Learning Time Series

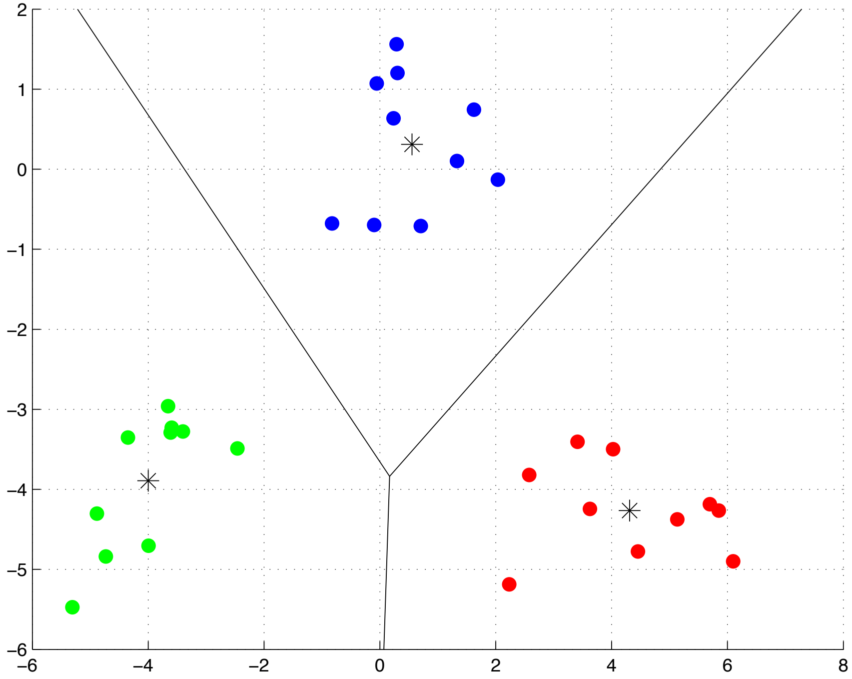
CS498

Today's lecture

- Doing machine learning on time series
- Dynamic Time Warping
- Simple speech recognition

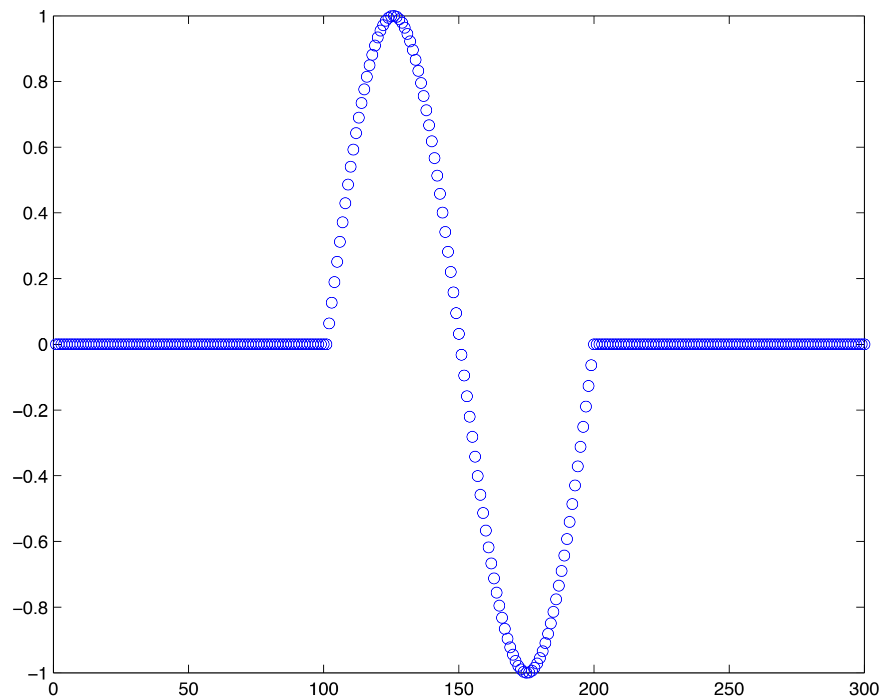
What we can do

- Data are points in a high-d space



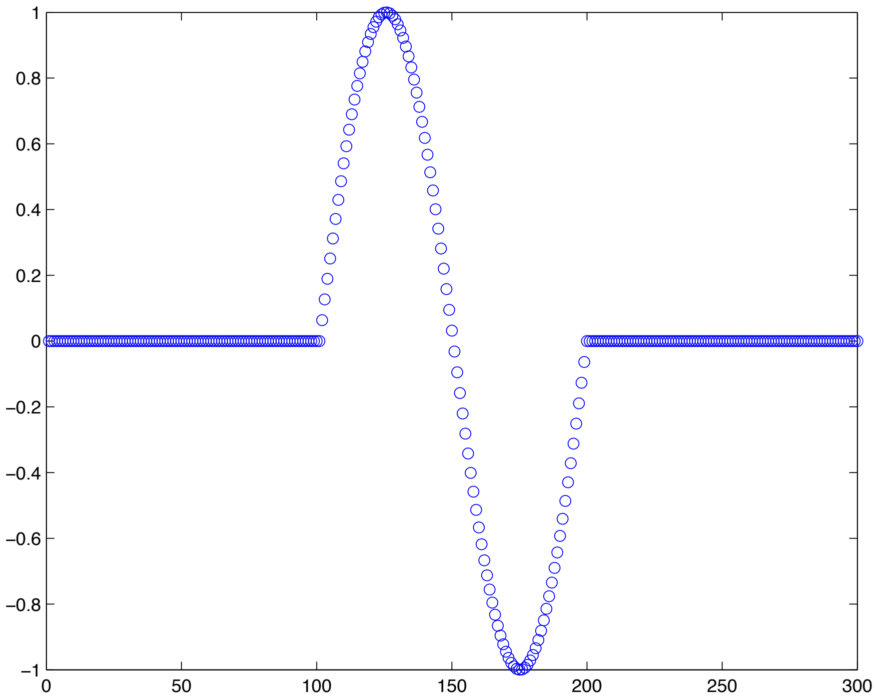
What time series are

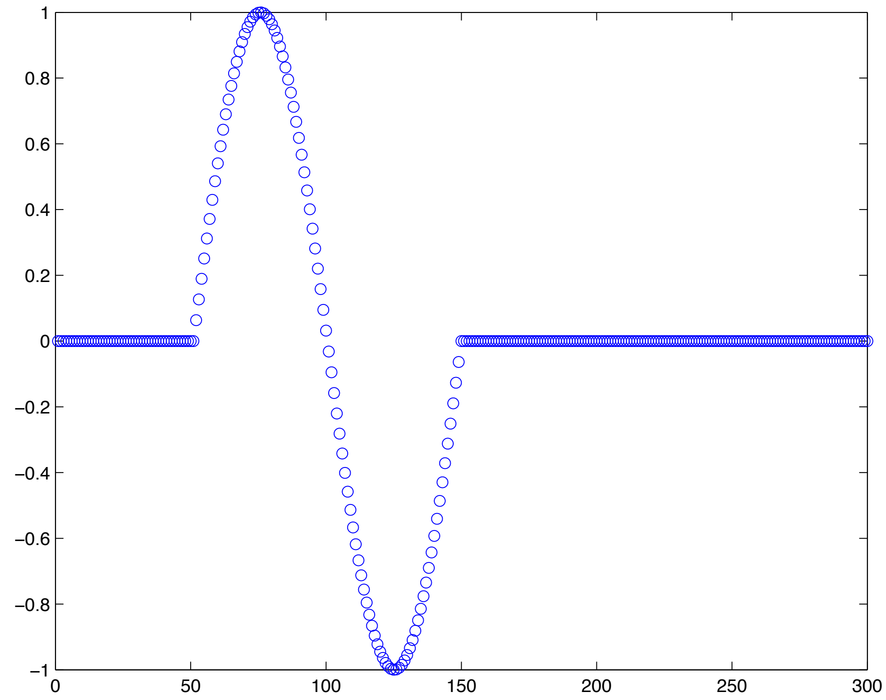
- Lots of points, can be thought of as a point in a very very high-d space
  - Bad idea ….



Shift variance

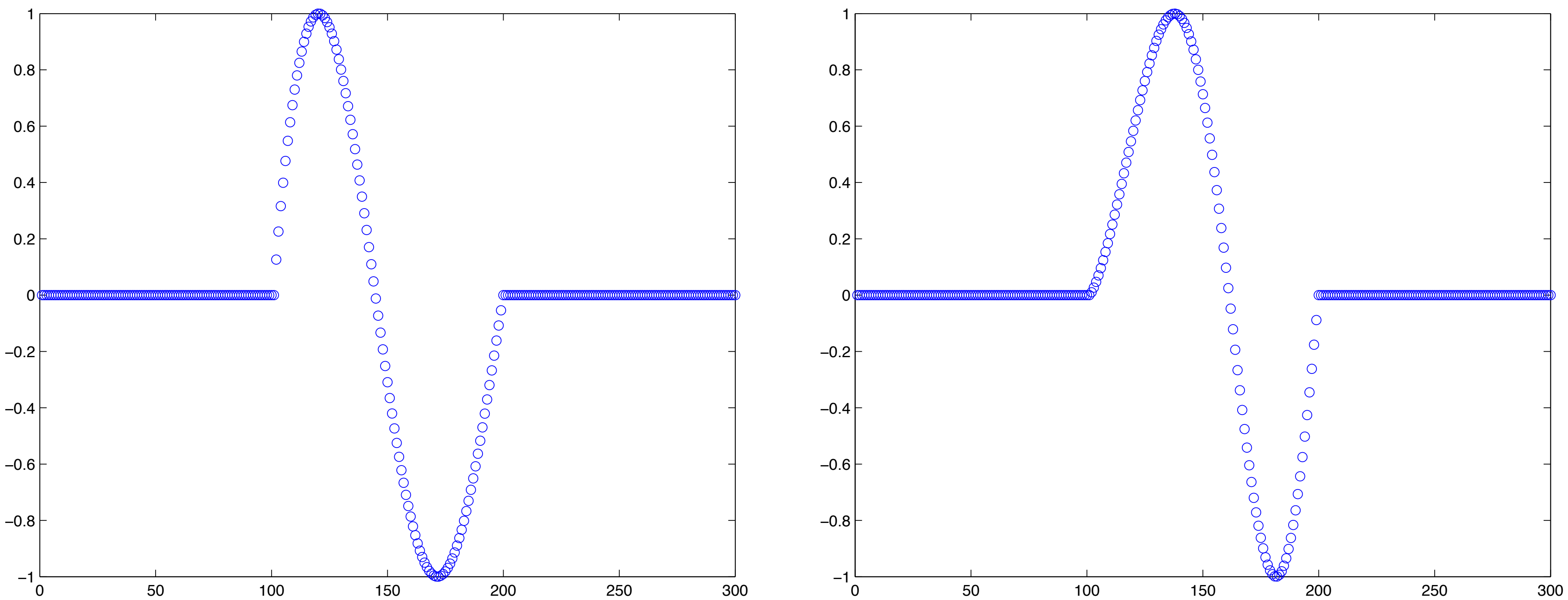
- Time series have shift variance
  - Are these two points close?





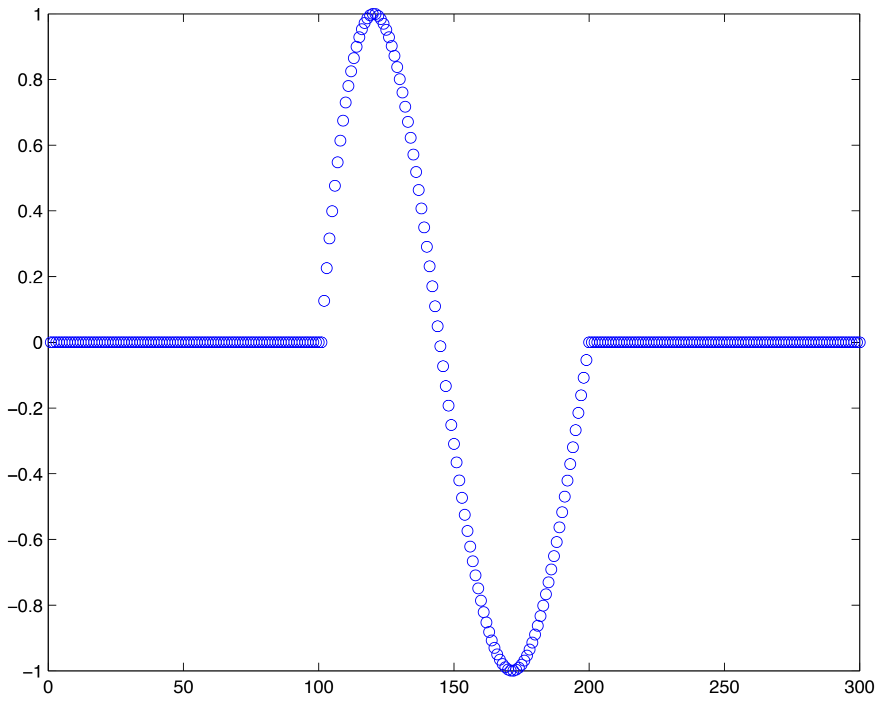
Time warp variance

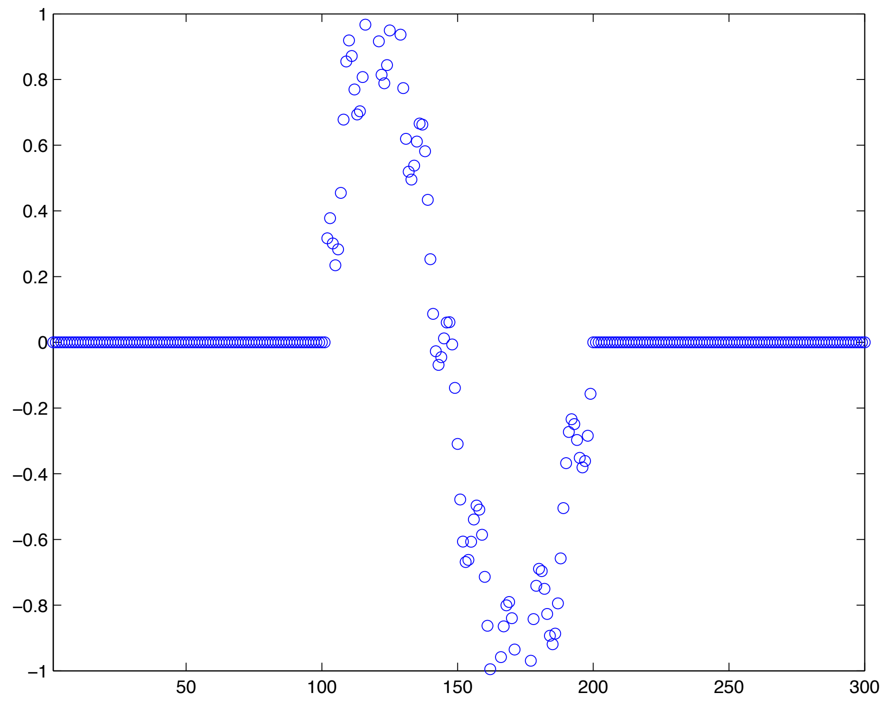
- Slight changes in timing are not relevant
  - Are these two point close?



Noise/filtering variance

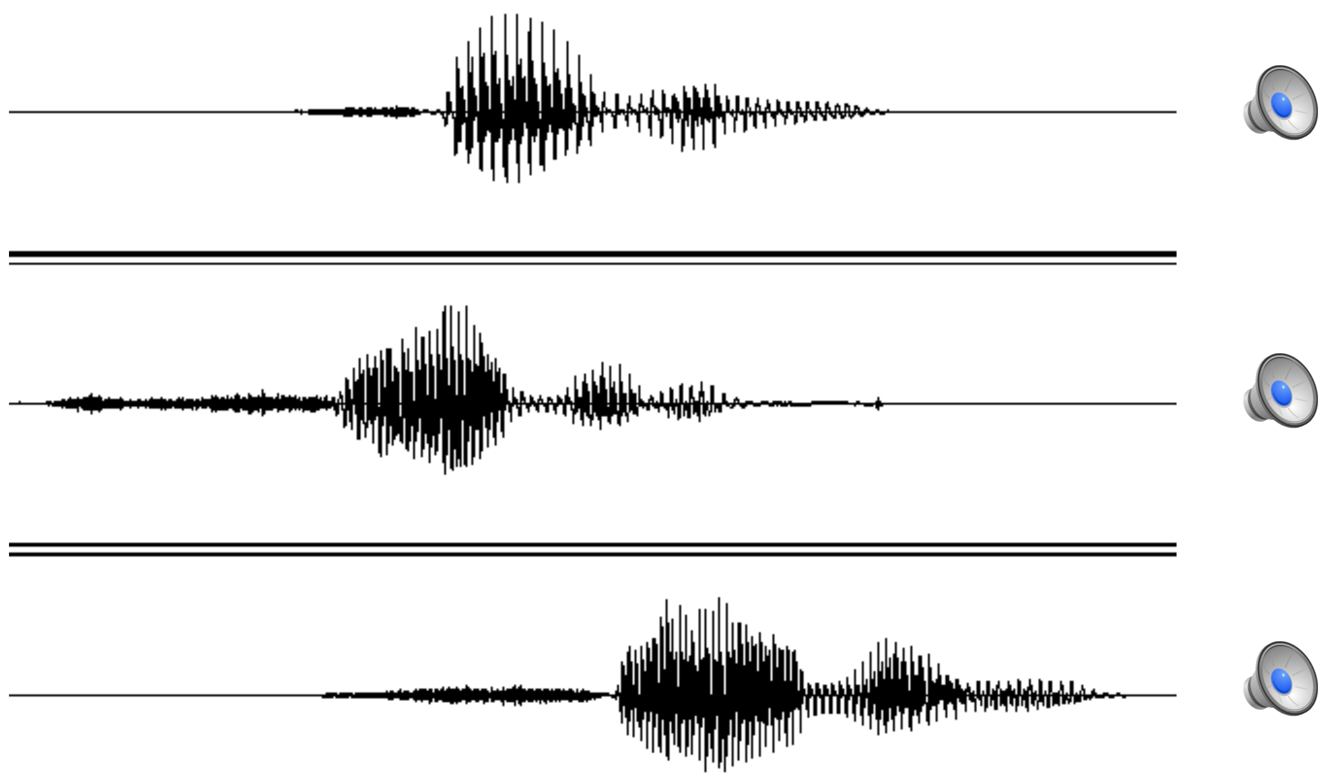
- Small changes can look serious
  - How about these two points?





A real-world case

- Spoken digits



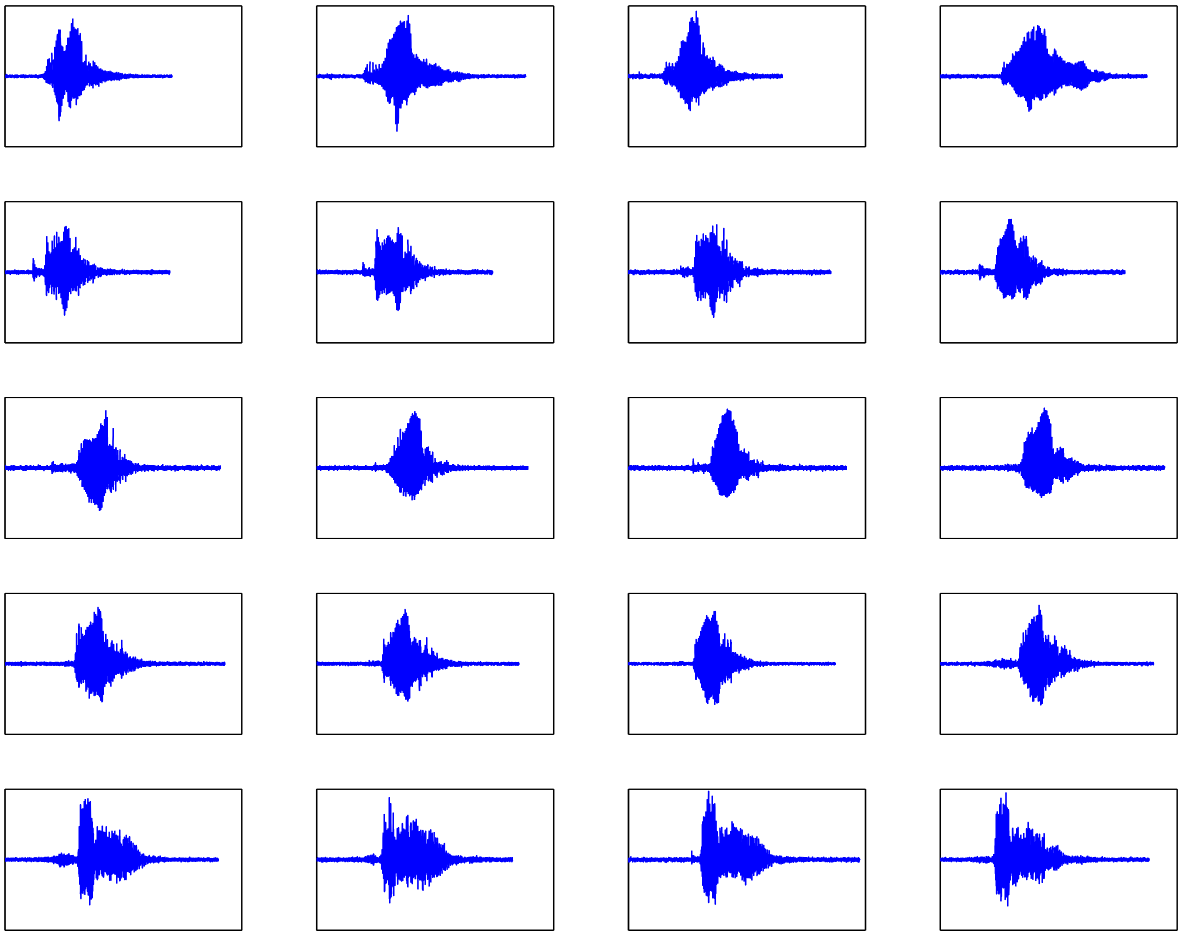
What now?

- Our models so far were too simple
- How do we incorporate time?
- How to 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 do time
  - Invariant to temporal differences in inputs

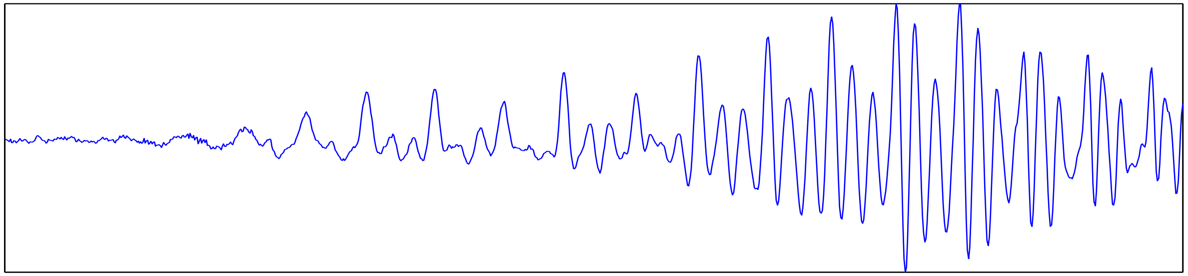
Example data

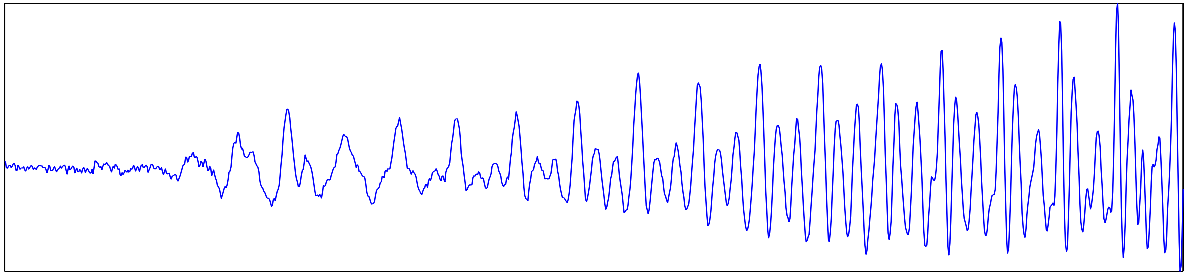


ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Going from fine to coarse

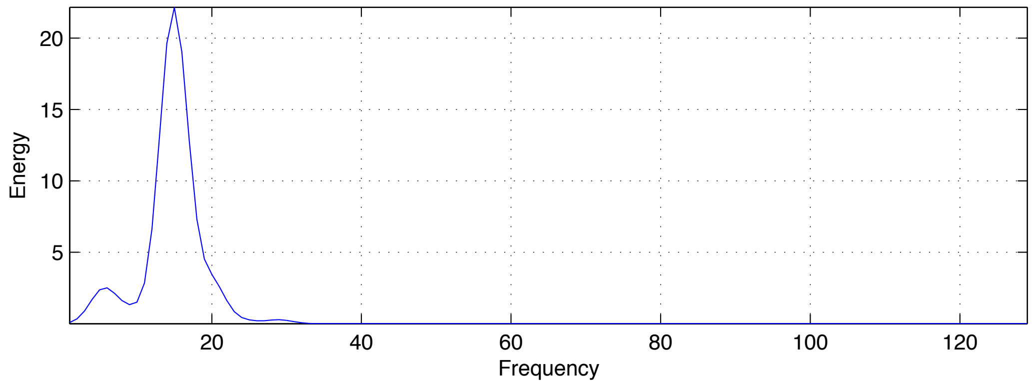
- Small differences are not important
  - Find features that obscure them

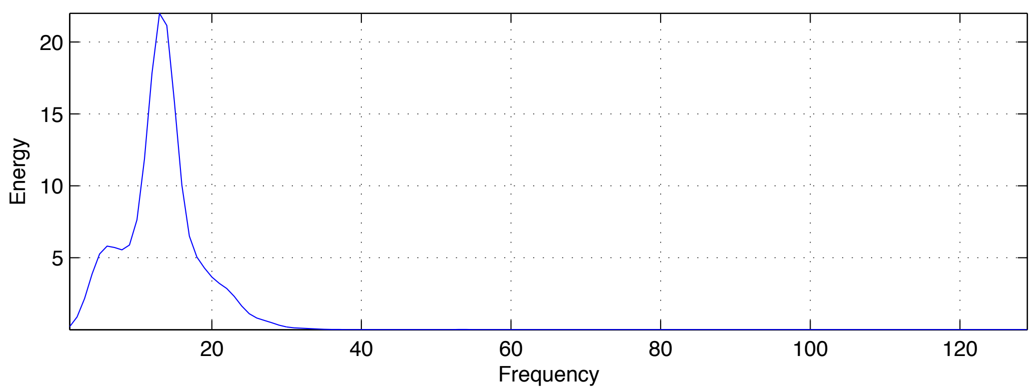




Frequency domain

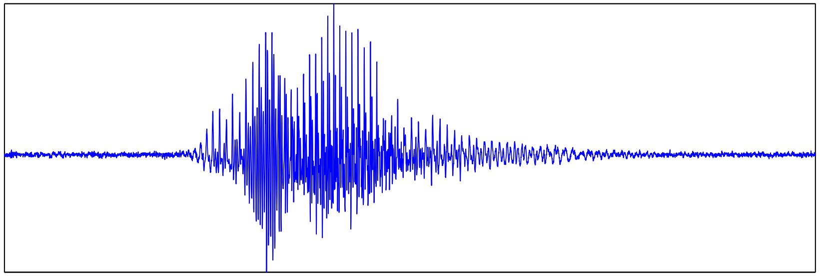
- Look at the magnitude Fourier transform

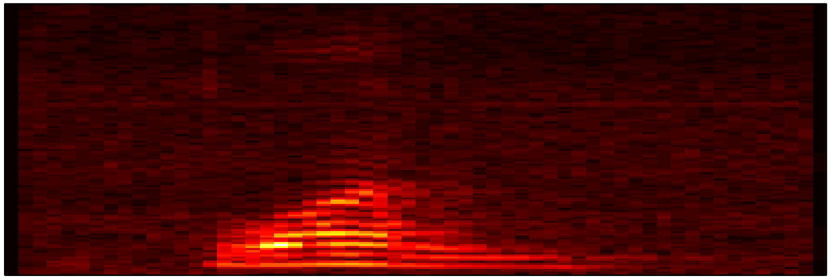


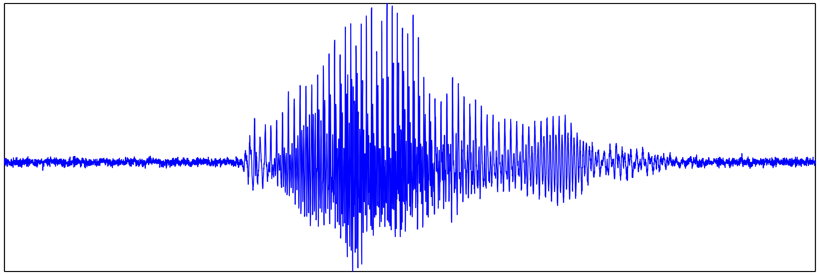


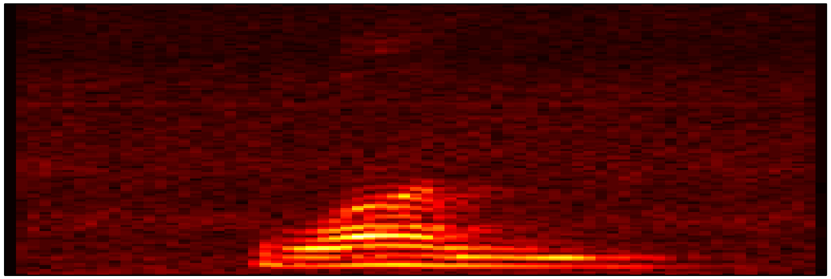
Time/Frequency features

- A more robust representation
  - Bypassing minute waveform differences



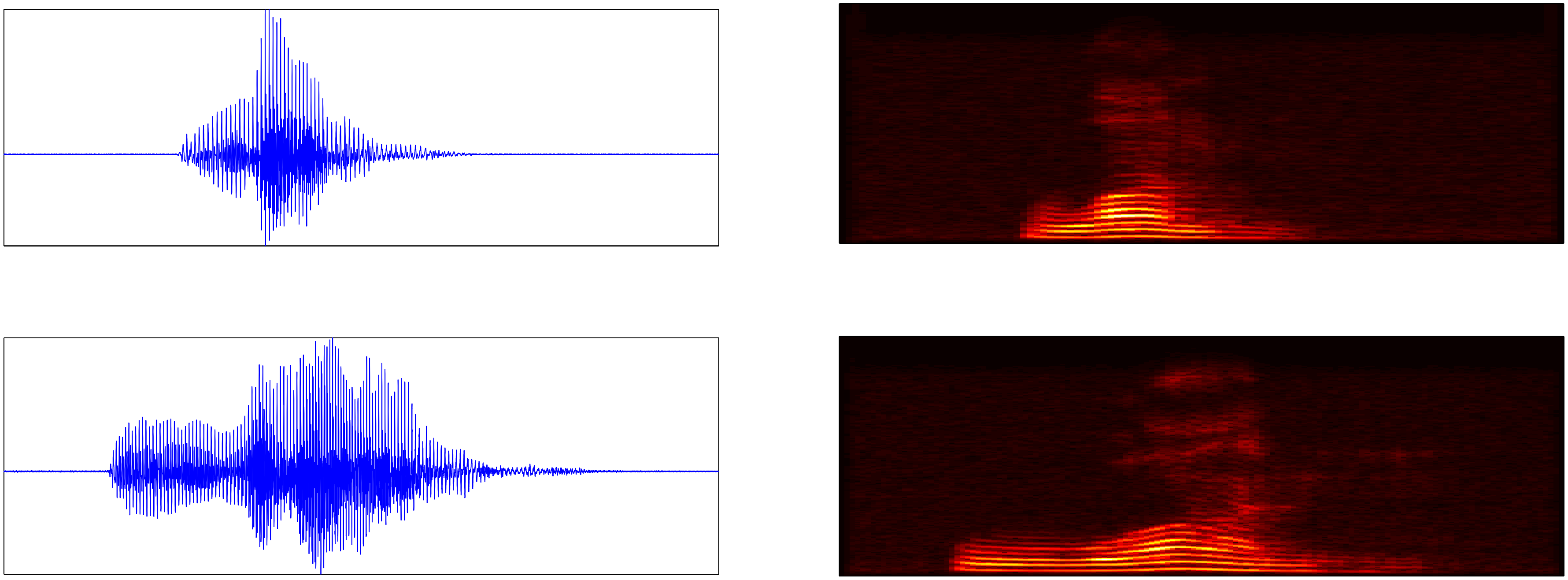






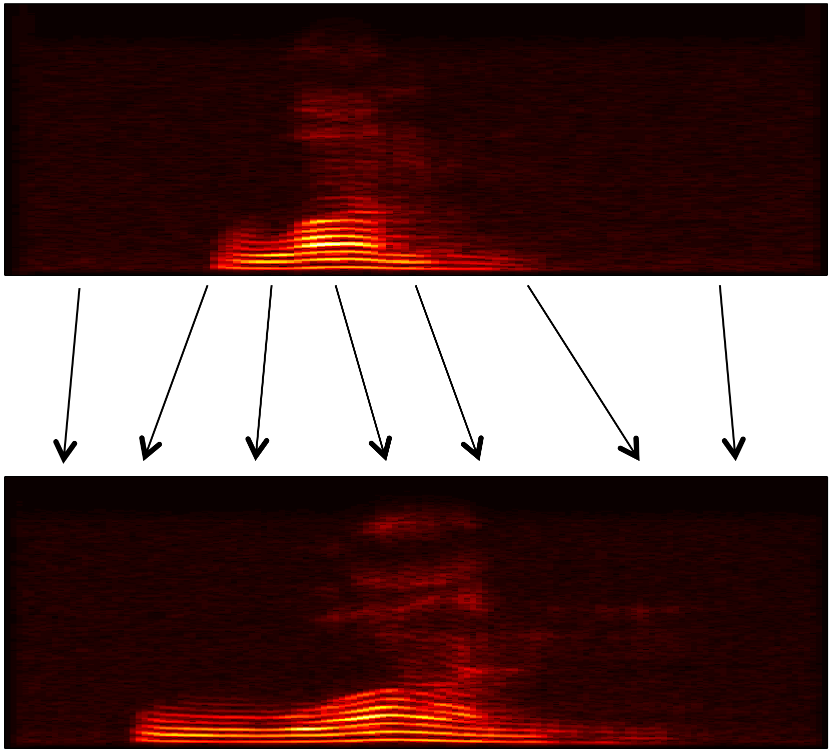
A new problem

- What about time warping?



Time warping

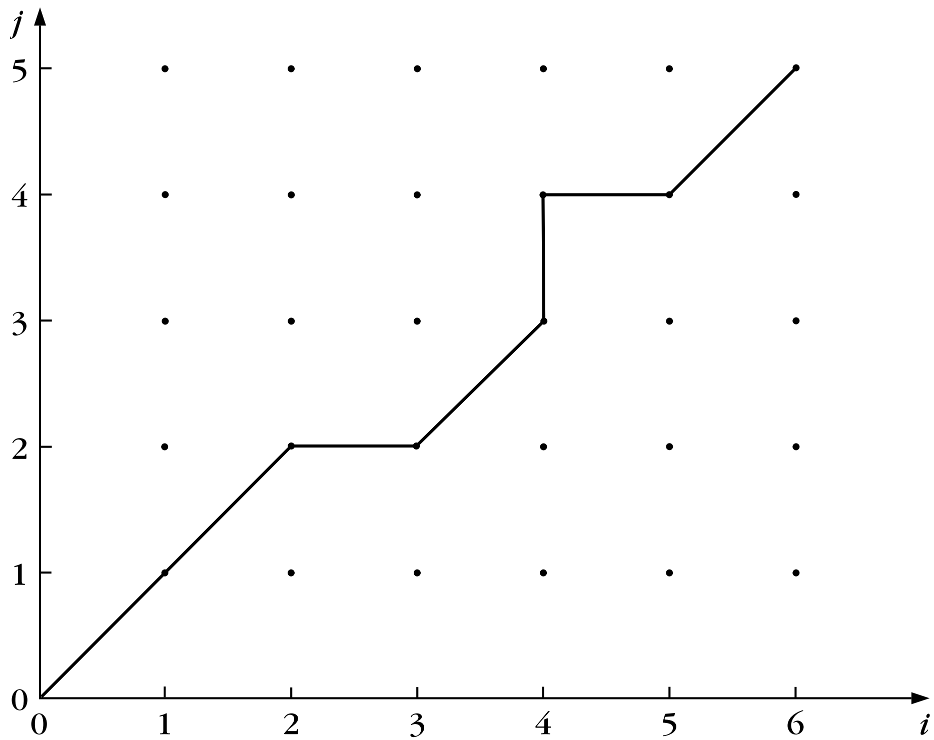
- There is a "warped" time map
  - How do we find it?



Matching warped series

- Represent the warping with a path

$$\mathbf{r}(i), i = 1, 2, \ldots, 6 \quad \mathbf{t}(j), j = 1, 2, \ldots, 5$$



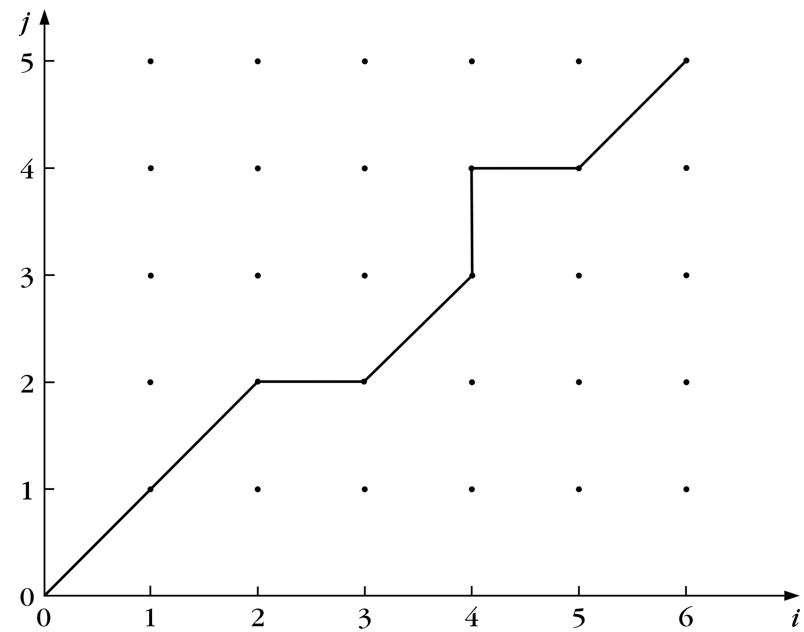
Finding the overall “distance”

- Each node will have a *cost*
  - e.g., $d(i,j) = \left\| \mathbf{r}(i) - \mathbf{t}(j) \right\|$

- Overall path *cost* is:

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

- Optimal D path defines the “distance” between two given sequences



ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

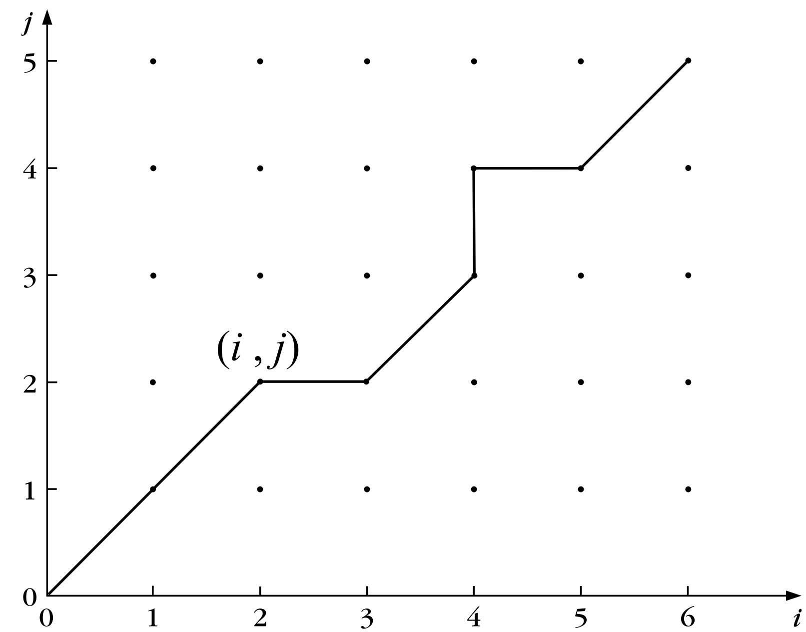
Bellman's optimality principle

- For an optimal path passing through $(i\ ,\ j)$:

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

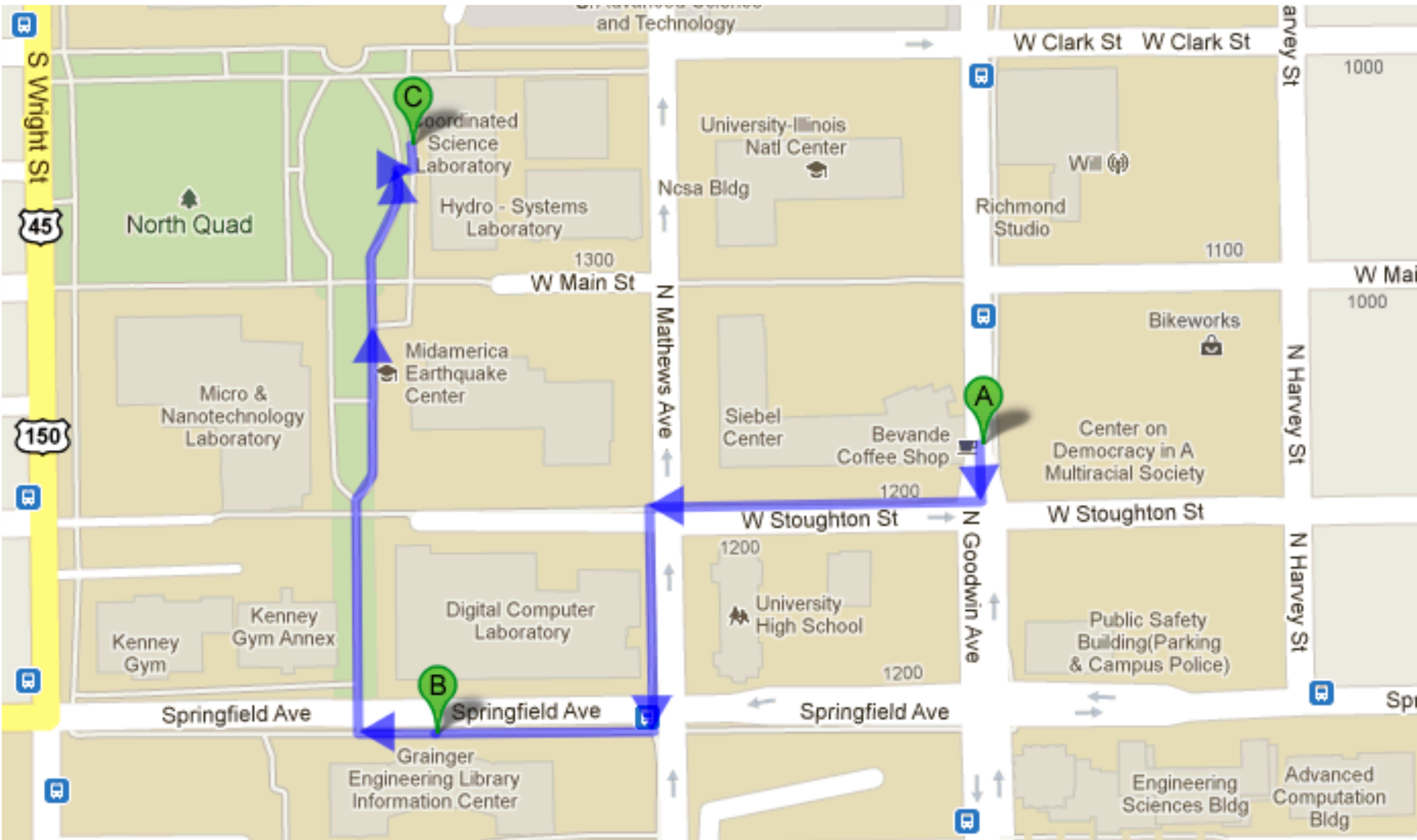
- Then:

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



ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

In real-life



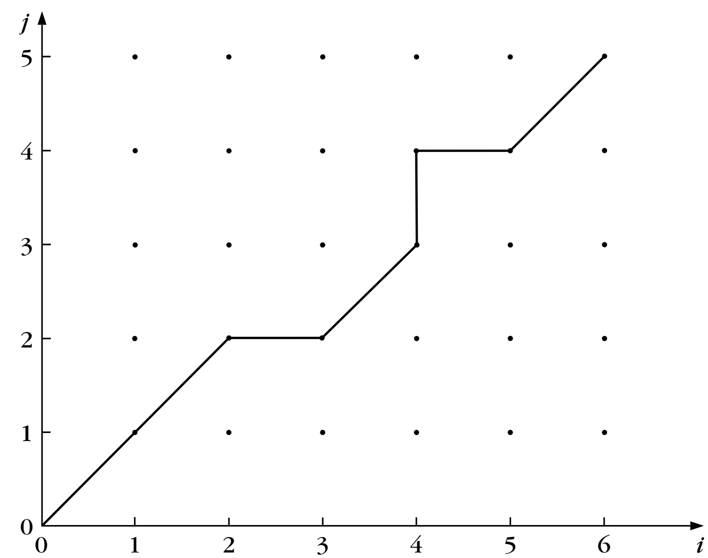
and Technology
W Clark St
W Clark St
S Wright St
45
150
North Quad
Coordinated Science Laboratory
Hydro - Systems Laboratory
University-Illinois Natl Center
Ncsa Bldg
Will
Richmond Studio
1000
1100
1300
W Main St
N Mathews Ave
1000
Bikeworks
Midamerica Earthquake Center
Micro & Nanotechnology Laboratory
Siebel Center
Bevande Coffee Shop
Center on Democracy in A Multiracial Society
N Harvey St
1200
W Stoughton St
W Stoughton St
N Goodwin Ave
1200
University High School
1200
Digital Computer Laboratory
Kenney Gym Annex
Kenney Gym
Public Safety Building(Parking & Campus Police)
Springfield Ave
Springfield Ave
Springfield Ave
Grainger Engineering Library Information Center
Engineering Sciences Bldg
Advanced Computation Bldg
A
B
C

Finding an optimal path

- Optimal path to $(i_k\ ,\ j_k)$:

$$D_{\min}(i_k, j_k) = \min_{i_k-1, j_k-1} D_{\min}(i_k - 1, j_k - 1) + d(i_k, j_k \mid i_k - 1, j_k - 1)$$

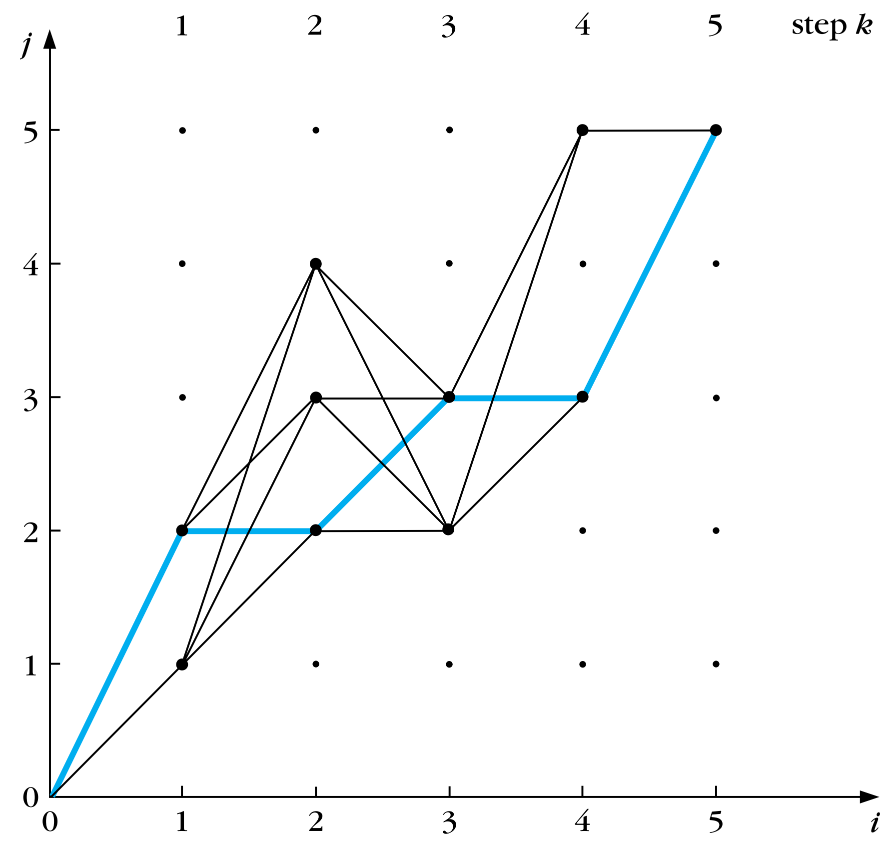
  - Smaller search!



- Local/global constraints
  - Limited transitions
  - Nodes we never visit

Example run

- Global constraints
  - bold dots
- Local constraints
  - Black lines
- Optimal path
  - Blue line



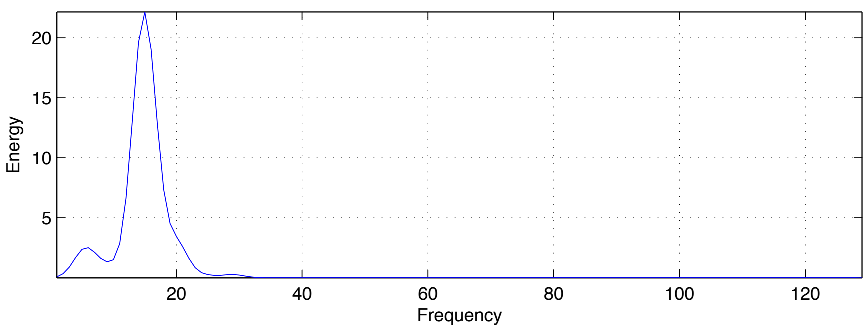
Making this work for speech

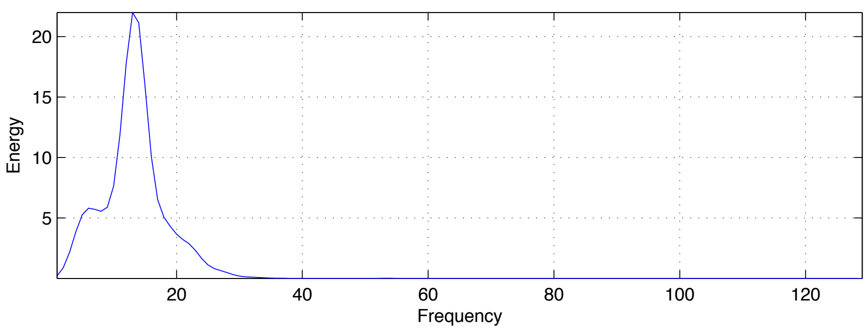
- Define a distance function
- Define local constraints
- Define global constraints

Distance function

- Given our robust feature we can use a simple measure like Euclidean distance

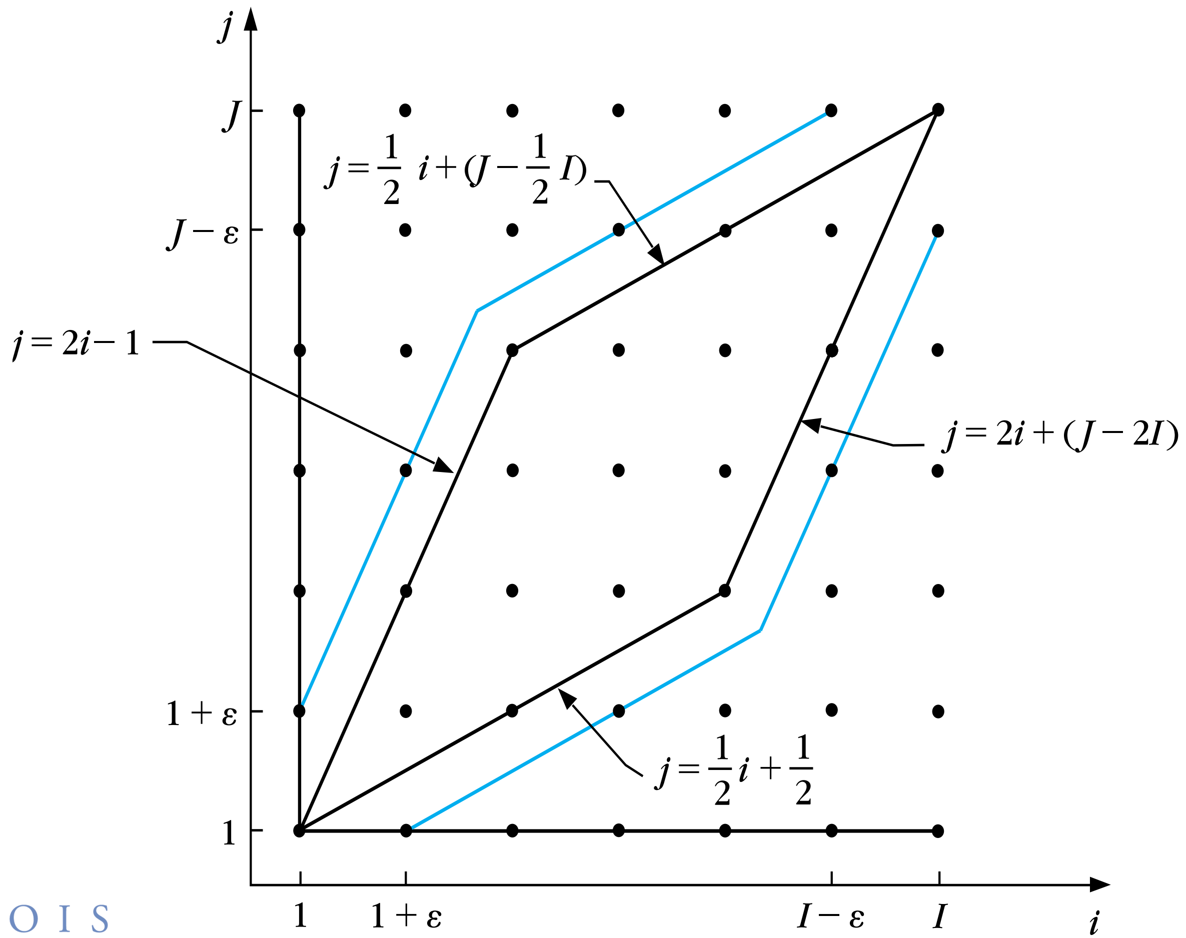
$$d(i,j) = \left\| \mathbf{f}_1(i) - \mathbf{f}_2(j) \right\|$$





Global constraints

- Define a ratio that is reasonable



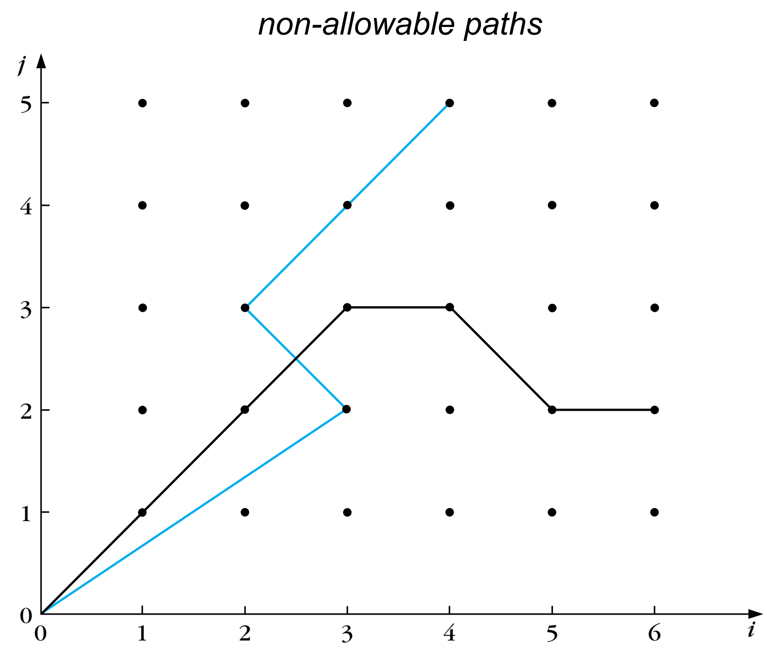
Local constraints

- Monotonicity

$$i_{k-1} \le i_k \qquad j_{k-1} \le j_k$$

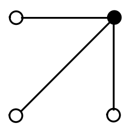
  – repeat but don't go back

- This enforces time order
  – don't get "cat" from "act"

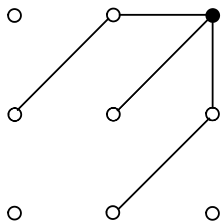


More local constraints

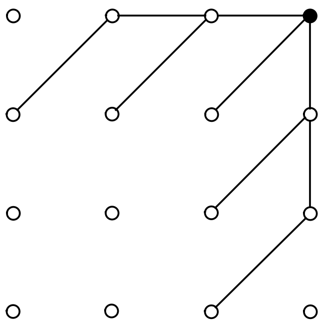
- Define acceptable paths
  - Application dependent



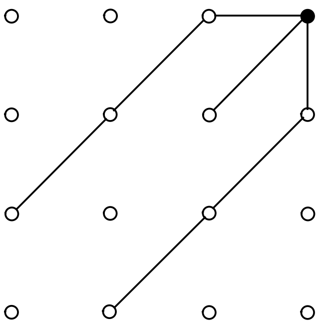
(a)



(b)



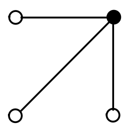
(c)

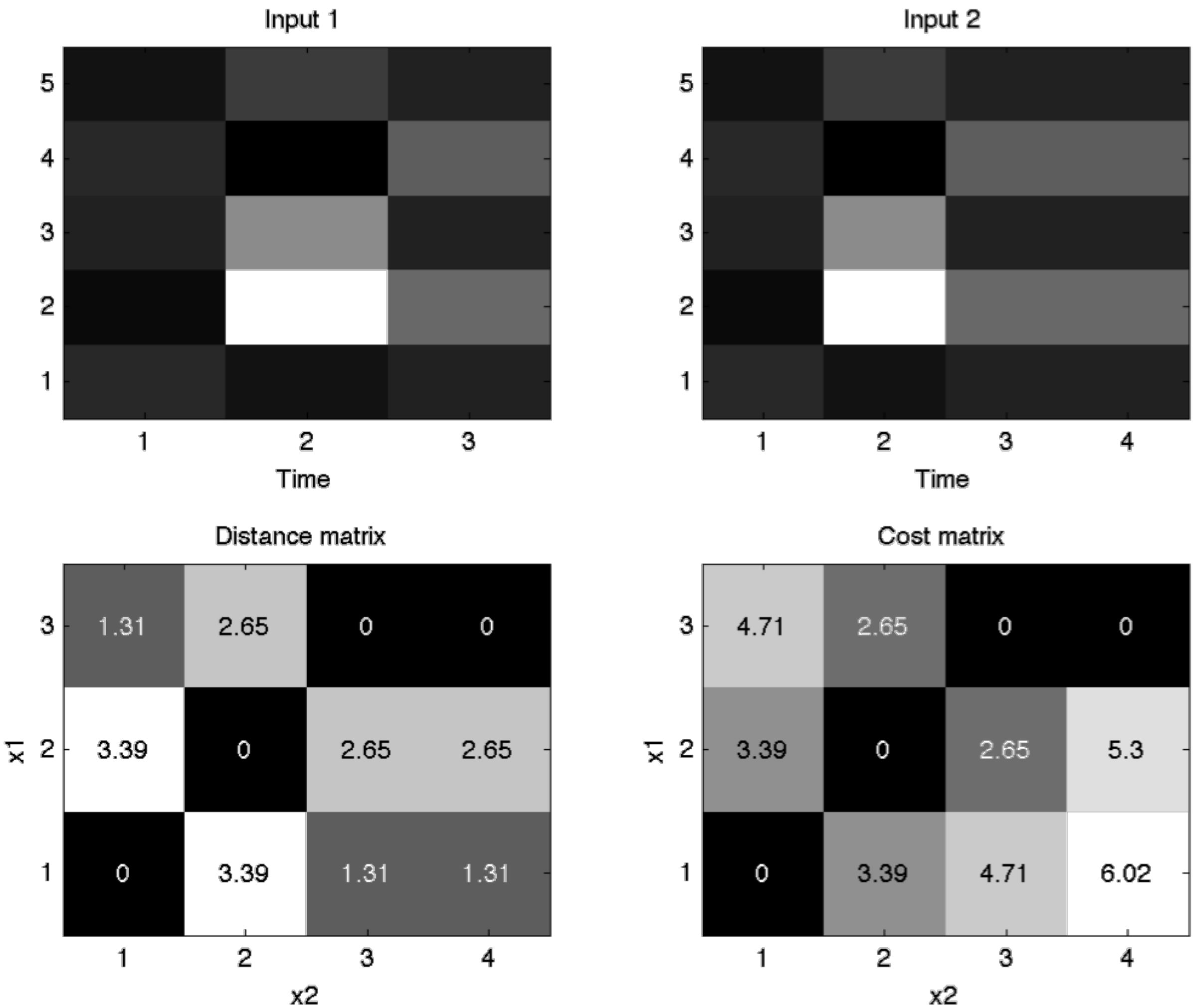


(d)

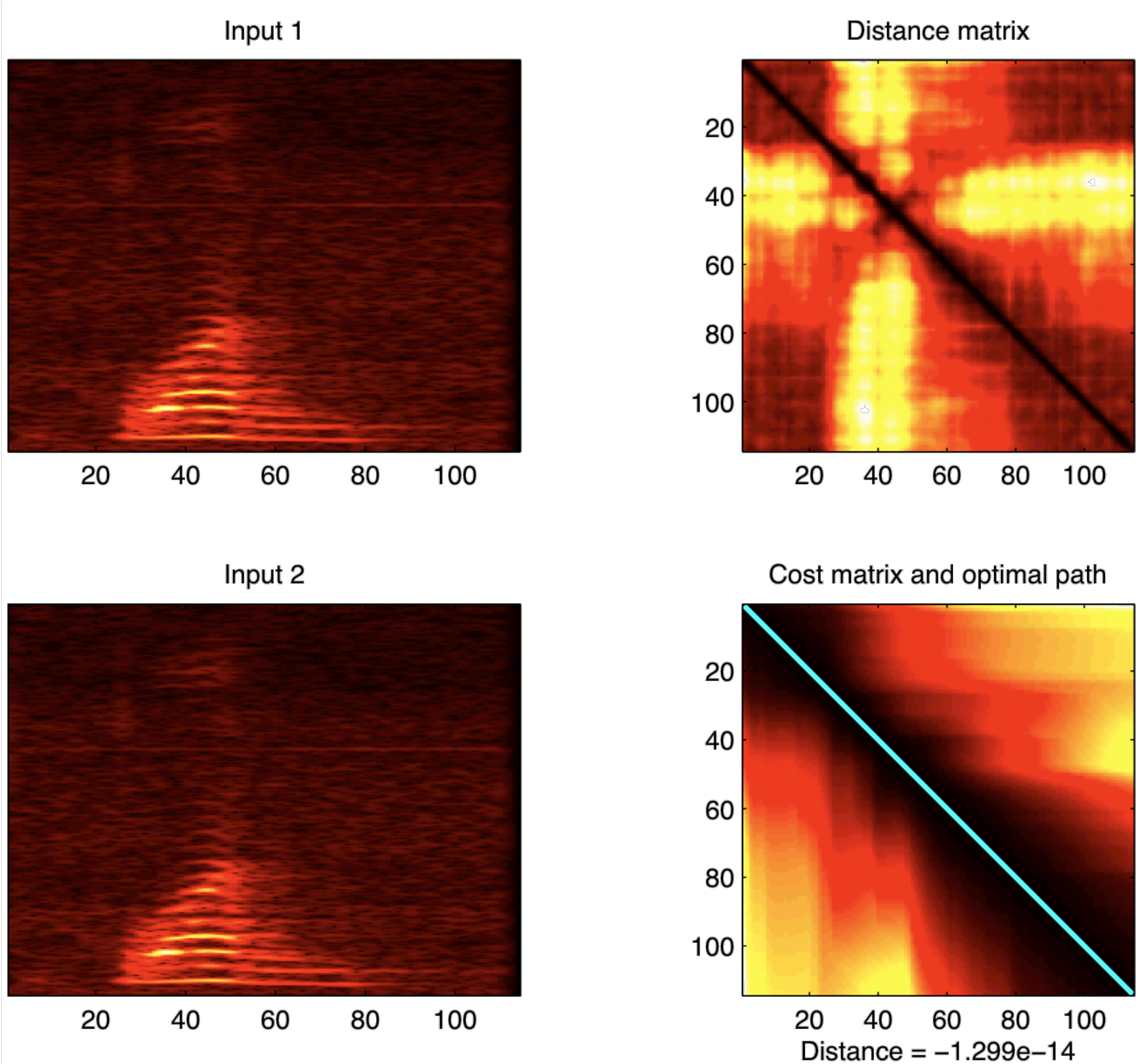
Toy data run

Local Constraint

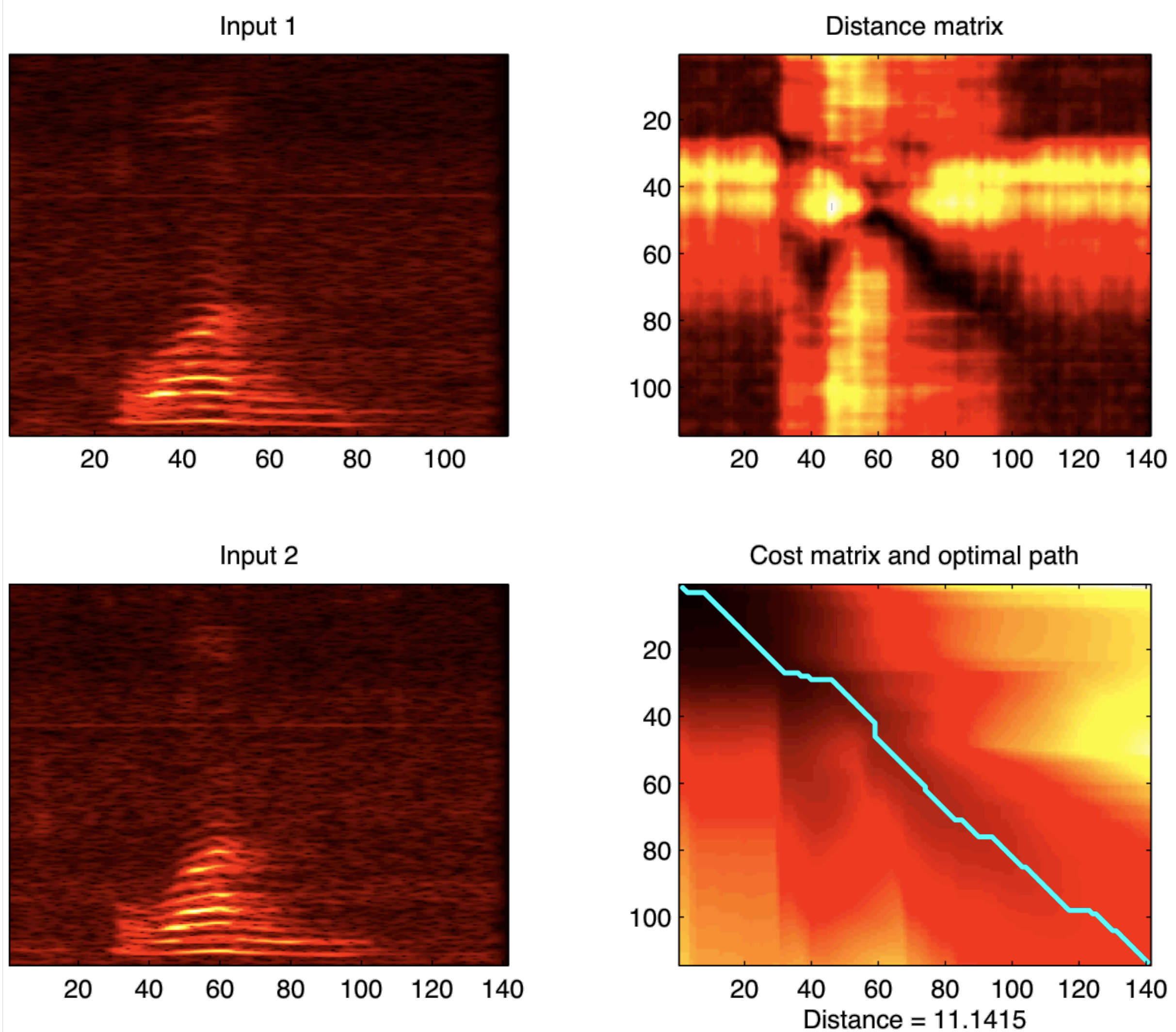




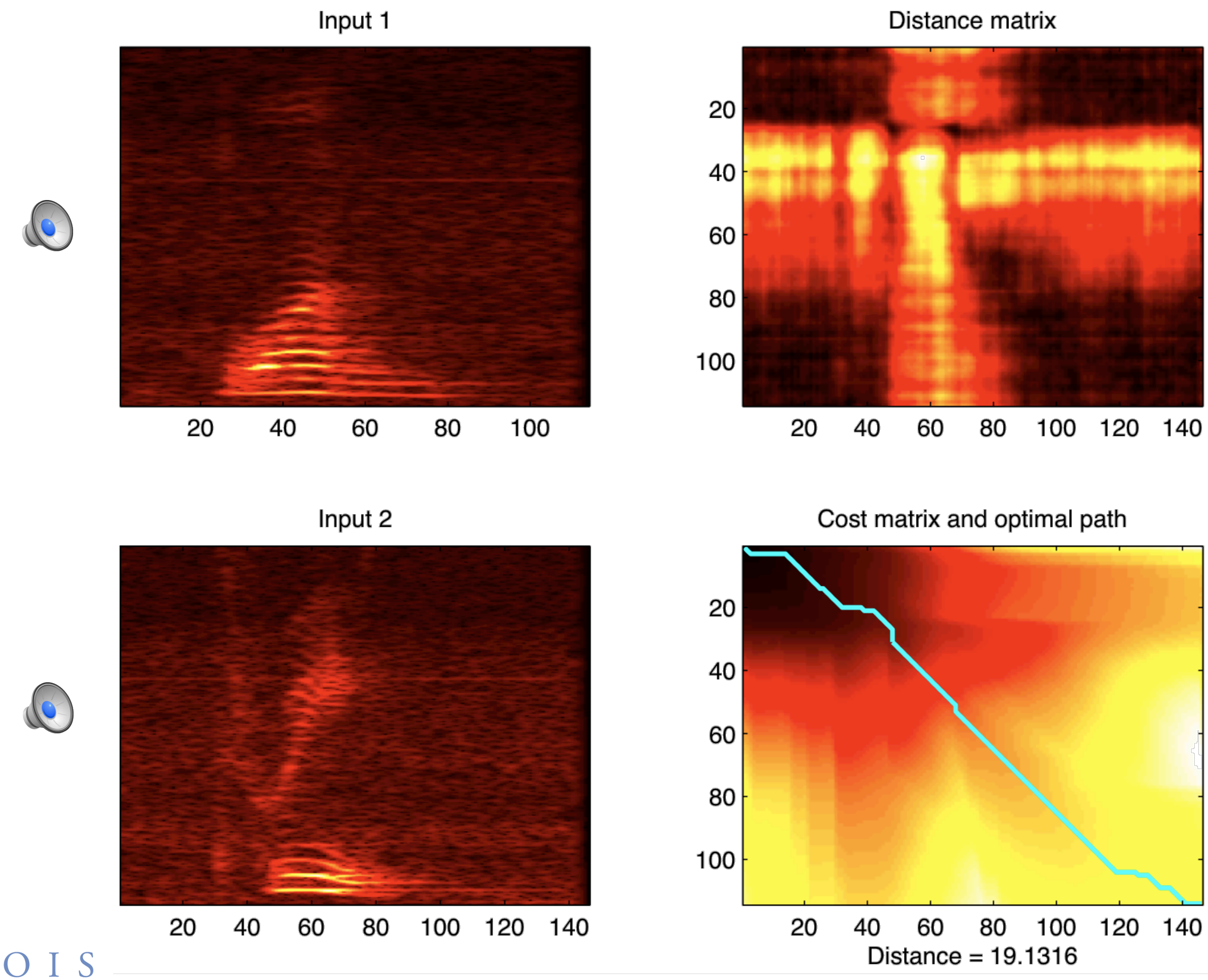
Speech example with same input



Same with similar utterance



Ditto, different input

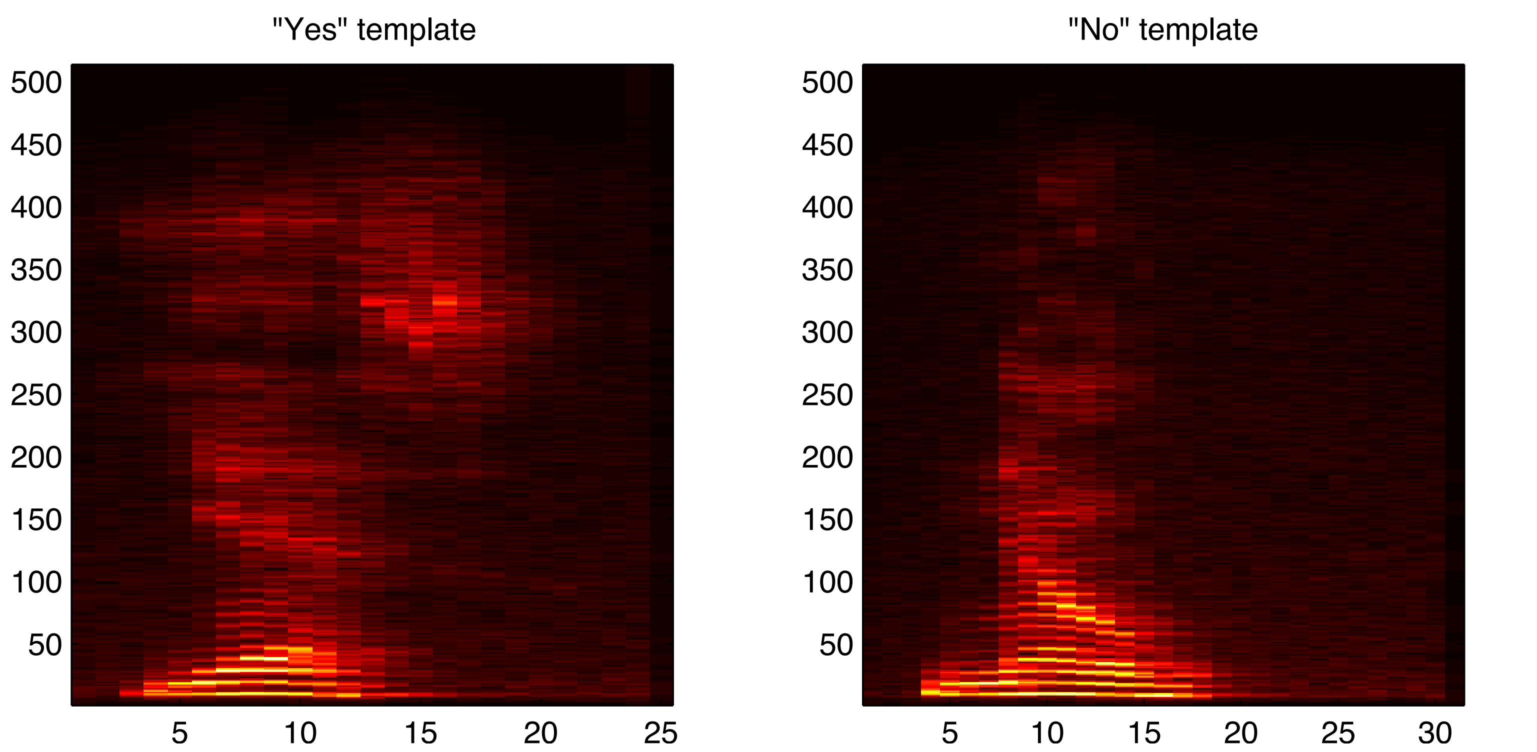


A simple yes/no recognizer

- Training phase
  - Collect data to use as prototypes
- Design phase
  - Figure out the best settings for features/DTW
- Evaluation phase
  - Test on data

Training phase

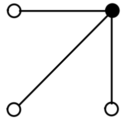
- Collect template data



ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Design Phase

- Select features/distance
  - Use spectrograms and Euclidean distance
- Global constraints
  - Don’t bother with ridiculous ratios
- Local constraints
  - Use only 0/+1 steps



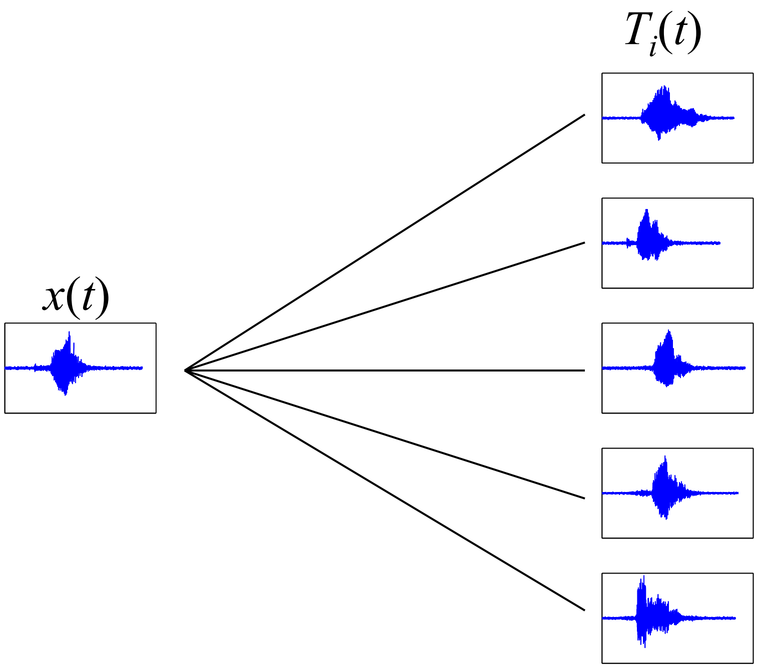
Test Phase

- Try with different utterances
  - Normal speech
  - Slow speech
  - Fast speech

- Classify according to distances between the input and the templates

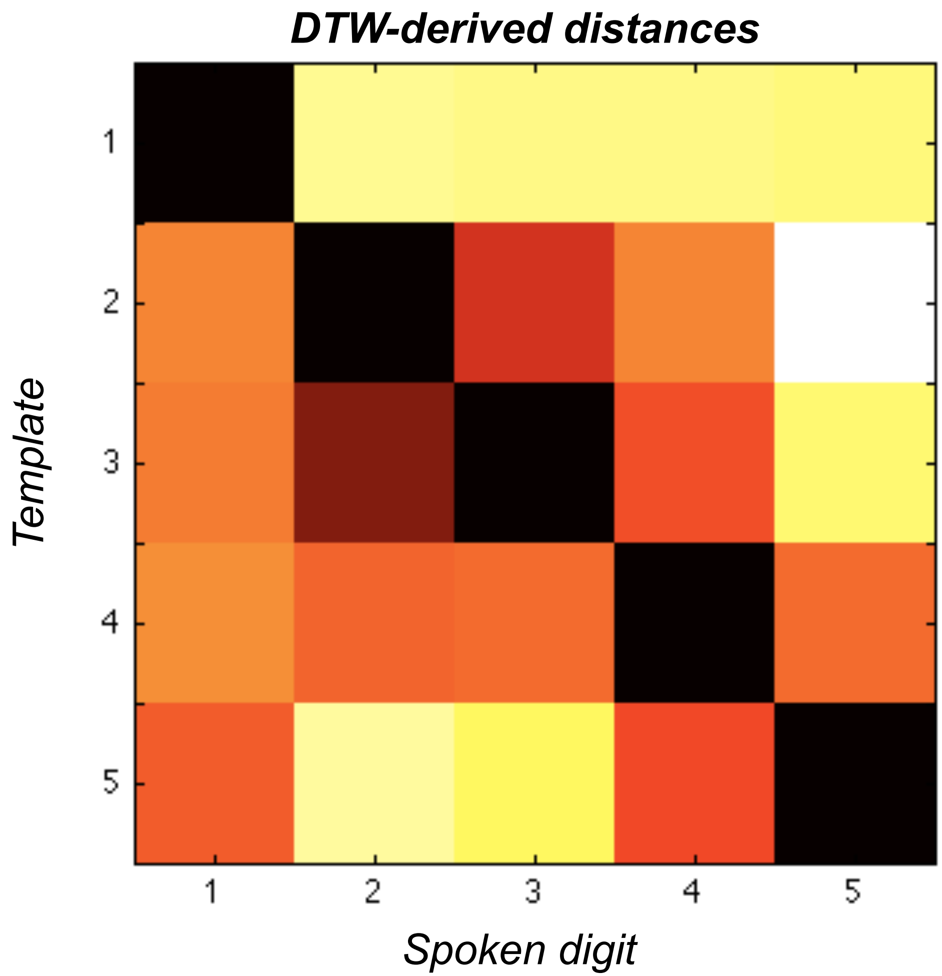
A basic speech recognizer

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



ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Recognizing digits



And that's all there is

- This is the basis if simple speech systems
  - Yes/no prompts, simple digit recognizers (e.g. in banks), phone calls by name
- Simple example-based idea
  - No need to learn about language/phonetics
  - But not very powerful in the end

Clustering Time Series

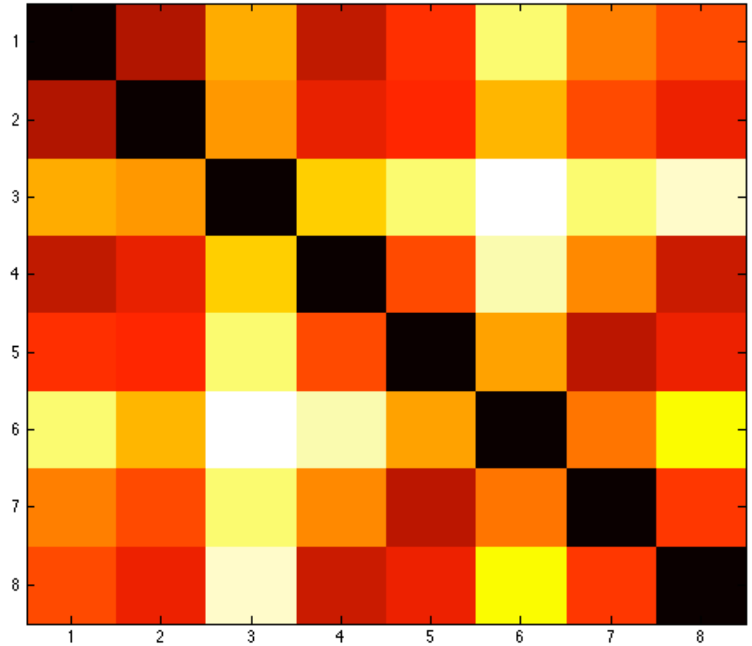
- How do we cluster time series?
  - We can't just use k-means …

- We can use DTW for this

Getting time series distances

- Compare all pairs of samples using DTW and obtain a distance $d(i,j)$ between them

Distances between all yes/no samples



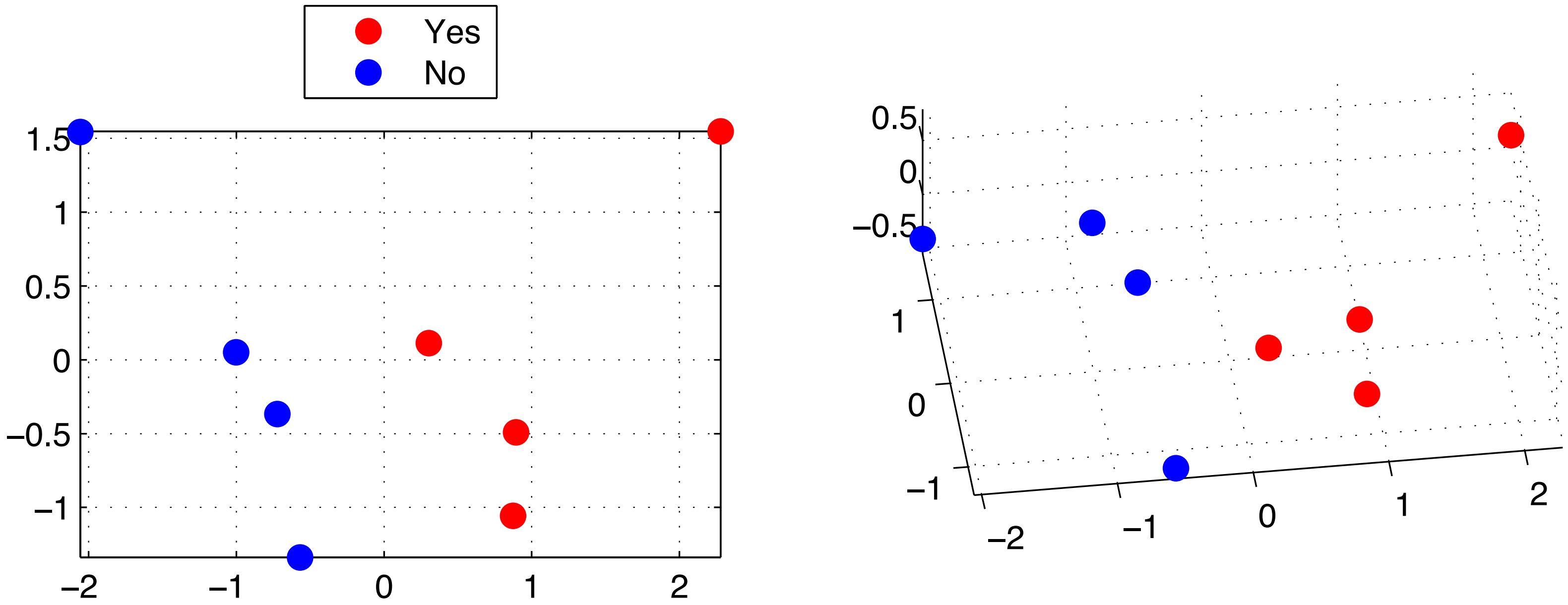
Converting distances to points

- Find points x to solve the problem:

$$\min_{x_1,\ldots,x_N} \sum \left[\left\| x_i - x_j \right\| - d(i,j) \right]^2$$

  - This is called Multidimensional Scaling (MDS)
- Resulting points simulate the data that has the prescribed distances
  - So we can use these instead

Resulting points



One more application of DTW

- Synchronization of time series
- Remember that DTW gives us temporal correspondence as well

Where that's useful

What we can do

- Use noisy audio from original take as template
- Compare to actor’s overdub take
- Find how to warp the second take to make it synchronized with original take

Example case

- Noisy audio, good video take

Using straight overdubbing

- Second take, clean audio
- Joining the two isn't good

DTW to the rescue

- Find optimal path in order to line up the two sequences
- Local constraints are now specific
  - Must maintain the timing of the video input

Using DTW alignment

Recap

- Learning with time series
- Dynamic Time Warping
- Some basic speech recognition
- Other applications of DTW