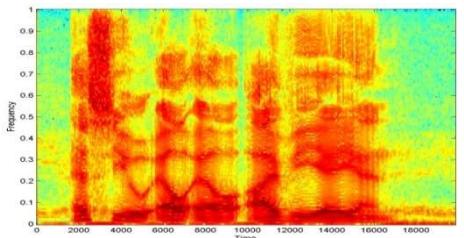
=



+

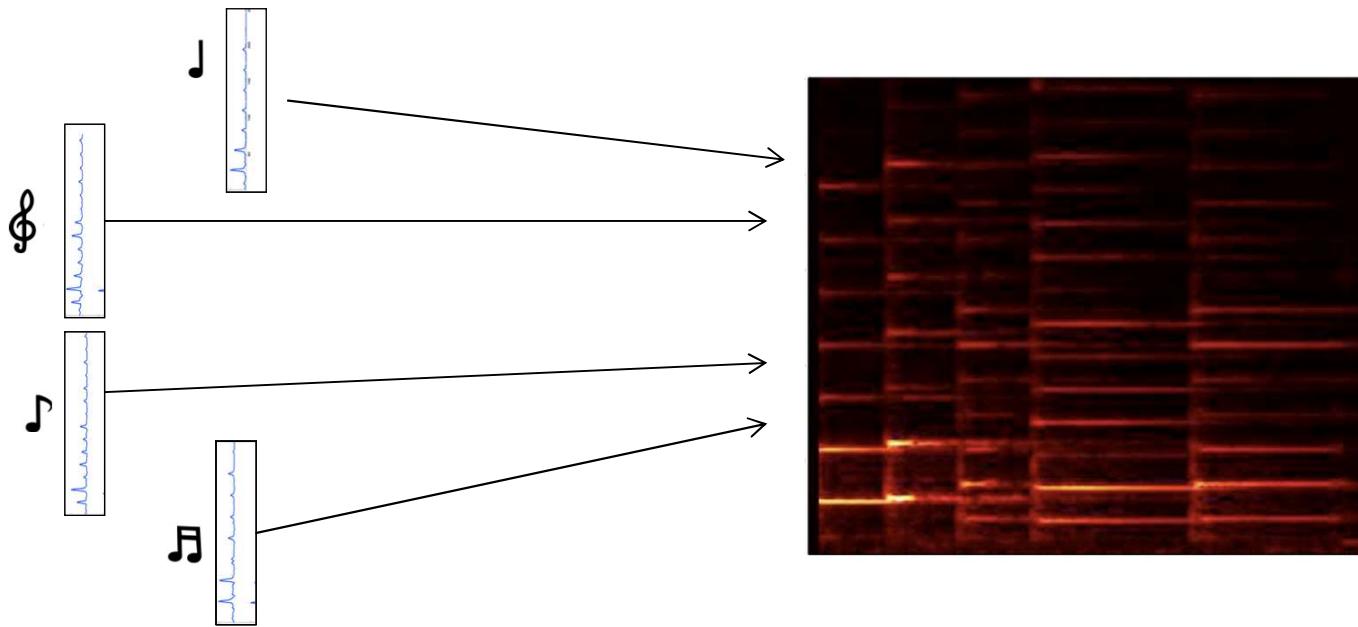


=



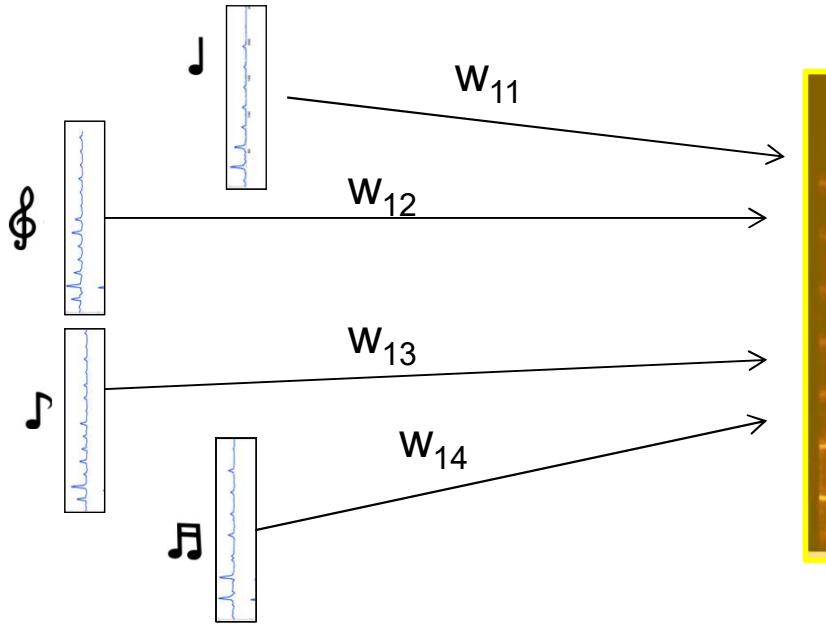
- When two or more independent signals are added, their power spectra (approximately) add
 - Their power spectrograms add as well

Building Blocks of Sound



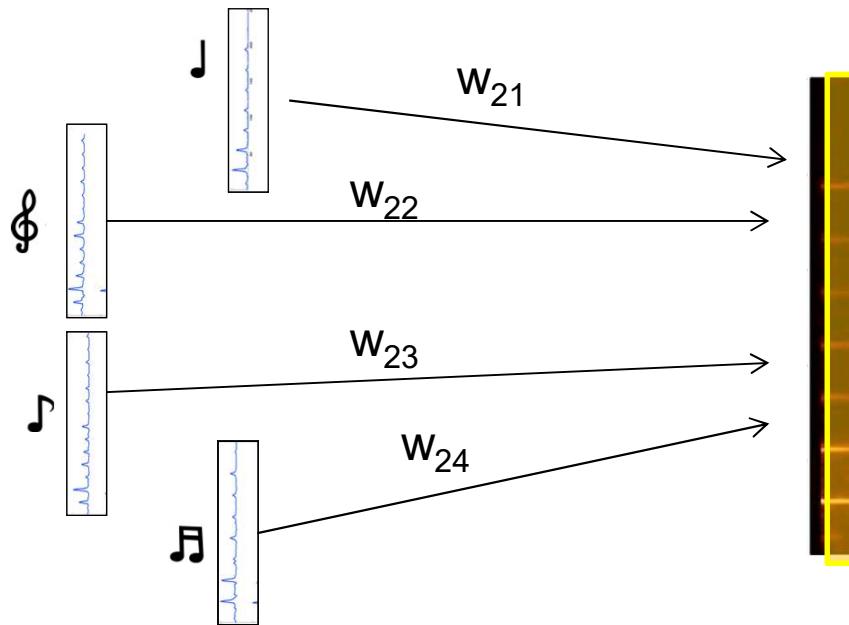
- The building blocks of sound are (power) spectral structures
 - E.g. notes build music
 - The spectra are entirely non-negative
- The complete sound is composed by *constructive* combination of the building blocks scaled to different non-negative gains
 - E.g. notes are played with varying energies through the music
 - The sound from the individual notes combines to form the final spectrogram
- The final spectrogram is also non-negative

Building Blocks of Sound



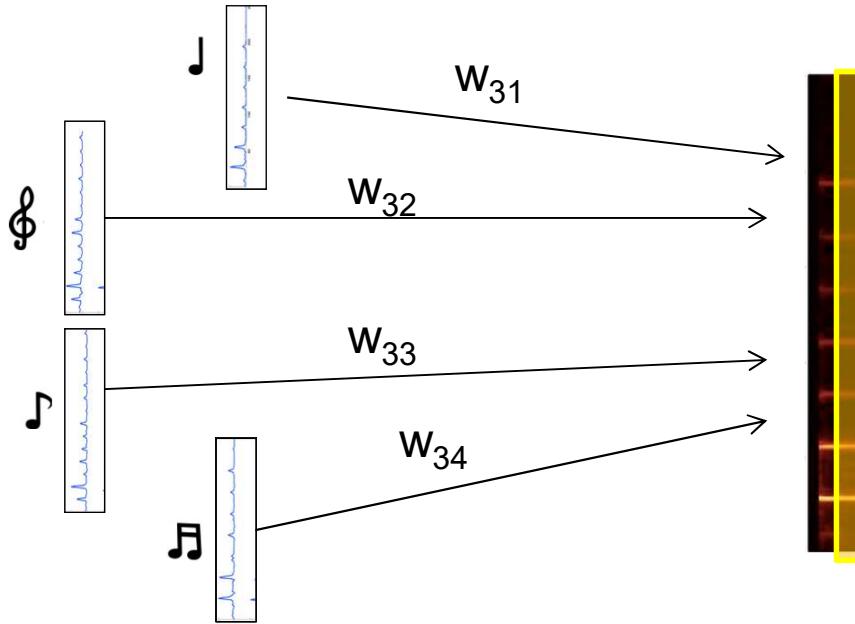
- Each frame of sound is composed by activating each spectral building block by a frame-specific amount
 - Individual frames are composed by activating the building blocks to different degrees
 - E.g. notes are strummed with different energies to compose the frame

Composing the Sound



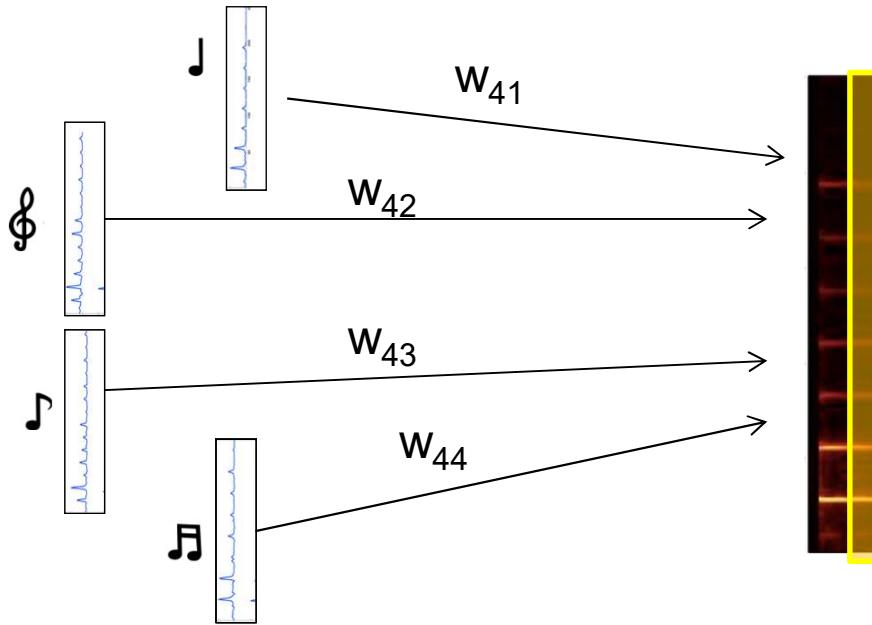
- Each frame of sound is composed by activating each spectral building block by a frame-specific amount
 - Individual frames are composed by activating the building blocks to different degrees
 - E.g. notes are strummed with different energies to compose the frame

Building Blocks of Sound



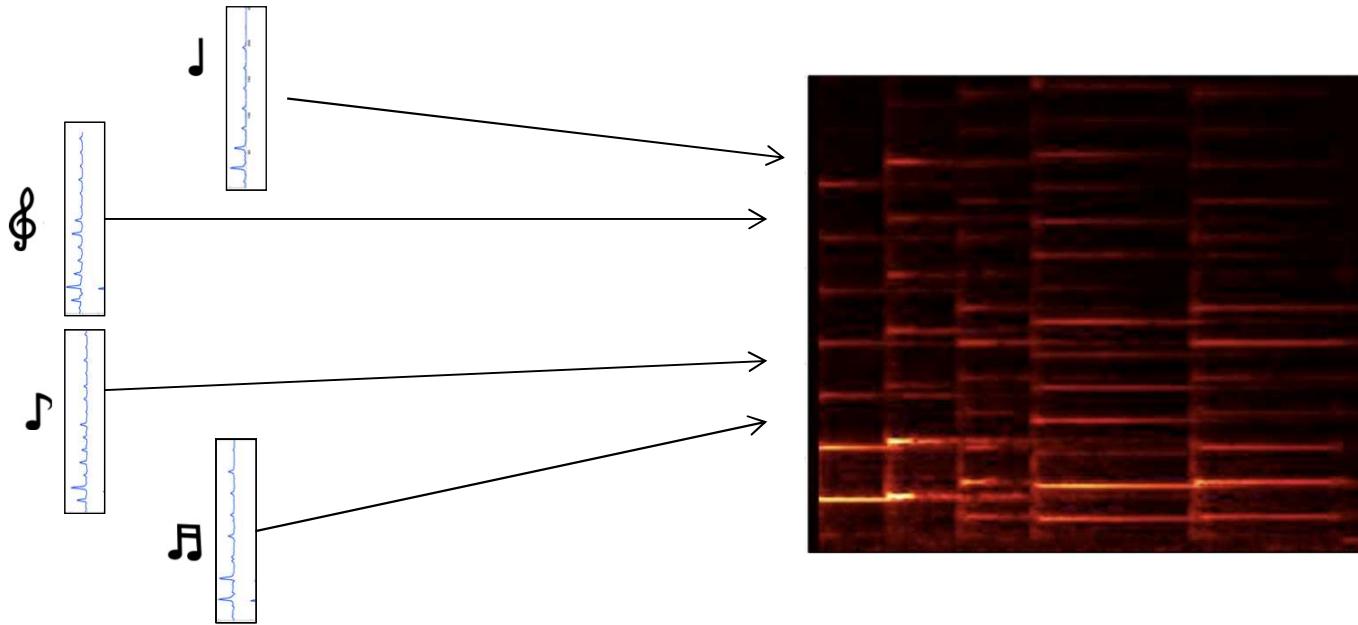
- Each frame of sound is composed by activating each spectral building block by a frame-specific amount
 - Individual frames are composed by activating the building blocks to different degrees
 - E.g. notes are strummed with different energies to compose the frame

Building Blocks of Sound



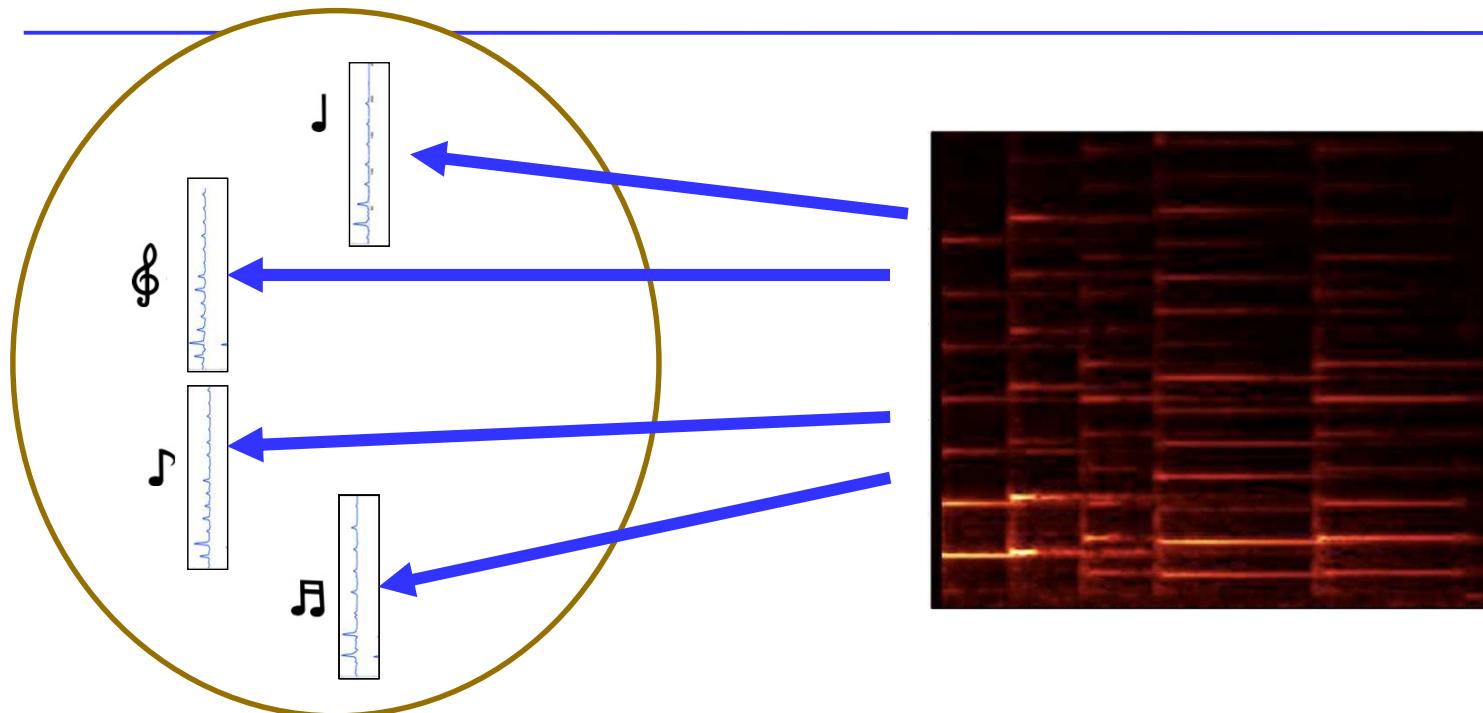
- Each frame of sound is composed by activating each spectral building block by a frame-specific amount
 - Individual frames are composed by activating the building blocks to different degrees
 - E.g. notes are strummed with different energies to compose the frame

Building Blocks of Sound



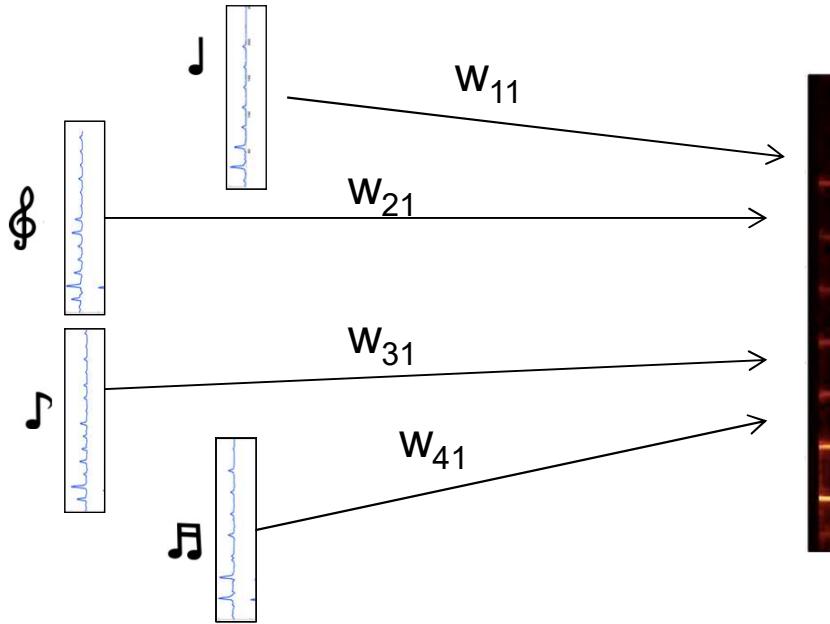
- Each frame of sound is composed by activating each spectral building block by a frame-specific amount
 - Individual frames are composed by activating the building blocks to different degrees
 - E.g. notes are strummed with different energies to compose the frame

The Problem of Learning



- Given only the final sound, determine its building blocks
 - From only listening to music, learn all about musical notes!

In Math



$$V_1 = w_{11}B_1 + w_{21}B_2 + w_{31}B_3 + \dots$$

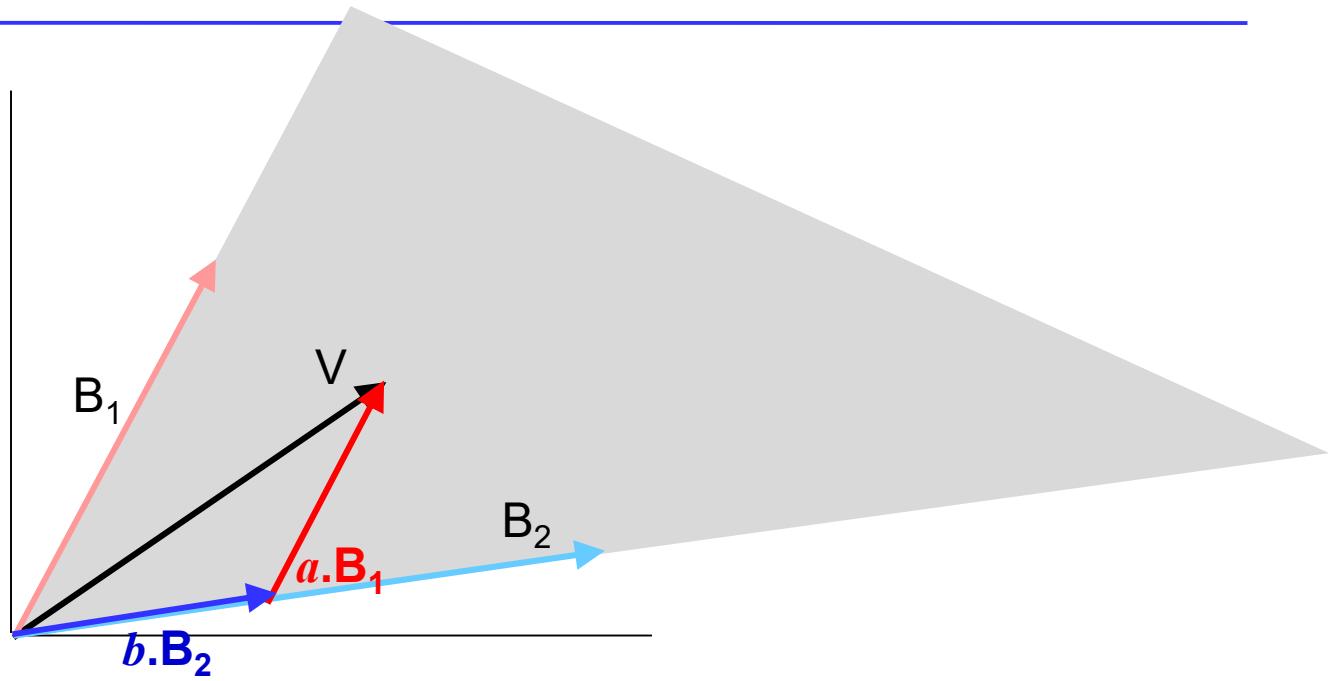
- Each frame is a non-negative power spectral vector
- Each note is a non-negative power spectral vector
- Each frame is a non-negative combination of the notes

Non-negative matrix factorization: Basics

- NMF is used in a *compositional* model
- Data are assumed to be non-negative
 - E.g. power spectra
- Every data vector is explained as a purely constructive linear composition of a set of bases
 - $V = \sum_i w_i B_i$
 - The bases B_i are in the same domain as the data
 - I.e. they are power spectra
- Constructive composition: no subtraction allowed
 - Weights w_i must all be non-negative
 - All components of bases B_i must also be non-negative

Understanding non-negative combination

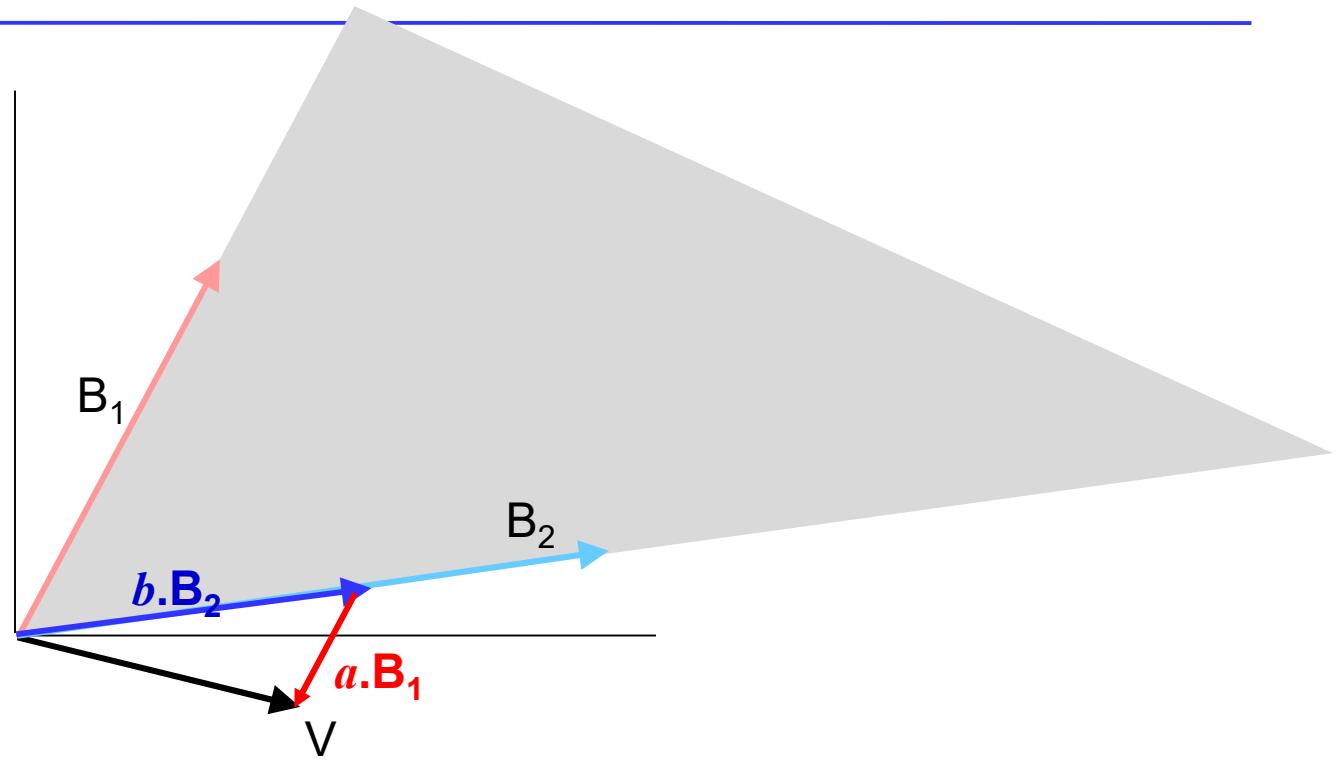
$$V = aB_1 + bB_2$$



- Non-negative combination: a and b are strictly non-negative
- Implies V must lie *inside the cone of B_1 and B_2*
 - V can be composed without reversing the directions of B_1 and B_2

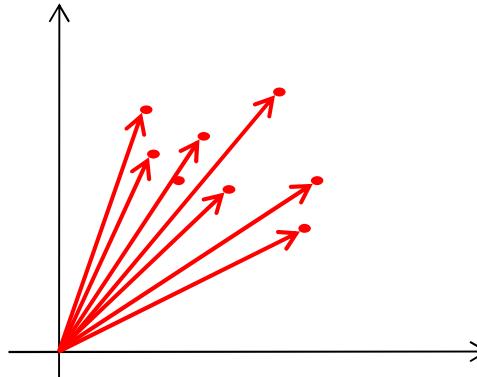
Understanding non-negative combination

$$V = aB_1 + bB_2$$

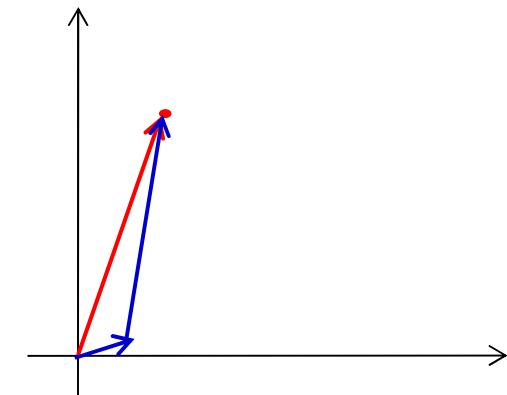
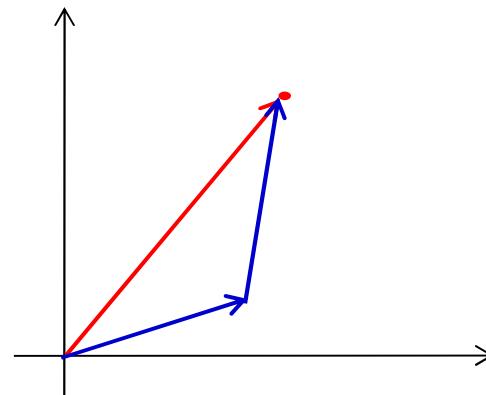
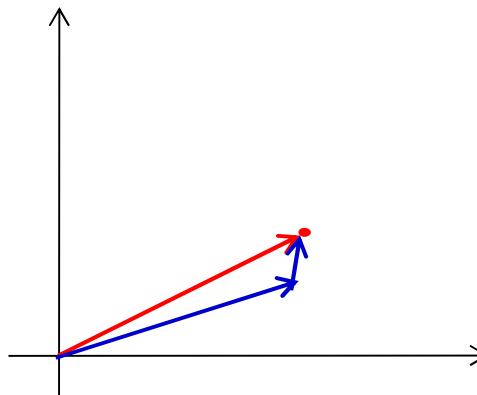
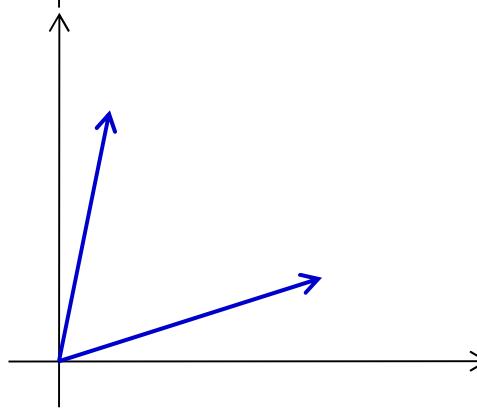


- If V lies outside the cone, at least one B_1 or B_2 must be reversed in direction to compose it
 - At least one of a and b must be negative

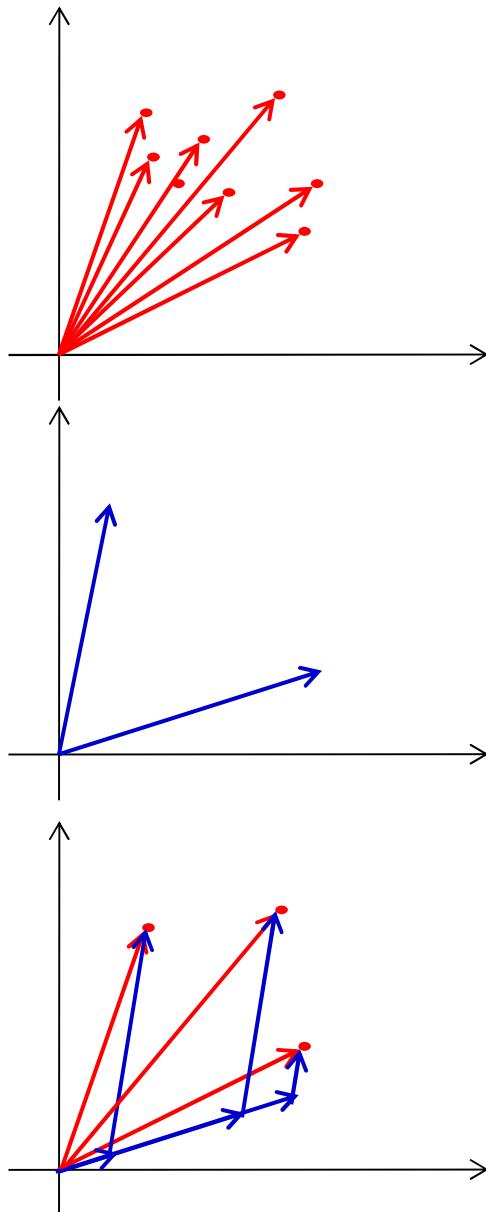
Learning building blocks: Restating the problem



- Given a collection of spectral vectors (from the composed sound) ...
- Find a set of “basic” sound spectral vectors such that ...
- All of the spectral vectors can be composed through constructive addition of the bases
 - We never have to flip the direction of any basis



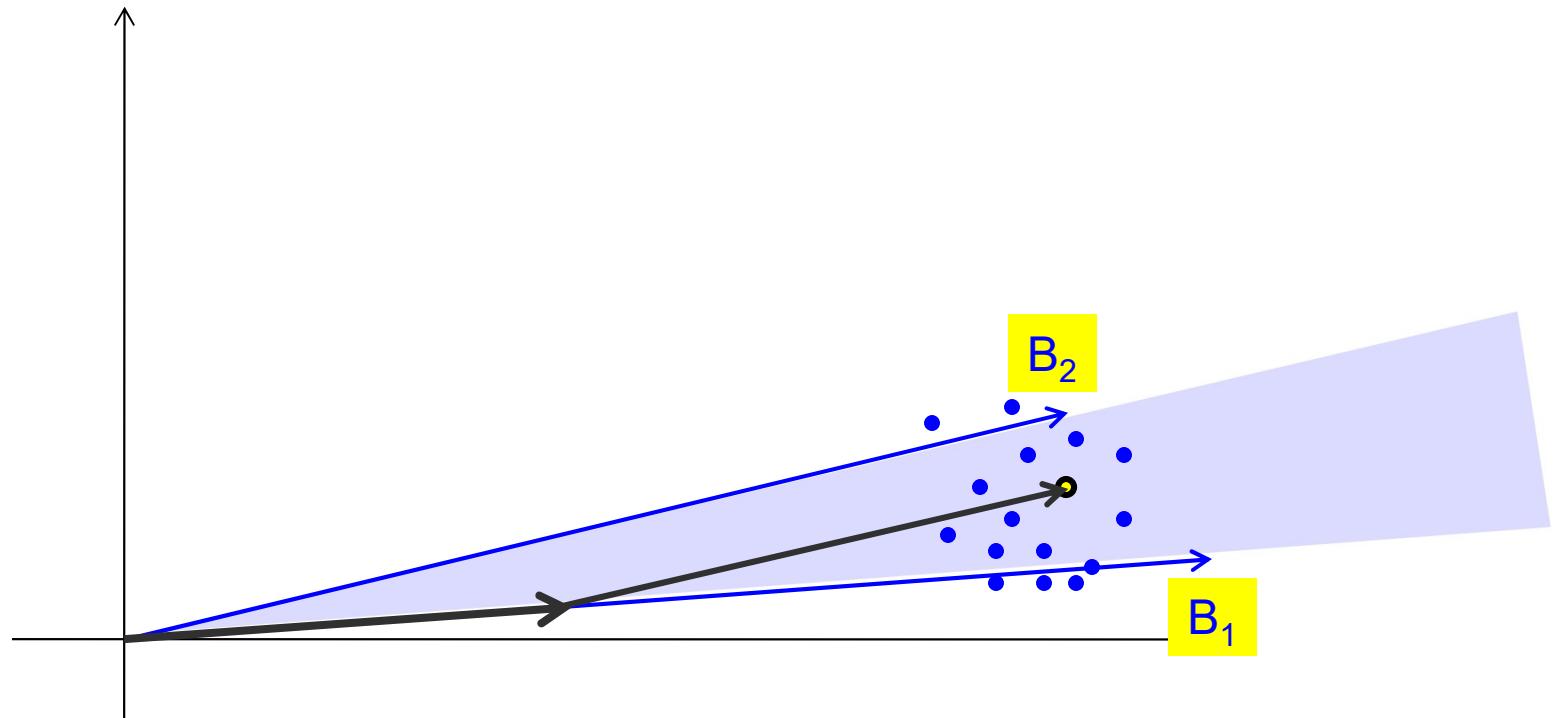
Learning building blocks: Restating the problem



$$\mathbf{V} = \mathbf{B}\mathbf{W}$$

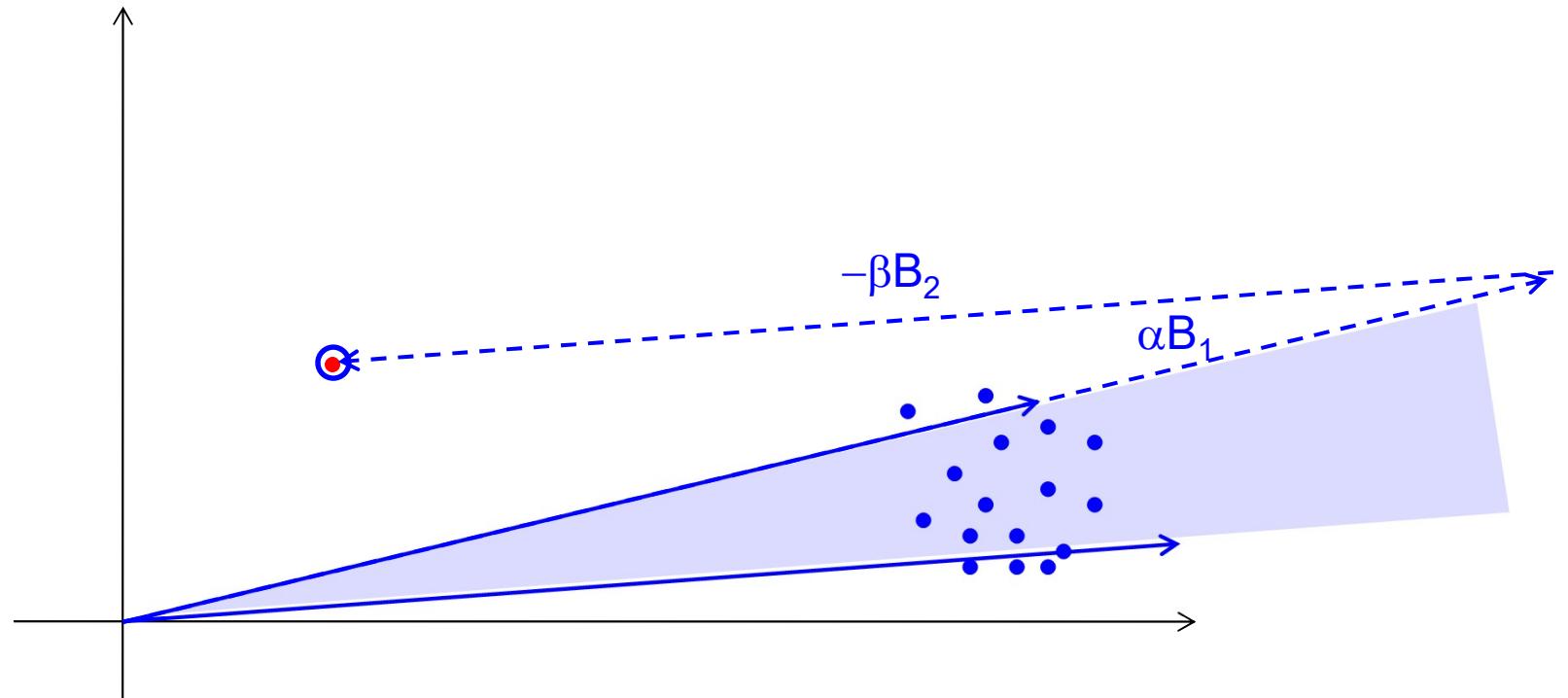
- Each column of \mathbf{V} is one “composed” spectral vector
- Each column of \mathbf{B} is one building block
 - One spectral basis
- Each column of \mathbf{W} has the scaling factors for the building blocks to compose the corresponding column of \mathbf{V}
- All columns of \mathbf{V} are non-negative
- All entries of \mathbf{B} and \mathbf{W} must also be non-negative

Interpreting non-negative factorization



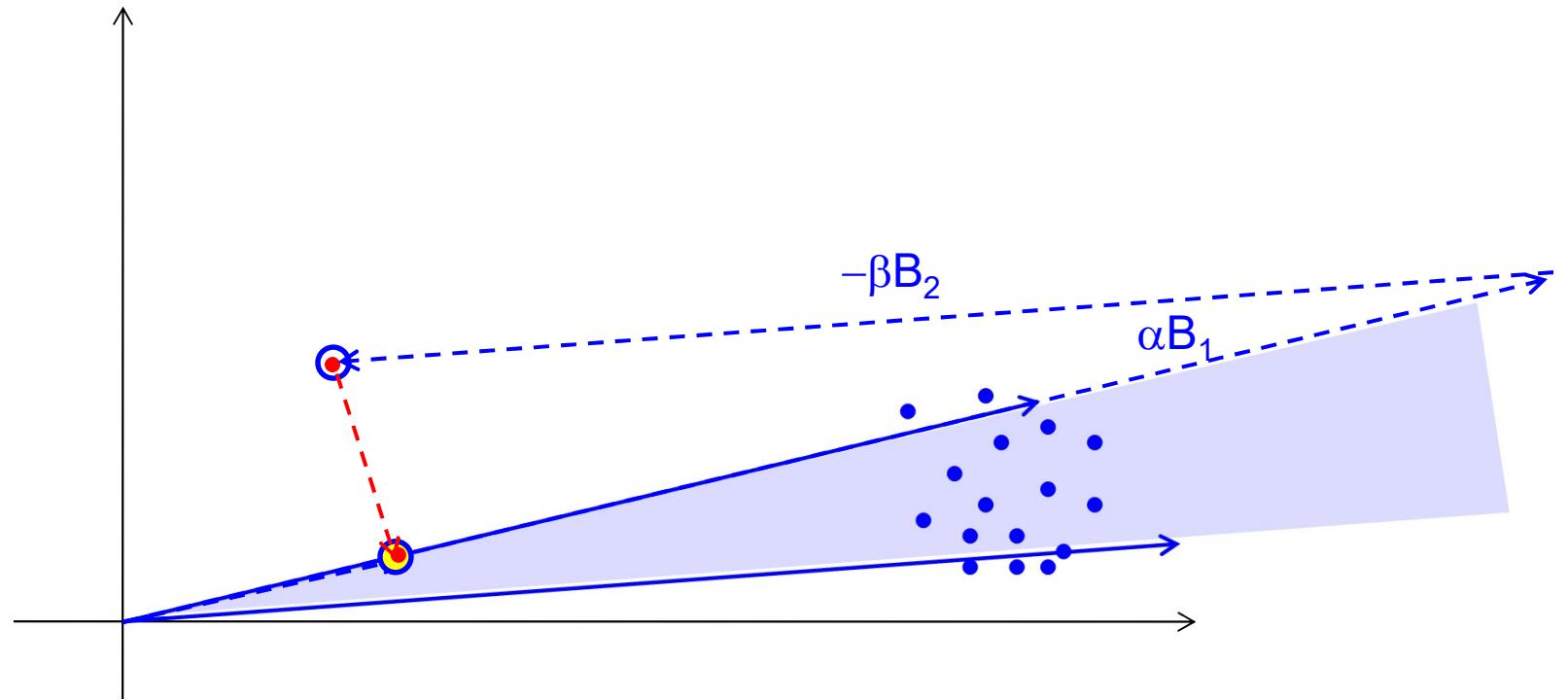
- Bases are non-negative, lie in the positive quadrant
- Blue lines represent bases, blue dots represent vectors
- Any vector that lies between the bases (highlighted region) can be expressed as a non-negative combination of bases
 - E.g. the black dot

Interpreting non-negative factorization



- Vectors outside the shaded enclosed area can only be expressed as a linear combination of the bases by reversing a basis
 - I.e. assigning a negative weight to the basis
 - E.g. the red dot
 - Alpha and beta are scaling factors for bases
 - Beta weighting is negative

Interpreting non-negative factorization



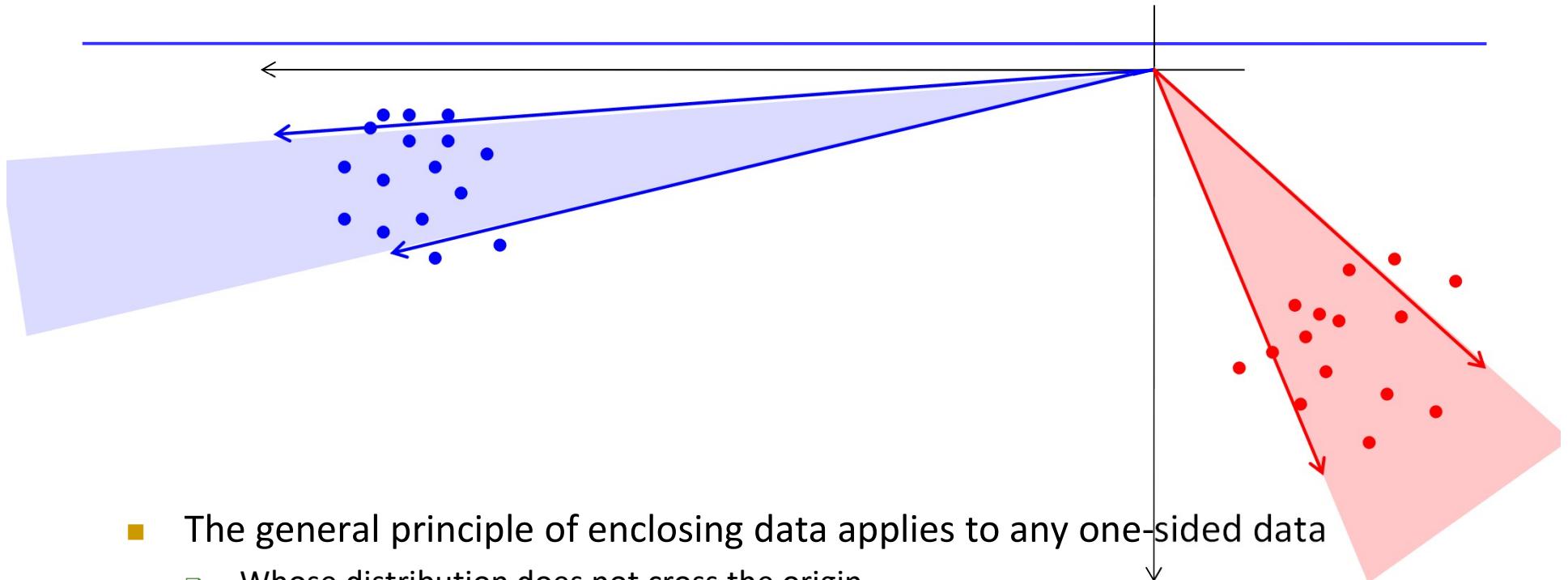
- If we approximate the red dot as a non-negative combination of the bases, the approximation will lie in the shaded region
 - On or close to the boundary
 - The approximation has error

The NMF representation

- The representation characterizes all data as lying within a compact convex region (a cone)
 - “Compact” → enclosing only a small fraction of the entire space
 - The more compact the enclosed region, the more it localizes the data within it
 - Represents the boundaries of the distribution of the data better
 - Conventional statistical models represent the mode of the distribution
- The *bases* must be chosen to
 - Enclose the data as compactly as possible
 - And also enclose as much of the data as possible
 - Data that are not enclosed are not represented correctly

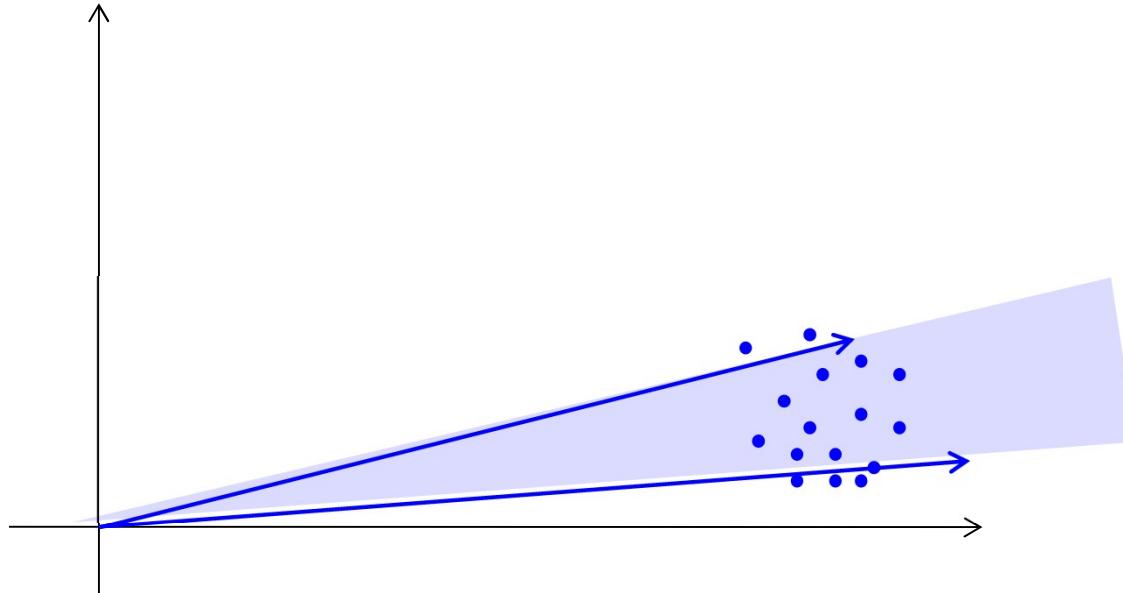
Poll 2

Data need not be non-negative



- The general principle of enclosing data applies to any one-sided data
 - Whose distribution does not cross the origin.
- The only part of the model that must be non-negative are the weights.
- Examples
 - Blue bases enclose blue region in negative quadrant
 - Red bases enclose red region in positive-negative quadrant
- Notions of compactness and enclosure still apply
 - This is a generalization of NMF
 - We won't discuss it further

NMF: Learning Bases

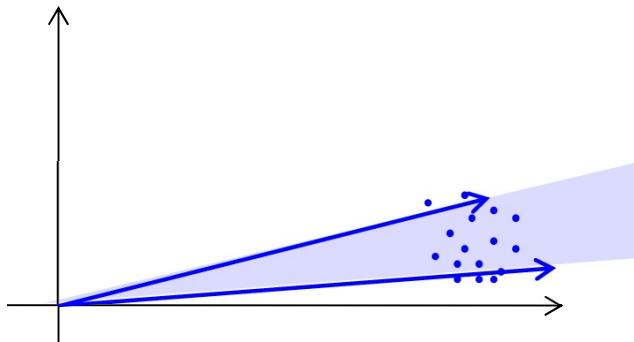


- Given a collection of data vectors (blue dots)
- Goal: find a set of bases (blue arrows) such that they enclose the data.
- Ideally, they must simultaneously enclose the smallest volume
 - *This “enclosure” constraint is usually not explicitly imposed in the standard NMF formulation*

NMF: Learning Bases

- Express every training vector as non-negative combination of bases
 - $V = \sum_i w_i B_i$
- In linear algebraic notation, represent:
 - Set of all training vectors as a data matrix **V**
 - A $D \times N$ matrix, D = dimensionality of vectors, N = No. of vectors
 - All basis vectors as a matrix **B**
 - A $D \times K$ matrix , K is the number of bases
 - The K weights for any vector V as a $K \times 1$ column vector **W**
 - The weight vectors for all N training data vectors as a matrix **W**
 - $K \times N$ matrix
- Ideally $V = BW$
 - All components of V , B and W are non-negative

NMF: Learning Bases



- $\mathbf{V} = \mathbf{BW}$ will only hold true if all training vectors in \mathbf{V} lie inside the region enclosed by the bases

- Learning bases is an iterative algorithm
- Intermediate estimates of \mathbf{B} do not satisfy $\mathbf{V} = \mathbf{BW}$
- Algorithm updates \mathbf{B} until $\mathbf{V} = \mathbf{BW}$ is satisfied as closely as possible

NMF: Minimizing Divergence

- Define a *Divergence* between data \mathbf{V} and approximation \mathbf{BW}
 - Divergence(\mathbf{V} , \mathbf{BW}) is the total error in approximating all vectors in \mathbf{V} as \mathbf{BW}
 - Must estimate *non-negative* \mathbf{B} and \mathbf{W} so that this error is minimized
- Divergence(\mathbf{V} , \mathbf{BW}) can be defined in different ways
 - L2: Divergence = $\sum_i \sum_j (V_{ij} - (BW)_{ij})^2$
 - Minimizing the L2 divergence gives us an algorithm to learn \mathbf{B} and \mathbf{W}
 - KL: Divergence(\mathbf{V}, \mathbf{BW}) = $\sum_i \sum_j V_{ij} \log(V_{ij} / (BW)_{ij}) + \sum_i \sum_j V_{ij} - \sum_i \sum_j (BW)_{ij}$
 - This is a *generalized* KL divergence that is minimum when $\mathbf{V} = \mathbf{BW}$
 - Minimizing the KL divergence gives us another algorithm to learn \mathbf{B} and \mathbf{W}
- Other divergence forms can also be used

NMF: Minimizing Divergence

- Define a *Divergence* between data \mathbf{V} and approximation \mathbf{BW}
 - Divergence(\mathbf{V} , \mathbf{BW}) is the total error in approximating all vectors in \mathbf{V} as \mathbf{BW}
 - Must estimate *non-negative* \mathbf{B} and \mathbf{W} so that this error is minimized
- Divergence(\mathbf{V} , \mathbf{BW}) can be defined in different ways
 - L2: Divergence = $\sum_i \sum_j (V_{ij} - (BW)_{ij})^2$
 - Minimizing the L2 divergence gives us an algorithm to learn \mathbf{B} and \mathbf{W}
 - KL: Divergence(\mathbf{V}, \mathbf{BW}) = $\sum_i \sum_j V_{ij} \log(V_{ij} / (BW)_{ij}) + \sum_i \sum_j V_{ij} - \sum_i \sum_j (BW)_{ij}$
 - This is a *generalized* KL divergence that is minimum when $\mathbf{V} = \mathbf{BW}$
 - Minimizing the KL divergence gives us another algorithm to learn \mathbf{B} and \mathbf{W}
- Other divergence forms can also be used

NMF: Minimizing L_2 Divergence

- Divergence(\mathbf{V} , \mathbf{BW}) is defined as
 - $E = \| \mathbf{V} - \mathbf{BW} \|_F^2$
 - $E = \sum_i \sum_j (V_{ij} - (BW)_{ij})^2$
- Iterative solution: Minimize E such that \mathbf{B} and \mathbf{W} are strictly non-negative

NMF: Minimizing L_2 Divergence

- Learning both \mathbf{B} and \mathbf{W} with non-negativity
- Divergence(\mathbf{V} , \mathbf{BW}) is defined as
 - $E = \| \|\mathbf{V} - \mathbf{BW} \| \|_F^2$
 - $$\mathbf{V} \approx \mathbf{BW}$$
- Iterative solution:
 - $\mathbf{B} = [\mathbf{V} \text{ Pinv}(\mathbf{W})]_+$
 - $\mathbf{W} = [\text{Pinv}(\mathbf{B}) \mathbf{V}]_+$
 - Subscript + indicates thresholding –ve values to 0

NMF: Minimizing Divergence

- Define a *Divergence* between data \mathbf{V} and approximation \mathbf{BW}
 - Divergence(\mathbf{V} , \mathbf{BW}) is the total error in approximating all vectors in \mathbf{V} as \mathbf{BW}
 - Must estimate \mathbf{B} and \mathbf{W} so that this error is minimized
- Divergence(\mathbf{V} , \mathbf{BW}) can be defined in different ways
 - L2: Divergence = $\sum_i \sum_j (V_{ij} - (BW)_{ij})^2$
 - Minimizing the L2 divergence gives us an algorithm to learn \mathbf{B} and \mathbf{W}
 - KL: Divergence(\mathbf{V}, \mathbf{BW}) = $\sum_i \sum_j V_{ij} \log(V_{ij} / (BW)_{ij}) + \sum_i \sum_j V_{ij} - \sum_i \sum_j (BW)_{ij}$
 - This is a *generalized* KL divergence that is minimum when $\mathbf{V} = \mathbf{BW}$
 - Minimizing the KL divergence gives us another algorithm to learn \mathbf{B} and \mathbf{W}
- For many kinds of signals, e.g. sound, NMF-based representations work best when we minimize the KL divergence

NMF: Minimizing KL Divergence

- Divergence(\mathbf{V} , \mathbf{BW}) defined as
 - $E = \sum_i \sum_j V_{ij} \log(V_{ij} / (BW)_{ij}) + \sum_i \sum_j V_{ij} - \sum_i \sum_j (BW)_{ij}$
- Iterative update rules
- Number of iterative update rules have been proposed
- The most popular one is the multiplicative update rule..

NMF Estimation: Learning bases

- The algorithm to estimate \mathbf{B} and \mathbf{W} to minimize the KL divergence between \mathbf{V} and \mathbf{BW} :
- Initialize \mathbf{B} and \mathbf{W} (randomly)
- Iteratively update \mathbf{B} and \mathbf{W} using the following formulae

$$\mathbf{B} = \mathbf{B} \otimes \frac{\left(\begin{array}{c} \mathbf{V} \\ \mathbf{BW} \end{array} \right) \mathbf{W}^T}{\mathbf{1} \mathbf{W}^T}$$

$$\mathbf{W} = \mathbf{W} \otimes \frac{\mathbf{B}^T \left(\begin{array}{c} \mathbf{V} \\ \mathbf{BW} \end{array} \right)}{\mathbf{B}^T \mathbf{1}}$$

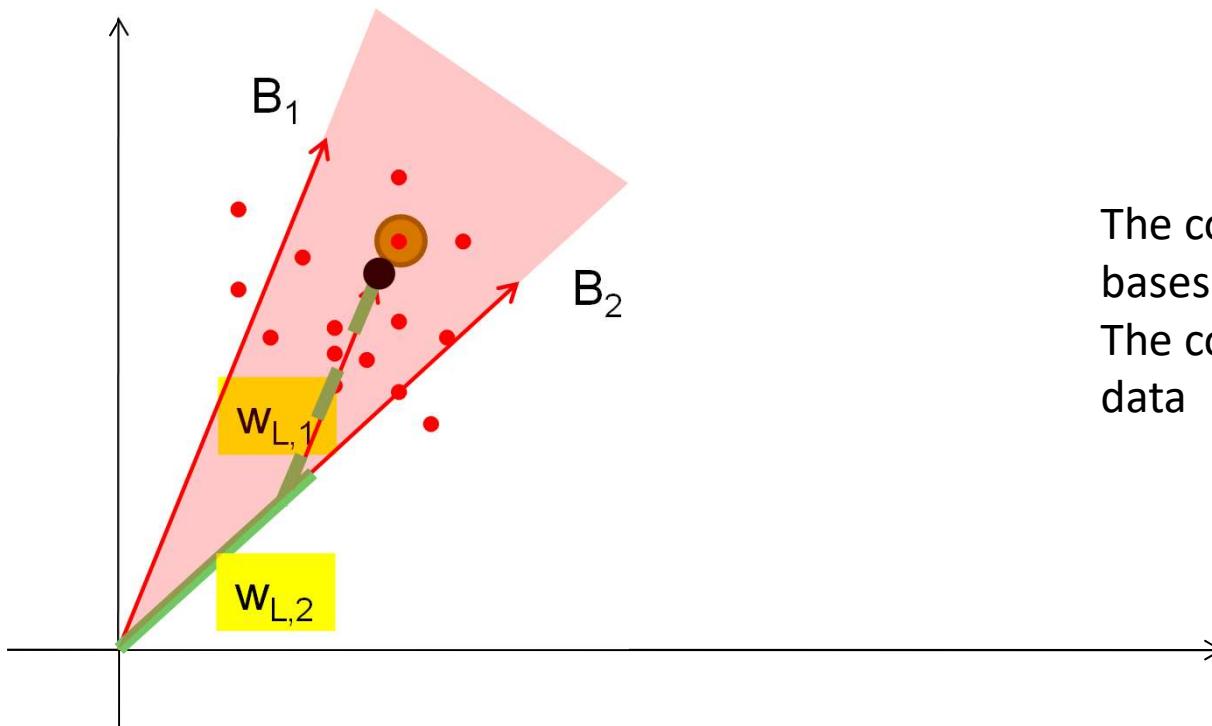
- Iterations continue until divergence converges
 - In practice, continue for a fixed no. of iterations

Reiterating

$$V_{D \times N} \approx B_{D \times K} W_{K \times N}$$

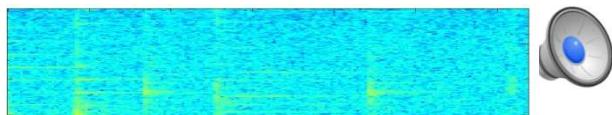
$$V_L \approx \sum_k w_{L,k} B_k$$

- NMF learns the *optimal set of basis vectors* B_k to approximate the data in terms of the bases
- It also learns how to compose the data in terms of these bases
 - Compositions can be inexact

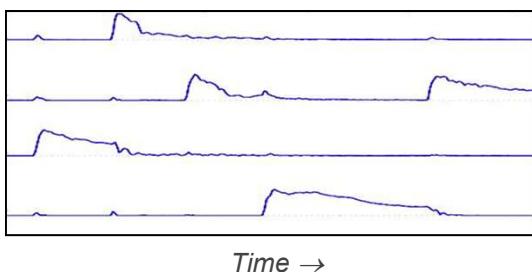
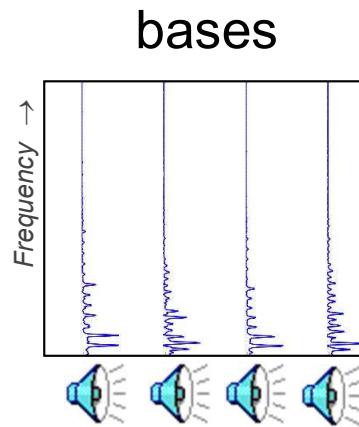


Learning building blocks of sound

From Bach's Fugue in Gm



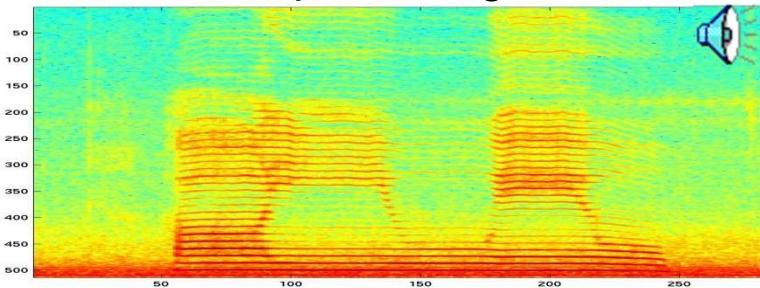
$$\mathbf{V} = \mathbf{BW}$$



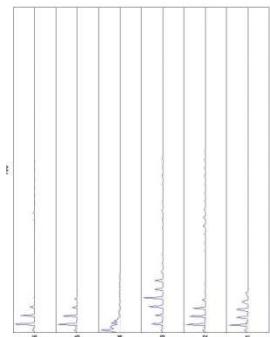
- Each column of \mathbf{V} is one spectral vector
- Each column of \mathbf{B} is one building block/basis
- Each column of \mathbf{W} has the scaling factors for the bases to compose the corresponding column of \mathbf{V}
- All terms are non-negative
- Learn \mathbf{B} (and \mathbf{W}) by applying NMF to \mathbf{V}

Learning Building Blocks

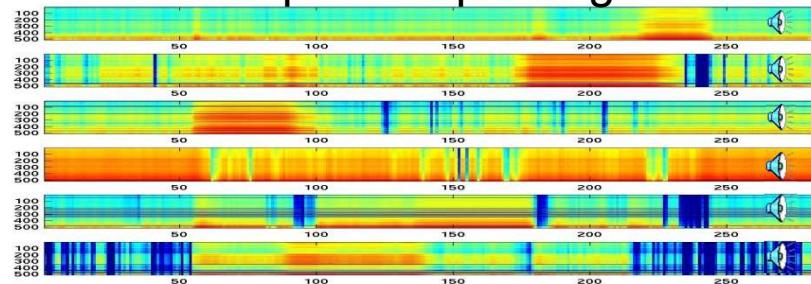
Speech Signal



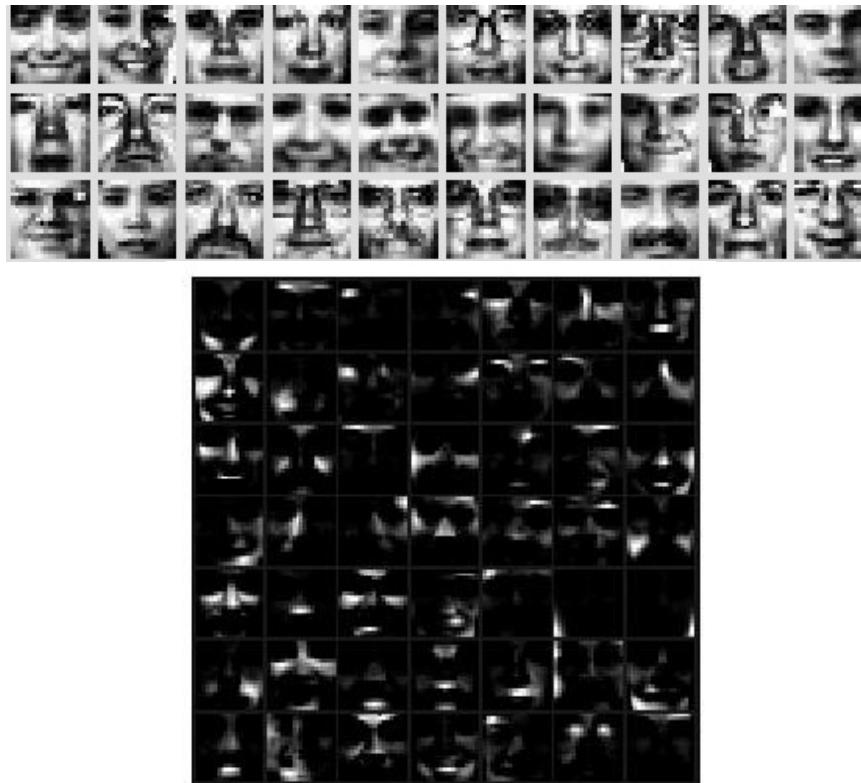
bases



Basis-specific spectrograms



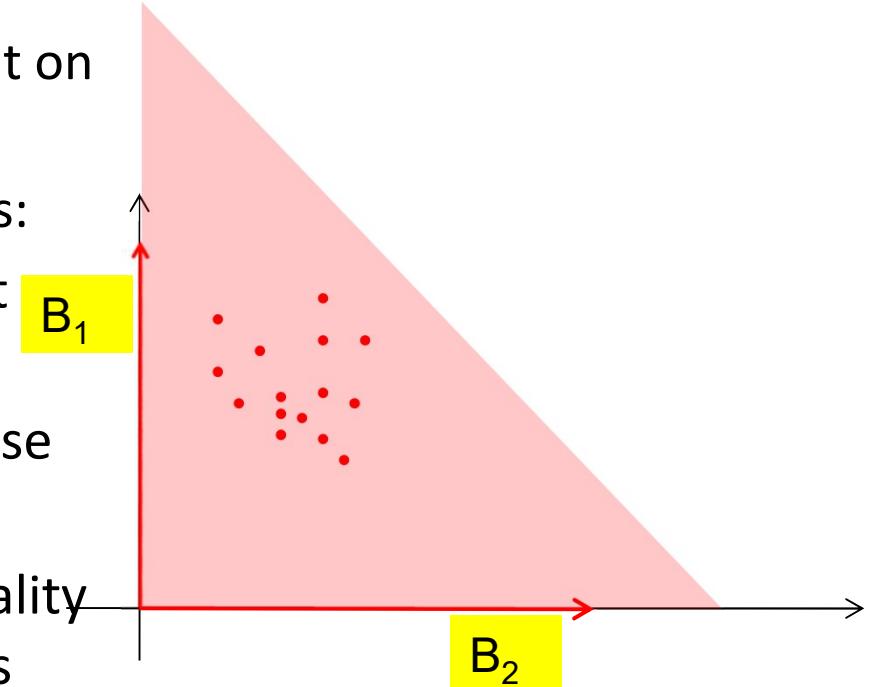
What about other data



- Faces
 - Trained 49 multinomial components on 2500 faces
 - Each face unwrapped into a 361-dimensional vector
 - Discovers parts of faces

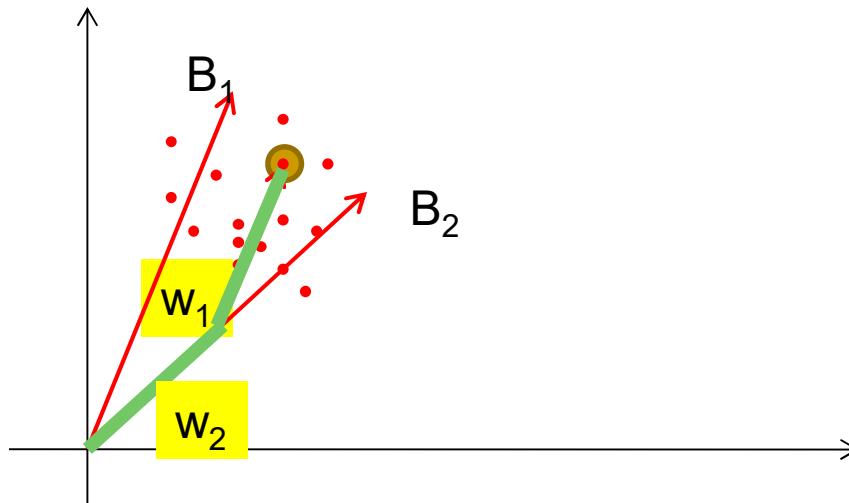
There is no “compactness” constraint

- No explicit “compactness” constraint on bases
- The red lines would be perfect bases:
 - Enclose all training data without error
 - Algorithm can end up with these bases
 - If no. of bases $K \geq$ dimensionality D , can get uninformative bases



- If $K < D$, we usually learn compact representations
 - NMF becomes a dimensionality reducing representation
 - Representing D -dimensional data in terms of K weights, where $K < D$

Representing Data using *Known* Bases



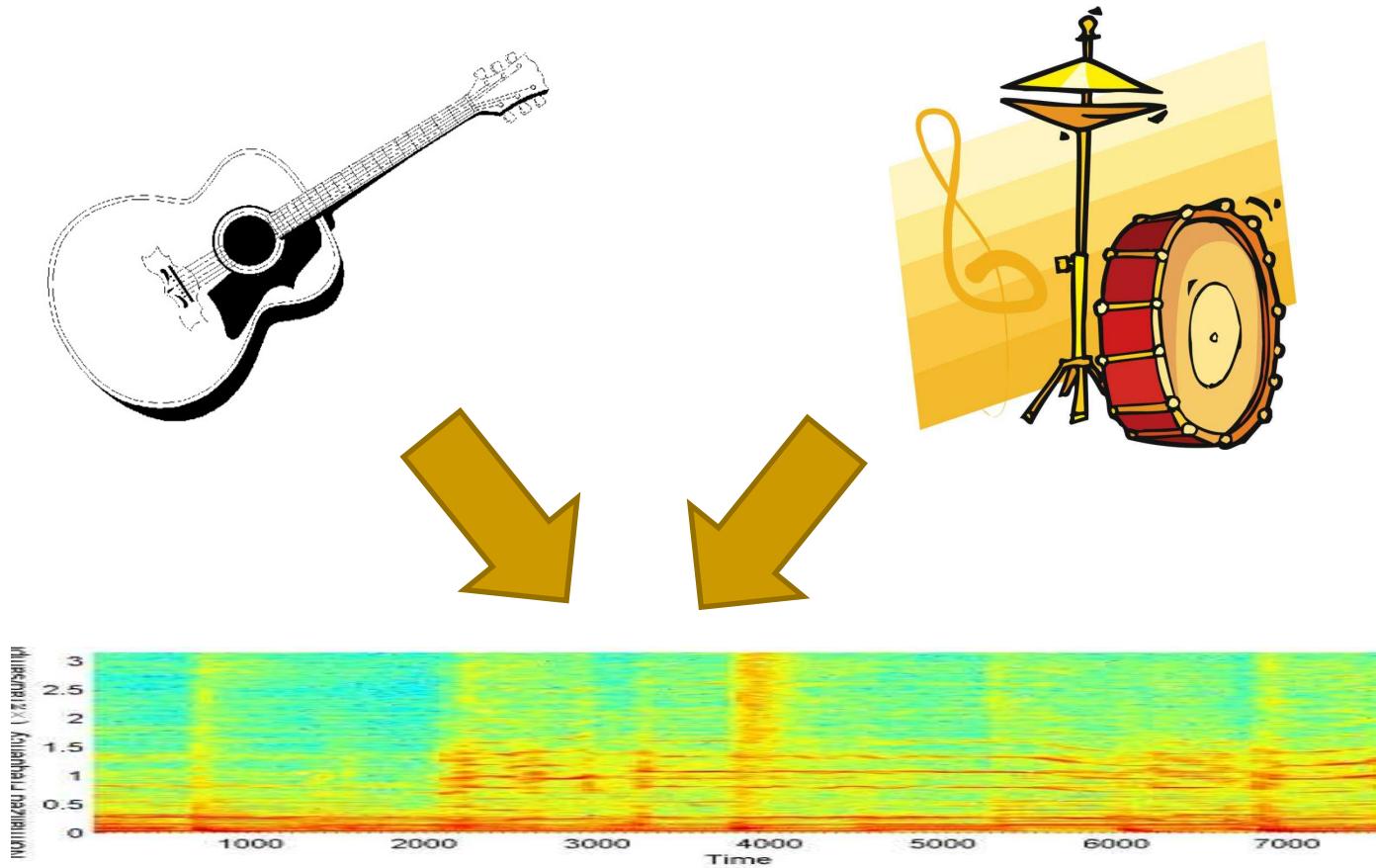
- If we already have bases B_k and are given a vector that must be expressed in terms of the bases: $V \approx \sum_k w_k B_k$
- Estimate weights as:
 - Initialize weights
 - Iteratively update them using

$$W = W \otimes \frac{B^T \left(\begin{matrix} V \\ BW \end{matrix} \right)}{B^T 1}$$

What can we do knowing the building blocks

- *Signal Representation*
- *Signal Separation*
- *Signal Completion*
- Denoising
- Signal recovery
- Music Transcription
- Etc.

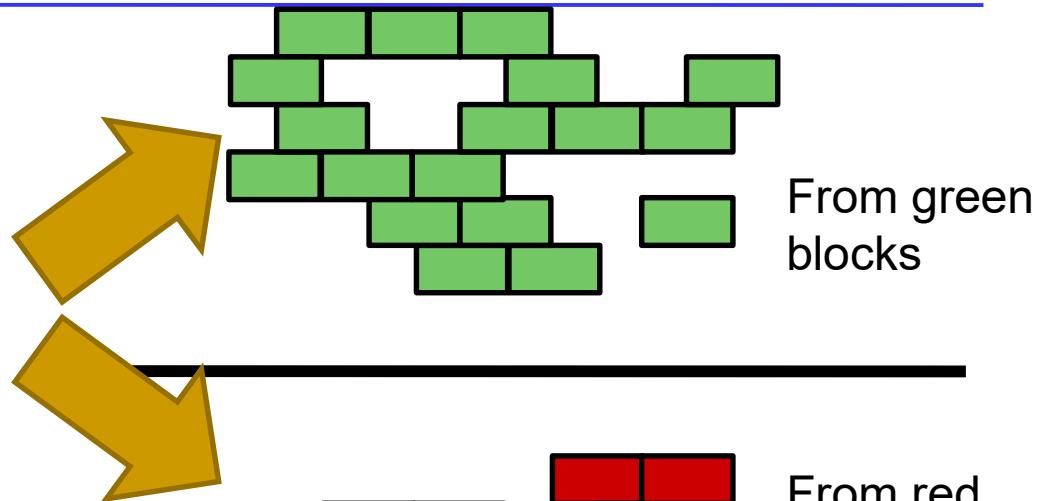
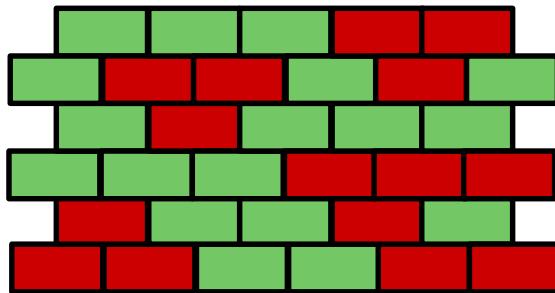
Signal Separation



- Can we separate mixed signals?

Undoing a Jigsaw Puzzle

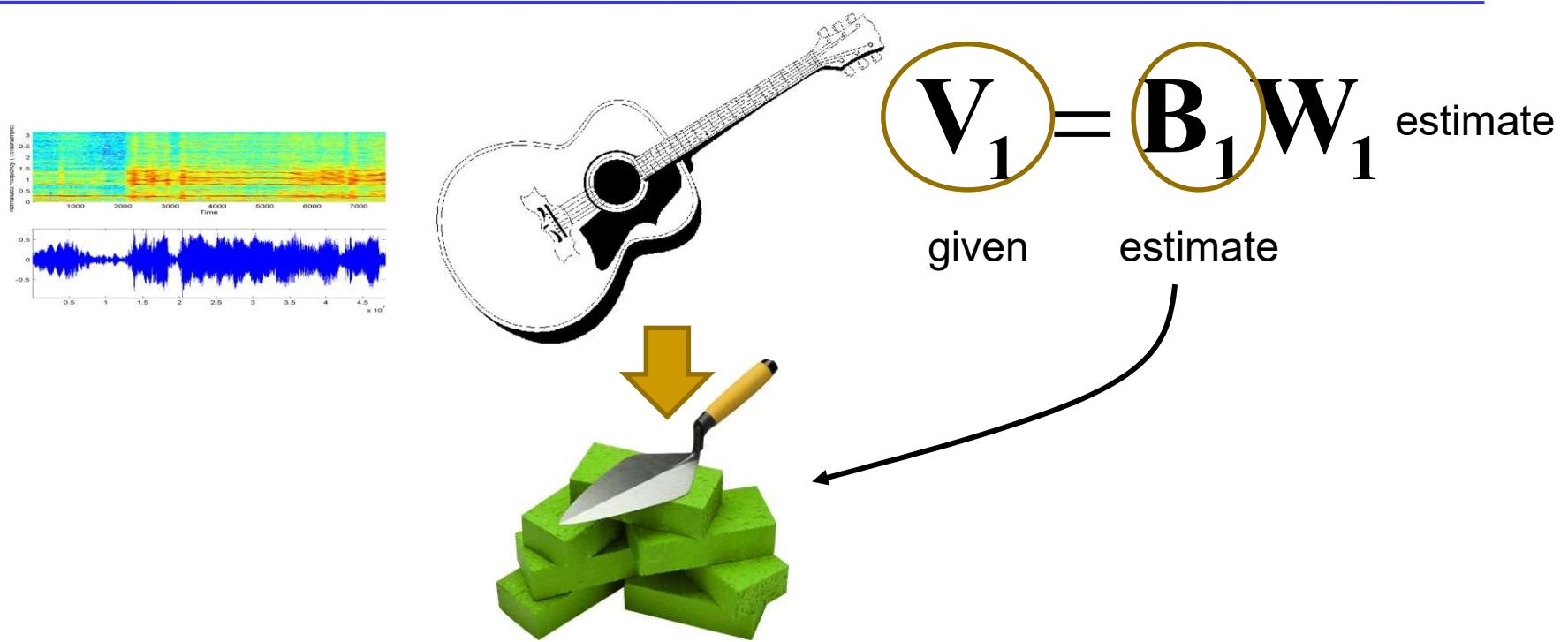
Composition



Building
blocks

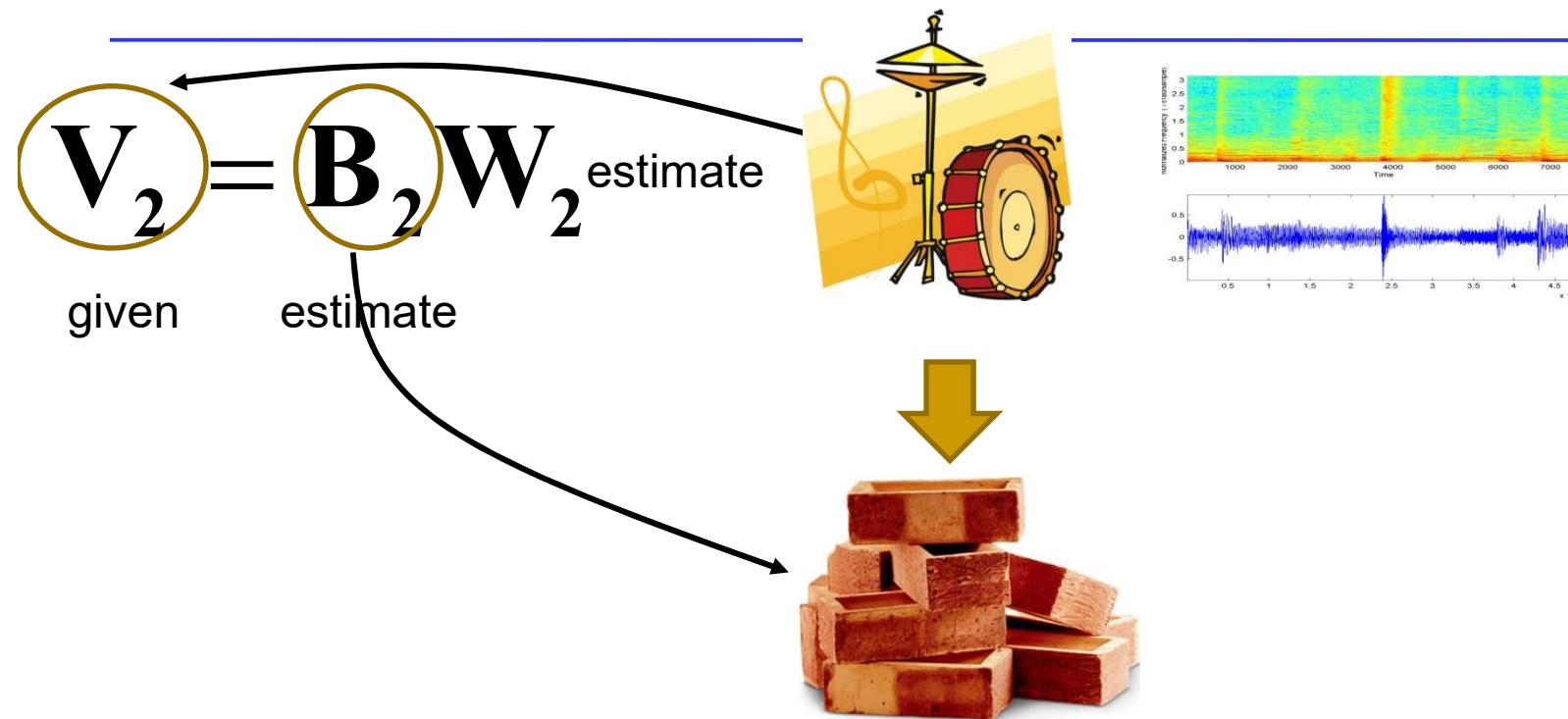
- Given two distinct sets of building blocks, can we find which parts of a composition were composed from which blocks

Separating Sounds



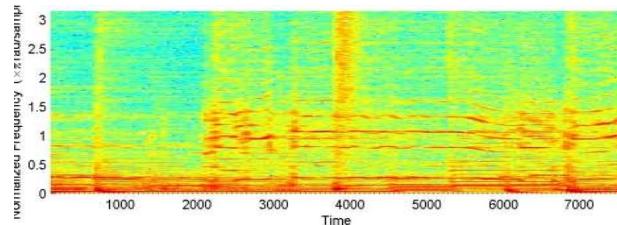
- From example of A, learn blocks A (NMF)

Separating Sounds



- From example of A, learn blocks A (NMF)
- From example of B, learn B (NMF)

Separating Sounds



given

$$V = BW$$

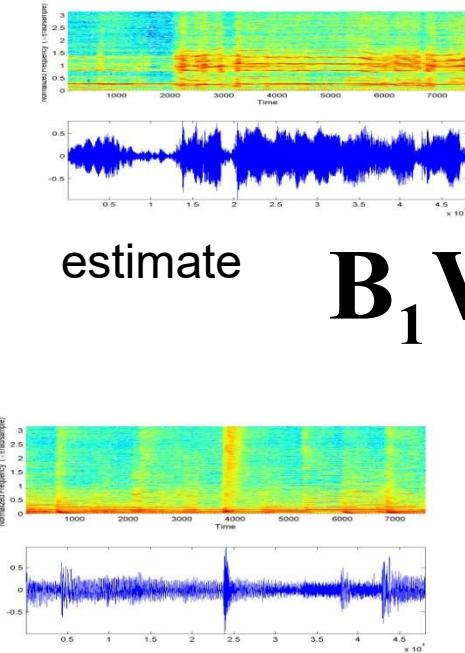
$\begin{bmatrix} B_1 & B_2 \end{bmatrix}$ given $\begin{bmatrix} W_1 \\ W_2 \end{bmatrix}$ estimate

The equation $V = BW$ represents the Non-negative Matrix Factorization (NMF) model. A yellow circle highlights the matrix W , and two arrows point from the text "given" to the matrices B_1 and B_2 , and from the text "estimate" to the matrix W_1 .



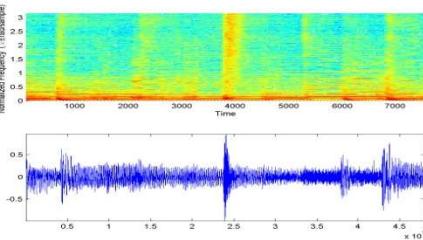
- From mixture, separate out (NMF)
 - Use known “bases” of both sources
 - Estimate the weights with which they combine in the mixed signal

Separating Sounds



estimate

$$\mathbf{B}_1 \mathbf{W}_1$$



estimate

$$\mathbf{B}_2 \mathbf{W}_2$$

$$\mathbf{V} = \mathbf{B} \mathbf{W}$$

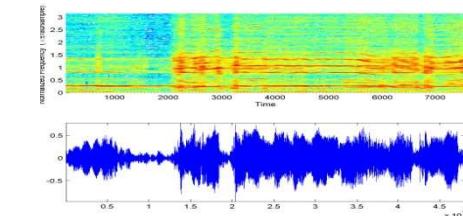
\uparrow

$$[\mathbf{B}_1 \quad \mathbf{B}_2] \begin{bmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \end{bmatrix}$$

given estimate

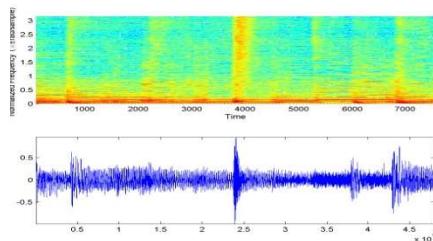
- Separated signals are estimated as the contributions of the source-specific bases to the mixed signal

Separating Sounds



estimate

$$\mathbf{B}_1 \mathbf{W}_1$$



estimate

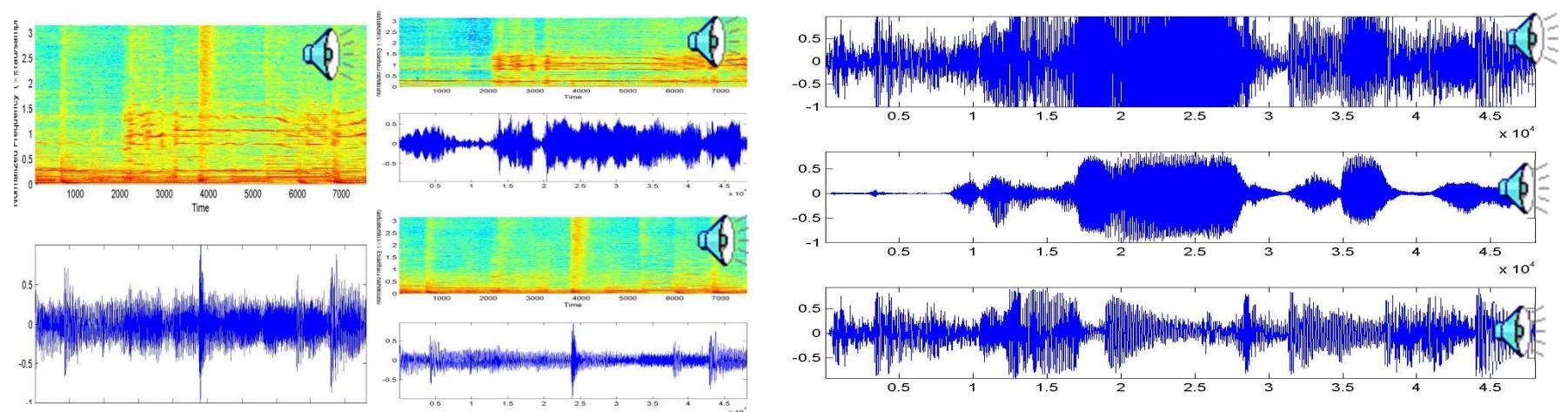
$$\mathbf{B}_2 \mathbf{W}_2$$

$$\mathbf{V} = \mathbf{B} \mathbf{W}$$

given $\begin{bmatrix} \mathbf{B}_1 & \mathbf{B}_2 \end{bmatrix}$ estimate $\begin{bmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \end{bmatrix}$ estimate

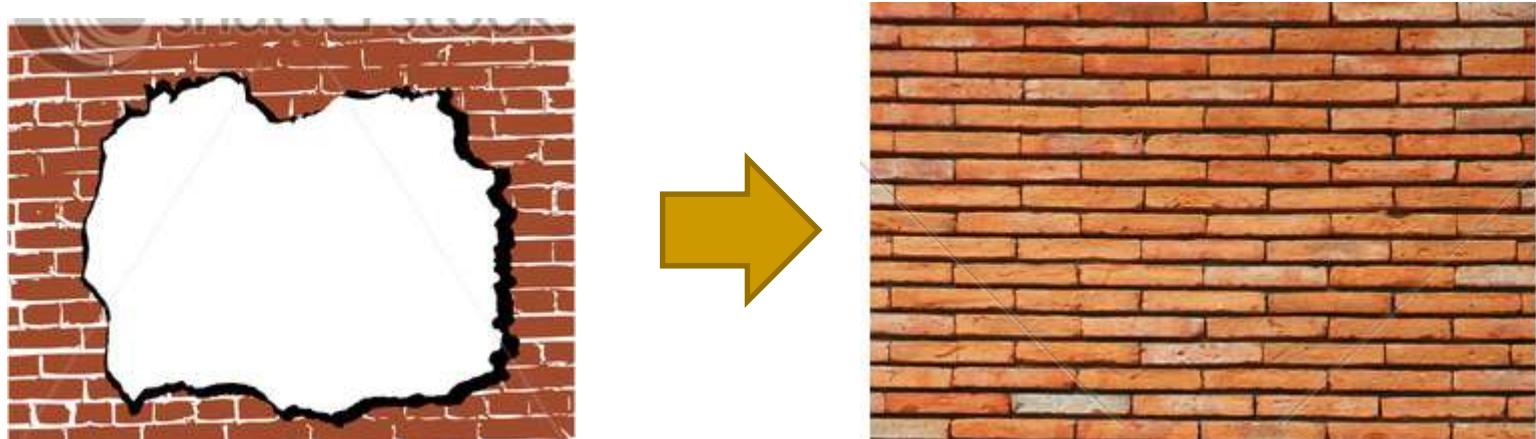
- It is sometimes sufficient to know the bases for only one source
 - The bases for the other can be estimated from the mixed signal itself

Separating Sounds



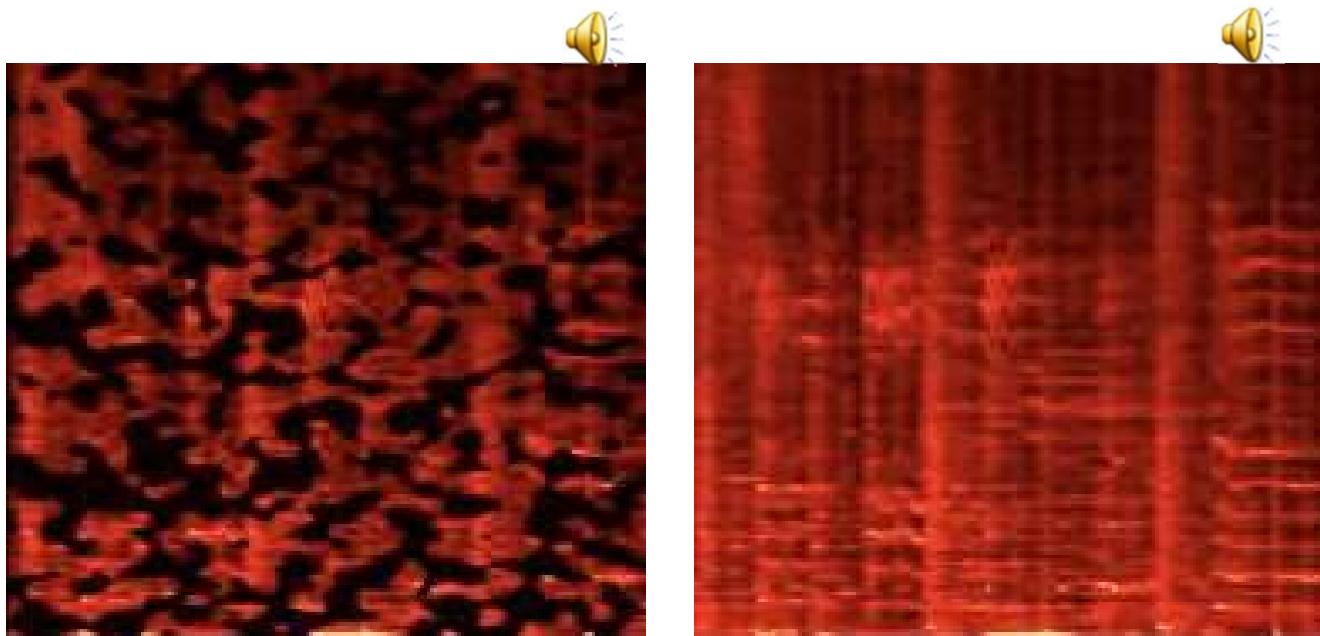
- “Raise my rent” by David Gilmour
- Background music “bases” learnt from 5-seconds of music-only segments within the song
- Lead guitar “bases” bases learnt from the rest of the song
- Norah Jones singing “Sunrise”
- Background music bases learnt from 5 seconds of music-only segments

Predicting Missing Data



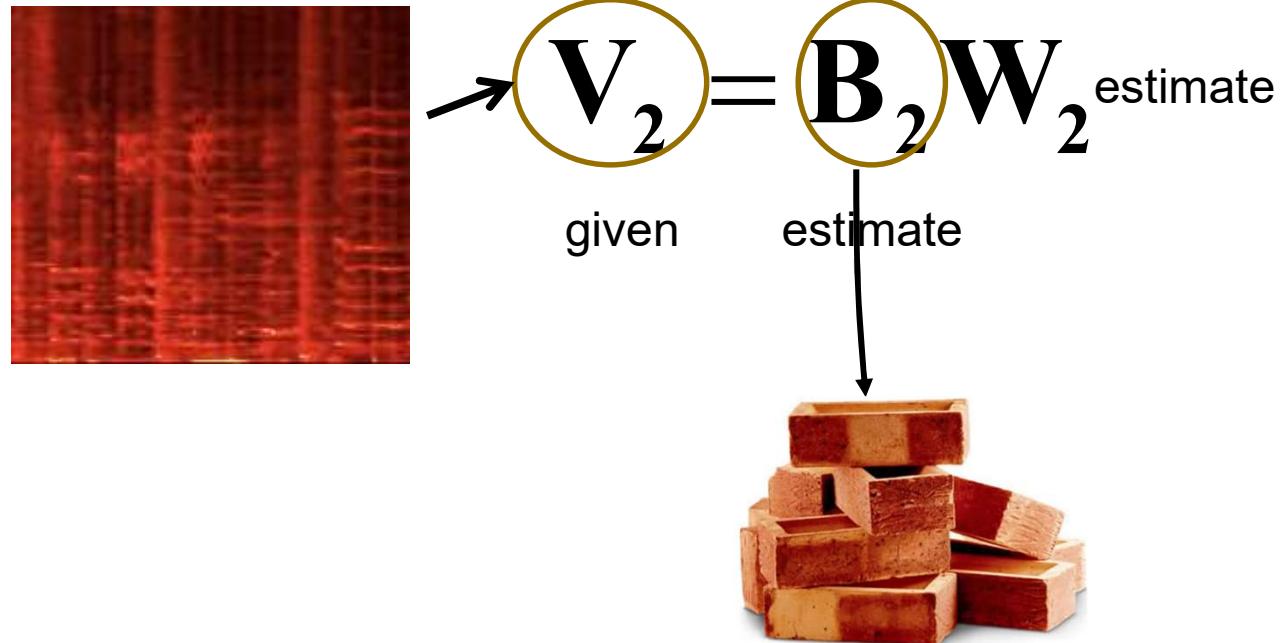
- Use the building blocks to fill in “holes”

Filling in



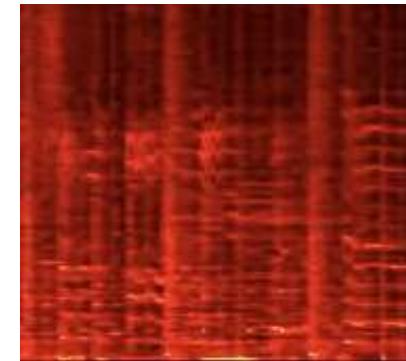
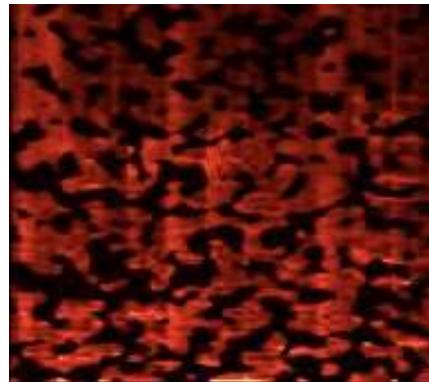
- Some frequency components are missing (left panel)
- We know the bases
 - But not the mixture weights for any particular spectral frame
- We must “fill in” the holes in the spectrogram
 - To obtain the one to the right

Learn building blocks



- Learn the building blocks from other examples of similar sounds
 - E.g. music by same singer
 - E.g. from undamaged regions of same recording

Predict data



$$\hat{V} = \hat{B}W \quad \text{estimate}$$

Modified bases (given)

$V = BW$

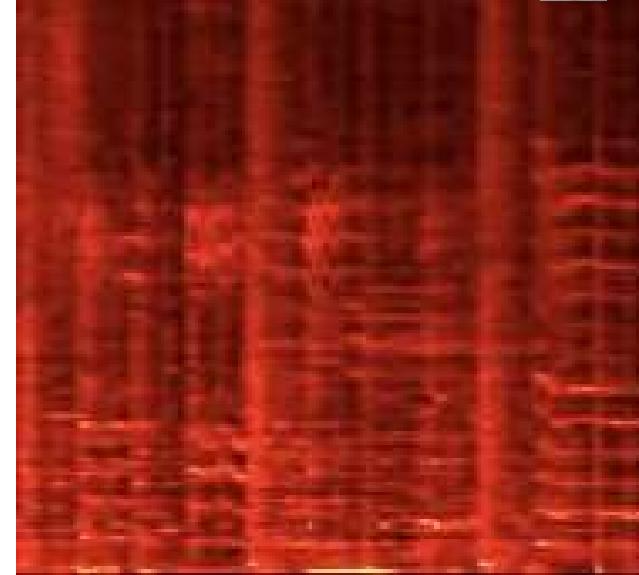
Full bases

estimate

A diagram illustrating the process of signal reconstruction. On the left, a noisy spectrogram \hat{V} is shown, labeled "Modified bases (given)". An arrow points from this to the equation $\hat{V} = \hat{B}W$, which is labeled "estimate". A large orange arrow points from this equation to the right. On the right, a clearer spectrogram V is shown, labeled "Full bases". Above the equation $V = BW$, a green box contains the word "estimate".

- “Modify” bases to look like damaged spectra
 - Remove appropriate spectral components
- Learn how to compose damaged data with modified bases
- Reconstruct missing regions with complete bases

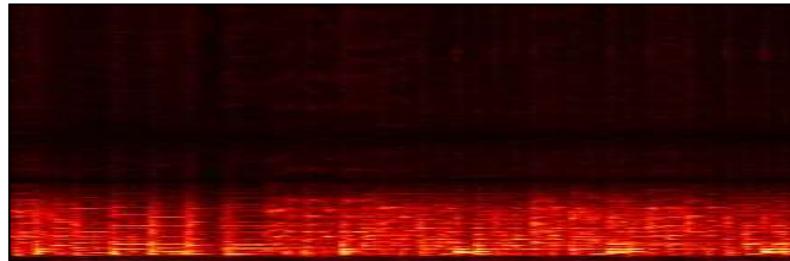
Filling in : An example



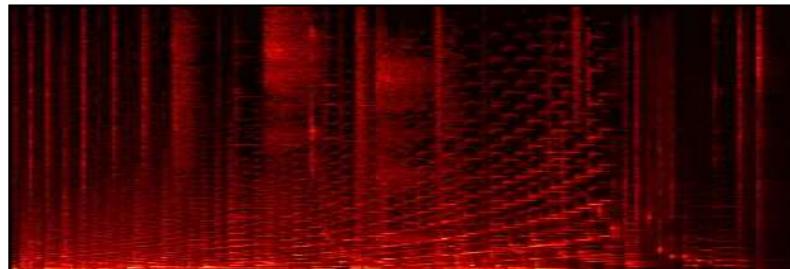
- Madonna...
- Bases learned from other Madonna songs

A more fun example

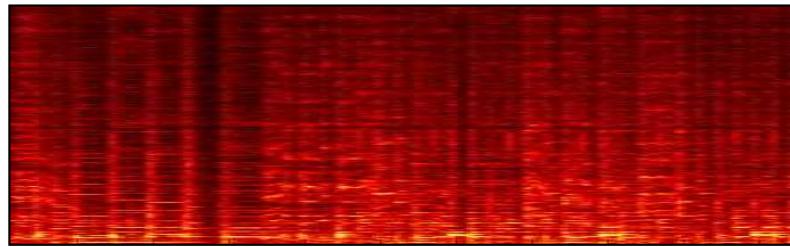
- Reduced BW data



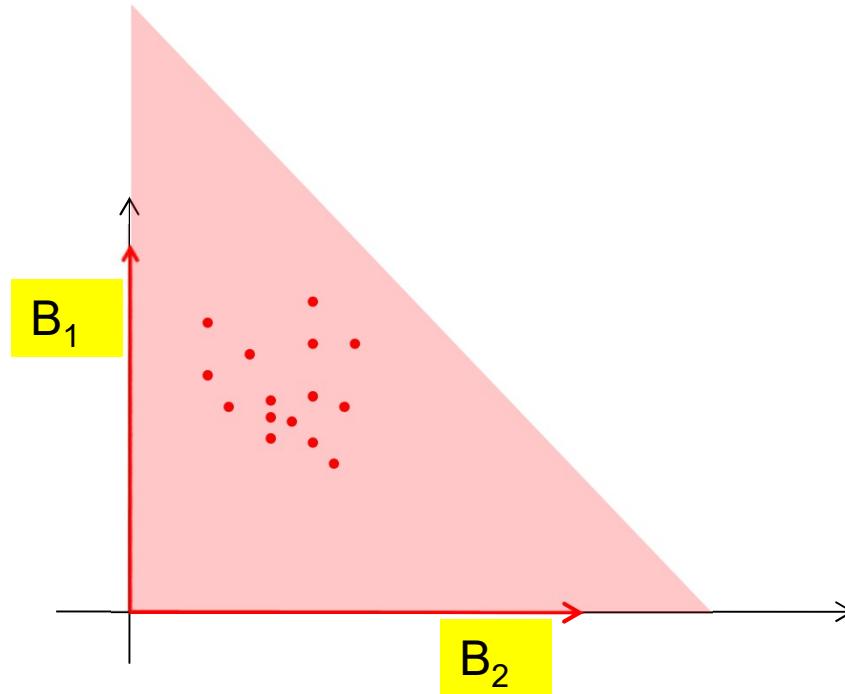
- Bases learned from this



- Bandwidth expanded version



A Natural Restriction



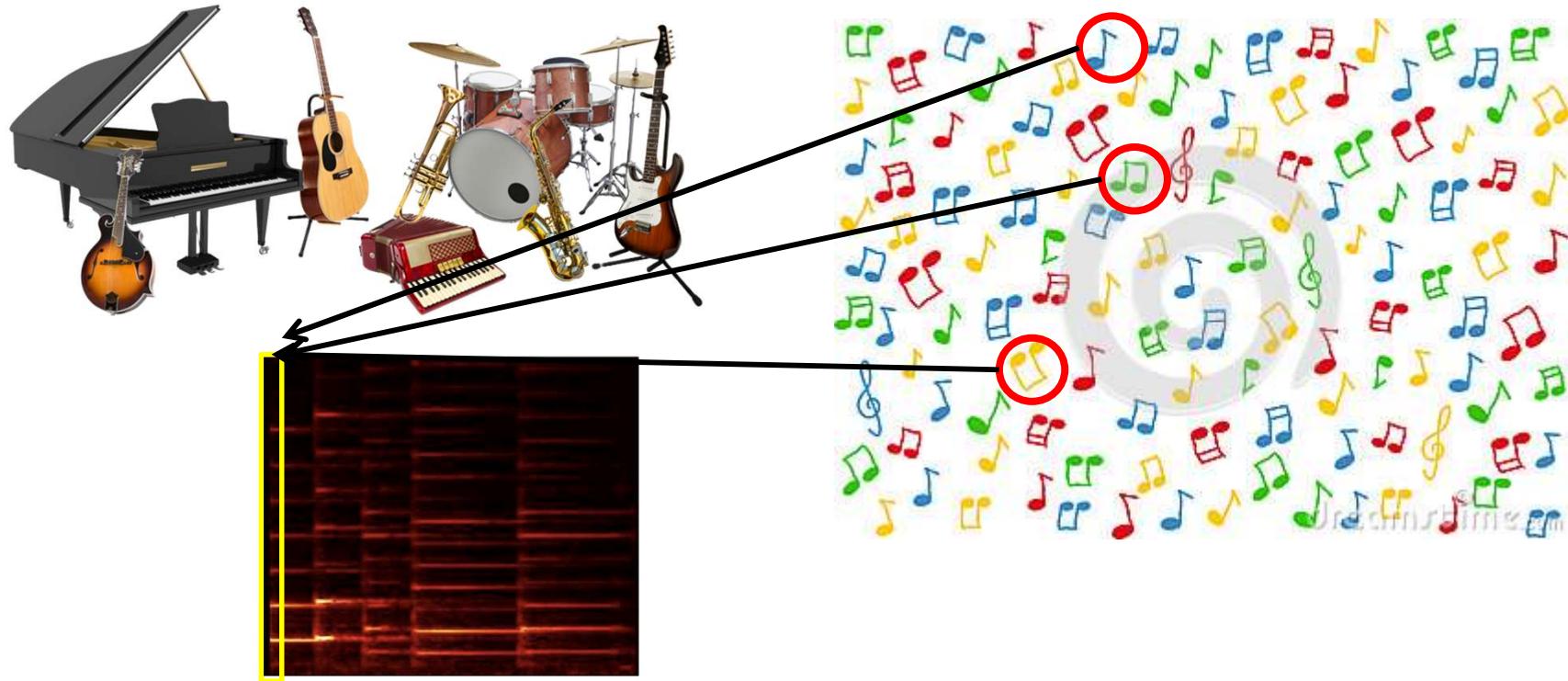
- For K -dimensional data, can learn no more than $K-1$ bases meaningfully
 - At K bases, simply select the axes as bases
 - The bases will represent *all* data exactly

Its an unnatural restriction



- For K-dimensional spectra, can learn no more than K-1 bases
- Nature does not respect the dimensionality of your spectrogram
- E.g. Music: There are tens of instruments
 - Each can produce dozens of unique notes
 - Amounting to a total of many thousands of notes
 - Many more than the dimensionality of the spectrum
- E.g. images: a 1024 pixel image can show millions of recognizable pictures!
 - Many more than the number of pixels in the image

Fixing the restriction: Updated model



- Can have a *very large* number of building blocks (bases)
 - E.g. notes
- But any *particular* frame is composed of only a small subset of bases
 - E.g. any single frame only has a small set of notes

Poll 3

The Modified Model

$$\mathbf{V} = \mathbf{B}\mathbf{W}$$

$$V = BW$$

For one vector



■ Modification 1:

- In any column of \mathbf{W} , only a small number of entries have non-zero value
- I.e. the columns of \mathbf{W} are *sparse*
- These are *sparse* representations

■ Modification 2:

- \mathbf{B} may have more columns than rows
 - These are called *overcomplete* representations
-
- Sparse representations need not be overcomplete, but overcomplete representations need sparsity to provide useful decompositions

Imposing Sparsity

$$\mathbf{V} = \mathbf{B}\mathbf{W}$$

$$E = \text{Div}(\mathbf{V}, \mathbf{B}\mathbf{W})$$

$$Q = \text{Div}(\mathbf{V}, \mathbf{B}\mathbf{W}) + \lambda |\mathbf{W}|_0$$

- Minimize a modified objective function
- Combines divergence and ell-0 norm of \mathbf{W}
 - The number of non-zero elements in \mathbf{W}
- Minimize Q instead of E
 - Simultaneously minimizes both divergence and number of active bases at any time

Imposing Sparsity

$$\mathbf{V} = \mathbf{B}\mathbf{W}$$

$$Q = \text{Div}(\mathbf{V}, \mathbf{B}\mathbf{W}) + \lambda \cancel{\|\mathbf{W}\|_0}$$

$$Q = \text{Div}(\mathbf{V}, \mathbf{B}\mathbf{W}) + \lambda \|\mathbf{W}\|_1$$

- Minimize the ell-0 norm is hard
 - Combinatorial optimization
- Minimize ell-1 norm instead
 - The sum of all the entries in \mathbf{W}
 - *Relaxation*
- Is equivalent to minimize ell-0
 - We cover this equivalence later
- Will also result in sparse solutions

Update Rules

- Modified Iterative solutions
 - In gradient based solutions, gradient w.r.t any W term now includes λ
 - I.e. if $dQ/dW = dE/dW + \lambda$
- For KL Divergence, results in following modified update rules

$$B = B \otimes \frac{\left(\begin{matrix} V \\ BW \end{matrix} \right) W^T}{1 W^T}$$

$$W = W \otimes \frac{B^T \left(\begin{matrix} V \\ BW \end{matrix} \right)}{B^T 1 + \lambda}$$

- Increasing λ makes the weights increasingly sparse

Update Rules

- Modified Iterative solutions
 - In gradient based solutions, gradient w.r.t any W term now includes λ
 - I.e. if $dQ/dW = dE/dW + \lambda$
- Both B and W can be made sparse

$$B = B \otimes \frac{\left(\frac{V}{BW}\right)W^T}{1W^T + \lambda_b}$$

$$W = W \otimes \frac{B^T \left(\frac{V}{BW}\right)}{B^T 1 + \lambda_w}$$

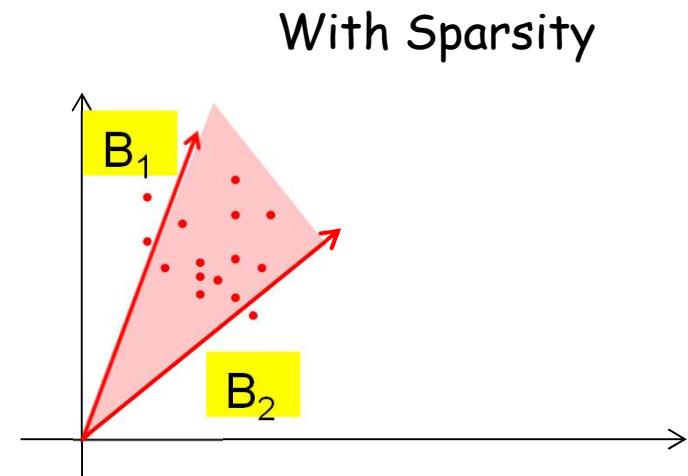
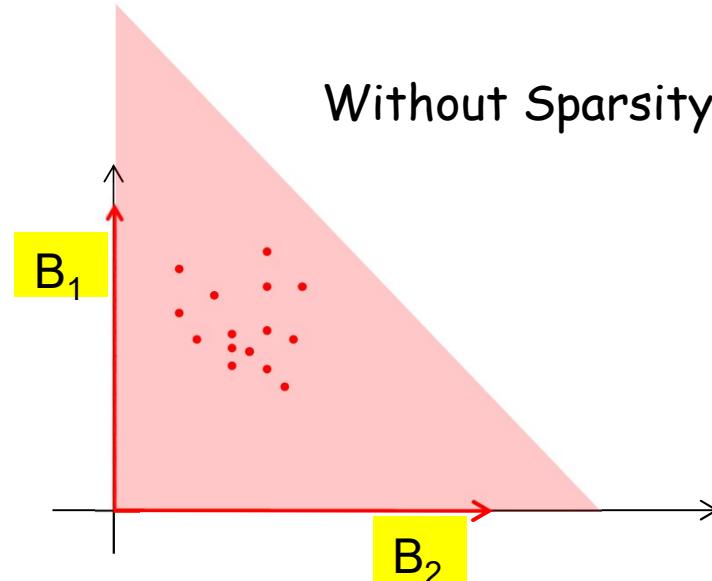
What about Overcompleteness?

- Use the same solutions
- Simply make **B** wide!
 - **W** must be made sparse

$$B = B \otimes \frac{\begin{pmatrix} V \\ BW \end{pmatrix}}{1W^T}$$

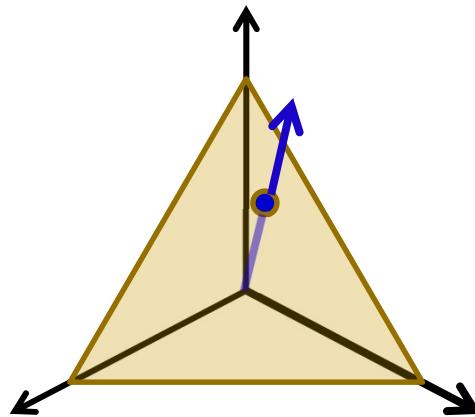
$$W = W \otimes \frac{B^T \begin{pmatrix} V \\ BW \end{pmatrix}}{B^T 1 + \lambda_w}$$

Sparsity: What do we learn

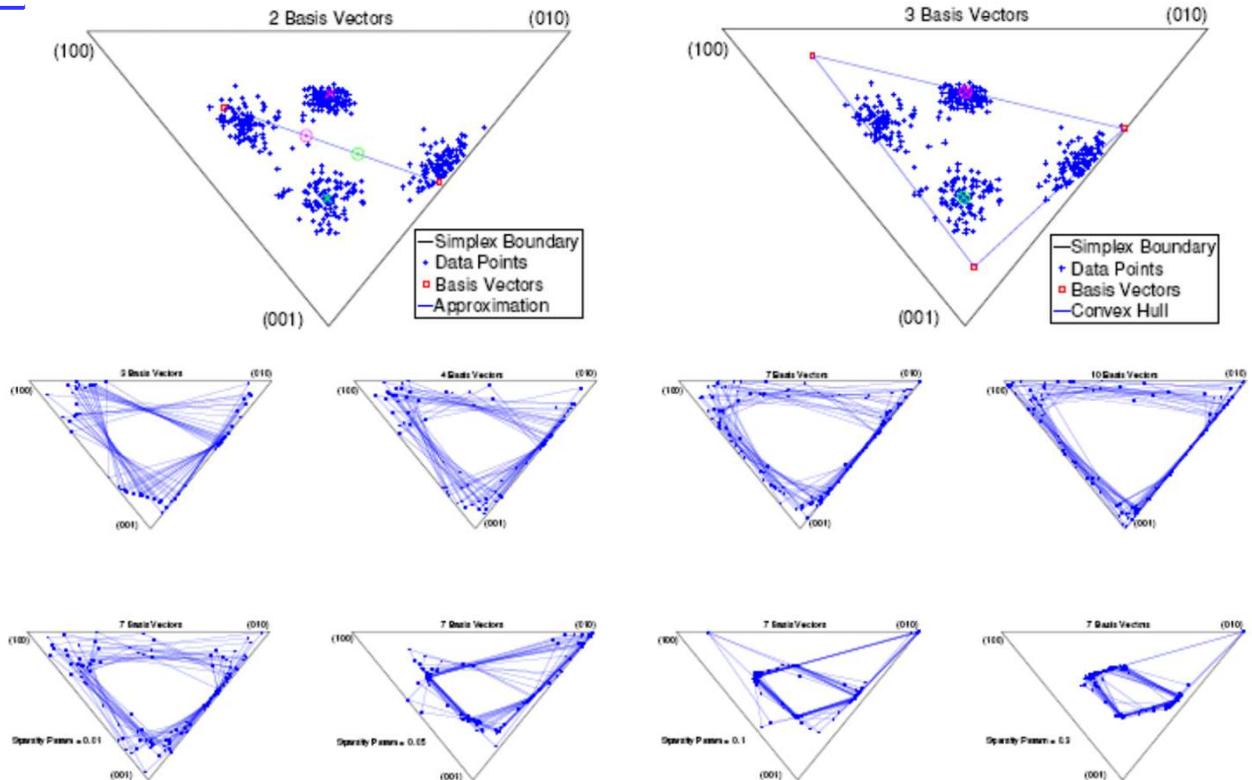


- Without sparsity: The model has an implicit limit: can learn no more than $D-1$ useful bases
 - If $K \geq D$, we can get uninformative bases
- Sparsity: The bases are “pulled towards” the data
 - Representing the distribution of the data much more effectively

Sparsity: What do we learn

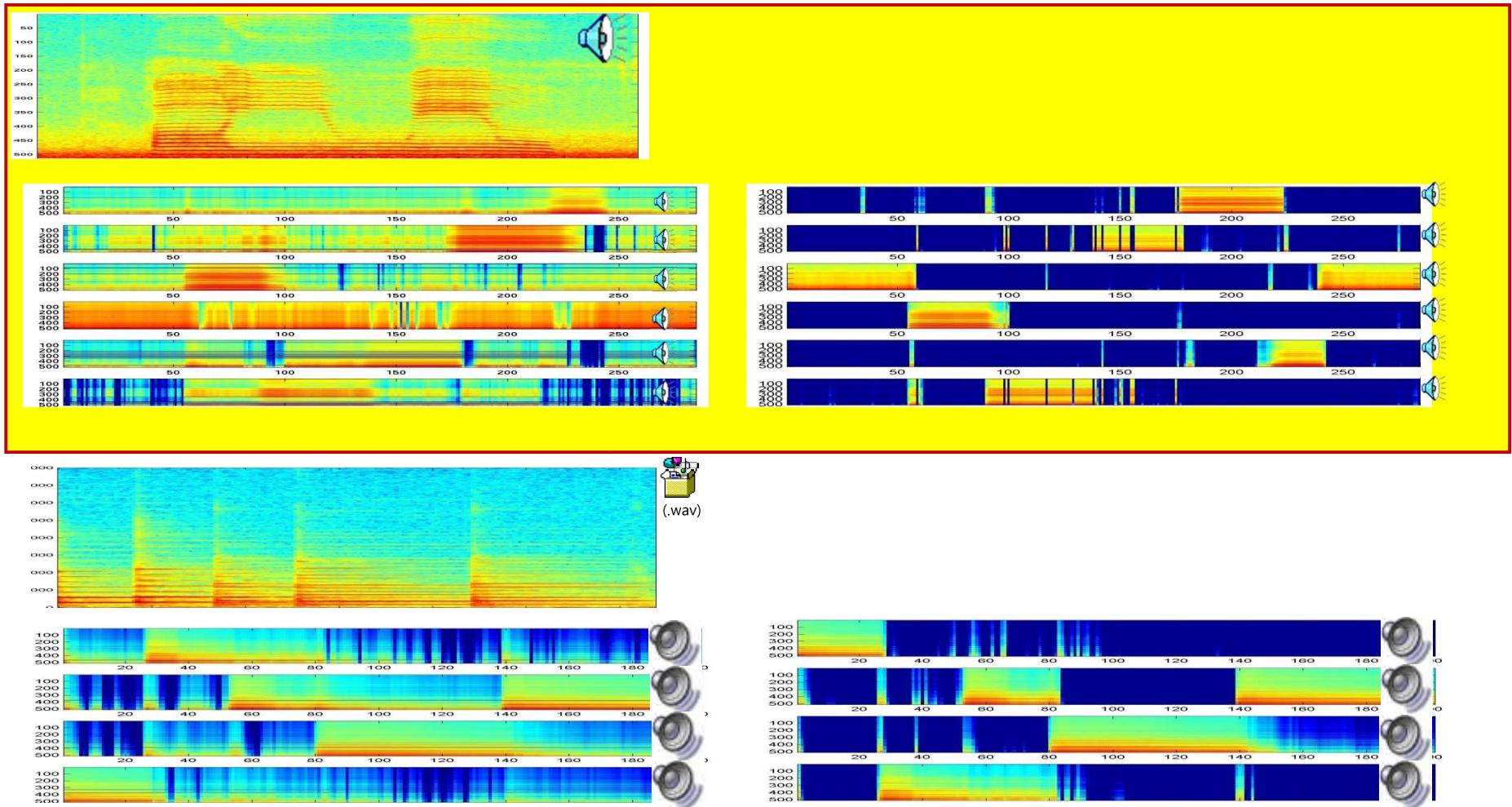


Each dot represents a location where a vector "pierces" the simplex



- Top and middle panel: Compact (non-sparse) estimator
 - As the number of bases increases, bases migrate towards corners of the orthant
- Bottom panel: Sparse estimator
 - Cone formed by bases shrinks to fit the data

The Vowels and Music Examples

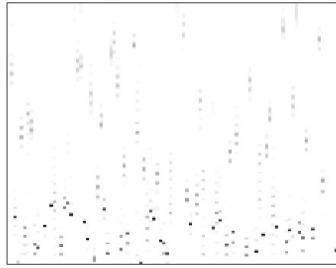


- Left panel, Compact learning: most bases have significant energy in all frames
- Right panel, Sparse learning: Fewer bases active within any frame
 - Decomposition into basic sounds is cleaner

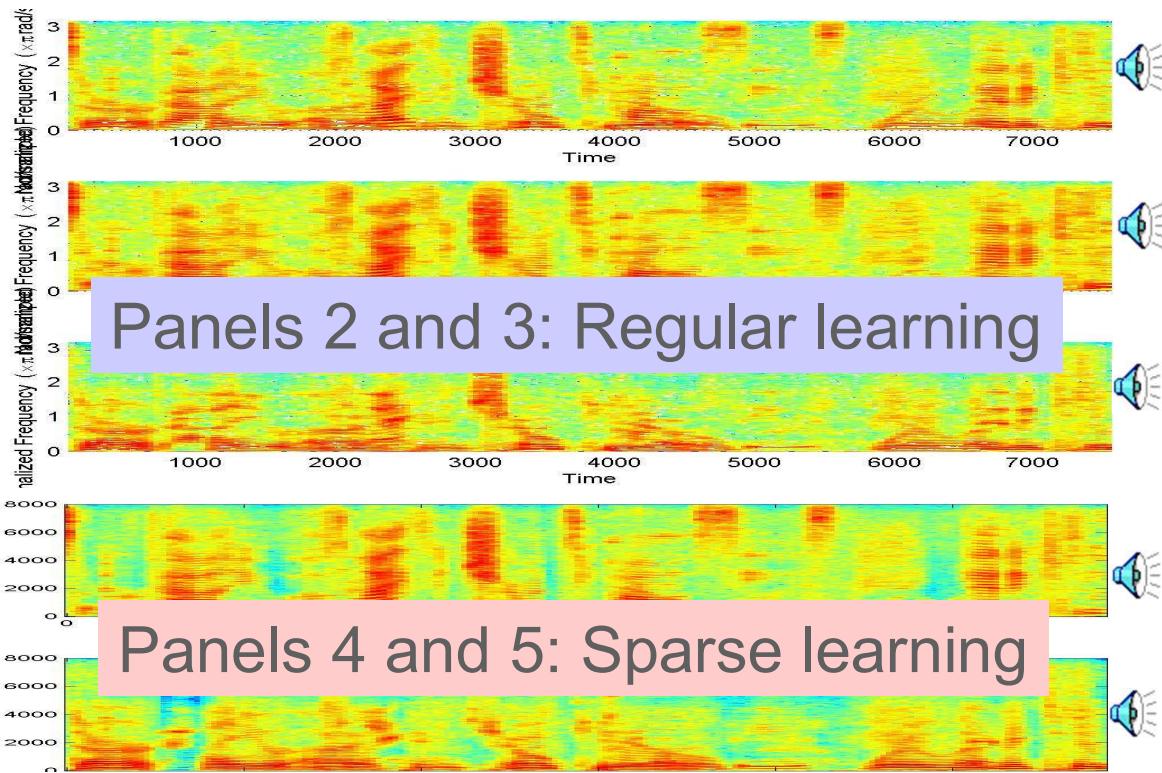
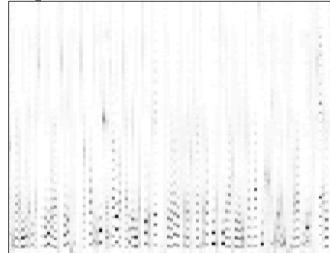
Sparse Overcomplete Bases: Separation

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

Regular bases



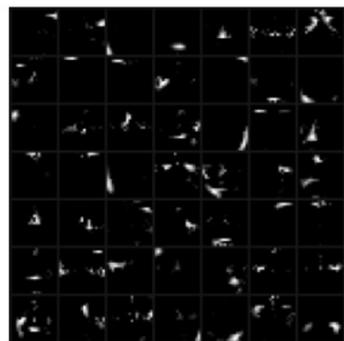
Sparse bases



Sparseness: what do we learn

- 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

Sparse bases

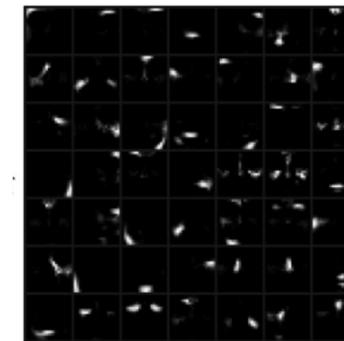


(a)

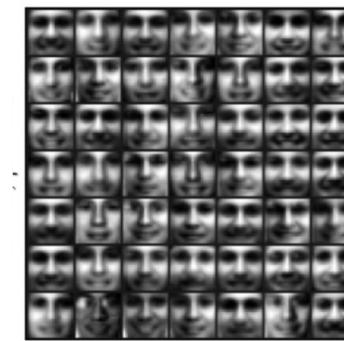


(b)

“Dense” weights



(d)

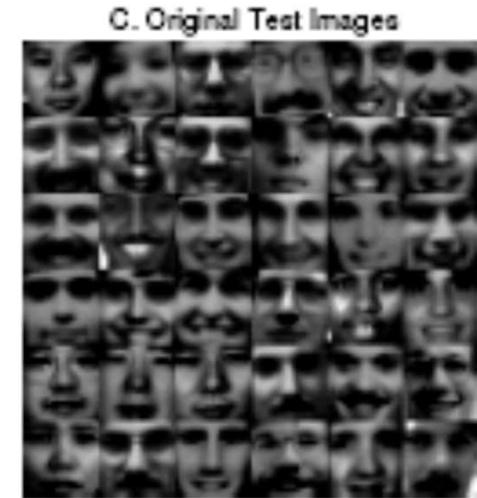
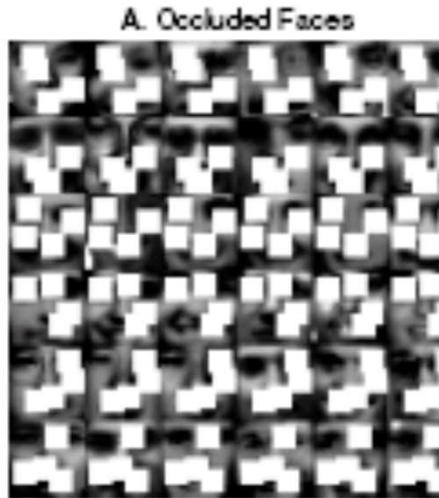


(e)

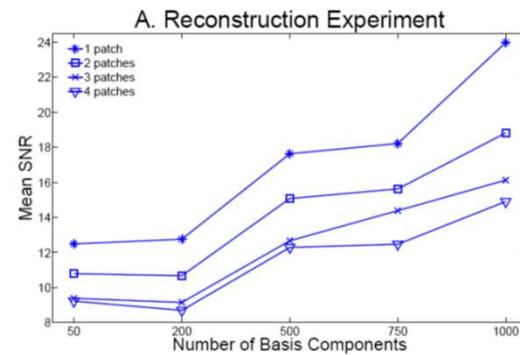
Dense bases

Sparse weights

Filling in missing information

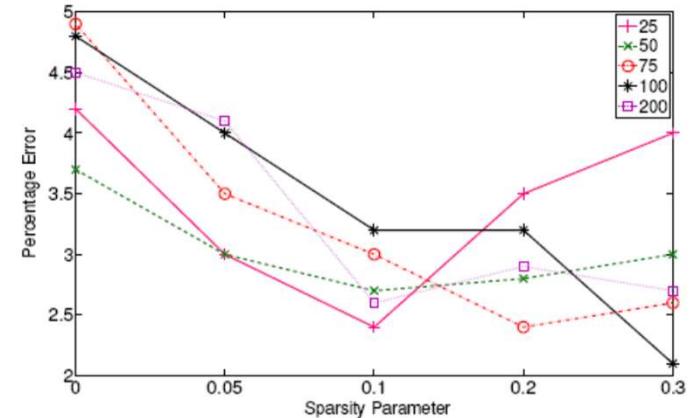
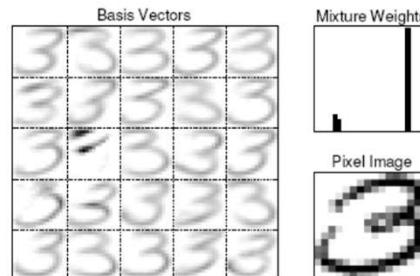
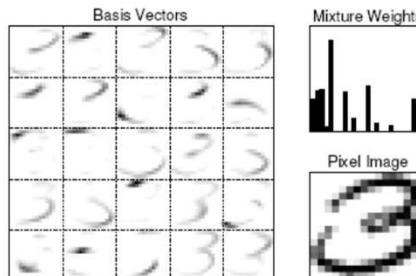
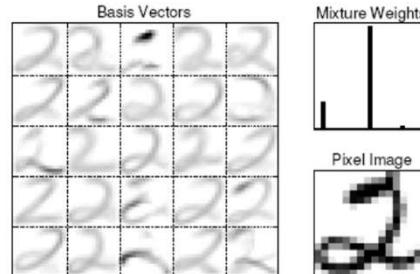
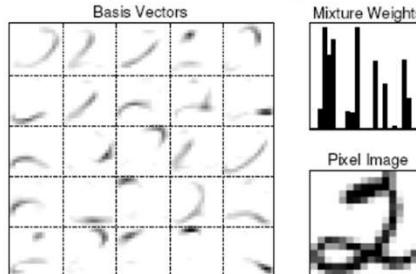


- 19x19 pixel images (361 pixels)
- 1000 bases trained from 2000 faces
- SNR of reconstruction from overcomplete basis set more than 10dB better than reconstruction from corresponding “compact” (regular) basis set



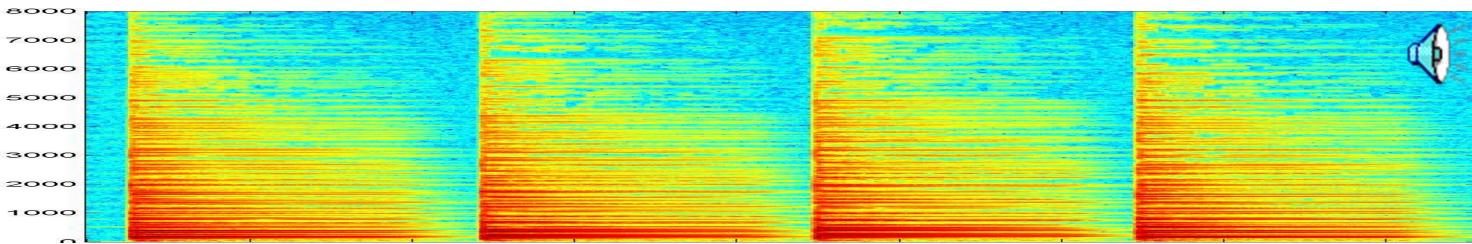
Sparse decomposition for

1 . . .



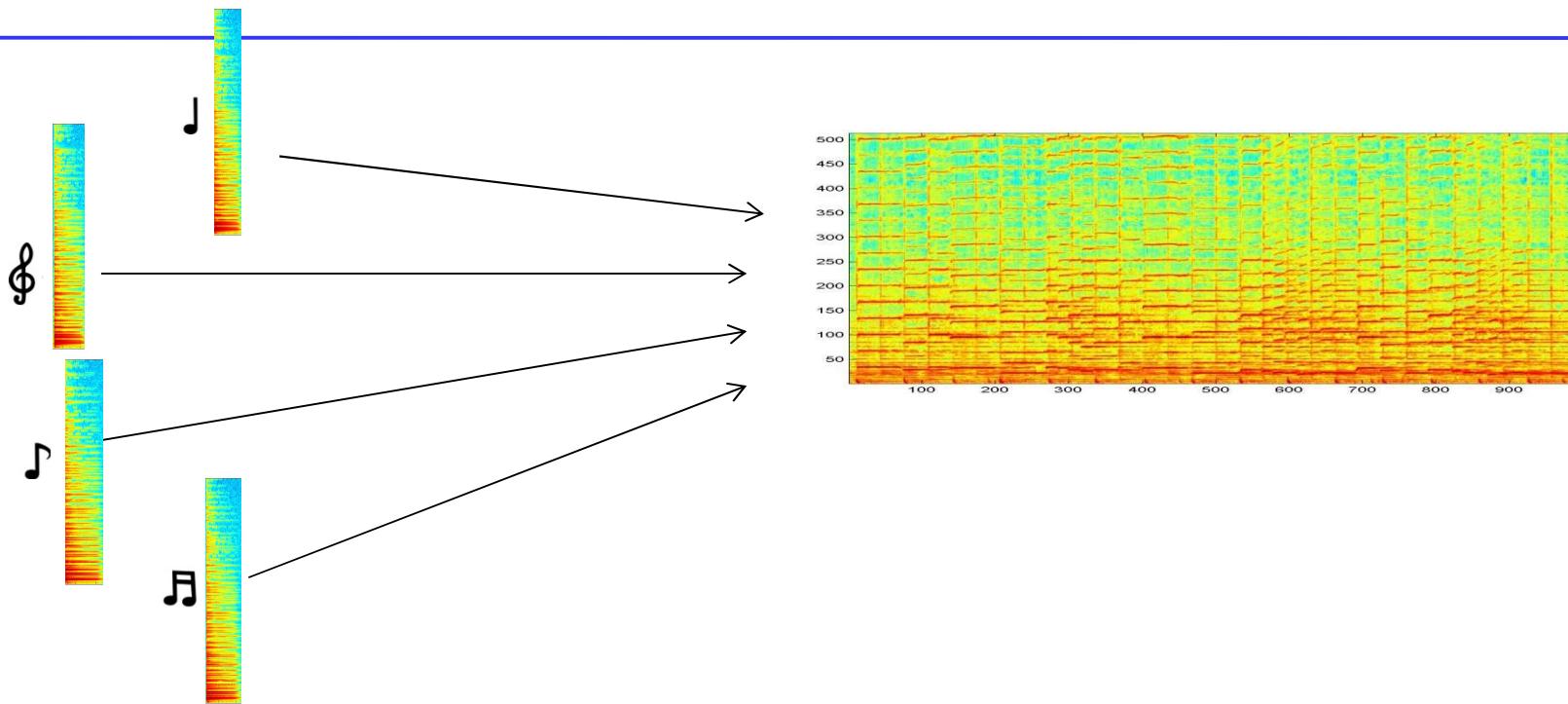
- Given a number of examples of handwritten instances of numbers “2” and “3”
 - Find bases for “2” and “3”
- For any test instance, attempt to construct it using the bases for 2 and (separately) the bases for 3
- The set whose bases result in the better reconstruction is selected
- Accuracy improves with increasing sparsity

Extending the model



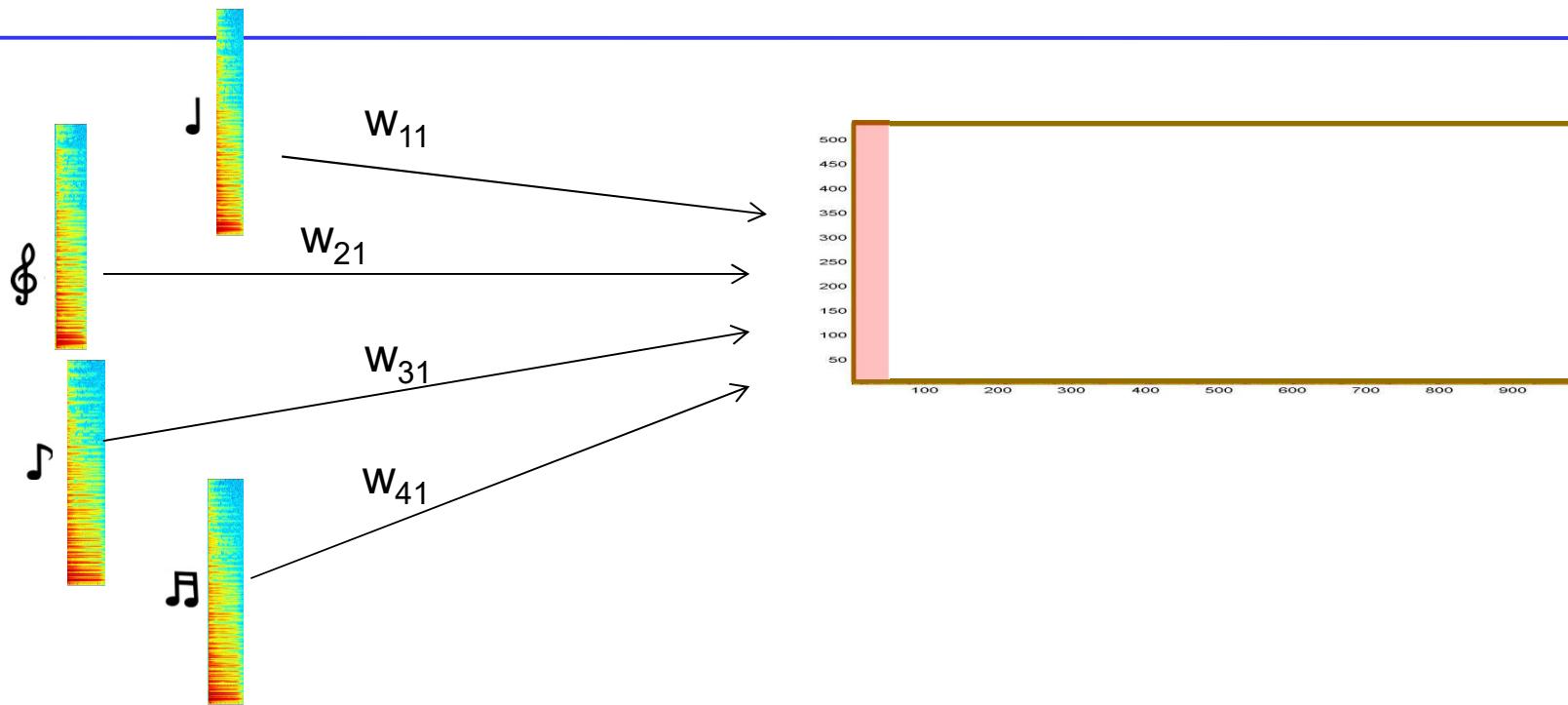
- In reality our building blocks are not spectra
- They are spectral patterns!
 - Which change with time

Convulsive NMF



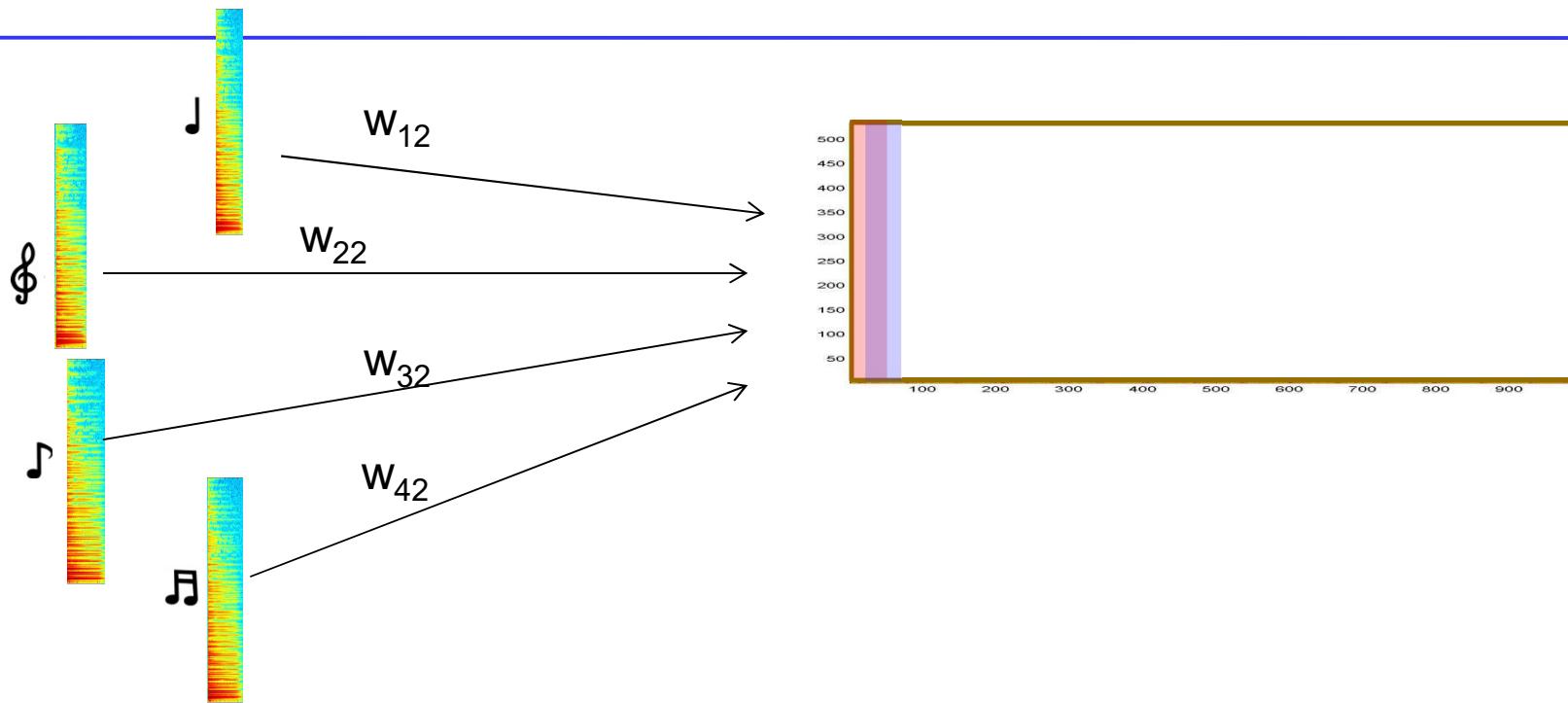
- The building blocks of sound are spectral patches!

Convulsive NMF



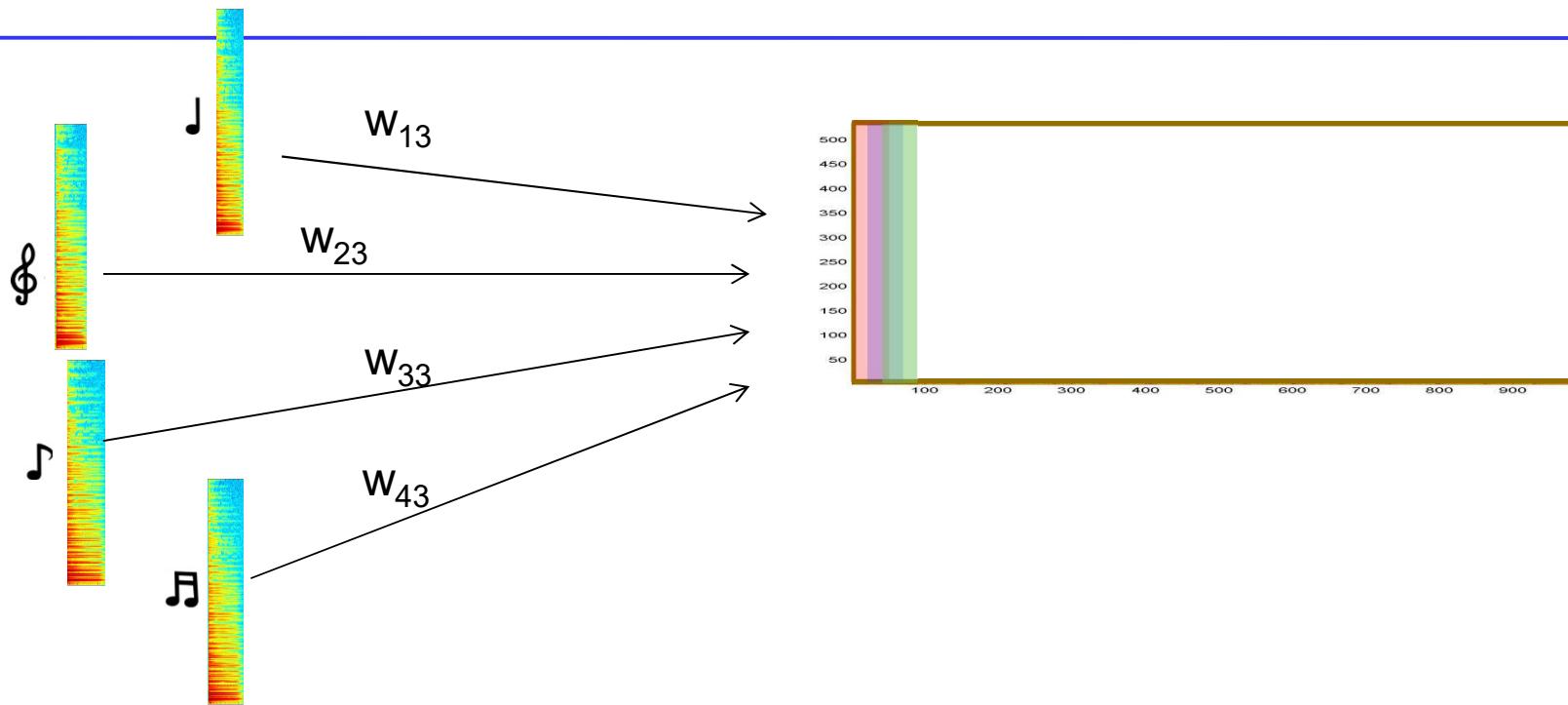
- The building blocks of sound are spectral patches!
- At each time, they combine to compose a patch starting from that time
- Overlapping patches *add*

Convulsive NMF



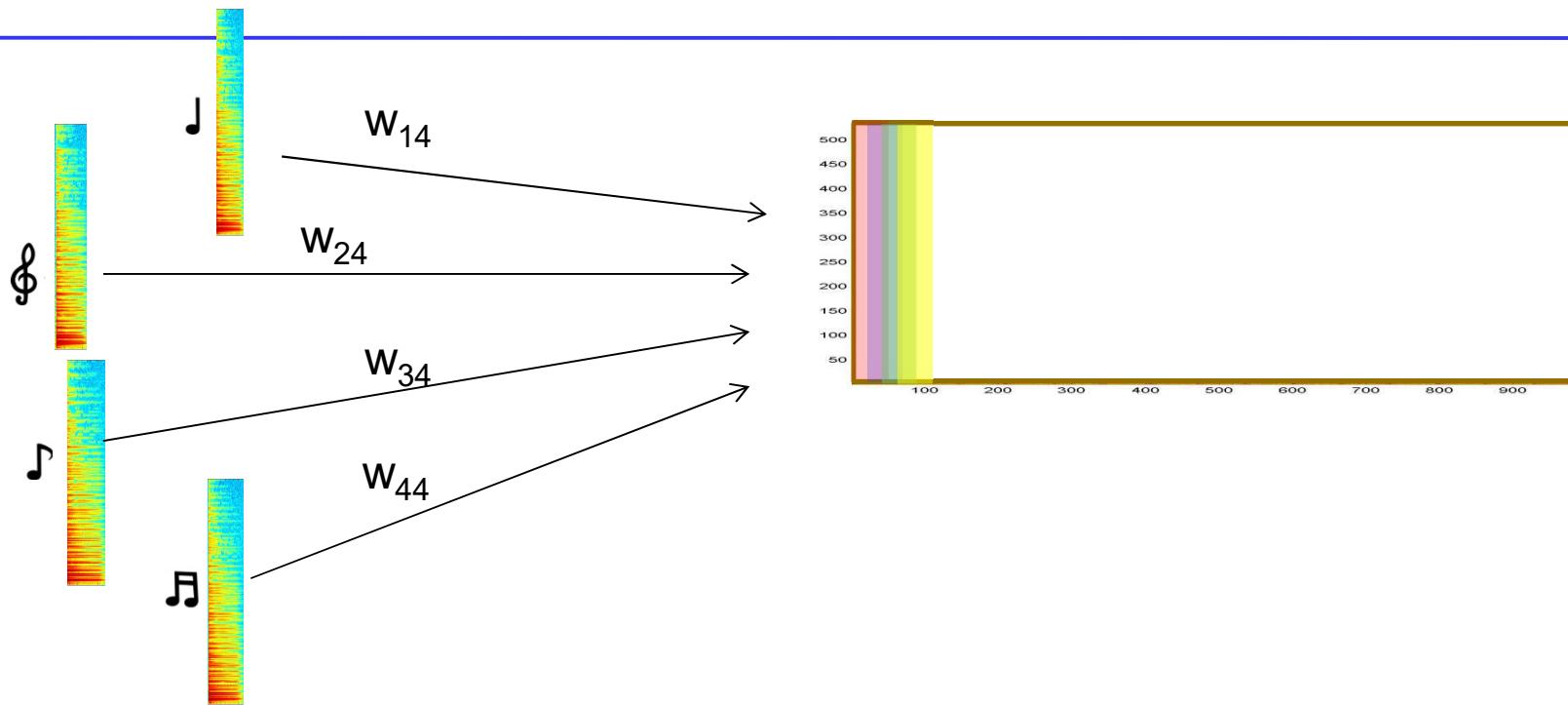
- The building blocks of sound are spectral patches!
- At each time, they combine to compose a patch starting from that time
- Overlapping patches *add*

Convulsive NMF



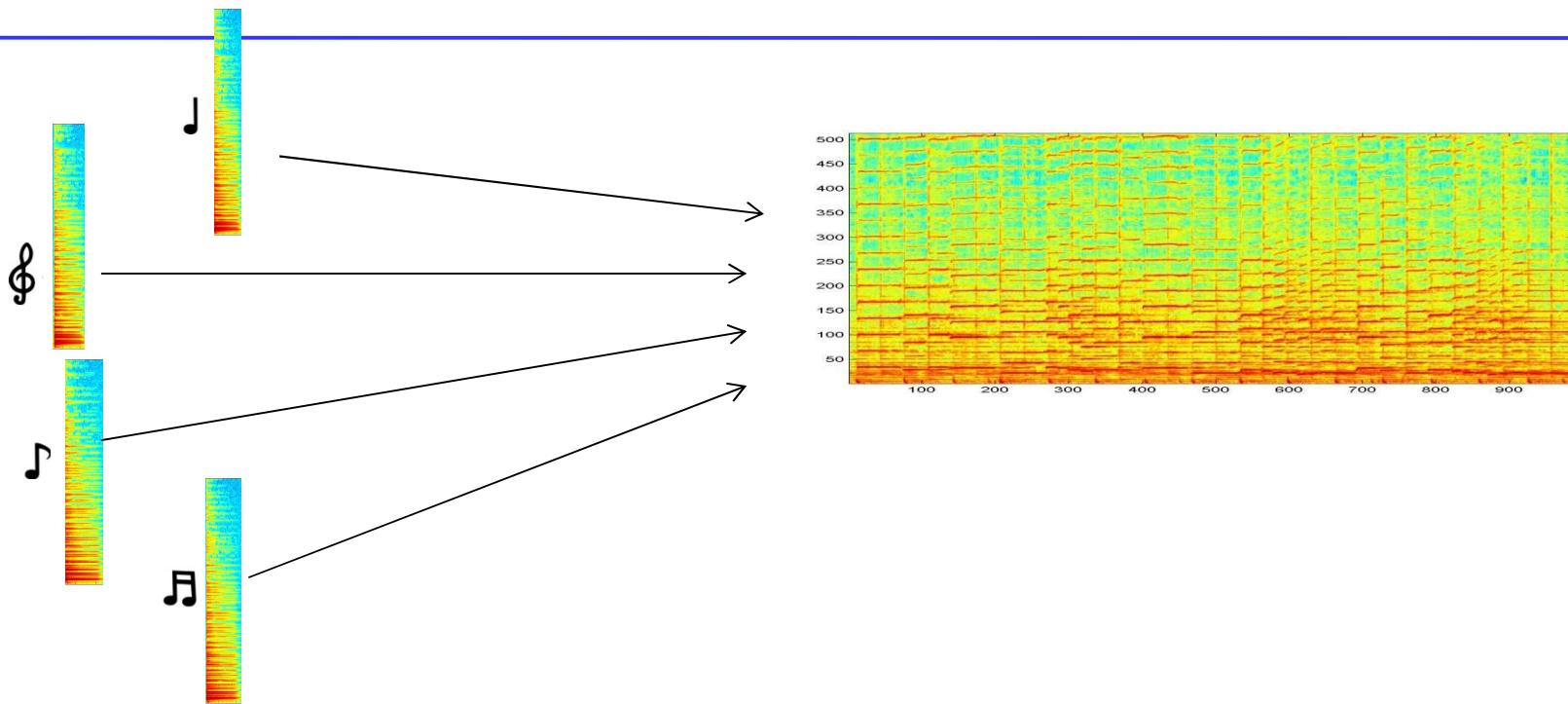
- The building blocks of sound are spectral patches!
- At each time, they combine to compose a patch starting from that time
- Overlapping patches *add*

Convulsive NMF



- The building blocks of sound are spectral patches!
- At each time, they combine to compose a patch starting from that time
- Overlapping patches *add*

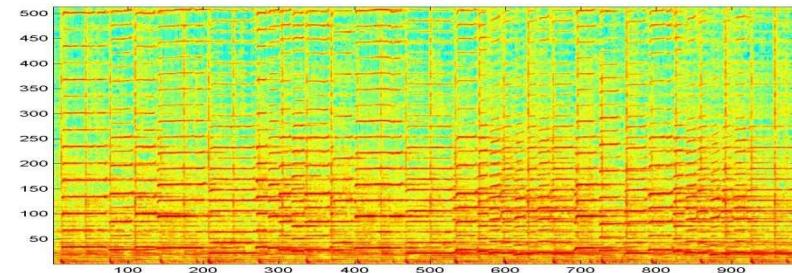
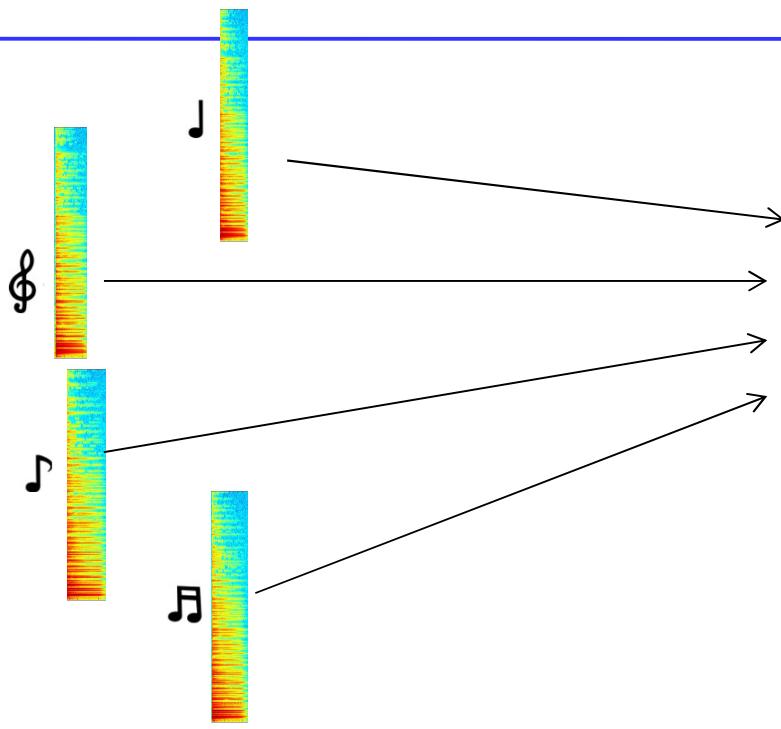
Convulsive NMF



- The building blocks of sound are spectral patches!
- At each time, they combine to compose a patch starting from that time
- Overlapping patches *add*

Poll 4

In Math

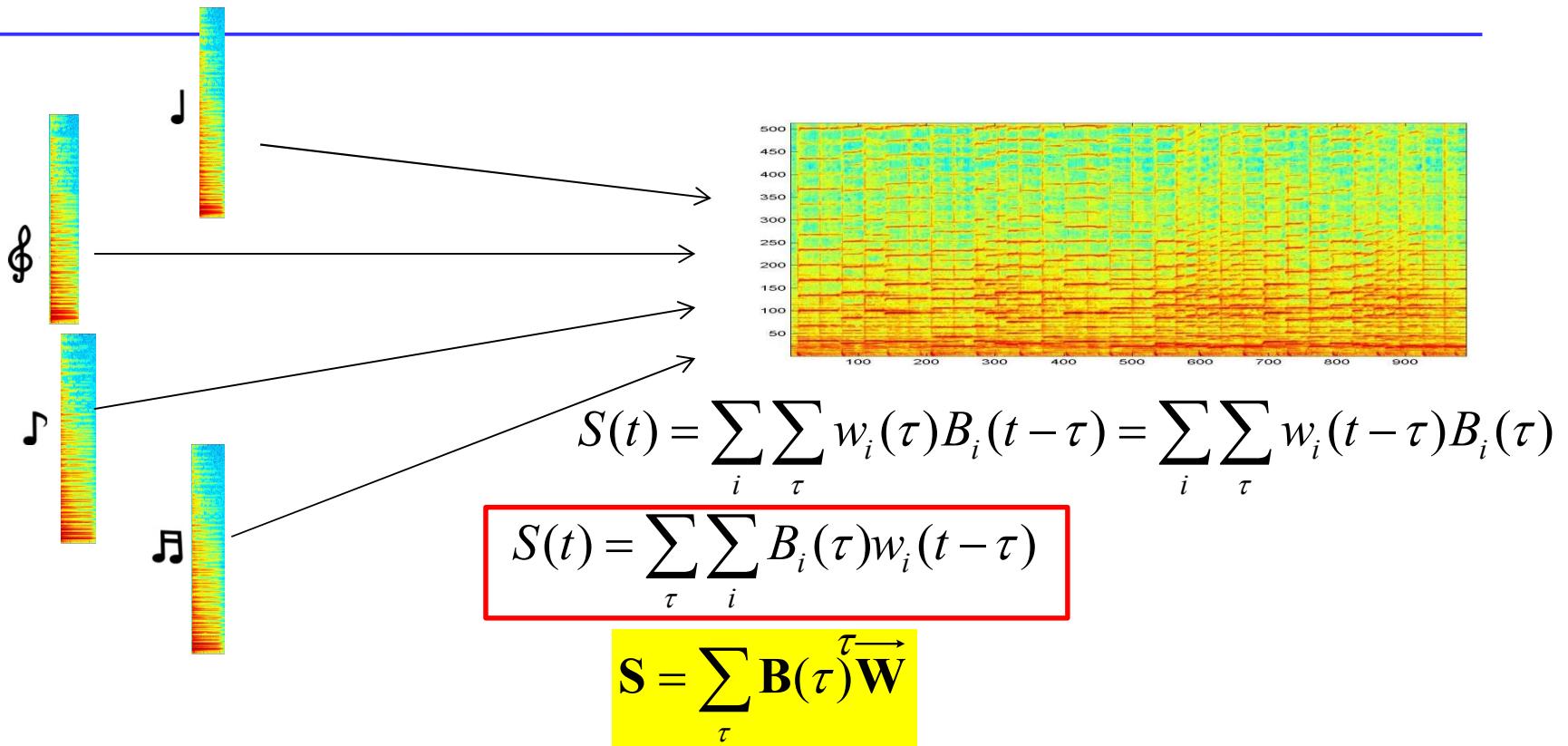


$$S(t) = \sum_i w_i(0)B_i(t) + \sum_i w_i(1)B_i(t-1) + \sum_i w_i(2)B_i(t-2) + \dots = \sum_i \sum_{\tau} w_i(\tau)B_i(t-\tau)$$

$$S(t) = \sum_i B_i(t) \otimes w_i(t)$$

- Each spectral frame has contributions from several previous shifts

An Alternate Representation



- $\mathbf{B}(t)$ is a matrix composed of the t -th columns of all bases
 - The i -th column represents the i -th basis
- \mathbf{W} is a matrix whose i -th row is sequence of weights applied to the i -th basis
 - The superscript $t \rightarrow$ represents a right shift by t

Convulsive NMF

$$\hat{\mathbf{S}} = \sum_{\tau} \mathbf{B}(\tau) \overrightarrow{\mathbf{W}}$$

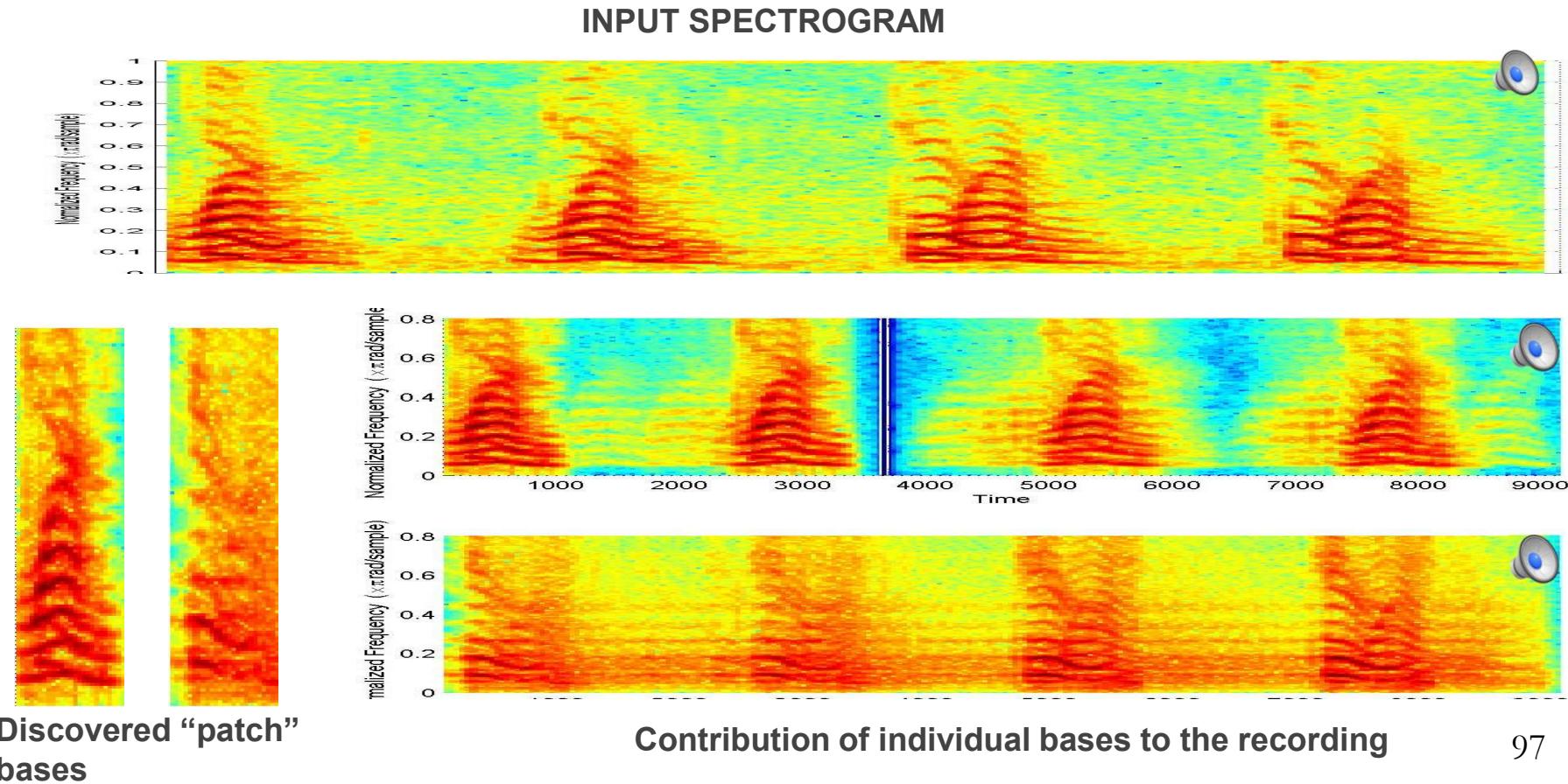
$$\mathbf{B}(t) = \mathbf{B}(t) \otimes \frac{\mathbf{S} \xrightarrow{t \rightarrow T} \hat{\mathbf{S}}}{\mathbf{1} \cdot \mathbf{W}}$$

$$\mathbf{W} = \frac{1}{T} \sum_t \mathbf{W} \otimes \frac{\mathbf{B}(t) \begin{bmatrix} \mathbf{S} \\ \hat{\mathbf{S}} \end{bmatrix}^{\leftarrow t}}{\mathbf{B}(t)^T \mathbf{1}}$$

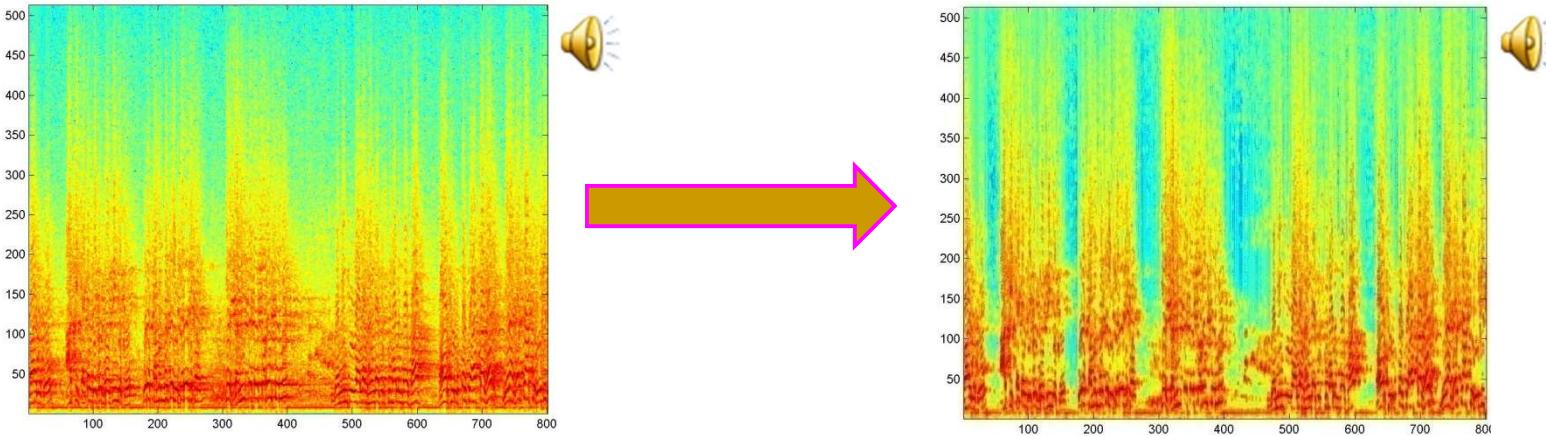
- Simple learning rules for \mathbf{B} and \mathbf{W}
- Identical rules to estimate \mathbf{W} given \mathbf{B}
 - Simply don't update \mathbf{B}
- Sparsity can be imposed on \mathbf{W} as before if desired

The Convulsive Model

- An Example: Two distinct sounds occurring with different repetition rates within a signal
 - Each sound has a time-varying spectral structure

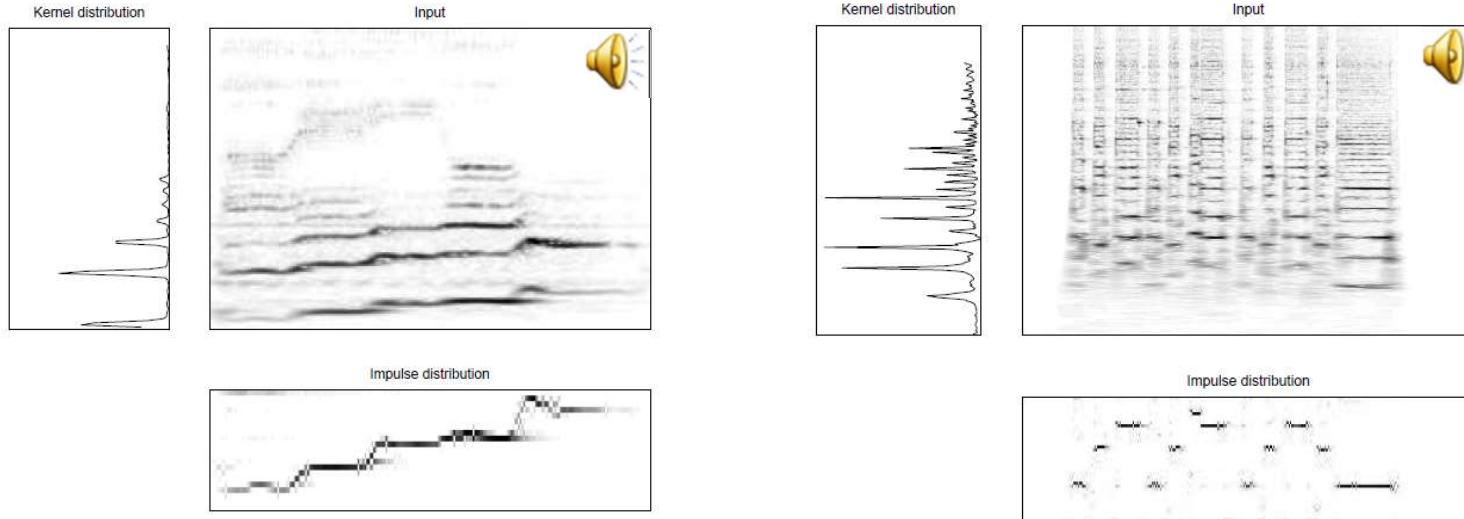


Example applications: Dereverberation



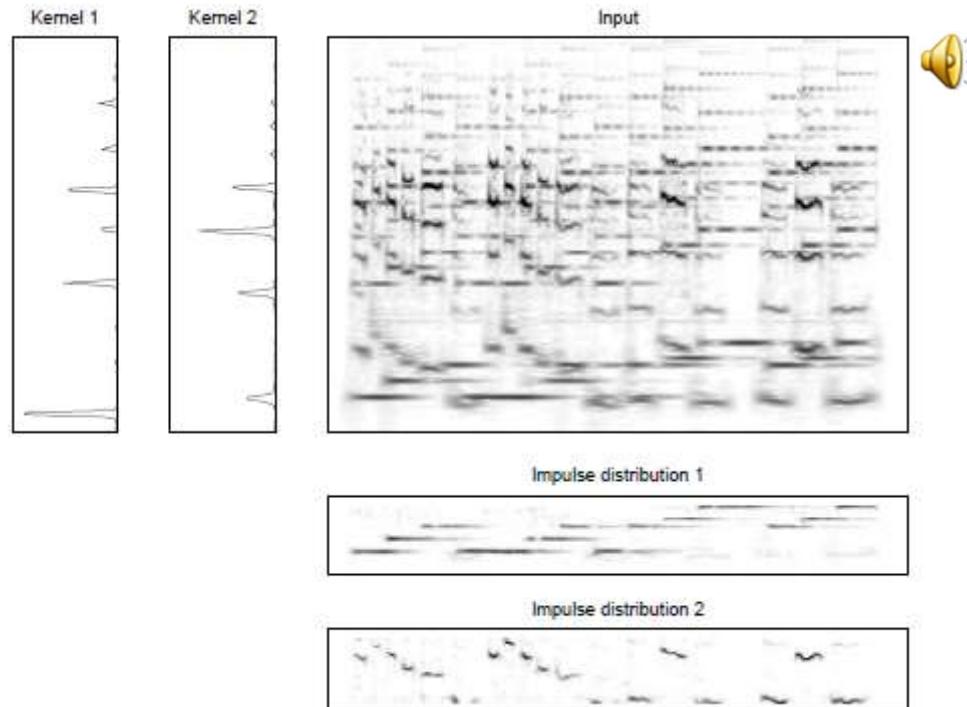
- From “Adrak ke Panje” by Babban Khan
- Treat the reverberated spectrogram as a composition of many shifted copies of a “clean” spectrogram
 - “Shift-invariant” analysis
- NMF to estimate clean spectrogram

Pitch Tracking



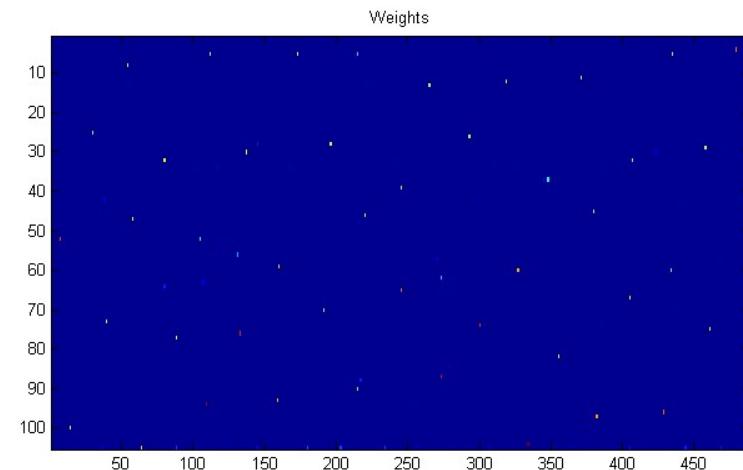
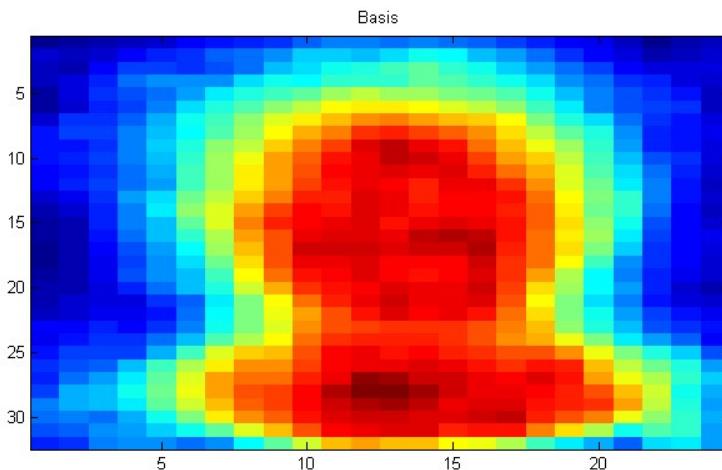
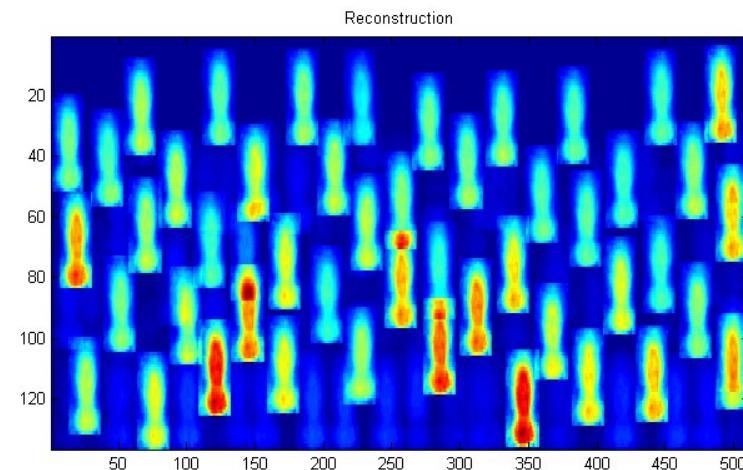
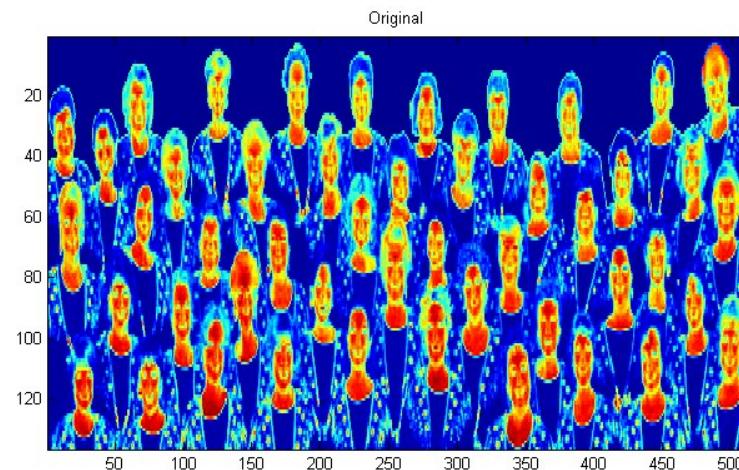
- Left: A segment of a song
- Right: Smoke on the water
 - “Impulse” distribution captures the “melody”!

Pitch Tracking



- Simultaneous pitch tracking on multiple instruments
- Can be used to find the velocity of cars on the highway!!
 - “Pitch track” of sound tracks Doppler shift (and velocity) 100

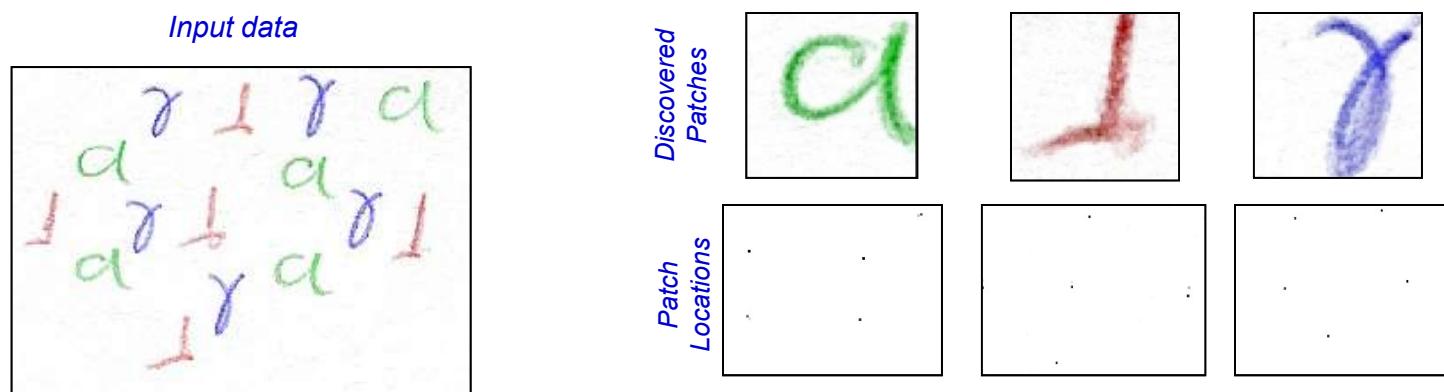
Example: 2-D shift invariance



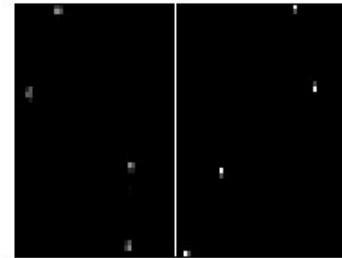
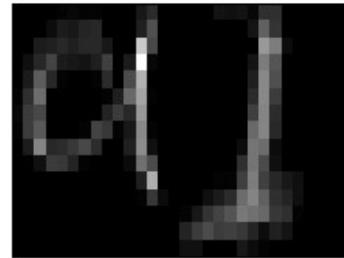
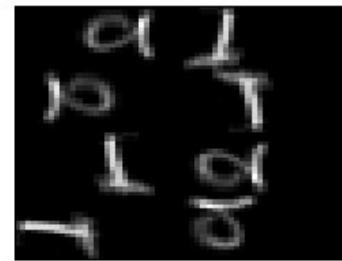
- Sparse decomposition employed in this example
 - Otherwise locations of faces (bottom right panel) are not precisely determined

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



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

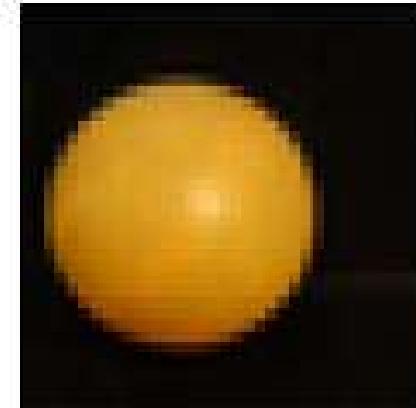
Example: Higher dimensional data

■ Video example

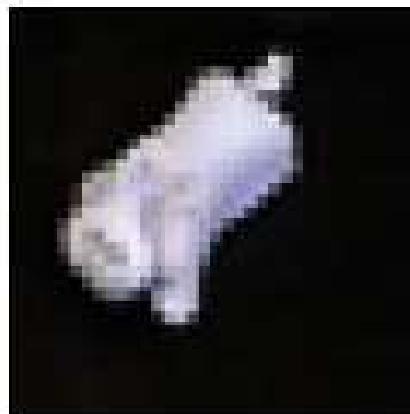
Description of Input



Kernel 1



Kernel 2



Kernel 3



Lessons learned

- Linear decomposition when constrained with semantic constraints e.g. non-negativity can result in semantically meaningful bases
- NMF: Useful *compositional* model of data
- Really effective when the data obey compositional rules..