Linear Dynamical Systems

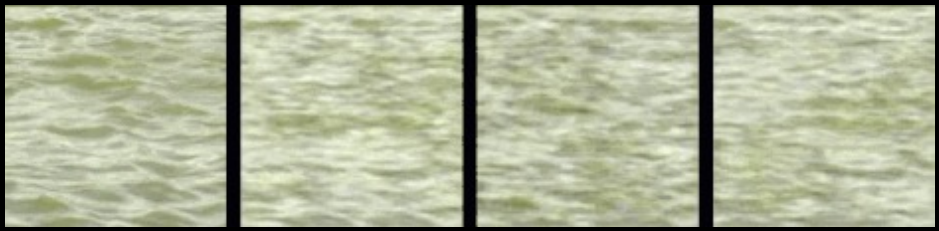
CSCI 4360/6360 Data Science II

Last time…

- Motion analysis via optical flow
- Parametric vs energy-based formulations
- Importance of assumptions
- Modern formulations
  - Robustness to outliers (large optical flow)
  - Relatedness to markov random fields
  - Coarse-to-fine image pyramids

Today

- A specific type of motion: **dynamic textures**



Dynamic textures

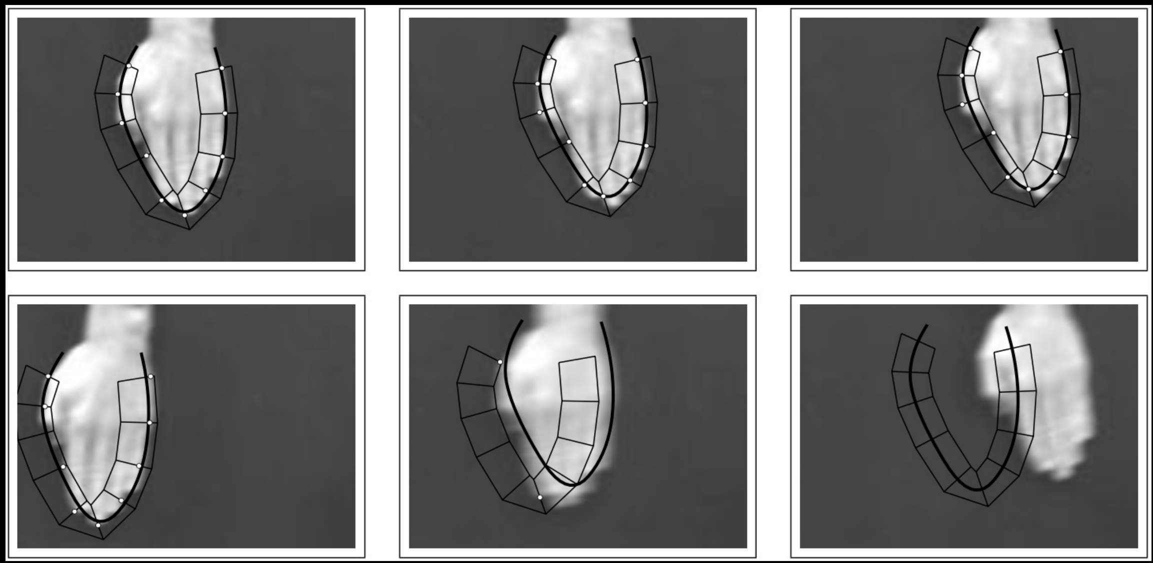
- *"Dynamic textured sequences are scenes with complex motion patterns due to interactions between multiple moving components."*
- Examples
  - Blowing leaves
  - Flickering flames
  - Water rippling
- **Multiple moving components: problematic for optical flow**
- How to analyze dynamic textures?

Dynamical Models

- Goal: an effective procedure for tracking changes over sequences of images, while maintaining a certain coherence of motions

Dynamical Models

- Hand tracking
- Top row: slow movements
- Bottom row: fast movements
- **Fixed curves or priors cannot exploit coherence of motion**



Linear Dynamical Models

- Two main components (using notation from Hyndman 2006):

Appearance Model

$$y_t = Cx_t + u_t$$

State Model

$$x_t = Ax_{t-1} + Wv_t$$

Autoregressive Models

- This is the definition of a 1st-order autoregressive (AR) process!

$$x_t = Ax_{t-1} + Wv_t$$

- Each observation (x_t) is a function of previous observations, plus some noise

- **Markov model!**

Autoregressive Models

- AR models can have higher orders than 1
- Each observation is dependent on the previous *d* observations

$$x_t = A_1 x_{t-1} + A_2 x_{t-2} + \ldots + A_d x_{t-d} + W v_t$$

Appearance Model

- y_t: image of height *h* and width *w* at time *t*, usually flattened into 1 x *hw* vector
- x_t: state space vector at time *t*, 1 x *q* (where *q* <<< *hw*)
- u_t: white Gaussian noise
- *C*: output matrix, maps between spaces, *hw* x *q*

Output matrix

$$y_t = Cx_t + u_t$$

Image in a sequence

Low-dimensional "state"

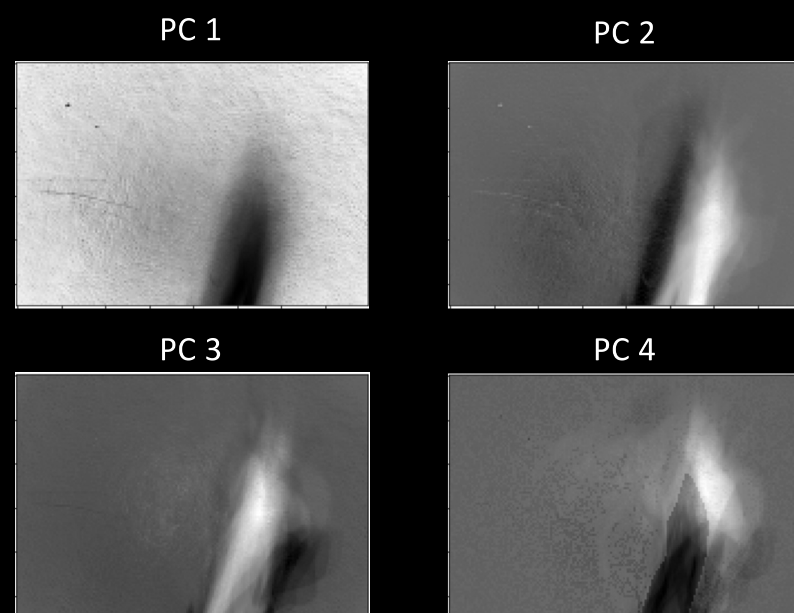
Noise inherent to the system

Appearance Model

$$y_t = Cx_t + u_t$$

Each of these is 1 column of *C.*

There are *q* of them (first 4 shown here).



Appearance Model

- How do we learn the appearance model?
- Choose state-space dimension size *q*
- Noise term is i.i.d Gaussian

$$y_t = Cx_t + u_t$$

$$Y = [\vec{y}_1, \vec{y}_2, ..., \vec{y}_f]^T$$

$$Y = U \Sigma V^T$$

$$C = \hat{U}$$

U-hat is a matrix of the first *q* columns of *U*

$$X = \hat{\Sigma} \hat{V}^T$$

V-hat is a matrix of the first *q* columns of *V*, and sigma-hat is a diagonal matrix of the first *q* singular values

State Model

- x_t and x_{t-1}: state space vectors at times *t* and *t – 1*, each 1 x *q* vector
- *A*: transition matrix, *q* x *q* matrix
- *W*: driving noise, *q* x *q* matrix
- v_t: white Gaussian noise

State transition matrix

$$x_t = Ax_{t-1} + Wv_t$$

Low-dimensional state at time *t*

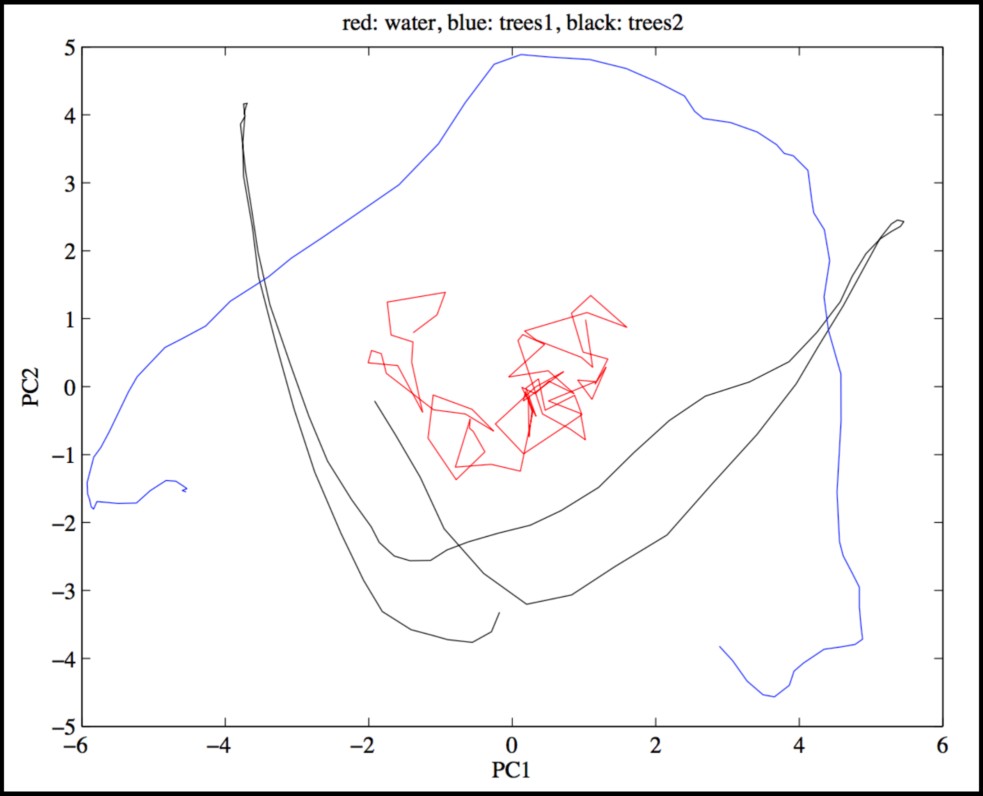
Low-dimensional state at *t - 1*

Driving noise

State Model

$$x_t = Ax_{t-1} + Wv_t$$

- Three textures
- *q = 2*



State Model

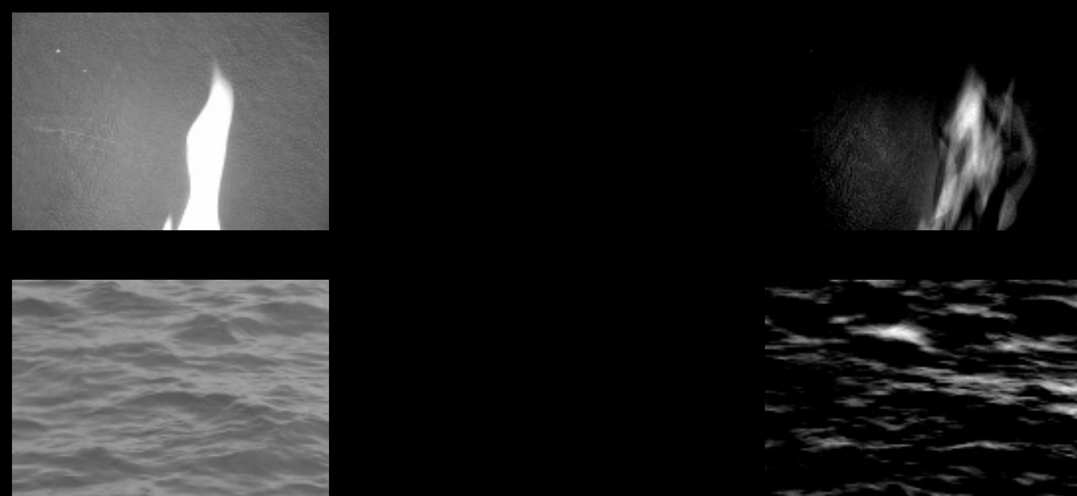
- How do we learn the state model?

$$x_t = Ax_{t-1} + Wv_t$$

- **Homework 2, ahoy!**

LDS as Generative Models

- Once we've learned the parameters, we can *generate new instances*



- Major strength of LDS!

Problems with LDS

- PCA = Linear + Gaussian
- What if the *state space* isn't linear, or data aren't Gaussian?

- Nonlinear appearance models
  - Wavelets
  - IsoMap
  - LLE
  - Kernel PCA
  - Laplacian Eigenmaps

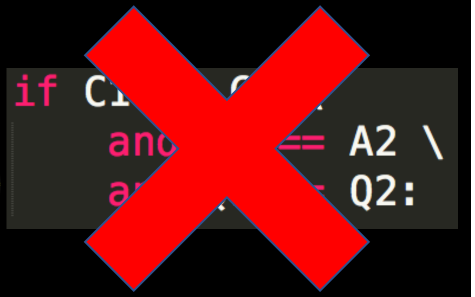
- These introduce their own problems!

Problems with LDS

- Comparing LDS models

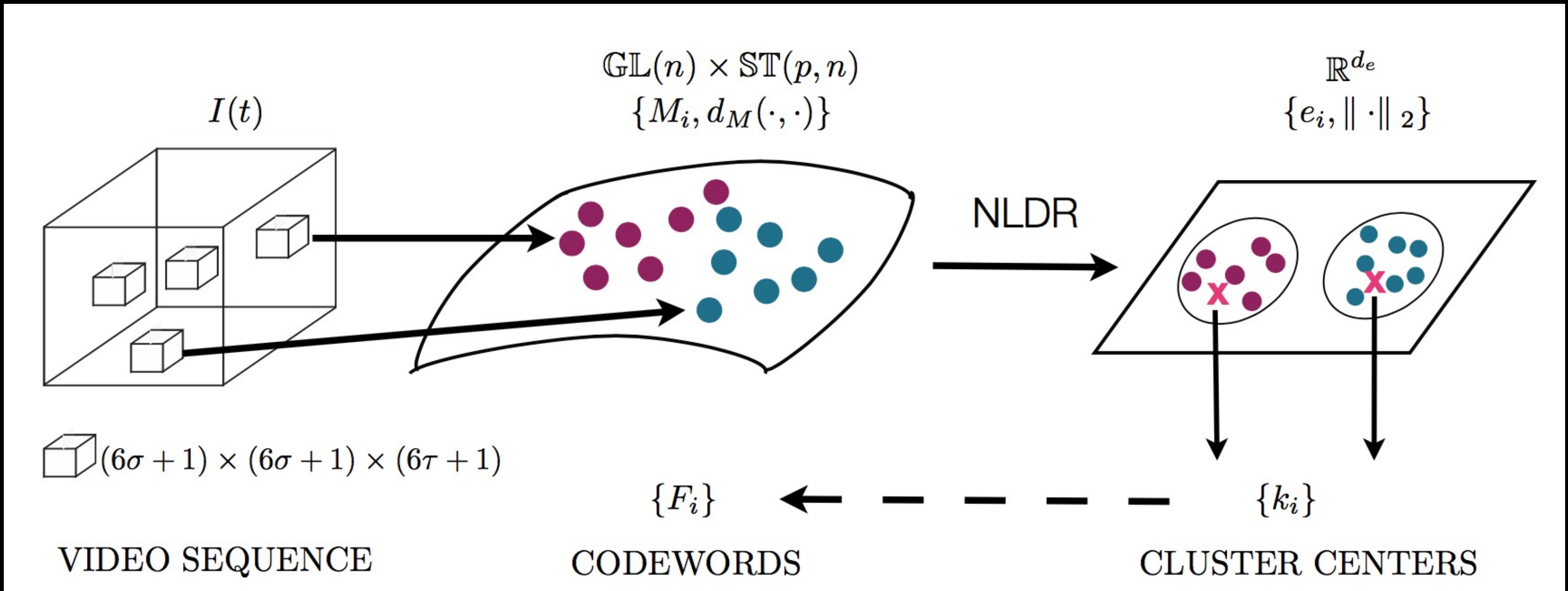
- Given a sequence *Y*:
- New sequence *Y'*:
- **How do we compare these systems?**
- Despite linear formulation, θ are NOT Euclidean
- Valid distance metrics include spectral methods and distribution comparators

$$\theta = (C, A, Q)$$
$$\theta' = (C', A', Q')$$



Comparing LDS

- Select multiple, non-overlapping patches from each video
- Build LDS for each patch



Comparing LDS

- Embed the LDS in low-dimensional space
  - We'll come back to this when we discuss embeddings!
- Compute cluster centroids in embedding space
  - These centroids become *codewords*
- Represent videos as a *document of codewords*
  - Compute TF-IDF
- Perform classification on document weight vectors

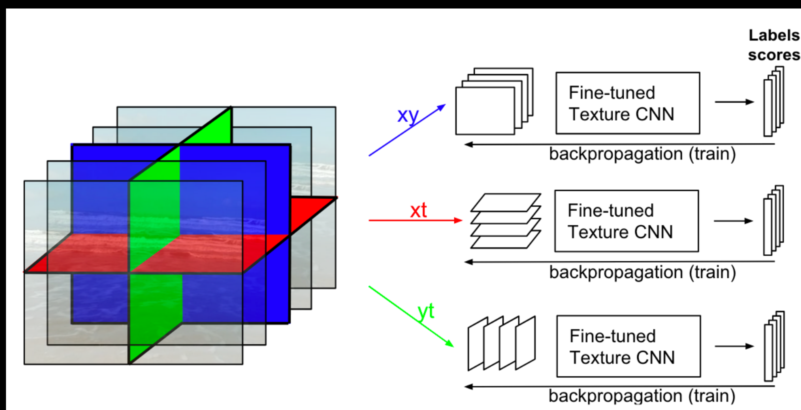
$$p = \arg\min_j \|e_j - k_i\|^2$$

$$w_{ik} = \frac{N_{ki}}{N_i} \ln\left(\frac{V}{V_i}\right)$$

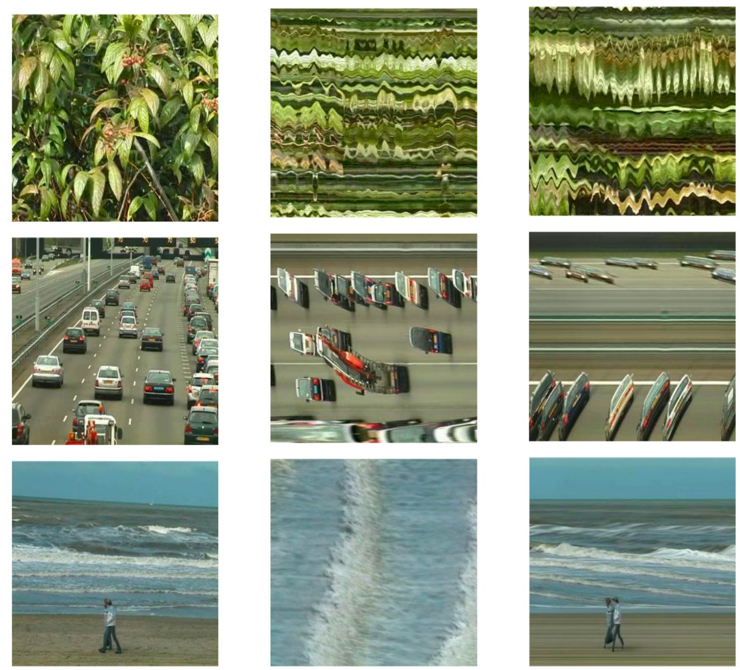
Comparing LDS

	Boiling	Fire	Flowers	Fountain	Sea	Smoke	Water	Waterfall
Boiling	1.0	.00	.00	.00	.00	.00	.00	.00
Fire	.00	.98	.00	.00	.00	.02	.00	.00
Flowers	.00	.00	1.0	.00	.00	.00	.00	.00
Fountain	.00	.00	.02	.50	.00	.00	.00	.48
Sea	.00	.00	.00	.00	1.0	.00	.00	.00
Smoke	.00	.90	.00	.00	.00	.10	.00	.00
Water	.00	.46	.00	.00	.00	.03	.51	.00
Waterfall	.00	.00	.00	.00	.00	.00	.00	1.0

Deep learning + dynamic textures



	boil	fire	flow.	pl.	fount	sea	sm.	wat.	wfall
boil	1	0	0	0	0	0	0	0	0
fire	0	0.97	0	0	0	0	0.03	0	0
flower	0	0	0.92	0	0.08	0	0	0	0
fount	0	0	0	0.96	0	0	0	0	0.04
plants	0	0	0	0	1	0	0	0	0
sea	0	0	0	0	0	0.97	0	0.03	0
smoke	0	0	0	0	0	0	0.75	0.25	0
water	0	0	0	0	0	0	0.03	0.97	0
wfalls	0	0	0	0.01	0	0	0	0	0.99



(a) xy **(b)** xt **(c)** yt

Figure 4.3: Examples of DT slices in three orthogonal planes of foliage, traffic and sea sequences from the DynTex database. (a) *xy* (spatial), (b) *xt* (temporal) and (c) *yt* (temporal).

Conclusion

- Dynamic textures are motion with statistical regularity
- Regularity can be exploited through parametric representation
- Linear dynamical systems (LDS)
  - Autoregressive models (AR)
  - Markov assumption
  - Representation model + State model
  - Generative models
- Deep networks can learn the same feature set and in some cases exceed the performance of LDS (though are harder to train)

References

- Hyndman *et al*, “Higher-order Autoregressive Models for Dynamic Textures”, BMVC 2007 http://www.cs.toronto.edu/~fleet/research/Papers/DynamicTextures.pdf
- Shumway and Stoffer, *Time Series Analysis and Its Applications* (3rd ed.), Chapters 3 and 6, http://www.db.ucsd.edu/static/TimeSeries.pdf
- Blake and Isard, *Active Contours*, Chapter 9-11, http://ww.vavlab.ee.boun.edu.tr/courses/574/material/books/blake_Active_Contours.pdf
- Doretto *et al*, “Dynamic Textures”, IJCV 2003 https://escholarship.org/uc/item/2m50f2fb
- Woolfe *et al*, “Shift Invariant Dynamic Texture Recognition”, ECCV 2006 https://link.springer.com/chapter/10.1007%2F11744047_42?LI=true
- Ravichandran *et al*, “View-invariant dynamic texture recognition using a bag of dynamical systems”, CVPR 2009 http://ieeexplore.ieee.org/abstract/document/5206847/
- Andrearczyk, Vincent. “Deep learning for texture and dynamic texture analysis”, 2020 https://doras.dcu.ie/22040/1/Vincent_Andrearczyk_Final_PhD_thesis.pdf