

Learning Time Series

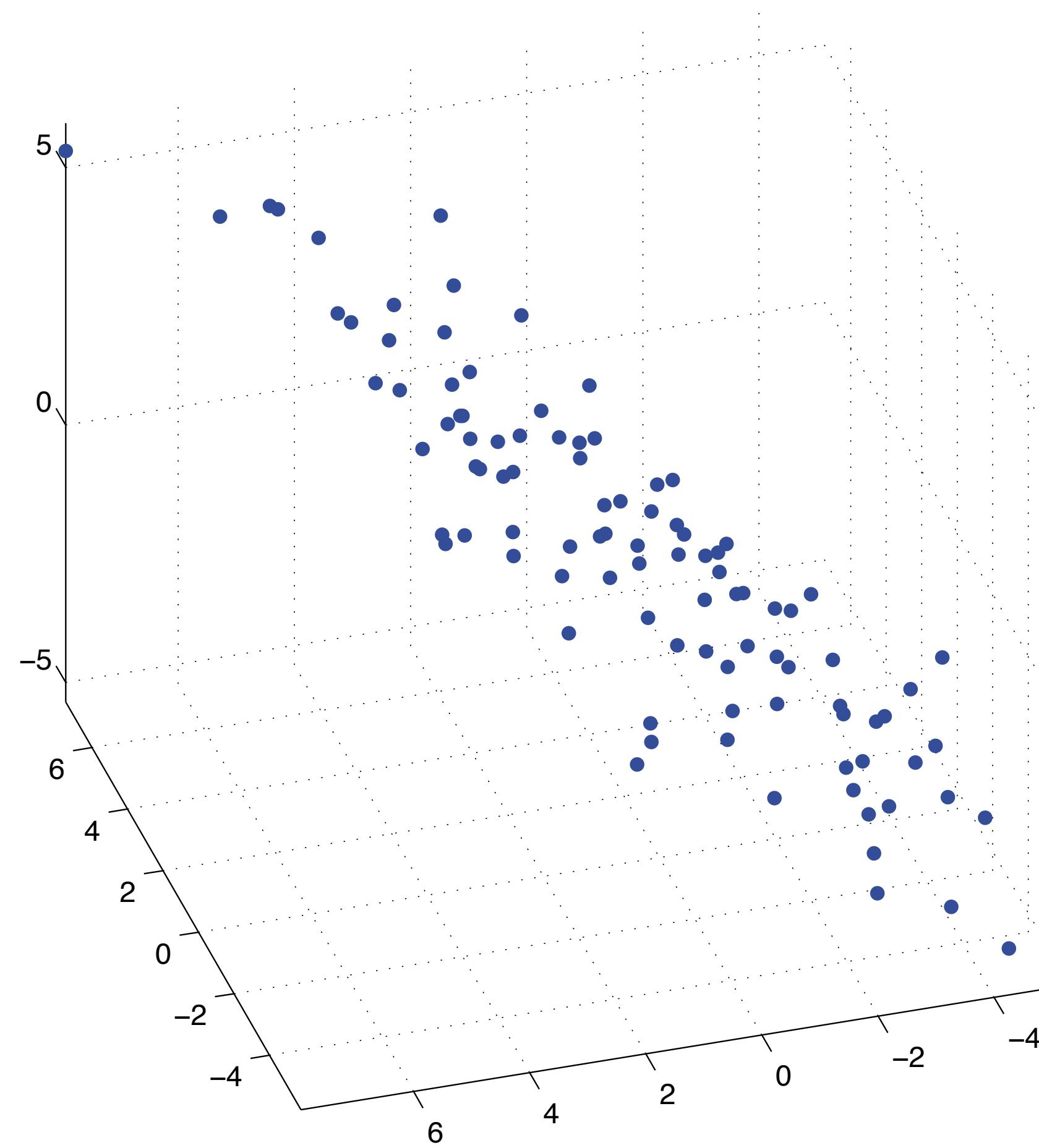
5 October 2020

Today's lecture

- Doing machine learning on time series
- Dynamic Time Warping
- Hidden Markov Models

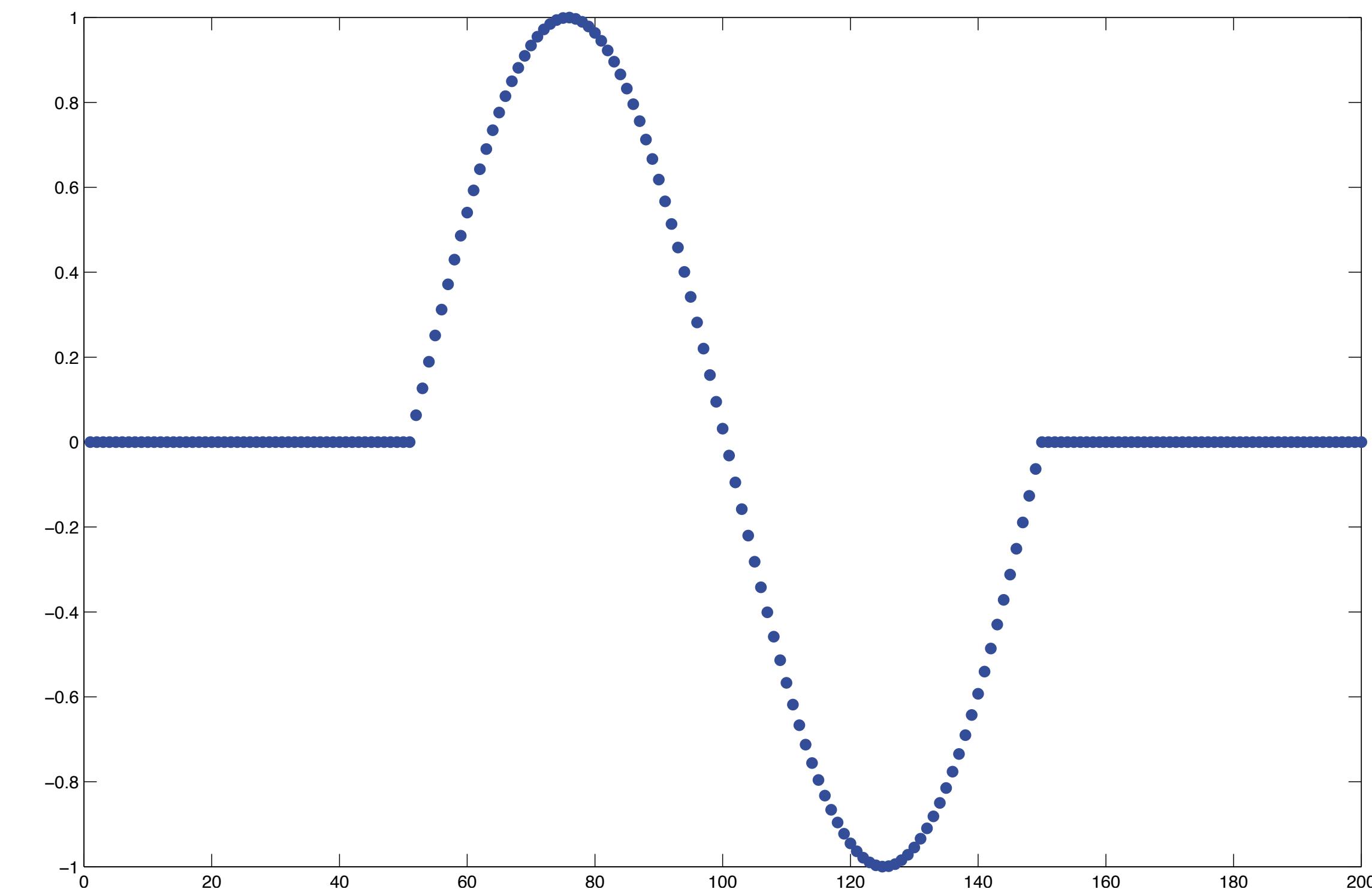
Our view of data so far

- Data are points in a high-dimensional space



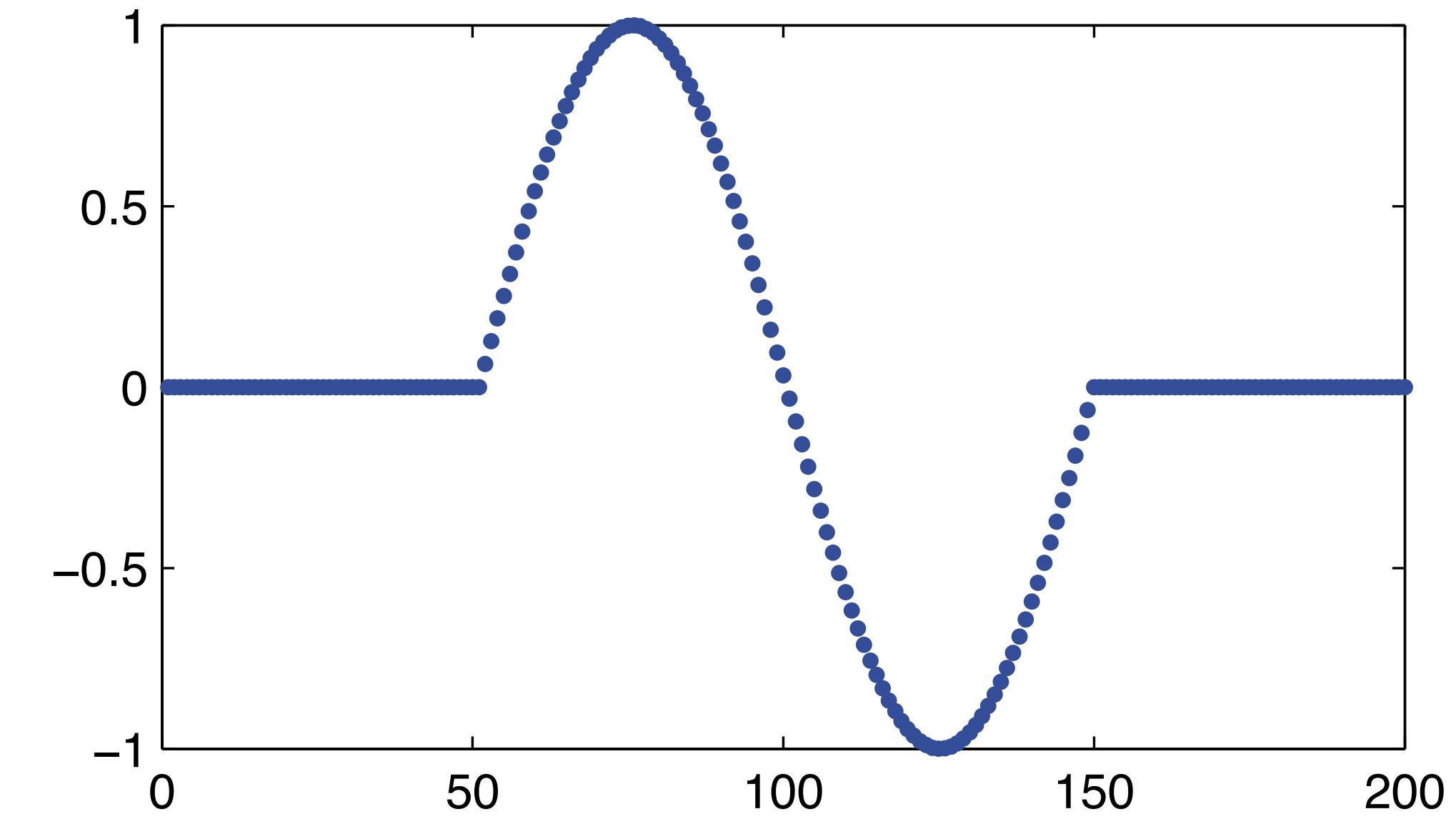
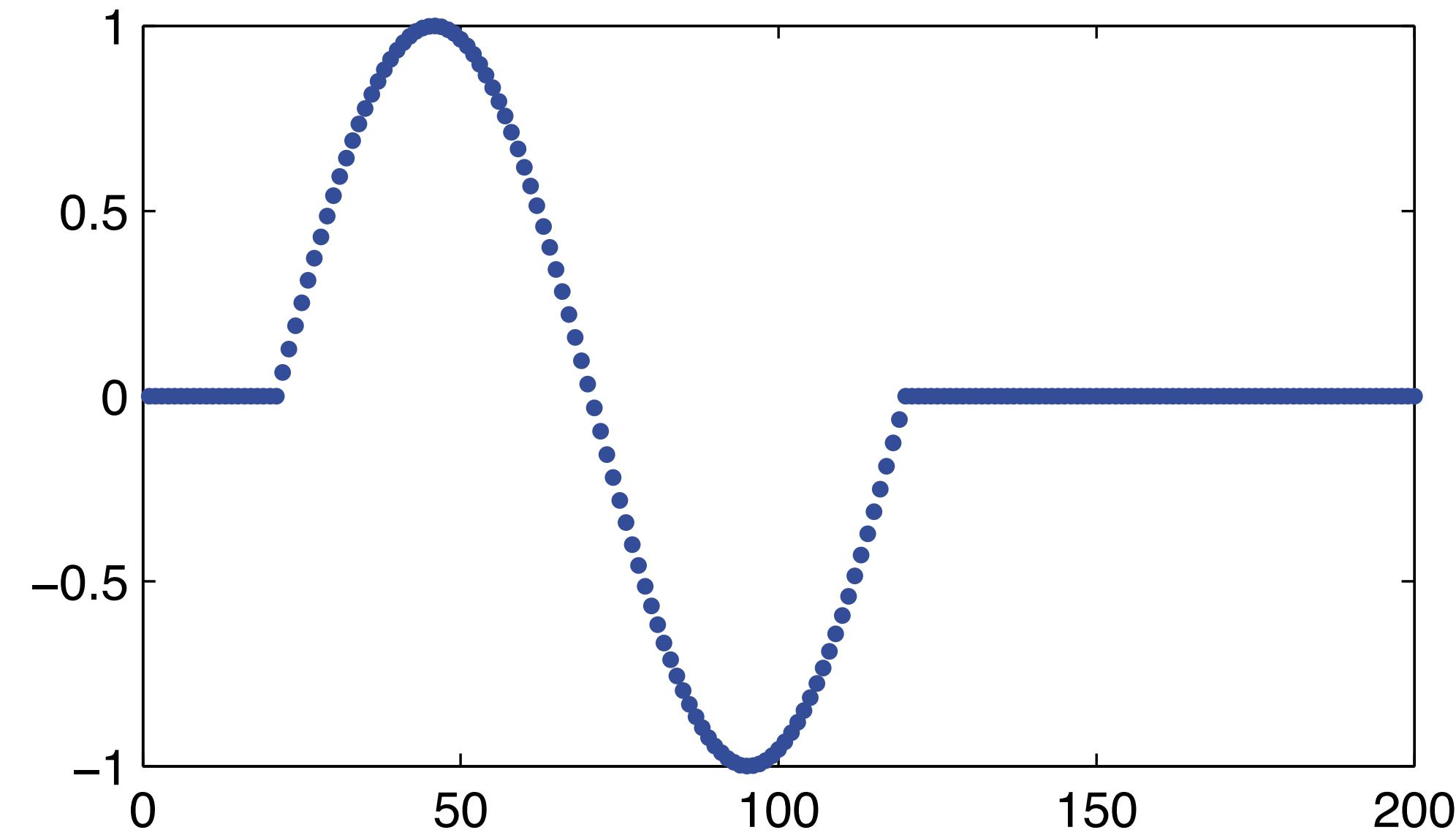
What about time series?

- A sequence of samples, can be thought of as a point in a **very very high-D space**
 - Often a bad idea
 - Remember the curse!!



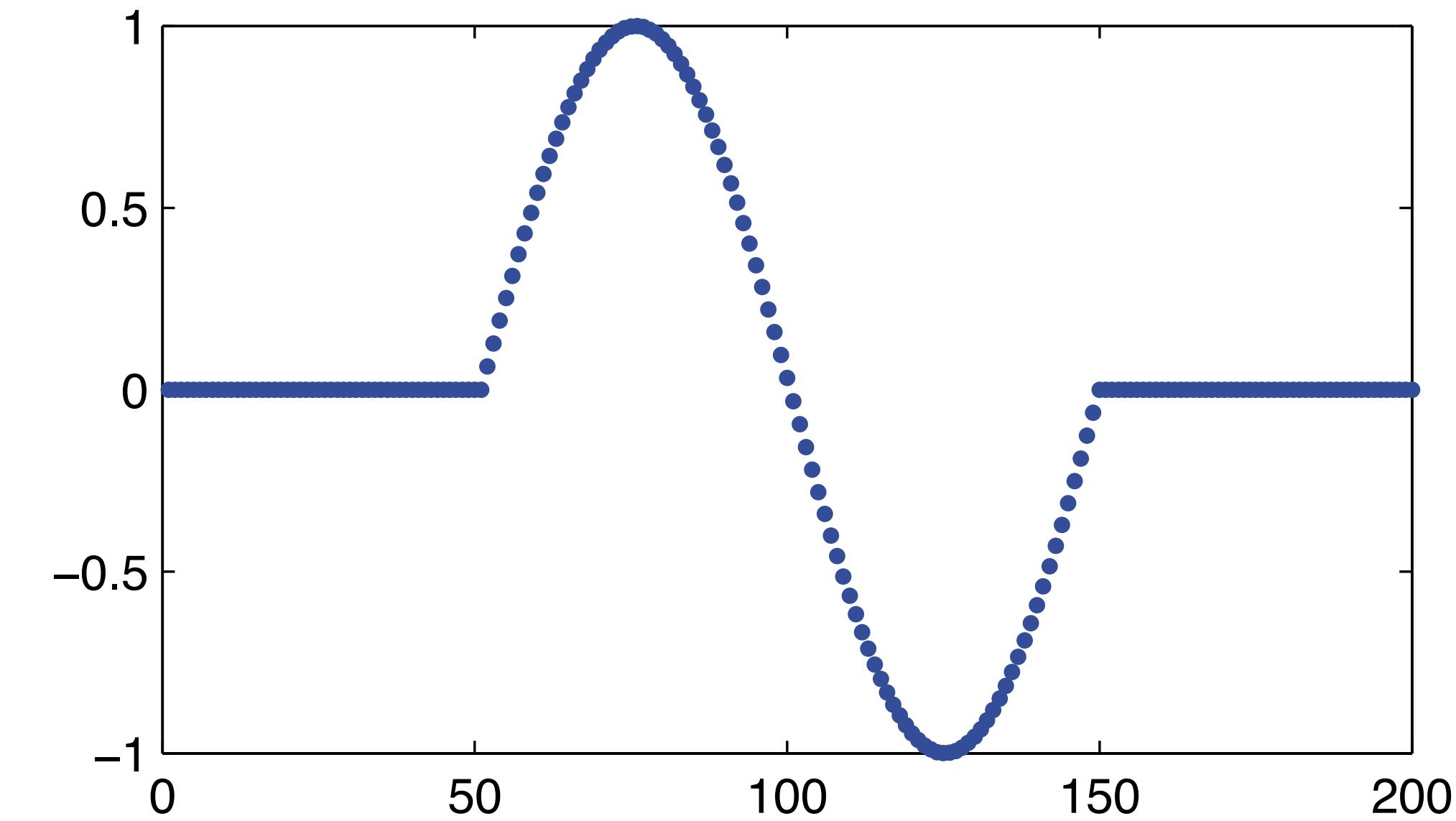
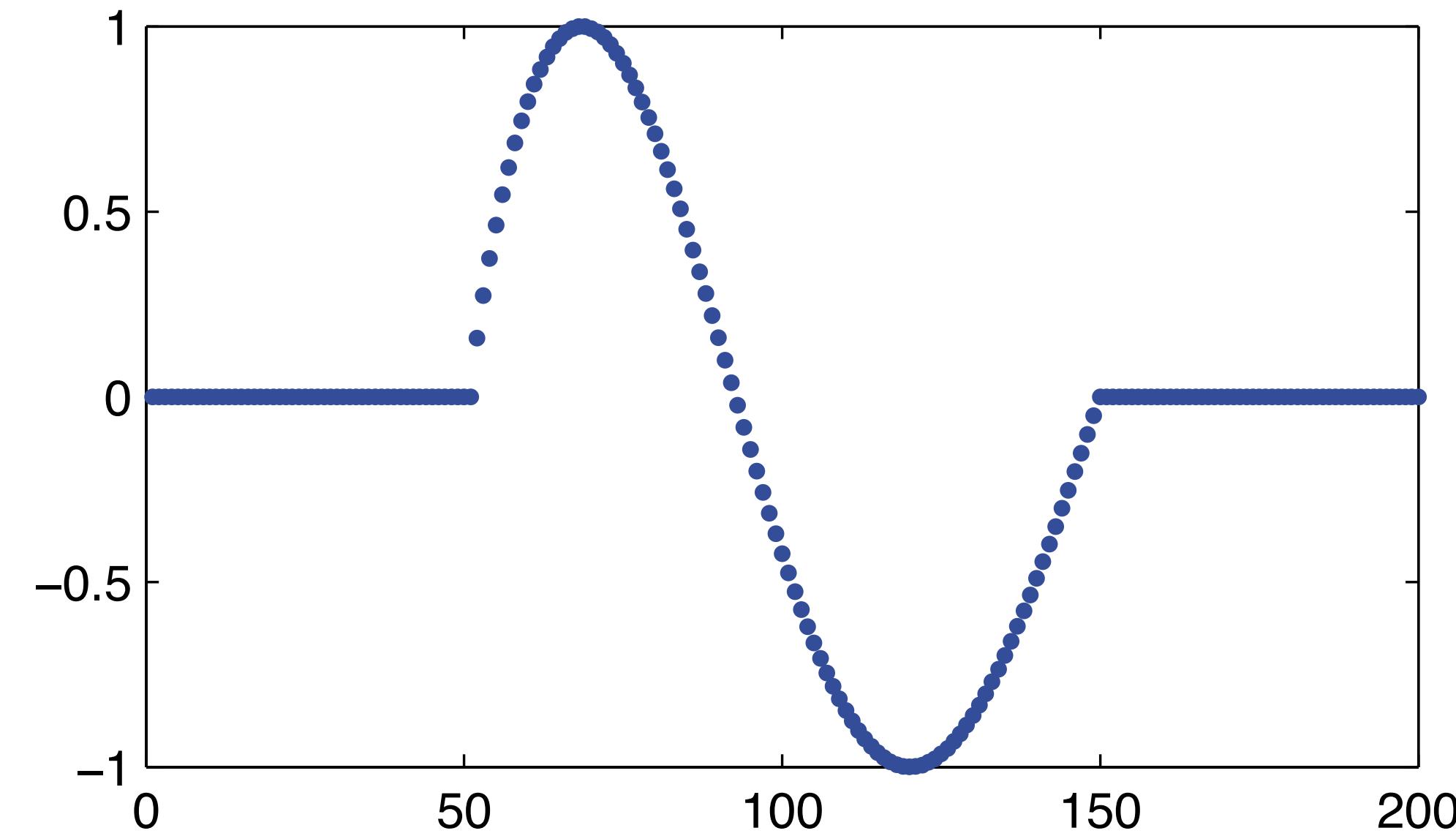
Shift variance

- Time series have shift variance
 - Are these two points close?



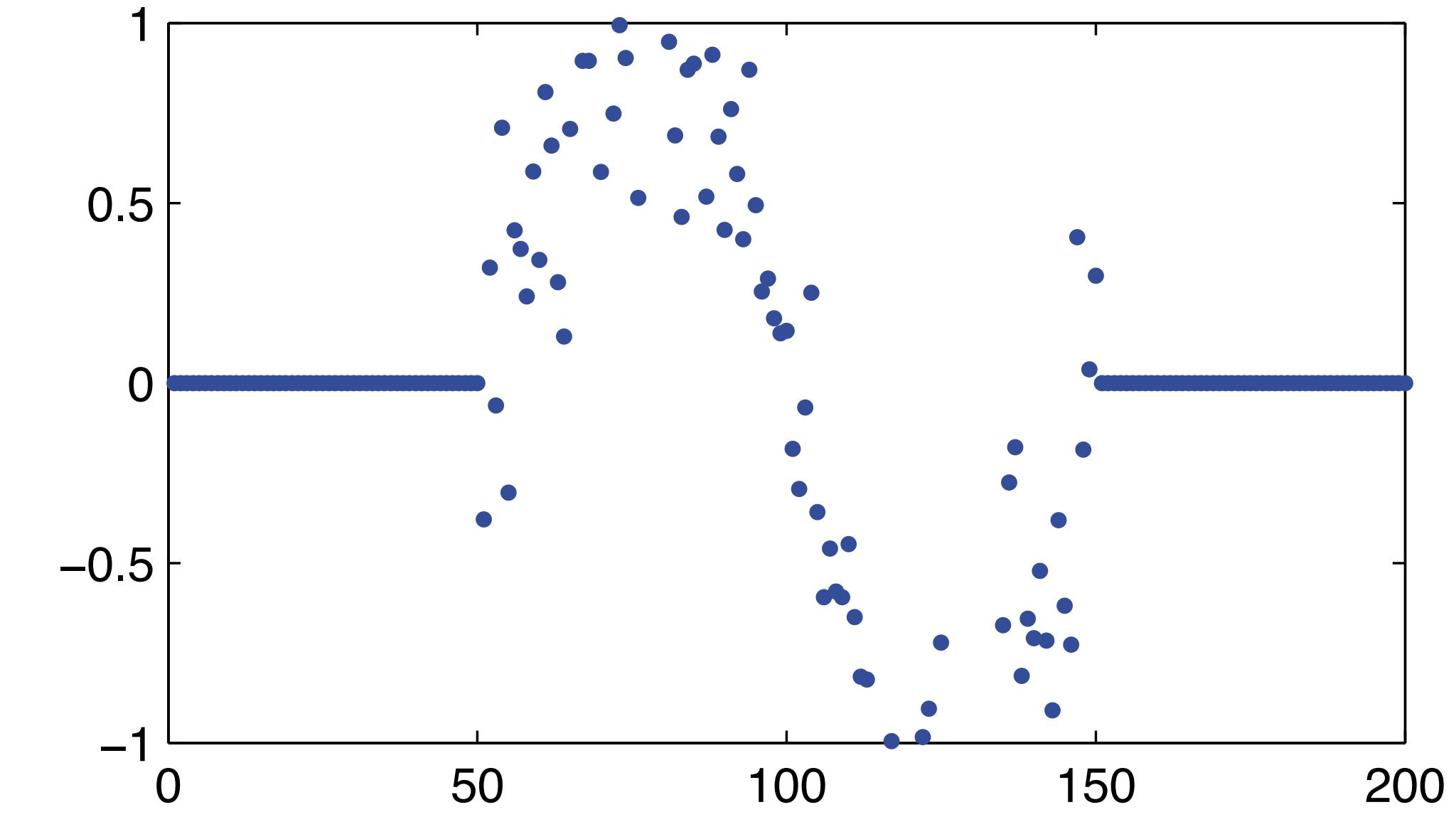
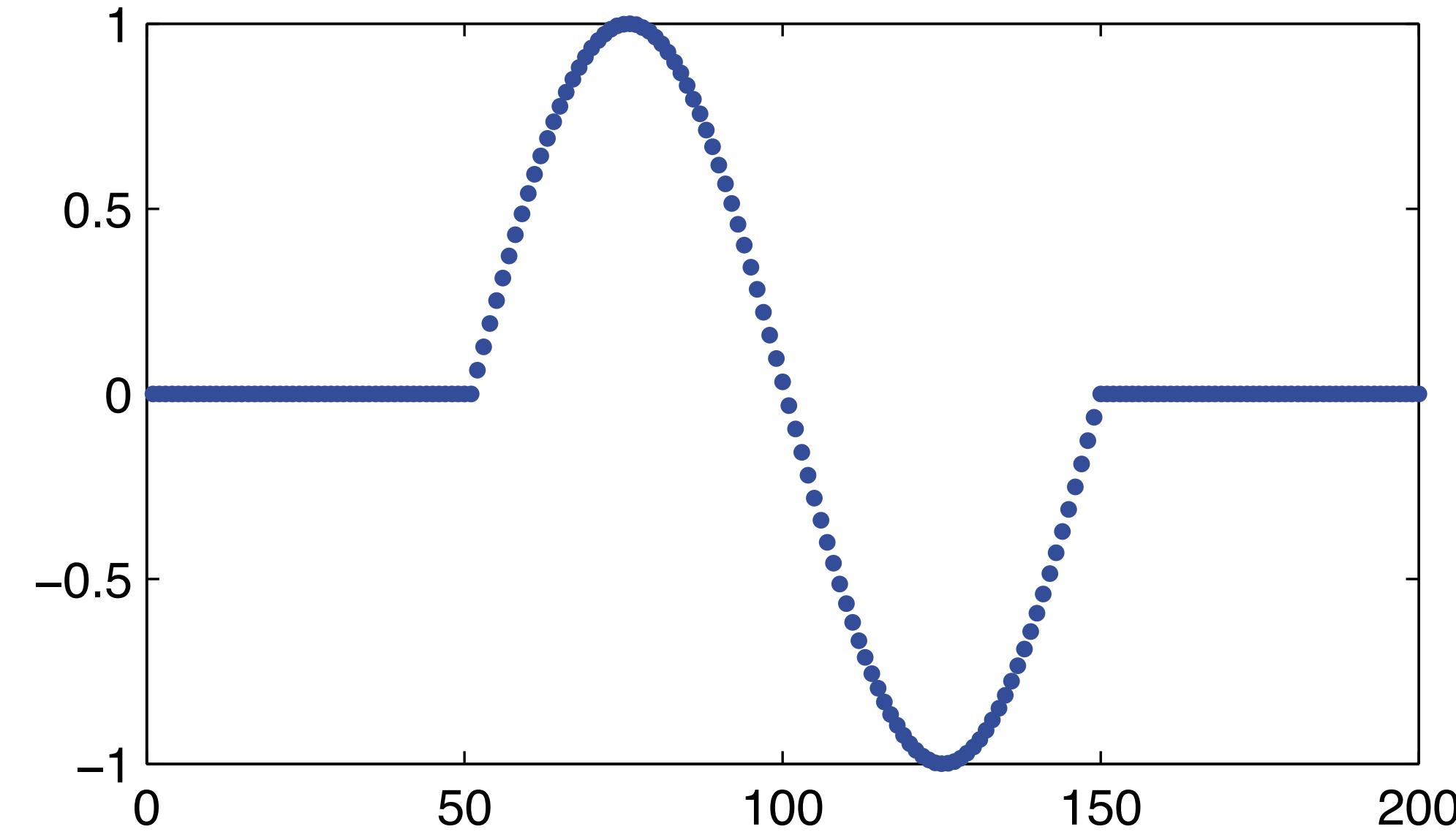
Time warping variance

- Slight changes in timing are not relevant
 - Are these two points close?



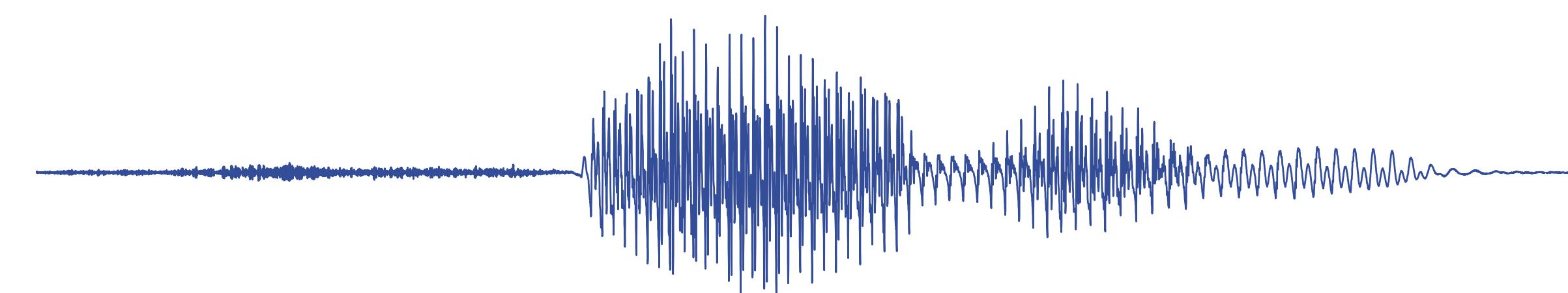
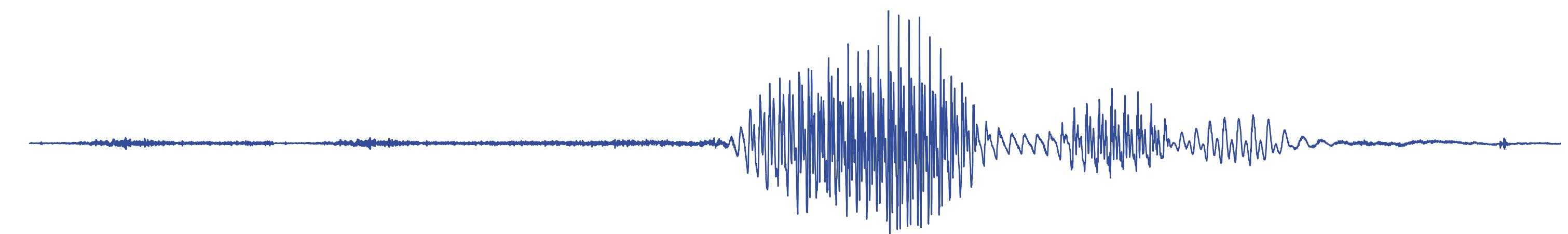
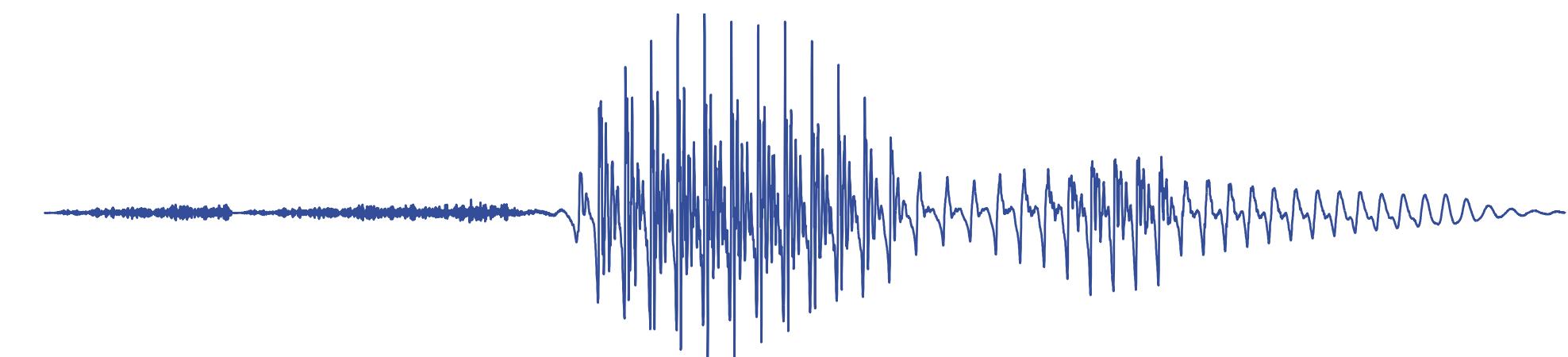
Noise/filtering variance

- Small changes can look serious
 - How about these two points?



A real-world case

- Spoken digits



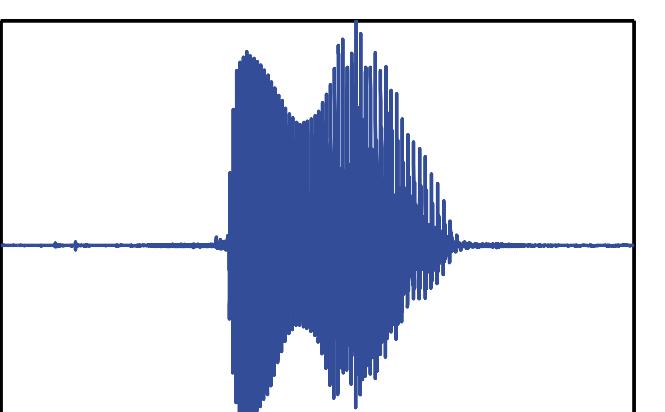
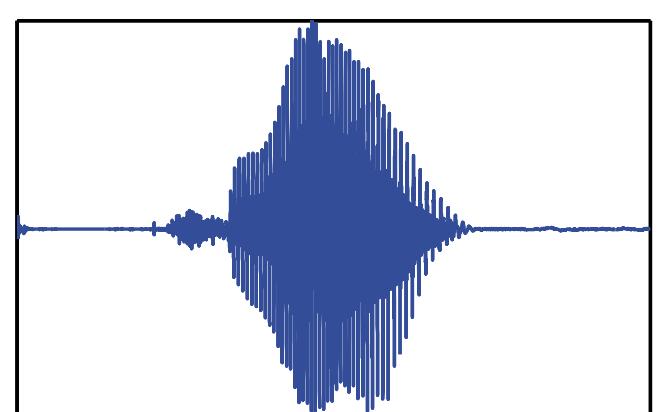
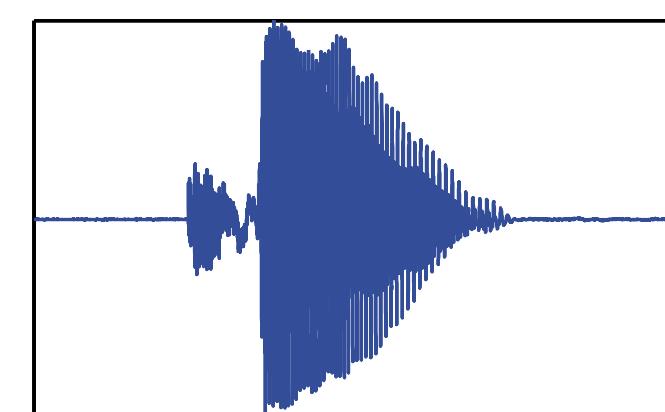
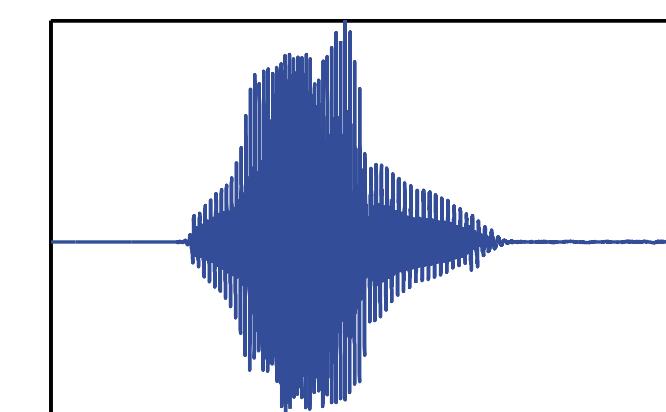
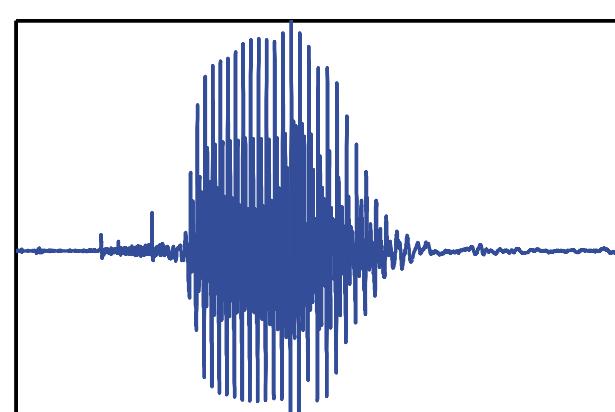
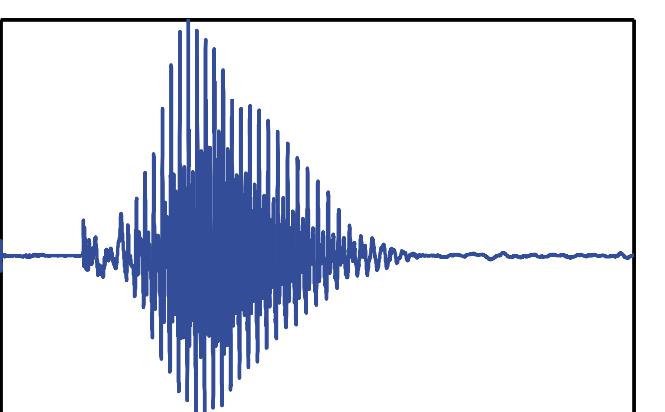
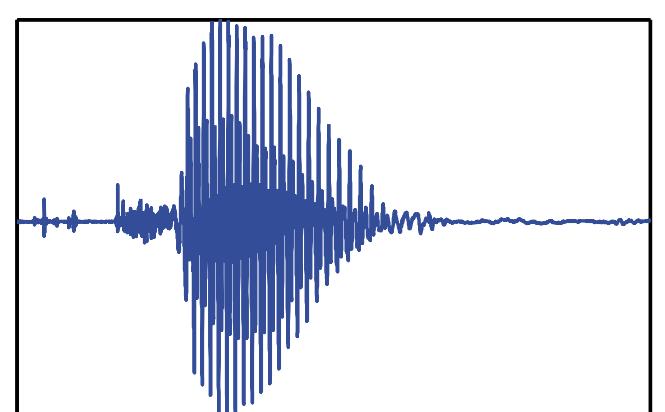
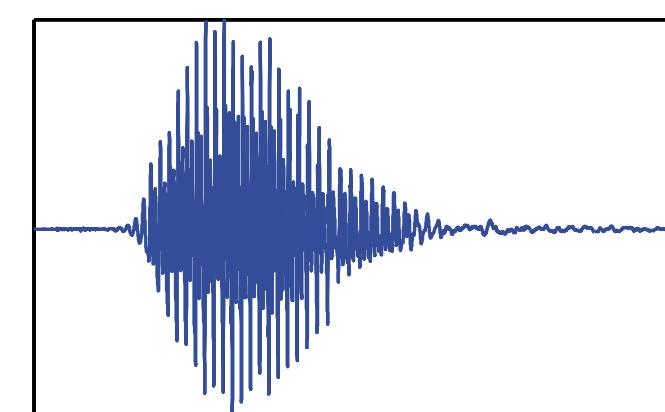
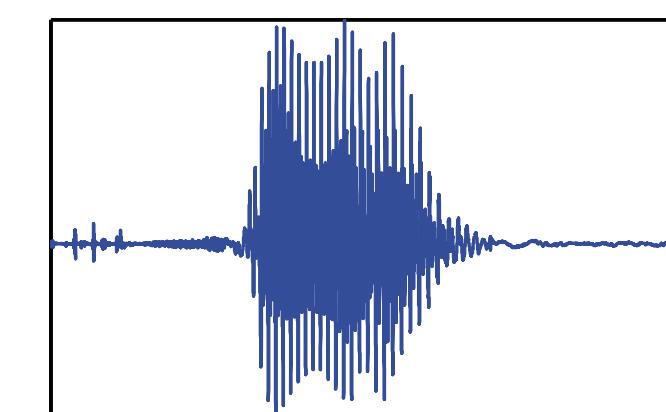
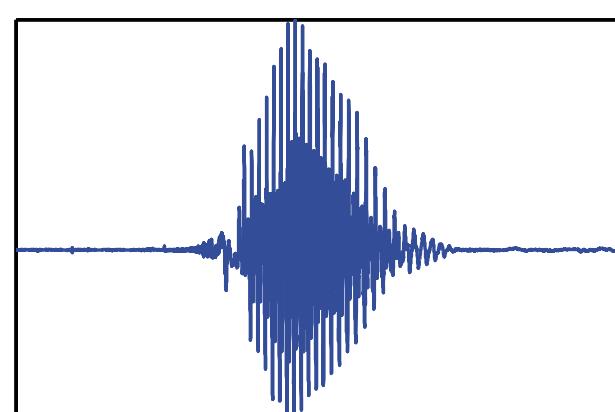
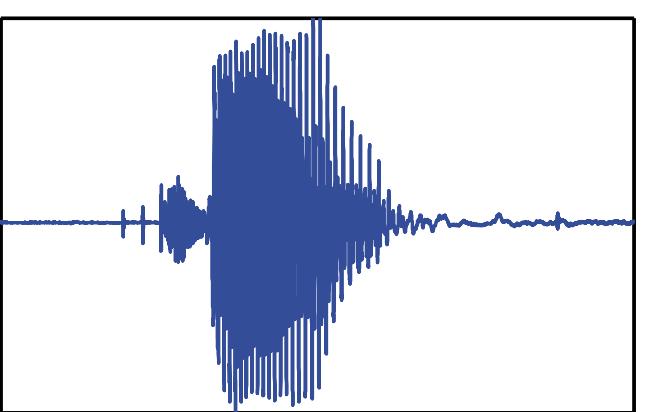
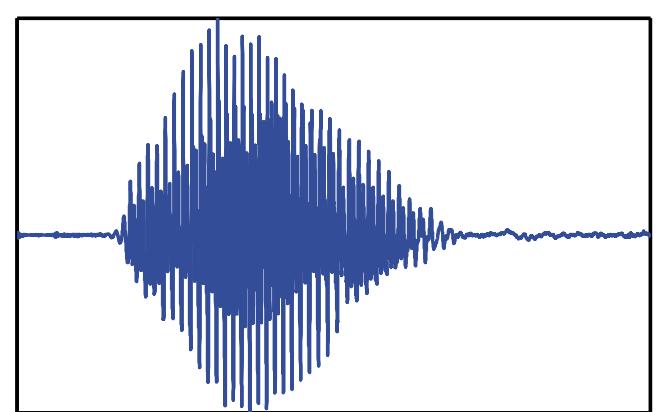
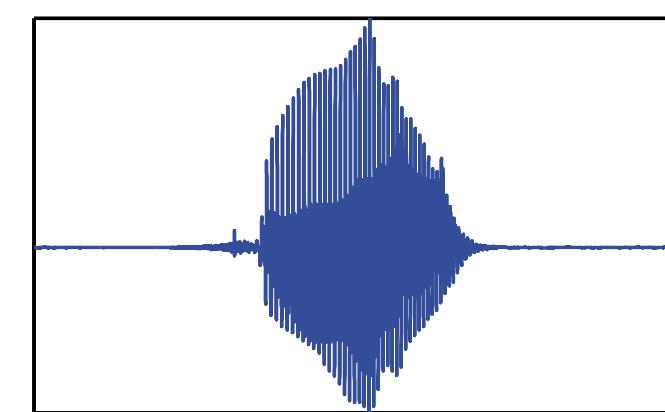
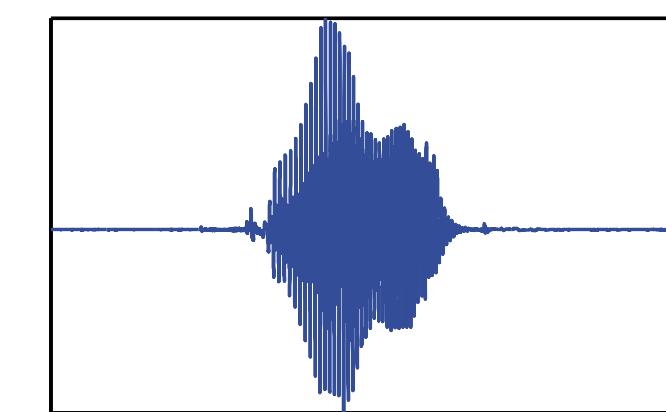
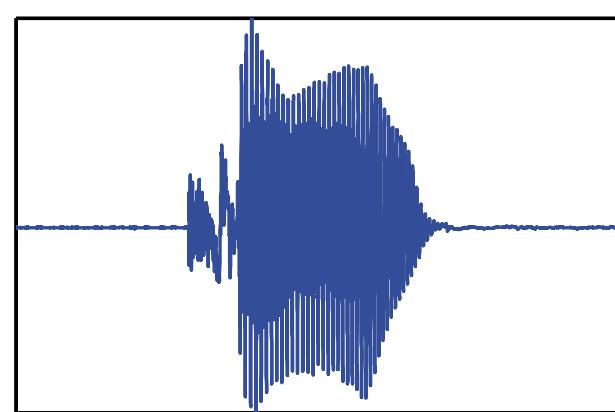
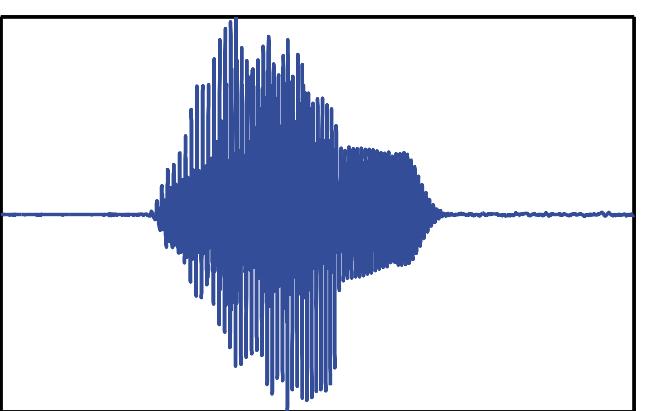
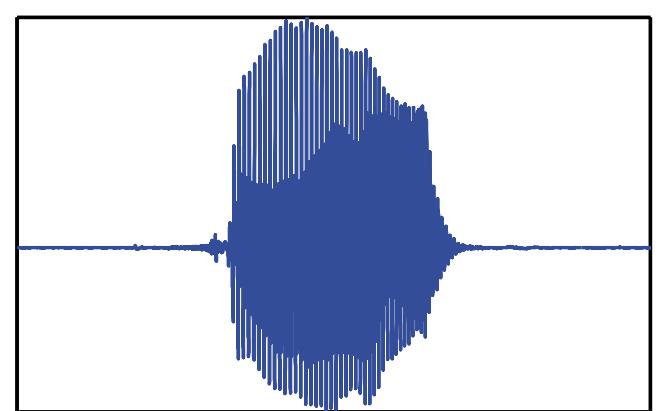
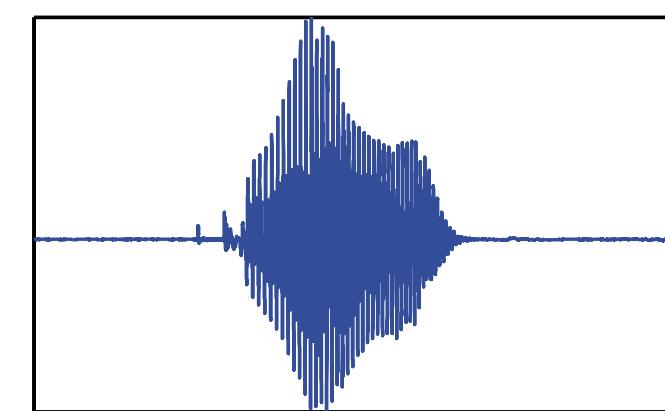
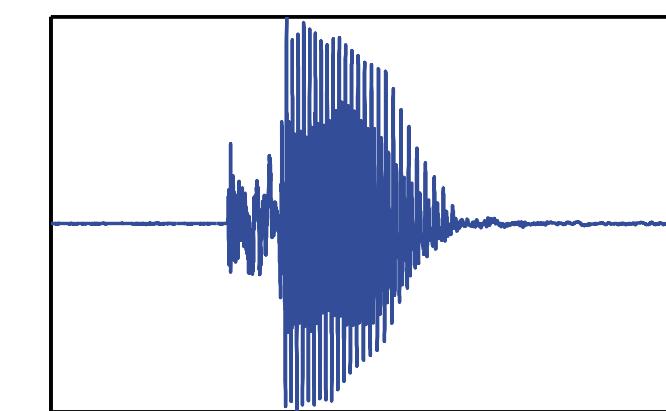
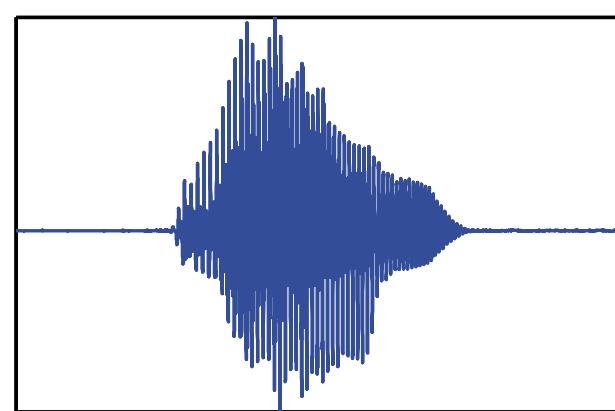
What now?

- Our models so far were too simple
- How do we incorporate time?
- How do we get around all these problems?

A small case study

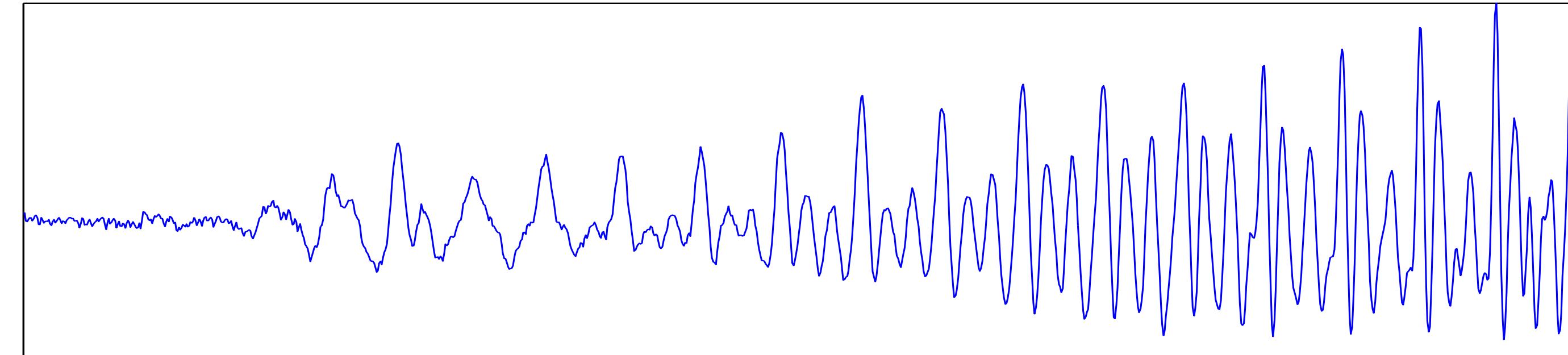
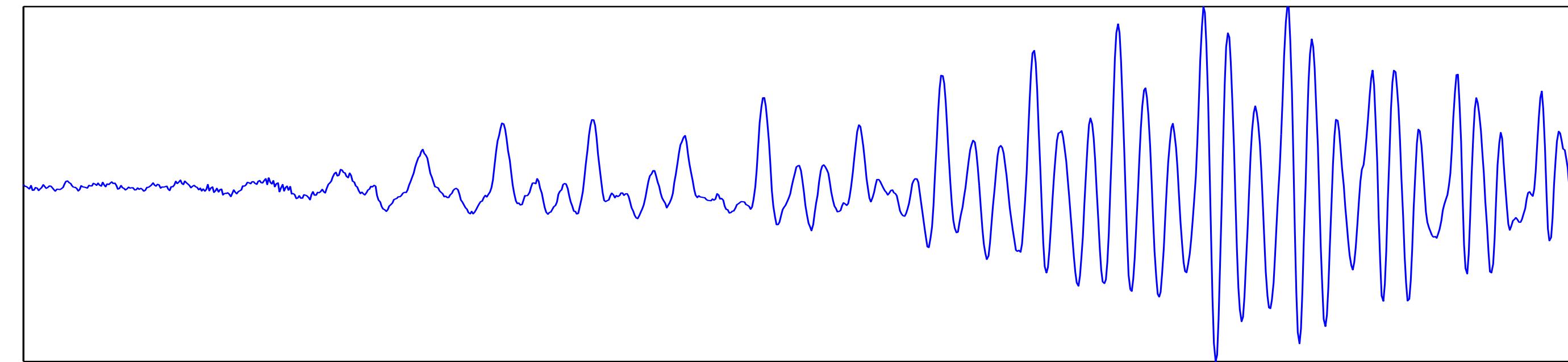
- How to recognize words
 - e.g. yes/no, or spoken digits
- Build reliable features
 - Invariant to minor differences in inputs
- Build a classifier that can deal with a time index
 - Invariant to temporal differences in inputs

Example data



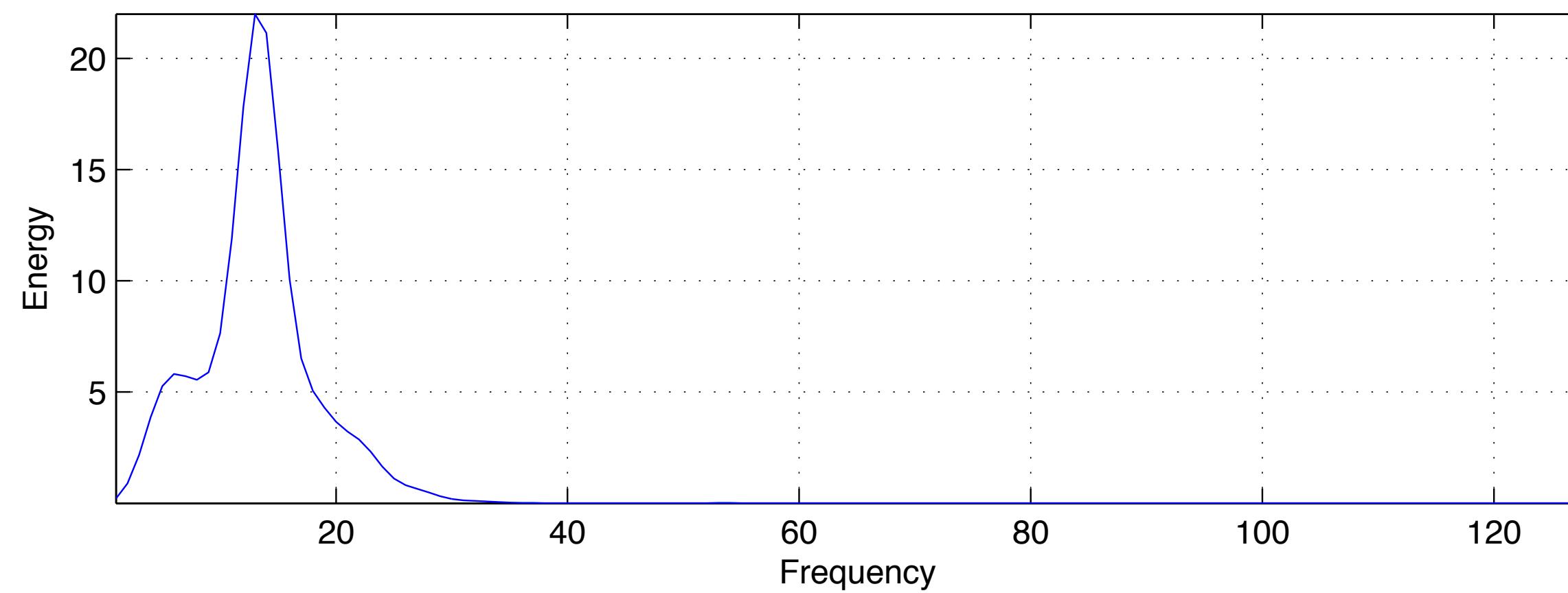
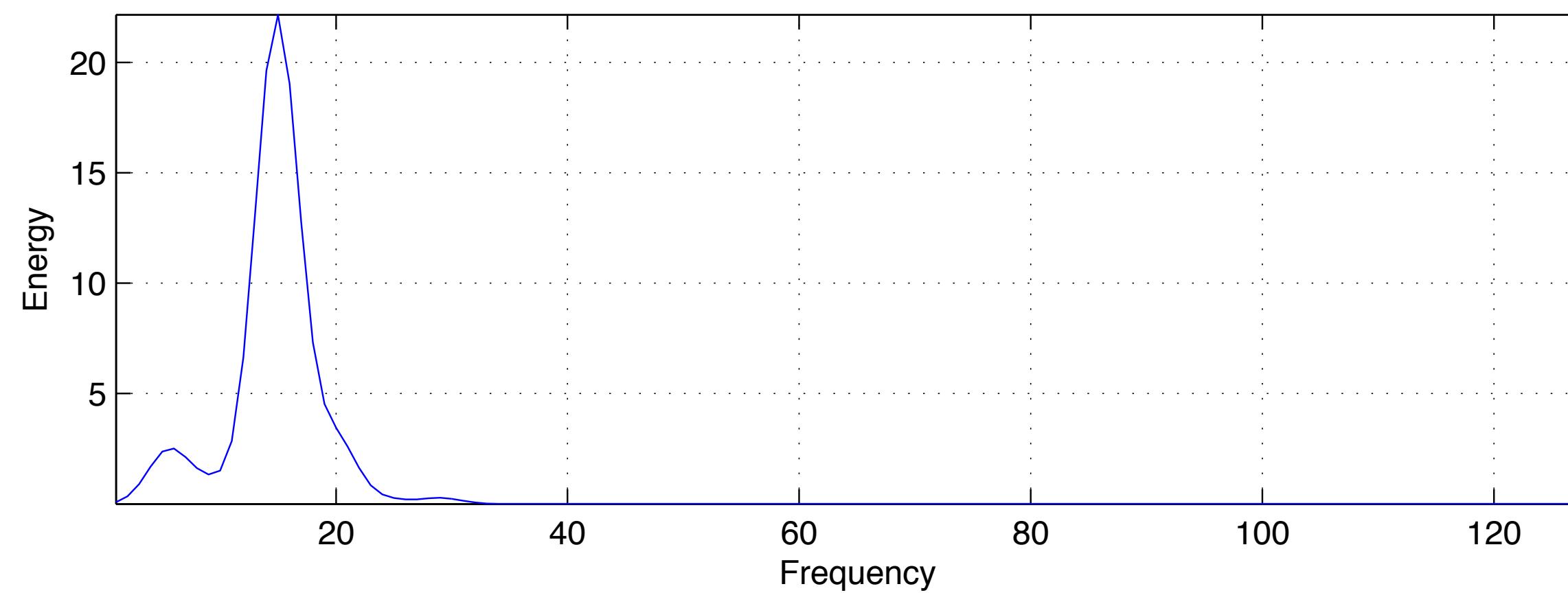
Going from fine to coarse

- Small differences are not important
 - Find features that obscure them



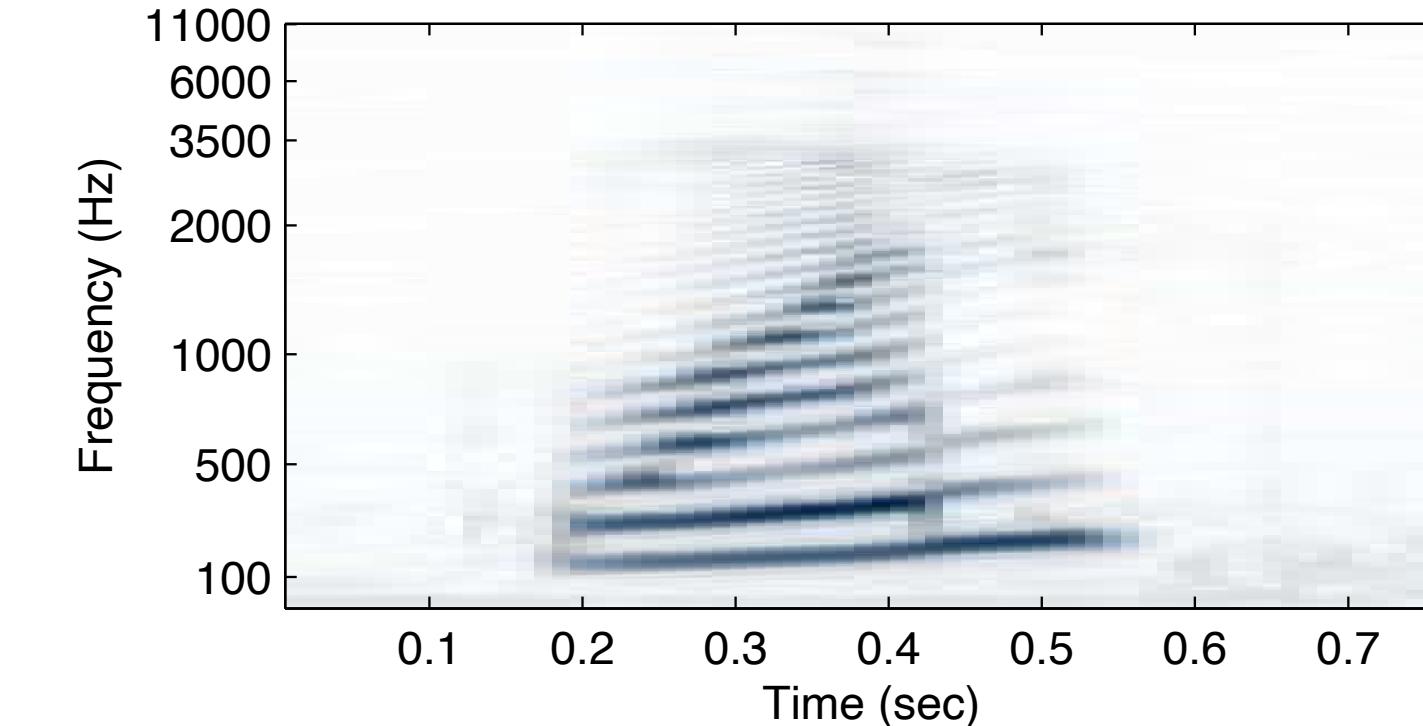
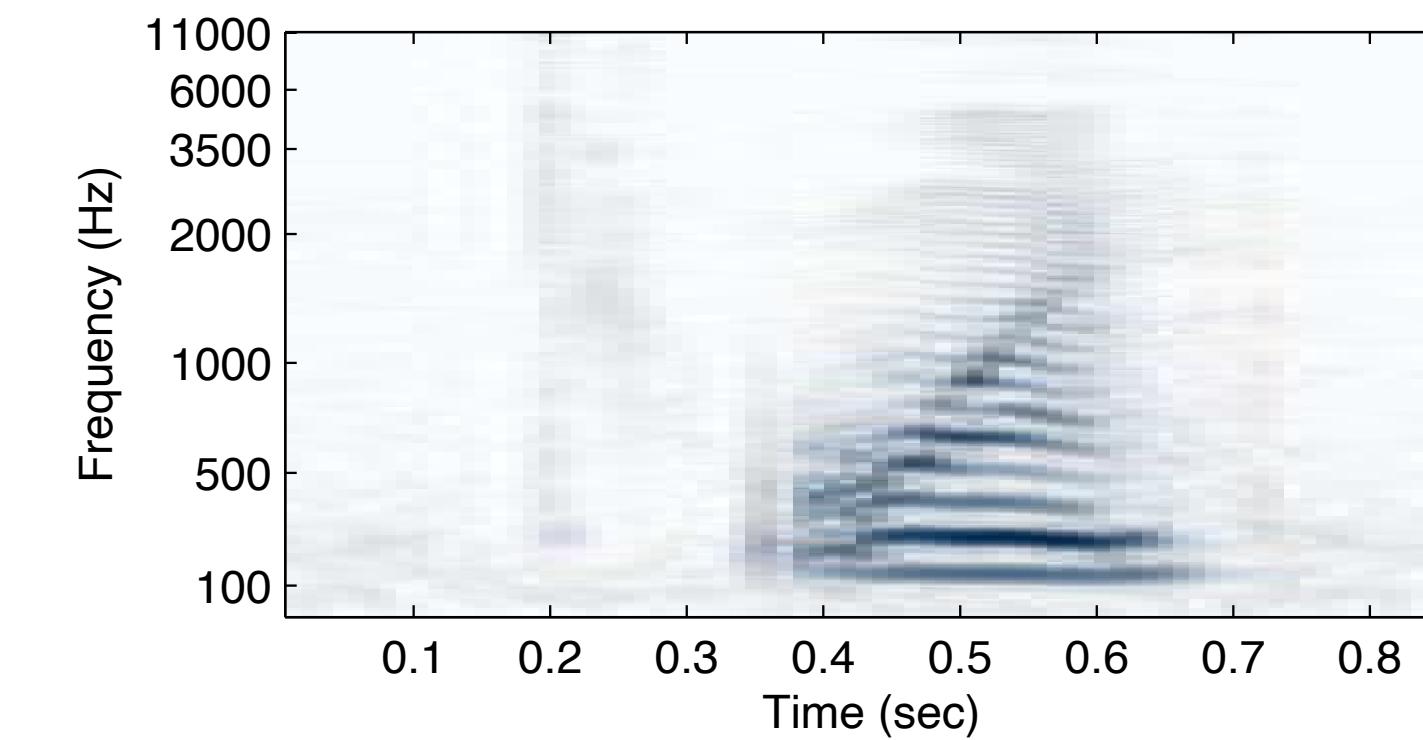
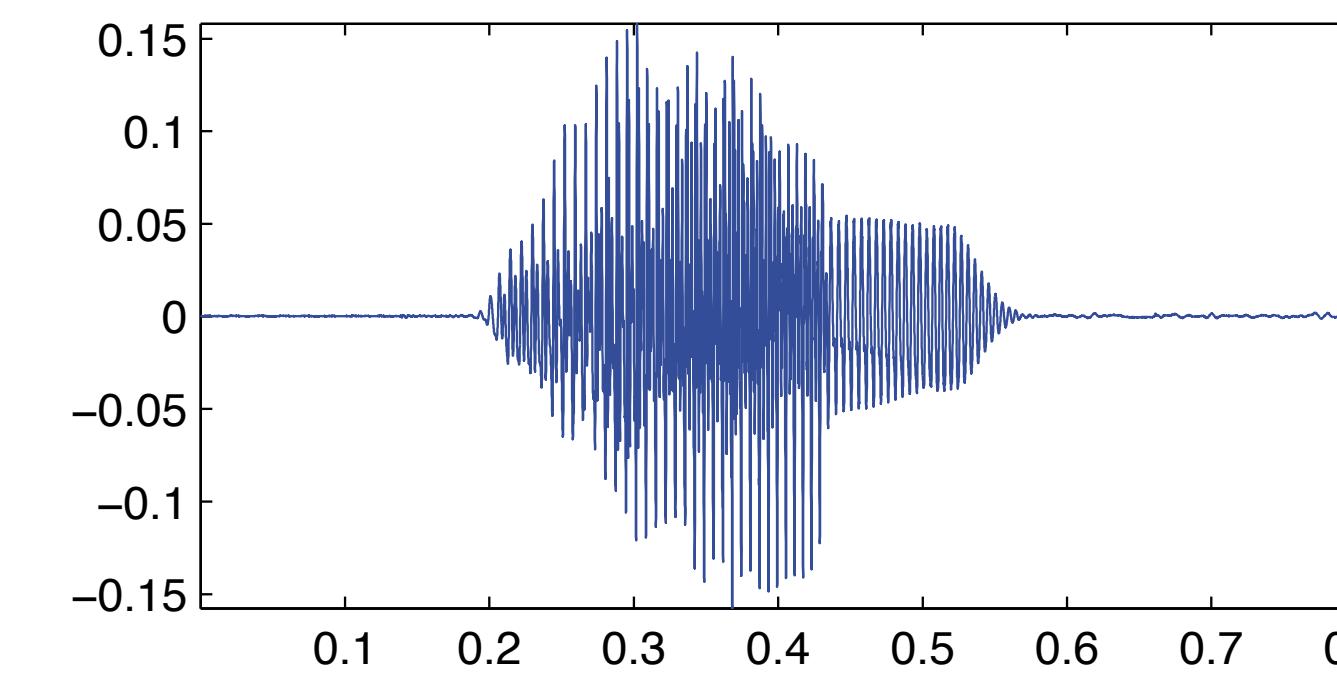
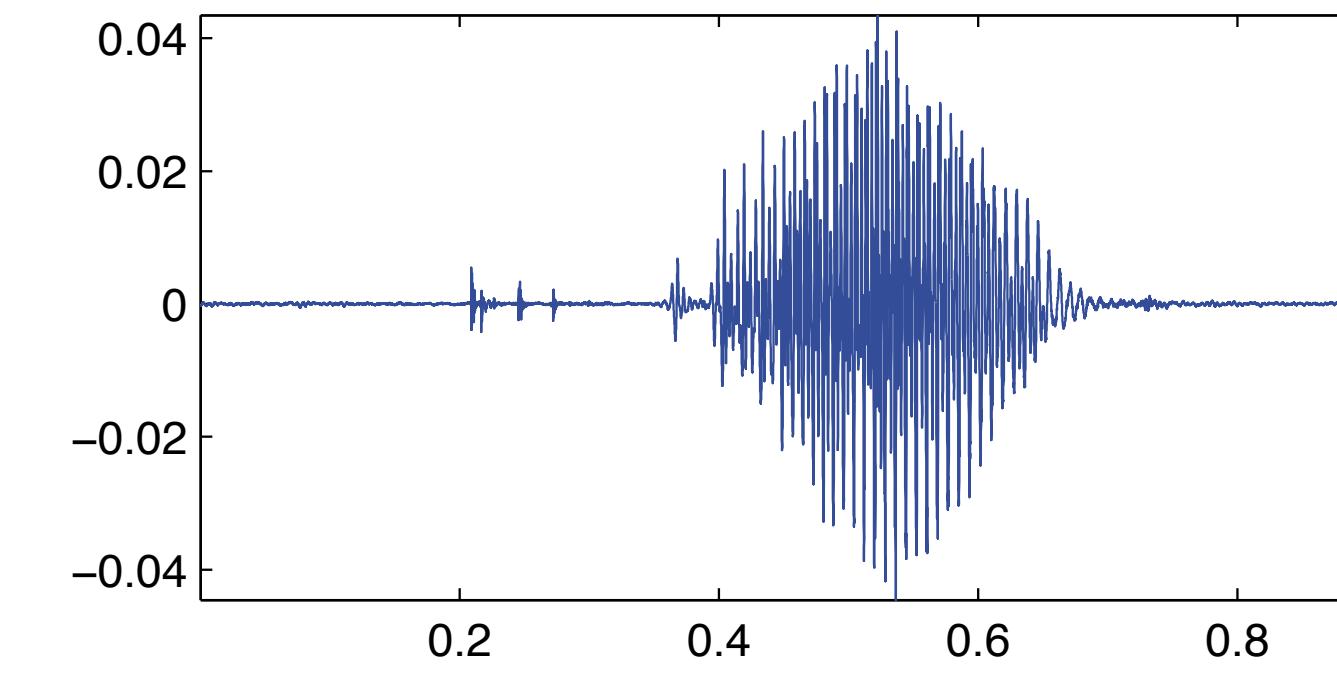
Frequency domain

- Look at the magnitude Fourier transform



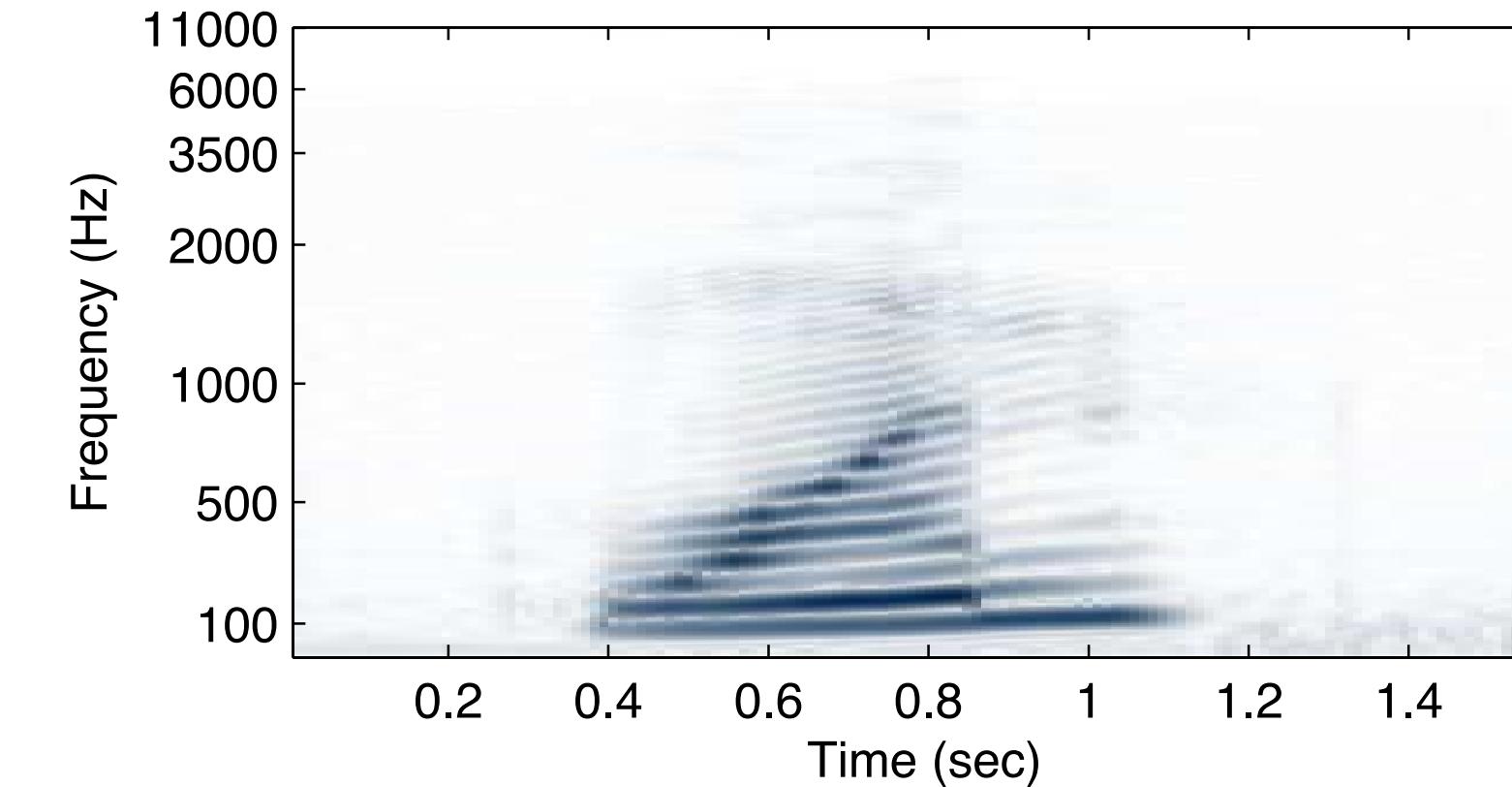
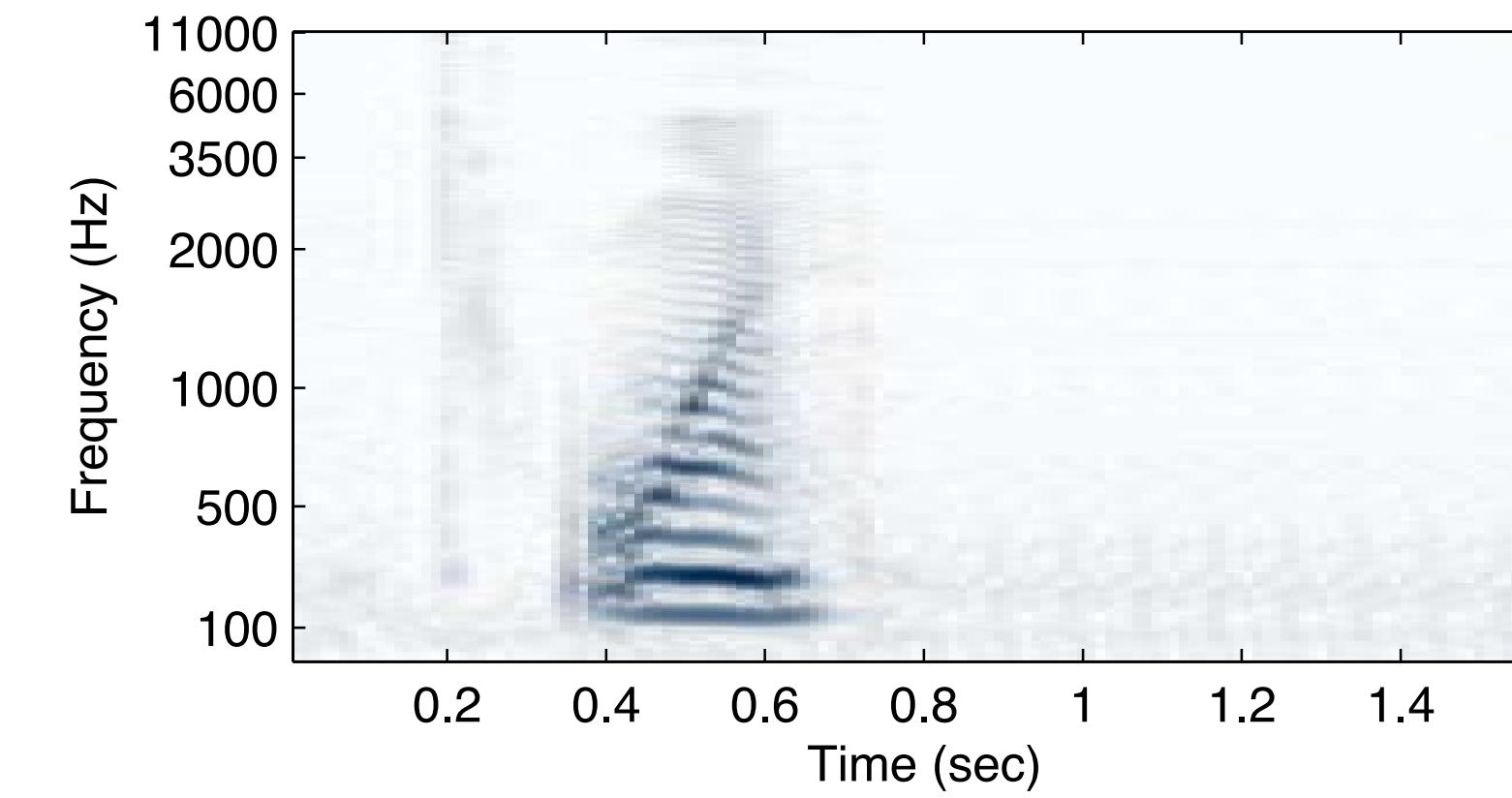
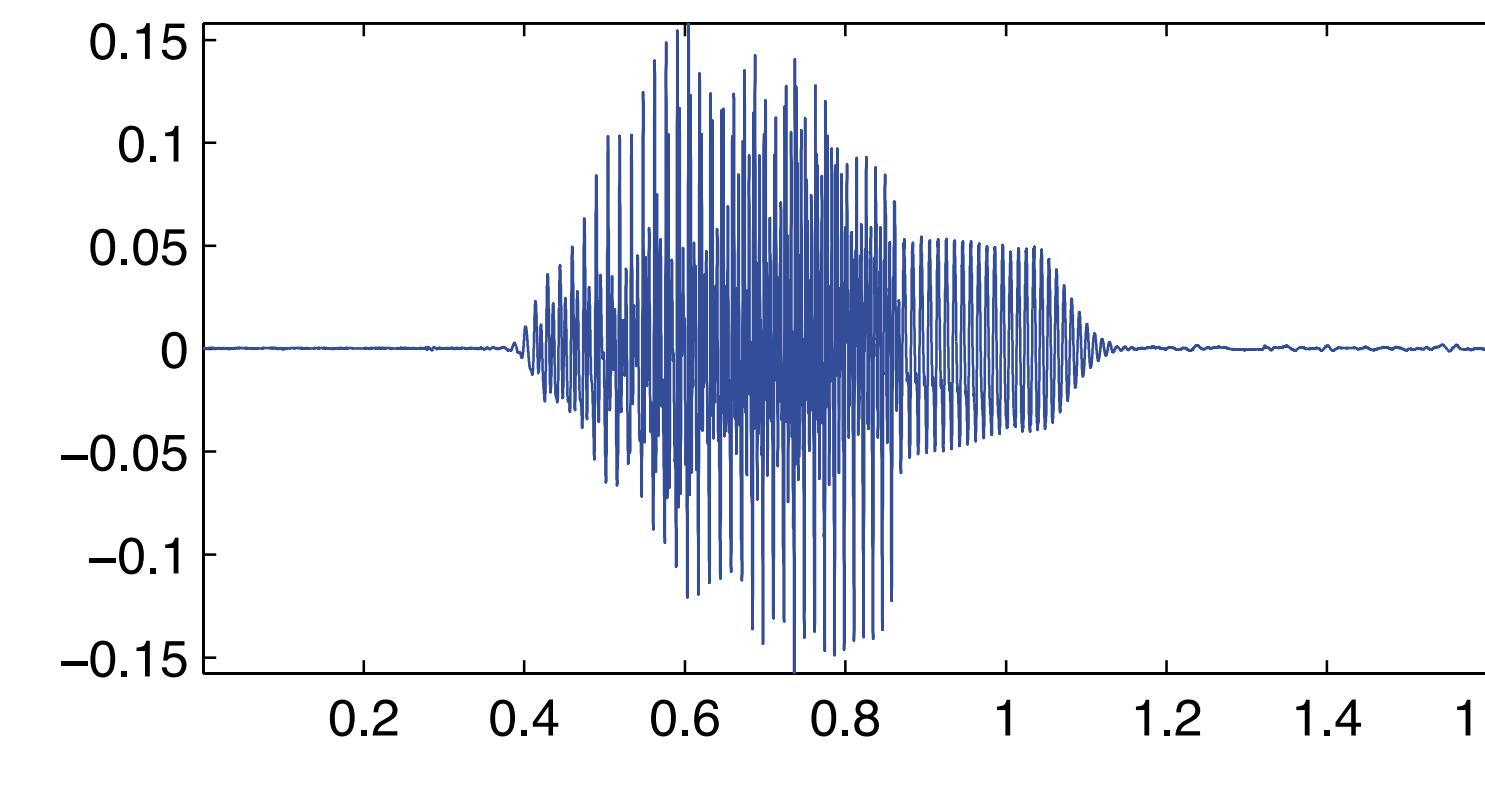
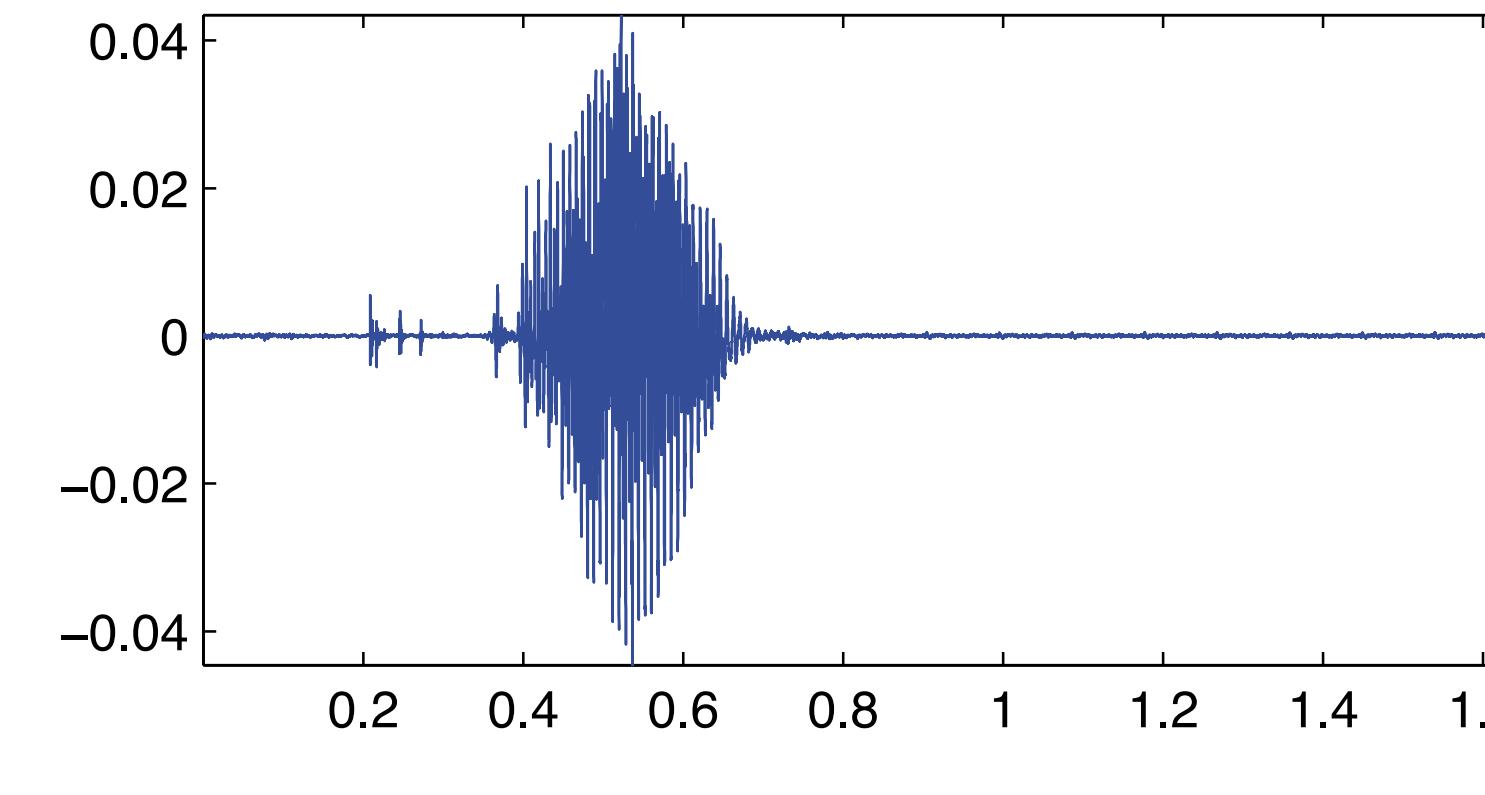
Time/Frequency features

- A more robust representation
 - Bypassing small waveform differences



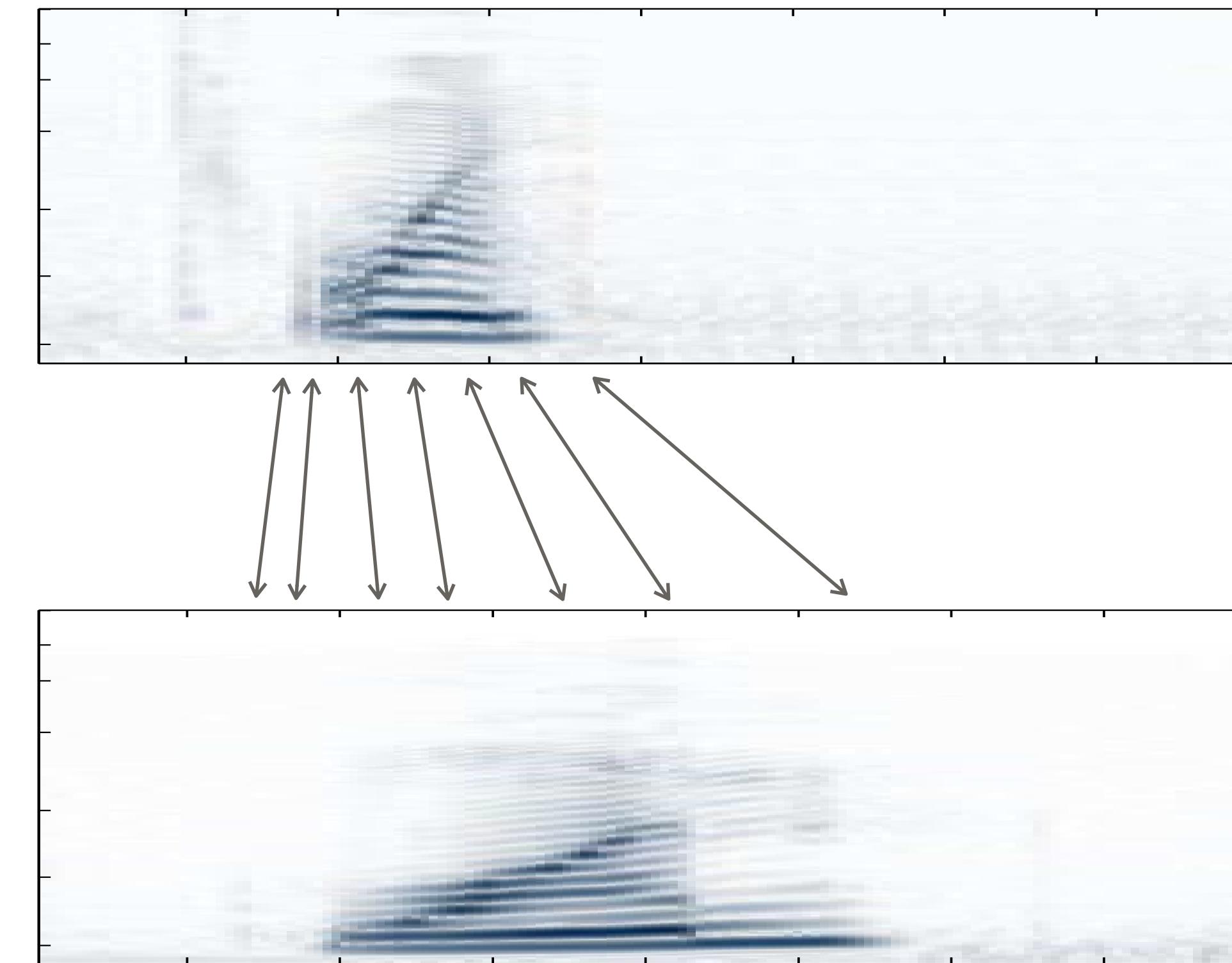
A new problem

- What about time warping?



Time warping

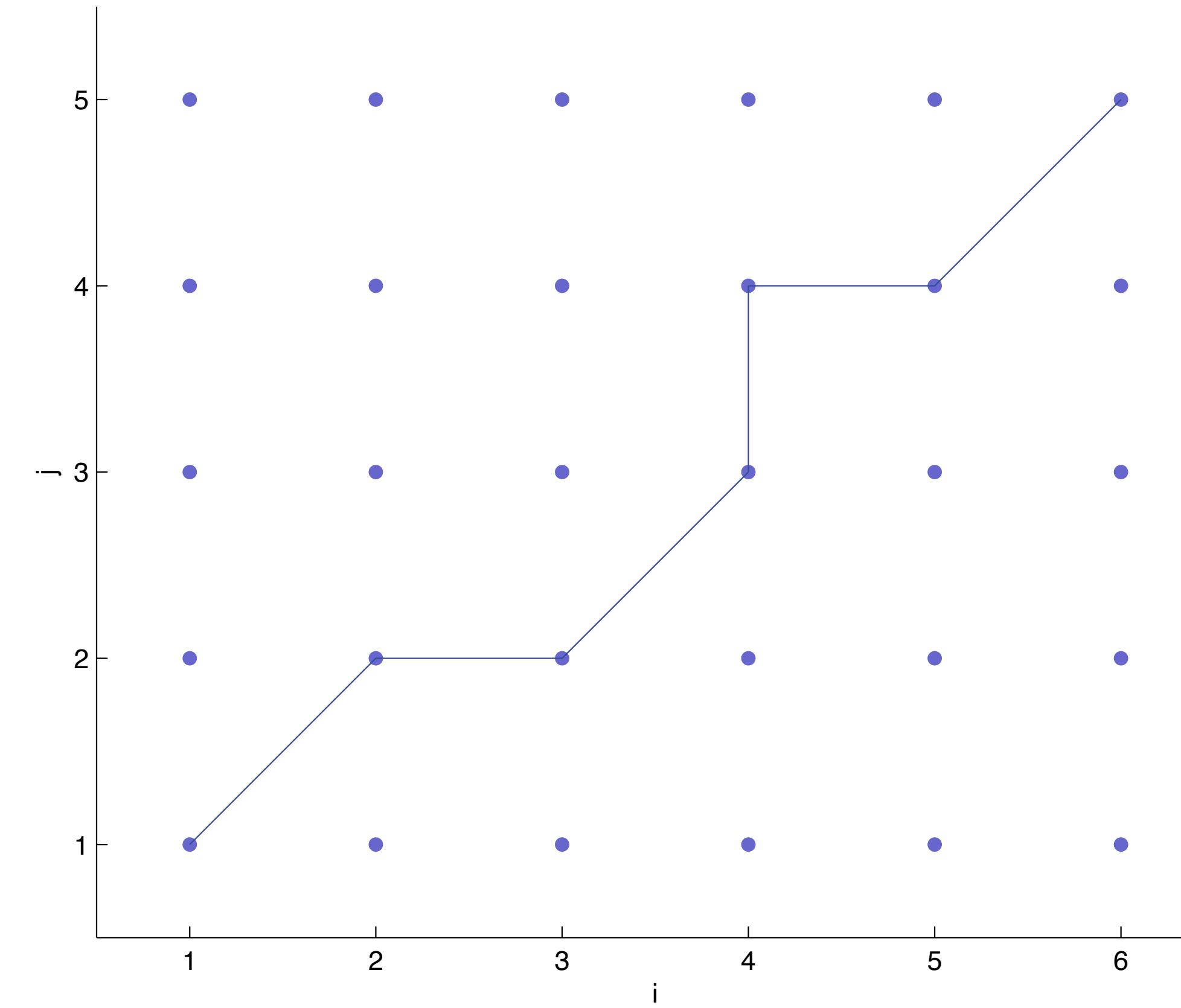
- There is a “warped” time map
 - How do we find it?



Matching warped series

- Represent the warping with a path on a grid

$$r(i), i = 1, 2, \dots, 6 \quad t(j), j = 1, 2, \dots, 5$$



Finding the overall “distance”

- Each node will have a cost

- e.g., $d(i, j) = \| \mathbf{r}(i) - \mathbf{t}(j) \|$

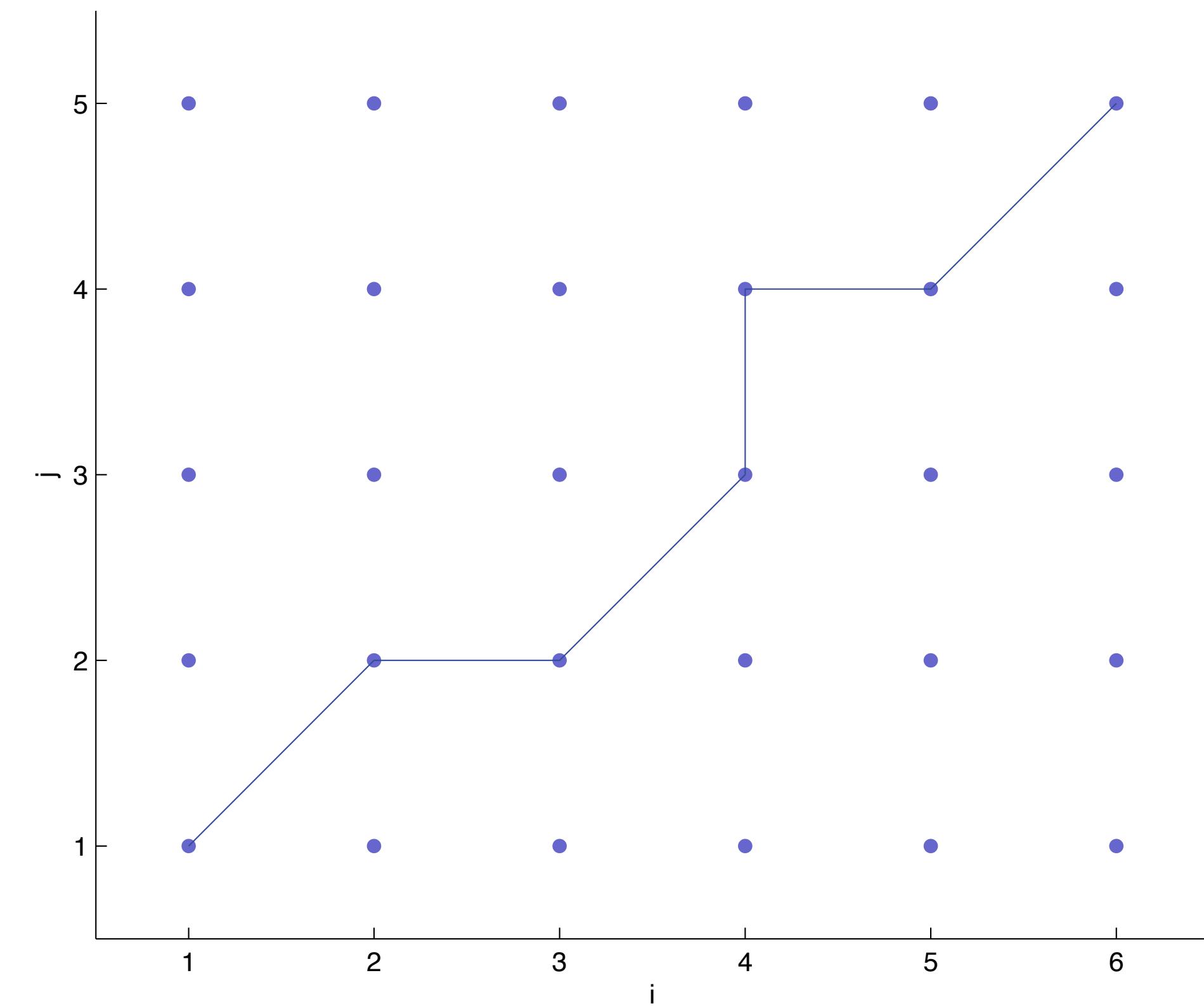
- Overall path cost is:

$$D = \sum_k d(i_k, j_k)$$

- Optimal path (i_k, j_k) defines the “distance” between two given sequences

- But how do we find the optimal path?

- Big search space ...



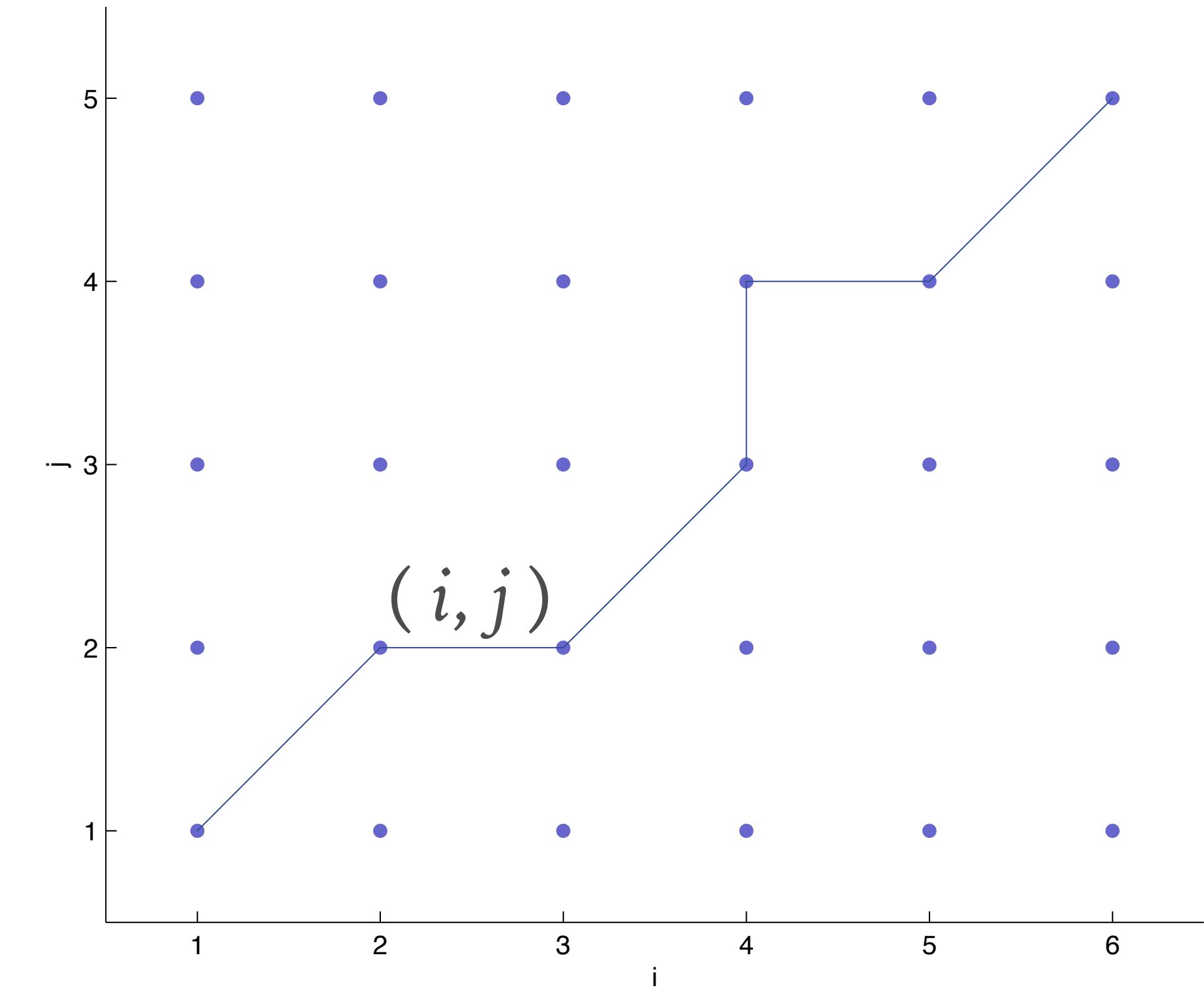
Bellman's optimality principle

- For an optimal path passing through

$$(i, j) : (i_0, j_0) \xrightarrow{opt} (i_f, j_f)$$

- Then:

$$(i_0, j_0) \xrightarrow{opt} (i_f, j_f) = \left\{ (i_0, j_0) \xrightarrow{opt} (i, j), (i, j) \xrightarrow{opt} (i_f, j_f) \right\}$$

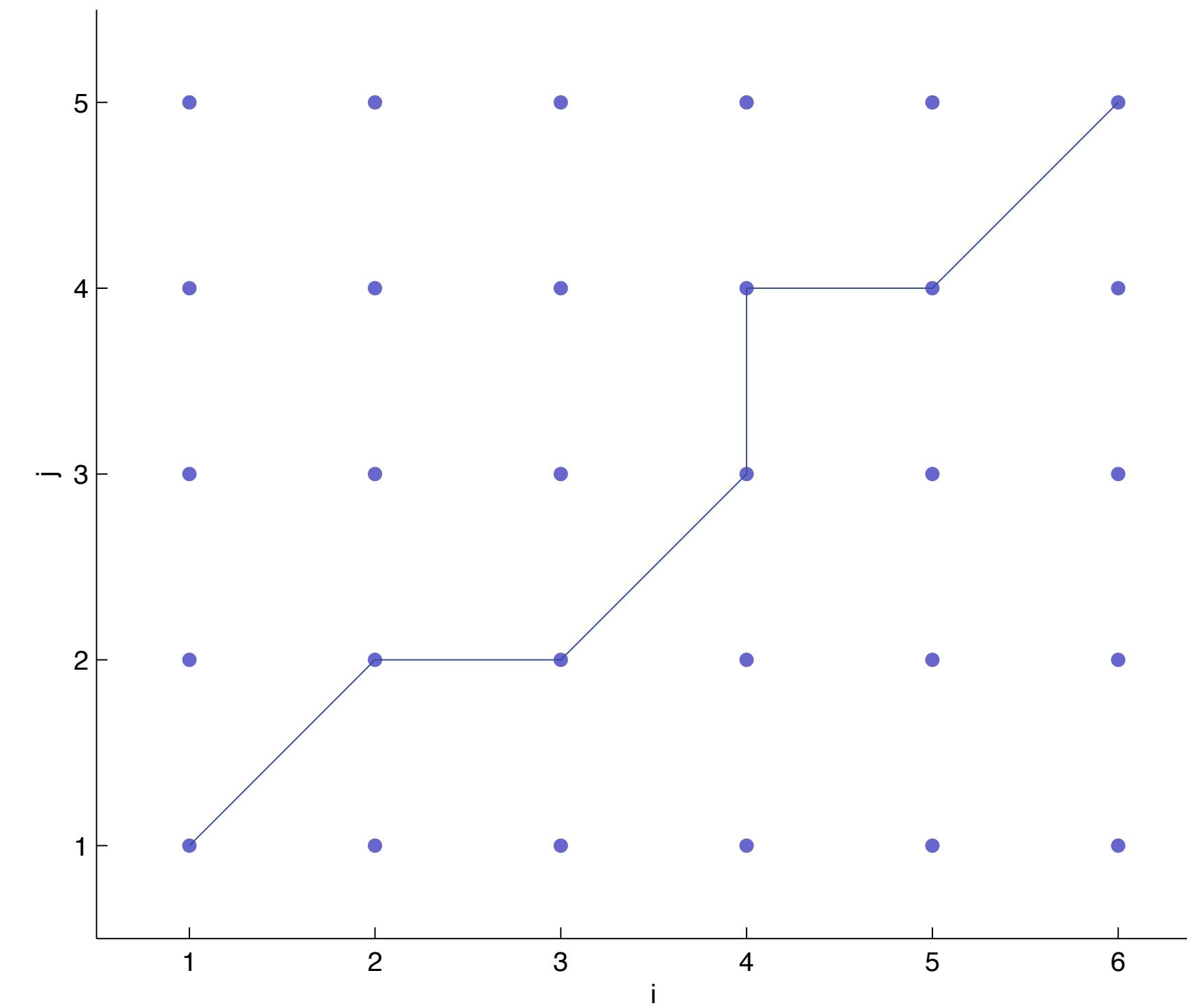


Finding an optimal path

- Optimal path to (i_k, j_k) :

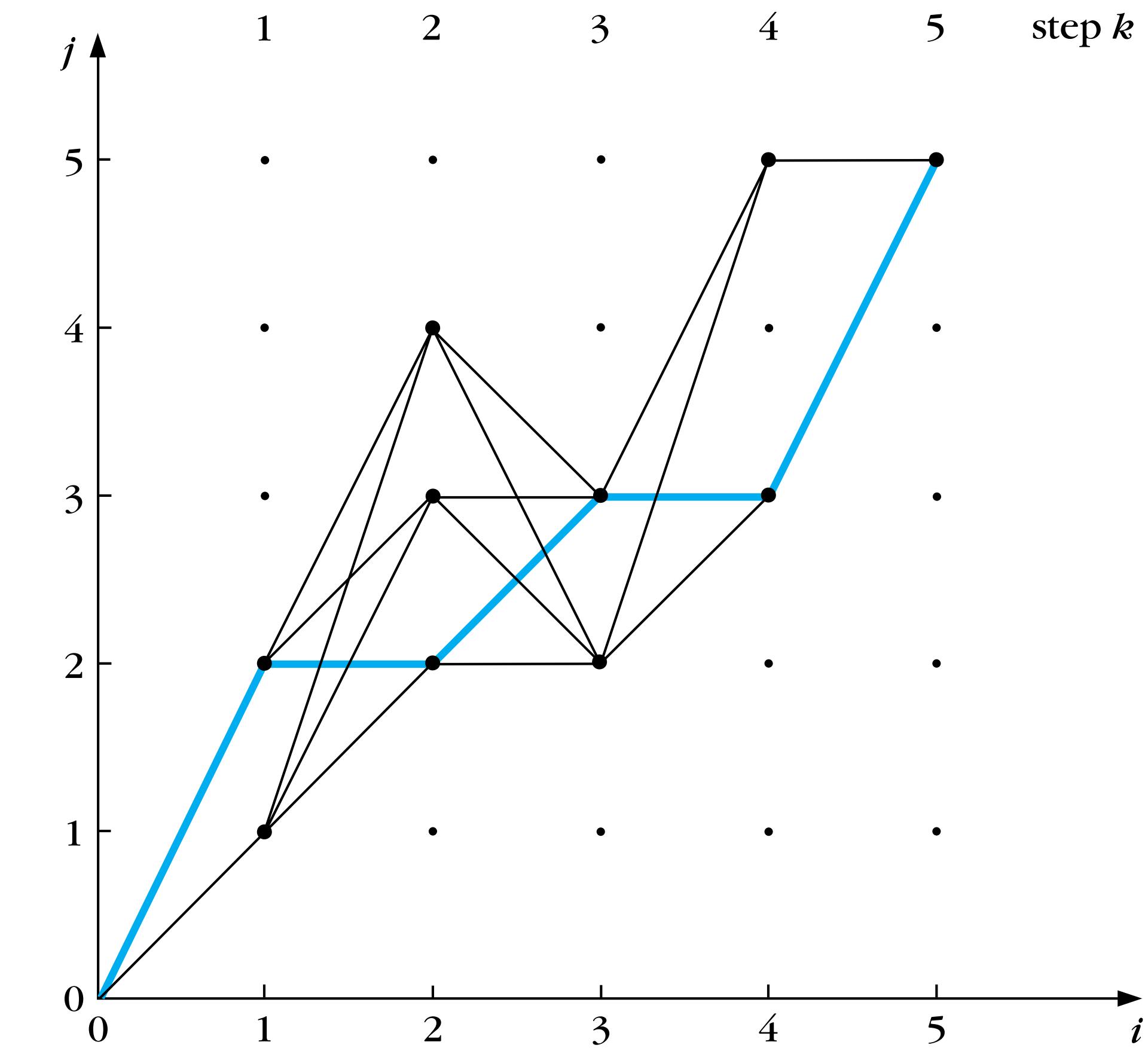
$$D_{\min}(i_k, j_k) = \min_{i_{k-1}, j_{k-1}} D_{\min}(i_{k-1}, j_{k-1}) + d(i_k, j_k | i_{k-1}, j_{k-1})$$

- Smaller search!
- Local/global constraints
 - Limited transitions
 - Nodes we never visit



And adding some constraints

- Global constraints
 - Can only visit these
 - Bold dots
- Local constraints
 - Can only transition using them
 - Black lines
- Optimal path
 - Blue line

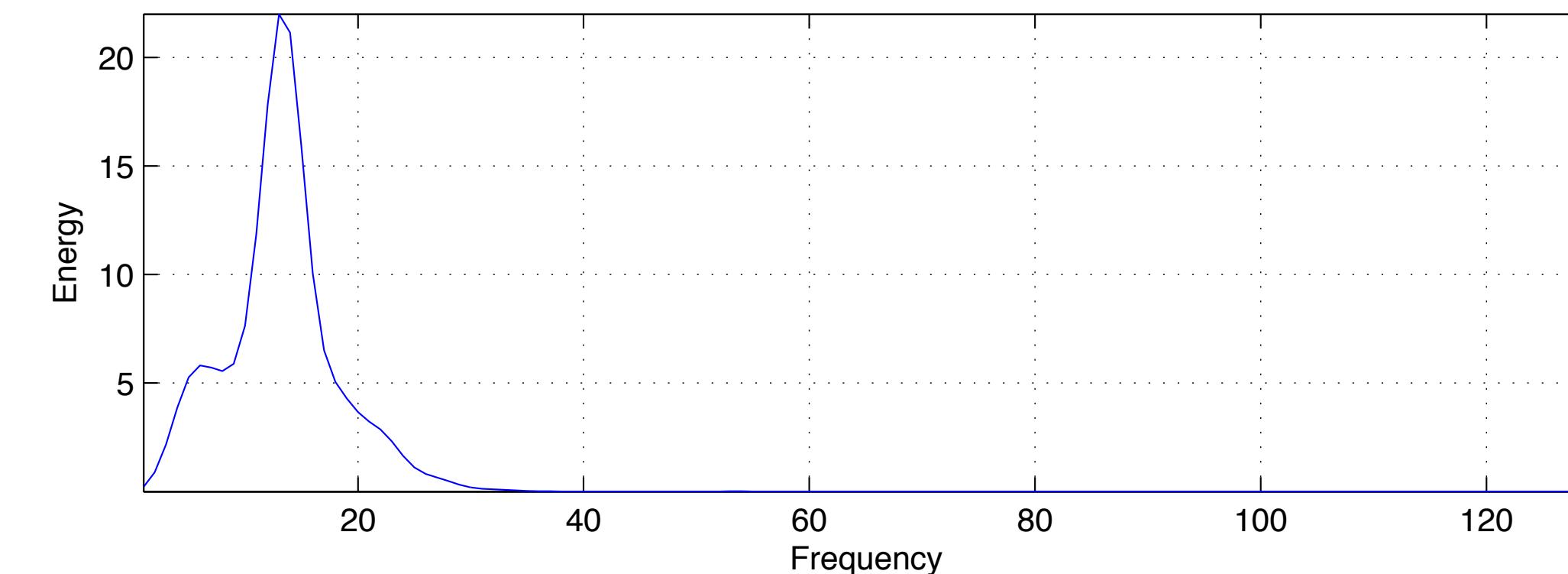
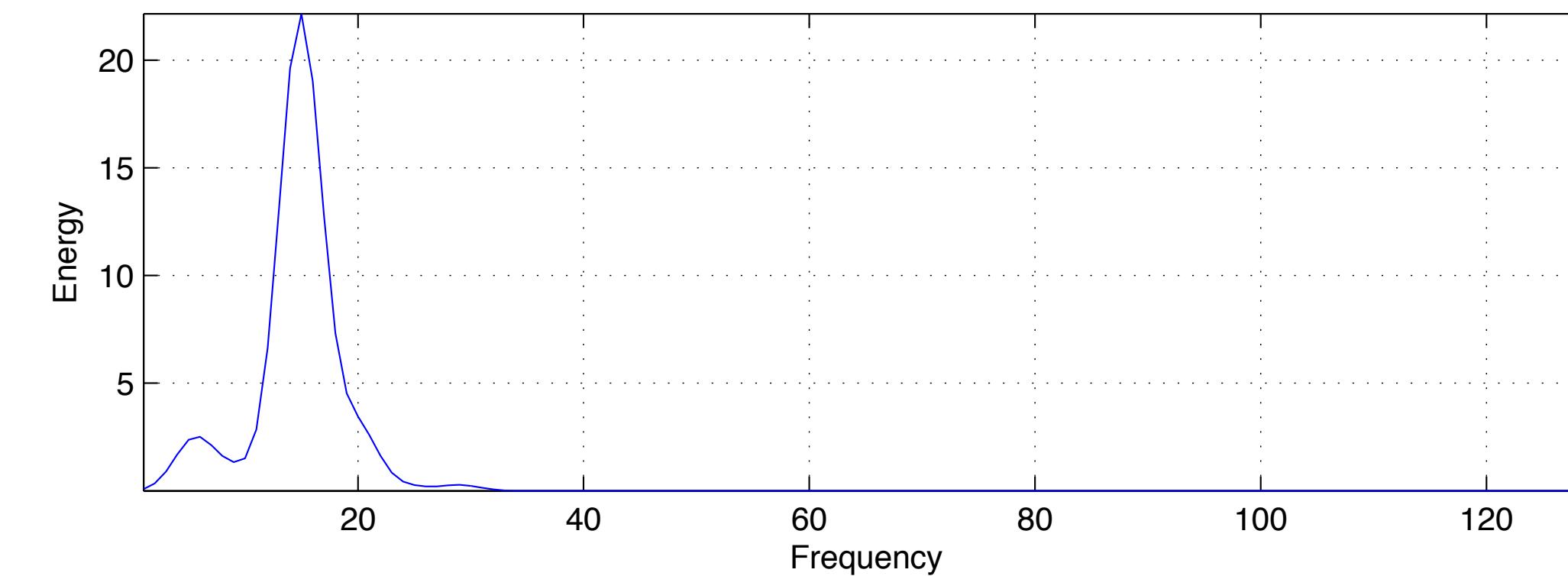


Making this work for speech recognition

- Define a distance function
- Define local constraints
- Define global constraints

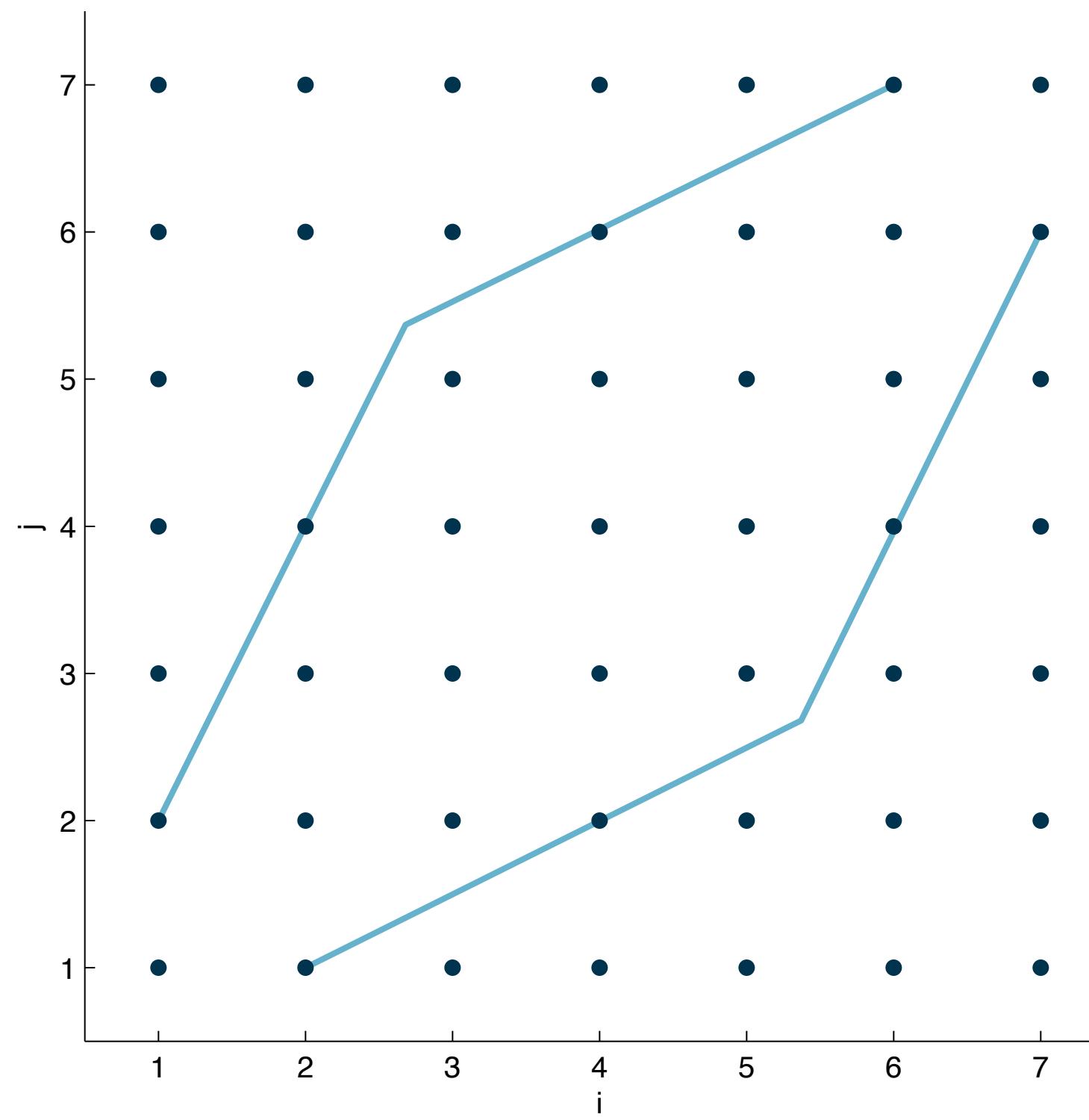
Distance function

- Given our robust feature we can use a simple measure like Euclidean distance: $d(i, j) = \|\mathbf{f}_1(i) - \mathbf{f}_2(j)\|$



Global constraints

- Define time ratios that make sense
 - e.g. no sequence can be half as slow as the other one



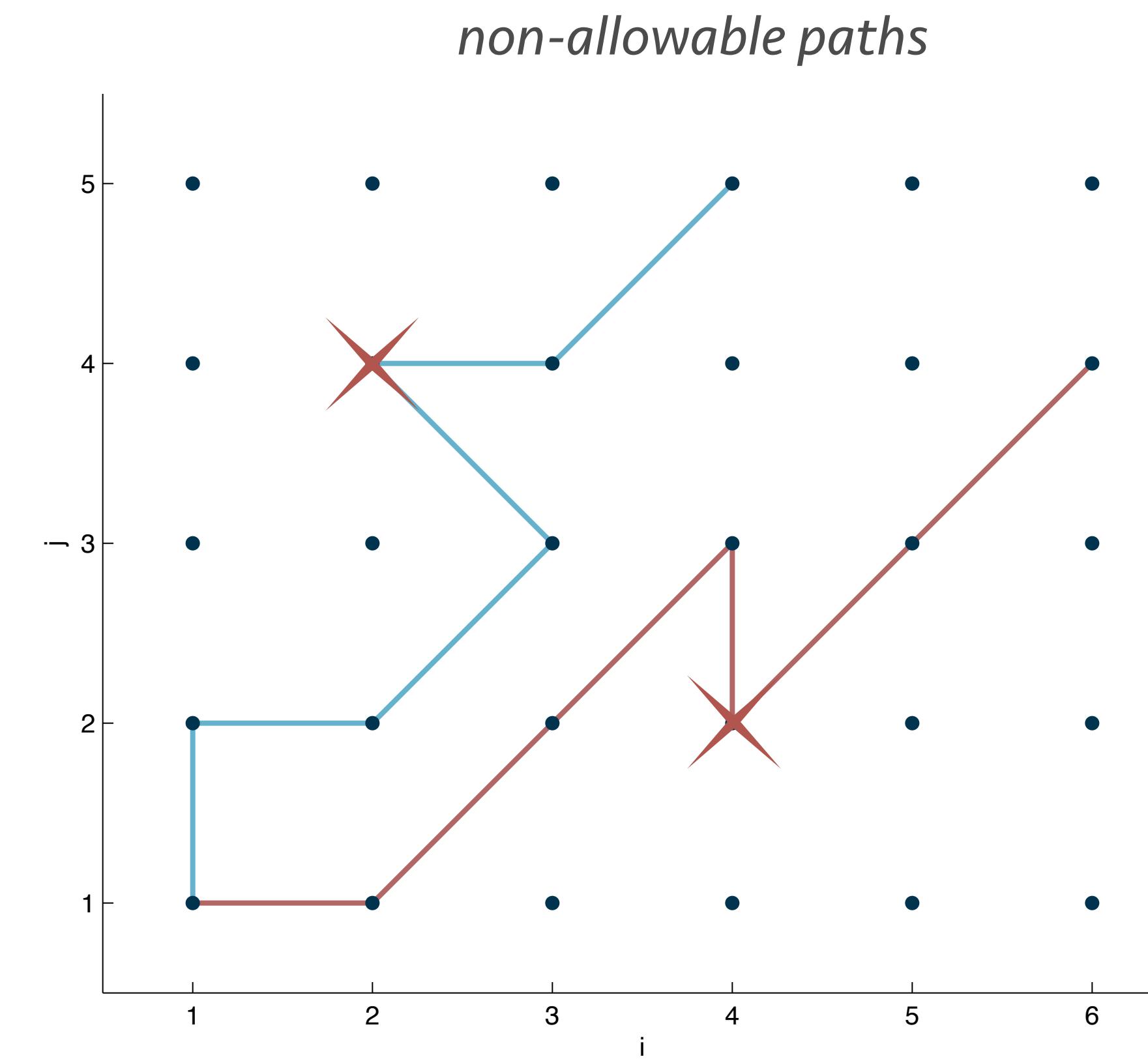
Local constraints

- Monotonicity:

$$i_{k-1} \leq i_k \quad j_{k-1} \leq j_k$$

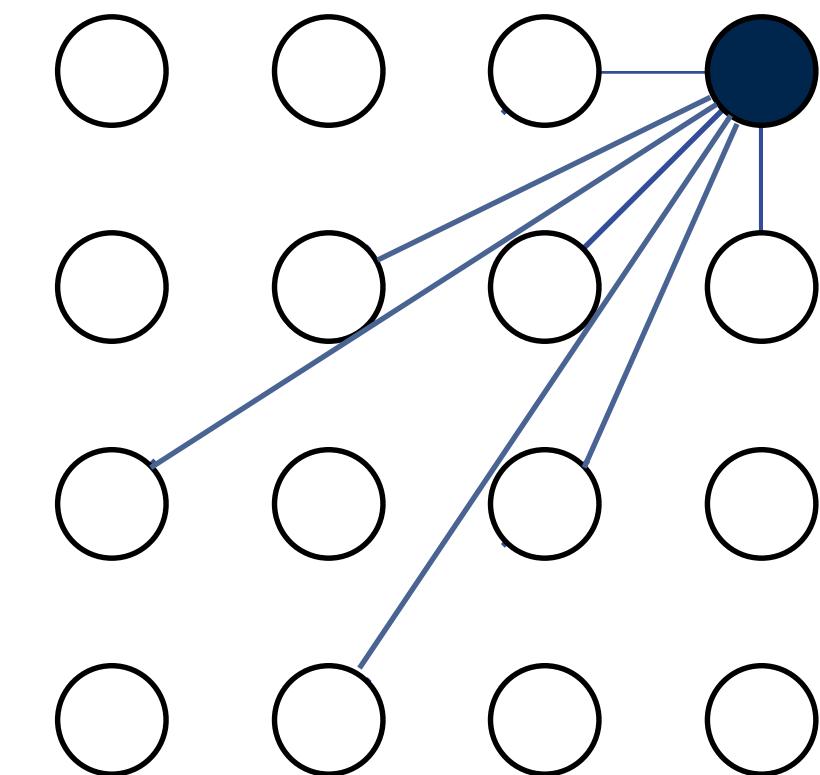
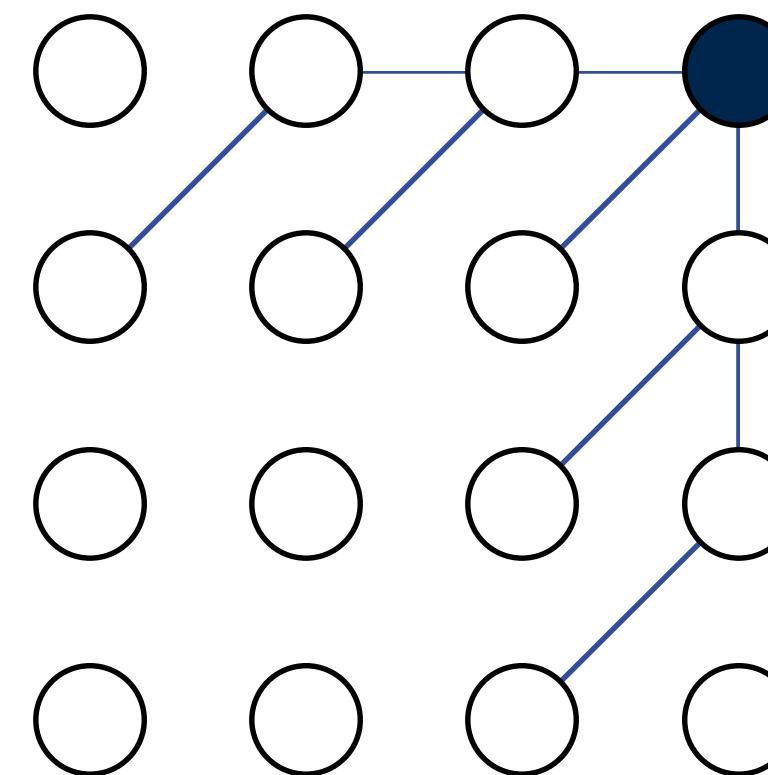
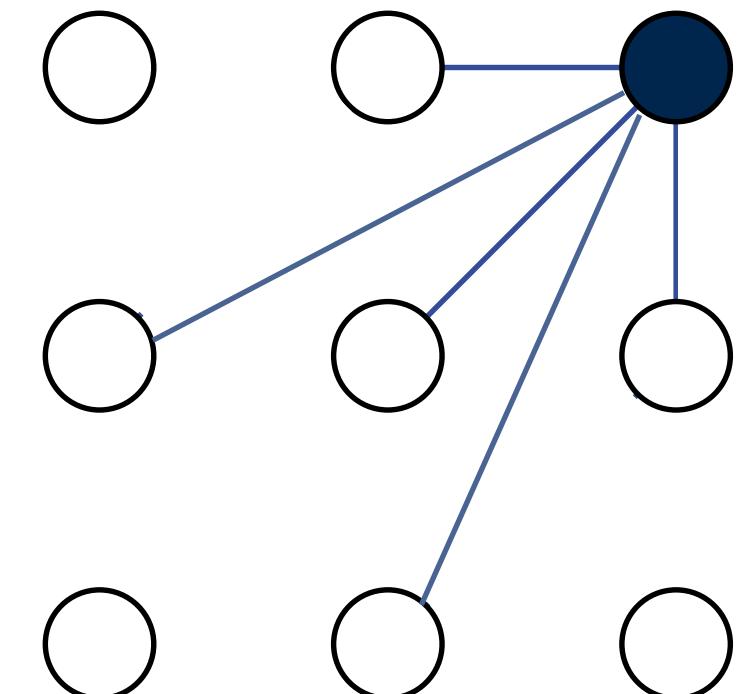
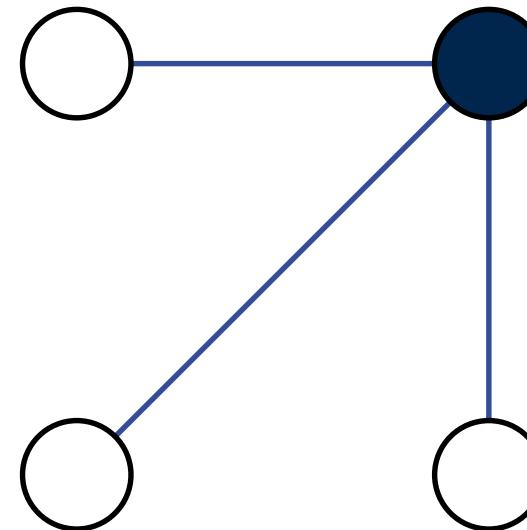
- Repeat, but don't go backwards
in time

- This enforces time order
 - Don't match "ban" with "banana"

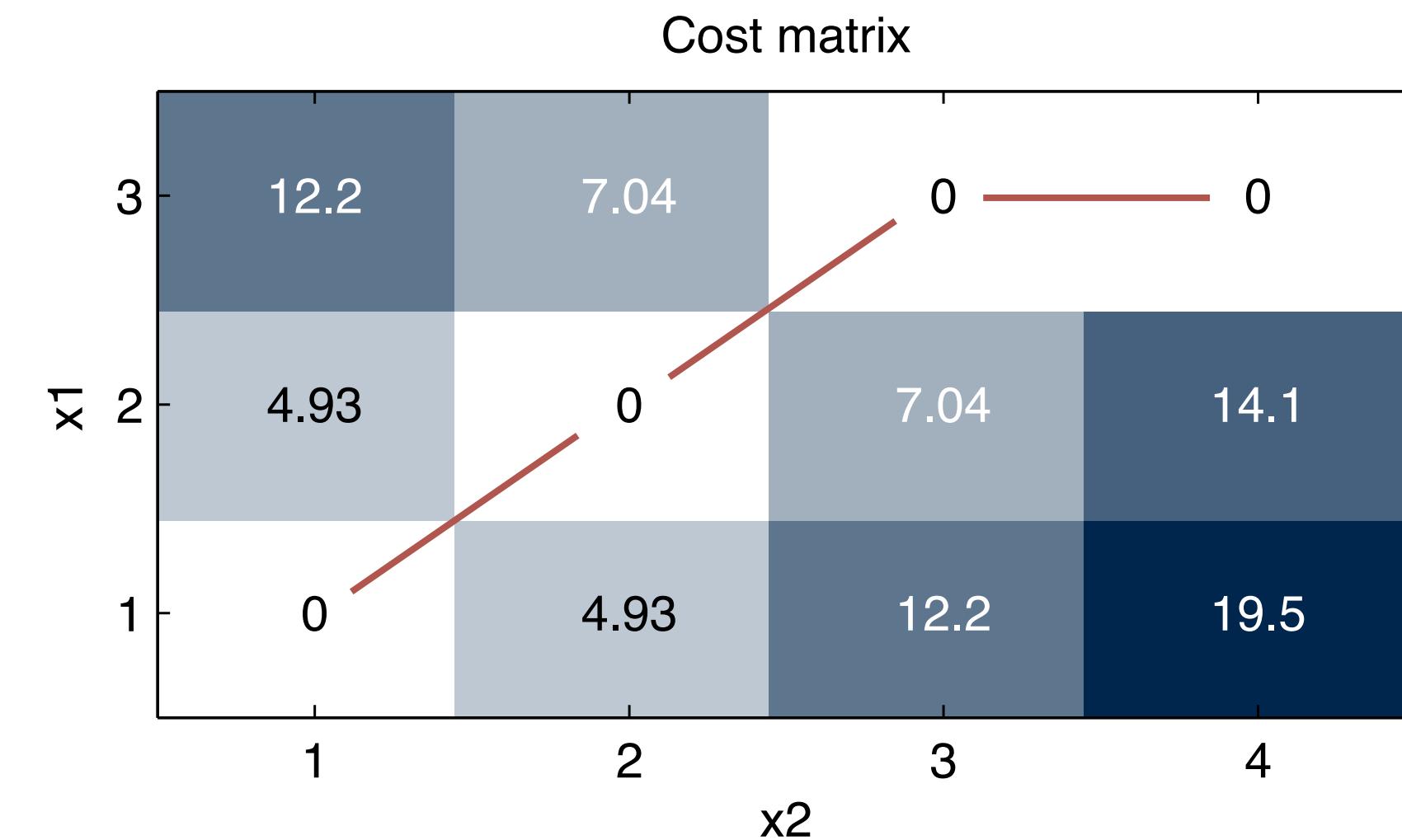
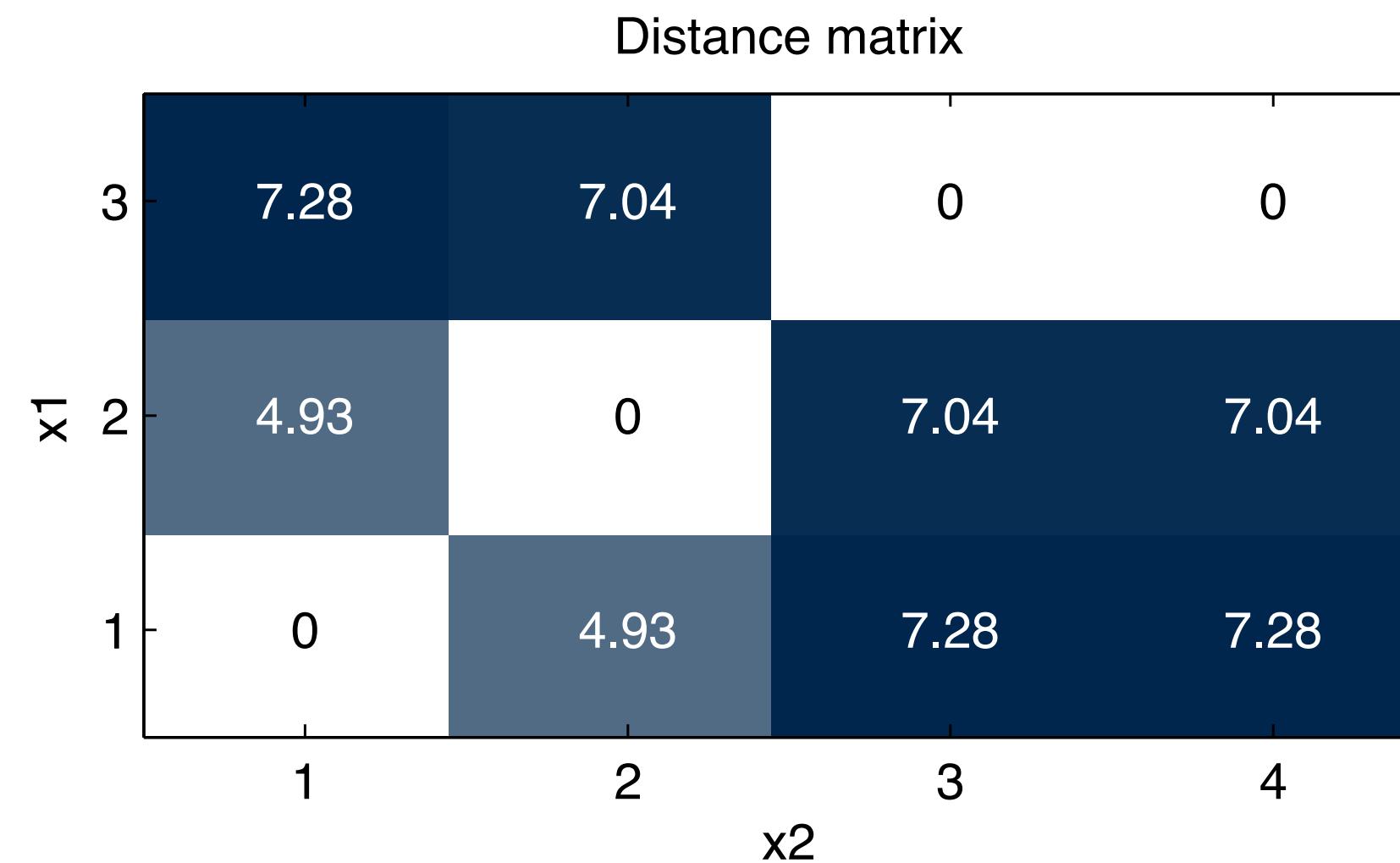
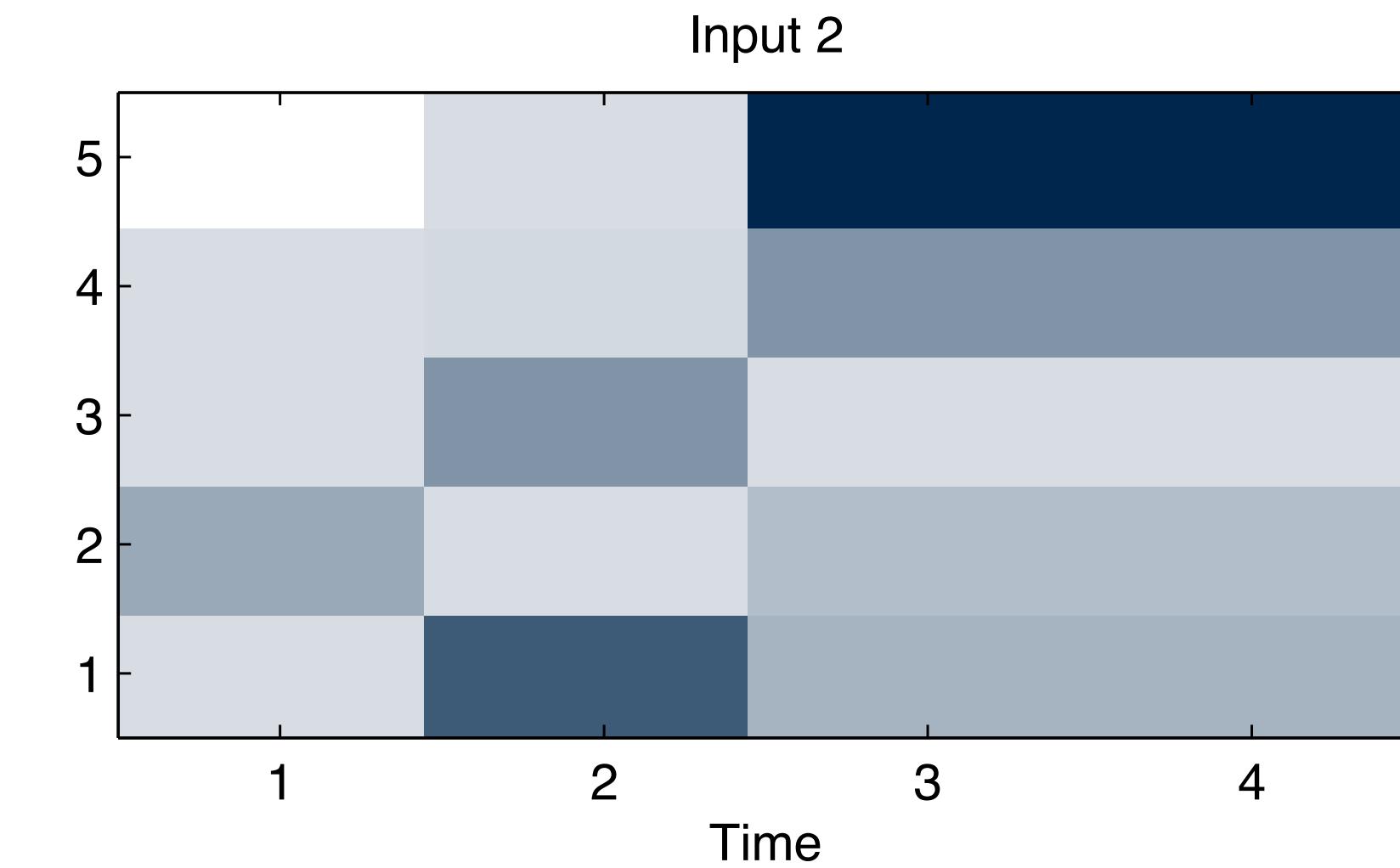
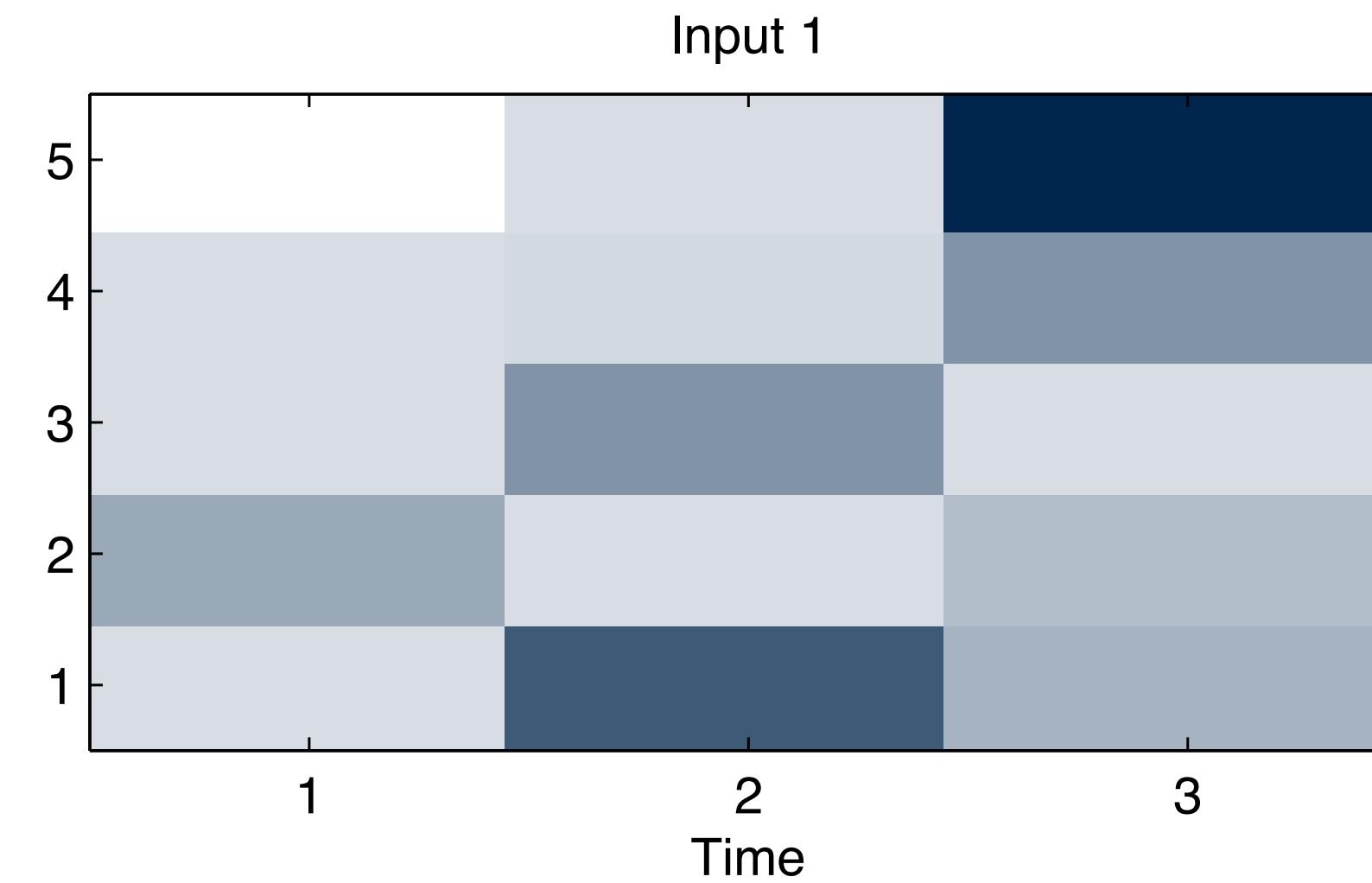
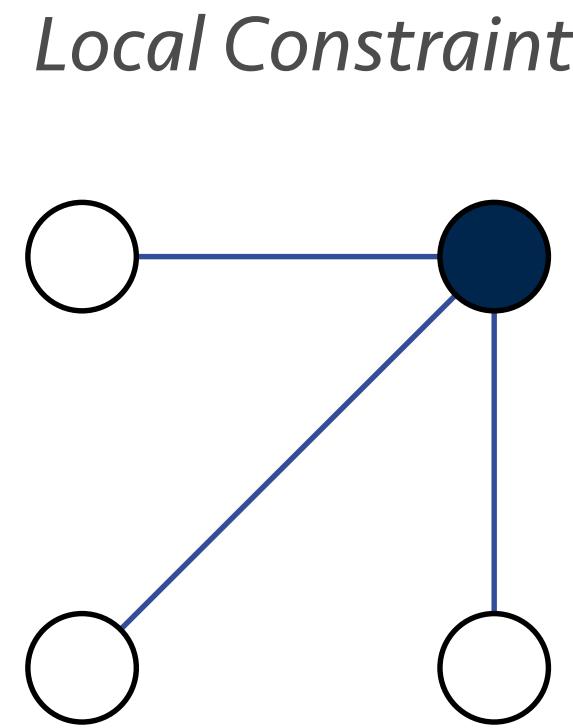


More local constraints

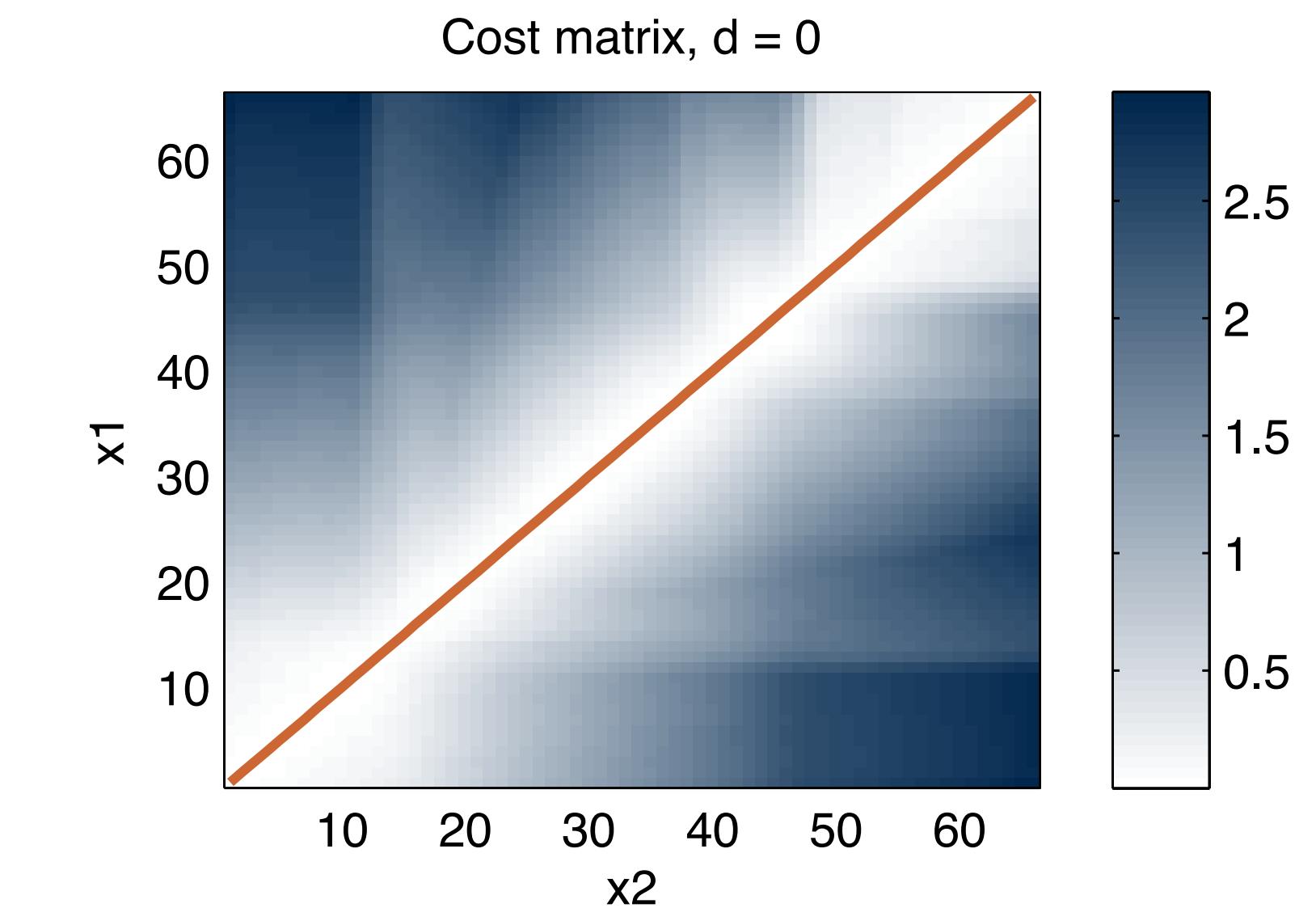
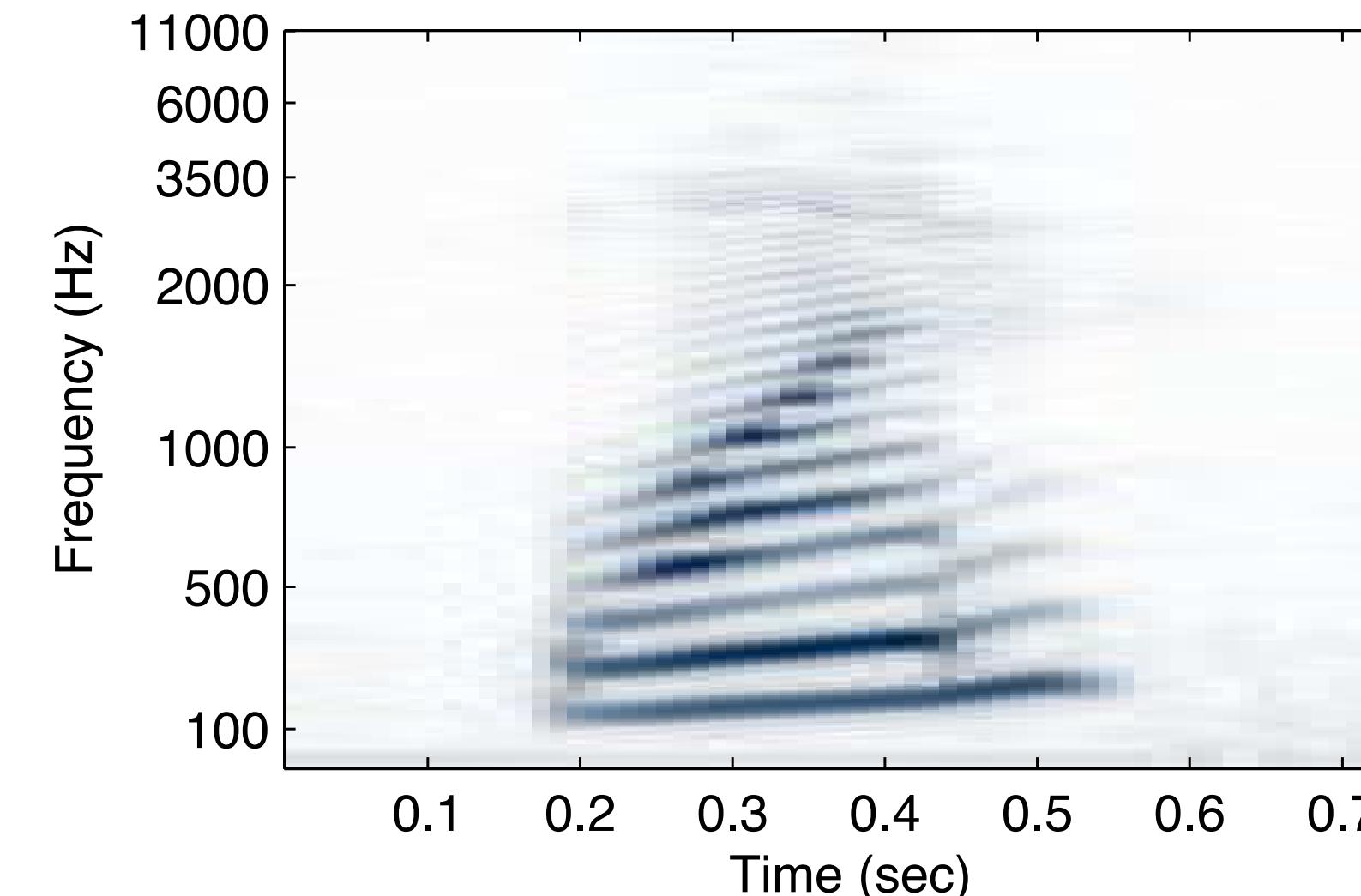
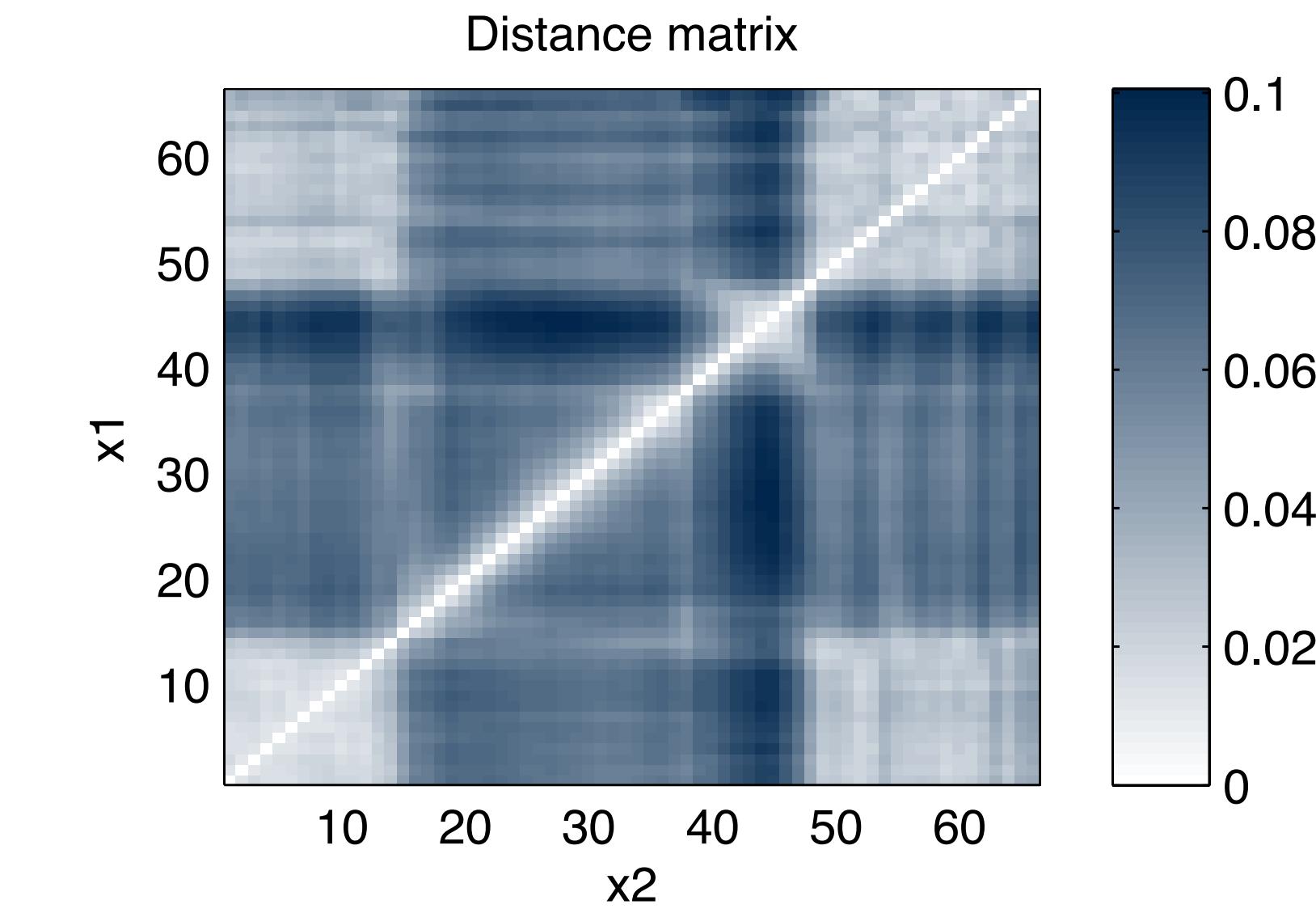
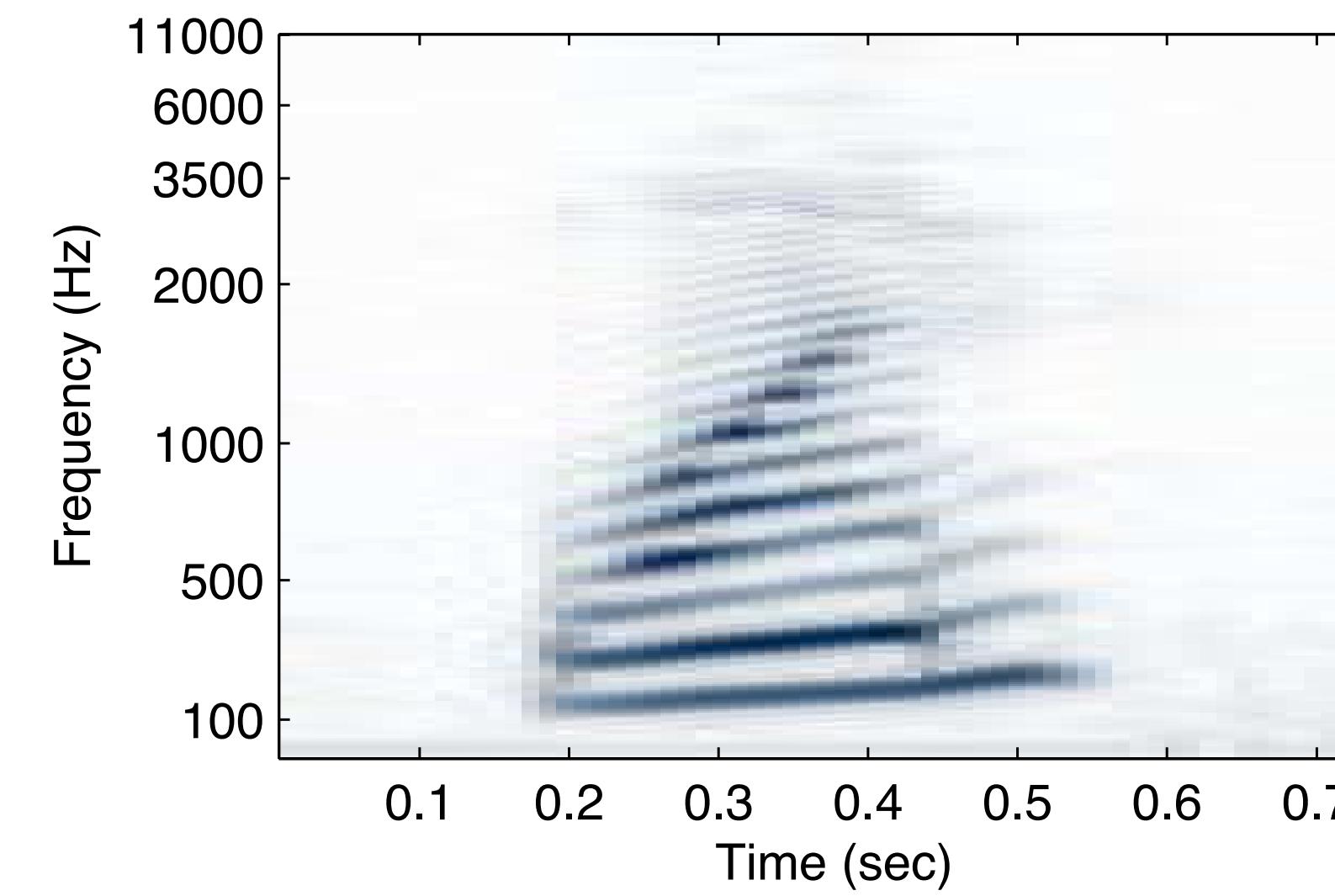
- Define acceptable local subpaths
 - Consider only these patterns
 - Application dependent



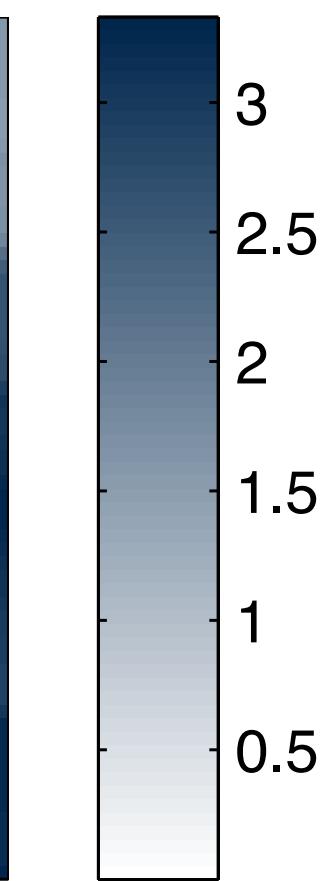
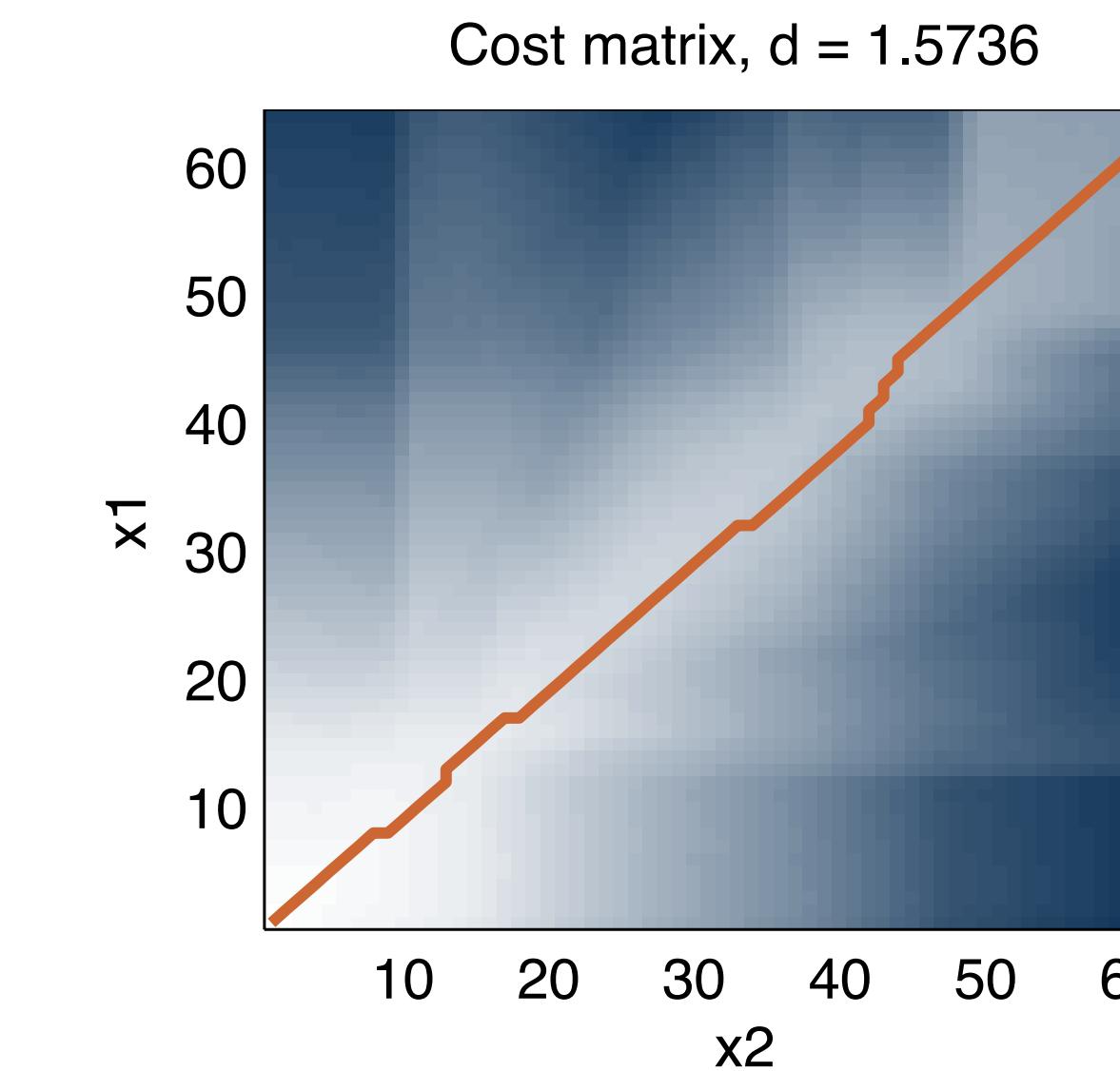
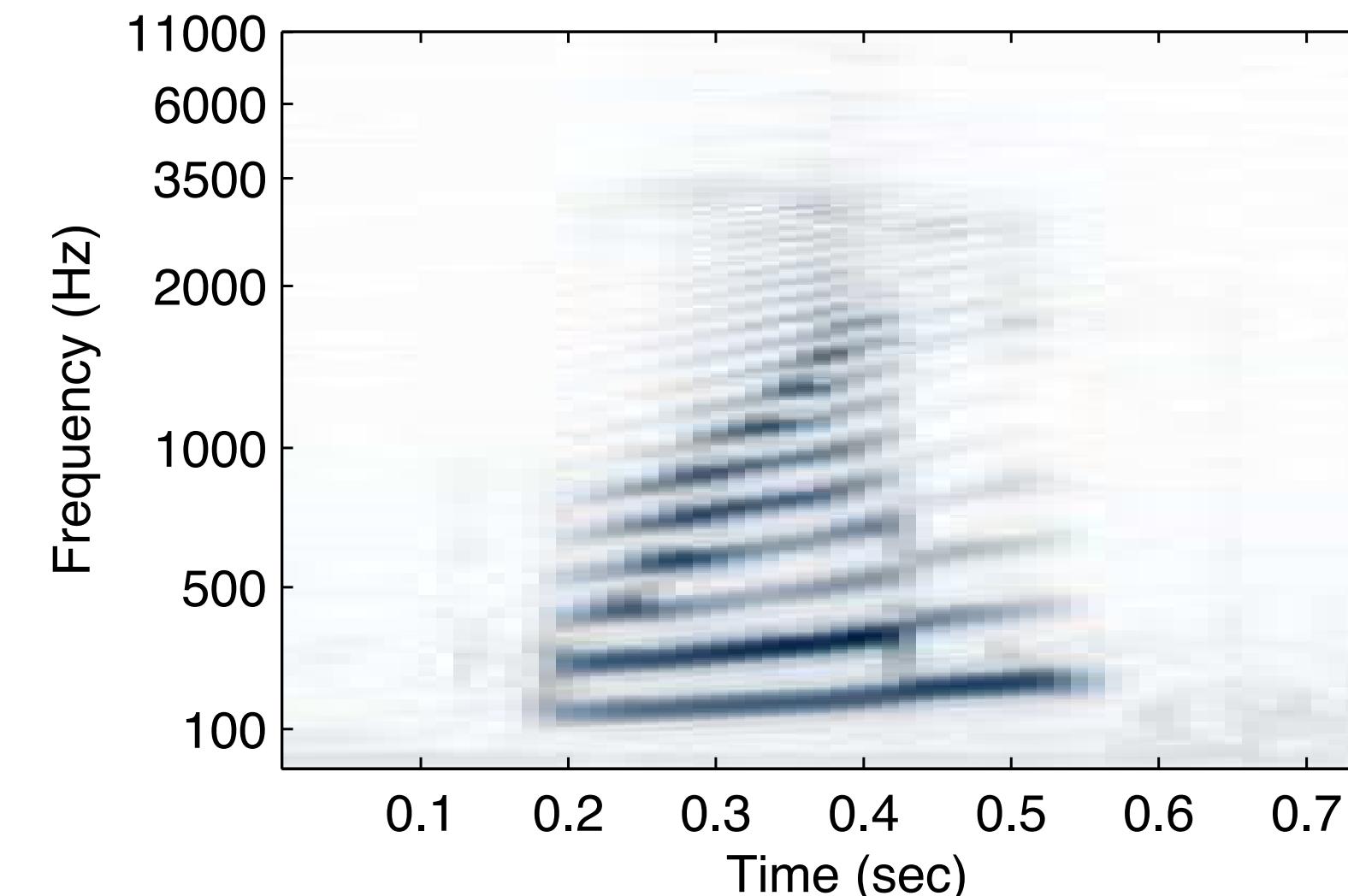
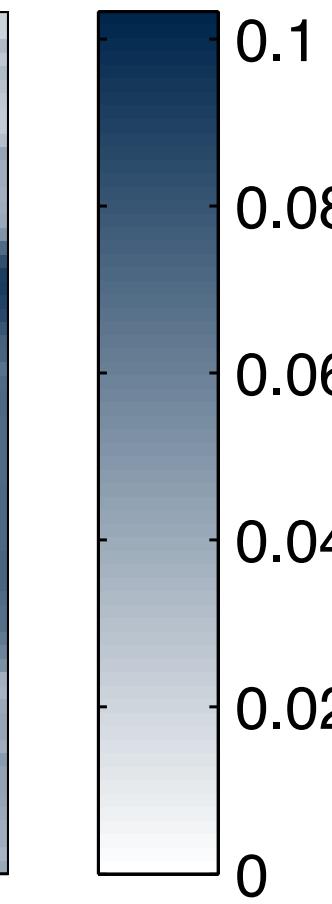
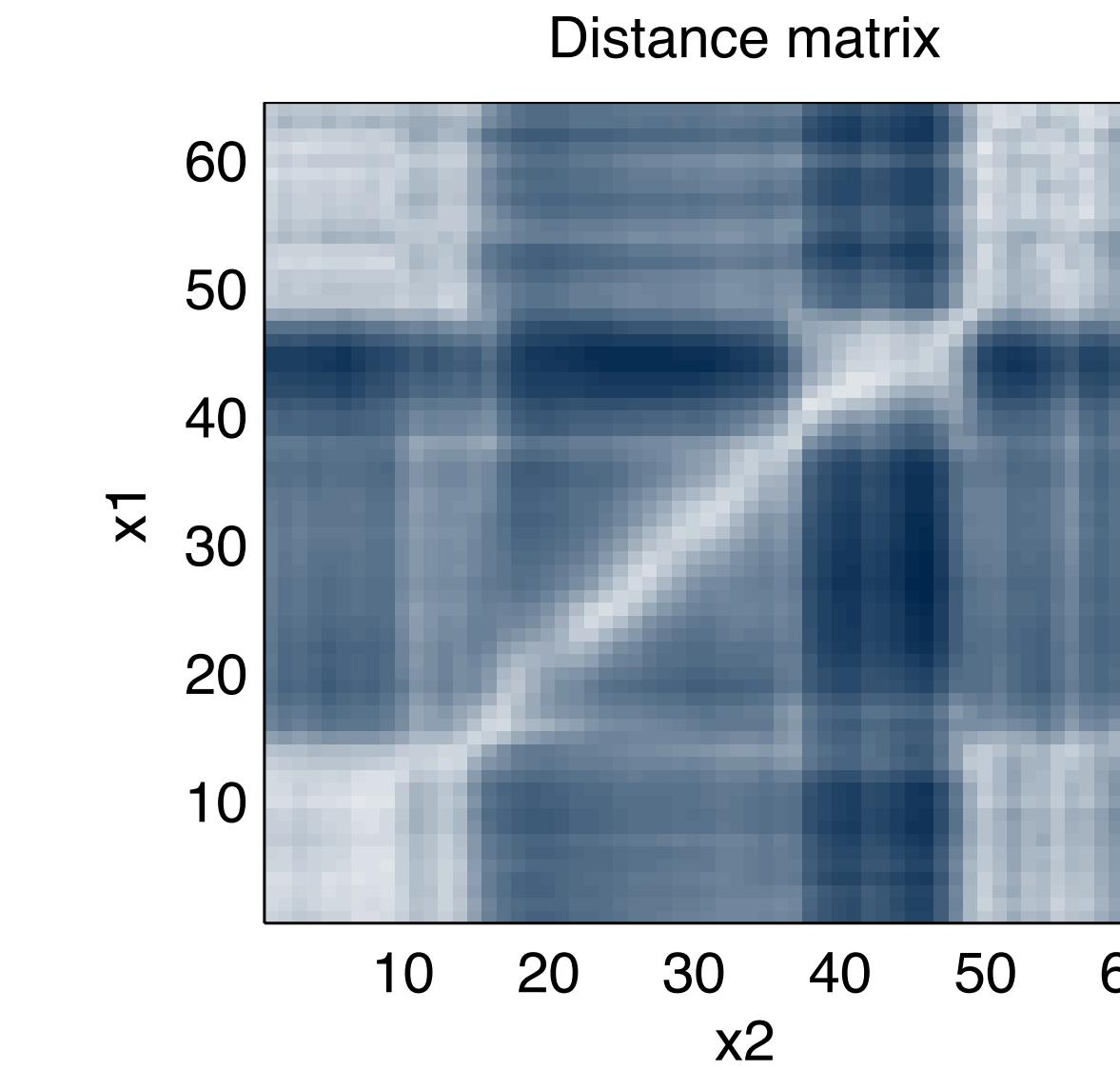
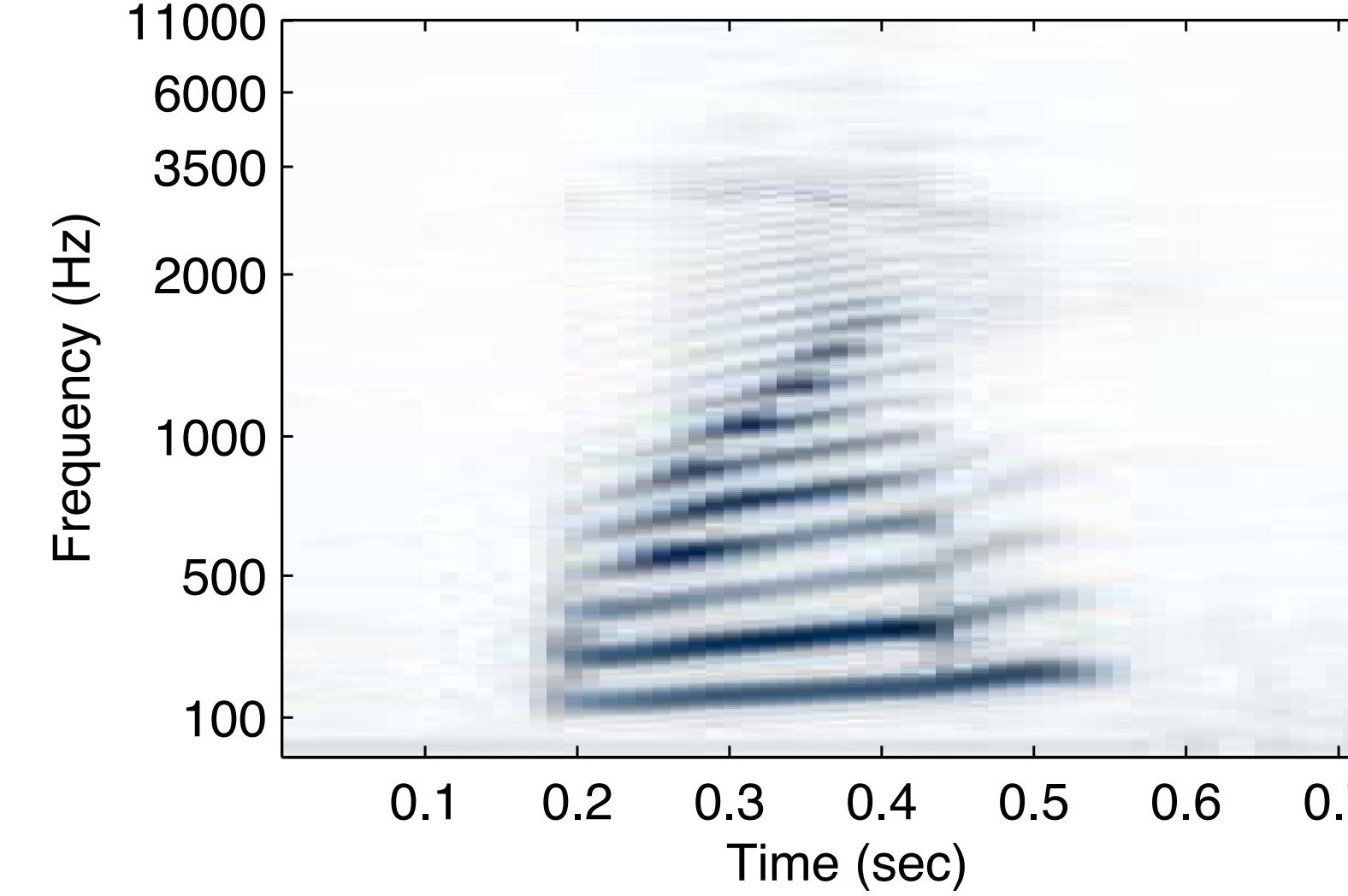
Toy data run



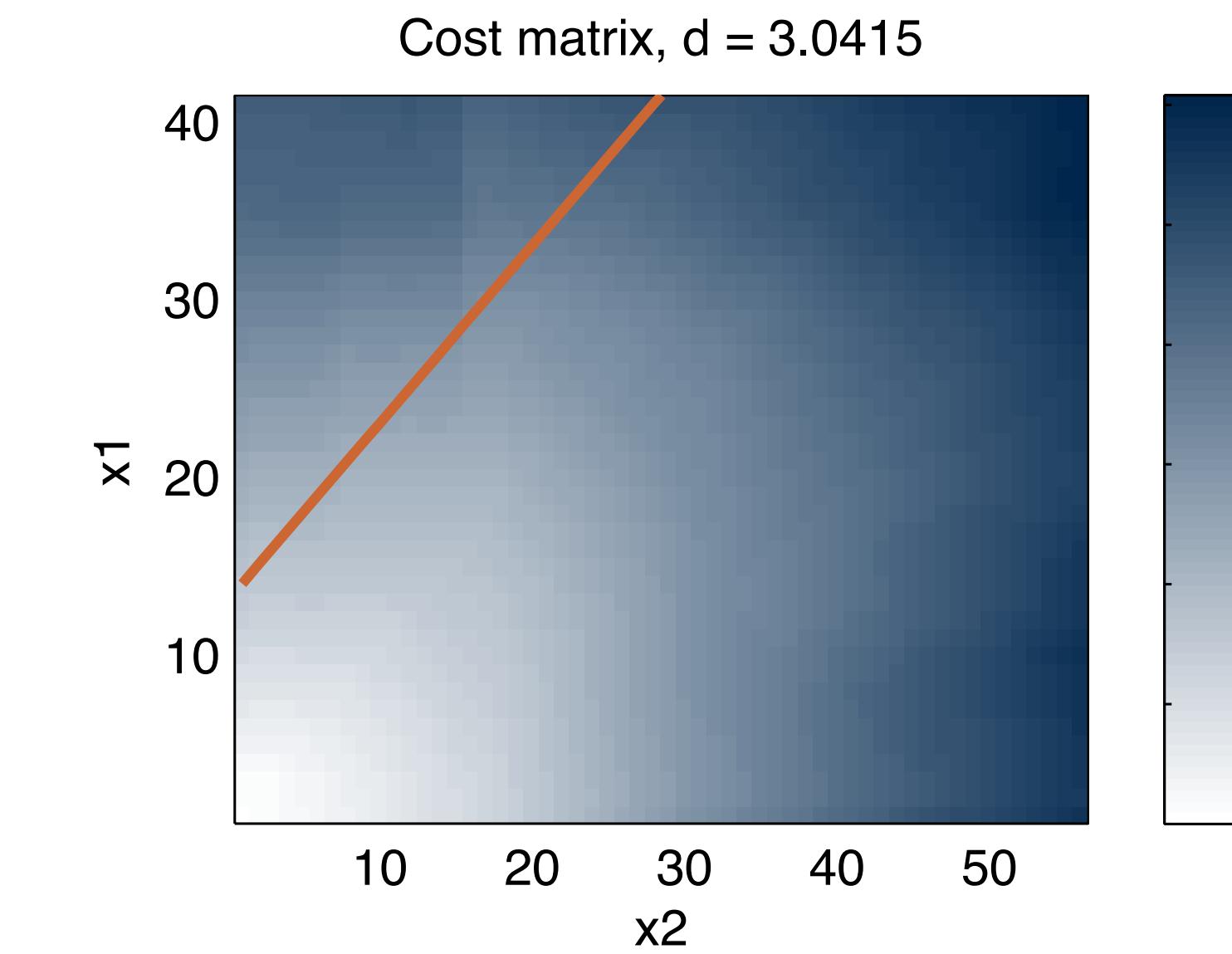
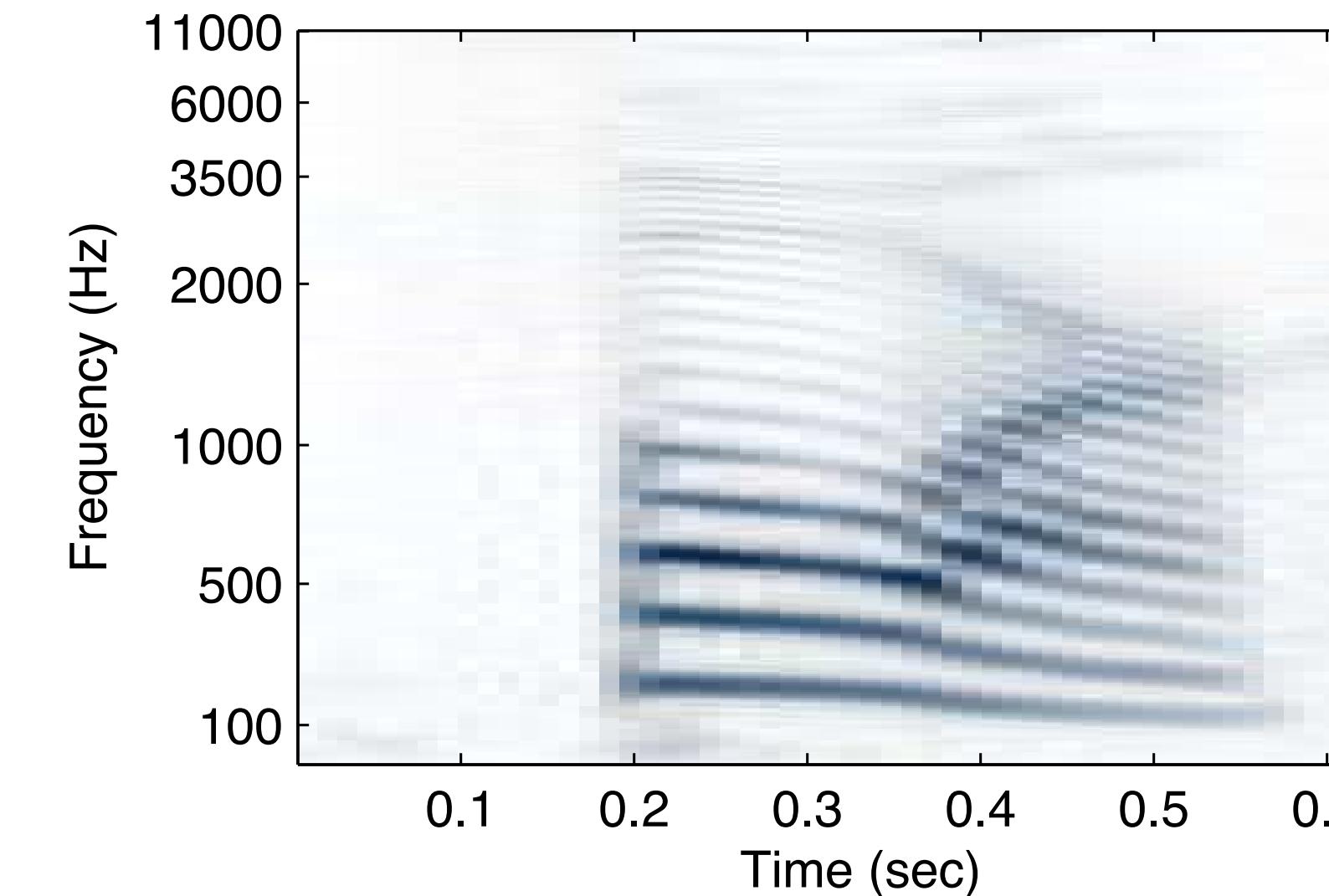
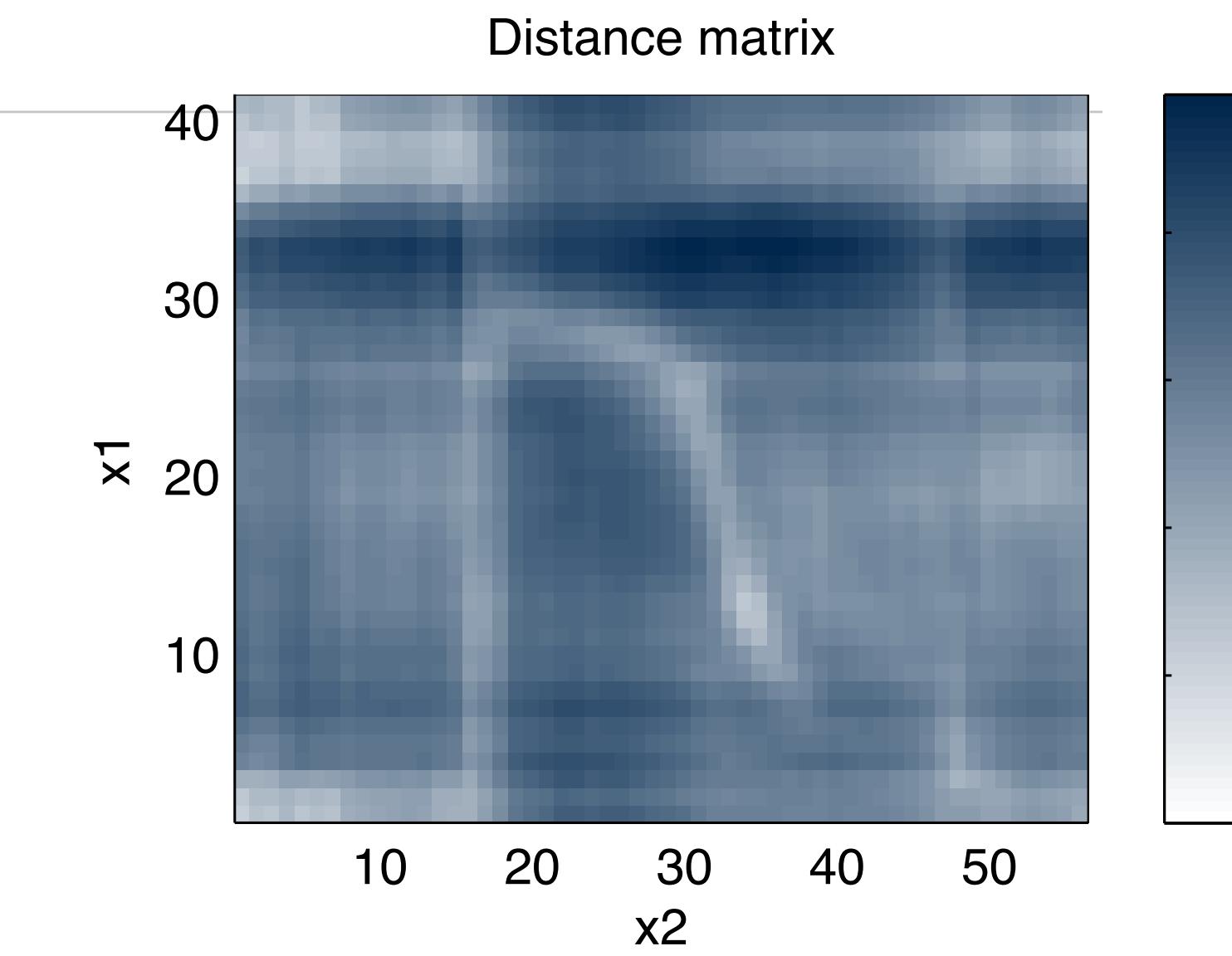
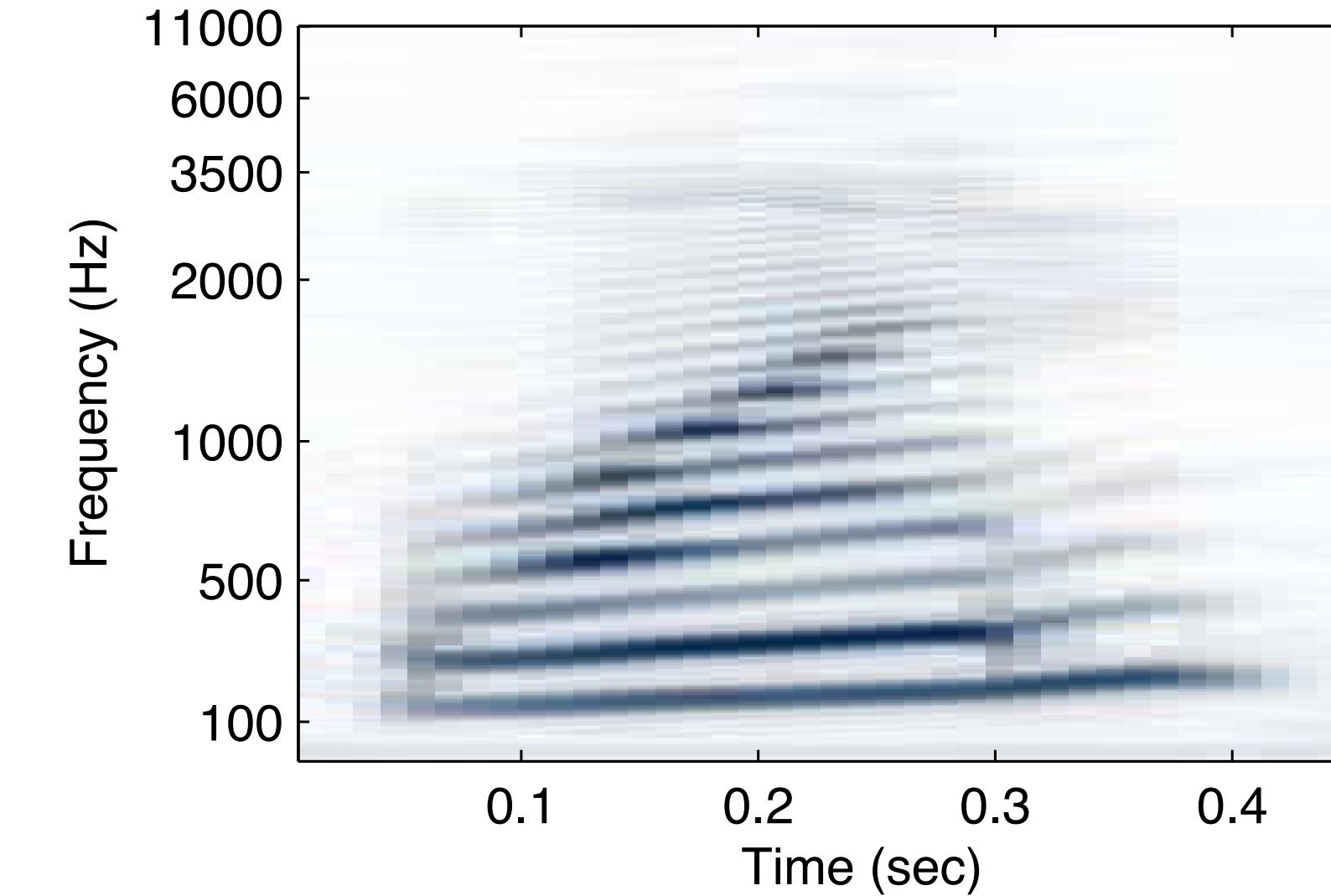
Speech example with identical utterance



Ditto with similar utterance

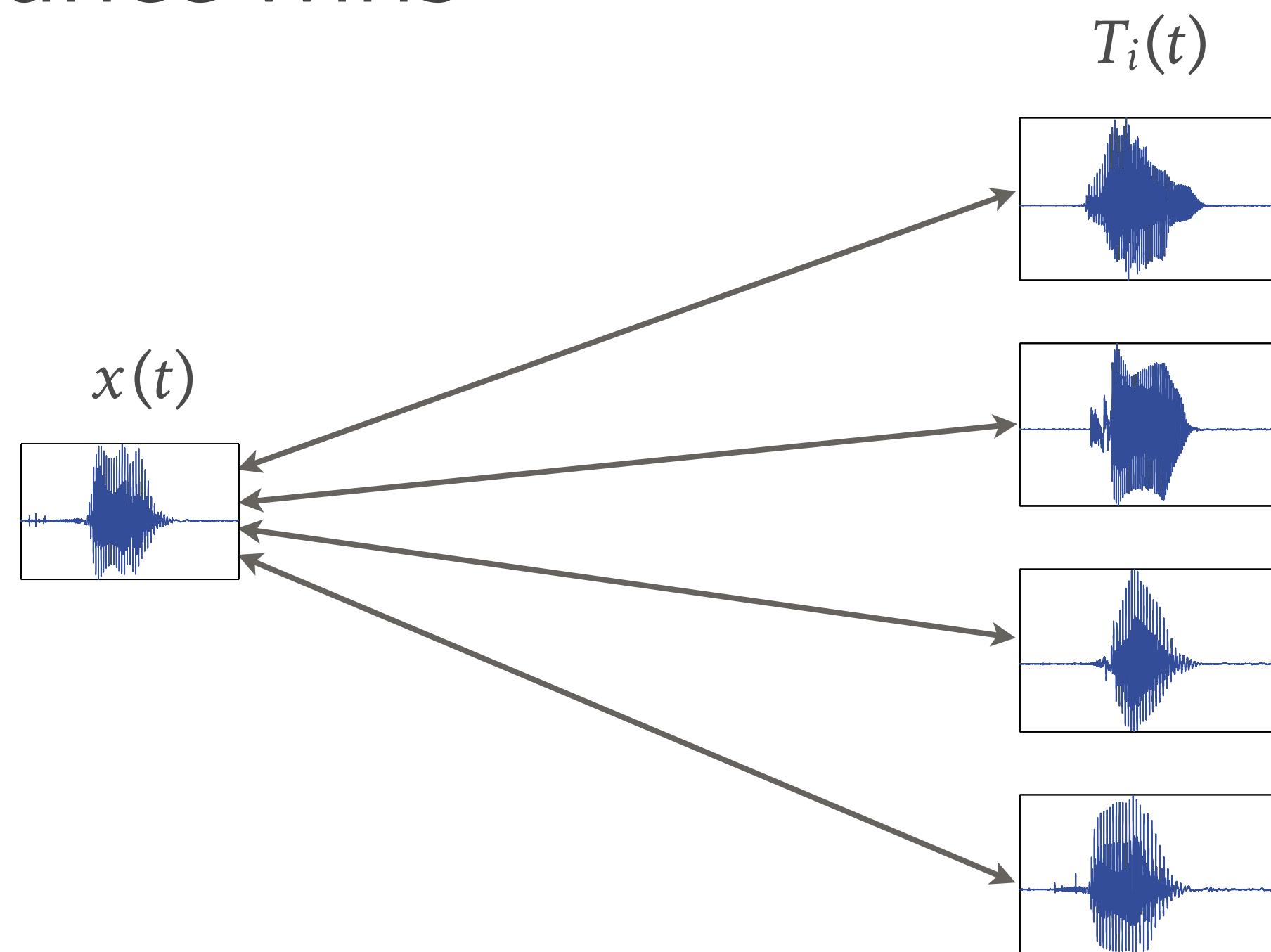


Ditto with different utterance

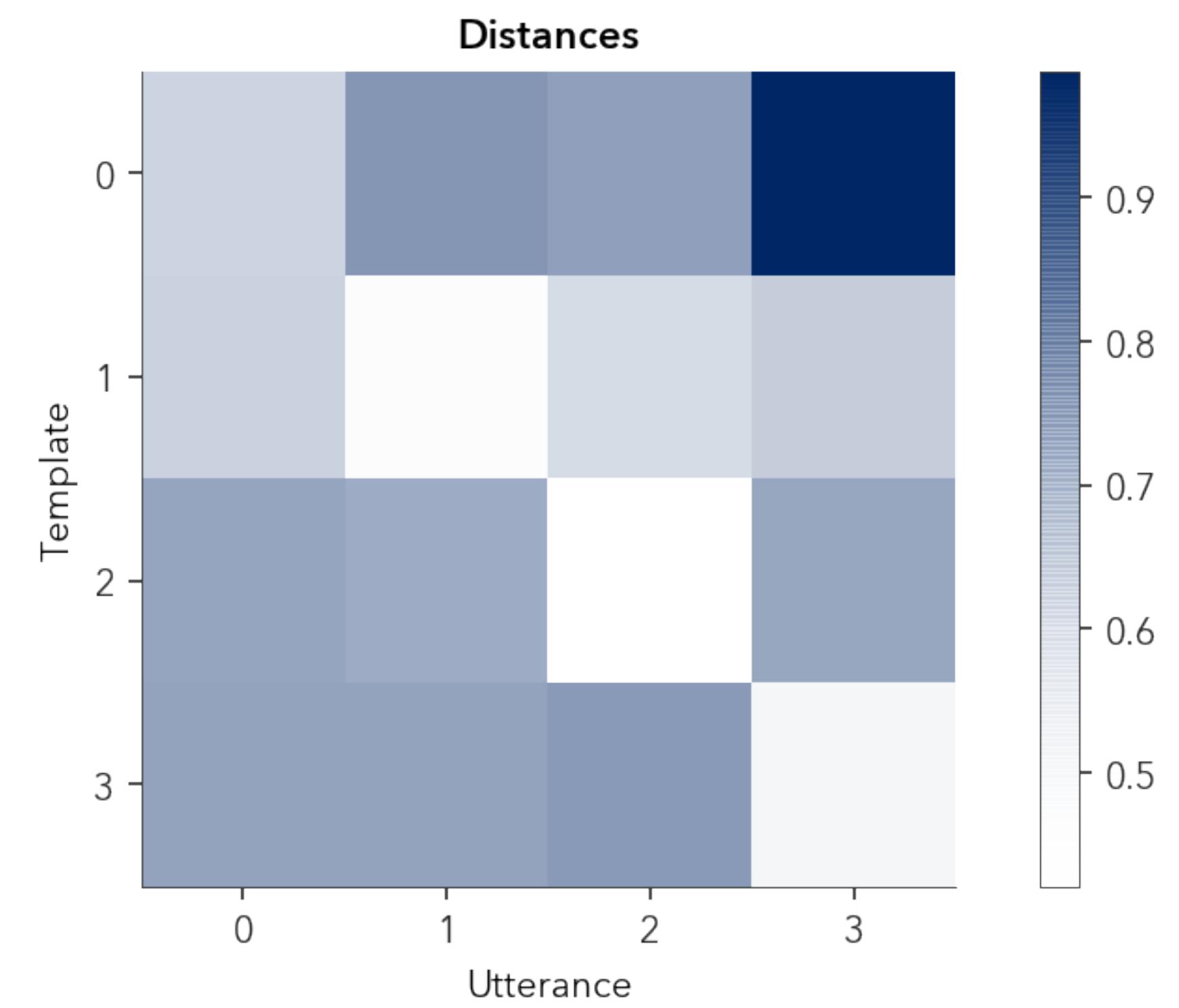
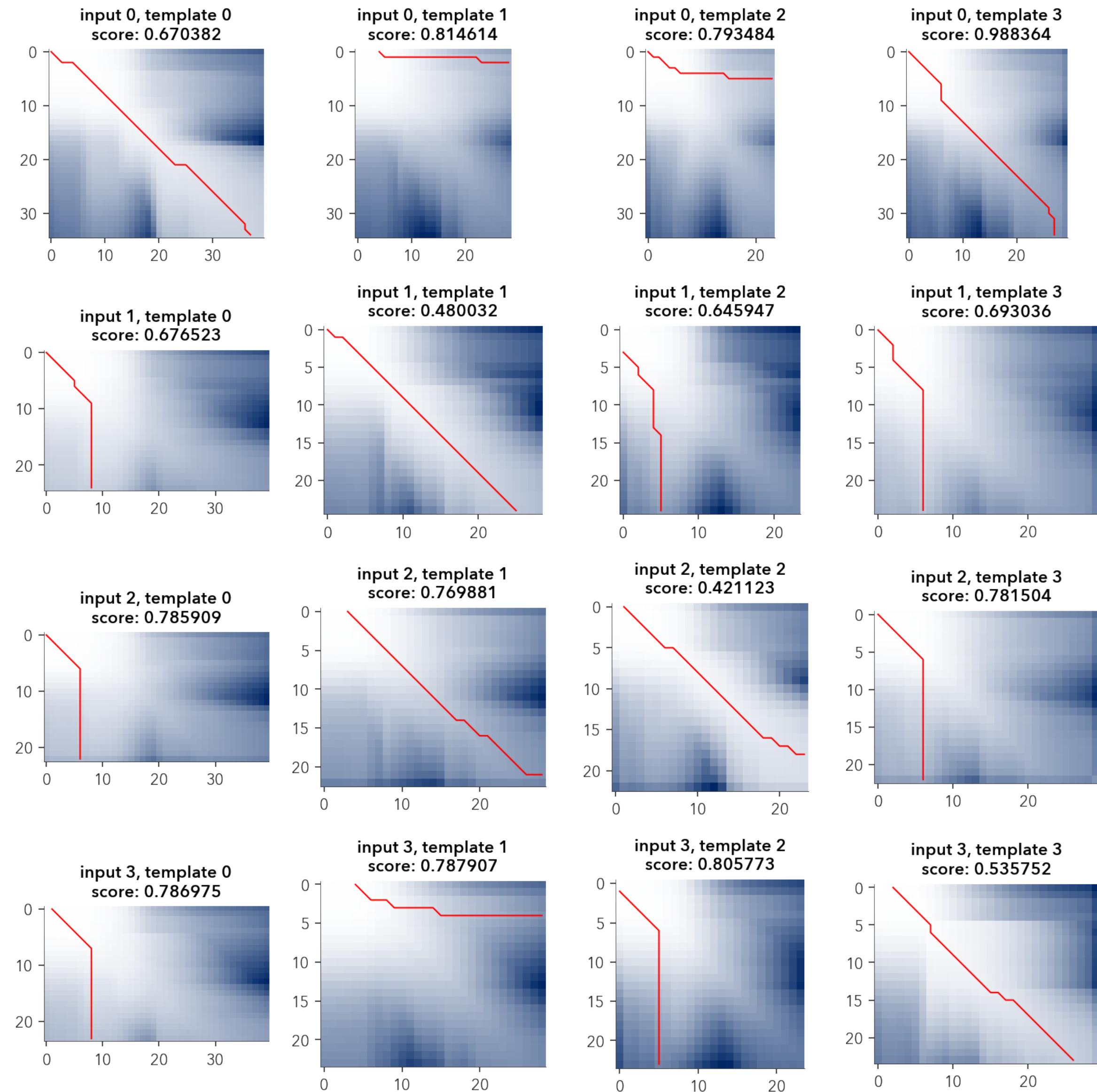


A basic speech recognizer

- Collect template spoken words $T_i(t)$
- Get their DTW distances from input $x(t)$
 - Smallest distance wins



Example case



Defining a distance for time series

- Dynamic Time Warping computes time series distances
 - Dependent on the constraints, etc
- Having a time series distance gives us options
 - Nearest-neighbor classifiers (just did that)
 - Clustering of time series
 - Manifold embeddings of time series
 - ...

A real-world application of DTW



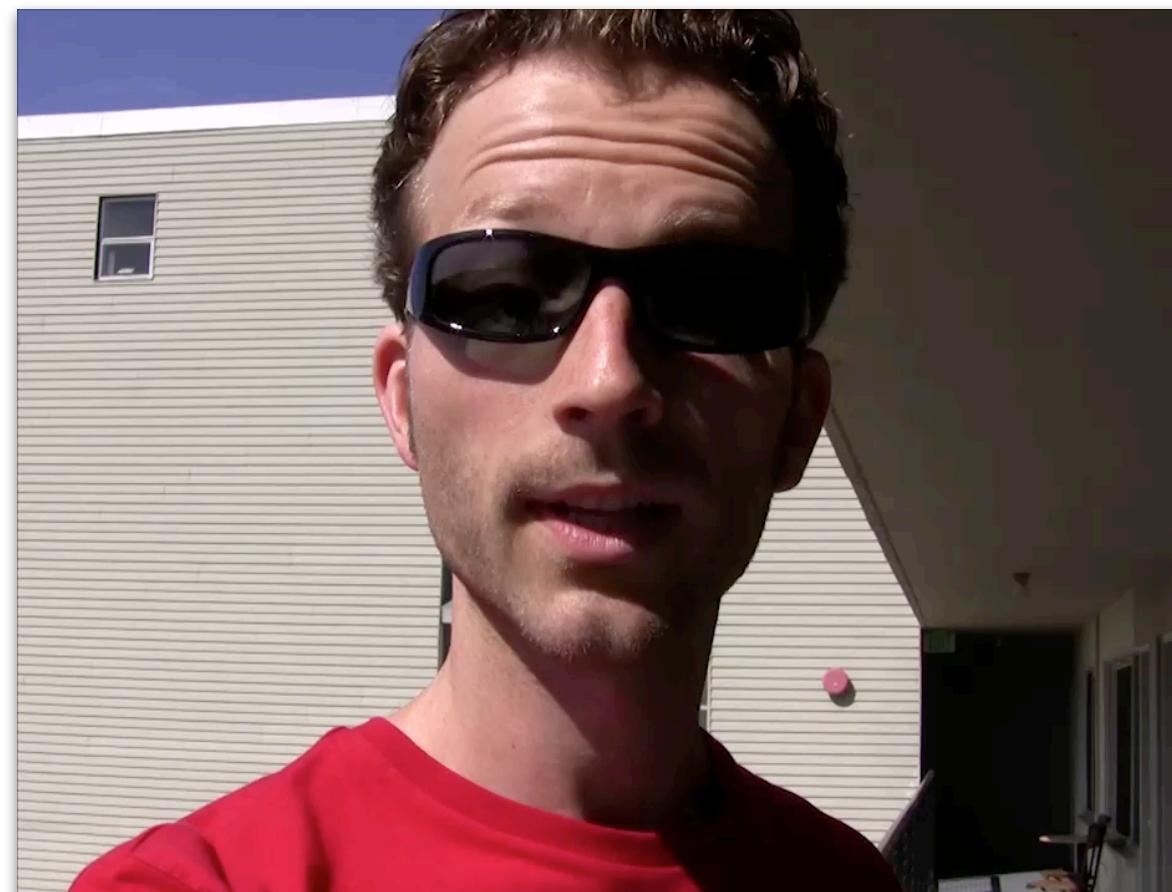
- Simplifying ADR
 - Costs time and money
- Automatically align audio tracks



Good take, lousy audio



Good audio, lousy sync



Good take, good audio!



What's not quite right so far?

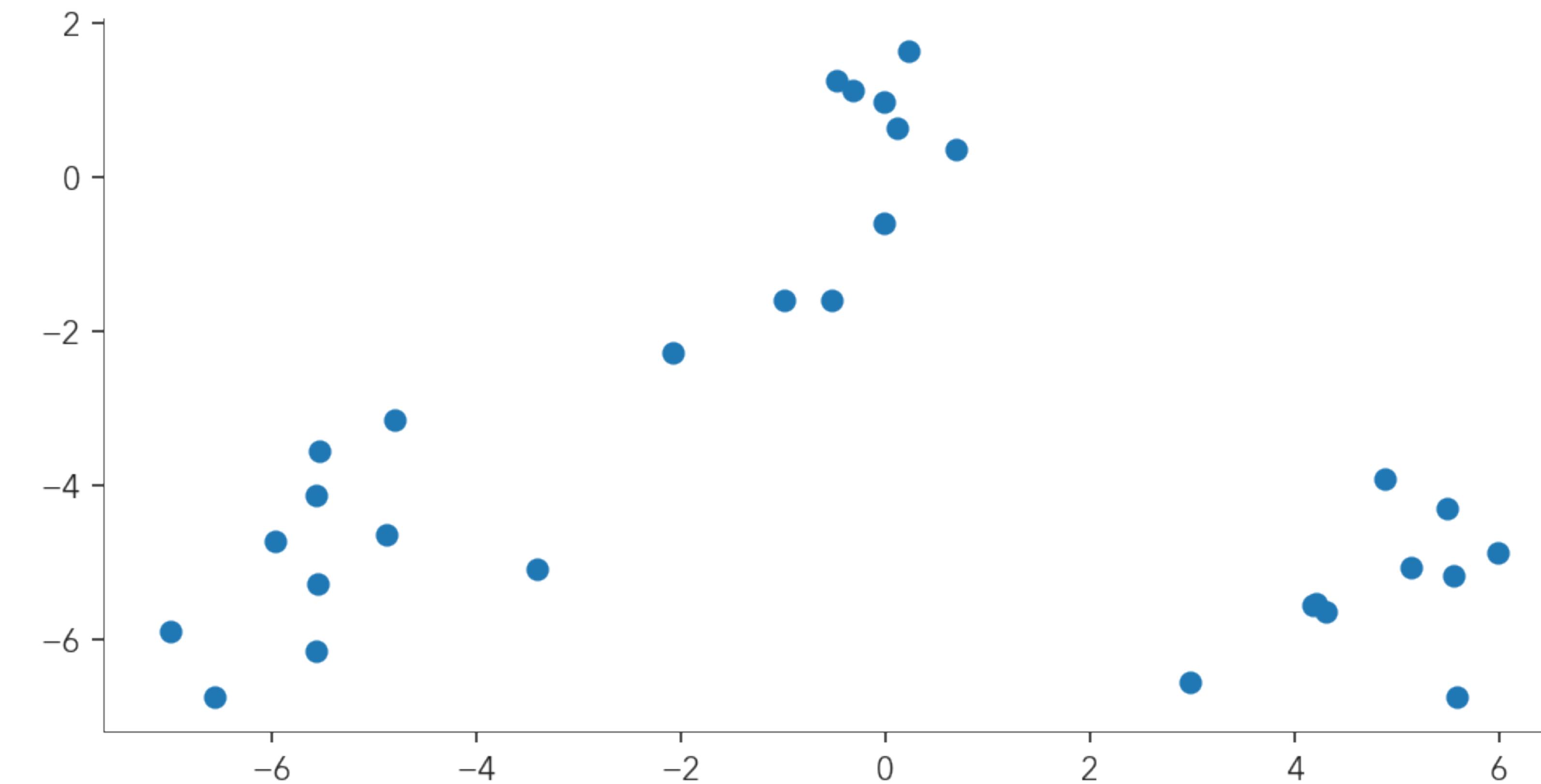
- Something I recently complain about a lot ...

We prefer soft models

- The paths are hard decisions
- We'd rather have soft assignments
- Also obtaining probabilities would help

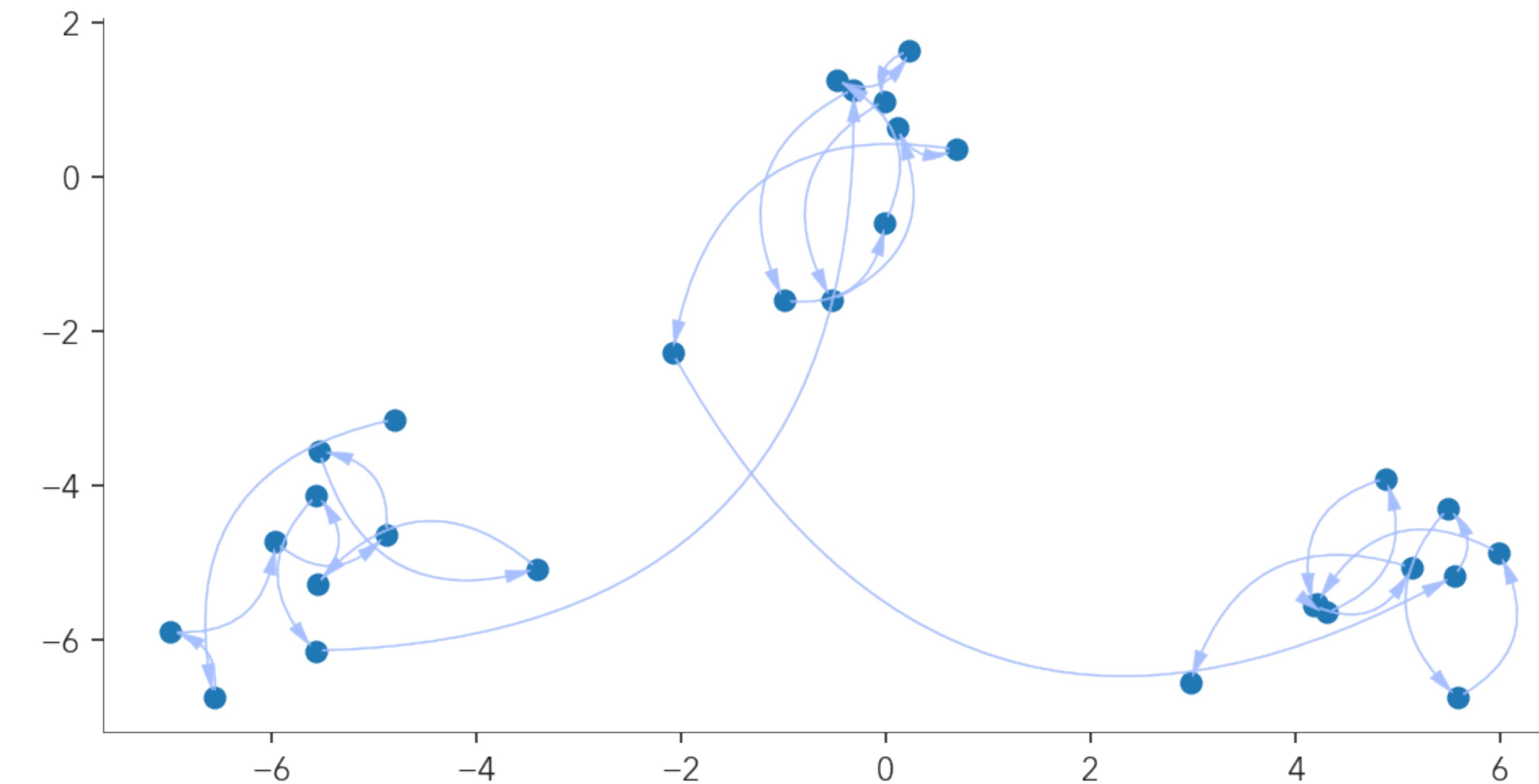
Starting with the GMM

- GMMs are “time agnostic”
 - We don’t have a notion of a sequence



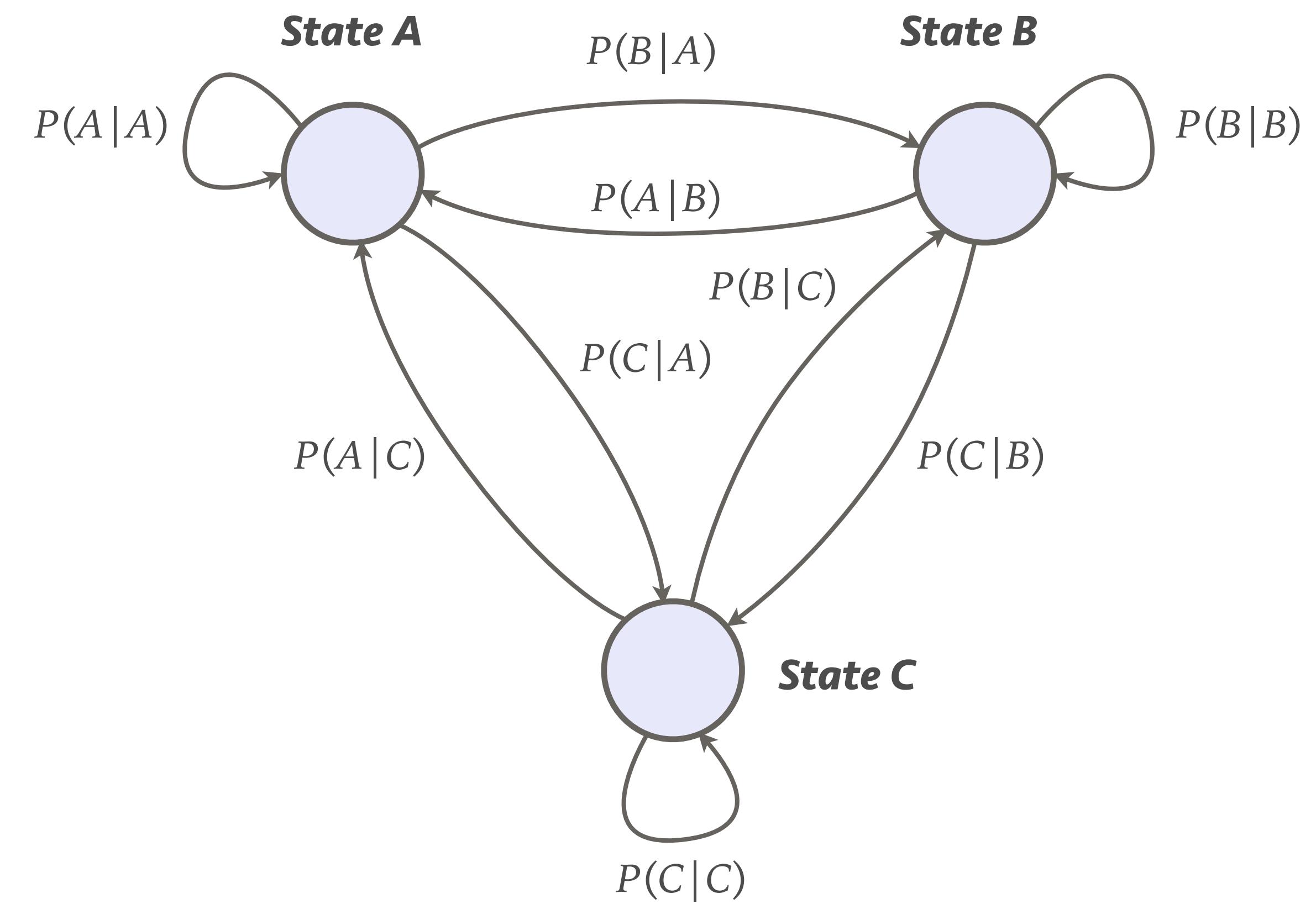
What if there is time order?

- We want to learn how the data progresses



Representing time

- Hidden Markov Model
 - Gaussian variant for now
- Each state is represented by a Gaussian distribution
- Transition probabilities between all states
 - Only the last state matters



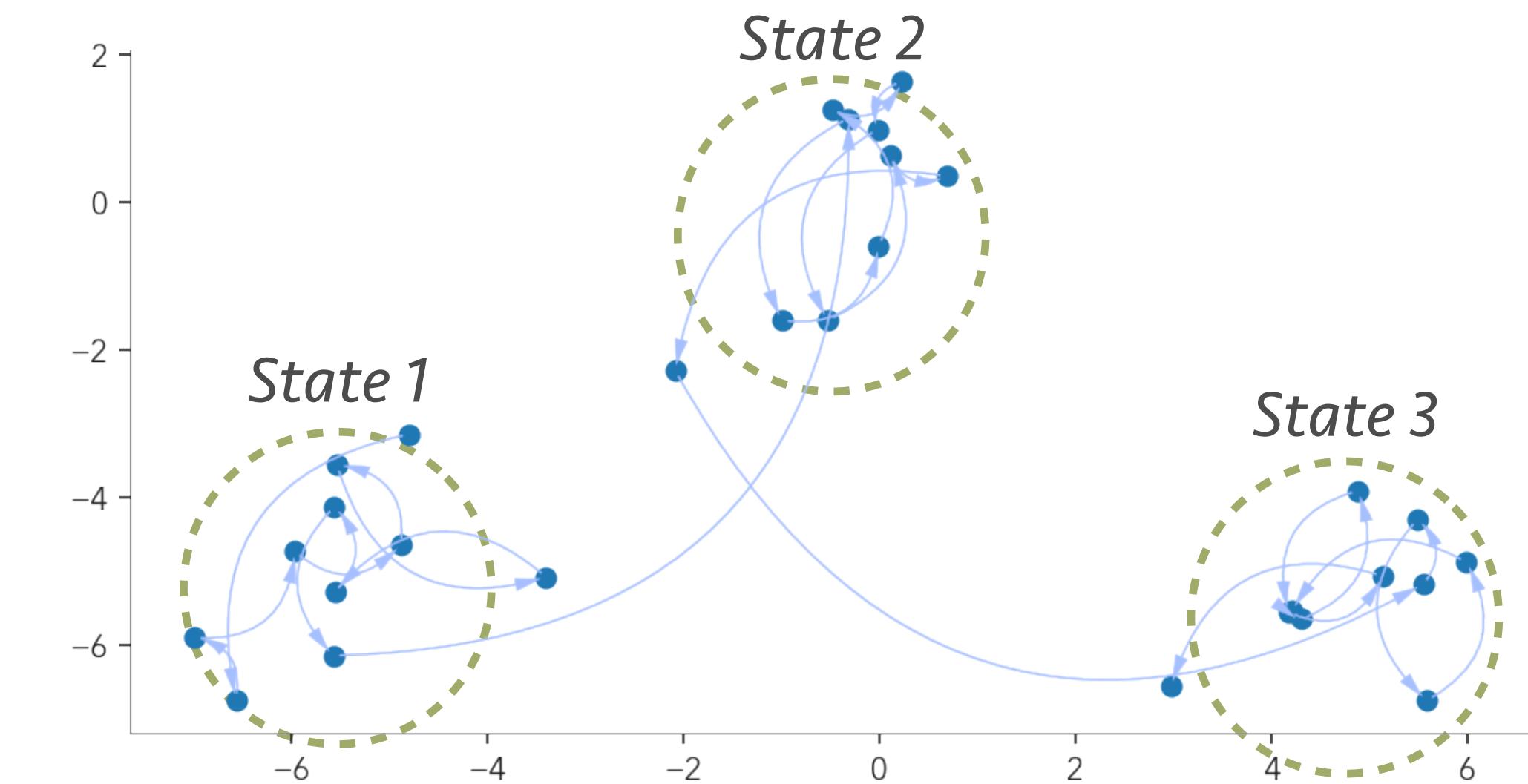
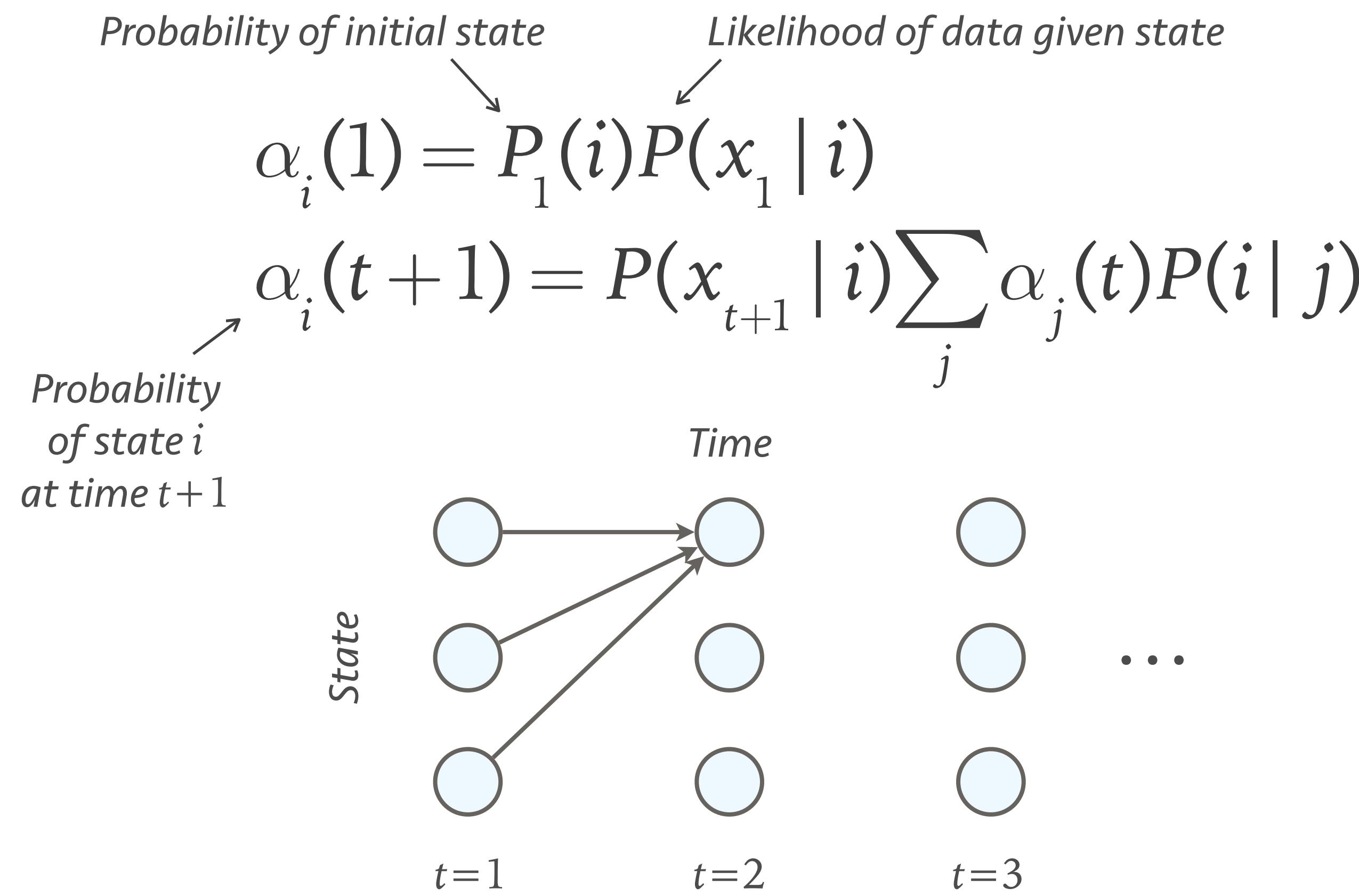
Initial Probabilities: $P(A)$, $P(B)$, $P(C)$

Problems to solve

- Evaluation
 - Given a model, evaluate likelihoods
- Decoding
 - Given a model, find state sequences
- Learning
 - Given the data, find the model parameters

Evaluation

- Propagate likelihoods using the *forward algorithm*:



Evaluation

- Forward pass provides probability of a specific state and time, given past inputs

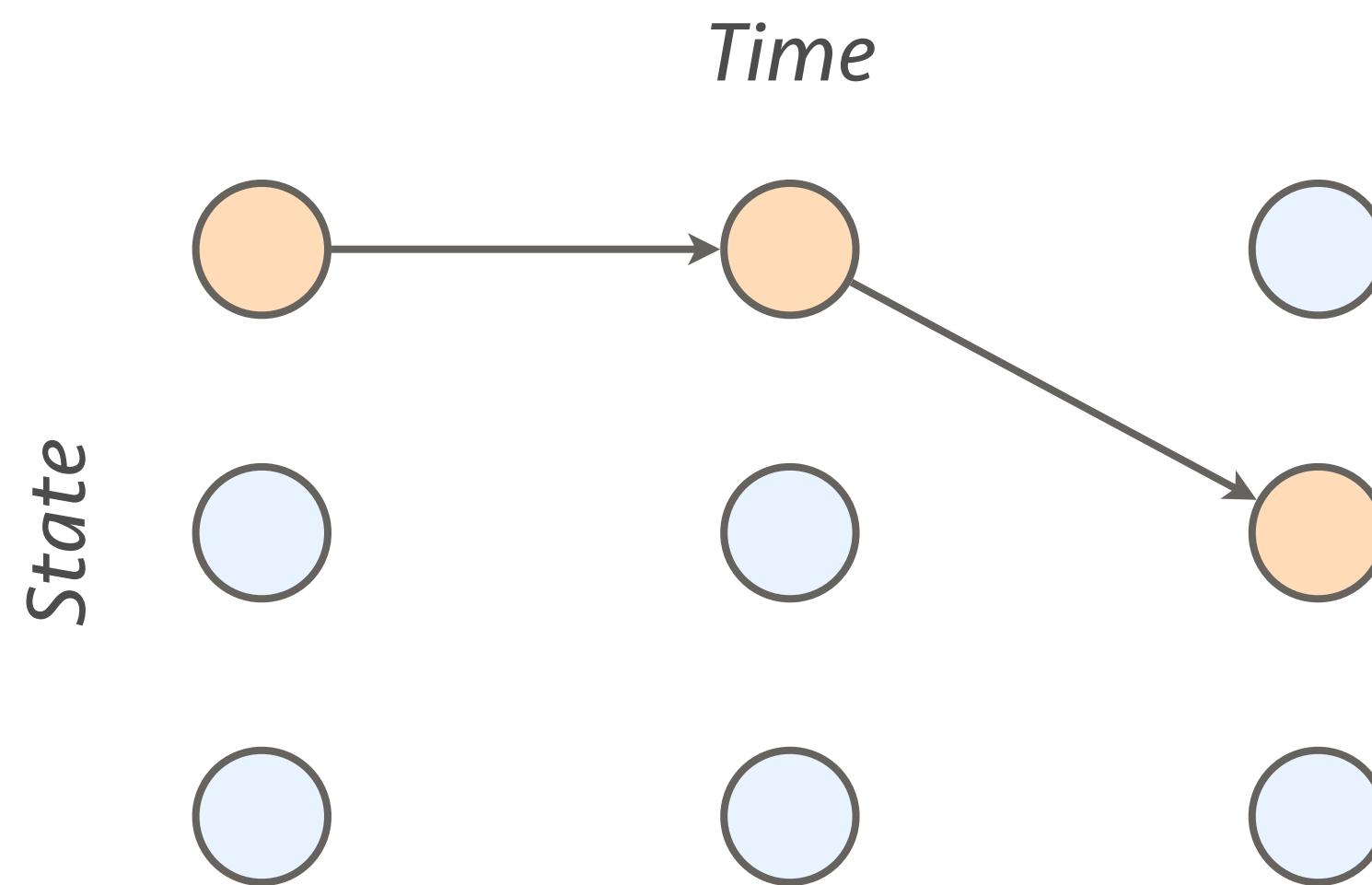
$$\alpha_i(t) = P(x_t | i) \sum_j \alpha_j(t-1) P(i | j)$$

- Terminal time point values provide the overall likelihood of model given an input

$$P(\mathbf{x}) = \sum_i \alpha_i(T)$$

Decoding

- The forward pass provides us with the state likelihoods
 - Similar to the DTW path costs
- With a backwards pass we can find most likely transitions
 - Just as we did with DTW
- The Viterbi algorithm



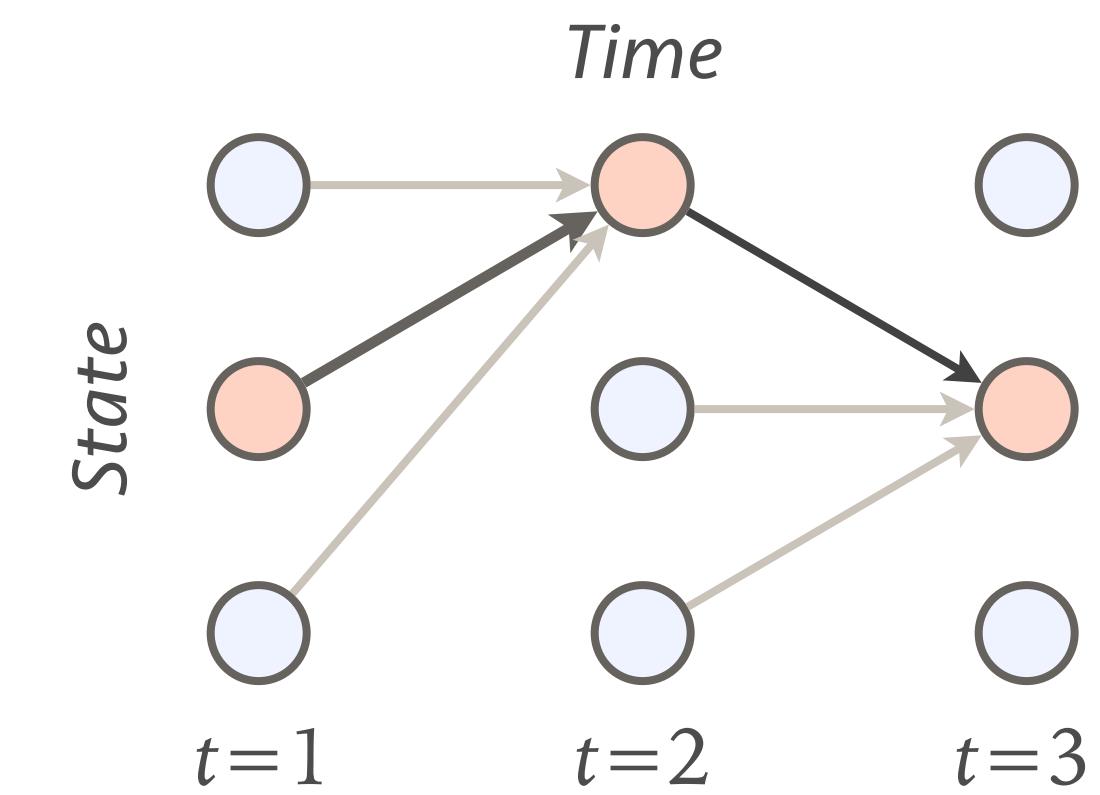
The Viterbi algorithm

- Similar to forward algorithm, but with a hard decision:

$$\nu_i(1) = P_1(i)P(x_1 | i)$$

$$\nu_i(t+1) = P(x_{t+1} | i) \max\left(P(i | j)\nu_j(t)\right)$$

- Probability of most likely path up to time $t+1$ that ends in state i
 - For every step remember most likely transition (as with DTW)
-
- Backwards pass
 - Get most likely terminal state
 - Work backwards using the most likely transitions



Learning

- An EM-type approach
 - Known as Baum-Welch training
- E-step
 - Find likelihood of each data point being associated with each state
- M-step
 - Estimate each state's parameters based on the associations from the E-step

E-step: Forward-backward pass

- Forward pass provides probability of a specific state and time given past inputs $a_i(t)$
- A similar backward pass will provide probability of given state and time given future inputs $\beta_i(t)$
 - Same as forward pass, but computed backwards in time
- The product of these will be the probability of a state and time given all inputs $\gamma_i(t) = a_i(t)\beta_i(t)$

M-step: Parameter estimation

- State models
 - Use $\gamma_i(t)$ as weights to compute state parameters, e.g.:

$$\mu_i = \frac{\sum_t x_t \gamma_i(t)}{\sum_t \gamma_i(t)}$$

$$\Sigma_i = \frac{\sum_t \gamma_i(t) (x_t - \mu_i)(x_t - \mu_i)^\top}{\sum_t \gamma_i(t)}$$

- Transition matrix and initial probabilities
 - Estimate by counting transitions and observing most likely start

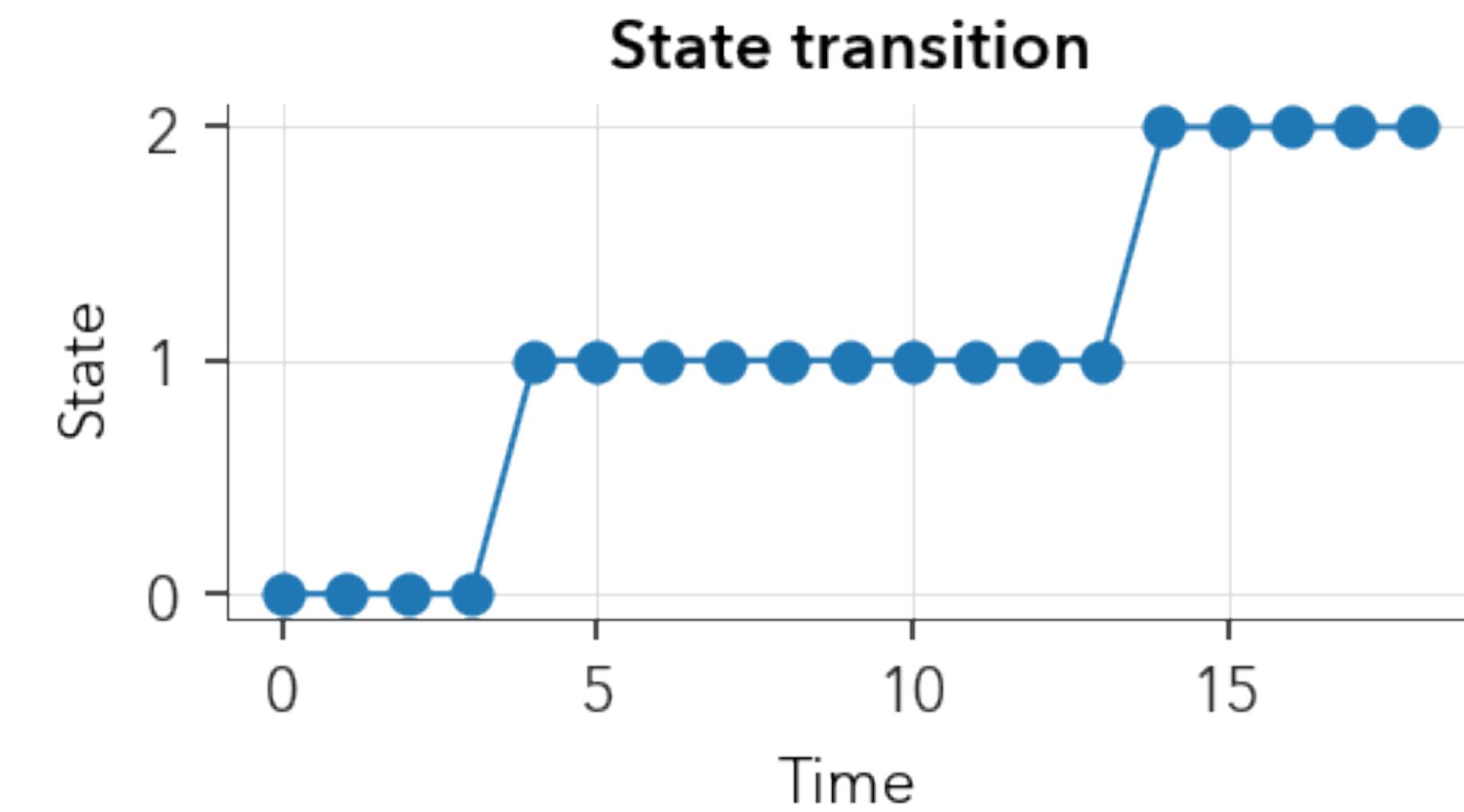
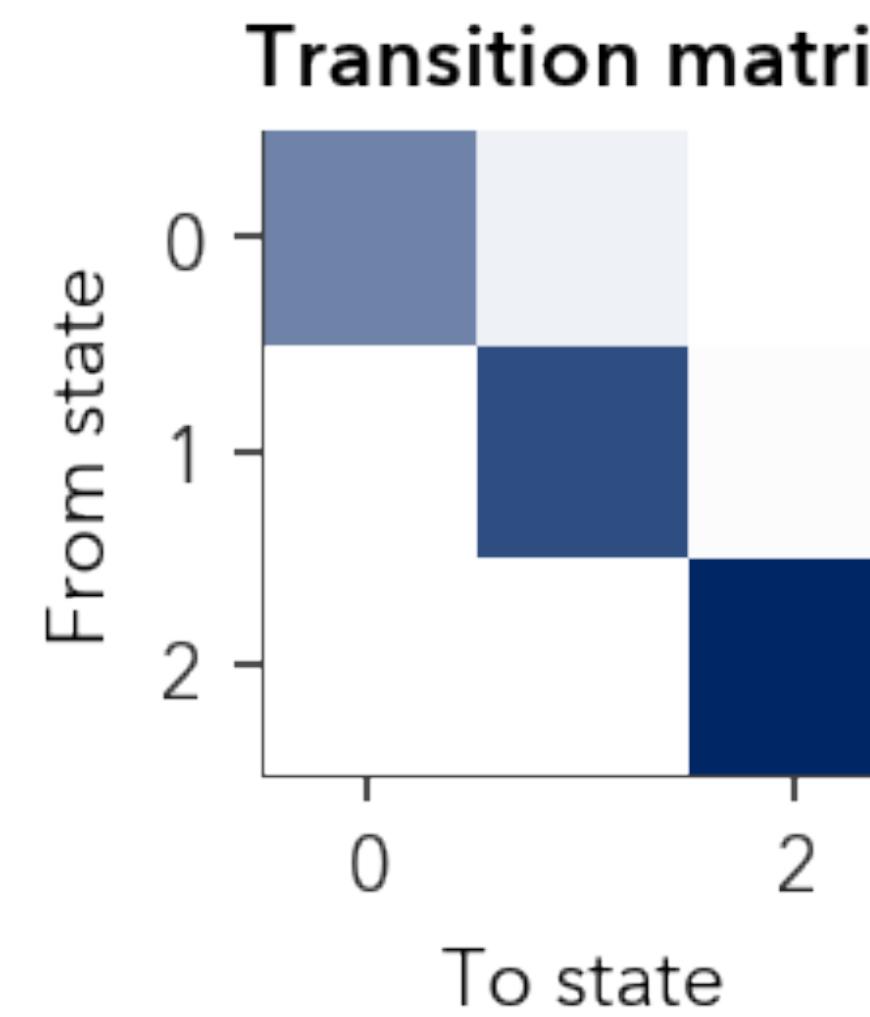
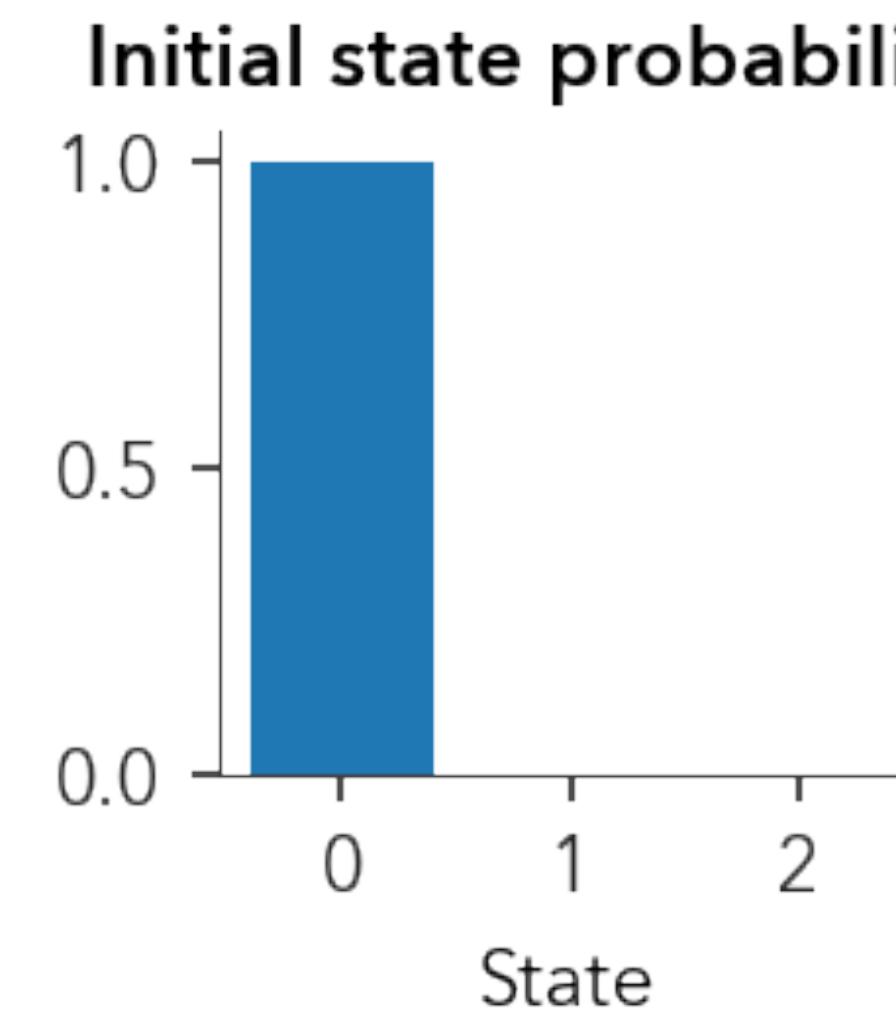
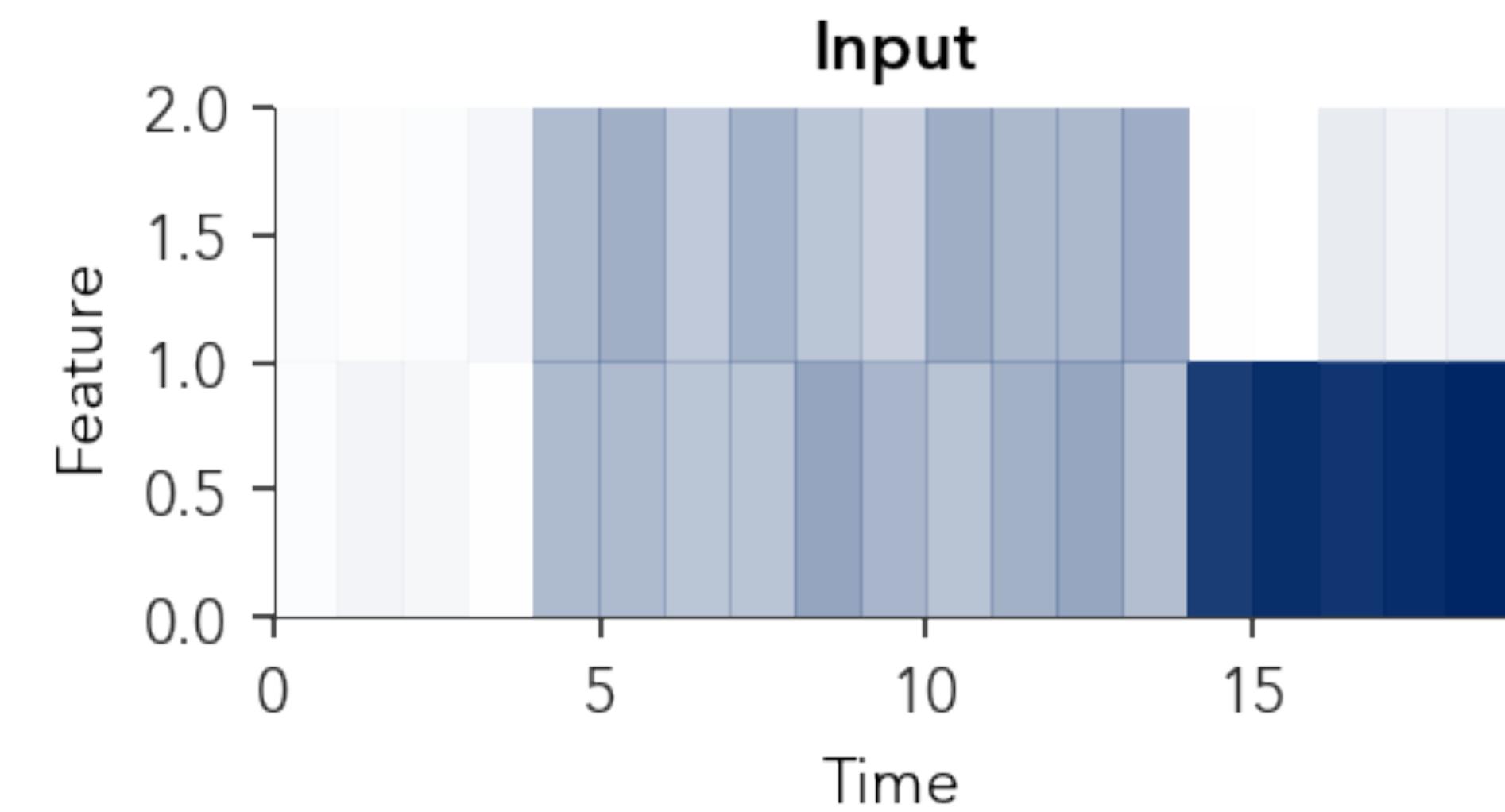
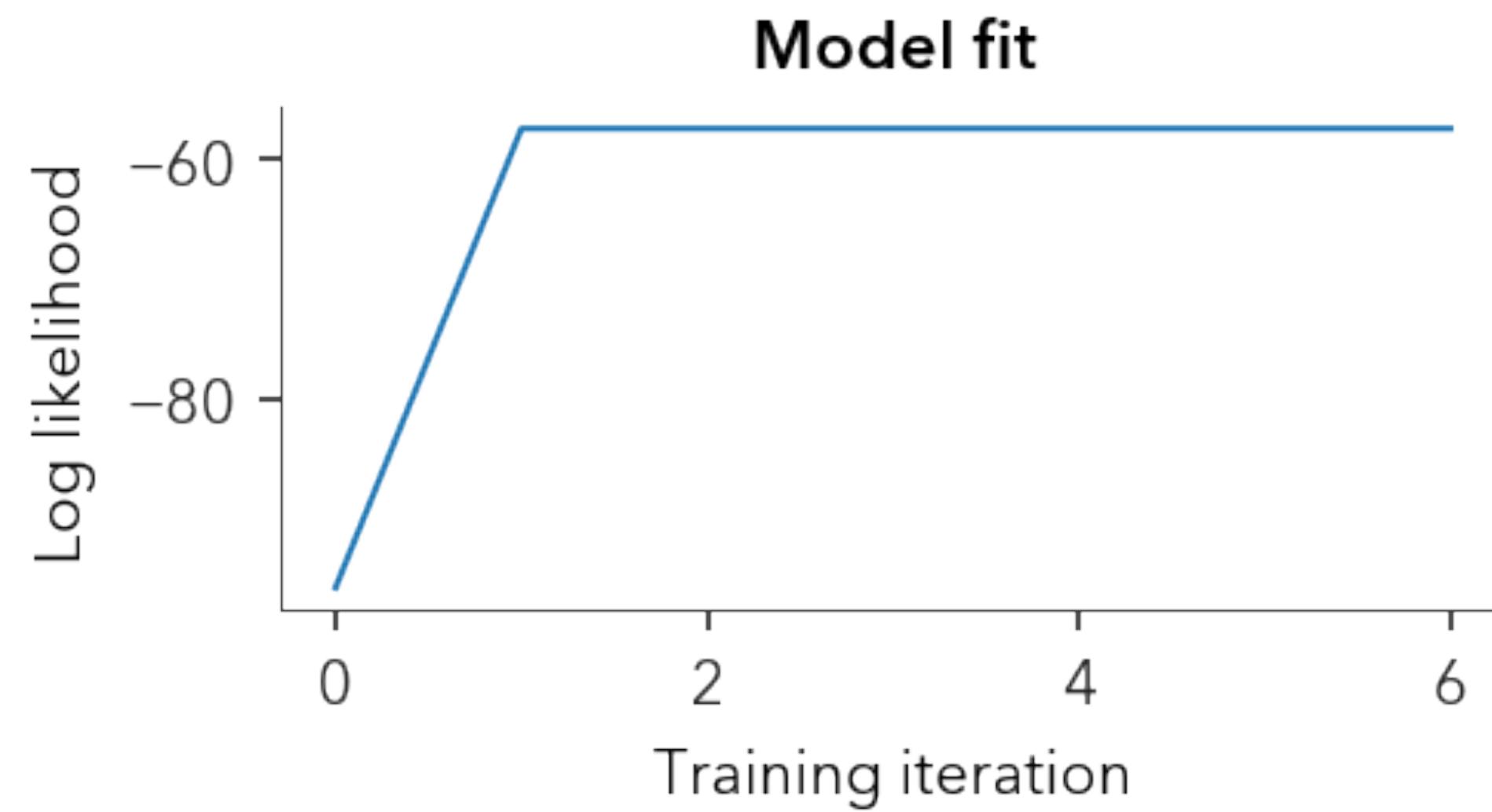
Noteworthy things

- HMM learning works with multiple training sequences
- You should use log probabilities
 - What's a state likelihood after a million time points? Can you represent it well numerically?

$$\alpha_i(t) = P(x_t | i) \sum_j \alpha_j(t-1) P(i | j)$$

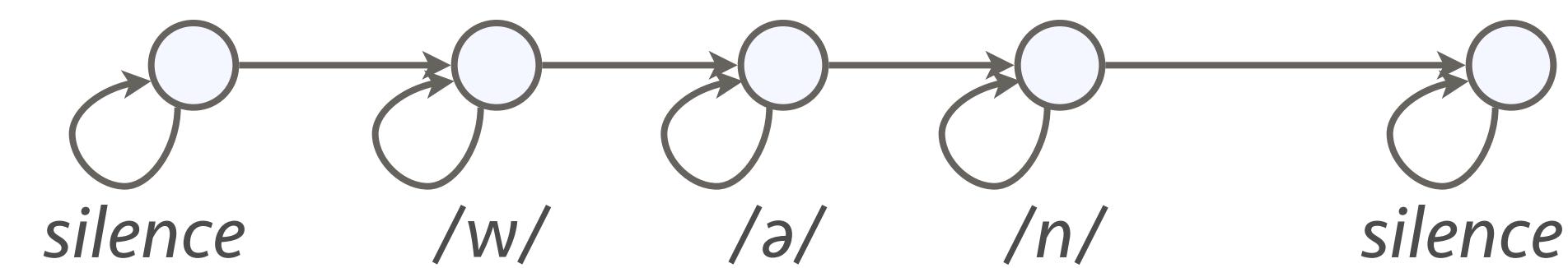
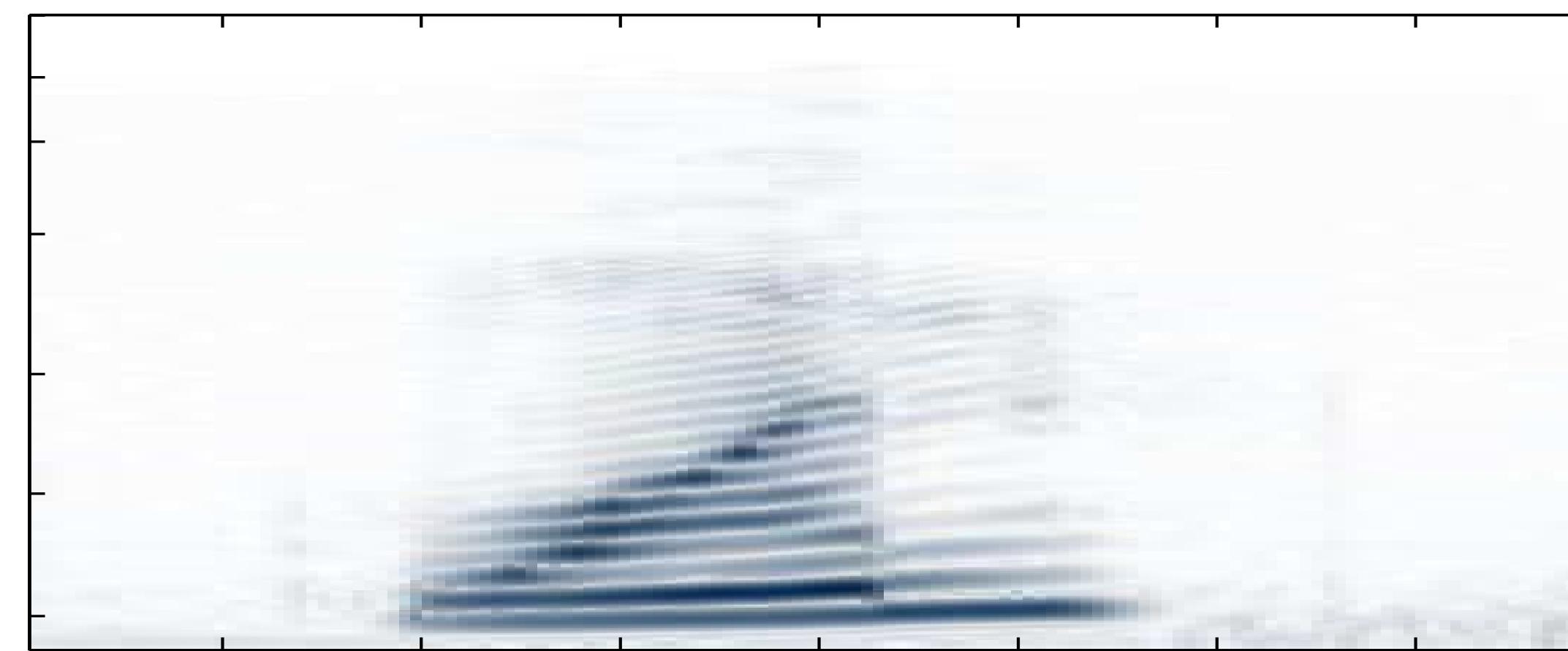
- You can use an arbitrary state model
 - e.g. a neural net, or a GMM

Learning an HMM model

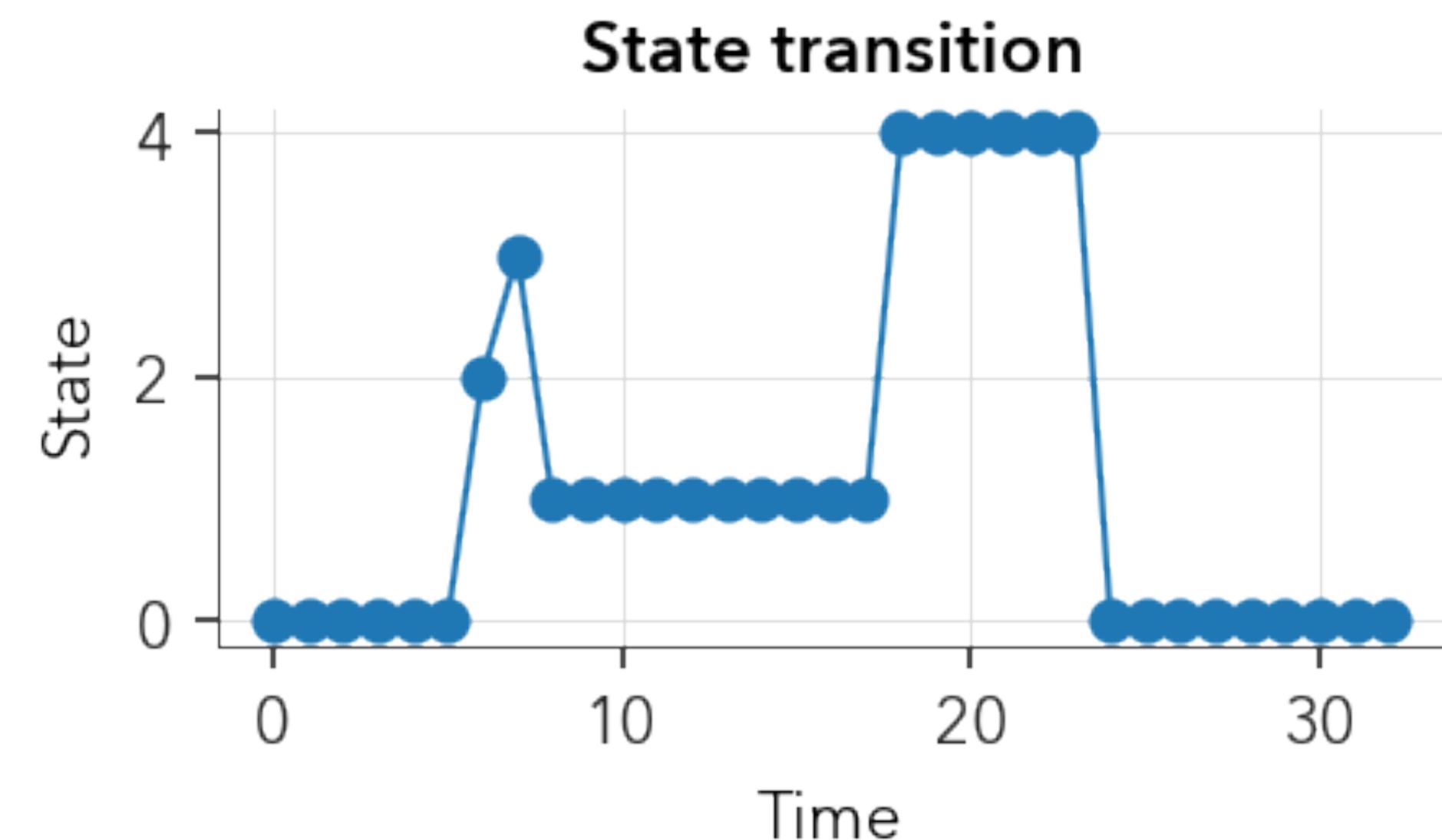
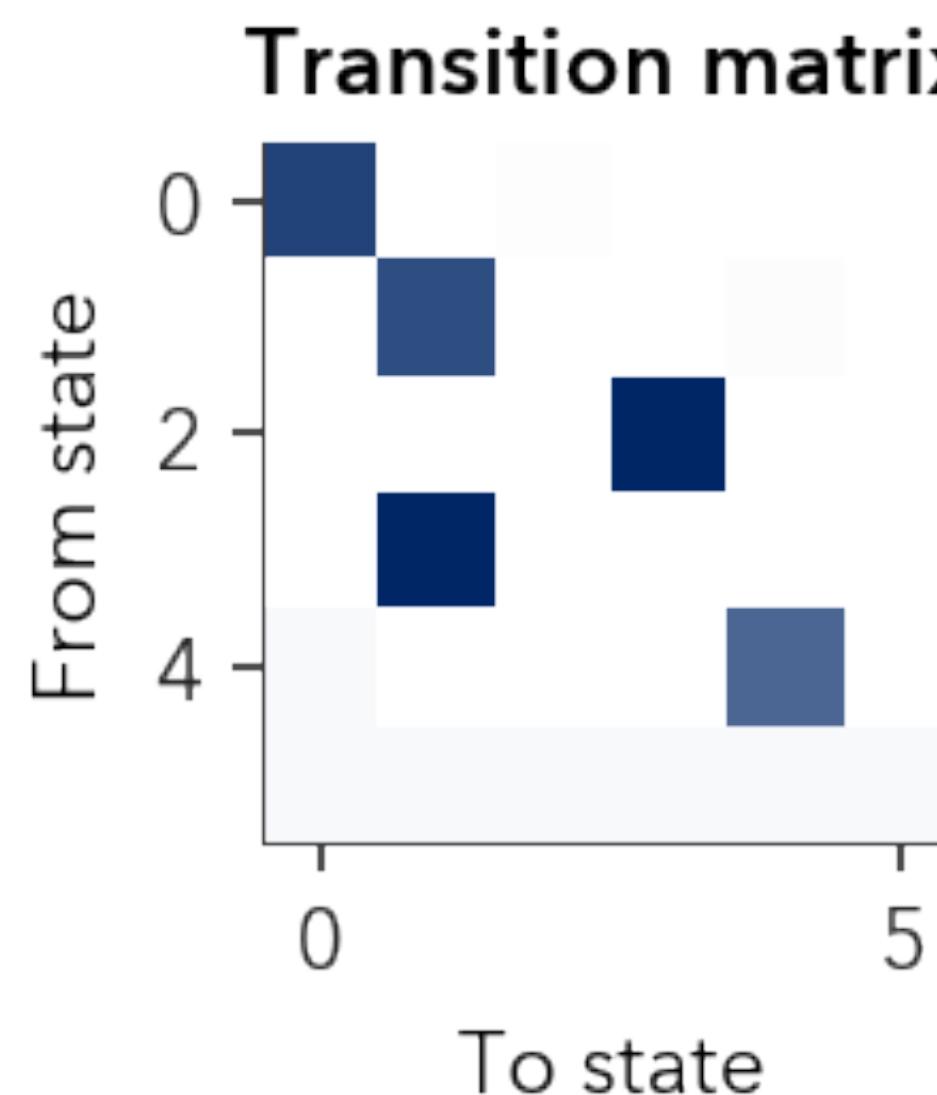
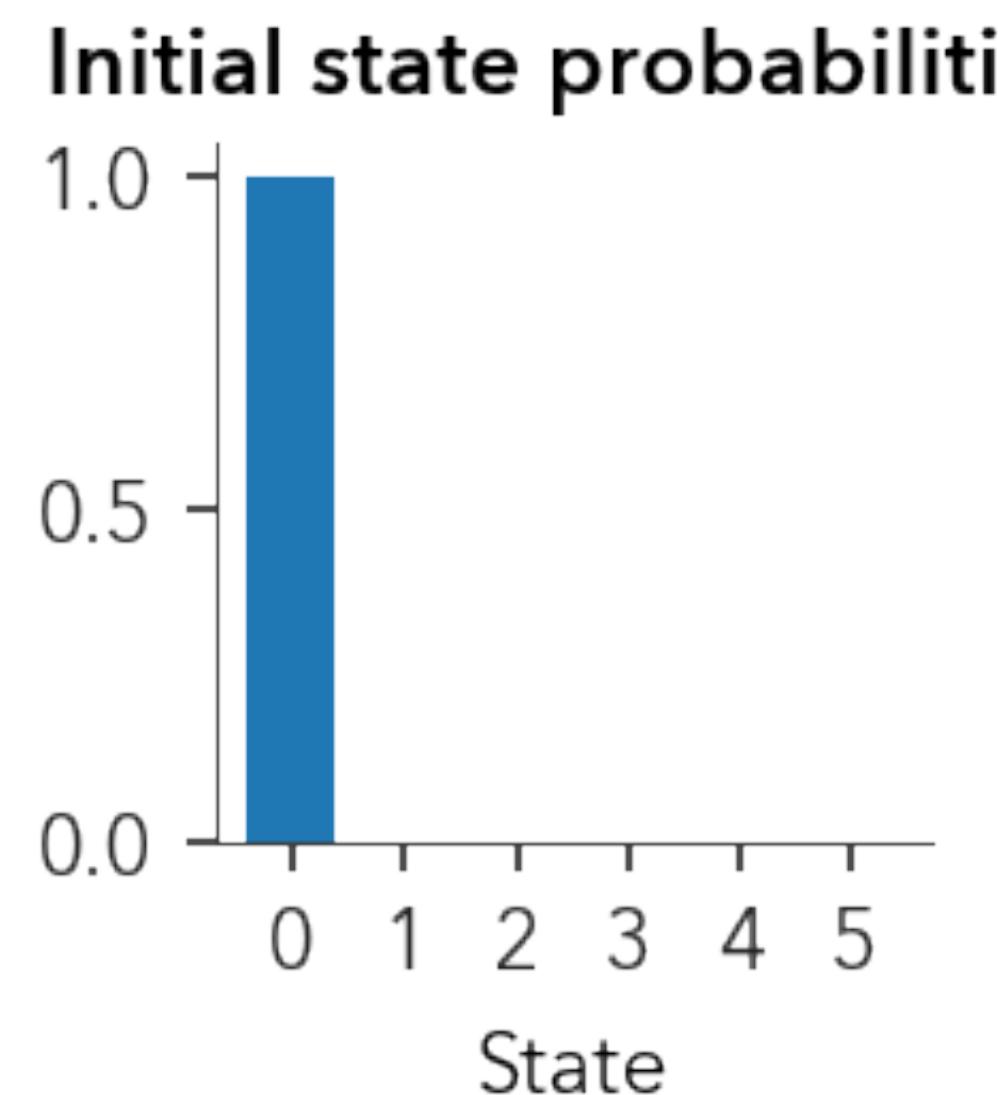
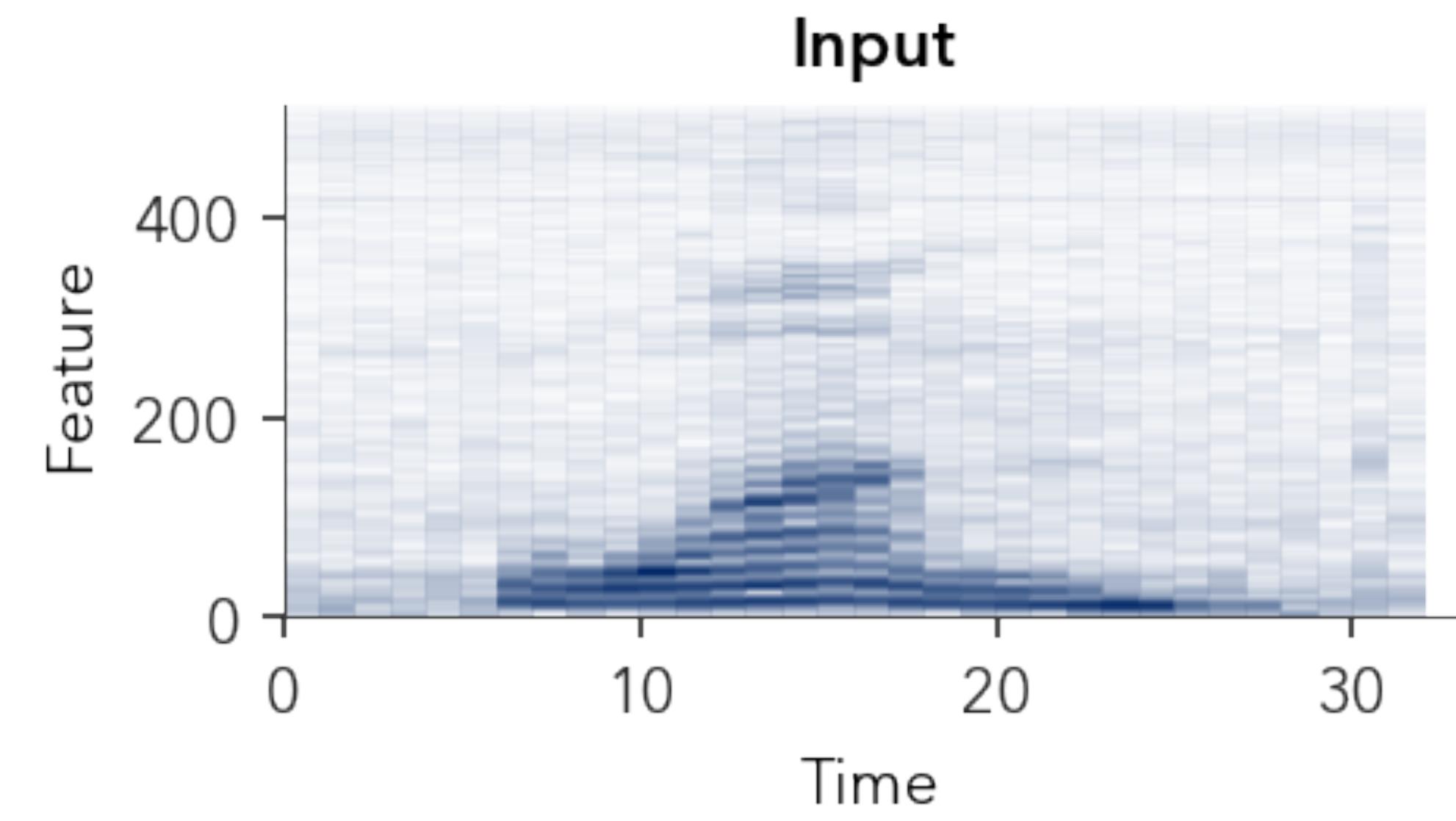
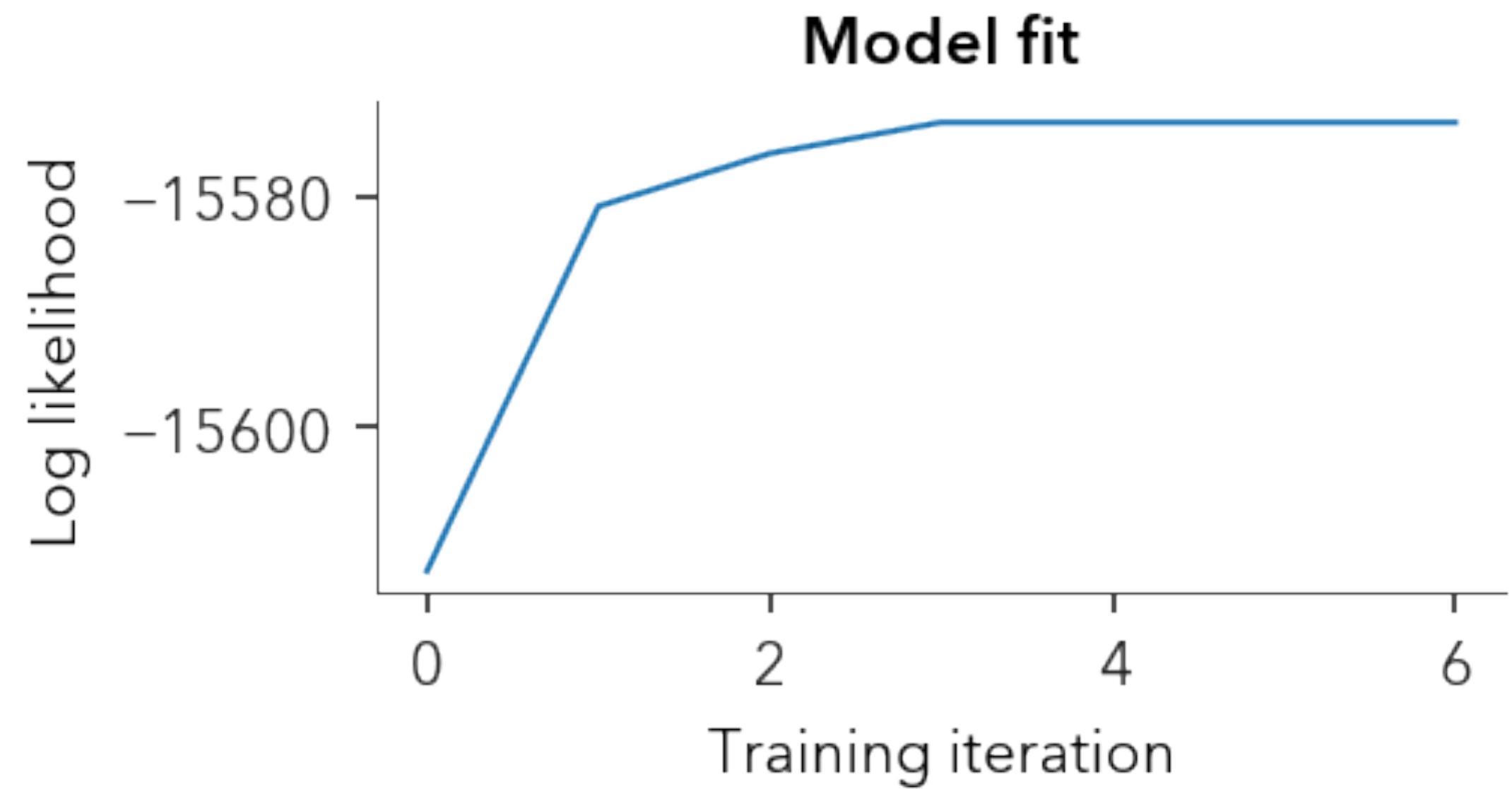


Back to speech

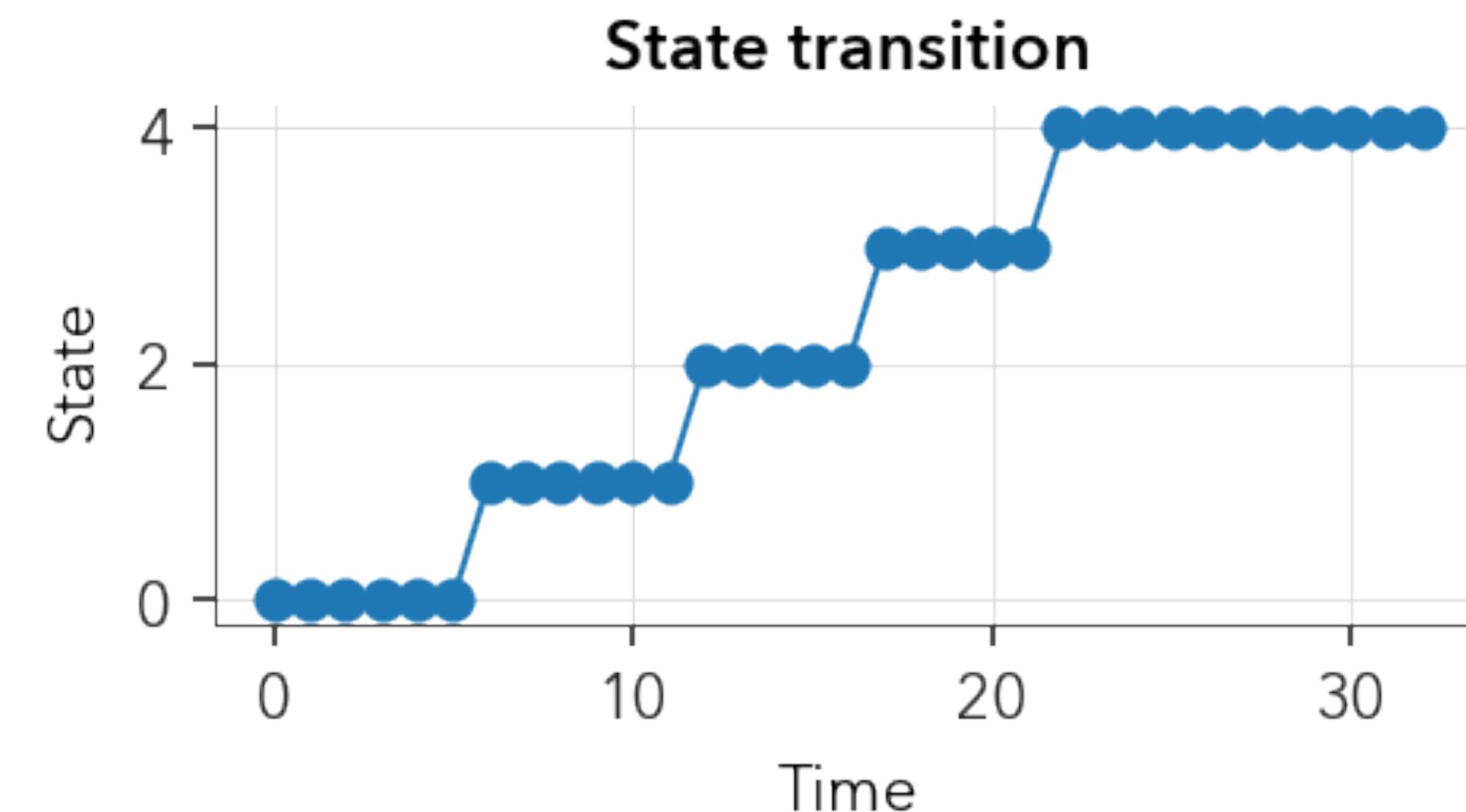
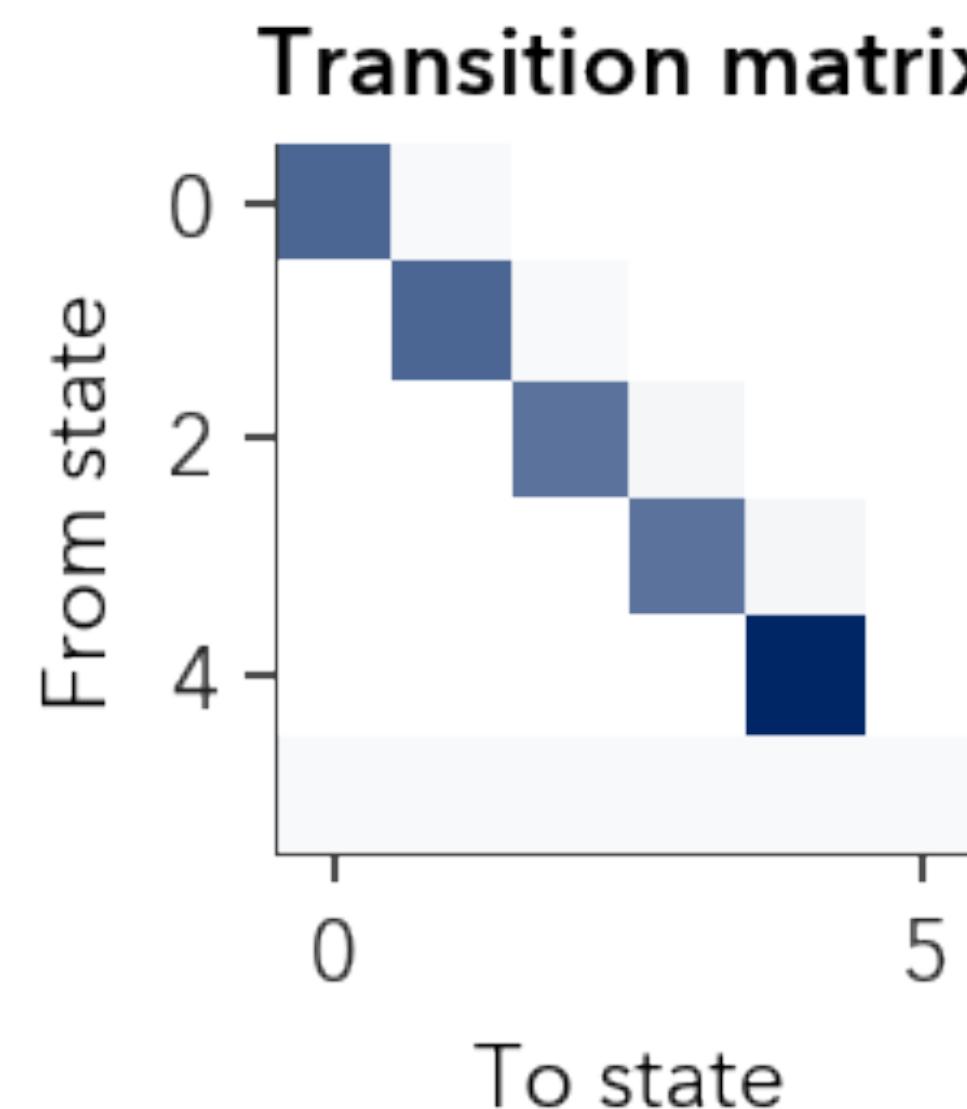
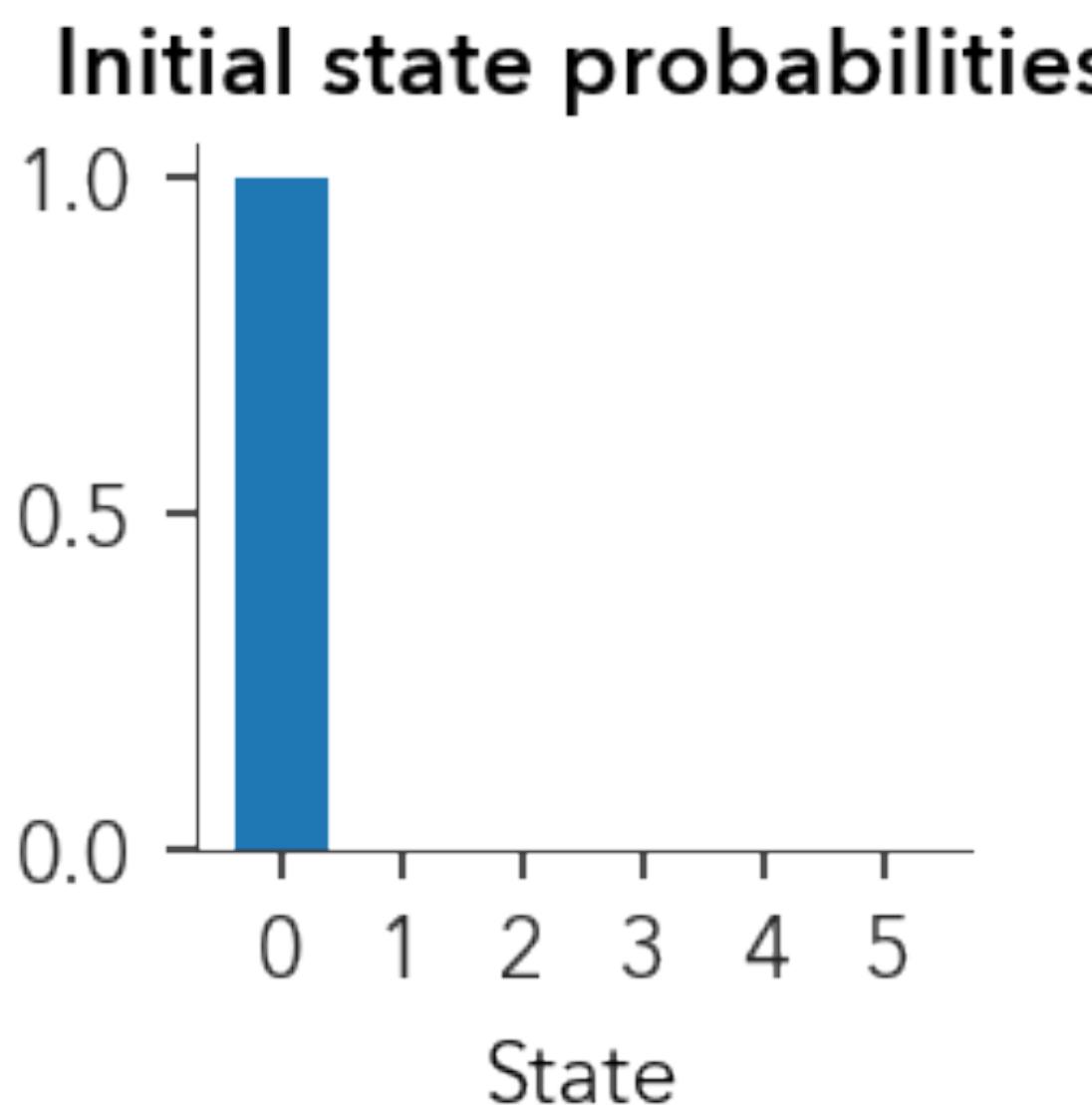
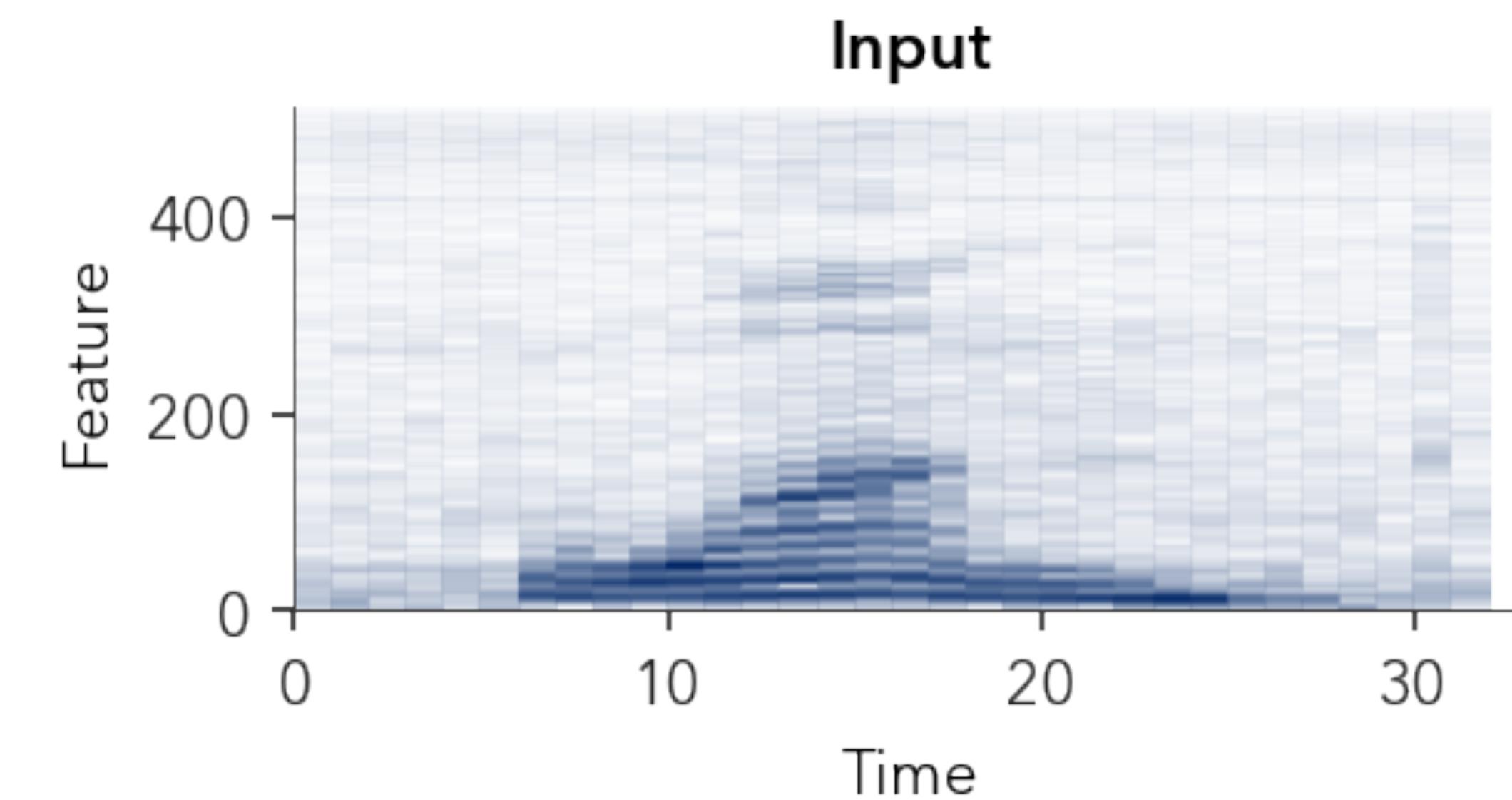
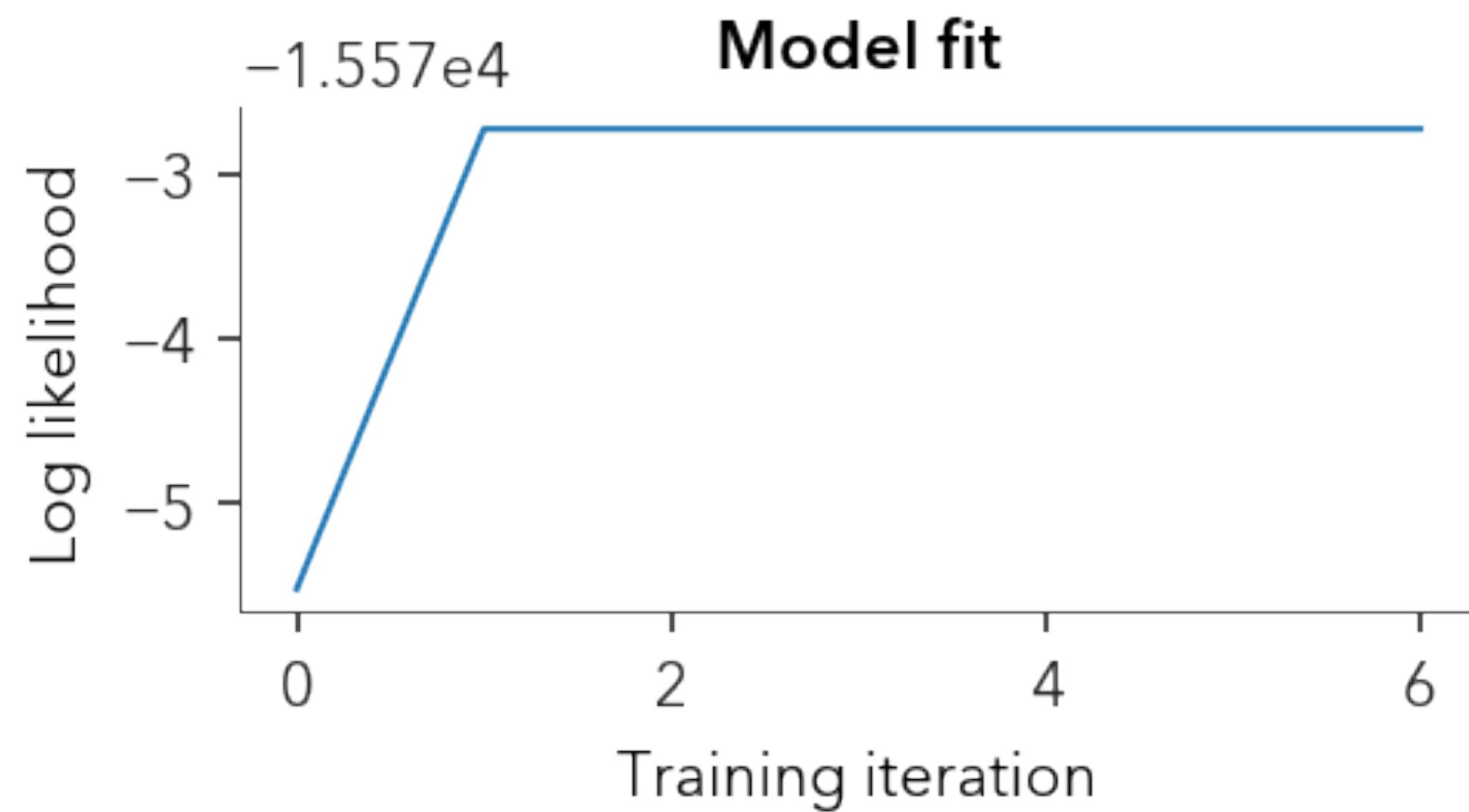
- Learning an HMM for a word
 - Each state should correspond to a coherent part
 - e.g. a syllable, a phoneme, etc



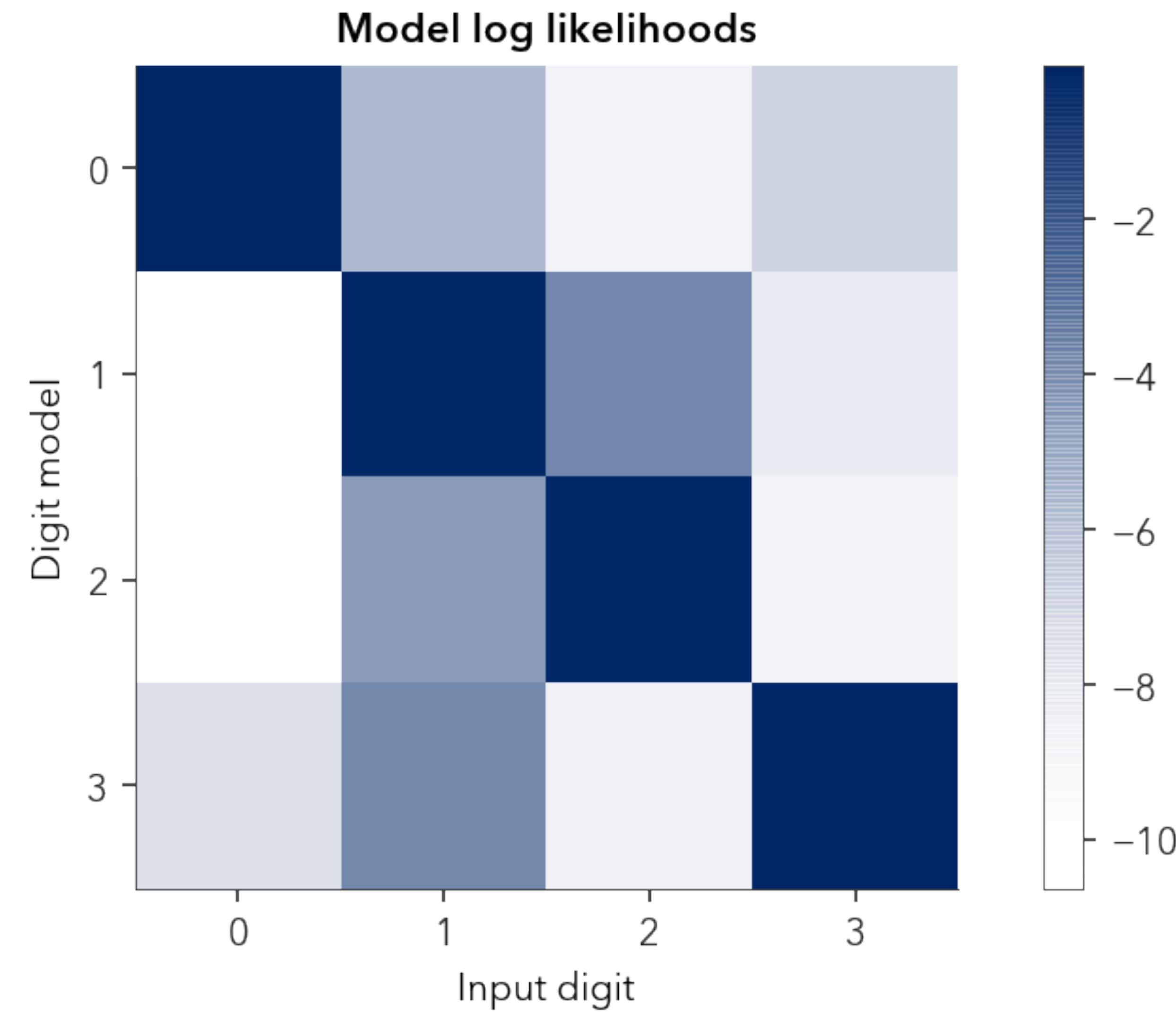
Fully-connected model



Left-to-right model

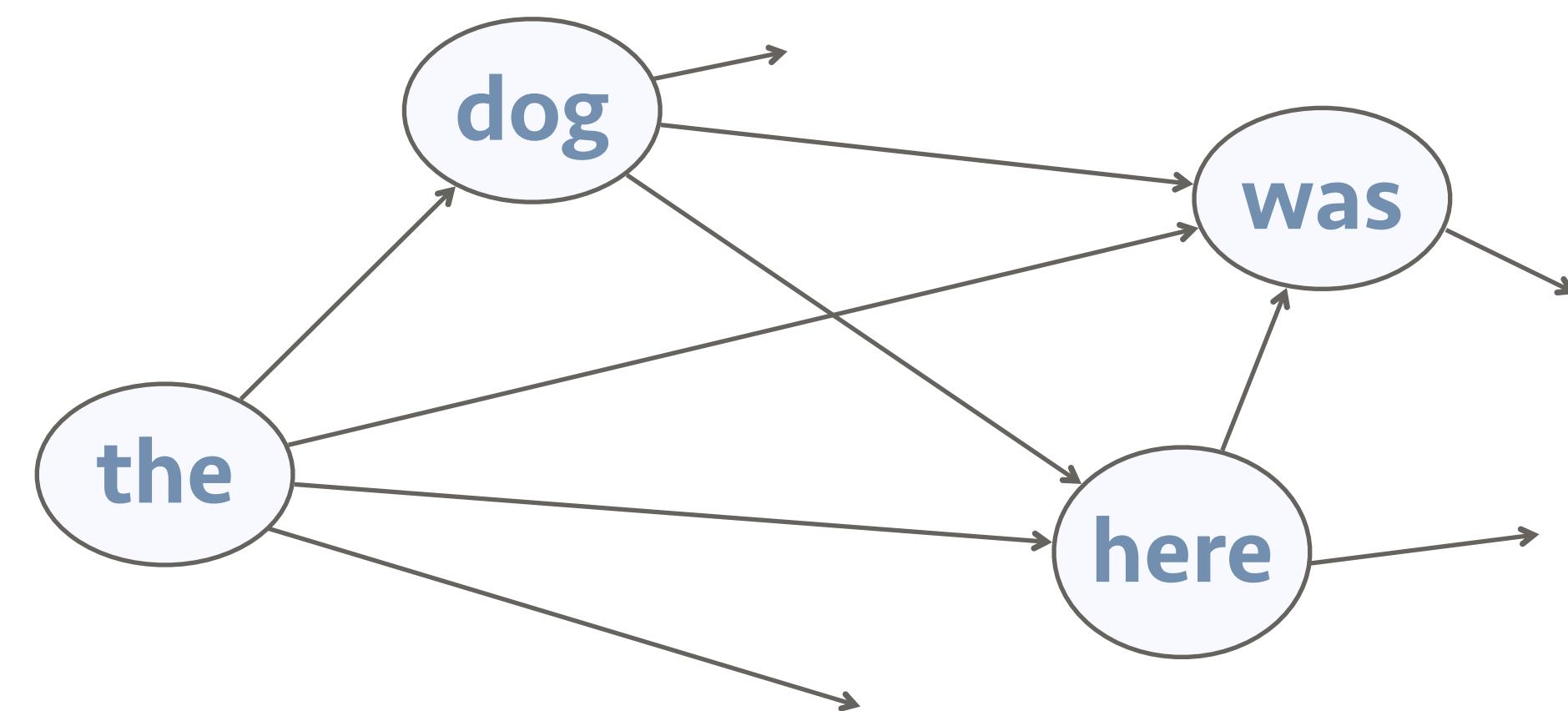


Digit recognition results



Speech recognition in a nutshell

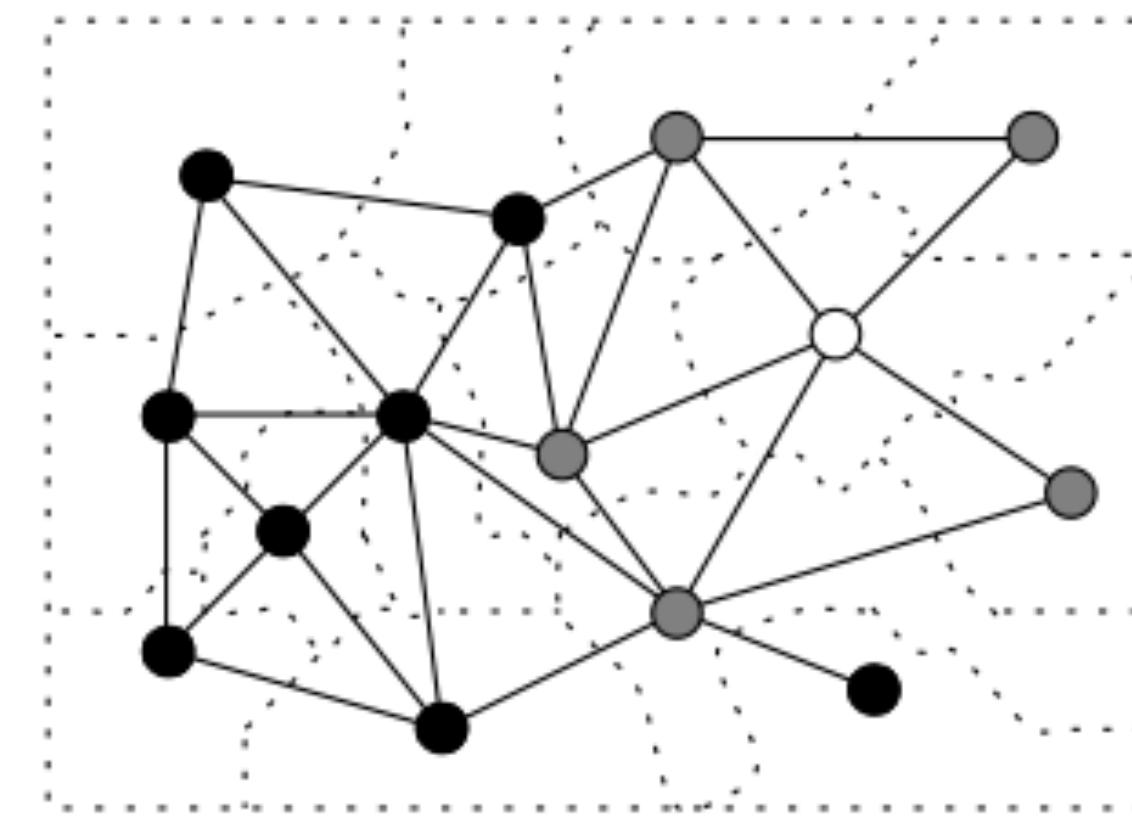
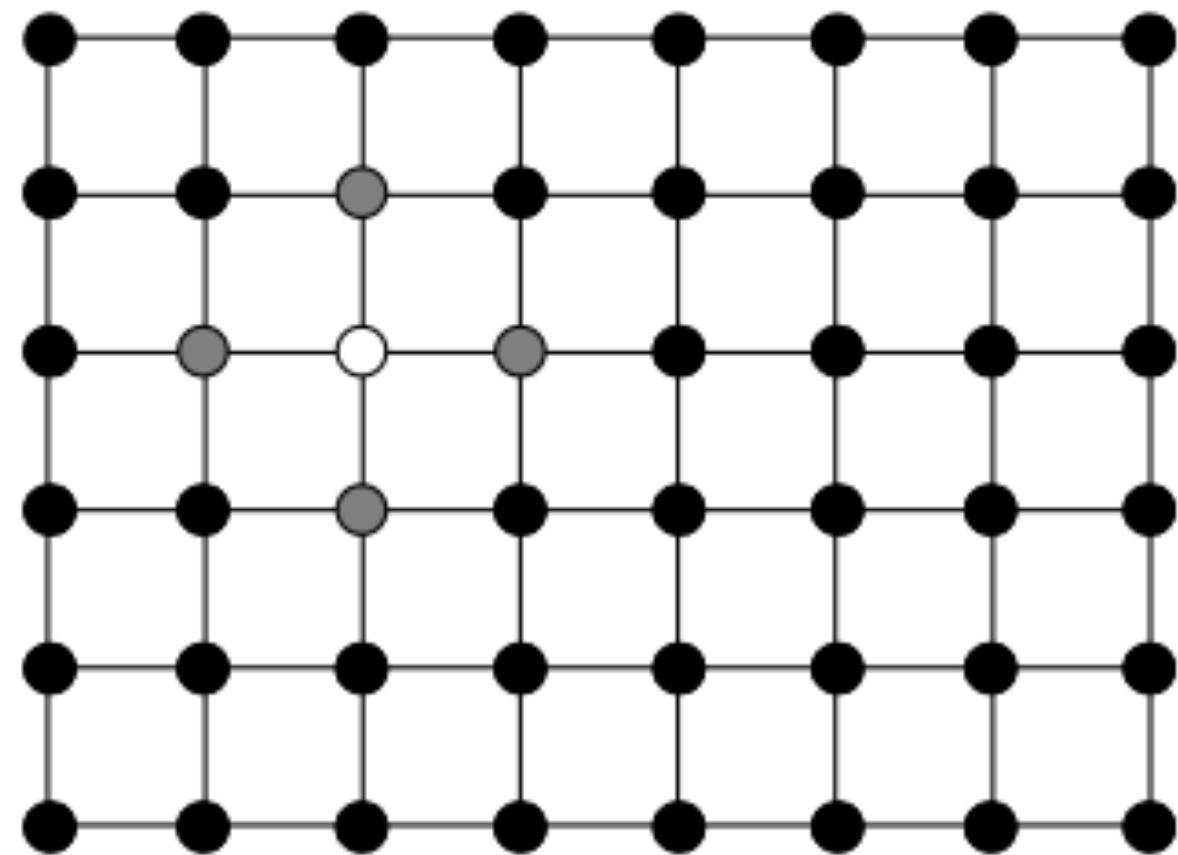
- Small scale speech recognition



- Large scale systems are another beast ...

Elaborations

- Denser interconnections
 - e.g., second order time ties
- Markov random fields



Recap

- Learning with time series
- Dynamic Time Warping
- Hidden Markov Models
- Some basic speech recognition

Next lecture

- Missing data
 - Using temporal dynamics to fill in missing values
- Dynamical systems
 - Learning continuous time-series

Reading

- Textbook chapters 8, 9
- A tutorial on HMMs
 - <http://www.cs.ubc.ca/~murphyk/Bayes/rabiner.pdf>

Some project advice

- Work on a domain you have expertise in (if any)
 - Bring your own (signal) data, don't stick to generic projects
- Show me that you learned something in class
 - Demonstrate proper use of the tools we cover, use some signals understanding
- Propose multiple objectives, see how far you can go
 - One objective will be really easy to do (you pass the class)
 - Another will be tough, but doable (you get a potential paper out of it)
 - Final should be an amazing result (here's your dissertation!)