



CS598PS – Machine Learning for Signal Processing

Matrix Factorizations and Beyond

26 October 2020

Today's lecture

- Latent Variable Models
 - Matrix and tensor decompositions
- Topic models
 - Probabilistic versions and signals
- Convulsive basis models

A non-signal divergence

- Latent Semantic Indexing/Analysis
- Deriving structure from documents
- Identifying topics and document classes

How text works

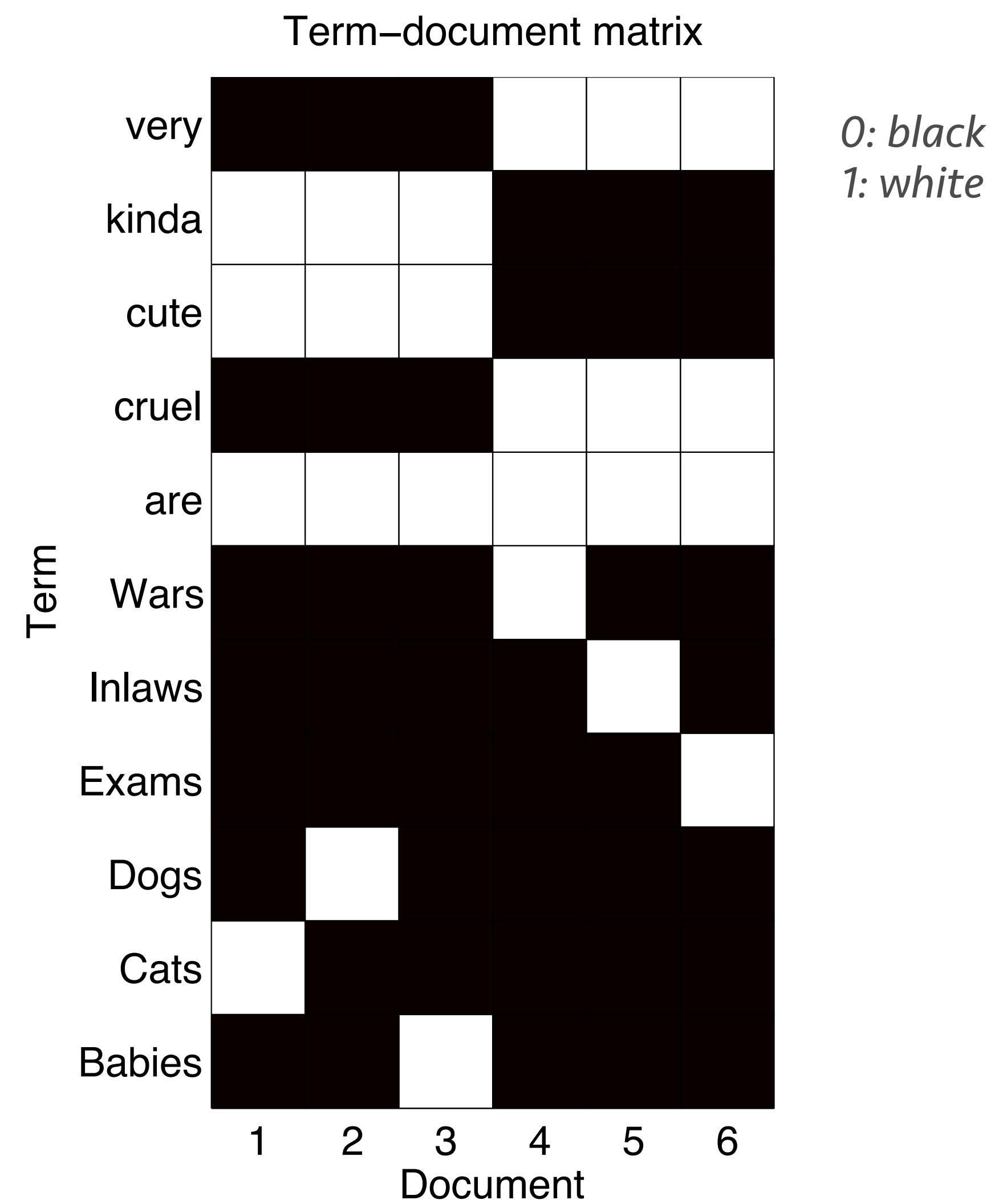
- Large collections of text
 - Varying topics
- Bag of words (BOW) concept
 - Similar documents share common words
- Make statistical tools to extract structure

A simple example

- Consider these *documents*:
 - “Cats are kinda cute”
 - “Wars are very cruel”
 - “Inlaws are very cruel”
 - “Babies are kinda cute”
 - “Exams are very cruel”
 - “Dogs are kinda cute”

Bag of words representation

- Term-document matrix
- Keep track of all the words in all of the documents
- Represent each document as a histogram of words



Finding topics

- With this representation we want to find *topics*
 - i.e. common word collections that imply a subject
- Can you relate this to anything we know?

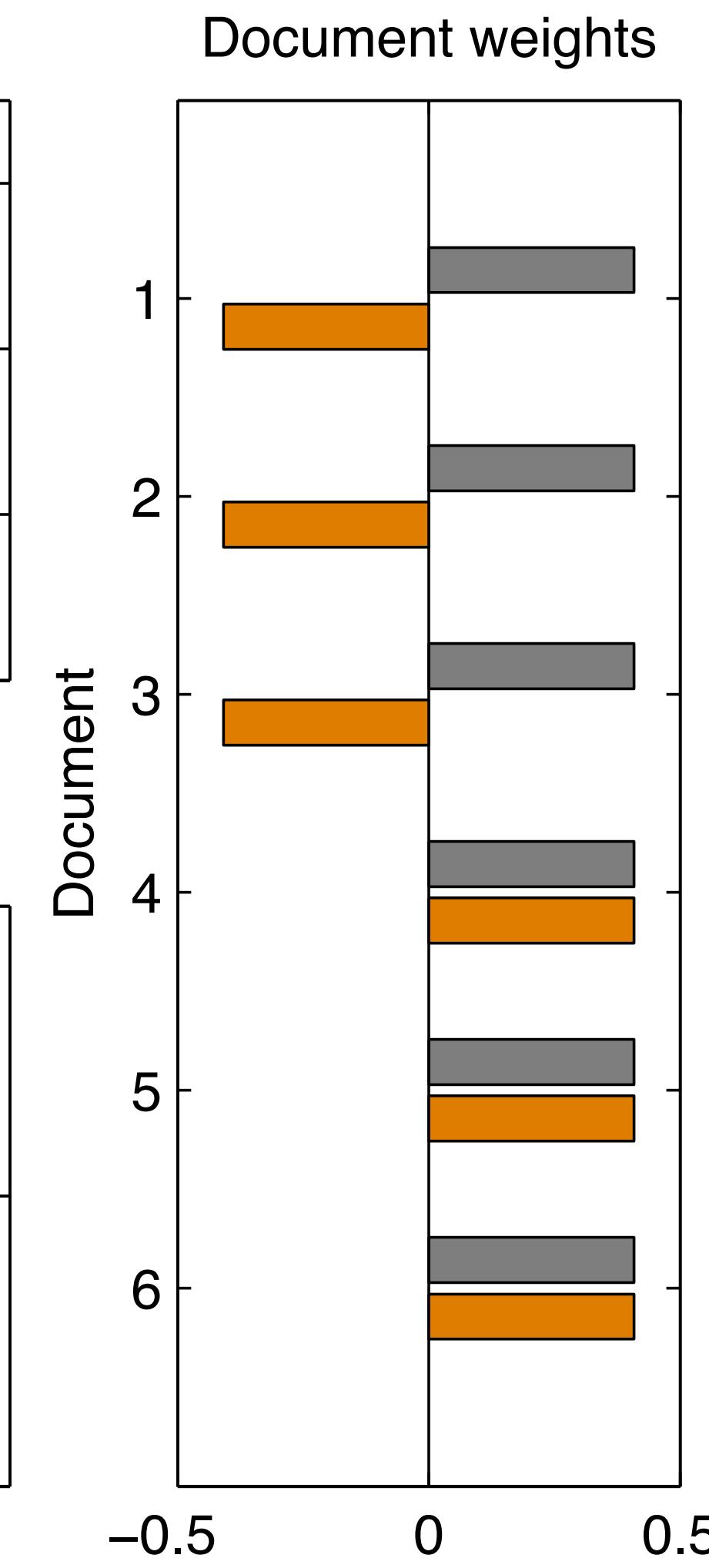
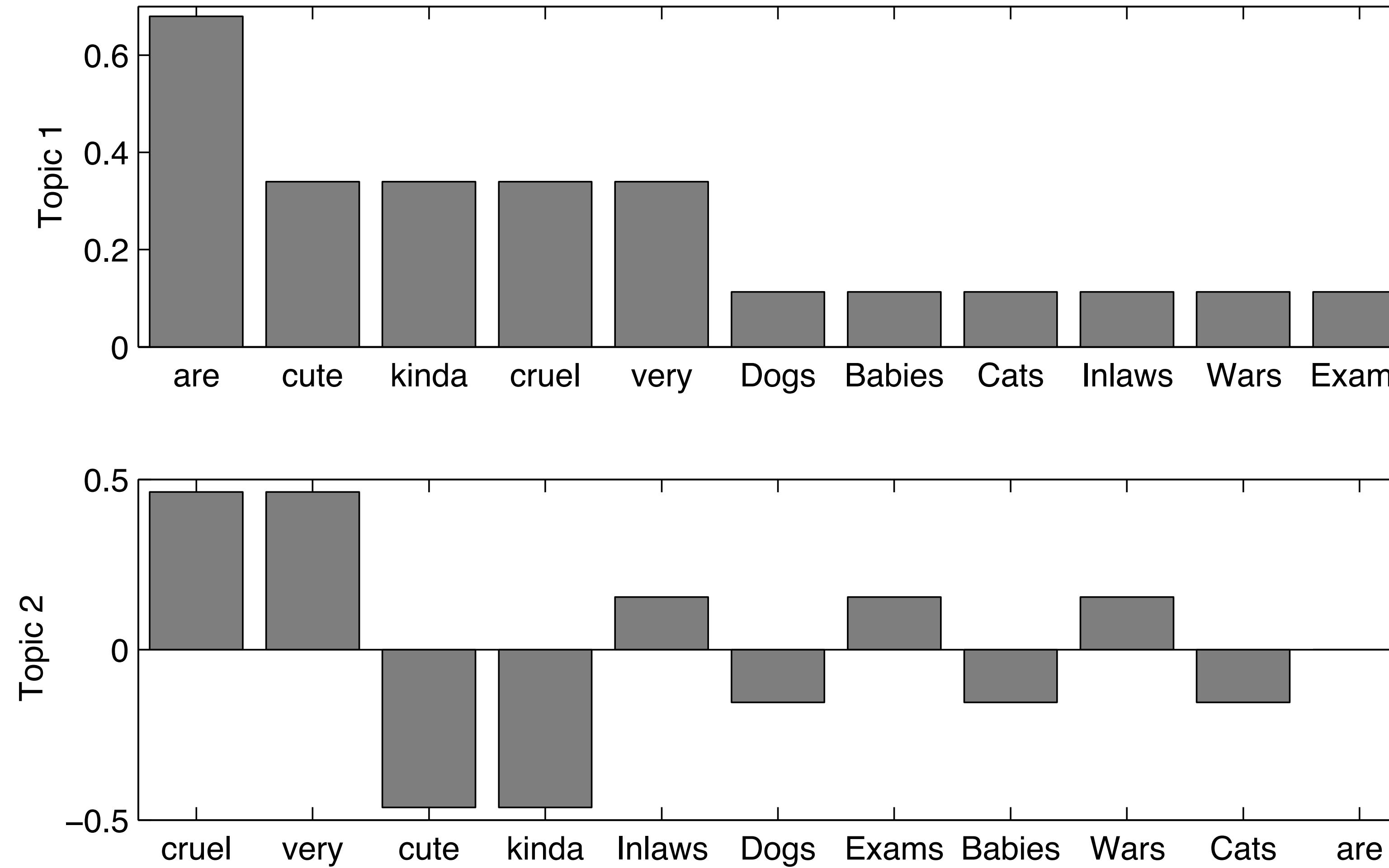
Finding eigen-BOWs

- We perform PCA (once again!)
- We SVD the term/doc matrix \mathbf{C} :

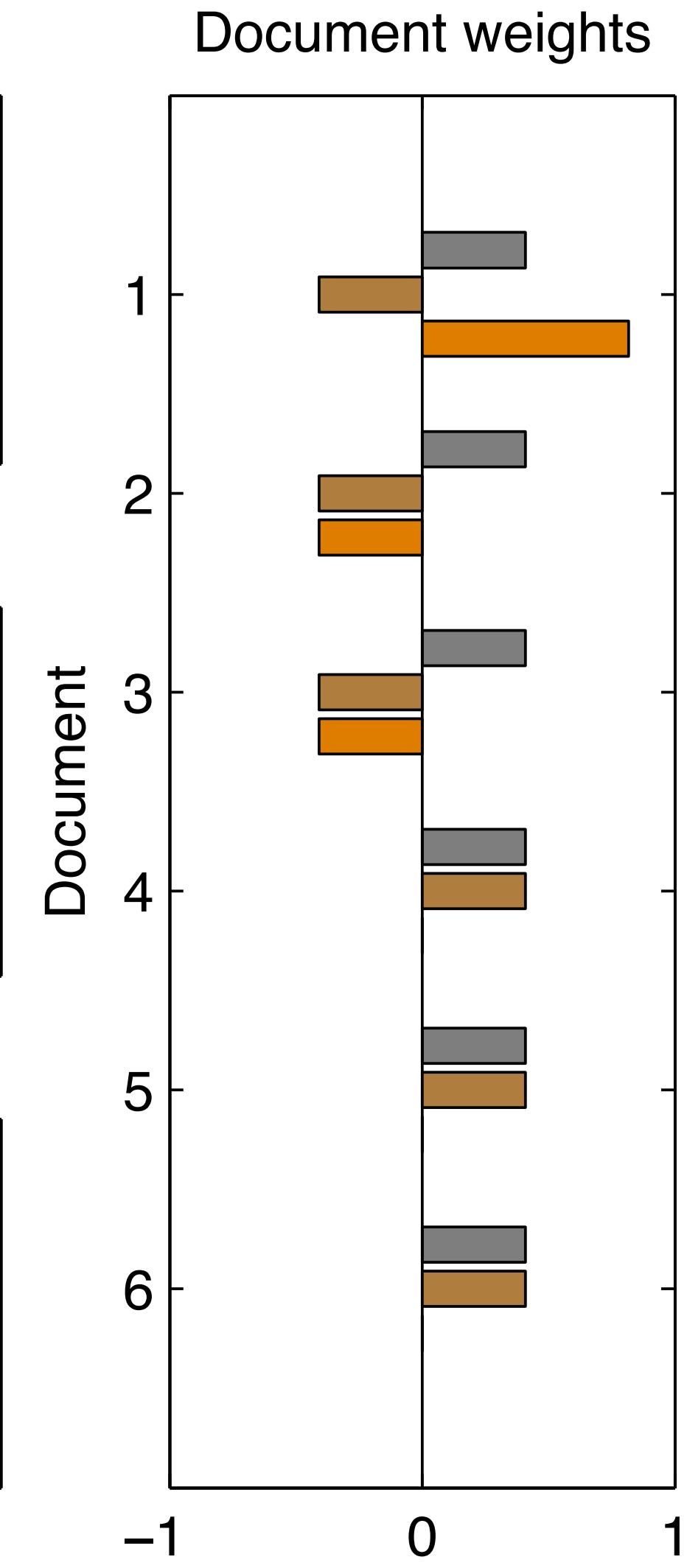
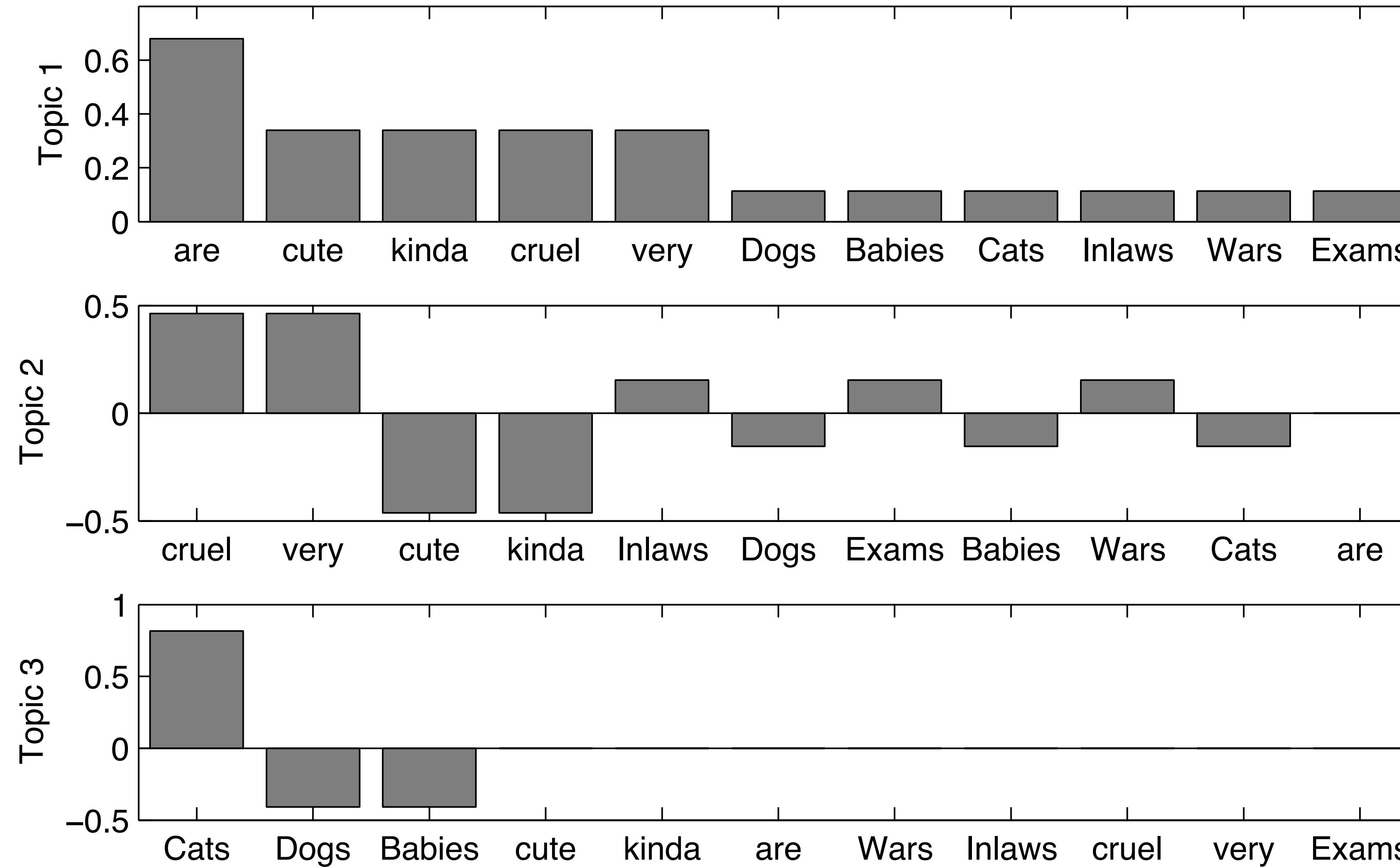
$$\mathbf{C} = \mathbf{U} \cdot \mathbf{S} \cdot \mathbf{V}^T$$

- Columns of \mathbf{U} are the eigen-BOWs
- Rows of \mathbf{V} are the document projections

Getting two eigen-BOWs



Getting three eigen-BOWs



Interpreting the data

- Eigen-BOWs
 - Are templates for “topics”
- Document projection
 - Mix of “topics” that document includes

Things we can do

- Characterize a document's semantic content
 - A mix of topics x, y, z, \dots
- Describe document similarity
 - Compare projections

$$S(\mathbf{v}_i, \mathbf{v}_j) = \frac{\mathbf{v}_i^\top \cdot \mathbf{v}_j}{\|\mathbf{v}_i\| \|\mathbf{v}_j\|}$$

But there's a problem

- There is something fundamentally wrong
- What is it?

It's a somewhat inappropriate model

- Negative words? Really?
- Why should the topics be orthogonal?

Can use NMF, but ...

- Our inputs are histograms, not just non-negative data, they mean something
- We should analyze them appropriately

A probabilistic approach

- Moving from counts to probabilities
- Occurrence matrix seen as a distribution

$$C(\text{word}, \text{doc}) \propto P(\text{word}, \text{doc}) = \frac{\#\text{word in doc}}{\#\text{words in doc}}$$

Probabilistic decomposition

- We define the PLSA model:

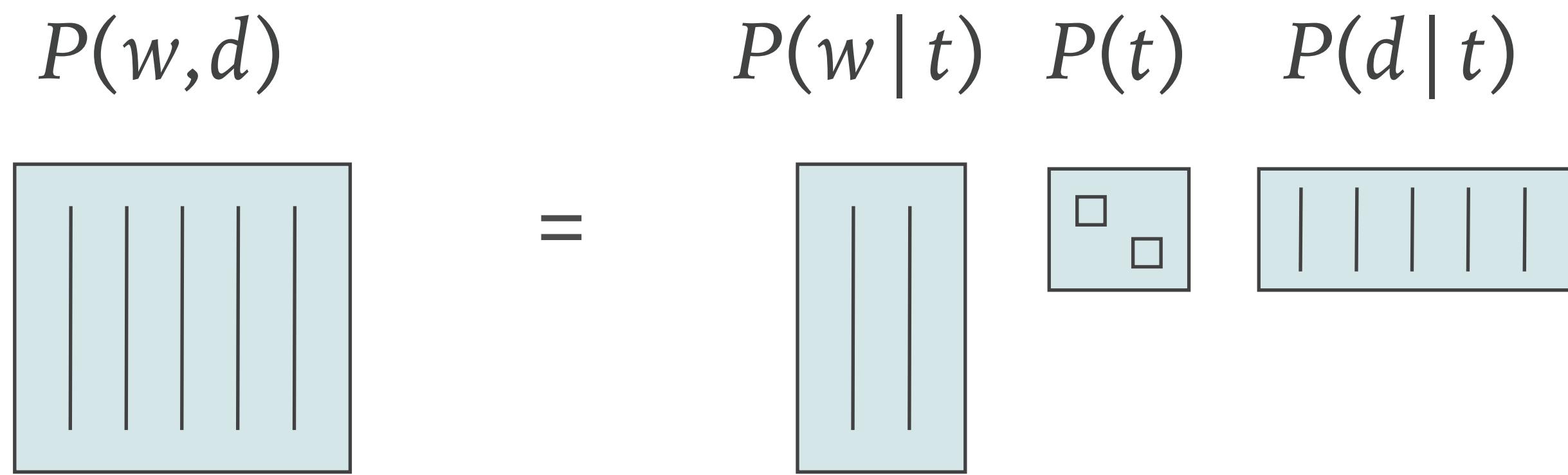
$$P(w, d) = \sum_t P(t)P(d | t)P(w | t)$$

- Which is a probabilistic version of the SVD

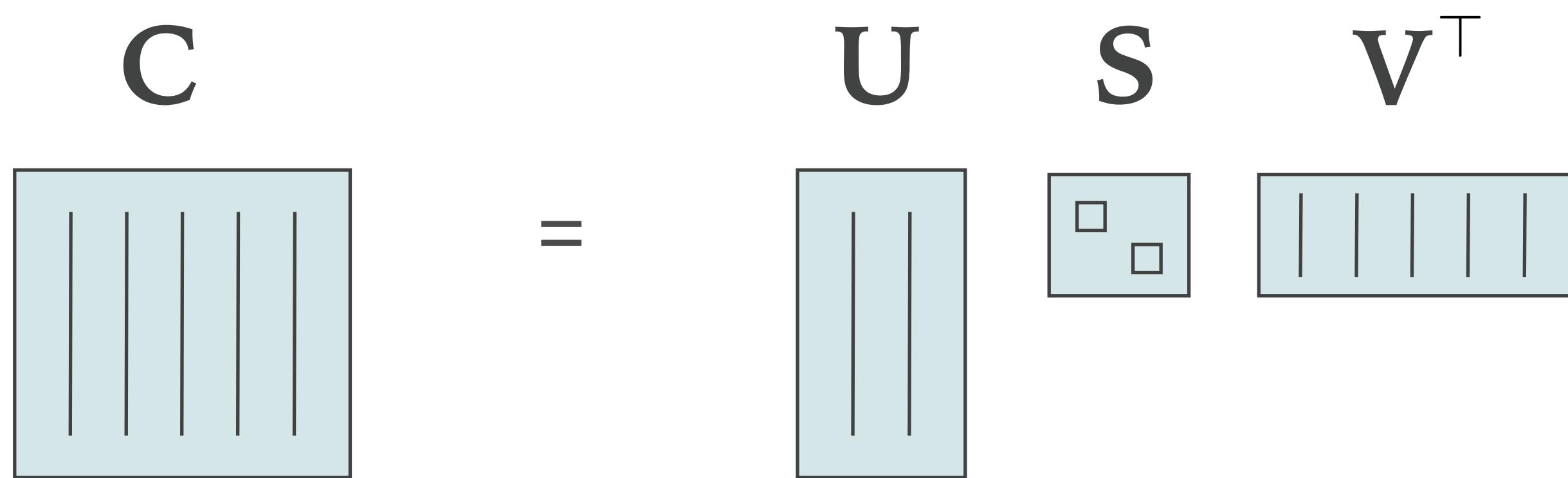
$$\mathbf{C}_{i,j} = \sum_k \mathbf{U}_{i,k} \mathbf{S}_{k,k} \mathbf{V}_{j,k}$$

Model correspondence

- PLSA

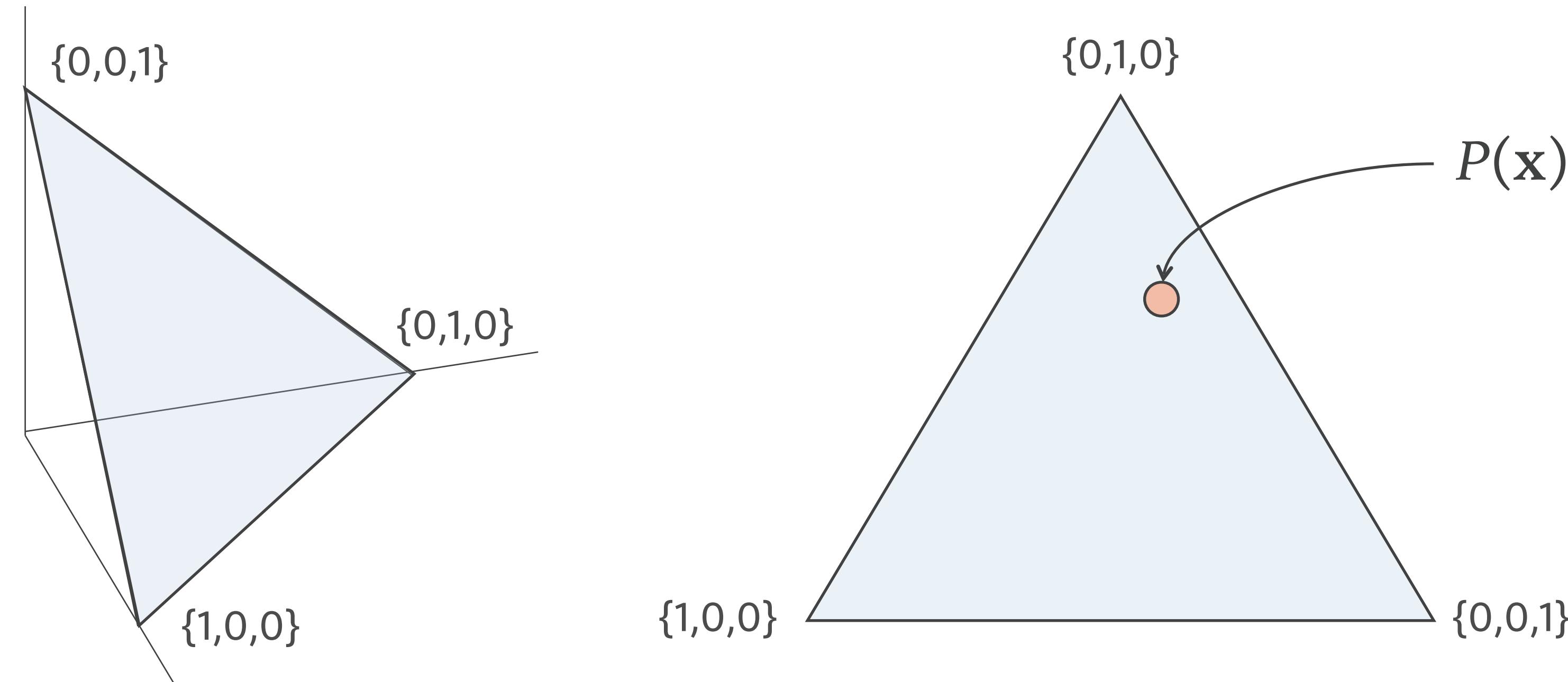


- SVD



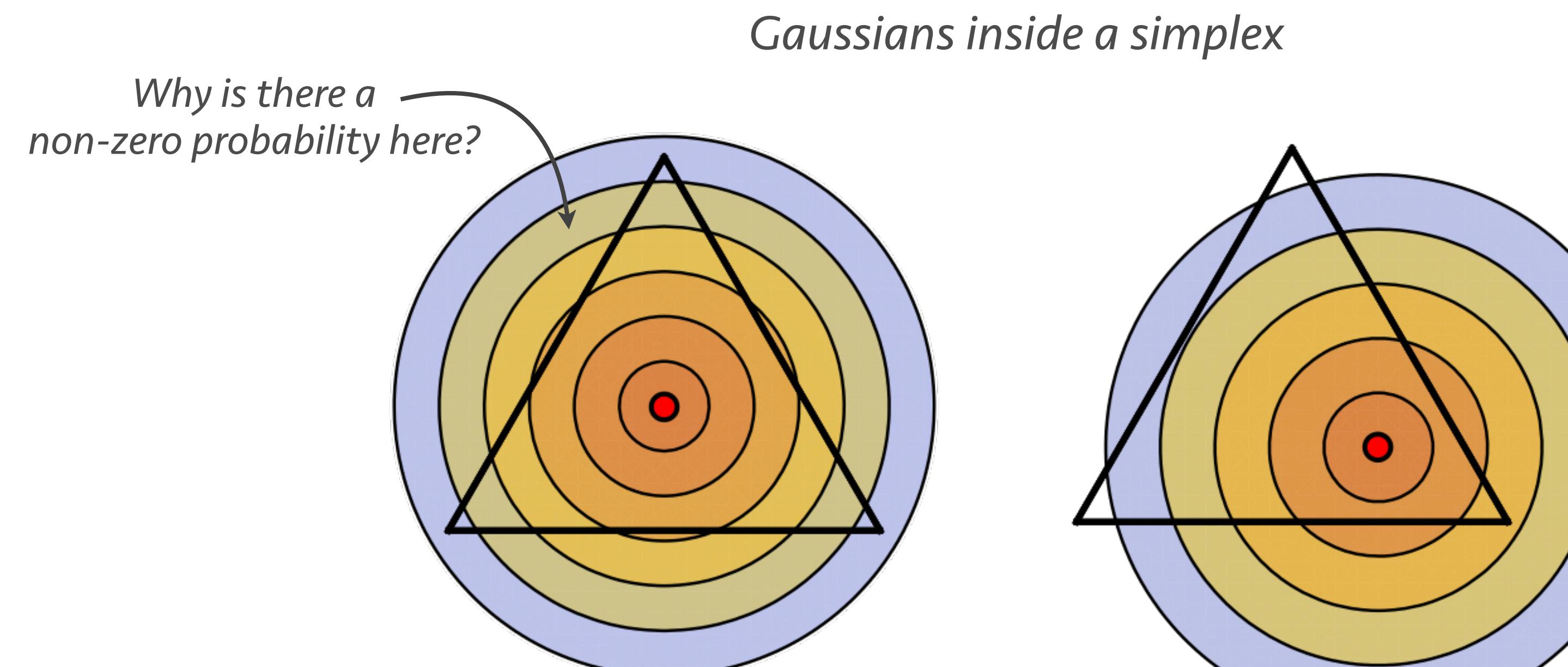
Being in a new space

- The quantities involved sum to 1 (they are probabilities)
 - They therefore live inside a simplex



Why the SVD is bad here

- The SVD makes Gaussian assumptions
 - Euclidean distance, orthogonality, etc ...
- This is a poor choice in this space

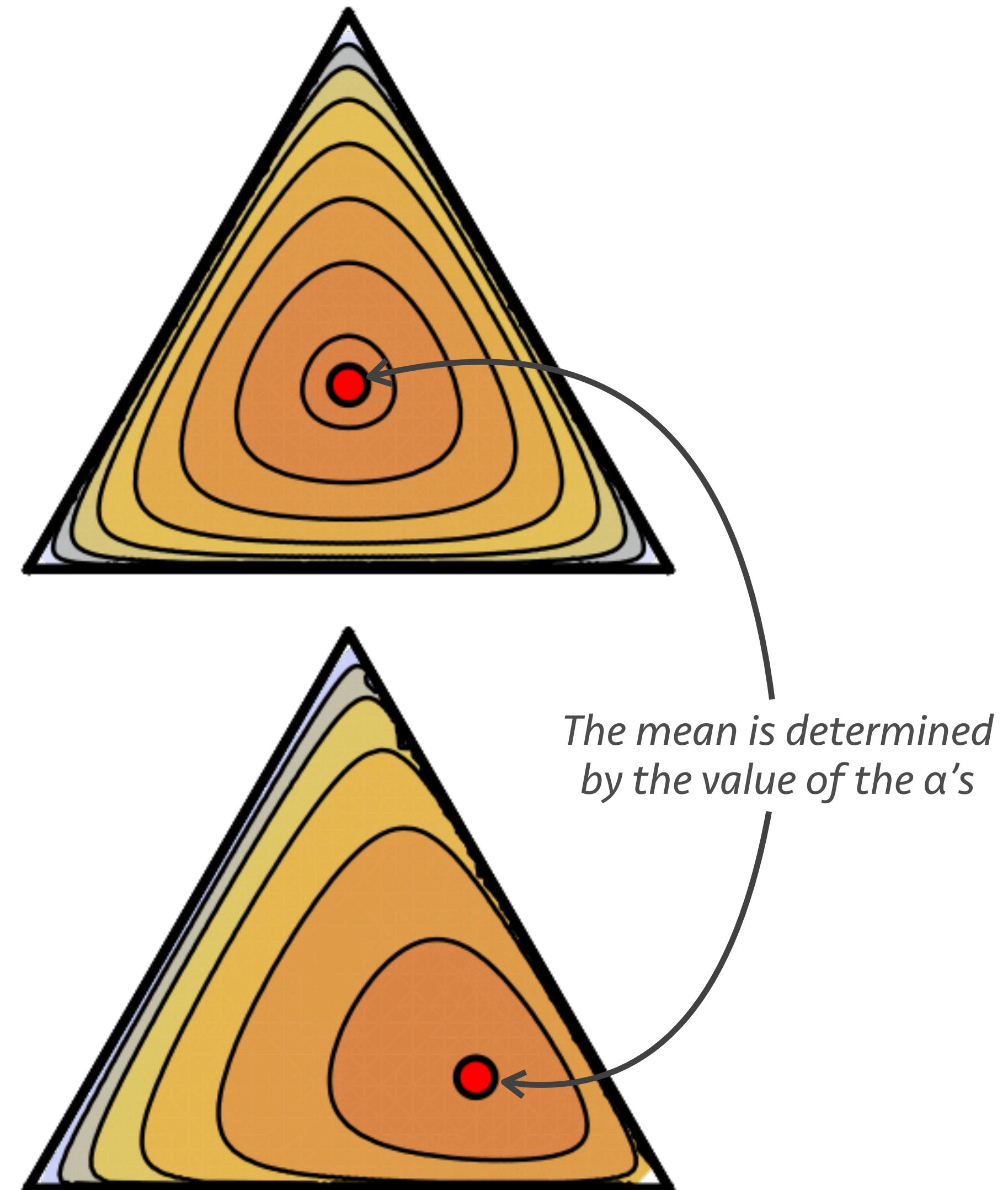


A proper distribution on the simplex

- The Dirichlet distribution

$$\mathcal{D}(x; \alpha) = \frac{\Gamma\left(\sum \alpha_i\right)}{\prod \Gamma(\alpha_i)} \prod x_i^{\alpha_i - 1}$$

- Respects the constraints of this new space



Implied cost function

- The Gaussian assumption means that we optimize the Euclidean distance

$$D(\mathbf{x}, \mathbf{y}) \propto \log \mathcal{N}(\mathbf{x}; \mu = \mathbf{y}) \propto \sum (x_i - y_i)^2$$

- The Dirichlet assumption results in optimizing the *cross-entropy* instead:

$$D(\mathbf{x}, \mathbf{y}) \propto \log \mathcal{D}(\mathbf{x}; \alpha - 1 = \mathbf{y}) \propto \sum x_i \log(y_i)$$

Relation to NMF

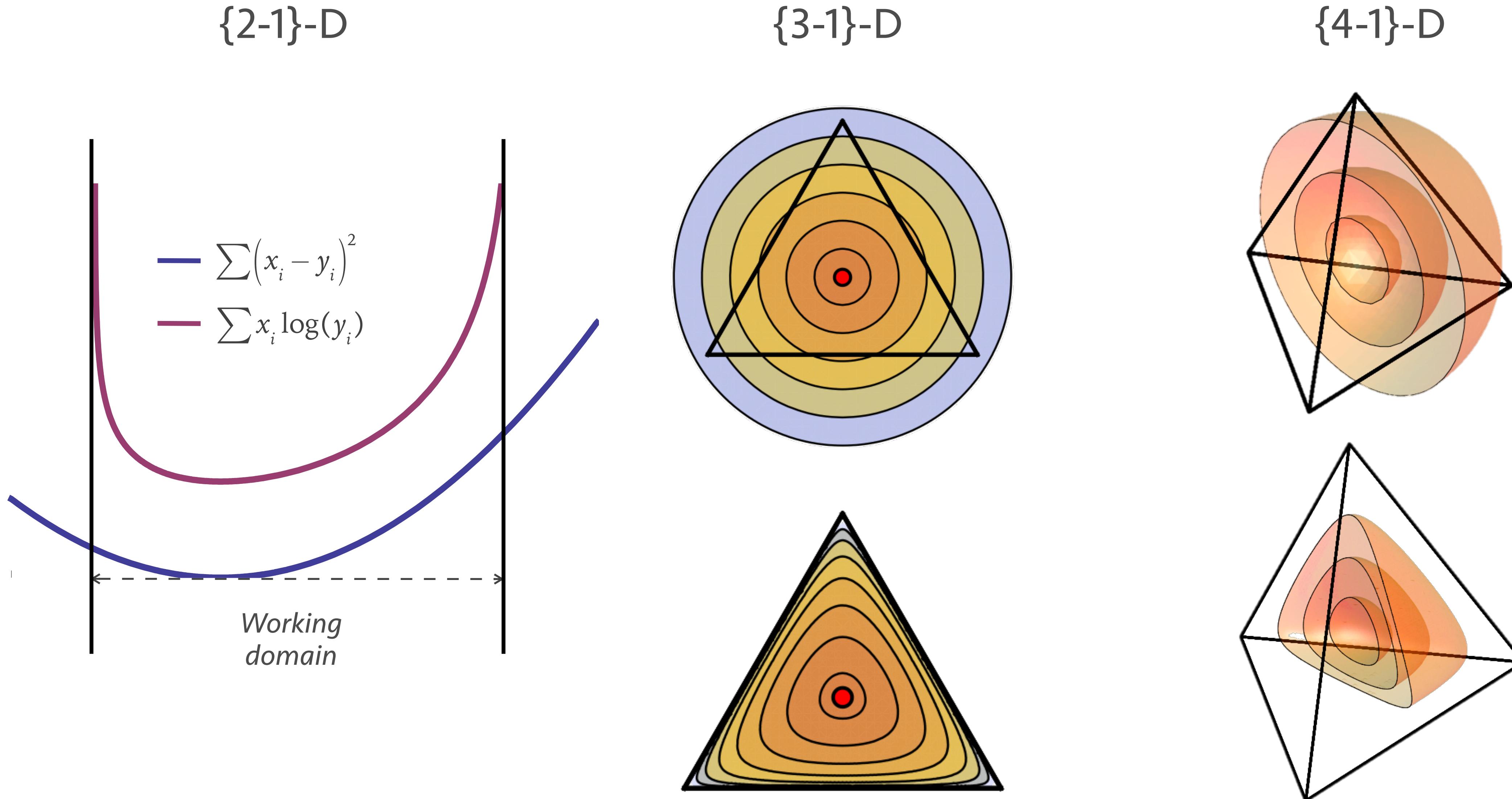
- In NMF we optimize a form of the Kullback-Leibler divergence

$$KL(\mathbf{x}, \mathbf{y}) = \sum x_i \log \frac{x_i}{y_i}$$

- The cross entropy is just an offset of that

$$C(\mathbf{x}, \mathbf{y}) = KL(\mathbf{x}, \mathbf{y}) - H(\mathbf{x})$$

Comparing the costs



Estimating the parameters

- We can use Expectation Maximization
 - This is a mixture model (mixing topics!)
- E-step (find each topic's contribution)

$$P(t | d, w) = \frac{P(t)P(d | t)P(w | t)}{\sum_{t'} P(t')P(d | t')|P(w | t')}}$$

Estimating the parameters

- M-step (use E-step weights to re-estimate)

$$P(w | t) \propto \sum_d P(w, d) P(t | d, w)$$

$$P(d | t) \propto \sum_w P(w, d) P(t | d, w)$$

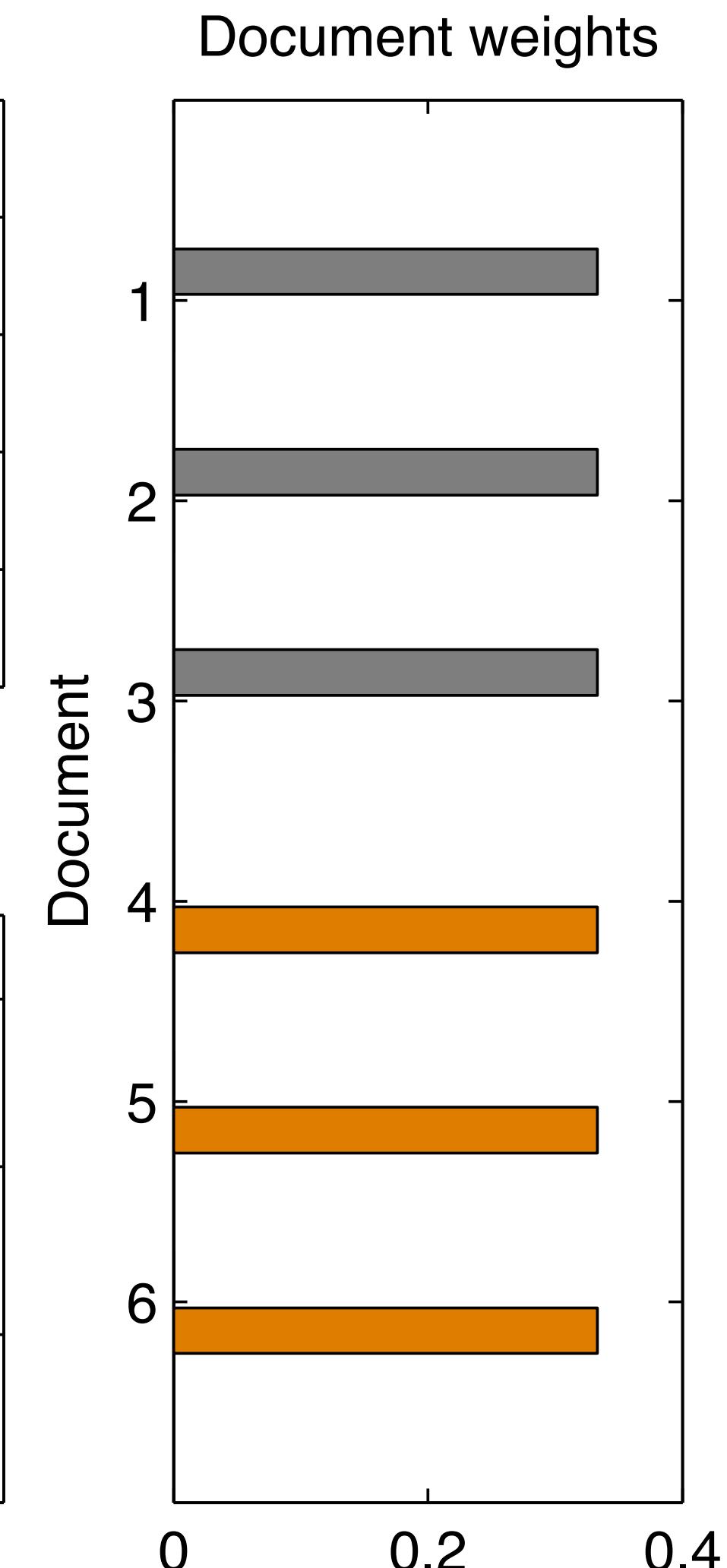
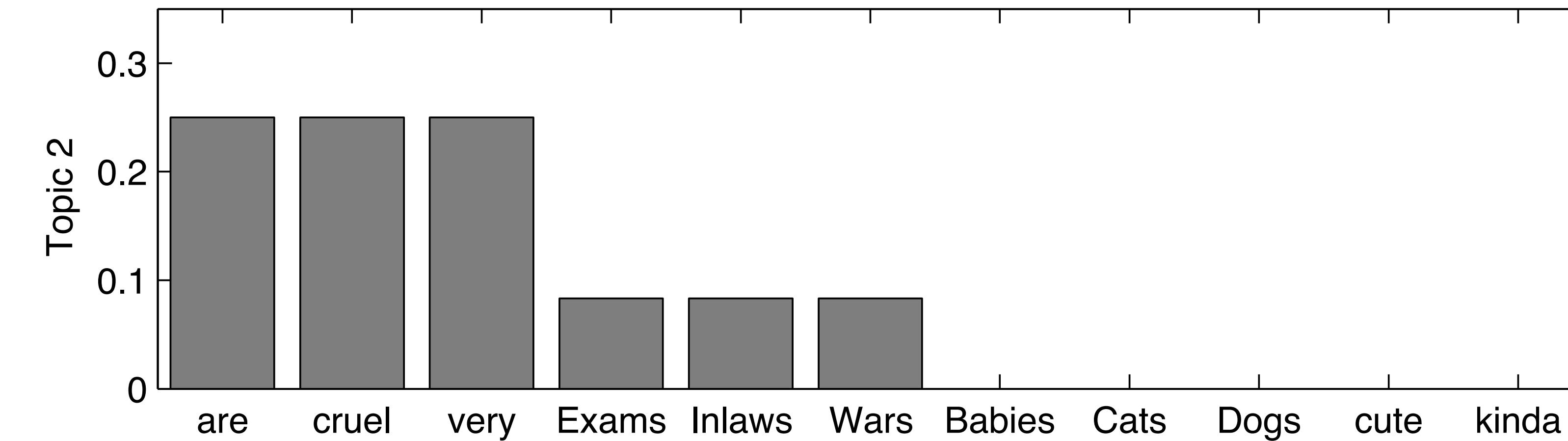
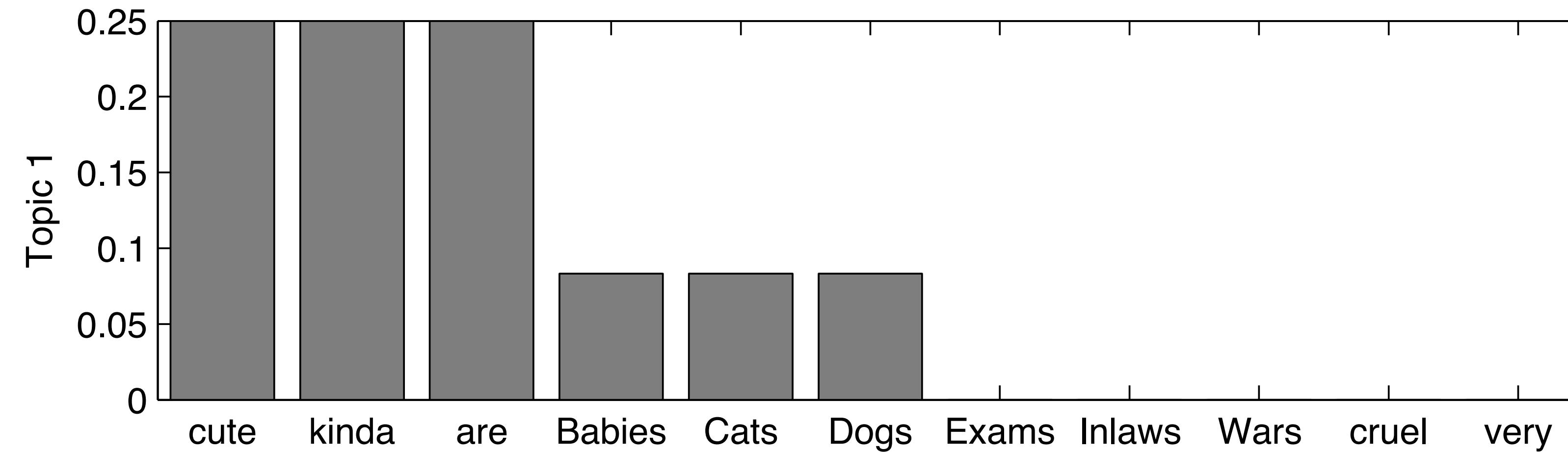
$$P(t) \propto \sum_{d, w} P(w, d) P(t | d, w)$$

- Which is pretty much the same as the NMF update

Back to text

- We now have a more satisfying representation of the documents/topics
 - $P(d|t)$ is the link between document and topic
 - $P(w|t)$ is the link between word and topic
 - $P(t)$ is the prior of a topic
- More interpretable quantities

On our mini document data



Much more satisfying results

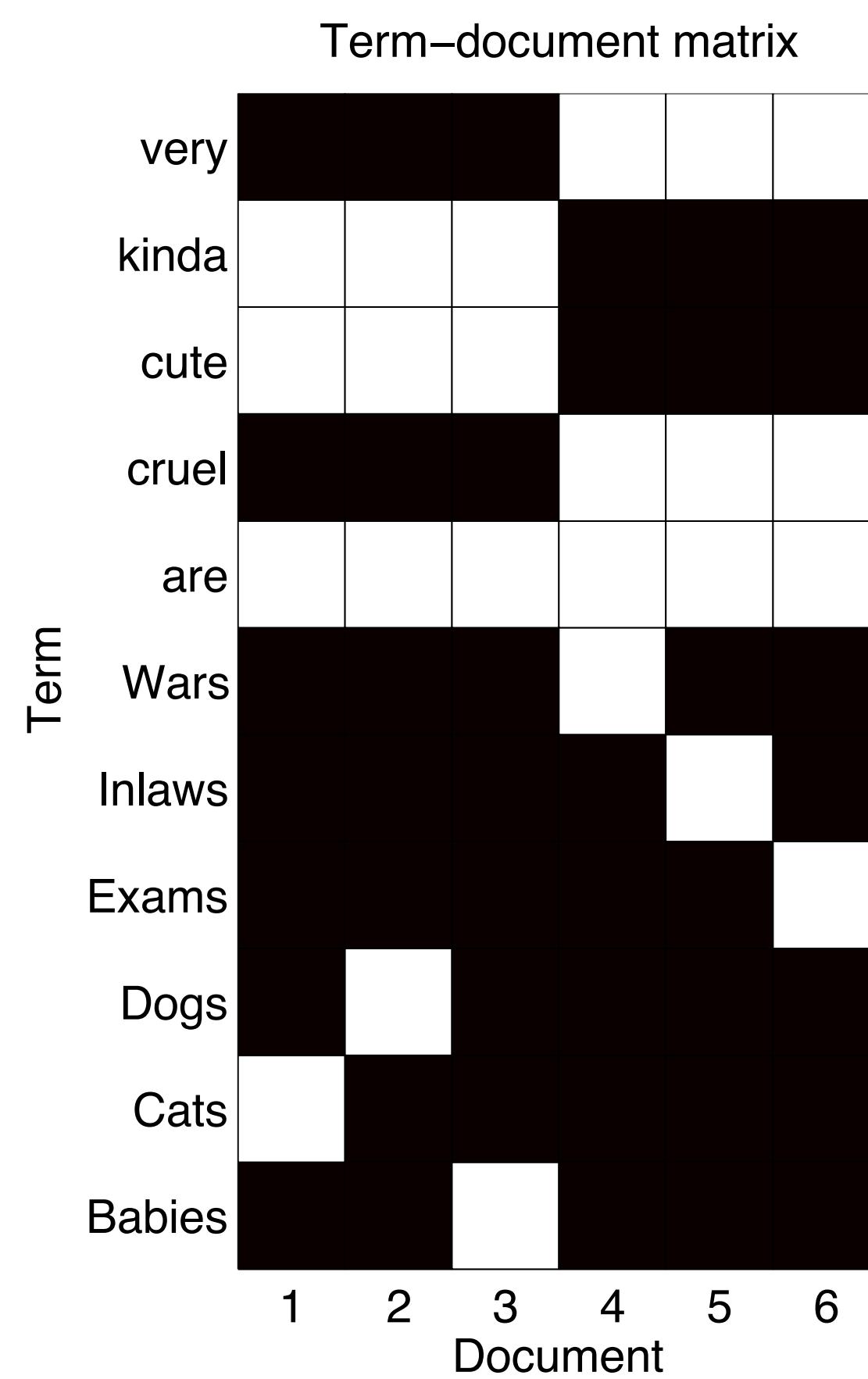
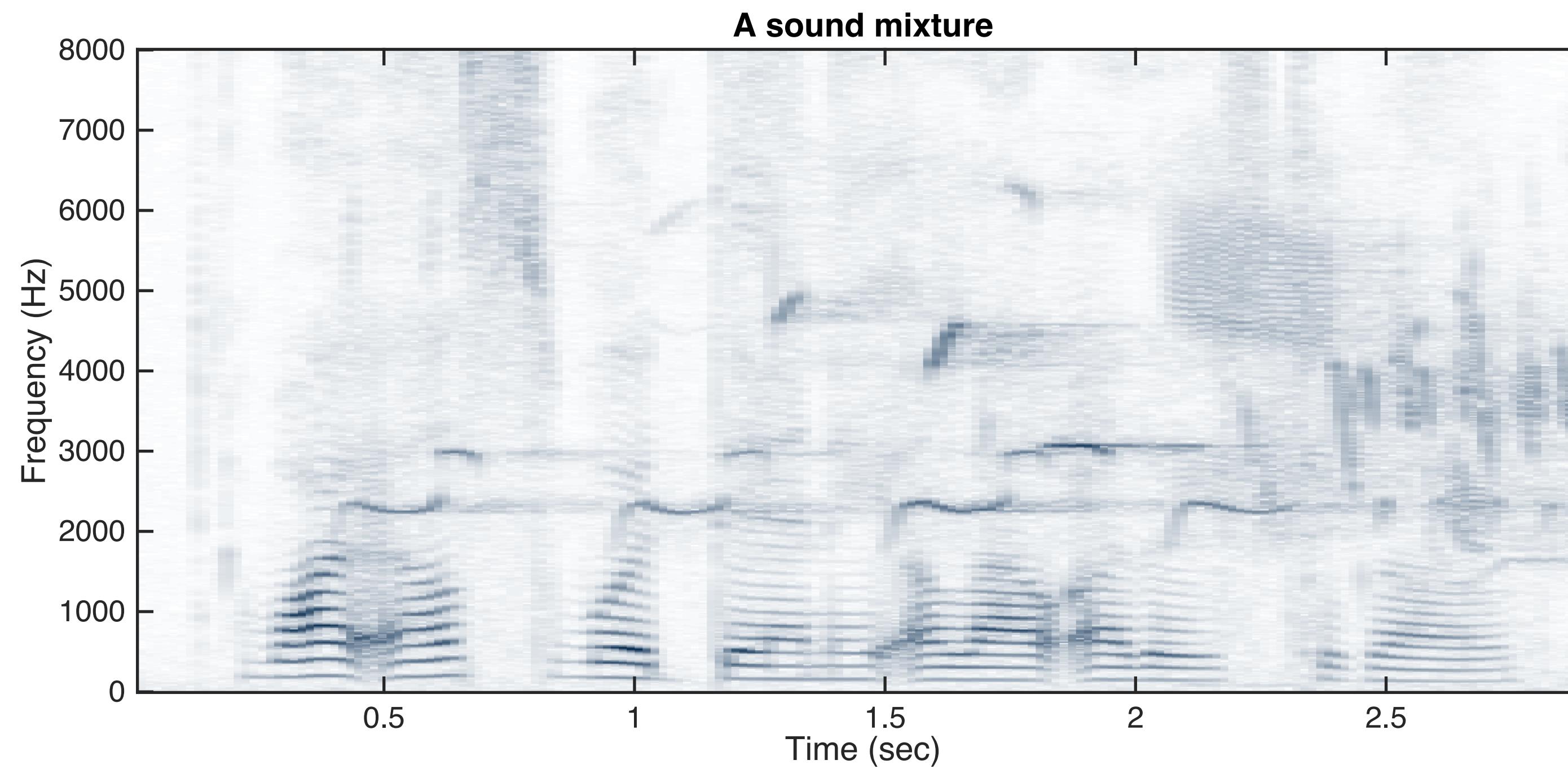
- Probabilistic “eigen-BOWs”
 - No orthogonality constraint
 - Sensible topics that can share words
 - Non-negative words probabilities in topics
- Document “weights”
 - Probabilistic mix of topics
- This is known as the PLSI/PLSA model

Back to signals

- Taking that lesson to the signal domain
- Analysis of any non-negative data
 - Counts, energies, variances, probabilities

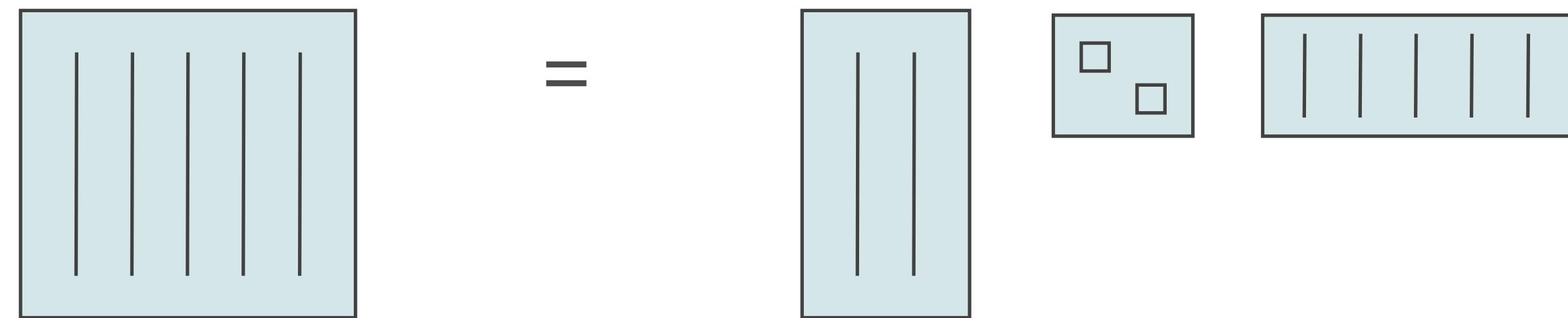
Incorporating time

- The PLSA doesn't use time information
 - But we do in signals!

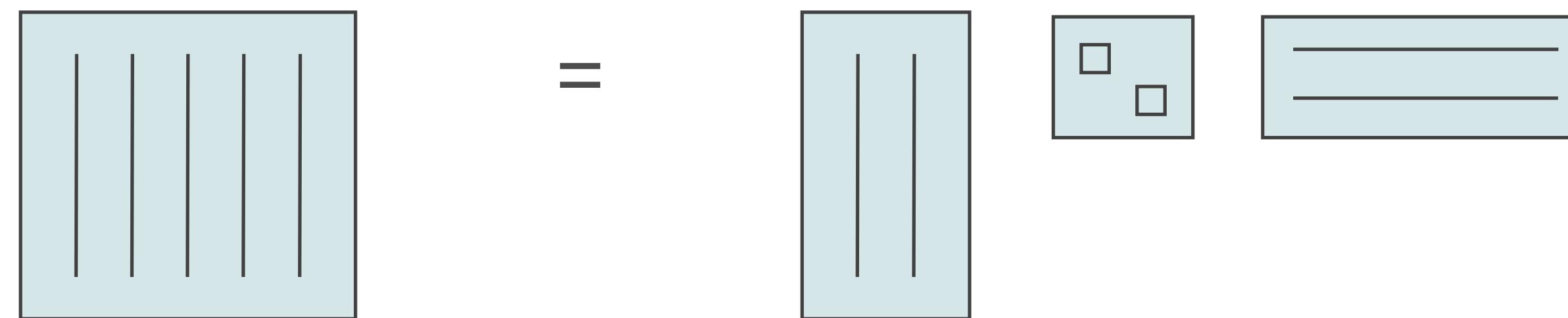


A variation

- PLSA model treats x-axis column-wise:



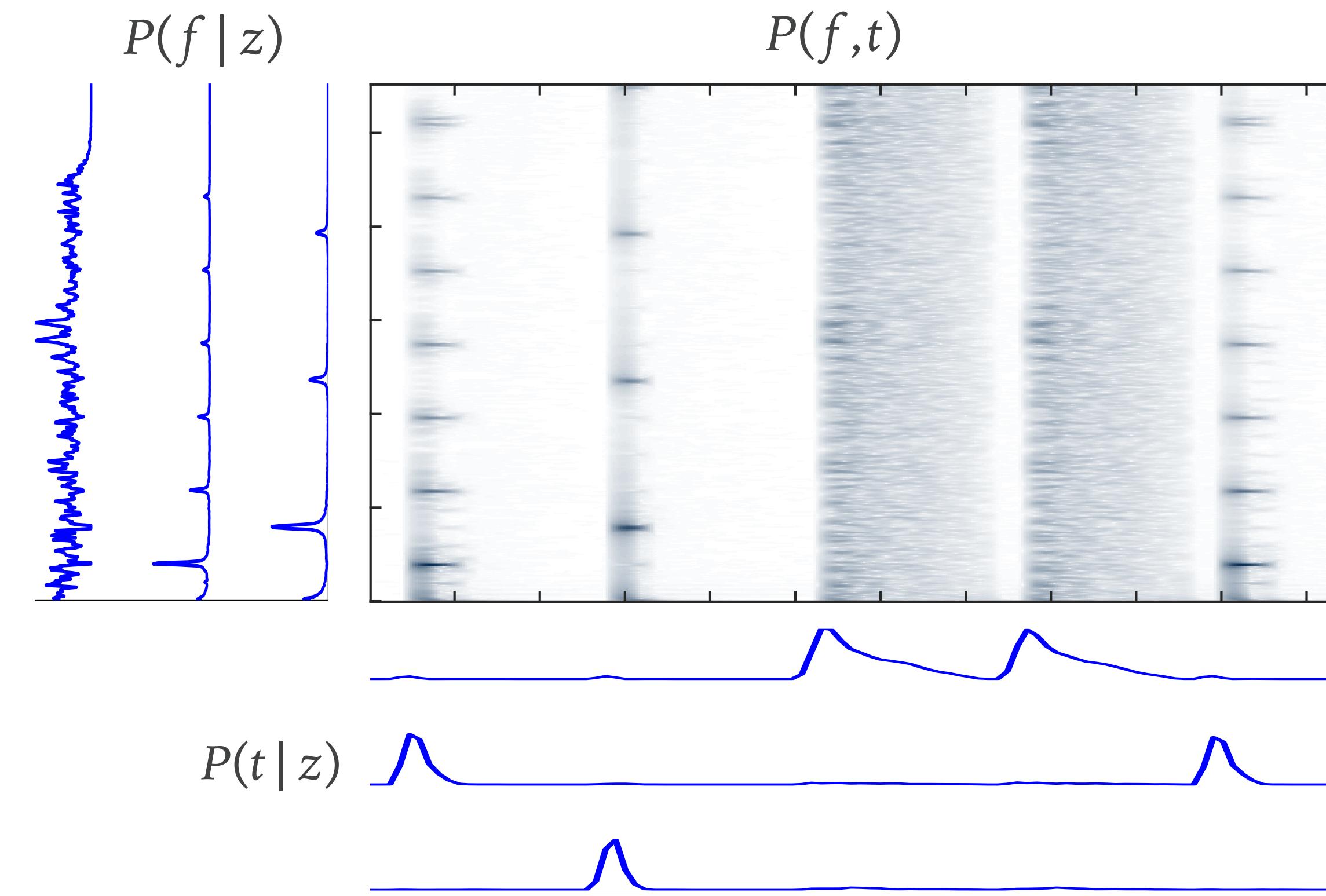
- We recast it as the PLCA model ("C" for component):



- Just a change in the normalization of last term

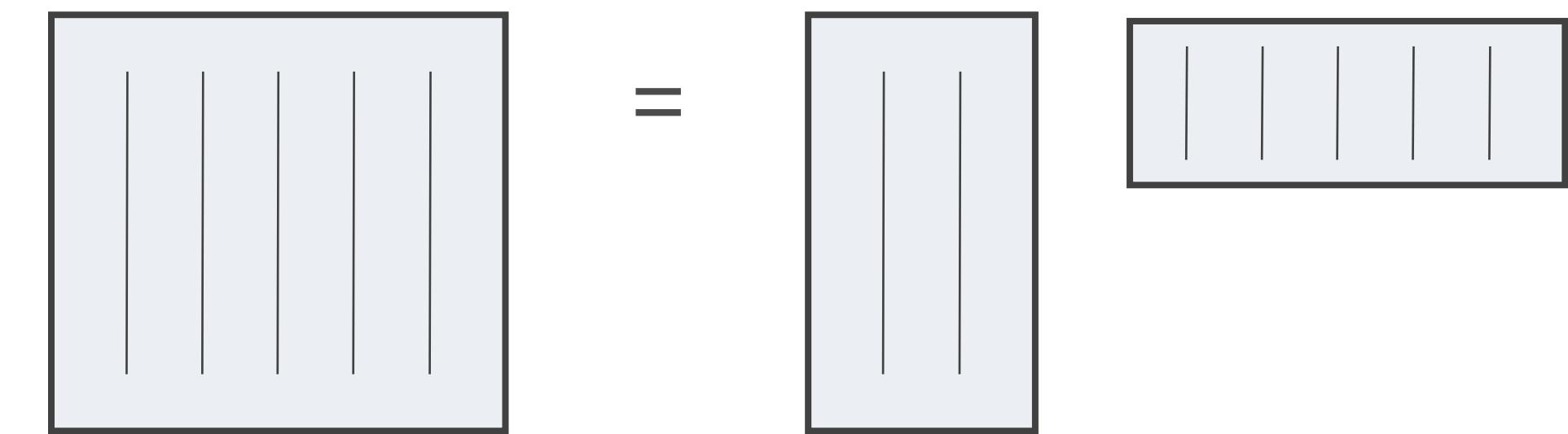
Now we can do NMF-like things

- Like spectrogram factorizations
 - And components are now interpretable

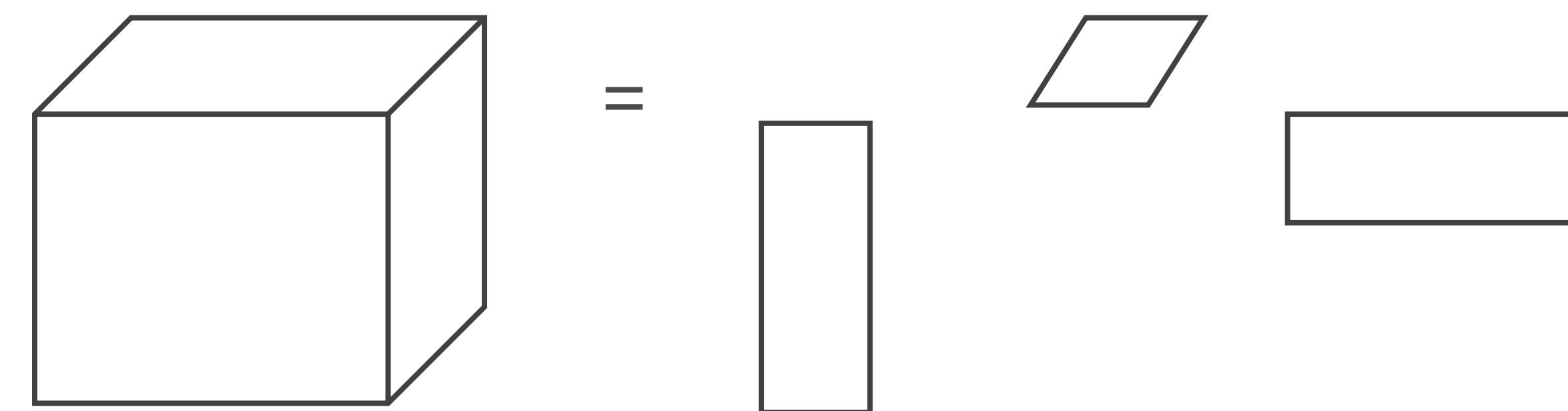


Tensor decompositions

- SVD/PCA model was for matrices

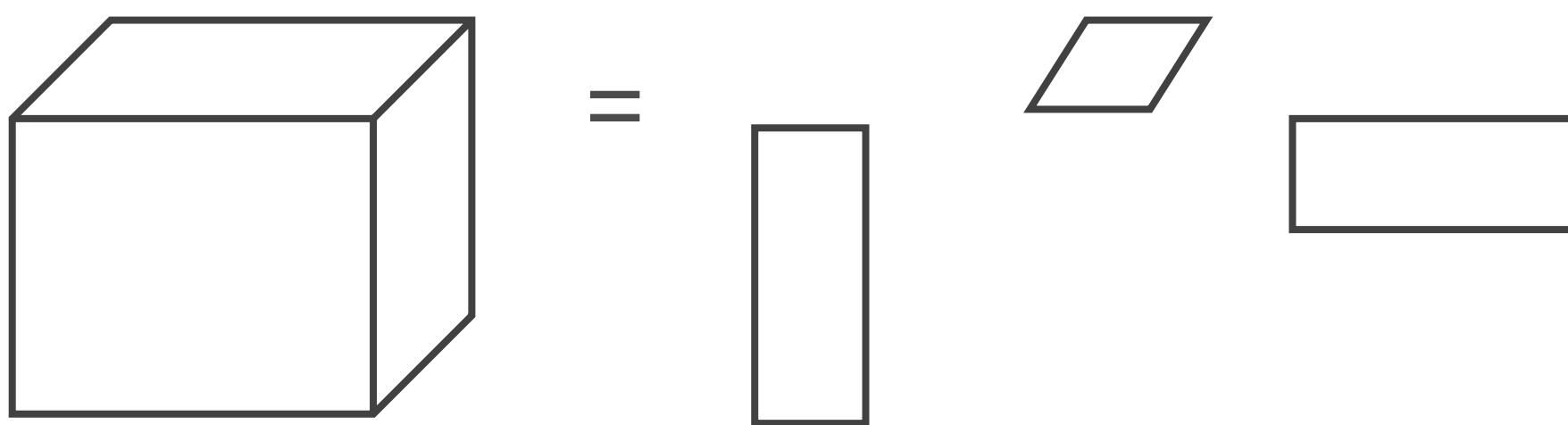


- Similar models exist for tensors



PARAFAC model

- Factor in terms of outer matrix products



- Easy notation: $x_{i,j,k} = \sum_l a_{i,l} b_{j,l} c_{k,l}$
- Matrix notation: $\mathbf{X}^{(I \times JK)} = \sum_l \mathbf{a}_l \left(\mathbf{b}_l^\top \otimes \mathbf{c}_l^\top \right)$
 $= \mathbf{A} \cdot (\mathbf{B}^\top \otimes \mathbf{C}^\top)$

Solving for PARAFAC

- Treat as three problems: $\mathbf{X}_{::,k} = \mathbf{A} \cdot \text{diag}(\mathbf{c}_k) \cdot \mathbf{B}^\top$
 $\mathbf{X}_{i,:,:} = \mathbf{B} \cdot \text{diag}(\mathbf{a}_i) \cdot \mathbf{C}^\top$
 $\mathbf{X}_{:,j,:} = \mathbf{C} \cdot \text{diag}(\mathbf{b}_j) \cdot \mathbf{A}^\top$
- And solve for them by alternating

$$\hat{\mathbf{A}} = \left(\sum_k \mathbf{X}_{::,k} \cdot \mathbf{B} \cdot \text{diag}(\mathbf{c}_k) \right) \cdot \left((\mathbf{B}^\top \cdot \mathbf{B}^\top) \odot (\mathbf{C}^\top \cdot \mathbf{C}^\top) \right)^{-1}$$
$$\hat{\mathbf{B}} = \left(\sum_i \mathbf{X}_{i,:,:} \cdot \mathbf{C} \cdot \text{diag}(\hat{\mathbf{a}}_i) \right) \cdot \left((\mathbf{C}^\top \cdot \mathbf{C}^\top) \odot (\hat{\mathbf{A}}^\top \cdot \hat{\mathbf{A}}^\top) \right)^{-1}$$
$$\hat{\mathbf{C}} = \left(\sum_j \mathbf{X}_{:,j,:} \cdot \hat{\mathbf{A}} \cdot \text{diag}(\hat{\mathbf{b}}_j) \right) \cdot \left((\hat{\mathbf{A}}^\top \cdot \hat{\mathbf{A}}^\top) \odot (\hat{\mathbf{B}}^\top \cdot \hat{\mathbf{B}}^\top) \right)^{-1}$$

Variations

- Extension to N -way tensors
 - Corresponding to N factors
- Non-negative PARAFAC
 - Same as before, factors are non-negative
- Tucker decompositions
 - More involved, mixes components between dimensions

Probabilistic version

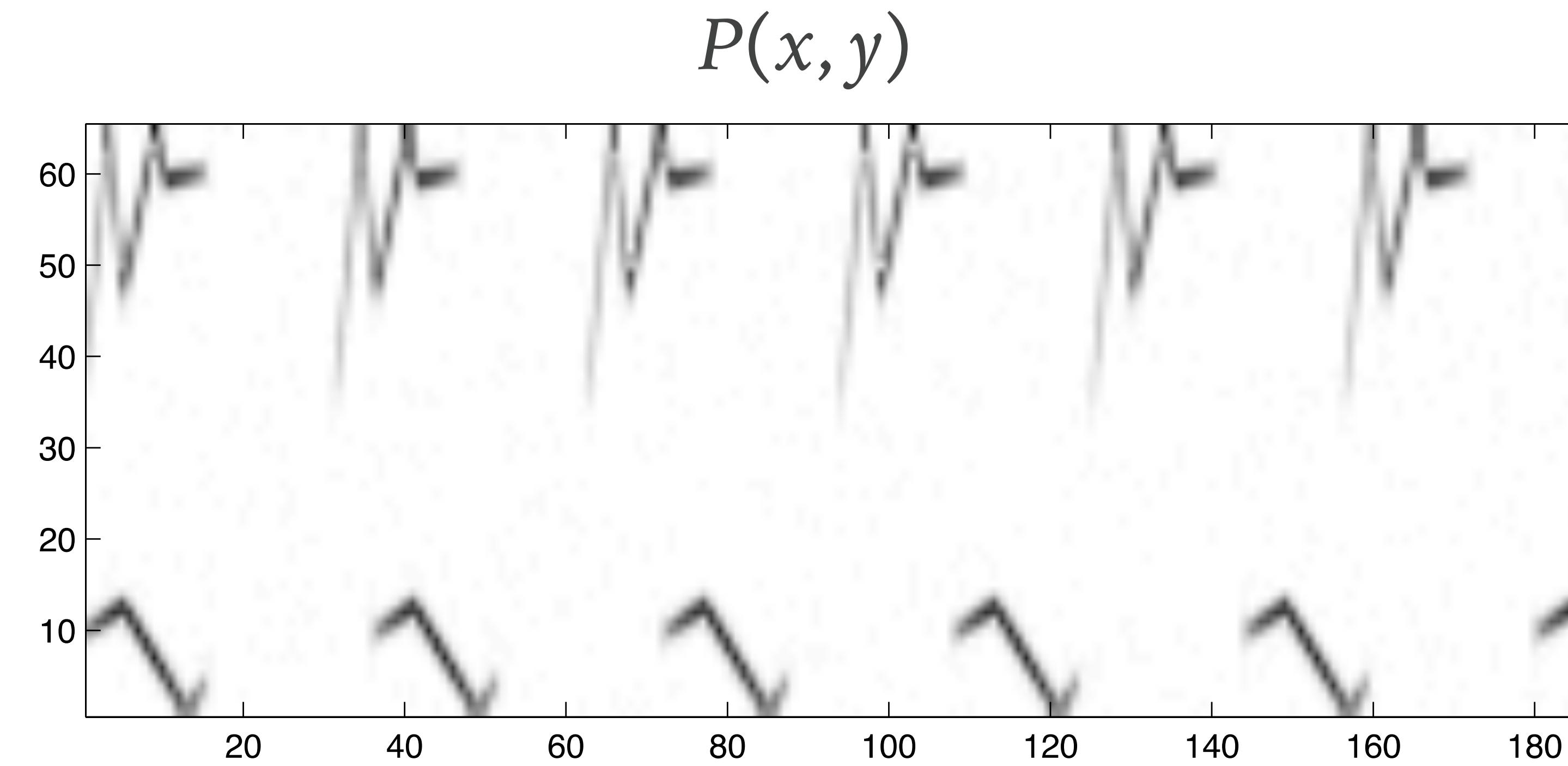
- We can recast this as the PLCA model

$$P(\mathbf{x}) = \sum_z P(z) \prod_j P(x_j | z)$$

- \mathbf{x} can be of arbitrary dimensions
 - And seen as a non-negative tensor
- Same estimation equations as with 2-D case
 - So we don't have to worry about more dimensions

Adding one more dimension in the mix

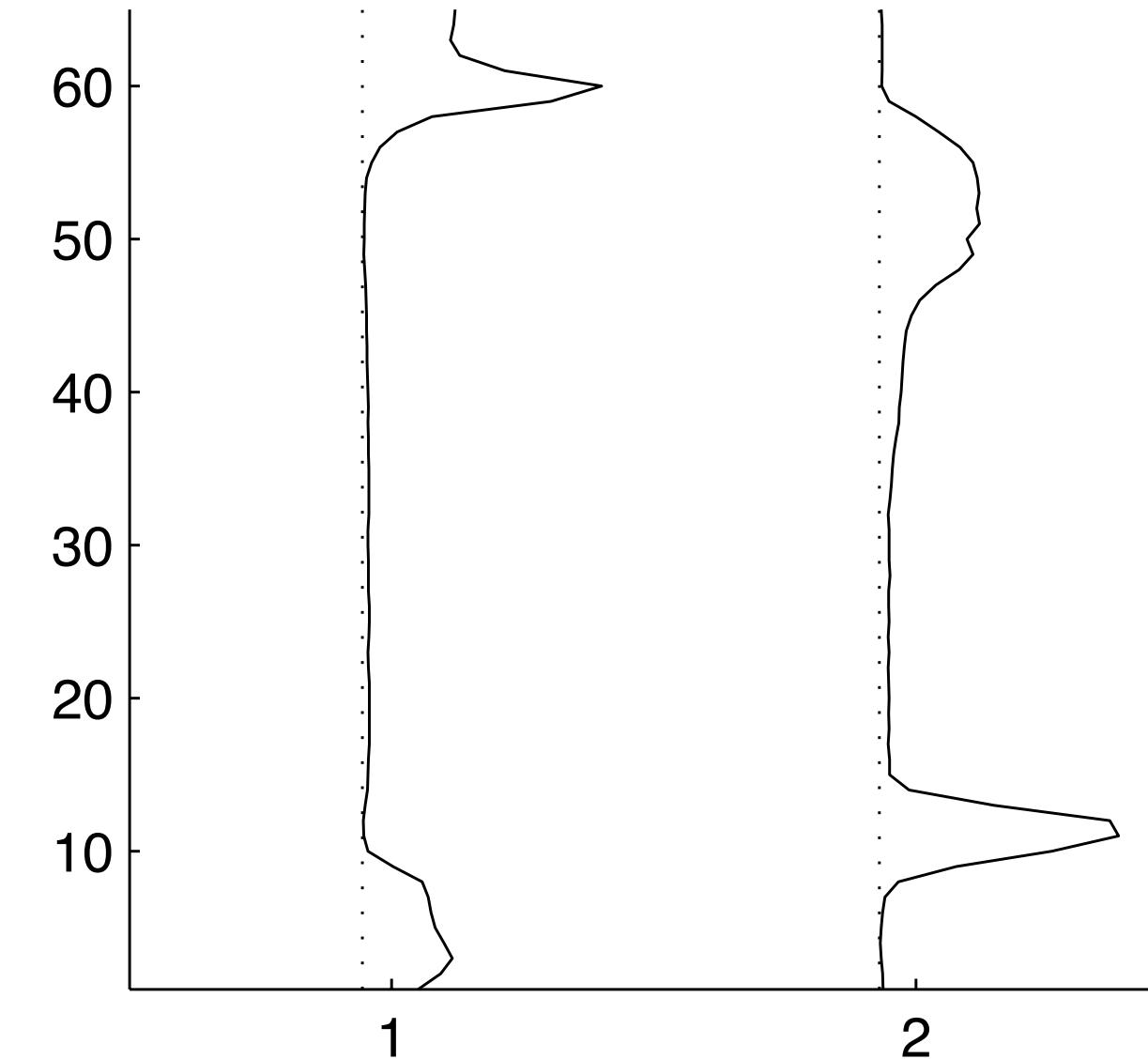
- Example input with temporal structure
 - What would PLCA give us?



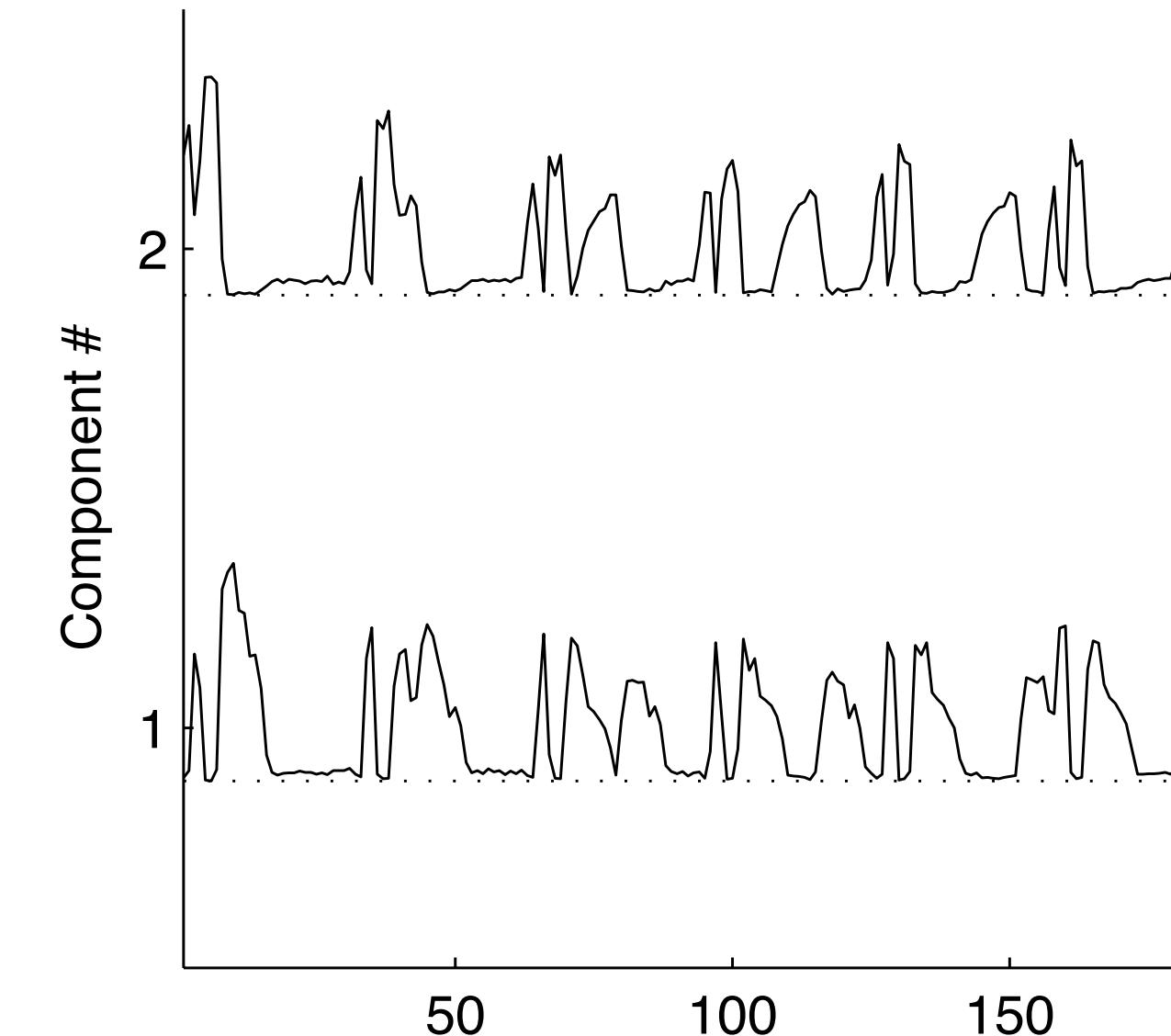
PLCA on temporal example

- Temporal structure is obscured
 - Components time-average the “objects”

$P(y|z)$



$P(x|z)$



Rethinking PLCA

- Convulsive component models
 - Substitute the multiplications with convolutions

$$P(x, y) = \sum_z P(z) \sum_{\tau} P(x, \tau | z) P(y - \tau | z)$$

- Each component is now 2-d
 - x-axis and y-axis presence
 - We shift these around along the x-axis

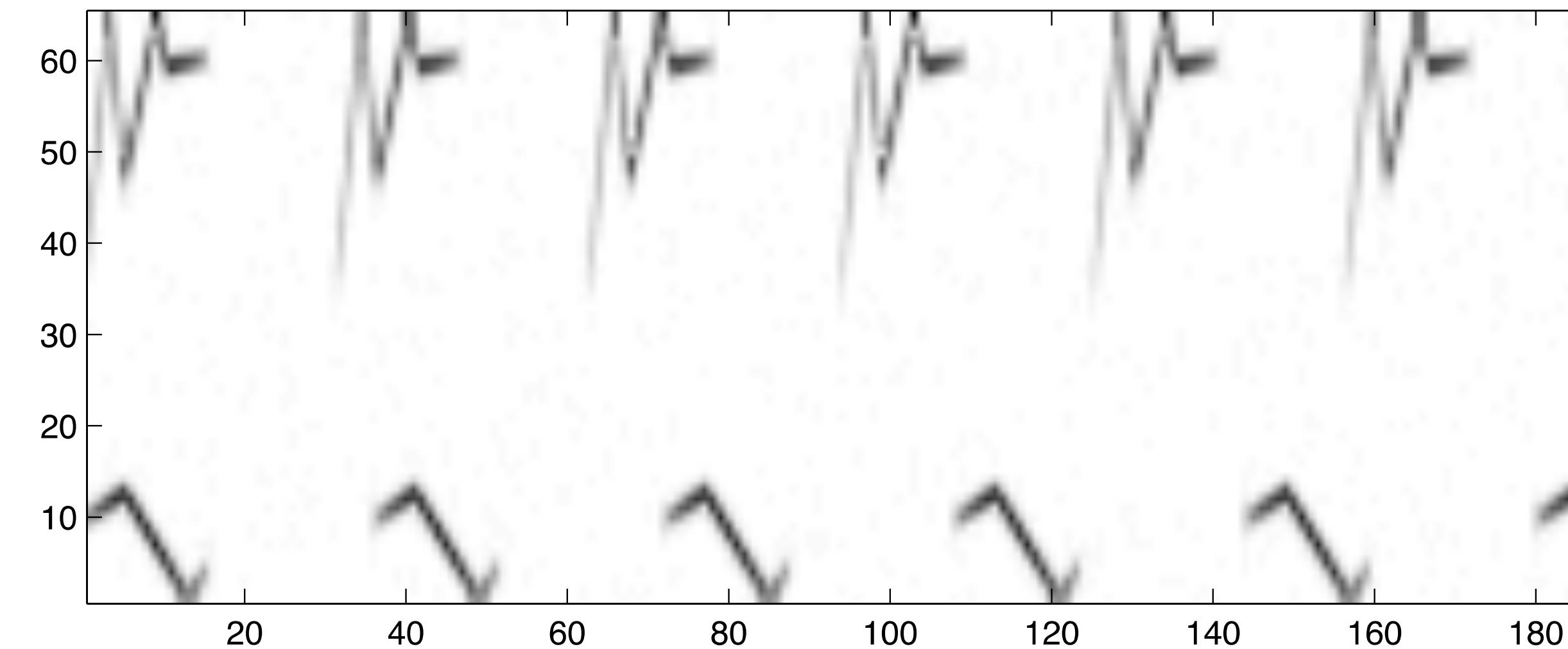
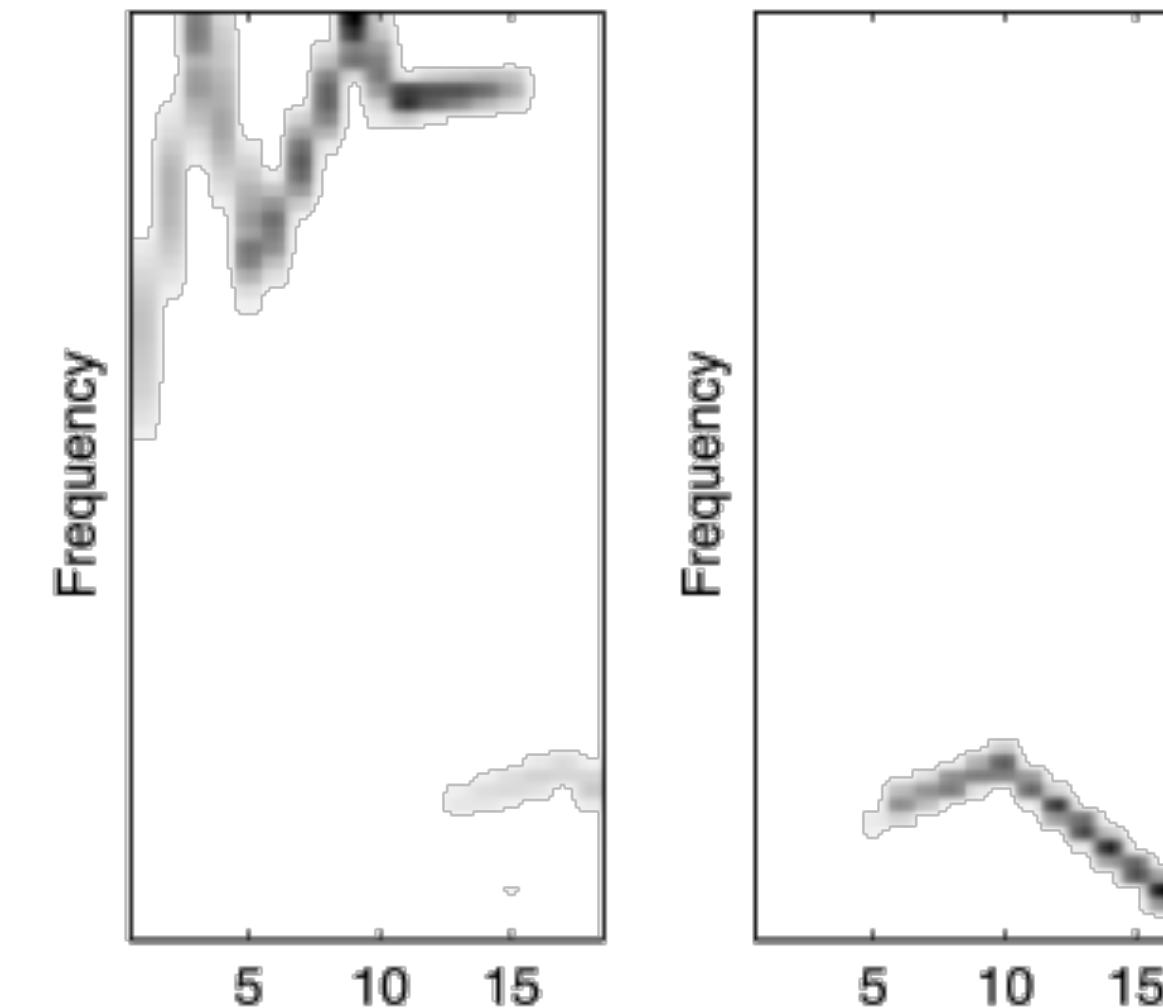
Estimating the parameters

- EM again, this time over two parameters
- E-step:
$$P(\tau, z | x, y) = \frac{P(z)P(x, \tau | z)P(y - \tau | z)}{\sum_{z'} P(z') \sum_{\tau'} P(x, \tau' | z')P(y - \tau' | z')}$$
- M-step:
$$P(z) = \sum_x \sum_y \sum_z P(x, y)P(\tau, z | x, y)$$

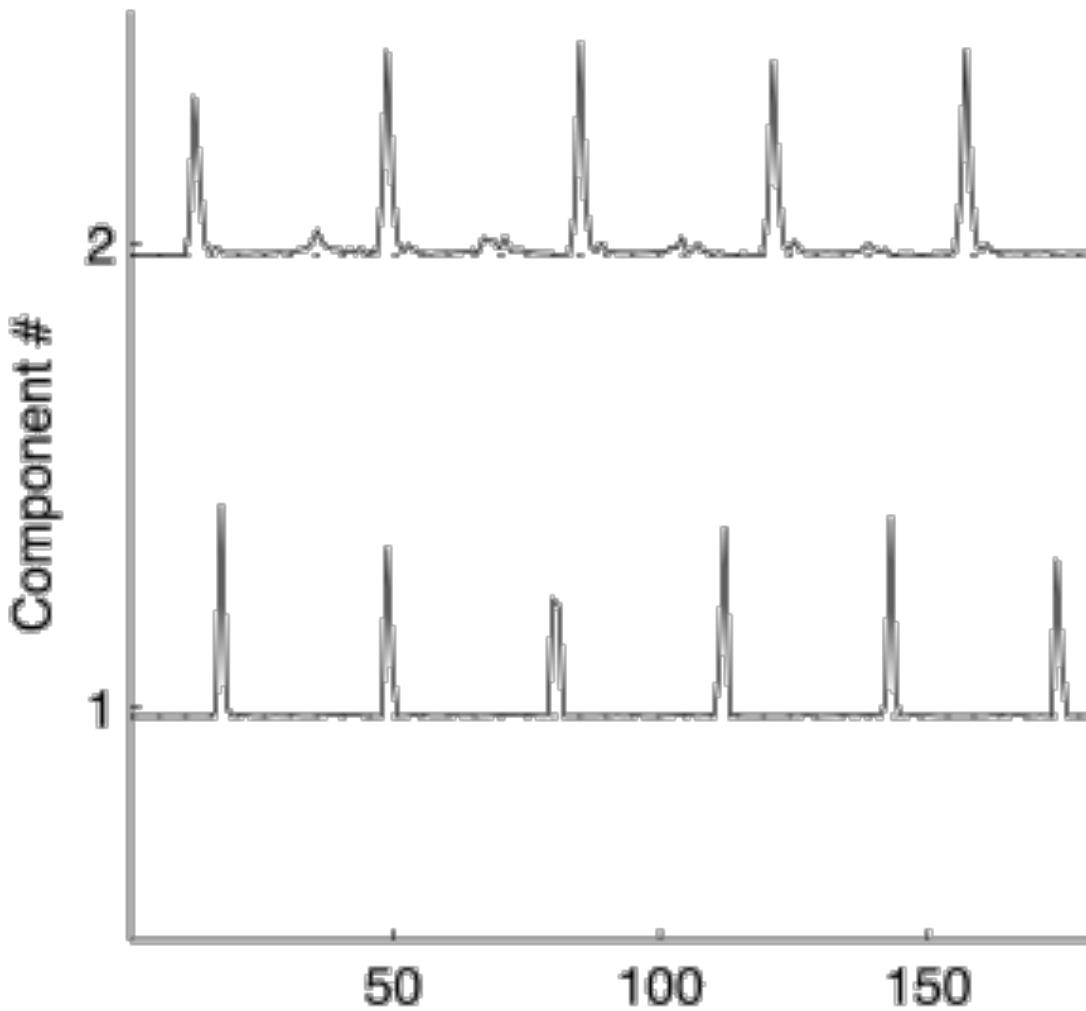
$$P(x, \tau | z) \propto \sum_y P(x, y)P(\tau, z | x, y)$$

$$P(y, z) \propto \sum_x \sum_{\tau} P(x, y + \tau)P(\tau, z | x, y + \tau)$$

Shift-invariant components

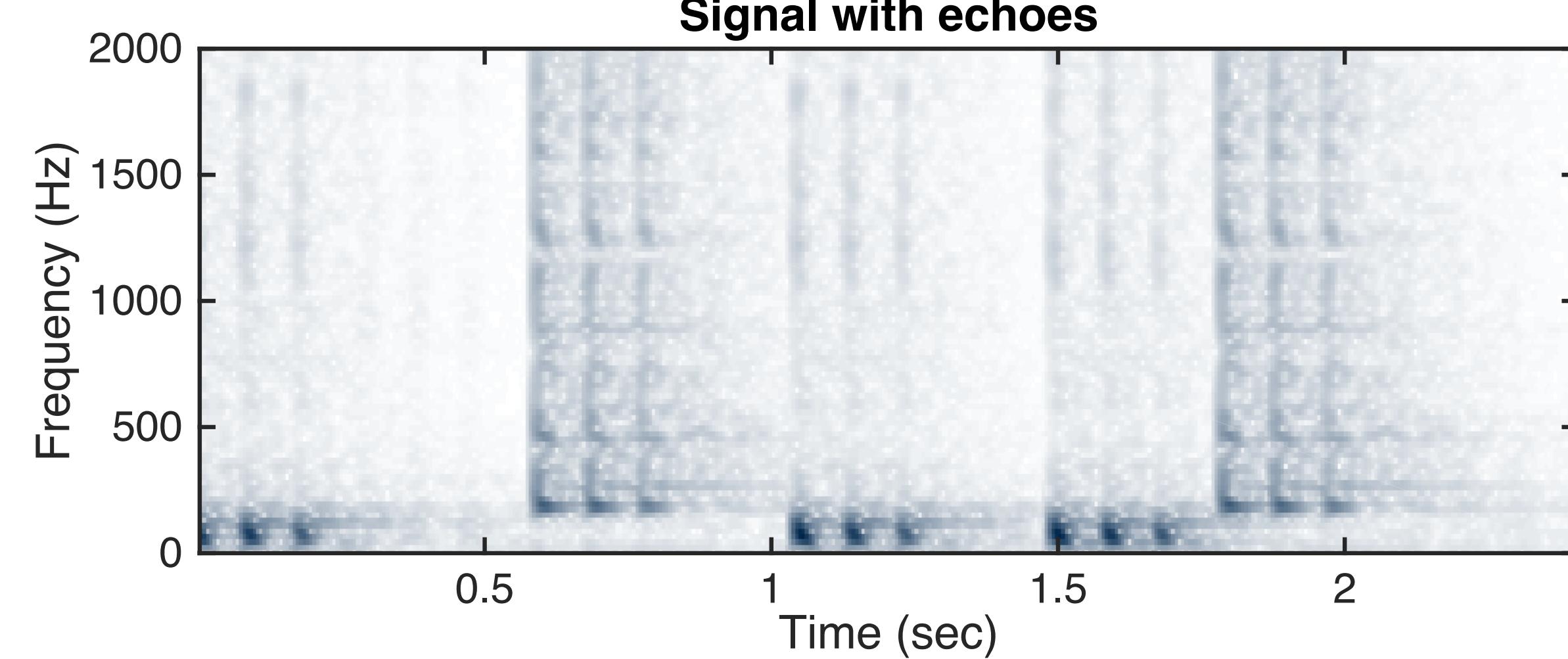
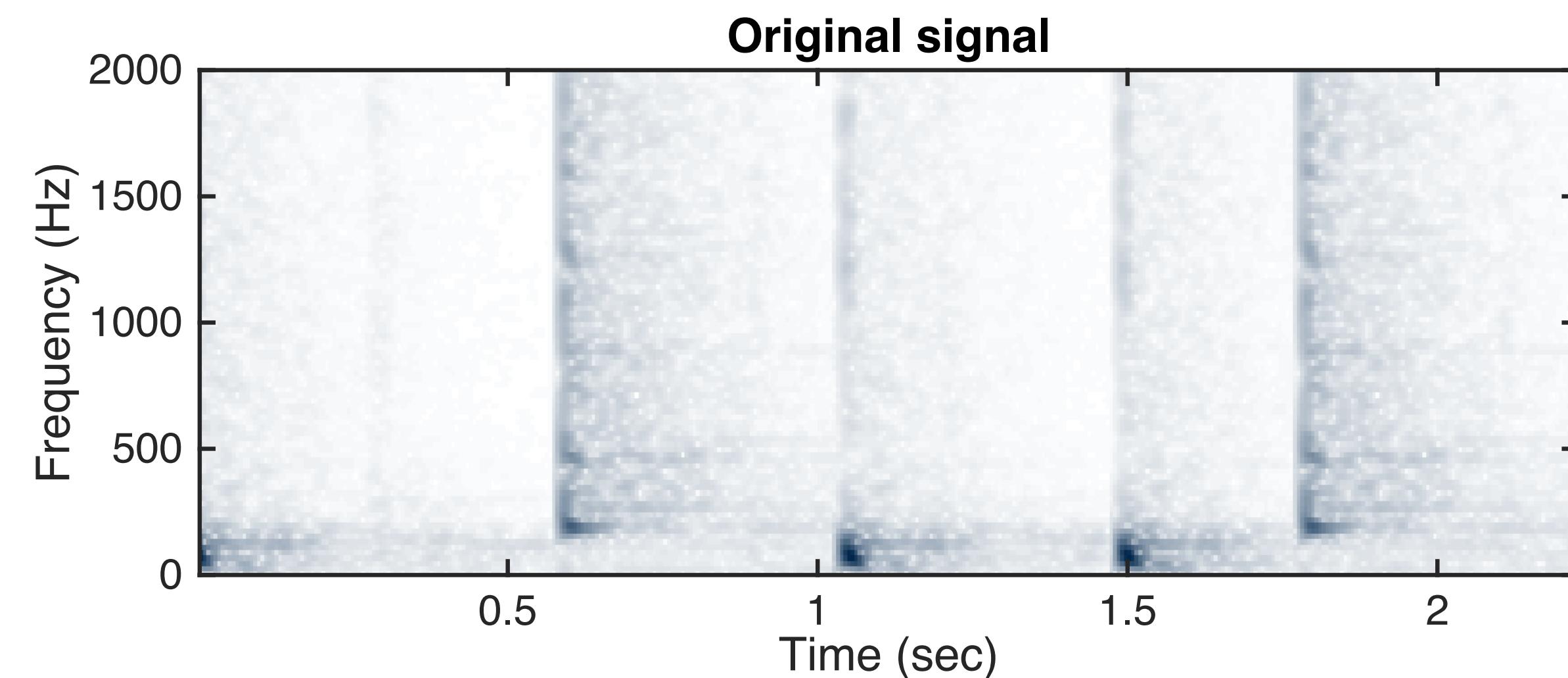
 $P(x, z)$  $P(x, \tau|z)$ 

Component #

 $P(y|z)$

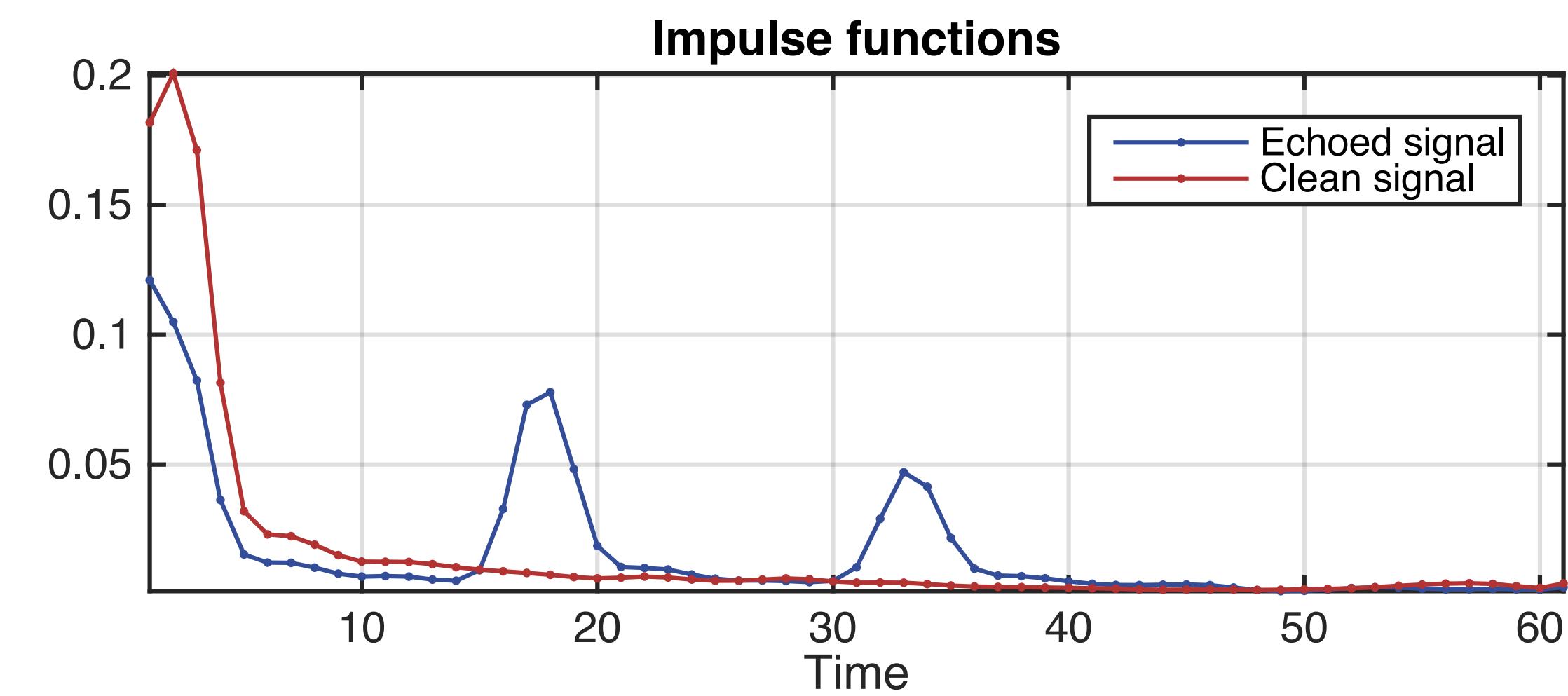
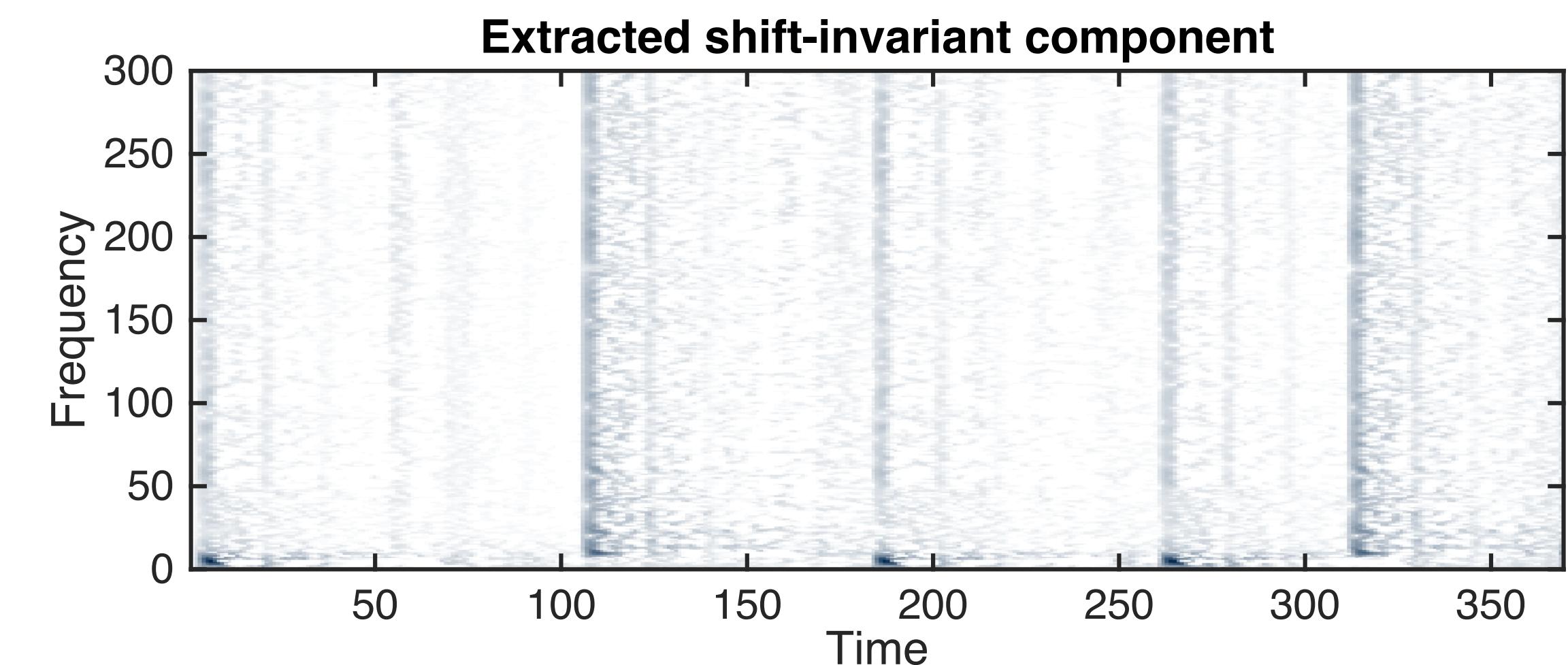
An application

- Deconvolution
 - Echoes are repetitions of the same pattern in time
- Decompose input using a 1-component analysis
 - Component is signal
 - Weight is echo pattern



Analyzing this

- We extract a time/frequency pattern (the one component)
 - Corresponds to the original sound
- Also get an impulse function
 - Corresponds to the echoes
- These two convolved approximate the input

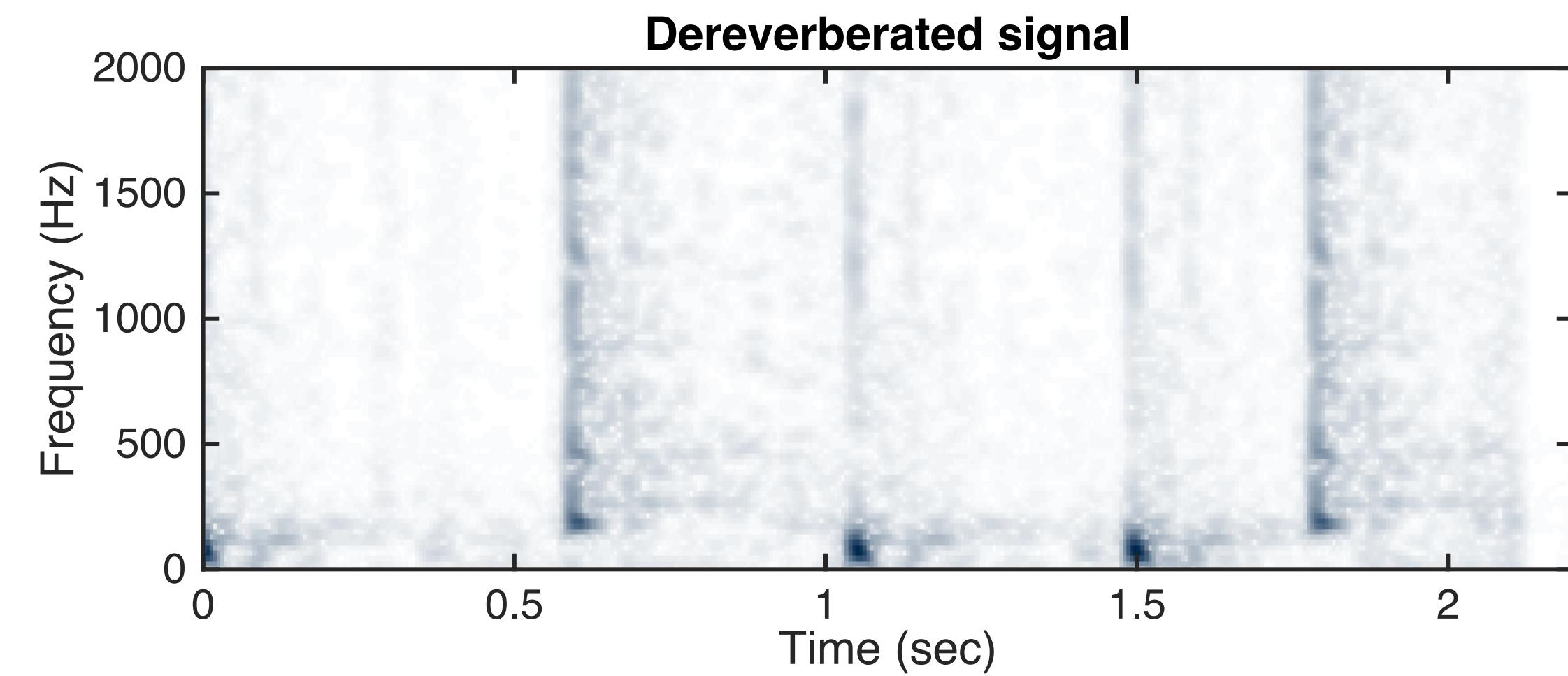
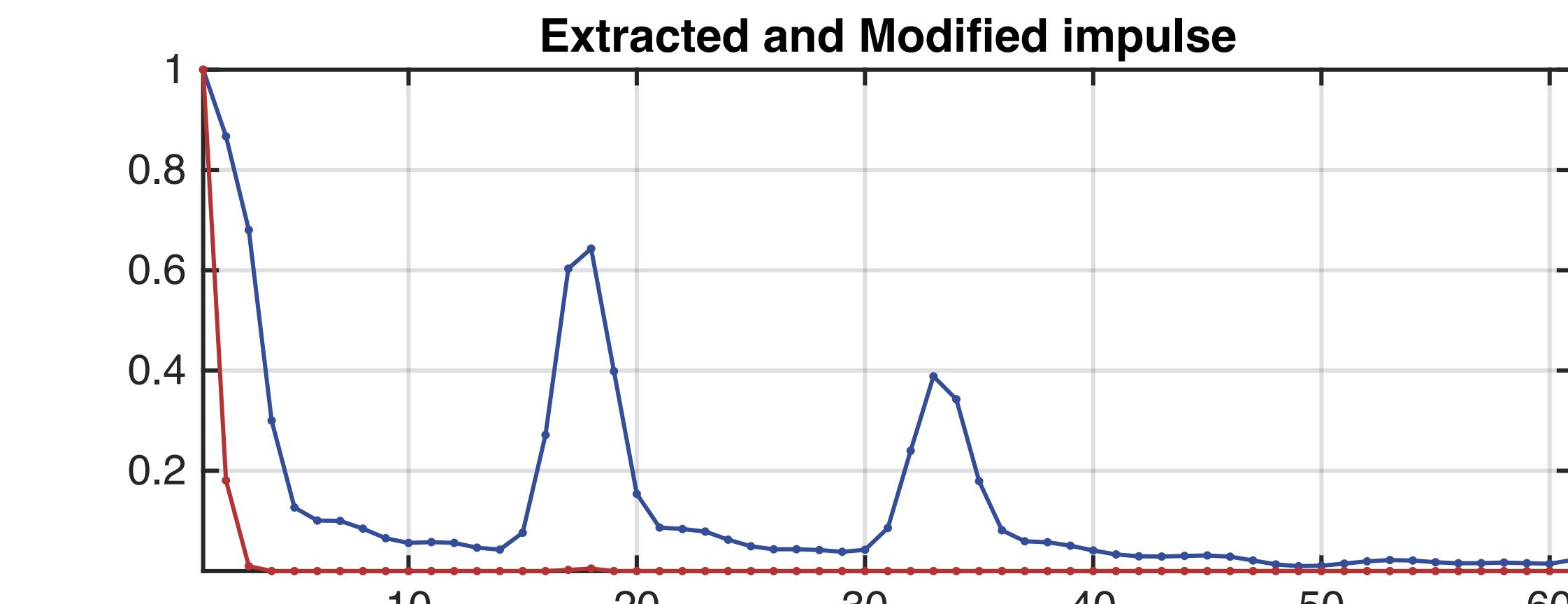


Removing the echoes

- We can suppress the echoes in the impulse and resynthesize
- Results in input without echoes



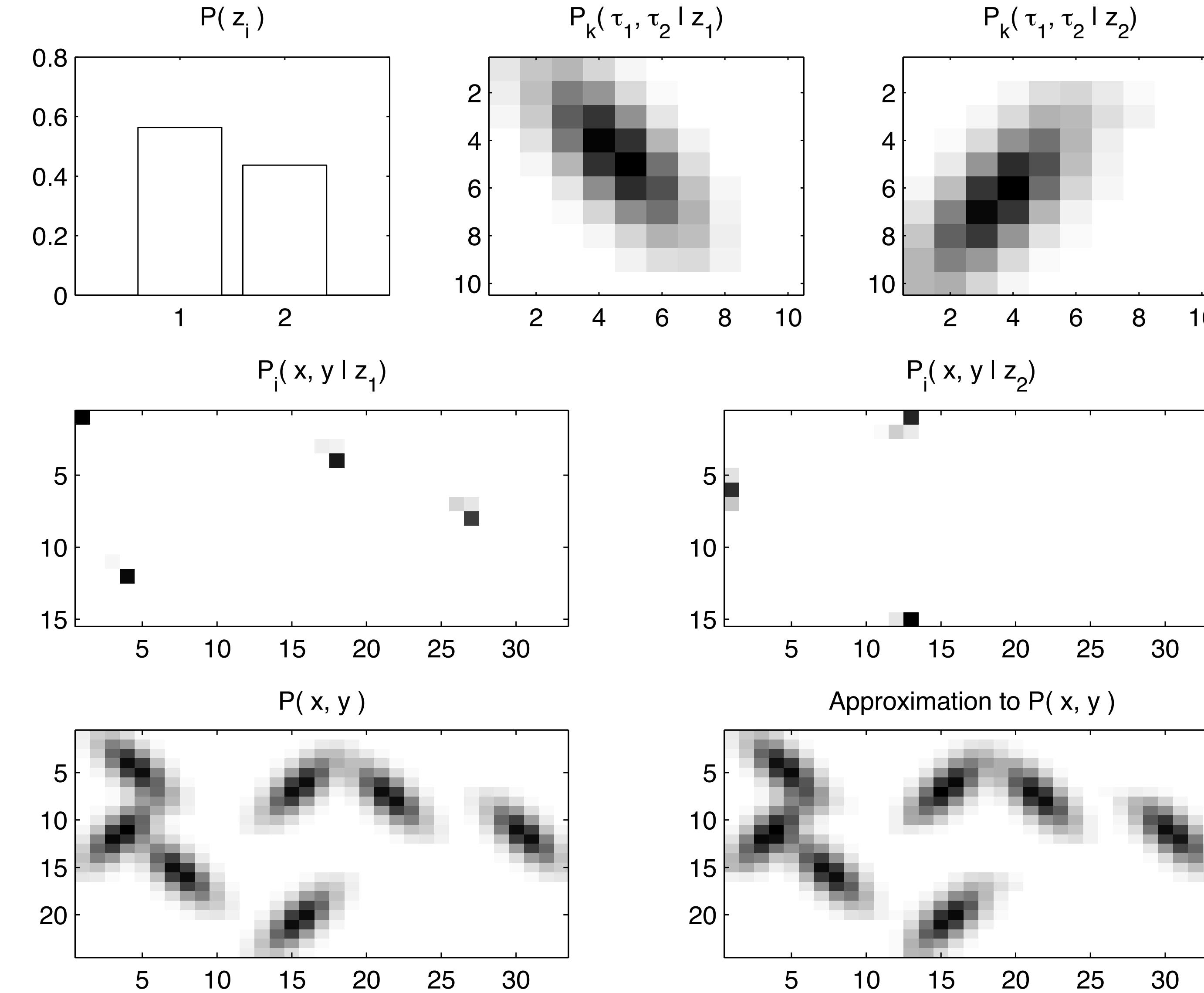
Cleaned up Sound



Generalizing to multiple axes

- We can also formulate impose shift-invariance over all dimensions
- In this case components are multi-dimensional and can shift everywhere

2-D example



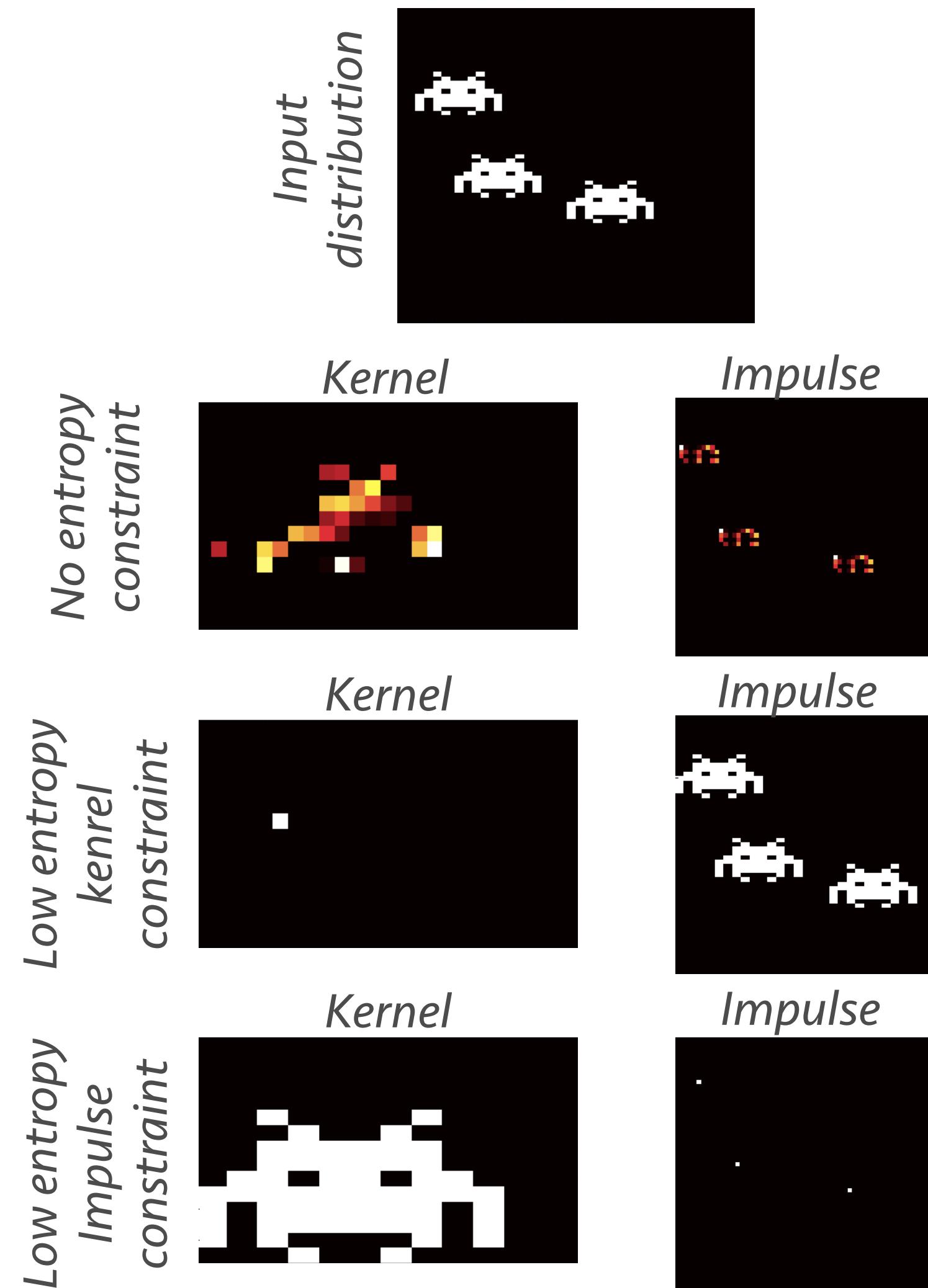
A bit of a complication

- Convolution is commutative
 - Component/weight confusion

- Using sparsity control
 - Impose entropic constraints
 - Apply as a prior

$$P(\mathbf{q}) = e^{-\beta H(\mathbf{q})}$$

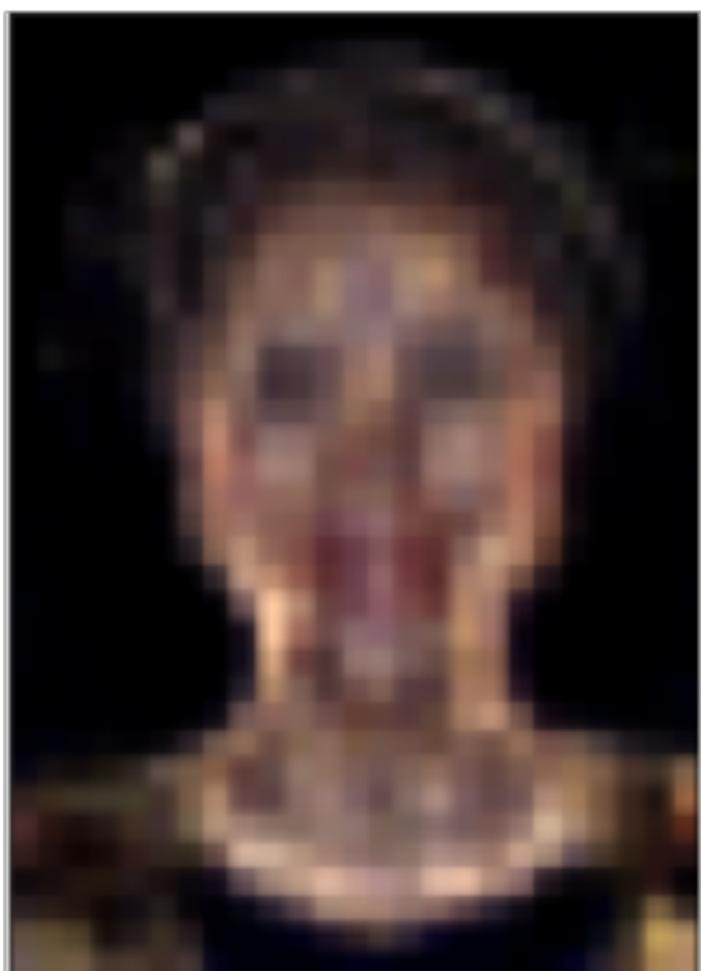
- Or use simpler forms
 - More at next lecture



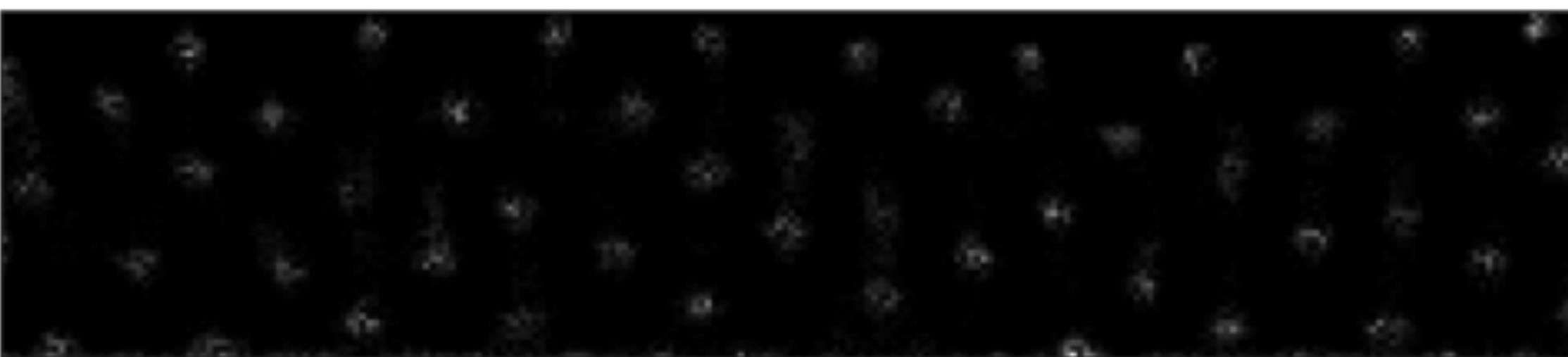
Pattern discovery



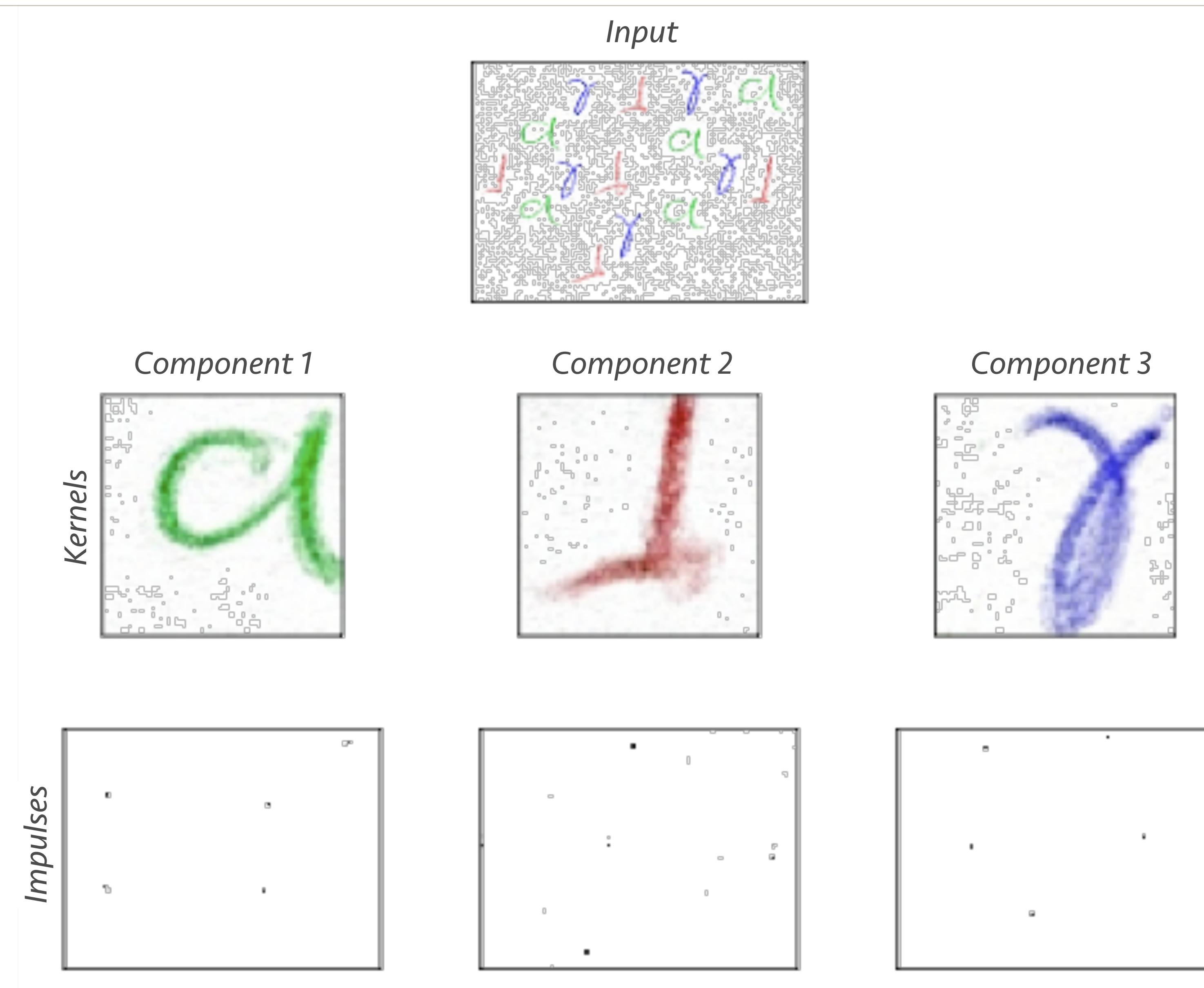
Kernel distribution



Impulse distribution

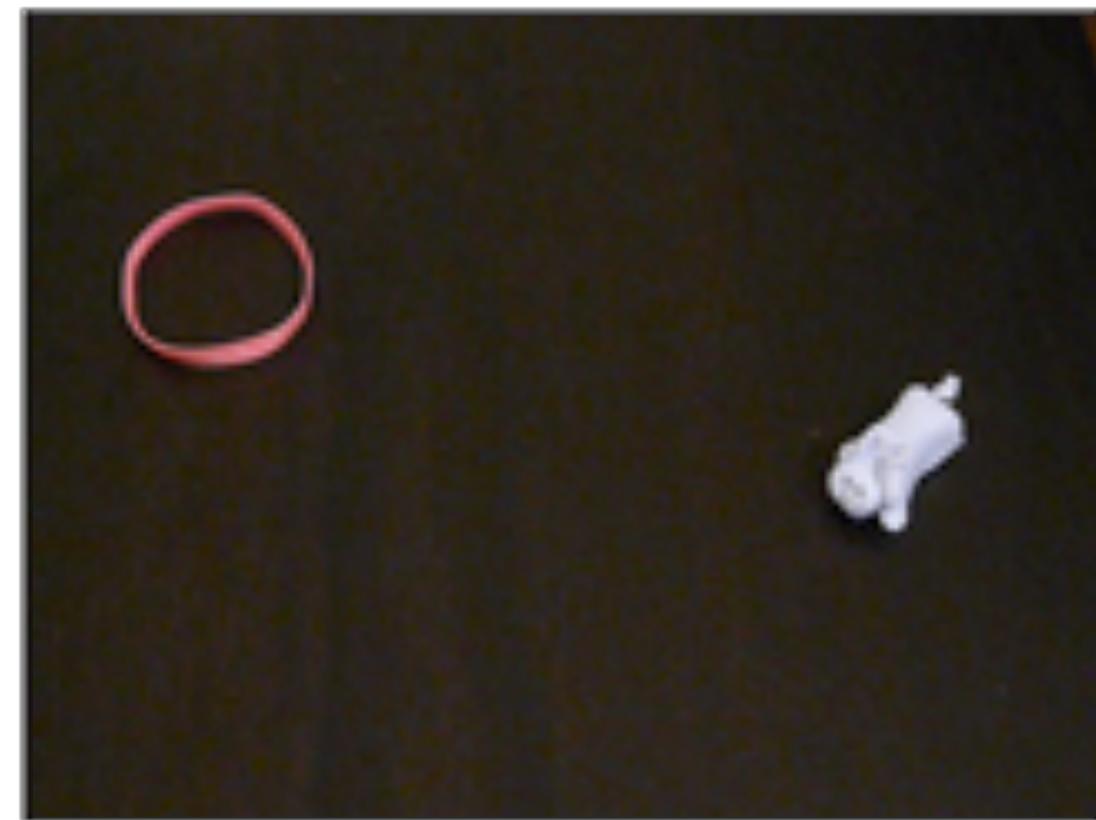


Shift-invariant features

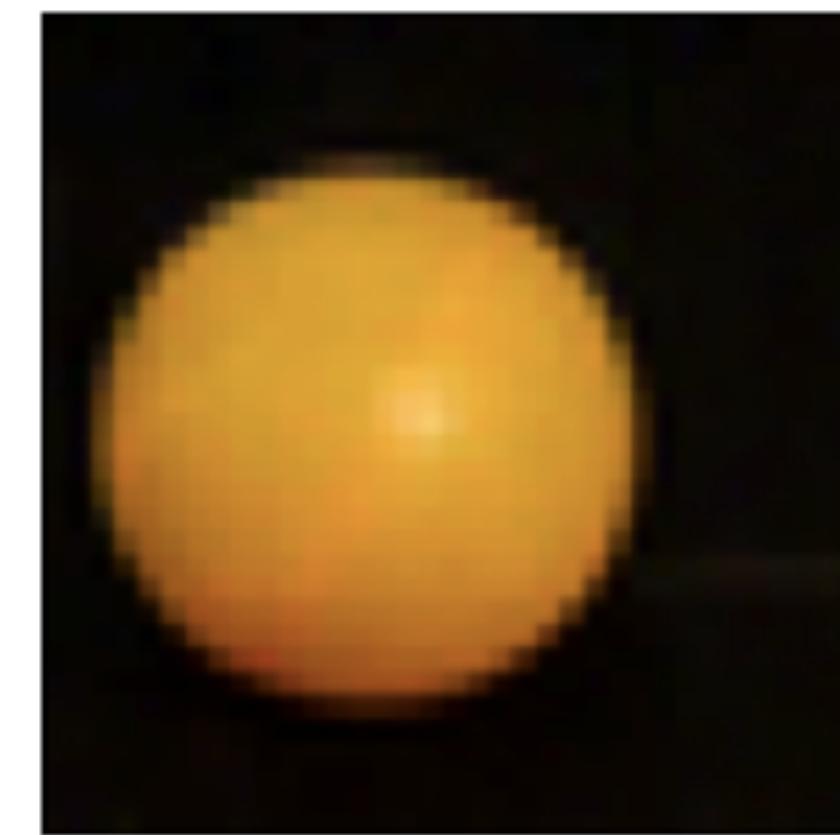


More shift-invariant features (in 4d)

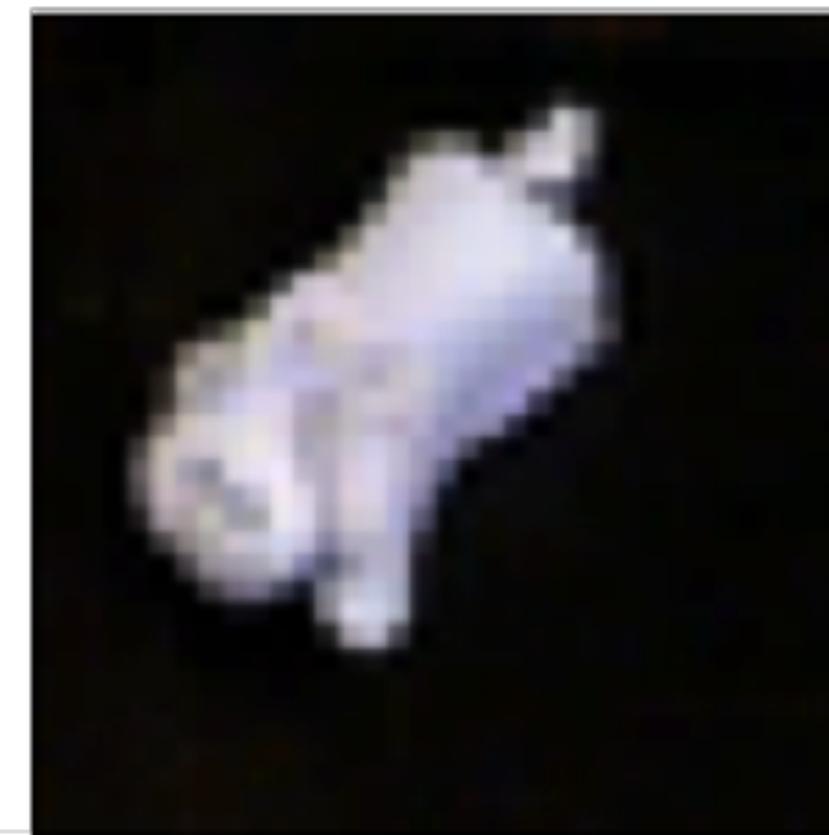
Description of input



Kernel 1



Kernel 2

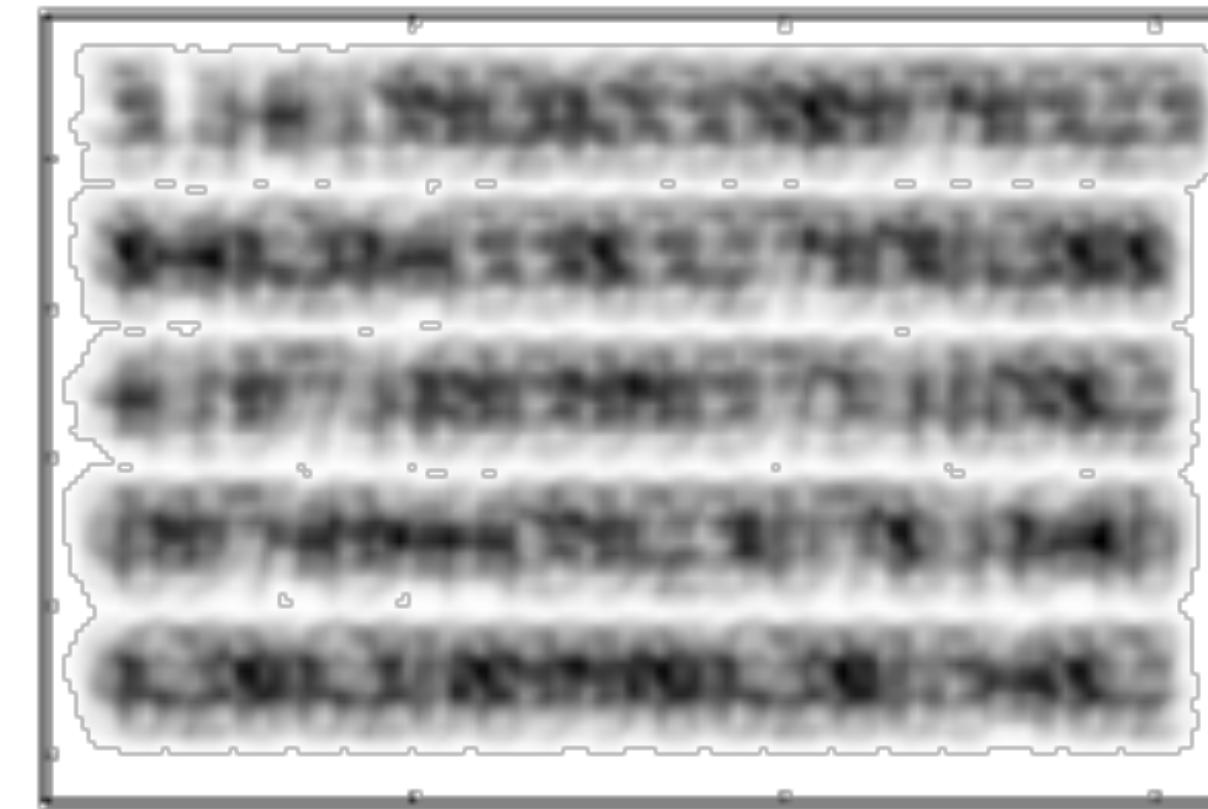


Kernel 3

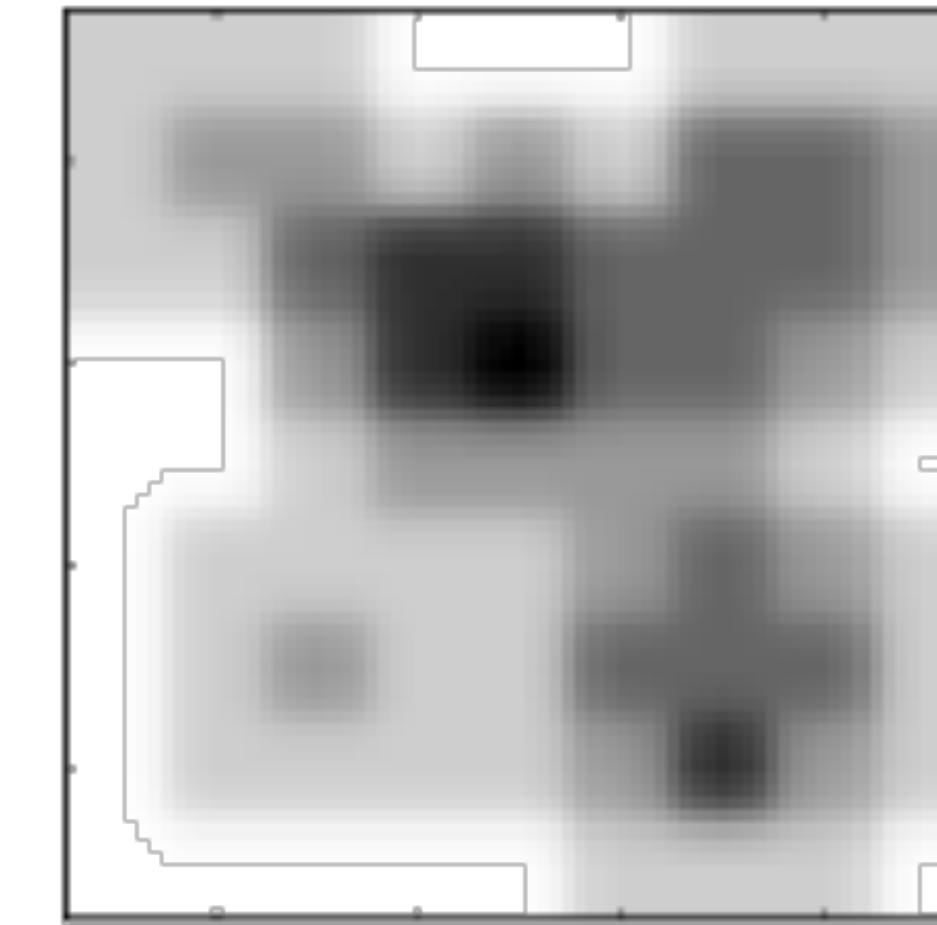


Image deconvolution

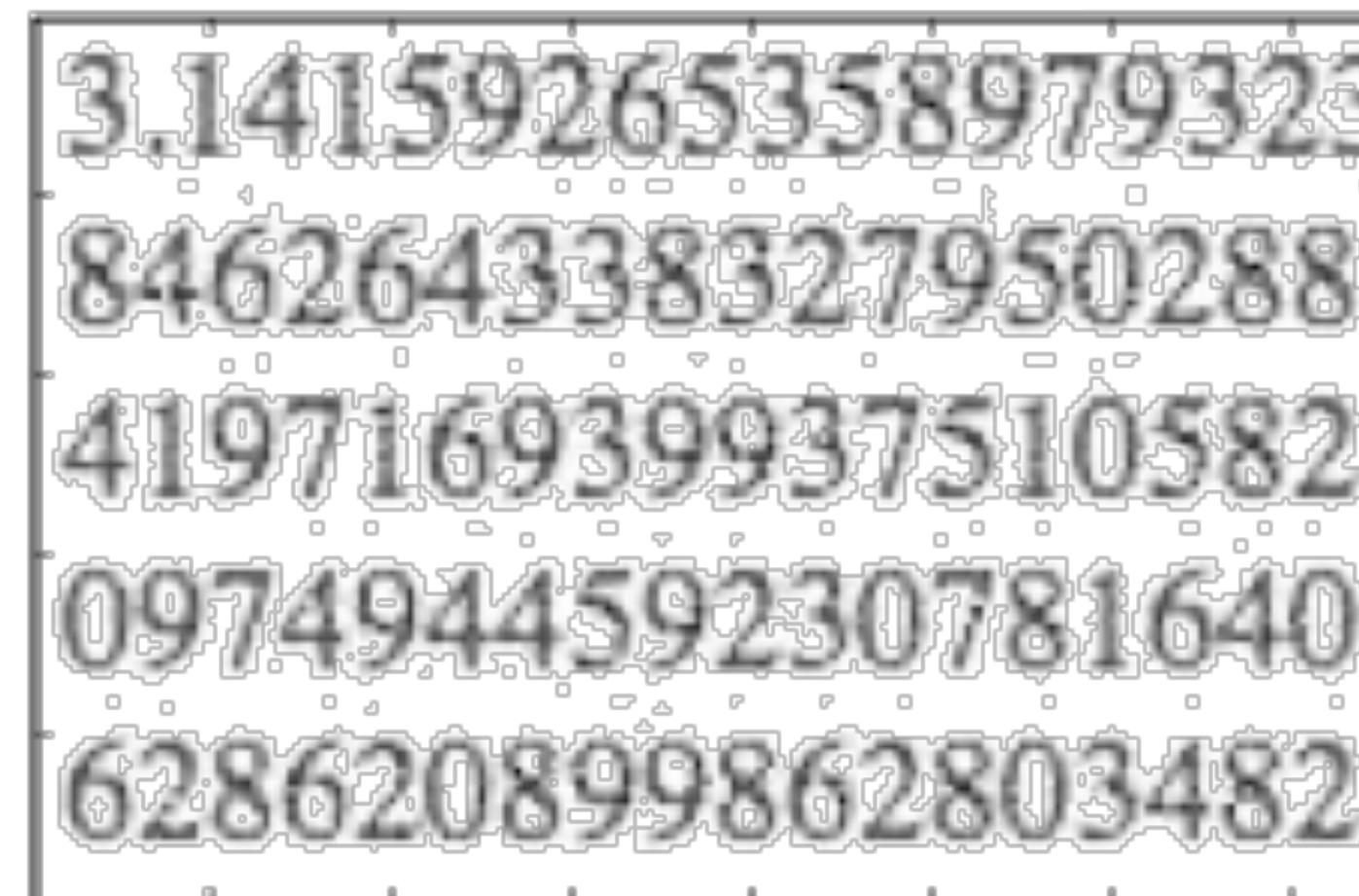
Blurred input



Blurring kernel



Deconvolution result



Many more models

- Probabilistic approach opens new doors
 - E.g. HMMs with PLCA state model
 - Mixtures of PLCA models
 - etc ...
- Remember, probabilities are nice!

Recap

- Latent Variable Models
 - Matrix and tensor decompositions
- Probabilistic versions of the above
- Convulsive models

Reading

- LSA
 - <http://openscienceasap.org/wp-content/uploads/2013/10/Mathematical-Foundations-for-LSA.pdf>
- PLSI/PLSA/LDA
 - <https://www.iro.umontreal.ca/~nie/IFT6255/Hofmann-UAI99.pdf>
 - <http://jmlr.csail.mit.edu/papers/volume3/blei03a/blei03a.pdf>
- N-way decompositions
 - http://www.models.life.ku.dk/sites/default/files/brothesis_0.pdf
- Convolutive models
 - <http://paris.cs.illinois.edu/pubs/smaragdis-icassp2008.pdf>