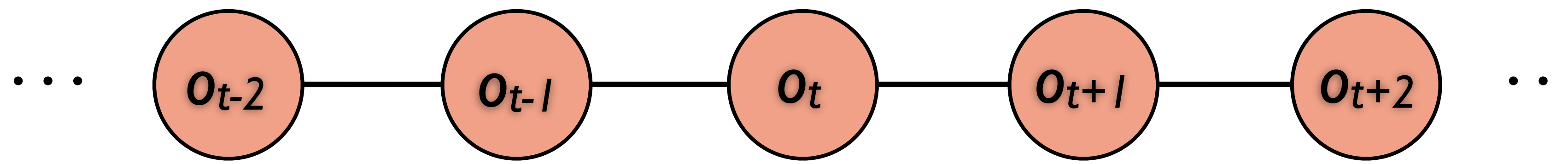
Learning about State

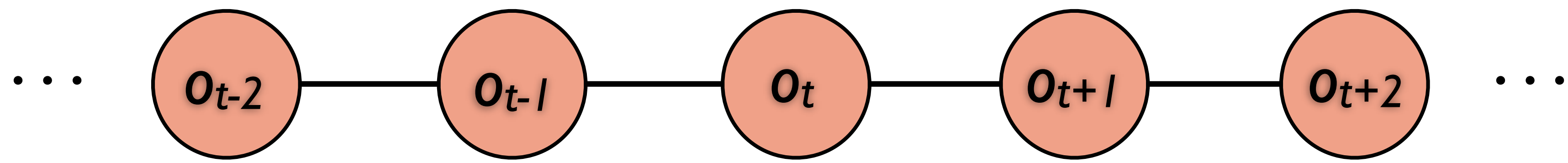
Geoff Gordon
Machine Learning Department
Carnegie Mellon University

joint work with Byron Boots, Sajid Siddiqi, Le Song, Alex Smola

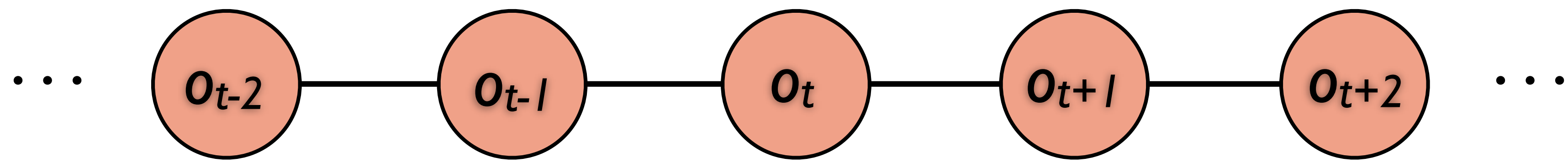
What's out there?



What's out there?

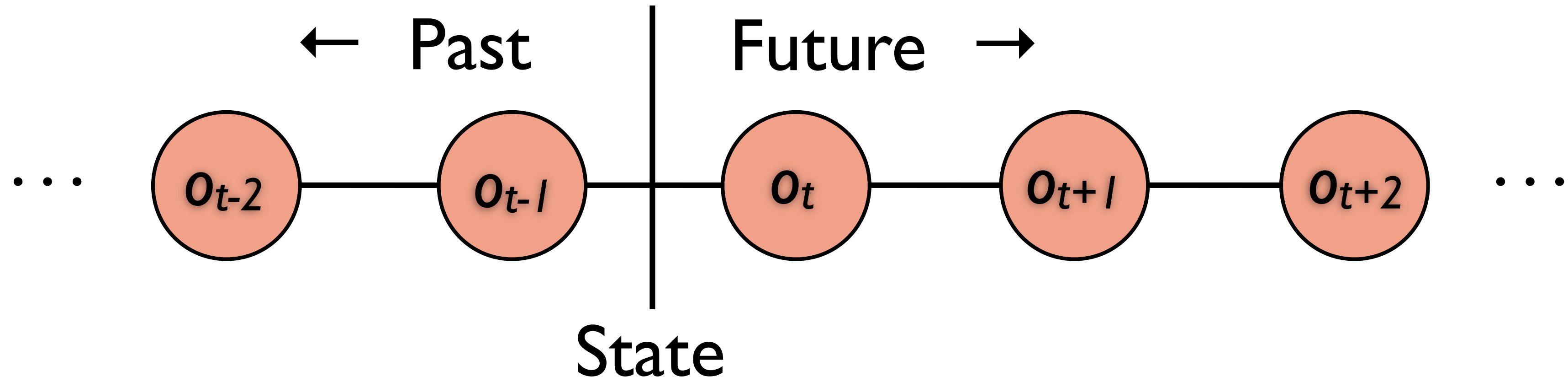


What's out there?



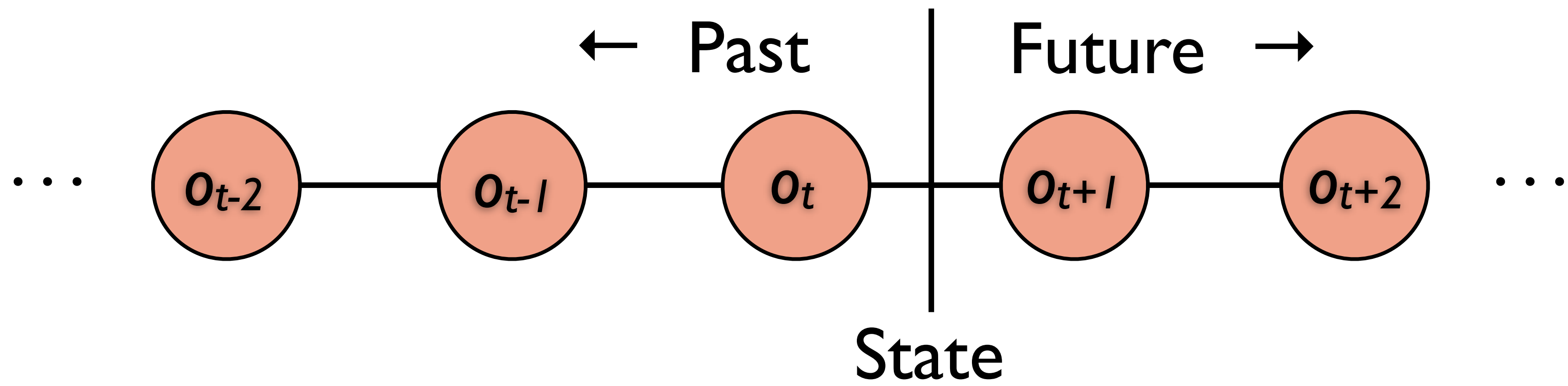
⇒ steam rising from a grate

What's out there? A dynamical system



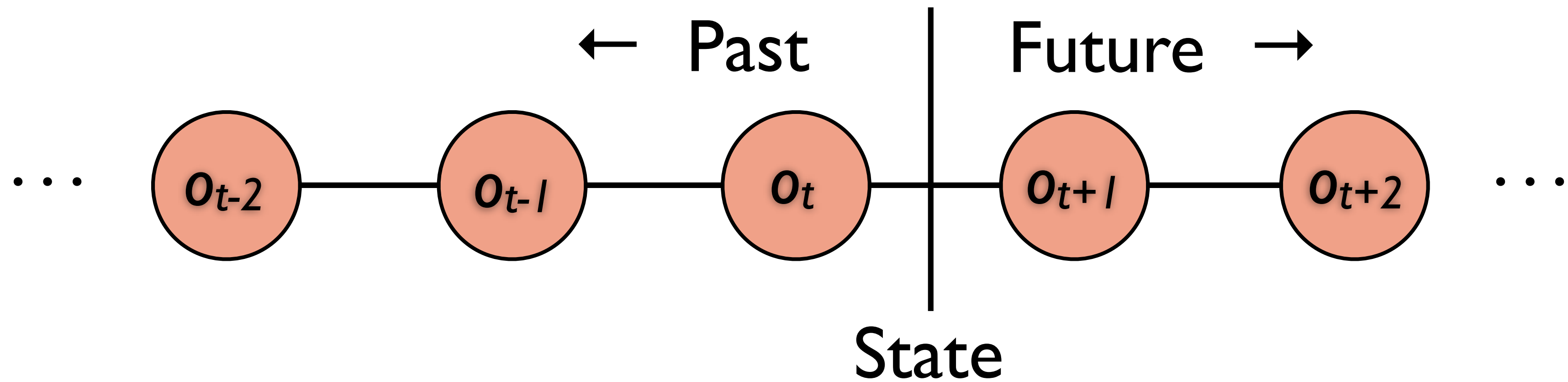
Dynamical system = recursive rule for updating state based on observations

What's out there? A dynamical system



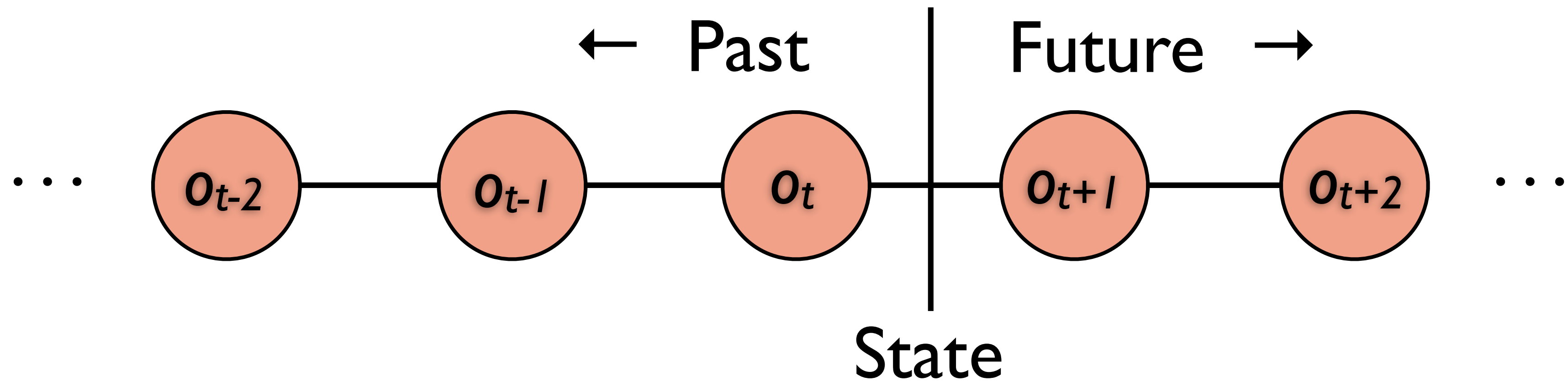
Dynamical system = recursive rule for updating state based on observations

Learning a dynamical system



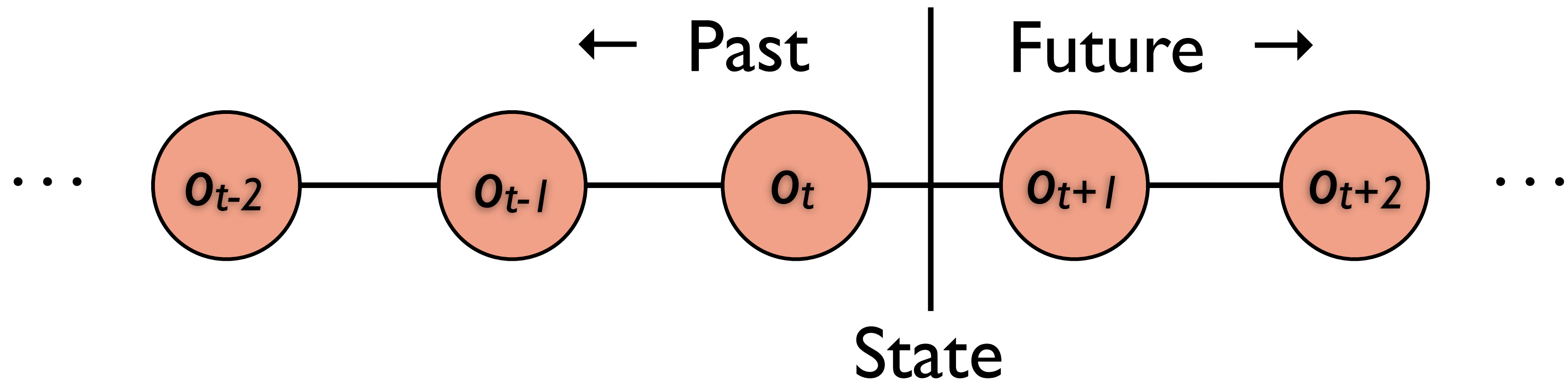
Given past observations from a partially observable system

Learning a dynamical system



Given past observations from a partially observable system

Learning a dynamical system



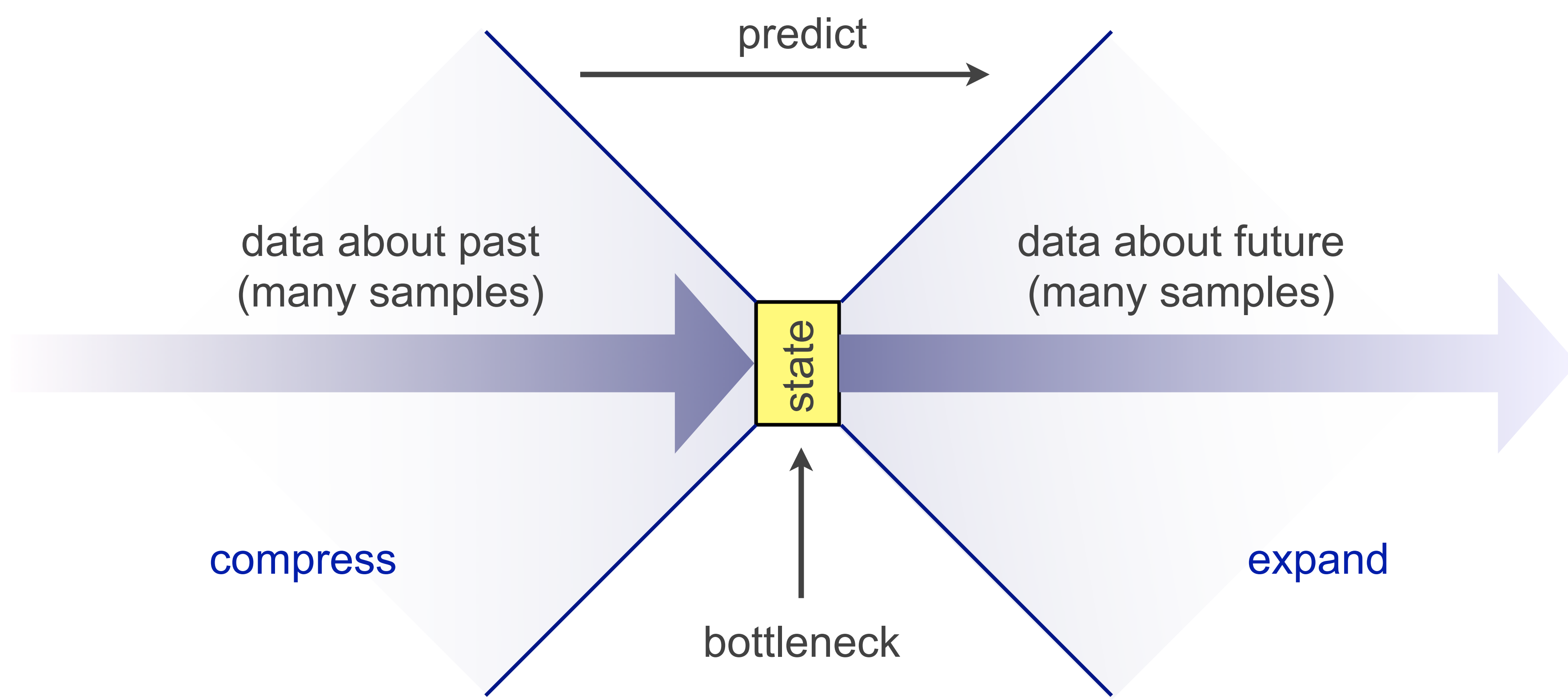
Given past observations from a partially observable system

Predict future observations

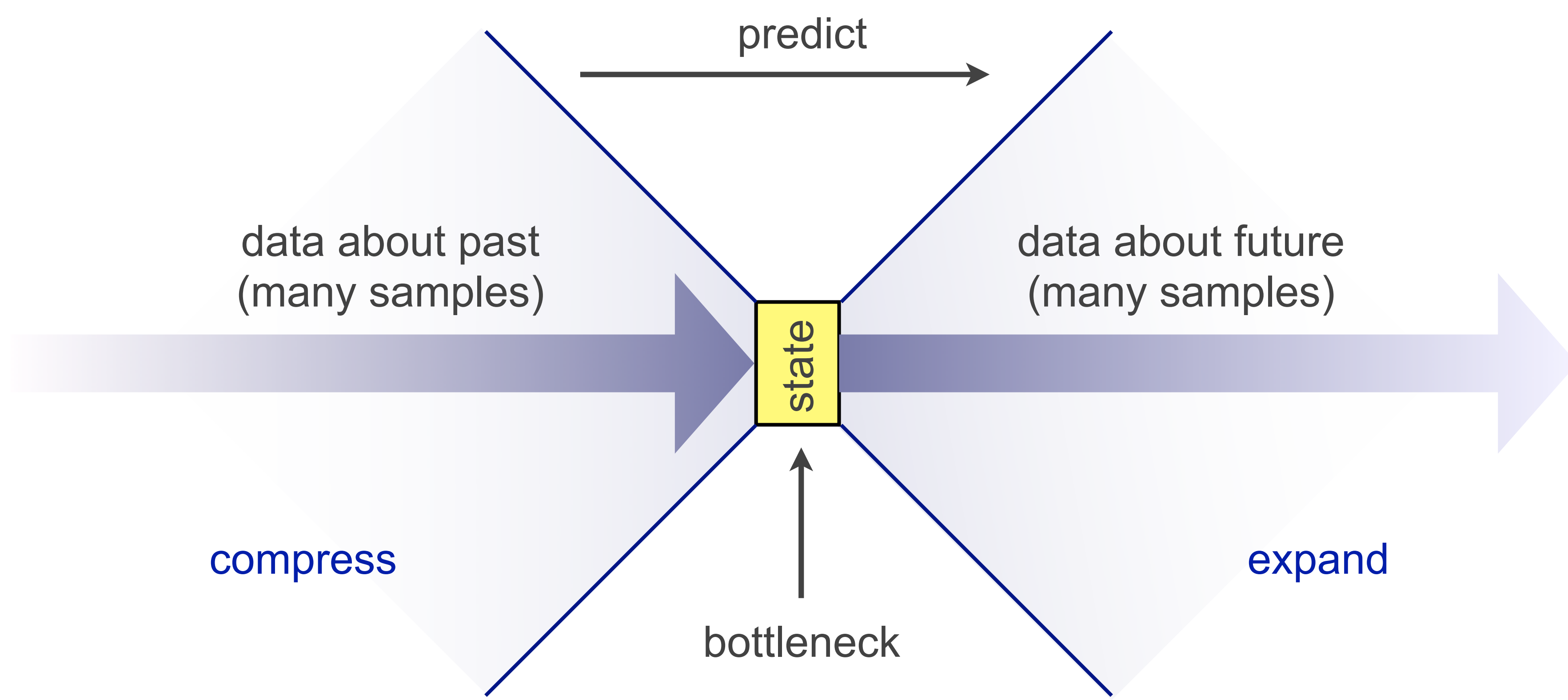
Examples

- Baum-Welsh EM algorithm for HMMs
- Tomasi-Kanade structure from motion
- Black-Scholes model of stock price
- SLAM (from lidars, cameras, beacons, …)
- System identification for Kalman filters
- …

A general principle



A general principle



If bottleneck = rank constraint, get a ***spectral*** method

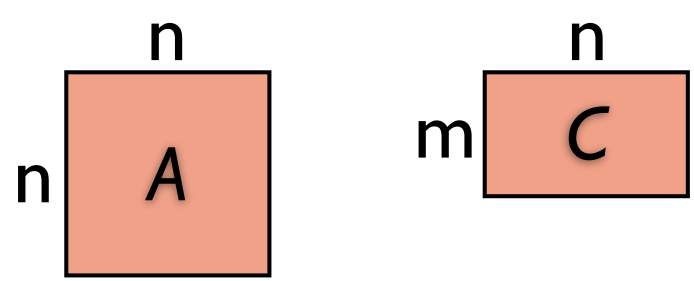
Why spectral methods?

- Many ways to learn models of dynamical systems
  - max likelihood via EM, gradient descent, …
  - Bayesian inference via Gibbs, MH, …
- In contrast to these, spectral methods give
  - ***no local optima!***

    ⇒ huge gain in computational efficiency
  - slight loss in statistical efficiency

Example: SSID for Kalman filter

$$x = A\,x_{-} + \textit{noise}$$
$$o = C\,x + \textit{noise}$$

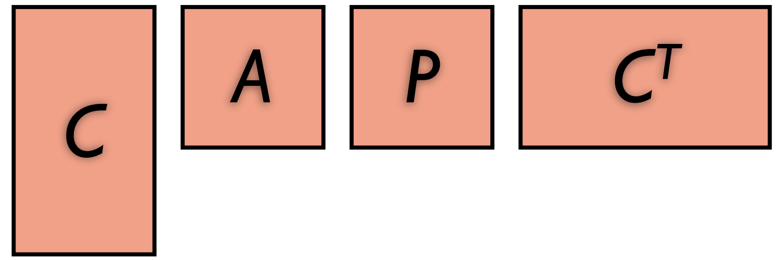


- Past data = last *k* observations
- Future data = next *k'* observations
  - must have *k* and *k'* big enough
- Prediction = linear regression
  - look at empirical covariance of past & future
- Spectral: bottleneck = SVD of covariance

[van Overschee & de Moor, 1996]

Kalman SSID

x = A x₋ + *noise*
o = C x + *noise*



- Assume for simplicity m ≥ n, both A and C full rank
- For k ≥ 1,

$$
\begin{aligned}
\mathbb{E}[o_{t+k} o_t^\top] &= \mathbb{E}[\mathbb{E}[o_{t+k} o_t^\top \mid x_t]] \\
&= \mathbb{E}[\mathbb{E}[o_{t+k} \mid x_t]\mathbb{E}[o_t^\top \mid x_t]] \\
&= \mathbb{E}[C A^k x_t (C x_t)^\top] \\
&= C A^k \mathbb{E}[x_t x_t^\top] C^\top \\
&= C A^k P C^\top
\end{aligned}
$$

Kalman SSID

$$\Sigma_k \;=\; \mathbb{E}[o_{t+k} o_t^\top] \;=\; C A^k P C^\top$$

- Let U = left n leading singular vectors of Σ_1

$$\begin{aligned}
\hat{A} &\stackrel{\text{def}}{=} U^\top \Sigma_2 (U^\top \Sigma_1)^\dagger \\
&= U^\top C A^2 P C^\top (U^\top C A P C^\top)^\dagger \\
&= (U^\top C A) A P C^\top (P C^\top)^\dagger (U^\top C A)^{-1} \\
&= S A S^{-1} \\
\hat{C} &\stackrel{\text{def}}{=} U (S A S^{-1})^{-1} \\
&= U S A^{-1} S^{-1} \\
&= U (U^\top C A) A^{-1} S^{-1} \;=\; C S^{-1}
\end{aligned}$$

Kalman SSID

- Algorithm: estimate Σ_1 and Σ_2 from data, get $\hat{U}$ by SVD, plug in for $\hat{A}$ and $\hat{C}$
- ***Consistent***: continuity of formulas for $\hat{A}$ and $\hat{C}$, law of large numbers for Σ_1 and Σ_2
  - wrinkle: SVD for $\hat{U}$ isn't continuous, but range($\hat{U}$) is
- Can also recover steady-state x

Variations

- Use arbitrary features of length-k window of past and future observations
  - work from covariance of past, future features
  - good features make a big difference in practice
- Impose constraints on learned model (e.g., stability)

Kalman SSID: example

- Works well for ***video textures***
  - steam grate example above
  - fountain:

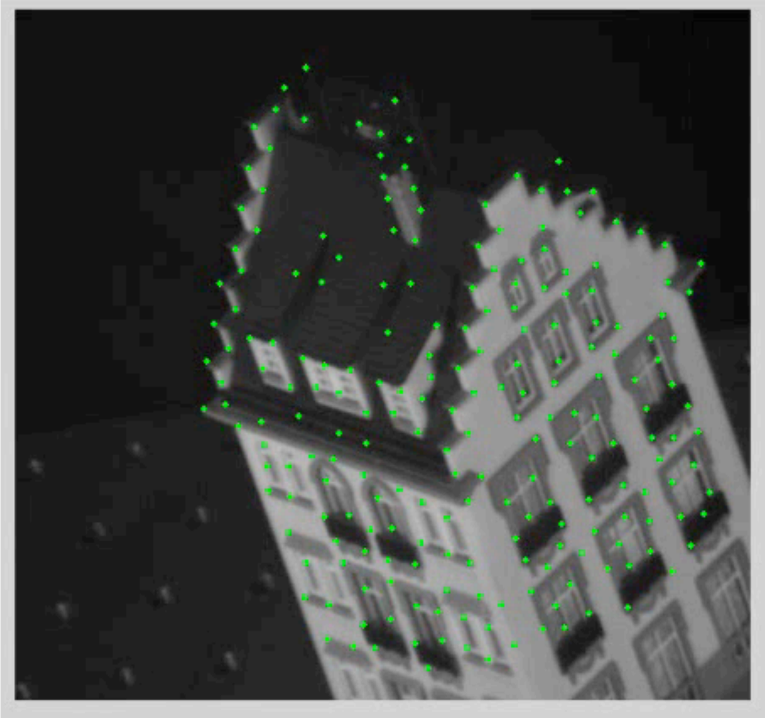
observation = raw pixels (vector of reals over time)

Structure from motion

feature 1, step 2

$$\begin{bmatrix} x_{11} & y_{11} & x_{12} & y_{12} & \cdots & x_{1T} & y_{1T} \\ x_{21} & y_{21} & x_{22} & y_{22} & \cdots & x_{2T} & y_{2T} \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ x_{N1} & y_{N1} & x_{N2} & y_{N2} & \cdots & x_{NT} & y_{NT} \end{bmatrix}$$

- Track N features over T steps

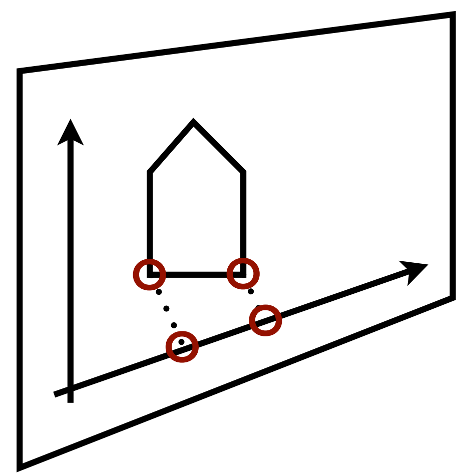


[Tomasi & Kanade, 1992]

Structure from motion

$$\begin{bmatrix} x_{11} & y_{11} & x_{12} & y_{12} & \cdots & x_{1T} & y_{1T} \\ x_{21} & y_{21} & x_{22} & y_{22} & \cdots & x_{2T} & y_{2T} \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ x_{N1} & y_{N1} & x_{N2} & y_{N2} & \cdots & x_{NT} & y_{NT} \end{bmatrix}$$

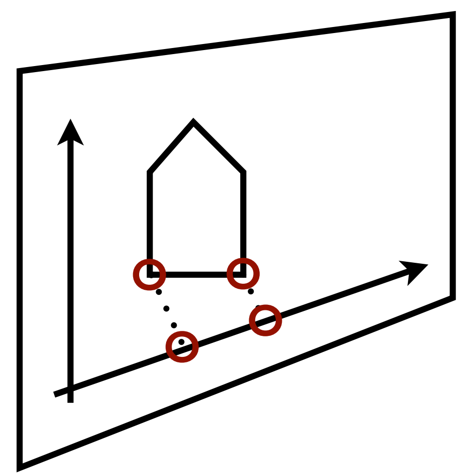
- x_{it} is projection of feature i onto camera's horizontal axis at time t (and y_{it}, vertical)
  - $[u_i, v_i, w_i]$ = feature i coordinates
  - $[h_{1t}, h_{2t}, h_{3t}]$ = camera horizontal axis @t
  - $[v_{1t}, v_{2t}, v_{3t}]$ = camera vertical axis @t



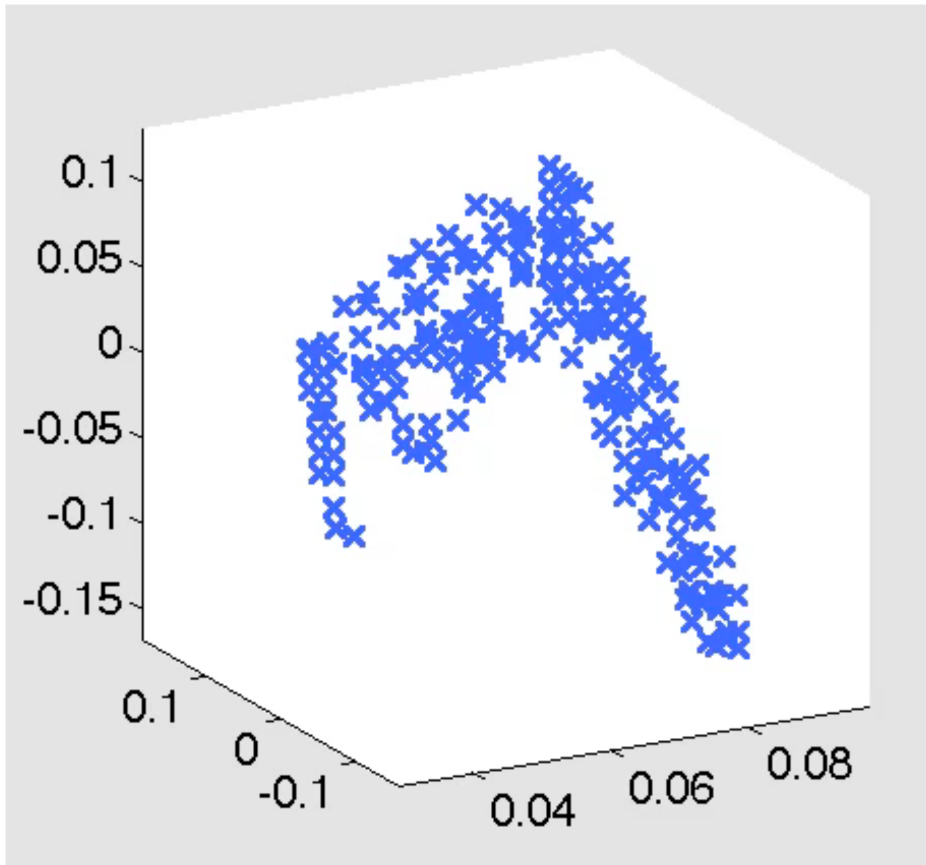
Structure from motion

$$\begin{bmatrix} u_1 & v_1 & w_1 \\ u_2 & v_2 & w_2 \\ \vdots & \vdots & \vdots \\ u_N & v_N & w_N \end{bmatrix} \begin{bmatrix} h_{11} & v_{11} & h_{12} & v_{12} & \dots & h_{1T} & v_{1T} \\ h_{21} & v_{21} & h_{22} & v_{22} & \dots & h_{2T} & v_{2T} \\ h_{31} & v_{31} & h_{32} & v_{32} & \dots & h_{3T} & v_{3T} \end{bmatrix}$$

- x_{it} is projection of feature i onto camera's horizontal axis at time t (and y_{it}, vertical)
  - $[u_i, v_i, w_i]$ = feature i coordinates
  - $[h_{1t}, h_{2t}, h_{3t}]$ = camera horizontal axis @t
  - $[v_{1t}, v_{2t}, v_{3t}]$ = camera vertical axis @t



Structure from motion



- only determined up to an invertible transform

SfM as SSID

$$\text{cov} = \begin{bmatrix} x_{11} & y_{11} & x_{12} & y_{12} & \cdots & x_{1T} & y_{1T} \\ x_{21} & y_{21} & x_{22} & y_{22} & \cdots & x_{2T} & y_{2T} \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ x_{N1} & y_{N1} & x_{N2} & y_{N2} & \cdots & x_{NT} & y_{NT} \end{bmatrix}$$

- Past data: indicator of time step & h/v axis
  - means we get to memorize each time step—no attempt to learn dynamics
- Future data: observed screen coordinates (column of matrix)

Kalman SSID: failure

HMM (Baum-Welsh) Kalman Filter (SSID) Preview…

all models: 10 latent dimensions

Kalman SSID: failure

HMM (Baum-Welsh) Kalman Filter (SSID) Preview…

all models: 10 latent dimensions

Kalman SSID: failure

HMM (Baum-Welsh) Kalman Filter (SSID) Preview…

all models: 10 latent dimensions

Kalman SSID: failure

HMM (Baum-Welsh)

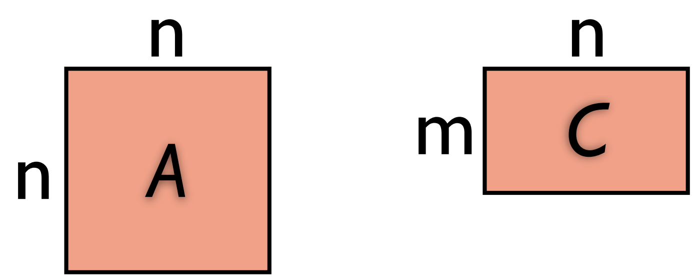
Kalman Filter (SSID)

Preview…

all models: 10 latent dimensions

Can we generalize?

$$x = A\,x_{-} + \textit{noise}$$
$$o = C\,x + \textit{noise}$$



- Get rid of Gaussian noise assumption
- HMM: same form as Kalman filter, but
  - $A \geq 0,\ A\mathbb{1} = 1,\ C \geq 0,\ C\mathbb{1} = 1$
  - noise ~ multinomial
  - x, o are indicators: e.g., "4" = $[0\ 0\ 0\ 1\ 0]^T$

Derivations for Kalman v. HMM

Kalman filter

HMM

$$\begin{aligned}
&\mathbb{E}[o_{t+k}o_t^\top] \\
&\quad = \mathbb{E}[\mathbb{E}[o_{t+k}o_t^\top \mid x_t]] \\
&\quad = \mathbb{E}[\mathbb{E}[o_{t+k} \mid x_t]\mathbb{E}[o_t^\top \mid x_t]] \\
&\quad = \mathbb{E}[CA^k x_t (Cx_t)^\top] \\
&\quad = CA^k\mathbb{E}[x_t x_t^\top]C^\top \\
&\quad = CA^kPC^\top
\end{aligned}$$

- Assume for simplicity m ≥ n, both A and C full rank

Derivations for Kalman v. HMM

Kalman filter

$$\begin{aligned}
&\mathbb{E}[o_{t+k}o_t^\top] \\
&\quad= \mathbb{E}[\mathbb{E}[o_{t+k}o_t^\top \mid x_t]] \\
&\quad= \mathbb{E}[\mathbb{E}[o_{t+k} \mid x_t]\mathbb{E}[o_t^\top \mid x_t]] \\
&\quad= \mathbb{E}[CA^k x_t (Cx_t)^\top] \\
&\quad= CA^k \mathbb{E}[x_t x_t^\top] C^\top \\
&\quad= CA^k P C^\top
\end{aligned}$$

HMM

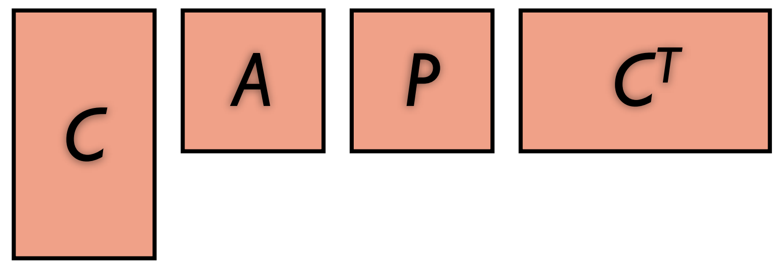
$$\begin{aligned}
&\mathbb{E}[o_{t+k}o_t^\top] \\
&\quad= \mathbb{E}[\mathbb{E}[o_{t+k}o_t^\top \mid x_t]] \\
&\quad= \mathbb{E}[\mathbb{E}[o_{t+k} \mid x_t]\mathbb{E}[o_t^\top \mid x_t]] \\
&\quad= \mathbb{E}[CA^k x_t (Cx_t)^\top] \\
&\quad= CA^k \mathbb{E}[x_t x_t^\top] C^\top \\
&\quad= CA^k P C^\top
\end{aligned}$$

- Assume for simplicity $m \geq n$, both A and C full rank

HMM SSID: first try

$$\begin{aligned} U^\top \Sigma_2 (U^\top \Sigma_1)^\dagger &= U^\top C A^2 P C^\top (U^\top C A P C^\top)^\dagger \\ &= (U^\top C A) A P C^\top (P C^\top)^\dagger (U^\top C A)^{-1} \\ &= S A S^{-1} \end{aligned}$$

- As before, recover Â and Ĉ from $E[o_{t+1} o_t^T]$ & $E[o_{t+2} o_t^T]$
- ***Doesn't*** satisfy A ≥ 0, A**1** = **1**, C ≥ 0, C**1** = **1**
  - is this a problem?



Merging A & C

- HMM tracking: write $b_t = P[x_t \mid o_{1:t}]$
  - $P[x_t \mid o_{1:t-1}] = b_{t-0.5} = A\, b_{t-1}$
  - $P[o_t \mid o_{1:t-1}] = C\, b_{t-0.5}$
  - if $o_t = o$:
    - $P(x_t{=}x \mid o_{1:t}) = P(o \mid x_t{=}x)\, P(x_t{=}x \mid o_{1:t-1}) / Z$
    - i.e., $b_t = \text{diag}(C_{o,:})\, b_{t-0.5} / Z$

where $Z = P(o_t{=}o \mid b_{t-1})$

- Write $A_o = \text{diag}(C_{o,:})\, A$
  - $b_t = A_o\, b_{t-1} / Z$

Merging A & C

- HMM tracking: write $b_t = P[x_t \mid o_{1:t}]$
  - $P[x_t \mid o_{1:t-1}] = b_{t-0.5} = A\, b_{t-1}$
  - $P[o_t \mid o_{1:t-1}] = C\, b_{t-0.5}$
  - if $o_t = o$:
    - $P(x_t{=}x \mid o_{1:t}) = P(o \mid x_t{=}x)\, P(x_t{=}x \mid o_{1:t-1}) / Z$
    - i.e., $b_t = \text{diag}(C_{o,:})\, b_{t-0.5} / Z$

where $Z = P(o_t{=}o \mid b_{t-1})$

- Write $A_o = \text{diag}(C_{o,:})\, A$
  - $b_t = A_o\, b_{t-1} / Z$

It's enough to estimate A_o

Merging A & C

- HMM tracking: write $b_t = P[x_t \mid o_{1:t}]$
  - $P[x_t \mid o_{1:t-1}] = b_{t-0.5} = A\, b_{t-1}$
  - $P[o_t \mid o_{1:t-1}] = C\, b_{t-0.5}$
  - if $o_t = o$:
    - $P(x_t=x \mid o_{1:t}) = P(o \mid x_t=x)\, P(x_t=x \mid o_{1:t-1}) \,/\, Z$
    - i.e., $b_t = \mathrm{diag}(C_{o,:})\, b_{t-0.5} \,/\, Z$

where $Z = P(o_t=o \mid b_{t-1})$

- Write $A_o = \mathrm{diag}(C_{o,:})\, A$
  - $b_t = A_o\, b_{t-1} \,/\, Z$

It's enough to estimate A_o

$P(o_t=o \mid b_{t-1}) = \mathbf{1}^T A_o\, b_{t-1}$

HMM SSID: try #2

$$
\begin{aligned}
\Sigma_2^o &\overset{\text{def}}{=} \mathbb{E}[o_{t+2}(\delta_o^\top o_{t+1})o_t^\top] \\
&= \mathbb{E}[\mathbb{E}[o_{t+2}(\delta_o^\top o_{t+1})o_t^\top \mid x_t]] \\
&= \mathbb{E}[\mathbb{E}[o_{t+2}(\delta_o^\top o_{t+1}) \mid x_t]\mathbb{E}[o_t^\top \mid x_t]] \\
&= \mathbb{E}[\mathbb{E}[o_{t+2} \mid x_t, o_{t+1} = o]\mathbb{P}[o_{t+1} = o \mid x_t](Cx_t)^\top] \\
&= \mathbb{E}[\mathbb{E}[o_{t+2} \mid x_t, o_{t+1} = o](\mathbf{1}^\top A_o x_t)(Cx_t)^\top] \\
&= \mathbb{E}\left[CA\left[\frac{A_o x_t}{\mathbf{1}^\top A_o x_t}\right](\mathbf{1}^\top A_o x_t)(Cx_t)^\top\right] \\
&= \mathbb{E}[CAA_o x_t(Cx_t)^\top] \\
&= CAA_o\mathbb{E}[x_t x_t^\top]C^\top \\
&= CAA_o PC^\top
\end{aligned}
$$

HMM SSID: *try #2*

$x = A_o x_- / P(o)$
$o \sim Cx_-$
$C_{o,:} = e^T A_o$

$$\begin{aligned}\hat{A}_o &\stackrel{\text{def}}{=} U^\top \Sigma_2^o (U^\top \Sigma_1)^\dagger \\ &= U^\top C A A_o P C^\top (U^\top C A P C^\top)^\dagger \\ &= (U^\top C A) A_o P C^\top (P C^\top)^\dagger (U^\top C A)^{-1} \\ &= S A_o S^{-1}\end{aligned}$$

- Estimate Σ_1 and Σ_2^o from data; get $\hat{U} = \text{SVD}(\Sigma_1)$
- Plug in to get $\hat{A}_o$ (for each o)
- Also need $e = S^{-1}\mathbf{1}$ = leading left eigenvector of $A_1 + A_2 + \ldots$

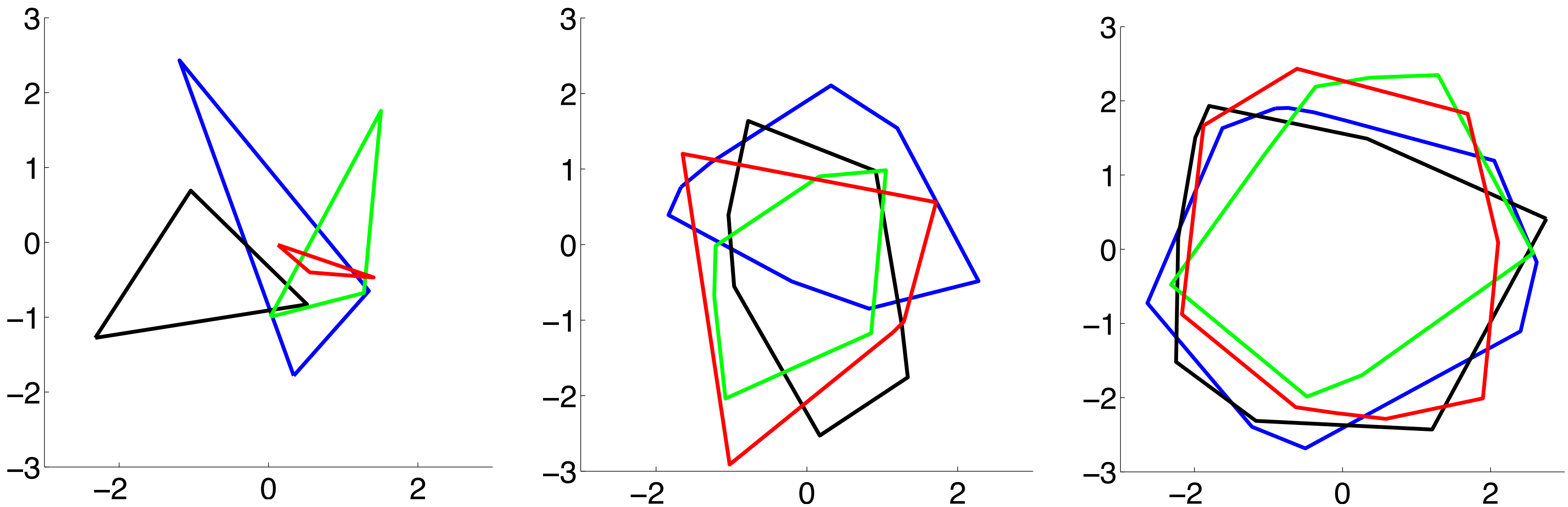
Example: clock

- Discrete observations: sampled frames from training video
  - when tracking: nearest neighbor or Parzen windows (mixture of Gaussians HMM)
- 10 latent dimensions

Can we generalize?

- HMMs had $x \in \Delta$
  - intuition: number of discrete states = number of dimensions
- We now have $x \in S\Delta$
  - essentially equally restrictive
- Can we allow $x \in X$ for general X?
  - # states > # dims

states > # dims: the picture



N=3 N=15 N=100

Random projections of N-dimensional simplex

SSID for OOMs

≈ PSRs without actions, multiplicity automata, …

OOM:
$$x = A_o x_- / P(o)$$
$$o \sim C x_-$$
$$C_{o,:} = e^T A_o$$

HMM:
$$x = A_o x_- / P(o)$$
$$o \sim C x_-$$
$$C_{o,:} = e^T A_o$$

- OOM: defined by transition matrices A_o, normalization vector e
  - like HMM, but lift restriction of $X = S\Delta$
  - instead of $A_o x \geq 0$, have $A_o x \in \lambda X$, $\lambda \geq 0$
  - includes HMM as special case

OOM SSID

- No change!

OOM example

- No change!
  - our HMM SSID was actually learning OOMs all along…