Smooth Sailing:
Learning from Sequences on Manifolds

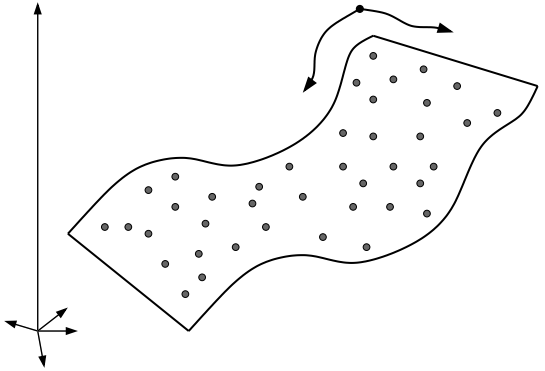
Sam Roweis

University of Toronto
Department of Computer Science

MAC Workshop **September 9, 2003**

Data on Manifolds

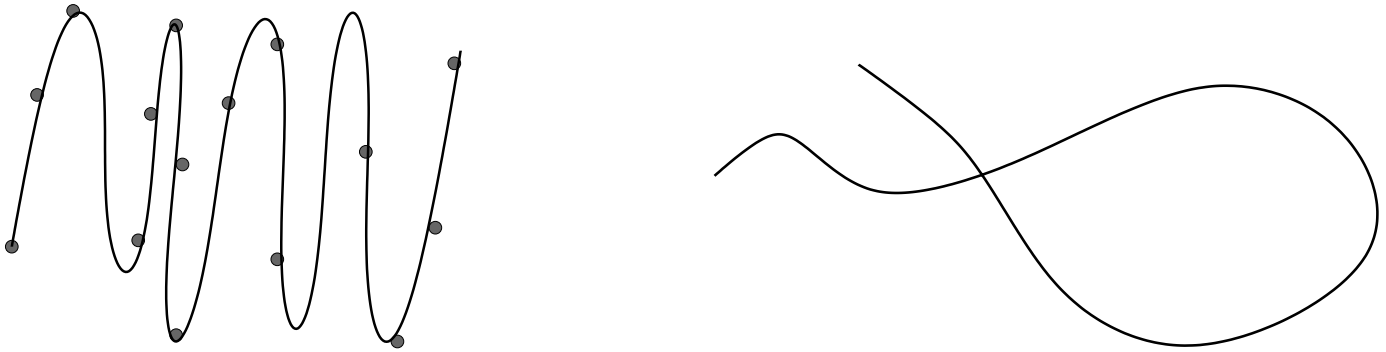
- Most data we can collect has many more correlated features than true degrees of freedom. Geometrically, this is equivalent to a **curved manifold** living in a high-dimensional space.
- Sensory percepts can be viewed as points in a high-D space. Coherent structure in the world generates **strong correlations** between inputs. Observations lie on or near **low dimensional manifolds**.



- **Nonlinearity** is key: we want to model the curved geometry of high-dimensional manifolds, not just linear subspaces.
- Applications: visualization, compression, classification interpolation, generation, denoising of complex data, efficient learning in huge feature spaces, finding compact representations of state,...

Learning Non-Smooth, Self-Intersecting Manifolds

- Many interesting algorithms have been proposed in recent and older work on nonlinear dimensionality reduction
- Almost all need to assume the manifold is **smooth** at some scale, or else we could explain *any* observations we wanted with a sufficiently convoluted manifold; but many data manifolds that we encounter in practice aren't smooth at all



- They also can't deal easily with **many-to-one** manifolds that **intersect themselves**, because nearby points in the high-D space might not be nearby in the low-D space.
- Are these cases hopeless for unsupervised manifold learning?

But...

But...

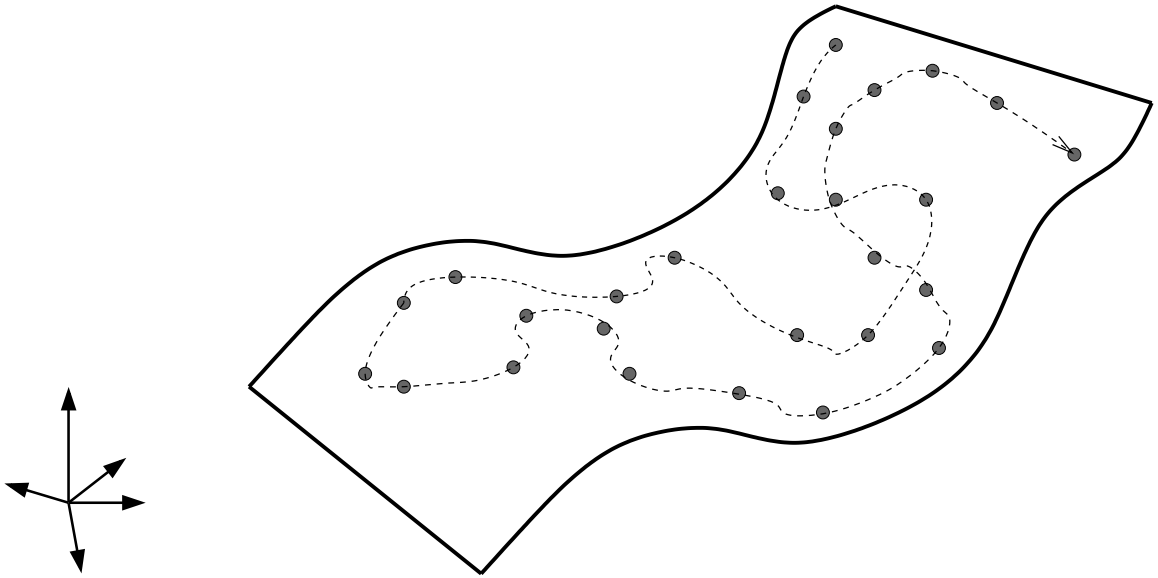
There are just two rules for success:

1. Never tell all you know.

- Roger H. Lincoln

Temporal Smoothness: Sequences on Manifolds

- There's another kind of smoothness: data might be generated using a **smooth trajectory** in the underlying manifold coordinates, even though the embedding function into the high dimensional observation space might not be smooth.
- "Things that are nearby in time are nearby on the manifold."



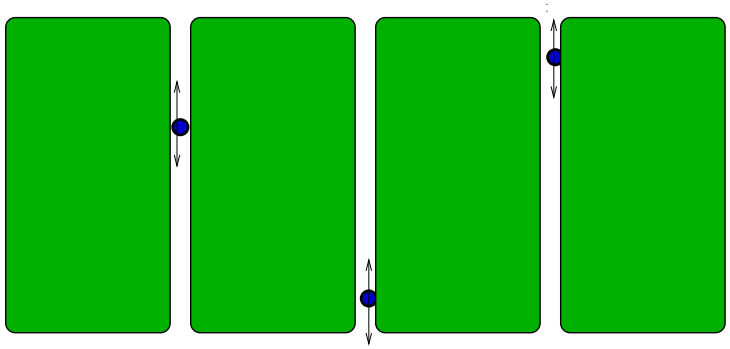
- Now we see a **vector time series** as an input, not just a random sample of points from the manifold.

A twist...

When you are stuck, try solving a different problem.

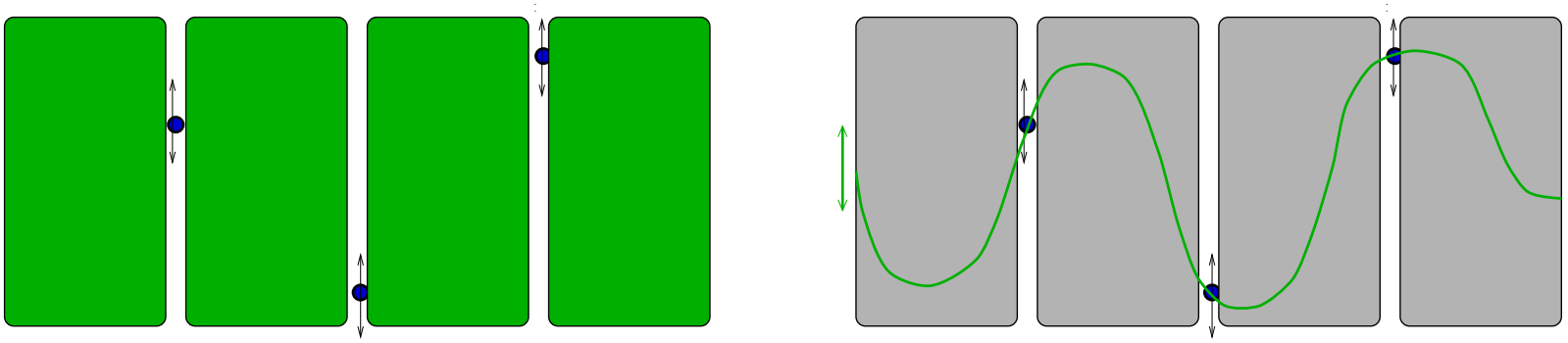
Structured Sequence Learning

- Many time series actually come from dynamical systems with a **few degrees of freedom** which produce complicated sequences of high-dimensional data.



Structured Sequence Learning

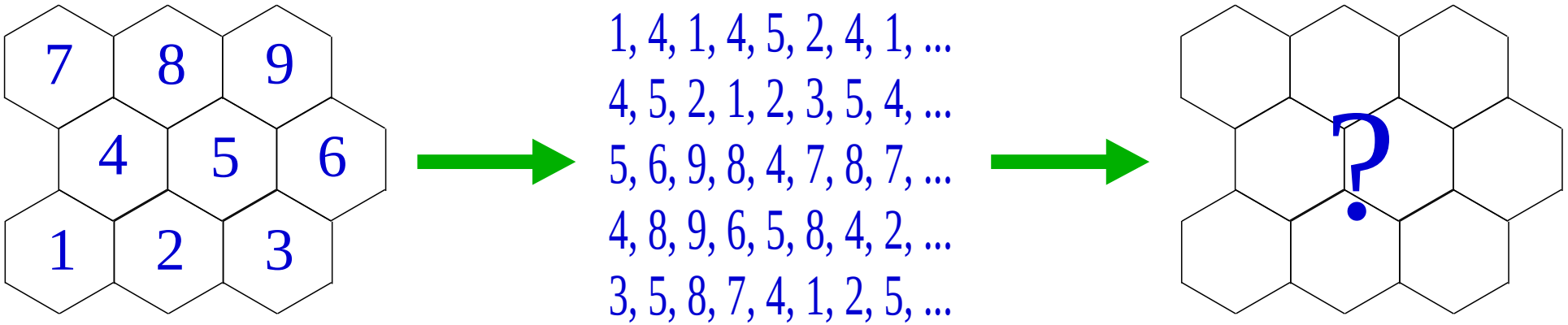
- Many time series actually come from dynamical systems with a **few degrees of freedom** which produce complicated sequences of high-dimensional data.



- Latent variable models with low-dimensional hidden states that evolve through time are ideal for learning such time series.
- Once learned, they can be used to **classify**, do **outlier detection**, **fill-in** (interpolate) or **predict** (extrapolate) measurements and **infer hidden states** given noisy observations.

A Game

- **Input data**: one or more sequence(s) of numbers generated by connected movement in an underlying map.

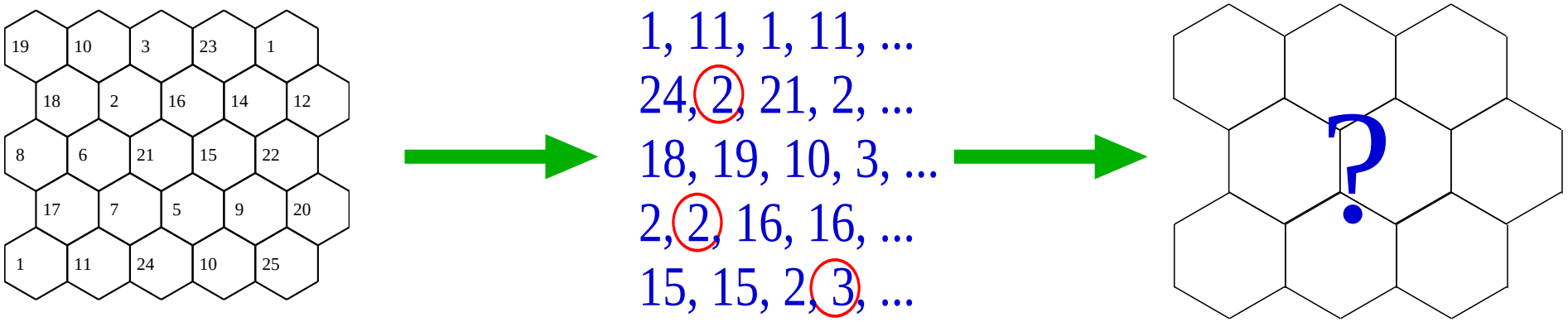


- **Learning task**: reconstruct the topology of underlying map (clearly absolute scale and rotation cannot be recovered).
- Many non-probabilistic methods can solve this problem (e.g. adjacency graphs).

(This simple example has discrete-valued data, but we can play the same game with continuous vector observations.)

A Harder Game

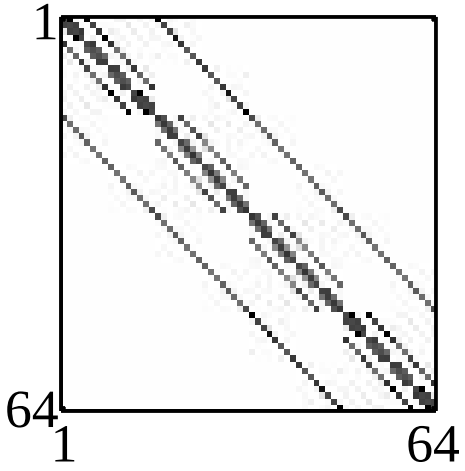
- Now make the output process **noisy** and allow repeated numbers in the underlying map (output process **many to one**).

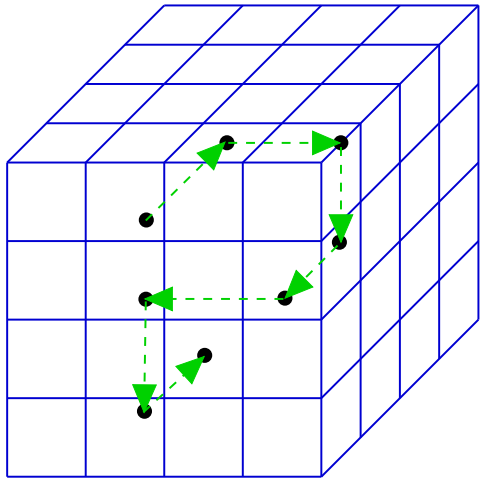


- If the sequence is "exciting enough" **learning is still possible**, but deterministic methods will not work.
- Sounds like a job for statistical learning, but which model should we use?

Idea: Constrained Hidden Markov Models

- We can create spatial dynamics in HMMs by thinking of each state as corresponding to a cell in a fictitious **topology space**.
- Now by **constraining the transition probabilities** we can ensure that all valid state sequences correspond to continuous (connected) paths in the topology space.

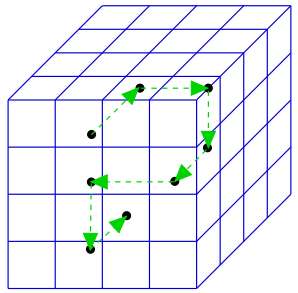




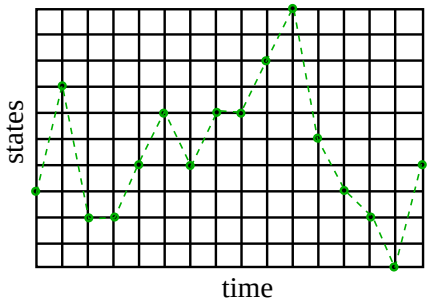
- We must choose
  - the **dimensionality** of the topology space
  - a **packing** (e.g. cubic,hexagonal)
  - a **neighbourhood rule** (e.g. share face/edge/corner)

Inference & Learning in CHMMs

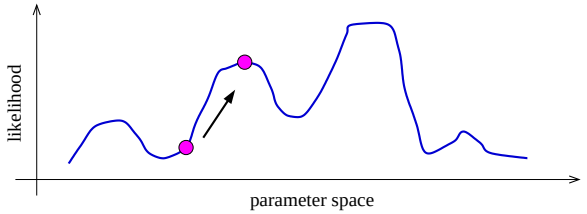
Initialization: chose
(1) dimensionality for the topology space,
(2) packing and (3) neighbourhood rule, then create CHMM by precomputing the transition matrix.



Inference: use forward-backward algorithm or Viterbi decoding (very sparse).



Learning:
Use Baum-Welch algorithm (EM)
to update the emission parameters,
holding transition matrix fixed.

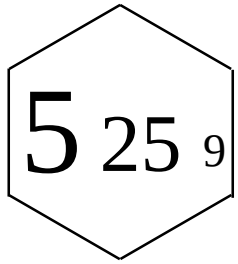


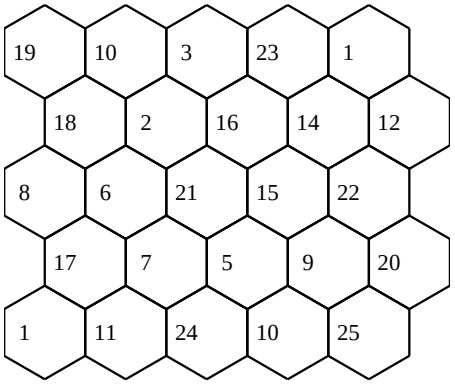
Local minima can be a problem (must use tricks).
Model structure (e.g. number of states), if not limited by computation, must be chosen using e.g. cross-validation.

CHMMs can win the game

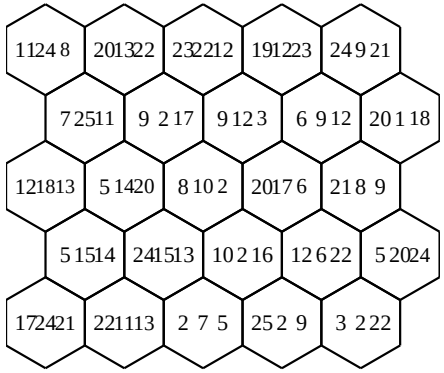
Training data:
600 symbols
15% noise

(In the pictures below,
$fontsize \propto \sqrt{probability}$
so that $ink \propto probability$**.)**

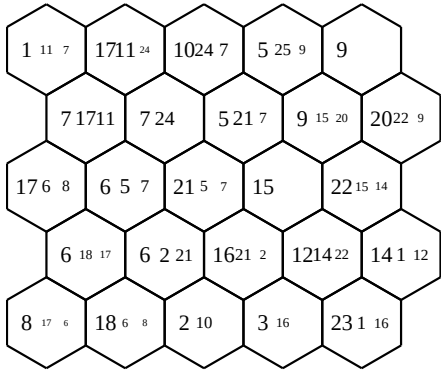




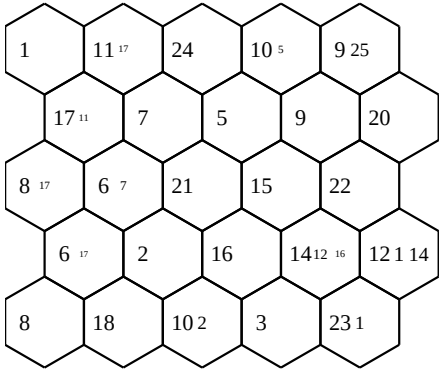
True Generative Map



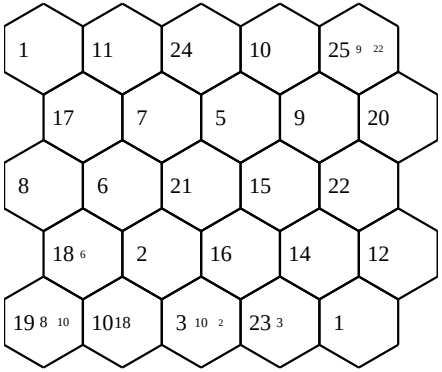
Random initialization logLikelihood:−3.2321



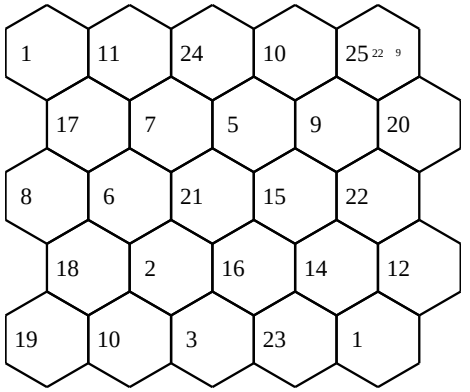
Iteration:005 logLikelihood:−2.4270



Iteration:010 logLikelihood:−2.1451



Iteration:025 logLikelihood:−1.9980

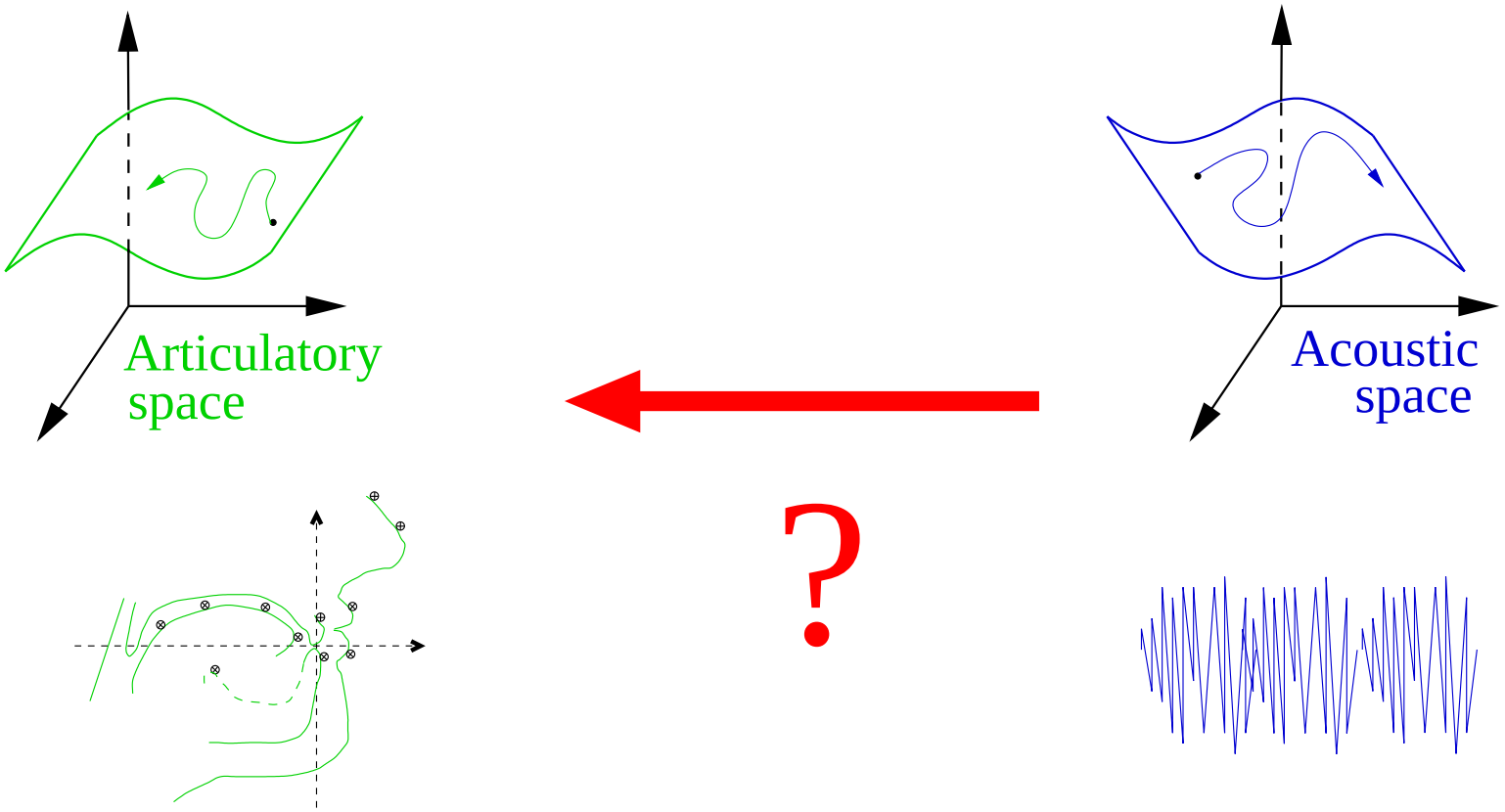


Iteration:030 logLikelihood:−1.9624

2D topology space with hex packing (cubic also works)

Example application: inferring mouth movements

Can you hear the shape of the mouth? (after Kac)

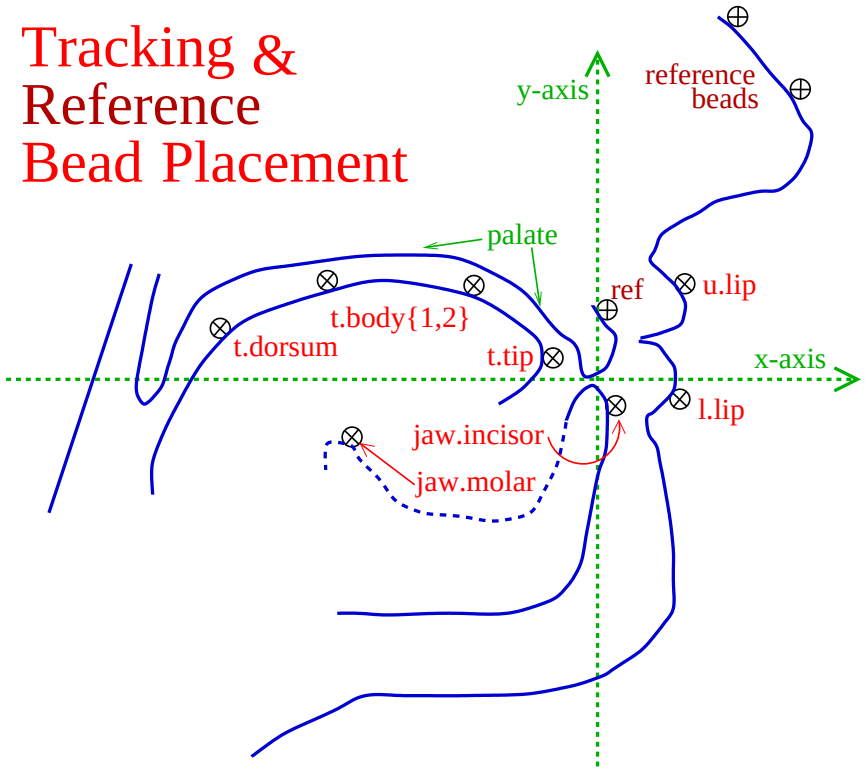


Old and interesting problem in speech science and engineering.

Manifold is **self-intersecting**, so naive regression won't work.
(Many articulatory positions can produce the same acoustics.)

An experiment with real data

X-ray microbeam data from U.Wisconsin, Madison.



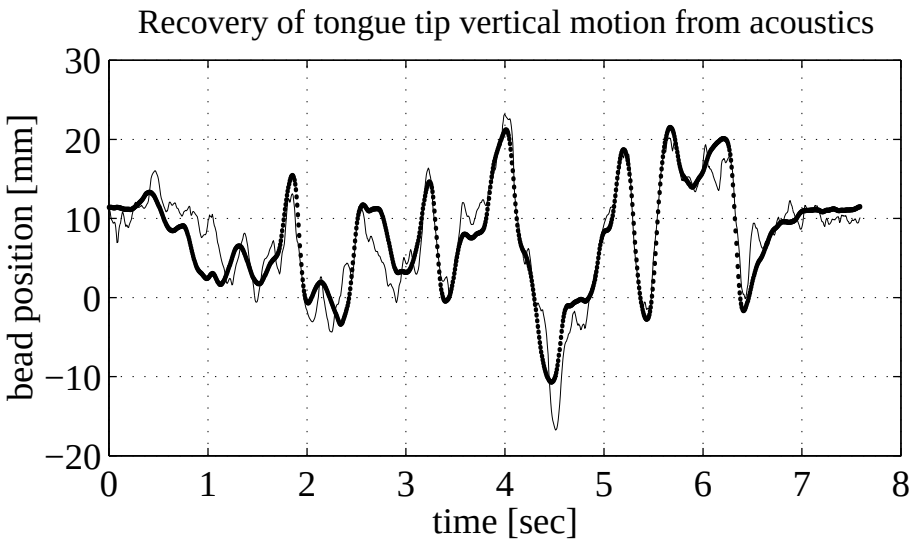
- **simultaneous audio and movements**
- **audio**: 21kHz, 16bit
- **movement**: 8 beads
  accuracy: 146Hz, $\pm$1mm
- midsaggital only
- lots of material (numbers, words, sentences, etc. read by 32 women, 25 men)

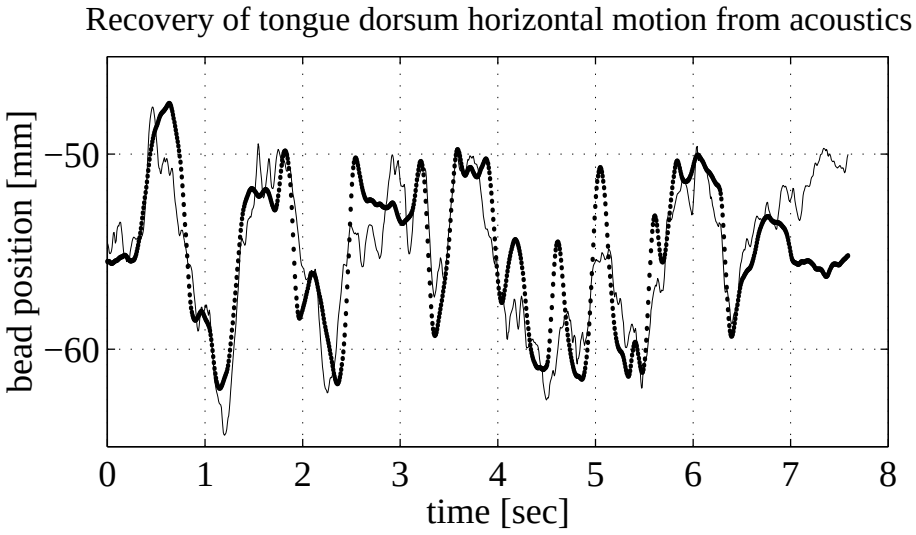
- **Movement parameterization**: 16 bead positions every 7ms (can be linearly projected down to 5–8 dims)
- **Audio parameterization**: 12 **line spectral pair (LSP)** coefficients plus log energy every 7ms (25 ms windows)

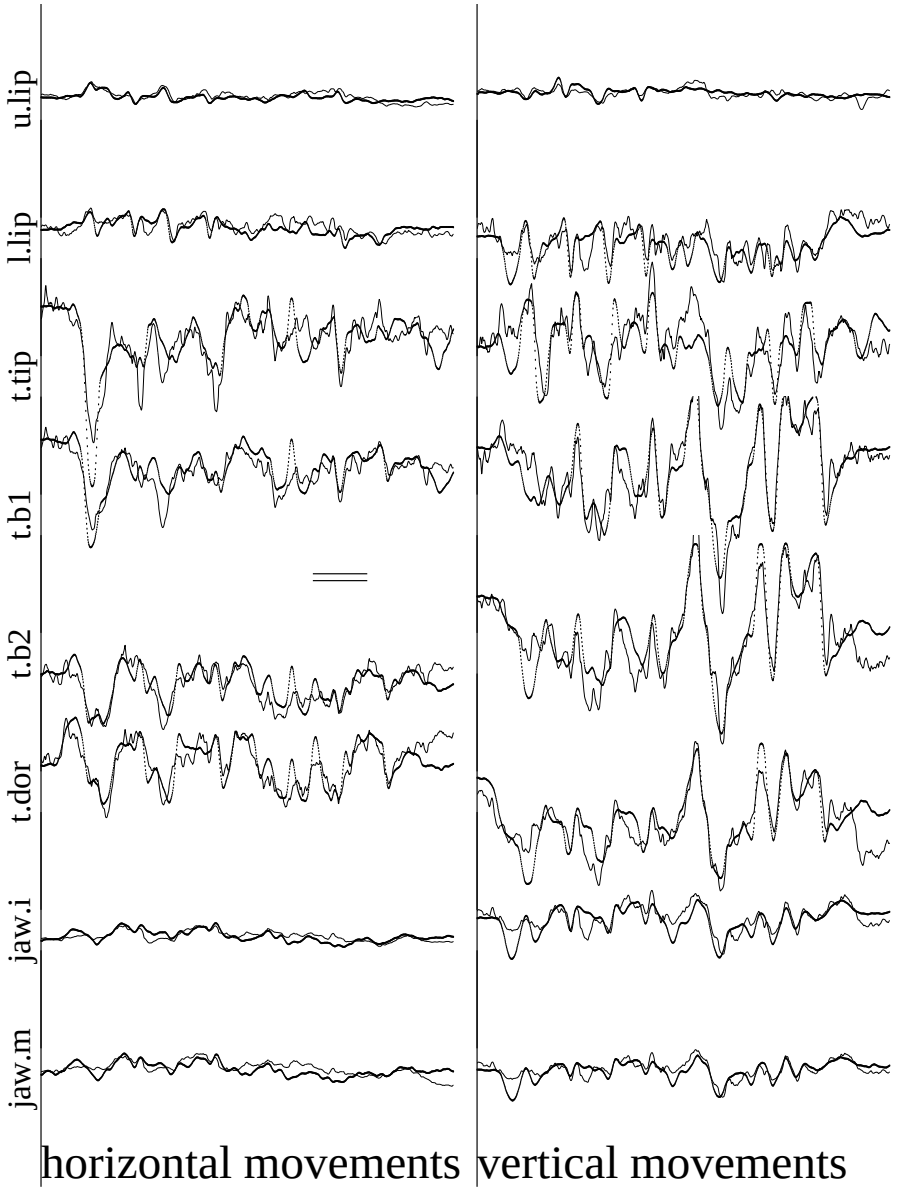
Articulatory Recovery with CHMMs

- Used a four-dimensional CHMM with several thousand states (cubic packing, face neighbours).
- Each state has a Gaussian distrib. over spectral params (LSP).
- To recover a continuous state trajectory, do Viterbi decoding and then interpolate smoothly between centres.
- After the unsupervised learning, find a single linear mapping between the 4D state trajectories and articulator movements.
- Vicious local minima. Need to both
  1) anneal very gently and carefully and
  2) use an initialization trick
- A good initialization trick is to train a similar 4D CHMM on sequences of **movement** data and then convert this it to an acoustic CHMM by taking conditional densities.

Recovered mouth movements using CHMMs

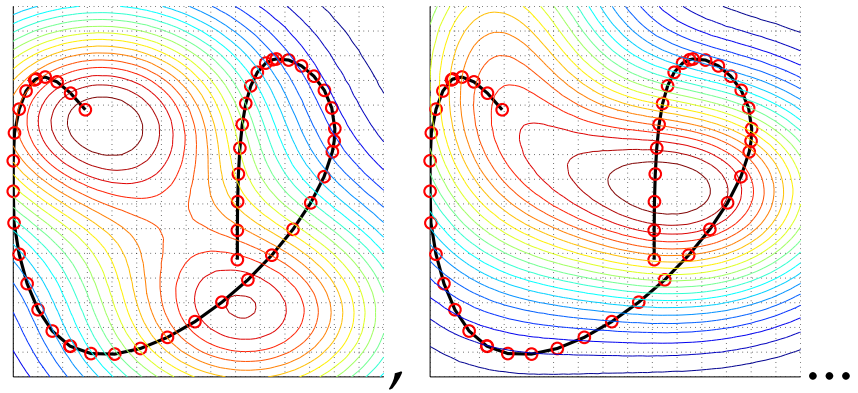


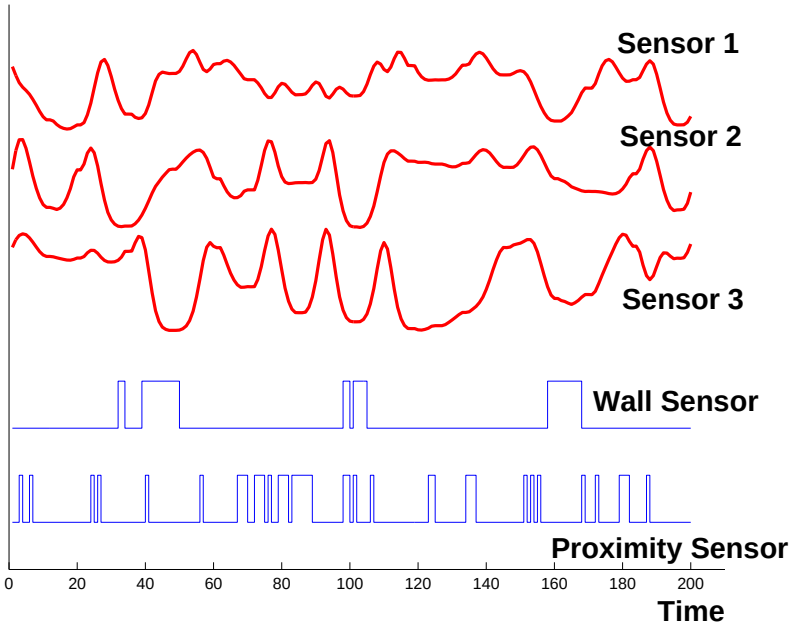




Simultaneous Localization and Surveying (work with Ruslan Salakhutdinov)

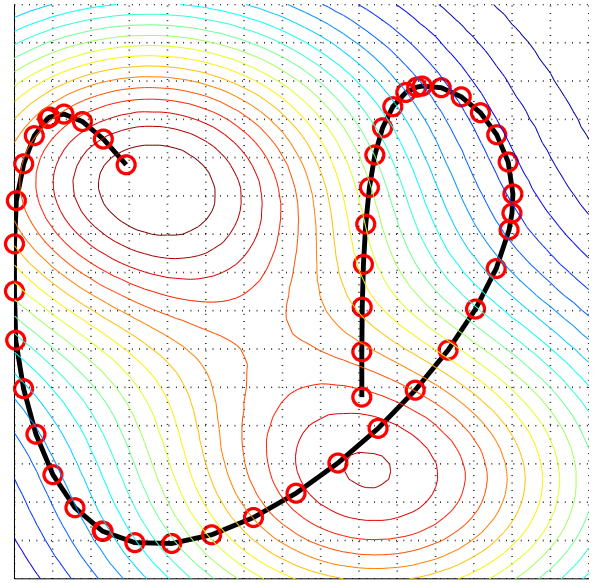
Goal: learn the responses of (survey) some variables across an unknown space, given only time series logs of noisy sensors.

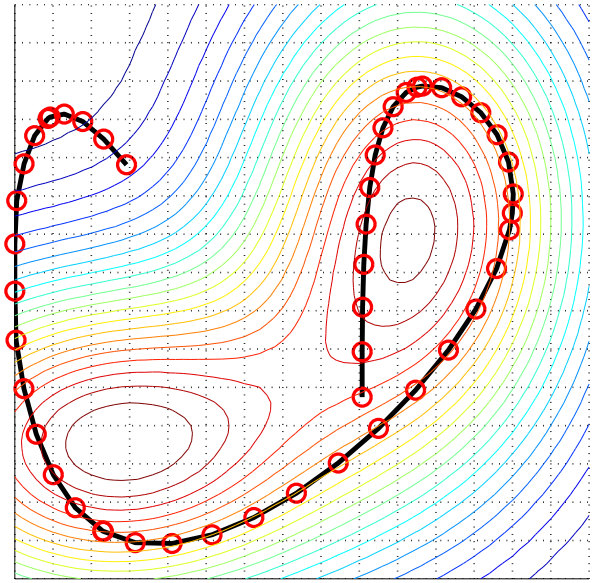




Occupancy grid of world is typically empty (except at boundary).
No odometry available (except binary boundary signal).

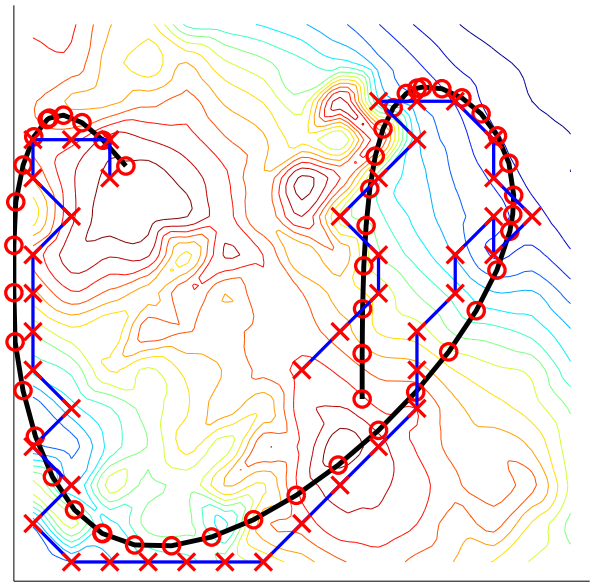
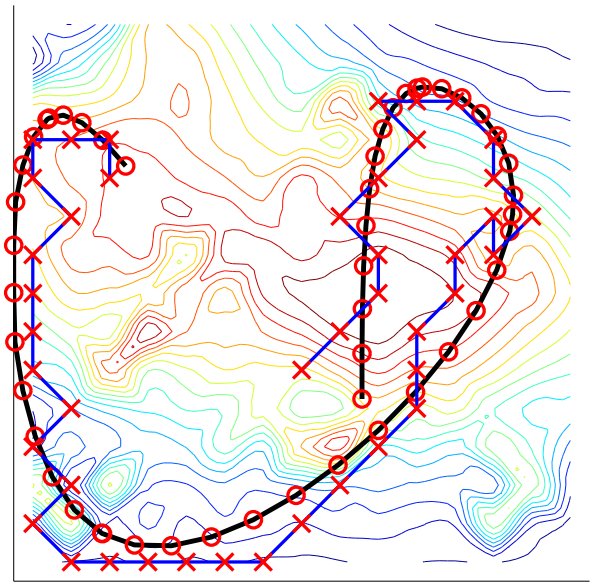
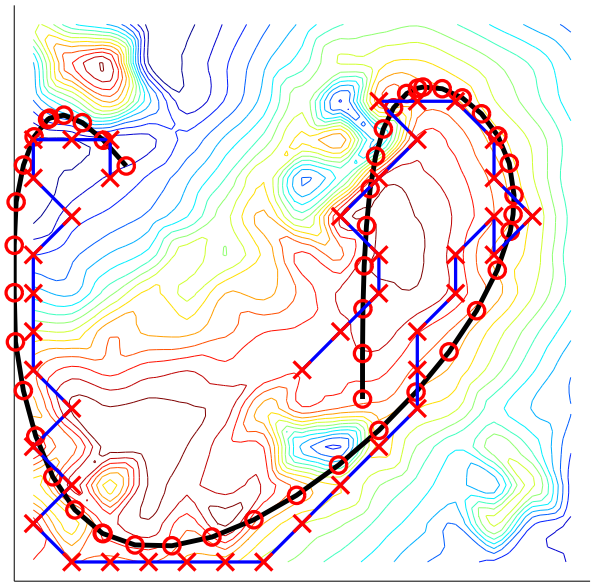
CHMM results for a simple 15x15 grid world





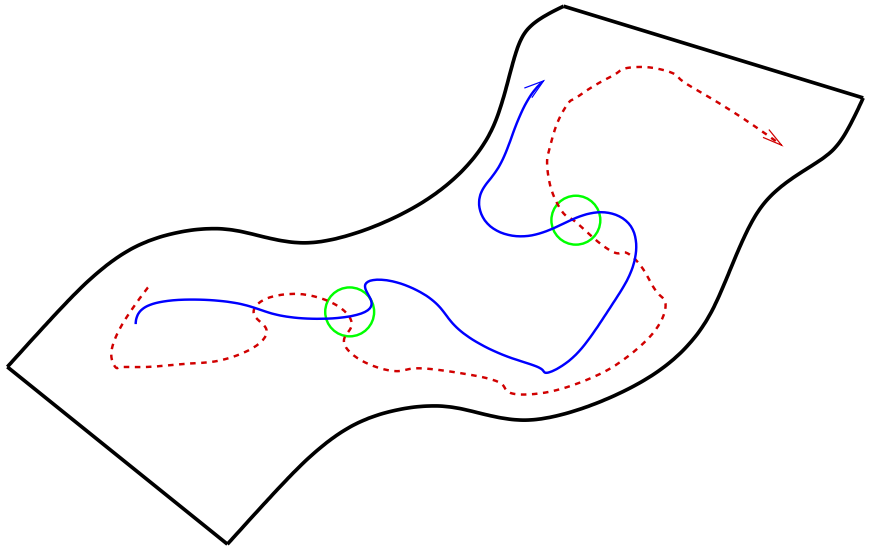
True Sensor Responses

Learned Survey

Multiple Agents

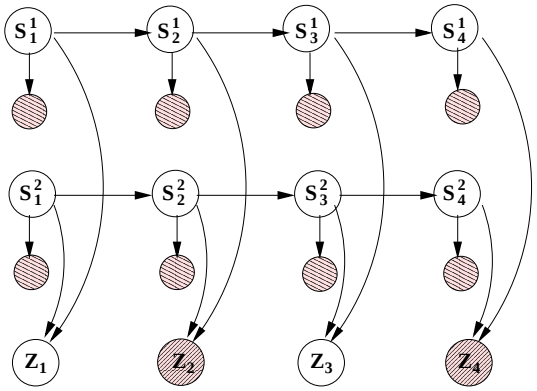
- A single robot takes a long time to cover a certain area, so surveying large maps requires huge amounts of data, and the outputs are unreliable in places never visited.
- Multiple robots can explore the space faster, but if they do not interact the resulting learning problem is the same as if a single robot went on many excursions.
- The interesting case is when **agents interact**, even rarely.
- Now we have **multiple simultaneous sequences** from the manifold but they are linked in some way by extra signals.

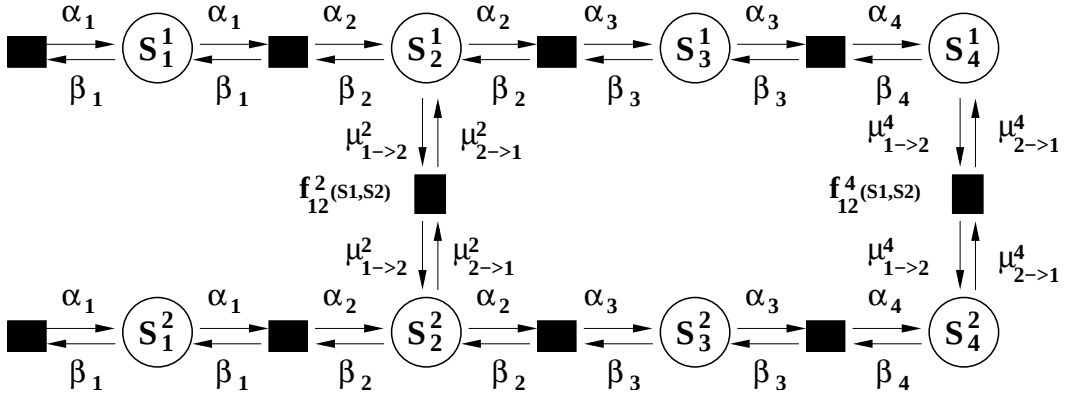


- Example: proximity detectors between multiple mobile agents.

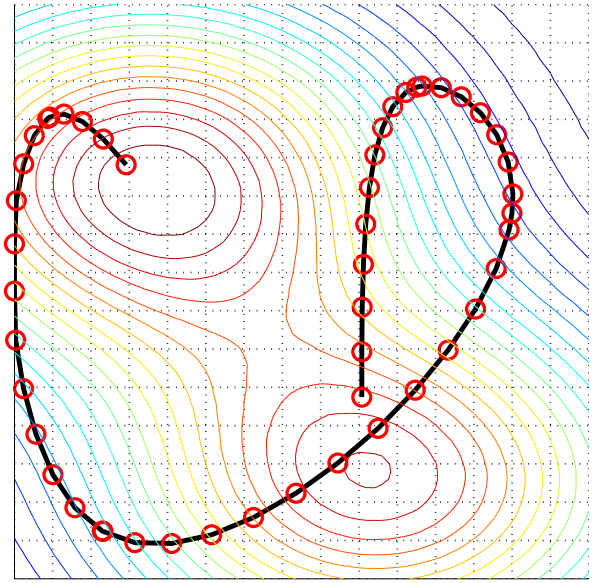
Approximate Inference with Loopy BP

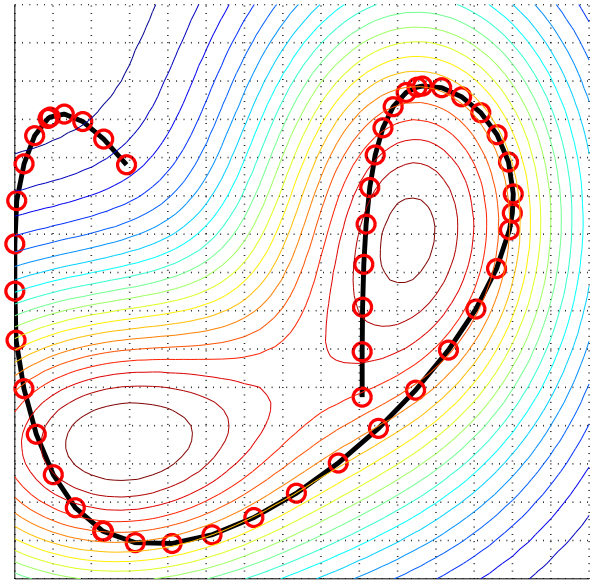
- Once we have multiple interacting sequences, we can no longer apply the standard HMM inference and learning rules.
- The graphical model (like a factorial HMM) **couples** our beliefs about the robot positions given the observations.
- We designed an efficient **loopy BP algorithm** for approximate inference, which works well when interactions are rare.
- Intuition: run FB on one robot, taking into account where other robots think you should be. Send your resulting "gammas" as messages telling other robots where you think they should be.





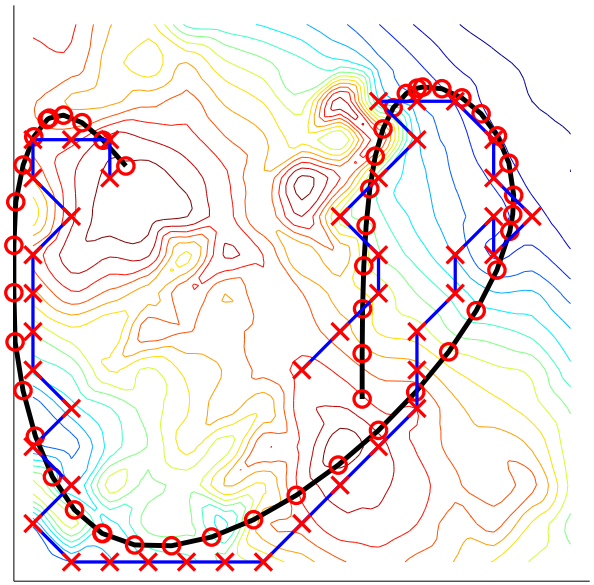
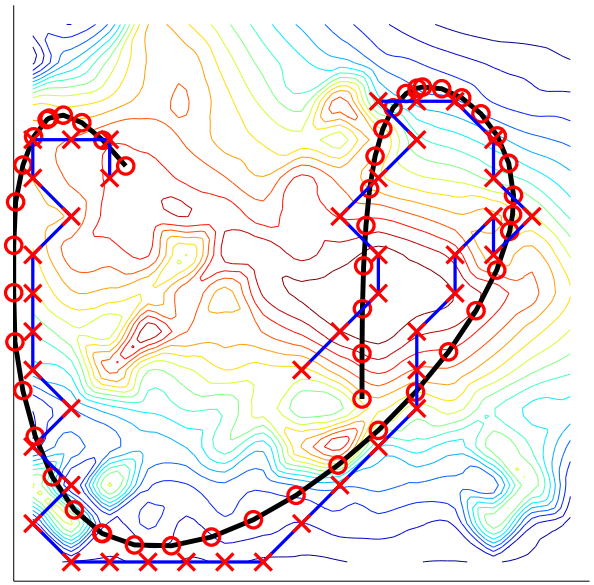
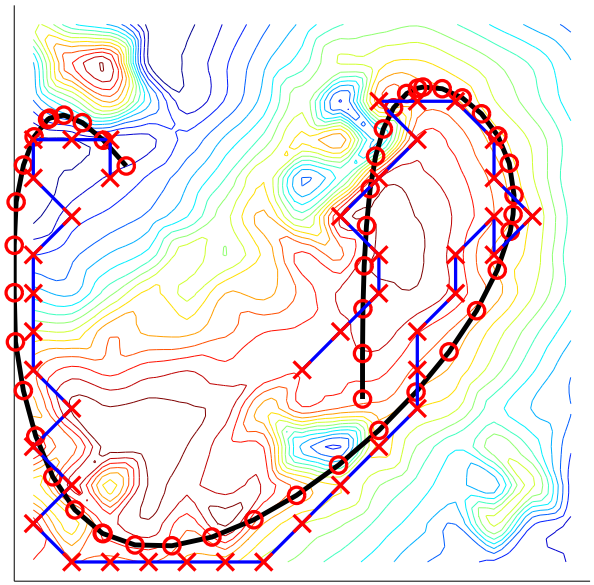
Results - 2 Robots, 1/2 data each



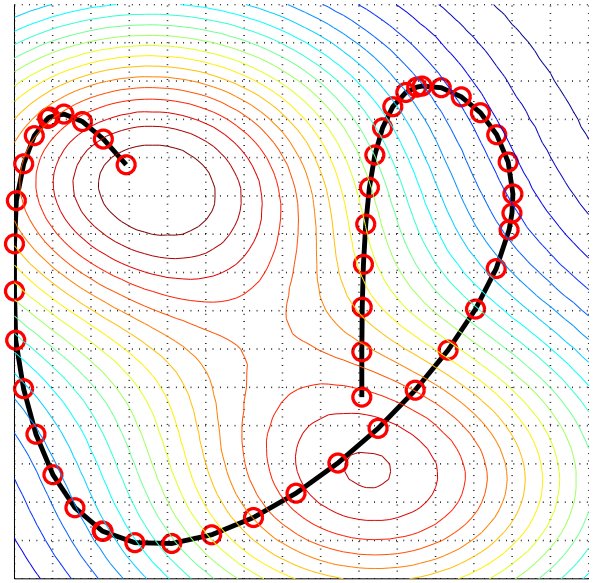


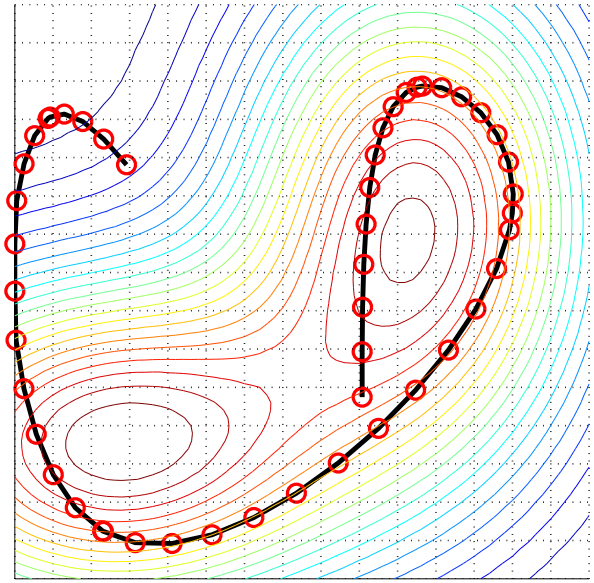
True Sensor Responses

Learned Survey

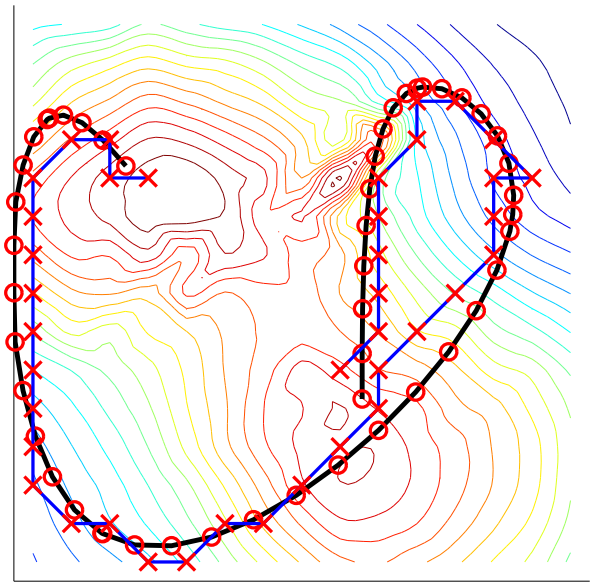
Results - 4 Robots, 1/4 data each

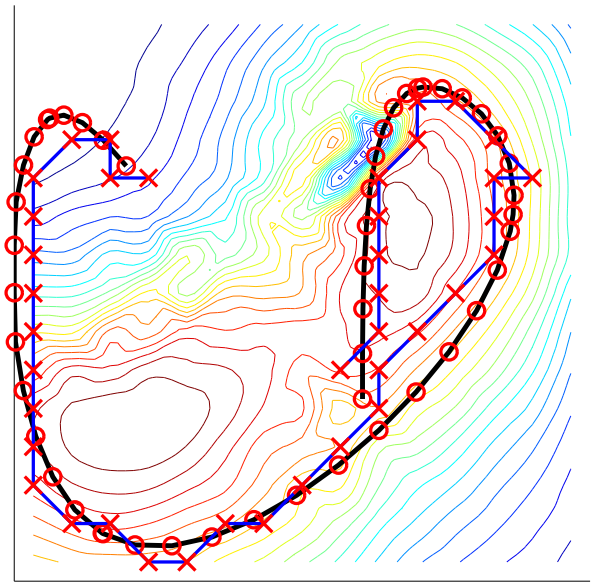




True Sensor Responses

Learned Survey





Local Inference Algorithm For Robot r

- Define & initialize α, β, γ as usual: $\alpha_t^r(i) = p(\boldsymbol{y}_{1:t}^r, s_t^r = i|\Theta)$; $\beta_t^r(i) = p(\boldsymbol{y}_{t+1:T}^r, s_t^r = i|\Theta)$; $\gamma_t^r(i) = p(s_t^r = i|Y, \Theta)$
  $\alpha_1^r(i) = \pi_i^r p(\boldsymbol{y}_1^r|s_t^r = i)$; $\beta_T^r(i) = 1$
- Initialize messages to unity: $\mu_{q\to r}^t = 1$
- Run forward-backward, except including incoming messages:

$$\alpha_{t+1}^r(j) = \left[\sum_i \alpha_t^r(i) T_{ij} \prod_q \mu_{q\to r}^t(i)\right] p(\boldsymbol{y}_{t+1}^r|s_t^r = j)$$

$$\beta_t^r(i) = \sum_j T_{ij} p(\boldsymbol{y}_{t+1}^r|s_t^r = j) \beta_{t+1}^r(j) \prod_q \mu_{q\to r}^{t+1}(j)$$

- Compute marginal beliefs and outgoing messages:

$$\gamma_t^r(i) = \frac{\alpha_t^r(i)\beta_t^r(i) \prod_q (\mu_{q\to r}^t)_i}{\sum_j \alpha_t^r(j)\beta_t^r(j) \prod_q (\mu_{q\to r}^t)_j}$$

$$\mu_{r\to q}^t(i) = \frac{\alpha_t^r(i)\beta_t^r(i)}{\sum_j \alpha_t^r(j)\beta_t^r(j)} \quad \forall q$$

Efficient Handling of Interaction Potentials

- In general, should account for both false positives and false negatives, but we assume fp rate is zero (though it isn't, even in our simulations) to keep loops large and sparse.
- Potential function $f_{qr}(s^q, s^r)$ is constant if proximity signal is not observed, otherwise if proximity signal is observed:

$$f^t_{qr}(s^q_t, s^r_t) = \left\{ \begin{array}{l} 1 \text{ if } s^q_t = s^r_t \\ 0 \text{ otherwise} \end{array} \right\}$$

- The message at time slice t that a factor node f^t_{qr} sends to the variable node s^r_t now takes the form

$$m^t_{q \to r} \propto \mu^t_{q \to r} \sum_{s^t_q} f^t_{qr}(s^q_t, s^r_t)$$

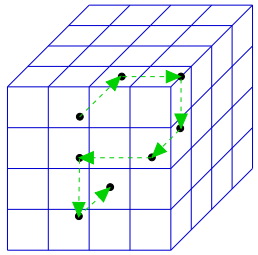
  where μ are computing ignoring the interactions.
- This trick allows efficient inference: considering proximity just another observation means pushing around a lot of zeros.

The obligatory demo...

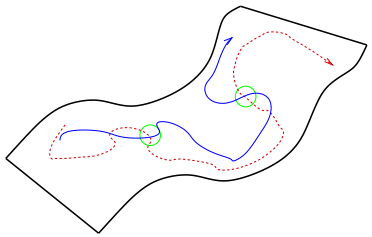
If you can, always show movies.
Otherwise, scan in the first page of a very old paper.

Conclusions

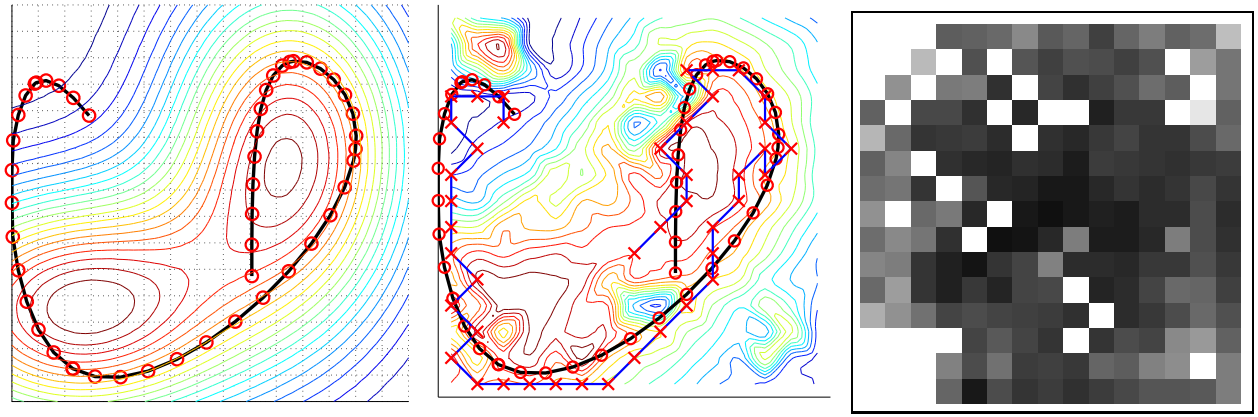
- Like clustering, nonlinear dimensionality reduction tries to find a set of small volumes in the high-dimensional observation space that have large data density.
- For highly curved or self-intersecting manifolds, nearby observations are not necessarily close in intrinsic coordinates.
- But if data are generated **sequentially**, by moving slowly and smoothly on the manifold, we can use temporal proximity as a learning signal, e.g. using **constrained HMMs**.



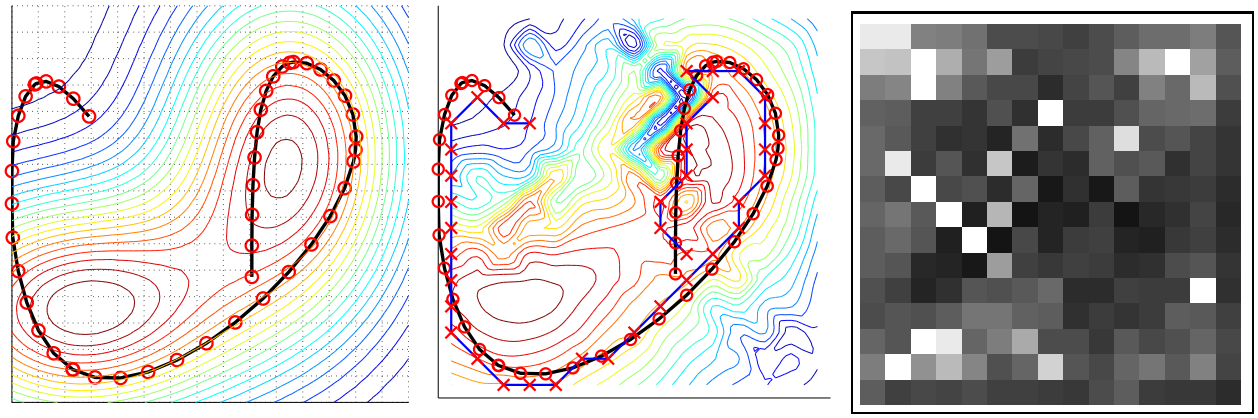
- When we have multiple simultaneous observation sequences and some interaction signals, some very interesting new unsupervised learning problems arise that use **several datasets** and exploit **partial correspondences** between them.



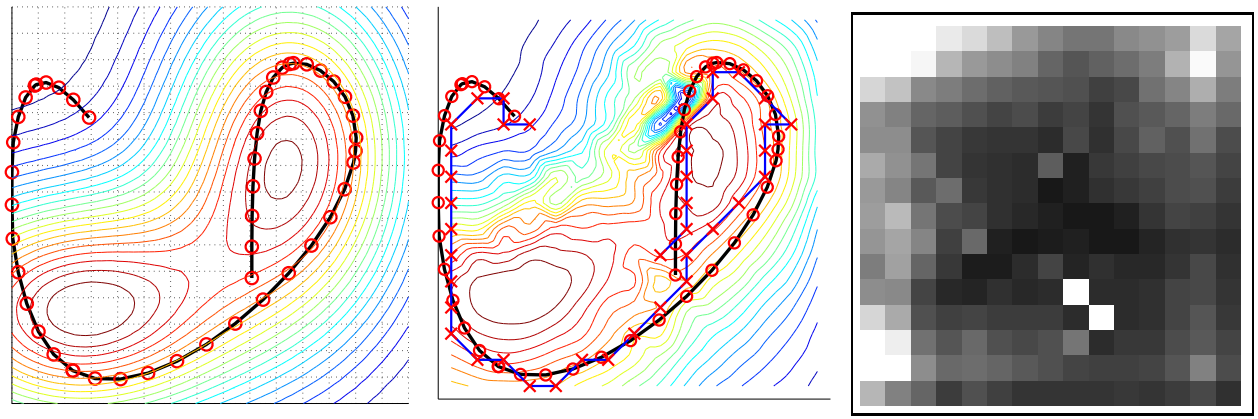
Gamma Maps for Active Learning



By computing expected occupancy of each state (cell), we can obtain confidence estimates on our survey.

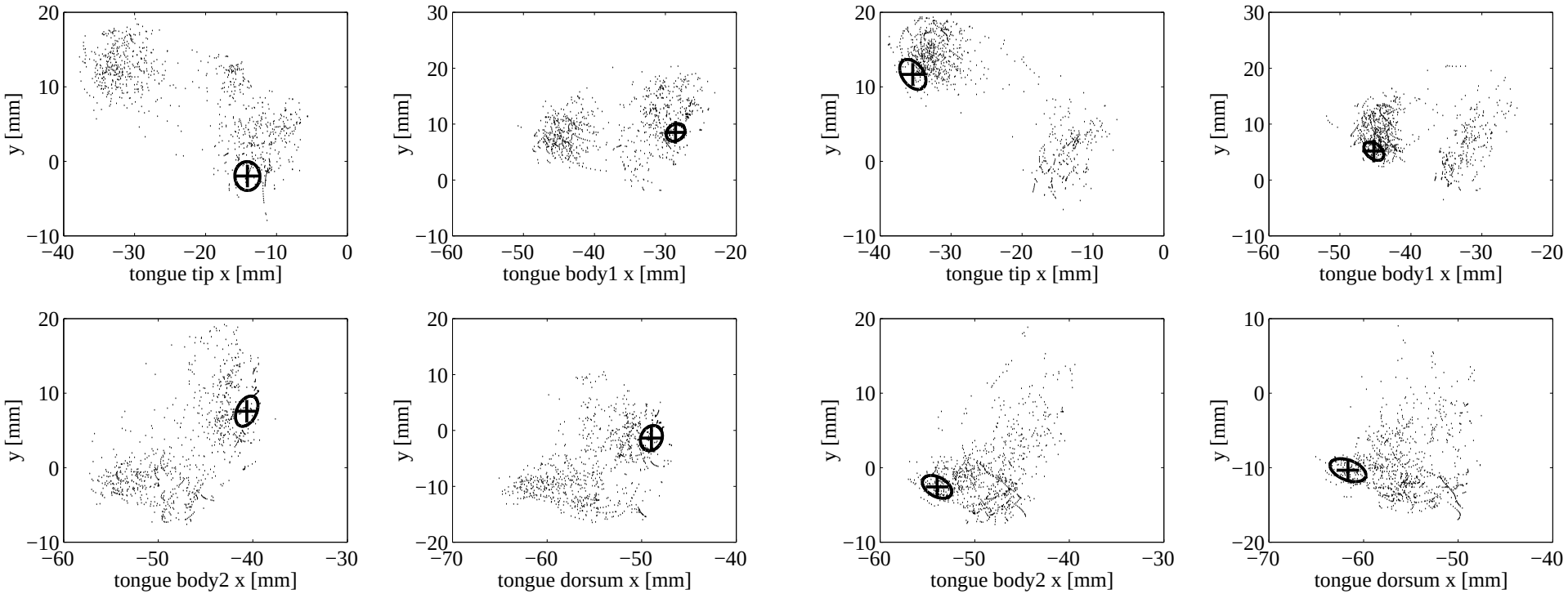


We can also send feedback to the controller to recommend active exploration strategies that will be maximally informative.



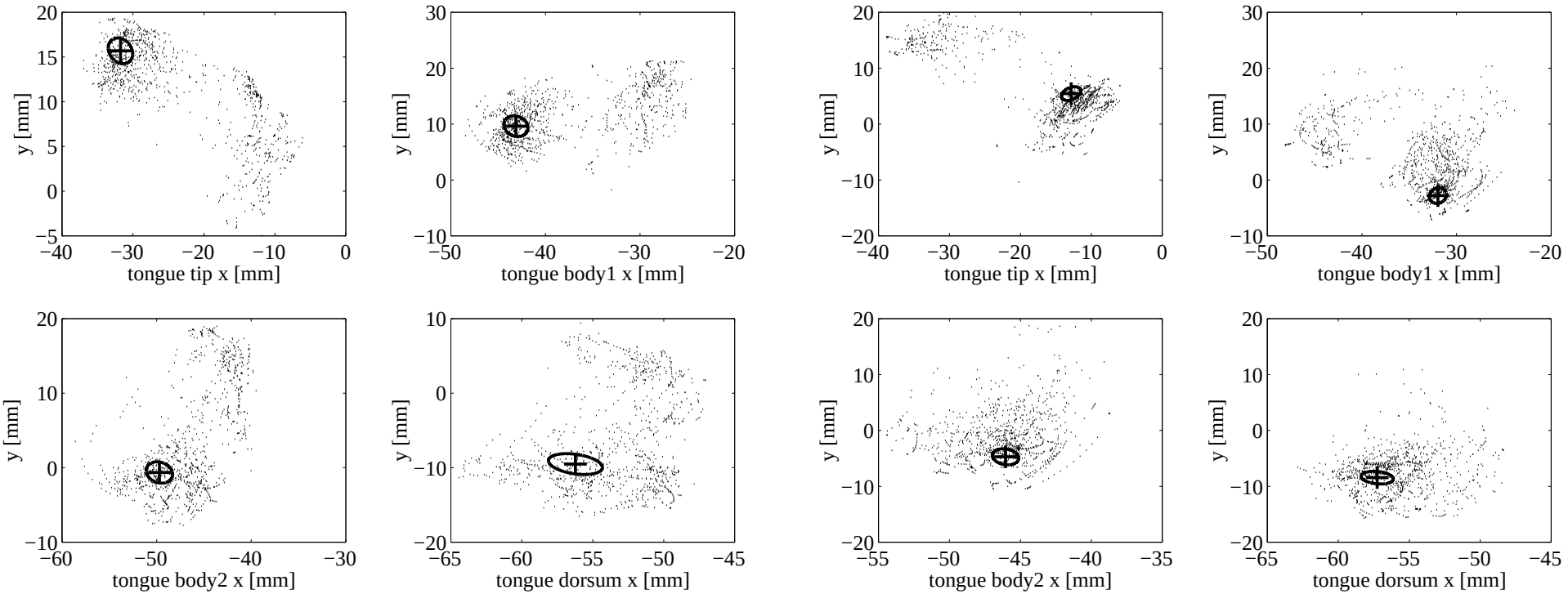
Sensor 3

Instantaneous inversion is ill-posed...



There are **multiple articulatory configurations which produce the same spectral shape**. This means that no instantaneous inversion function is possible since it would be one-to-many. This problem is known from physical models of the vocal tract, but we can **verify it experimentally** by looking at this dataset.

...articulatory to acoustic map is many-to-one



Each group of four plots shows the articulatory configuration of:

1) a single "key-frame" (thick cross) from the database and
2) the 1000 frames (dots) which **sound most like** the key frame (i.e. nearest in spectral distance as measured by LSP Gaussian).

[The four tongue beads are plotted. Data from single speaker.]

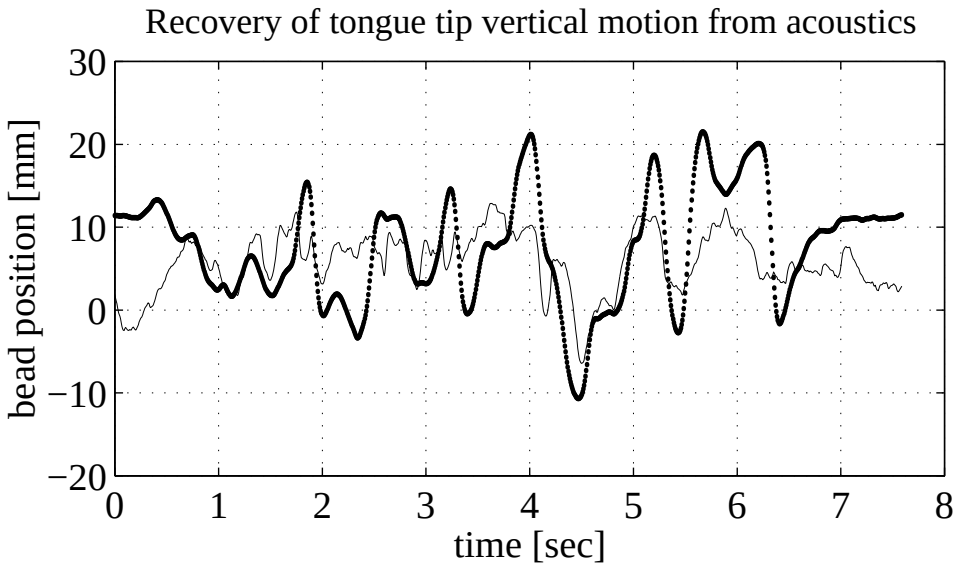
Global Kalman smoothing fails

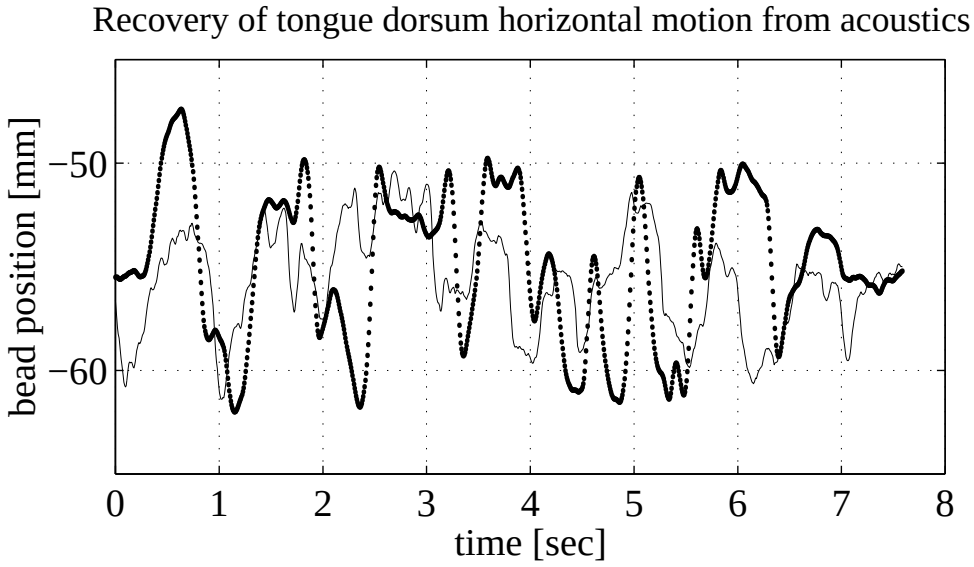
We can construct the **optimal** linear dynamical system model for the data since we know both the noisy observations (short time spectral features of the acoustics) and the hidden states (articulator positions).

Dynamics is random walk in acceleration.

Then we can do exact inference using **Kalman smoothing** to attempt to recover the articulator movements from the speech.

Even when tested on training data, this procedure fails – the globally linear output model simply is not powerful enough.



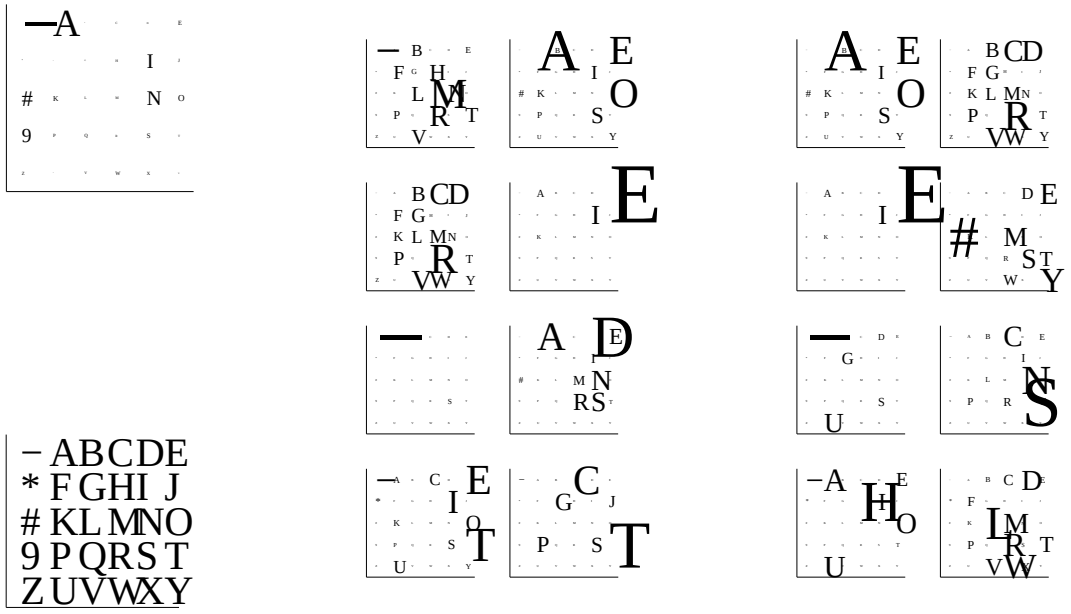


A different example: English text

Transition probabilities do not have to be constrained to allow only **diffusion** type dynamics. Other dynamics constraints, e.g. momentum are possible (may not be exactly Markov).

Unidirectional flow is another extreme, giving gives rise to **left-to-right** HMMs (chain graphs). Constrained HMMs allow mixtures of these chains, possibly of various lengths.

Here, a mixture of chains (18 of len. 3; 8 of 2; 1 of 1) were trained on sequences of characters from English text.



(In the pictures on right, $fontsize \propto \sqrt{probability}$ **so that** $ink \propto probability$**.)**

Decline and Fall of the Roman Empire