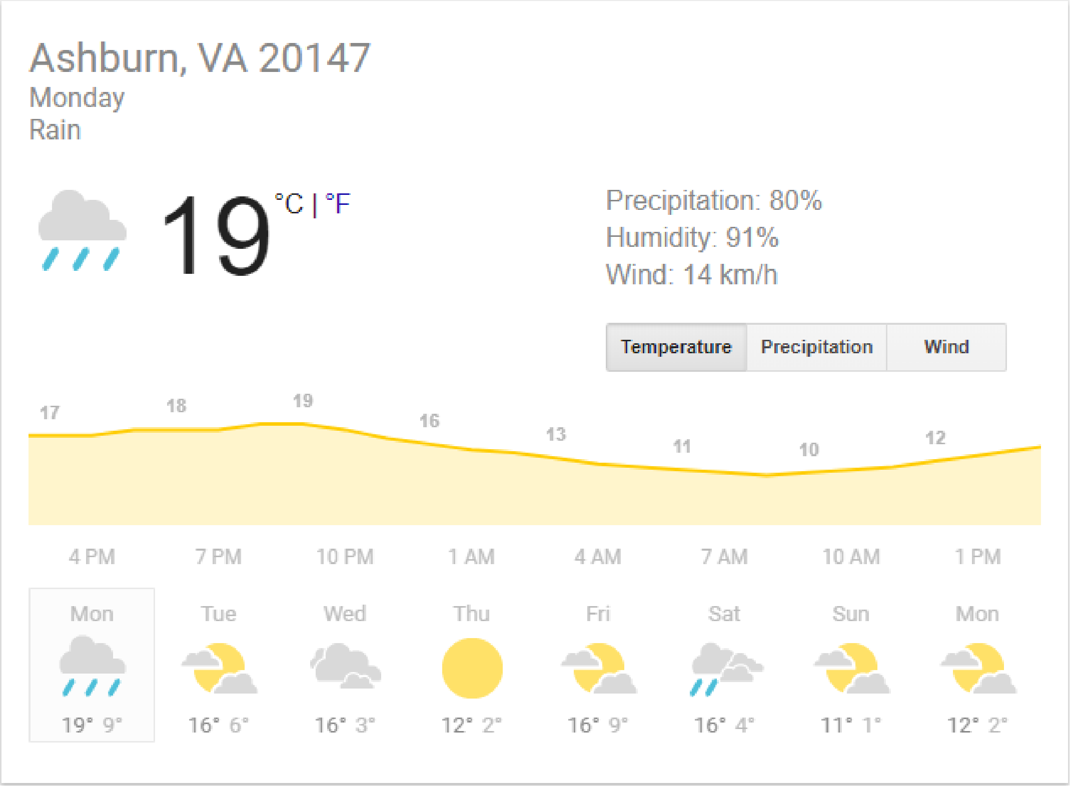
Time series

Marius Pachitariu

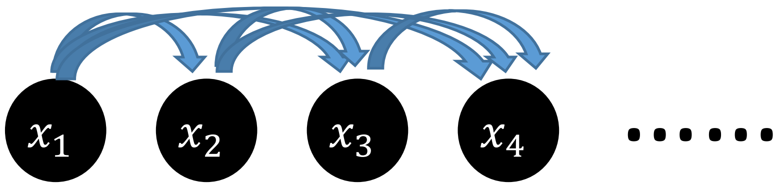
Why model time series data?

- Prediction, i.e. stocks, weather



Prediction can be useful for modelling

- Predict next item based on all previous items.
- Probabilities decompose.



$$\mathrm{P}(x_1, \ldots, x_n) = \mathrm{P}(x_n | x_1, \ldots, x_{n-1}) \cdot \mathrm{P}\,(x_1, \ldots, x_{n-1})$$

$$\ldots$$

$$\mathrm{P}(x_1, \ldots, x_n) = \mathrm{P}(x_n | x_1, \ldots, x_{n-1}) \cdot \mathrm{P}(x_{n-1} | x_1, \ldots, x_{n-2}) \cdot \ldots \cdot \mathrm{P}(x_2 | x_1) \cdot \mathrm{P}(x_1)$$

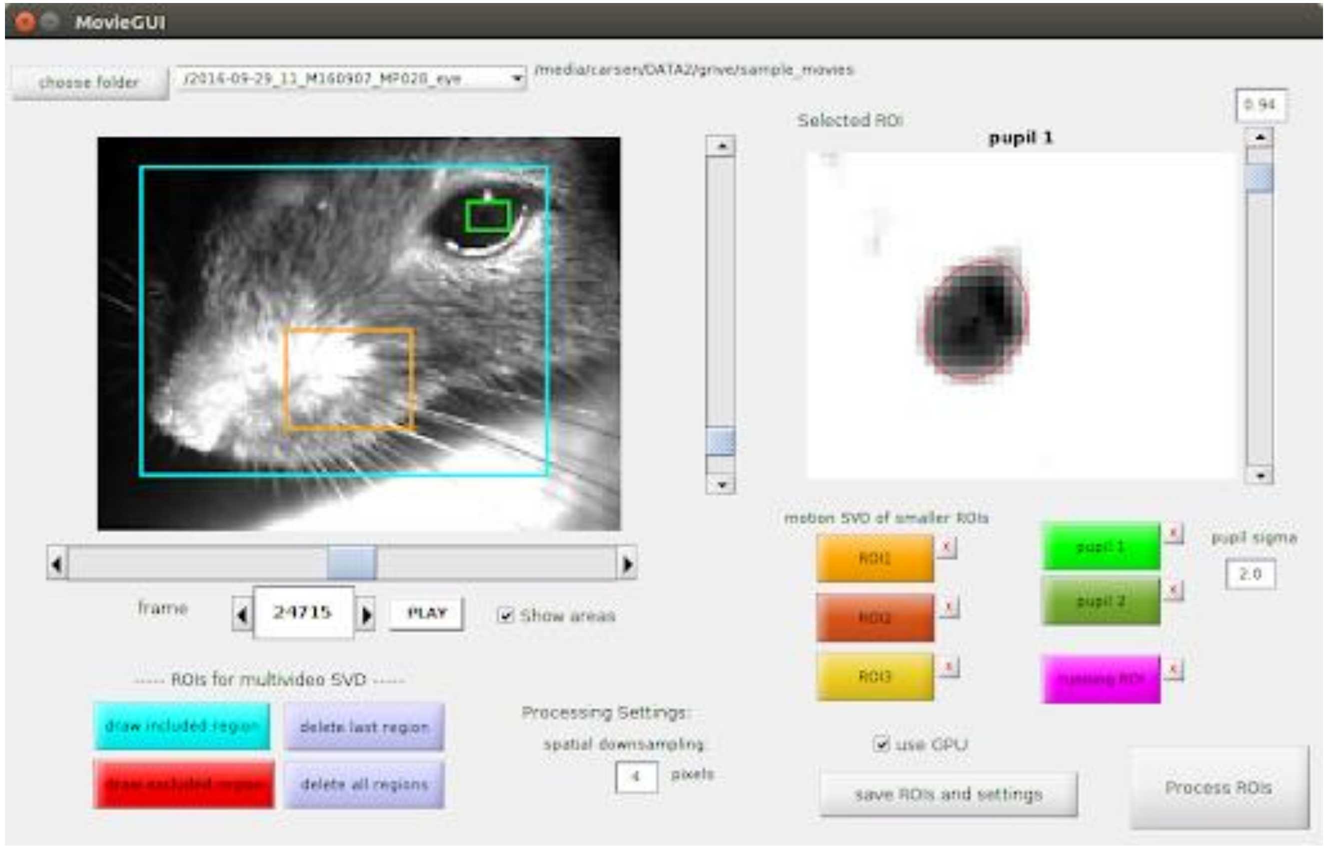
The German Land Forces had been reversed in the early 1990s , although the Soviet Union continued to deter NDH forces in the nation . The area was moved to Sarajevo , and the troops were despatched to the National Register of Historic Places in the summer of 1918 for the establishment of full political and social parties . The Polish language was protected by the Soviet Union , which was the first Polish continental conflict of the newly formed Union in North America , and the Polish Front with the last of the Polish Communist Party .

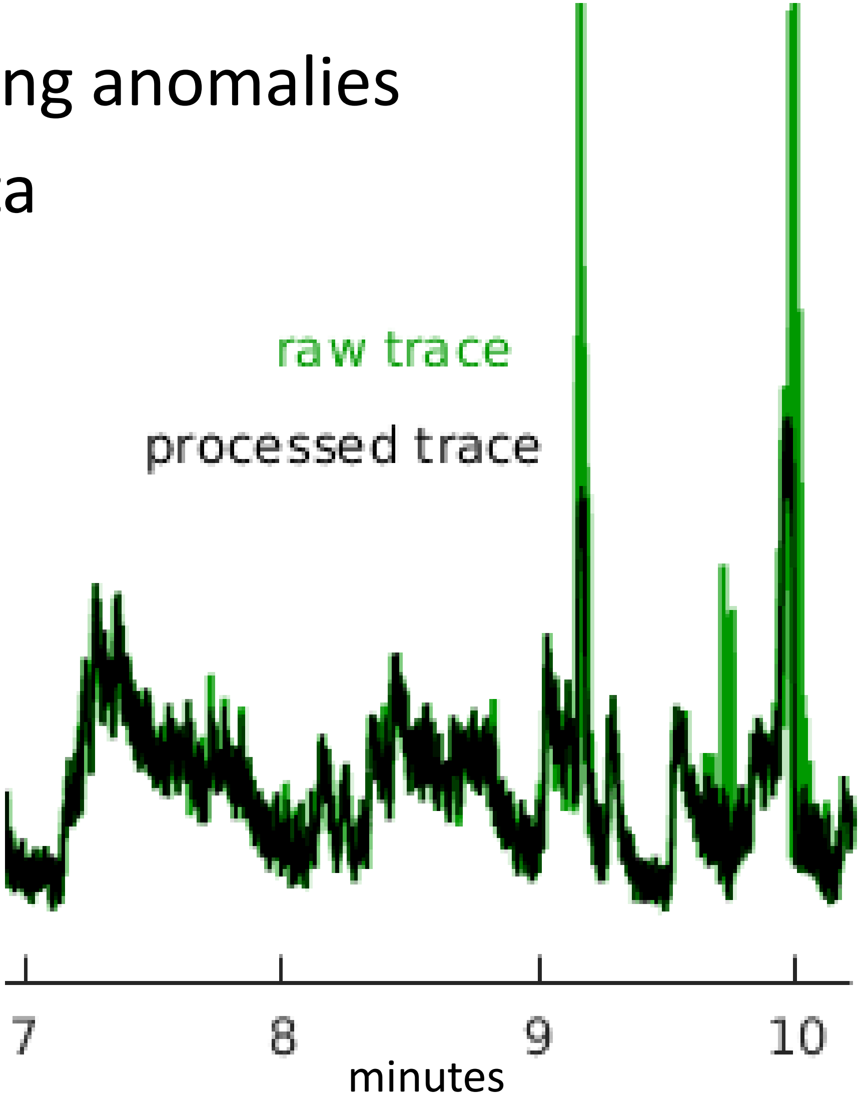
Merity et al, ICML 2017

@DeepDrumpf: I'm a Neural Network trained on Trump's transcripts.

Why model time series data?

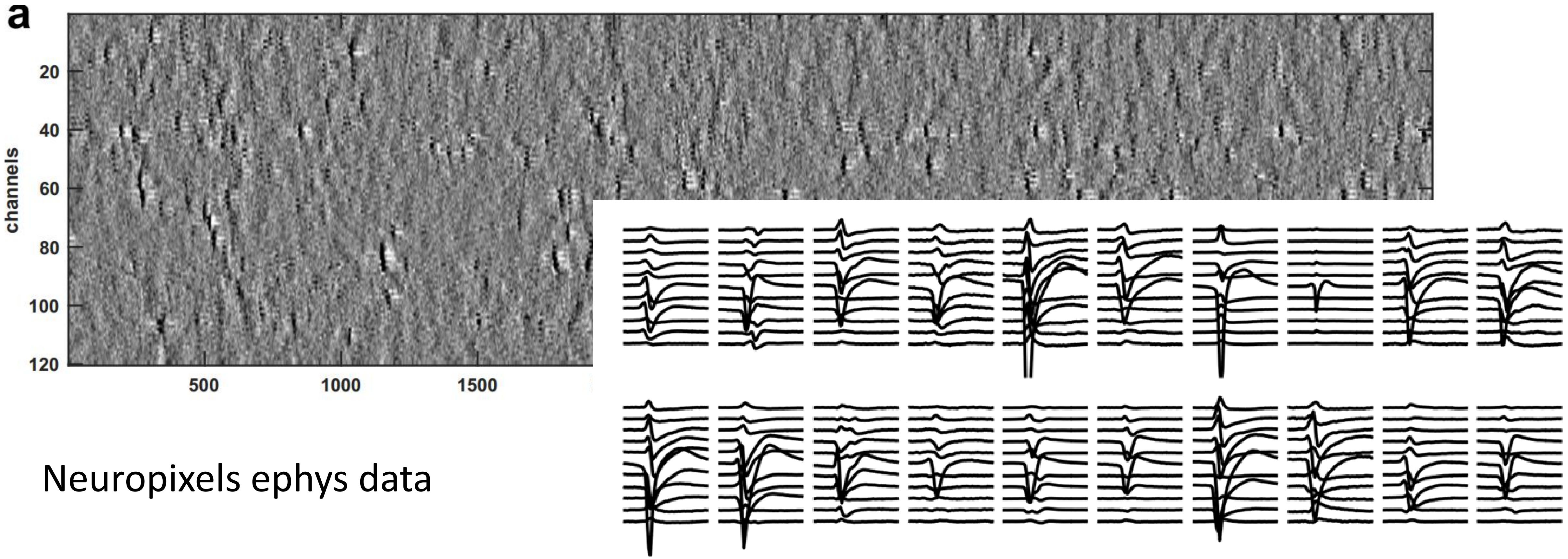
- Outlier detection, i.e. corrupted data, interesting anomalies
- Change detection, i.e. non-stationarities in data





Why model time series data?

- Data mining: i.e. find repeating patterns (ephys data, DNA)



Neuropixels ephys data

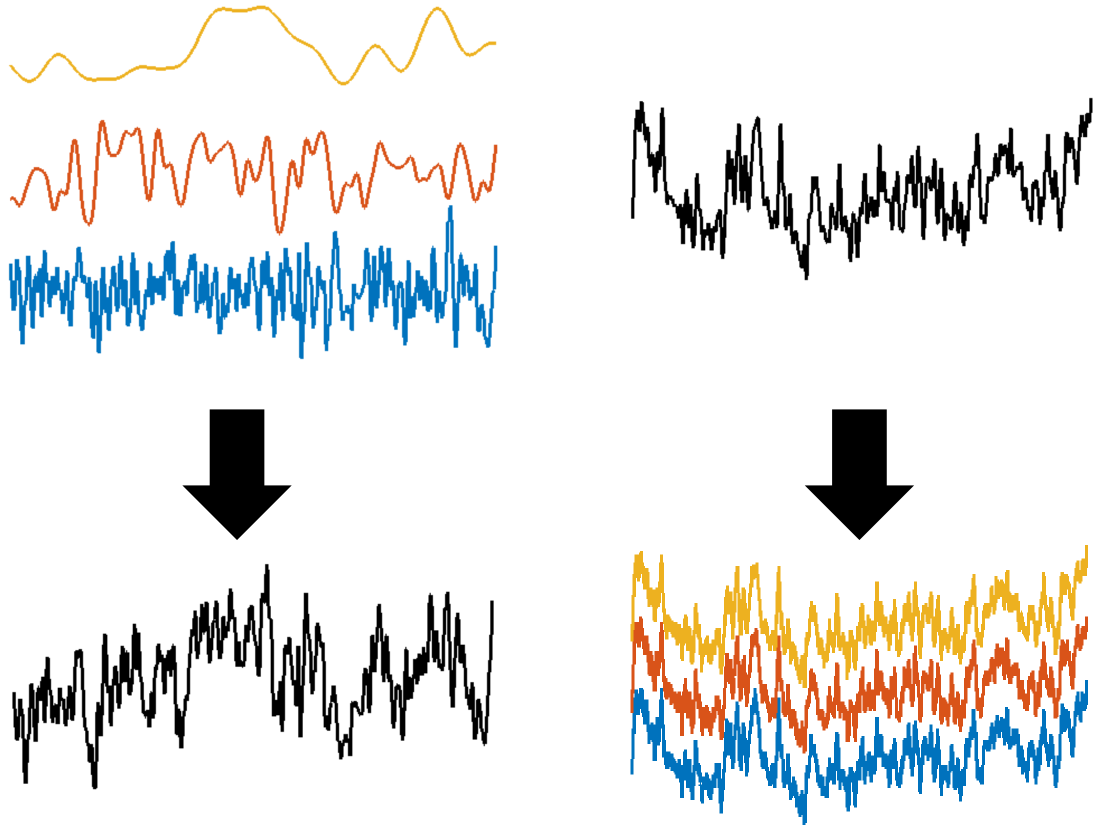
Why model time series data?

- Visualization: i.e. seeing the dynamics of an evolving process

Churchland, Cunningham et al, 2012

Overview of methods

- univariate vs multivariate
  - no difference, other than computational load
  - however:
    - a univariate signal can be generated by a multivariate process
    - a multivariate signal can be generated by a univariate (or low-D) process

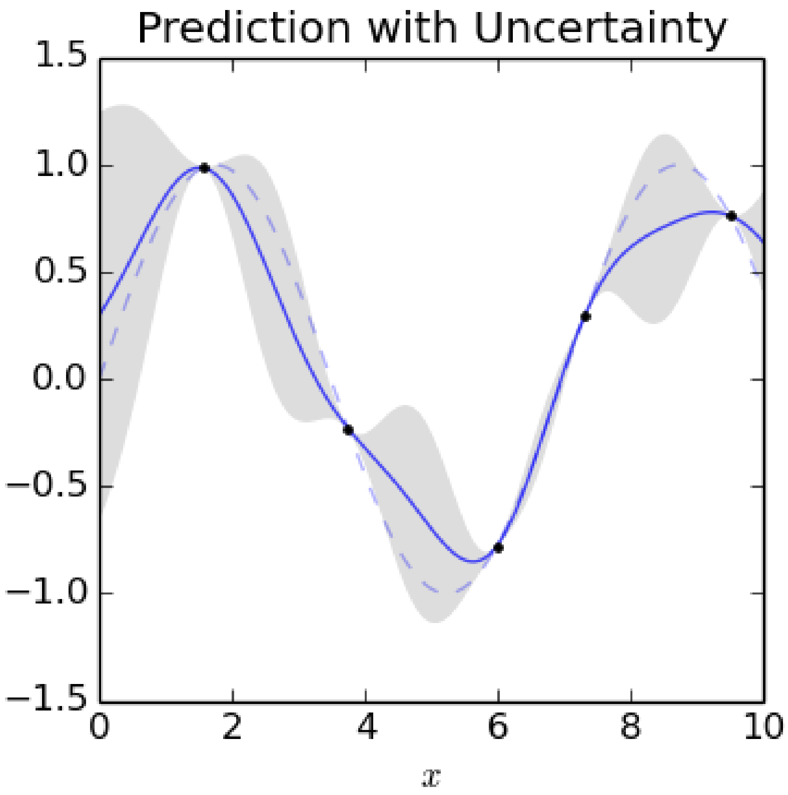


Overview of methods

- Linear models (or "filtering")
  - Fourier-domain
  - Wiener (aka regression)
  - autoregressive filter
  - Kalman filter
  - Gaussian process (or kriging)
- Nonlinear models
  - median filtering
  - deconvolution
  - Wavelets
  - HMM (discrete)
  - blackbox: RNN, seq-to-seq

Basics1. Interpolation.

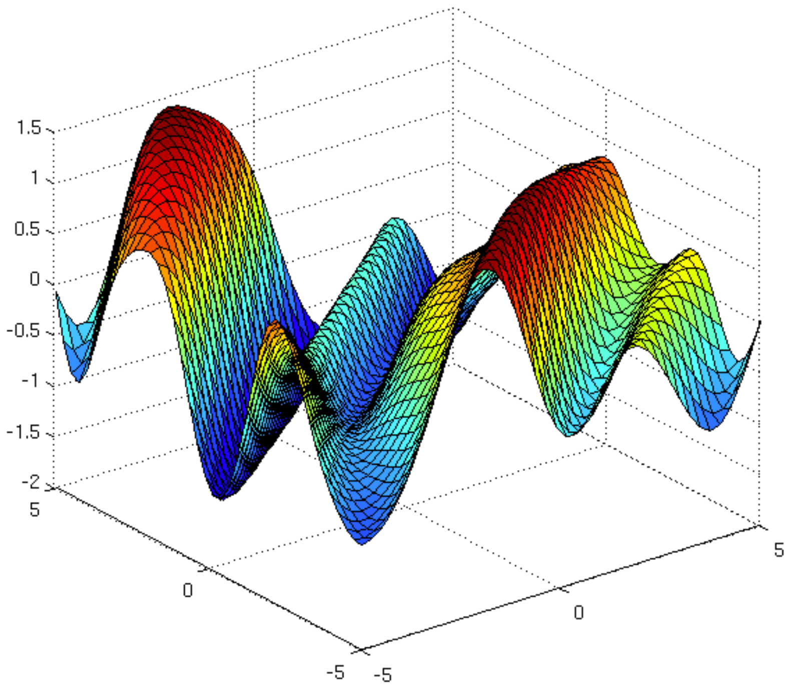
- linear interpolation
- quadratic, cubic interpolation
- Gaussian process interpolation (kriging)



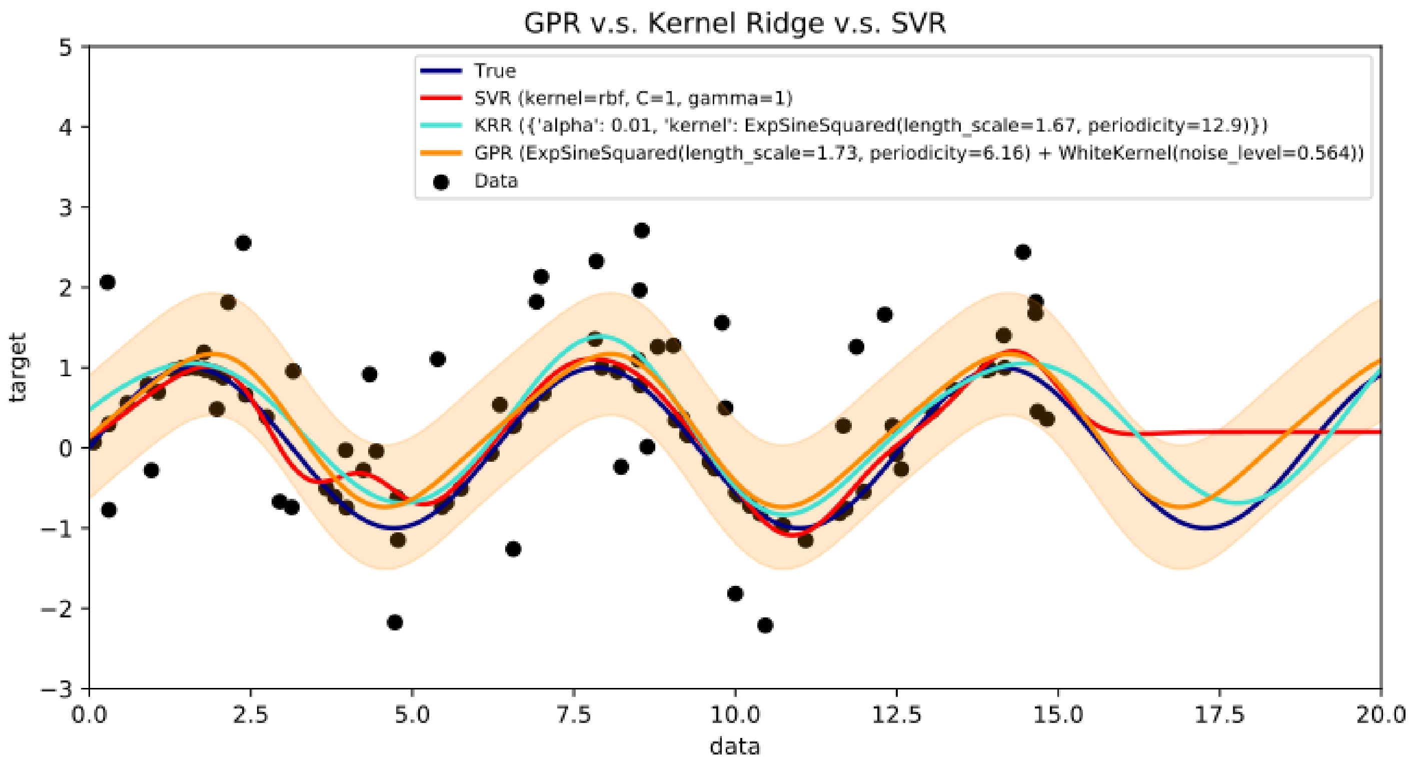
Gaussian processes

- multi-dimensional

- flexible: different kernels produce different filters!



http://katbailey.github.io/post/gaussian-processes-for-dummies/



from scikit-learn docs

Basics2. Fourier filtering

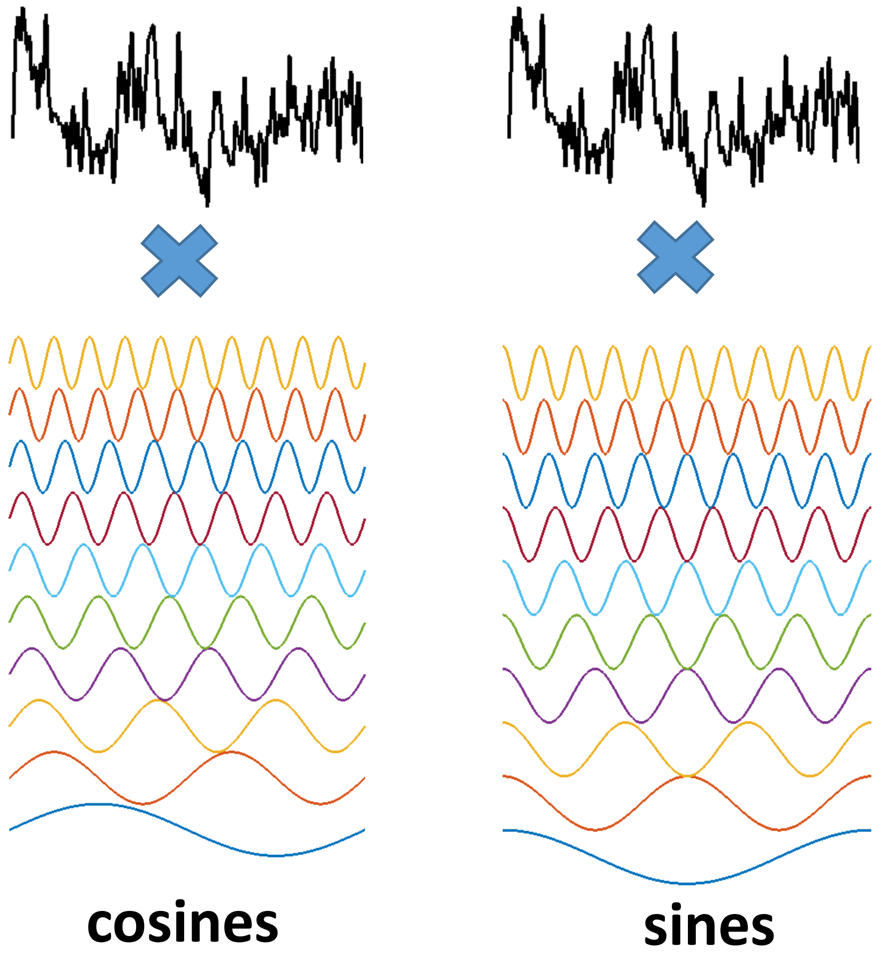
- linear operation
- can think of it as multiplying with sines and cosines ($= \mathcal{F}$, the Fourier basis)

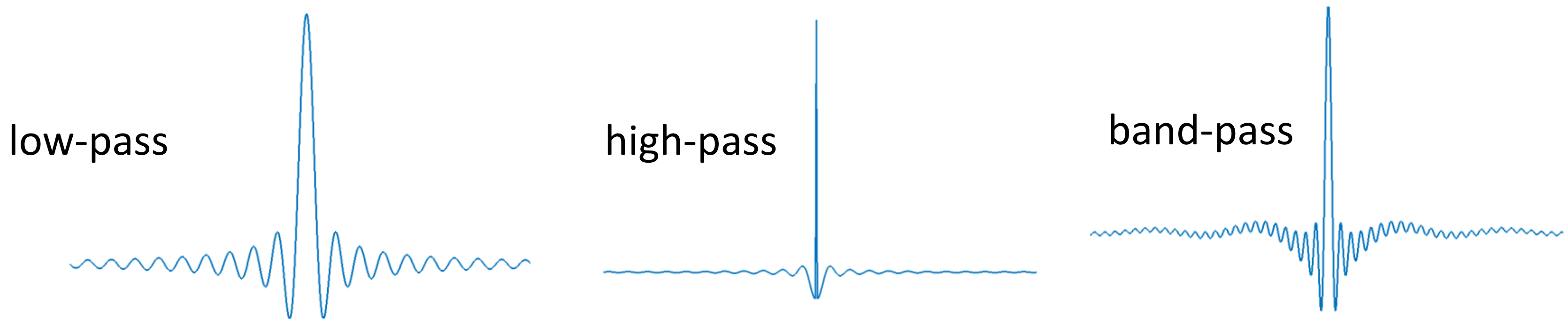
$$\boldsymbol{s} = \mathcal{F}(\mathcal{F}^T \boldsymbol{s})$$

- low-, high-, band- pass filtering, i.e.

$$s_{\text{filtered}} = \mathcal{F}_{10-100\text{Hz}}\left(\mathcal{F}^T_{10-100\text{Hz}}\,\boldsymbol{s}\right)$$

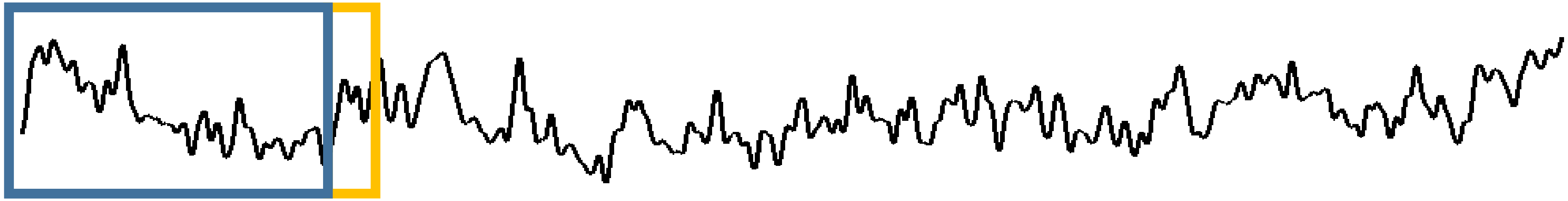
- the linear filter is $\mathcal{F}_{10-100\text{Hz}}\mathcal{F}^T_{10-100Hz}$





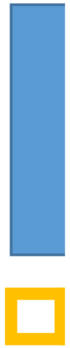
Linear models. Regression.

- best linear predictor from time-lagged, causal or non-causal data (Wiener filter)



z_t

x_t



$$x_t^{new} \sim A\, z_t^{new}$$
$$A = \boldsymbol{x}\boldsymbol{z}^T\, (\boldsymbol{z}\boldsymbol{z}^T)^{-1}$$

Linear models. Regression.

- regression covariance $z_t z_t^T$ is a Toeplitz matrix
  - eigenvectors of a Toeplitz matrix are always Fourier components, i.e. sines and cosines

$\boldsymbol{zz^T} = \mathcal{F} P^2 \mathcal{F}^T$, P has the Fourier coefficients

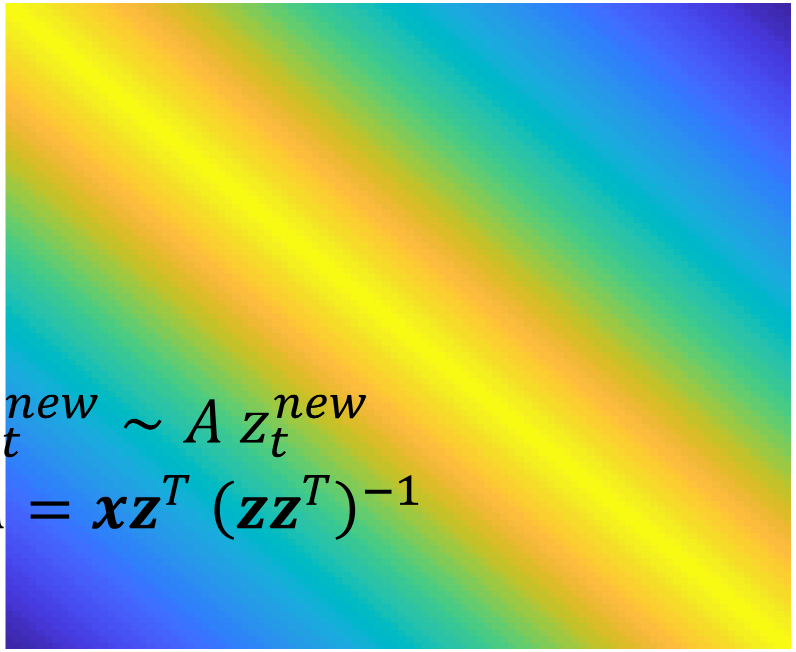
$$A = (\boldsymbol{xz}^T)\,(\mathcal{F} P^2 \mathcal{F}^T)^{-1}$$

$$A = \boldsymbol{x}(\boldsymbol{z}^T \mathcal{F}) P^{-2} \mathcal{F}^T$$

$$A = \boldsymbol{x} P \mathcal{F}^T$$

$$x_t^{new} = (\boldsymbol{x}P)(\mathcal{F}^T z_t^{new})$$

$\boldsymbol{zz}^T =$



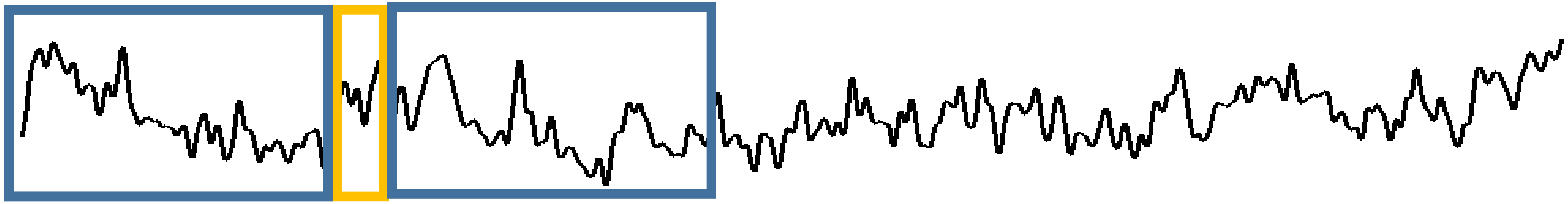
$$x_t^{new} \sim A\, z_t^{new}$$

$$A = \boldsymbol{xz}^T\,(\boldsymbol{zz}^T)^{-1}$$

it's like predicting each Fourier component separately

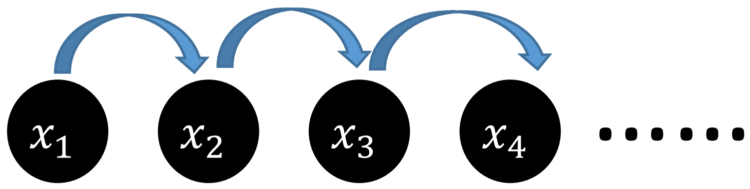
Linear models. Regression.

- Non-causal prediction:



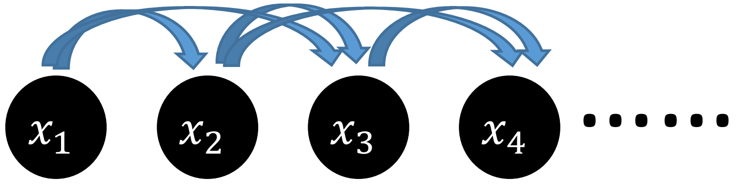
Autoregressive models (1D and ND)

$$x_t = A_0 + A_{-1}\, x_{t-1} + A_{-2} x_{t-2} + \ \ldots$$



AR(1) model

- depends on past 1,2, or n samples



AR(2) model

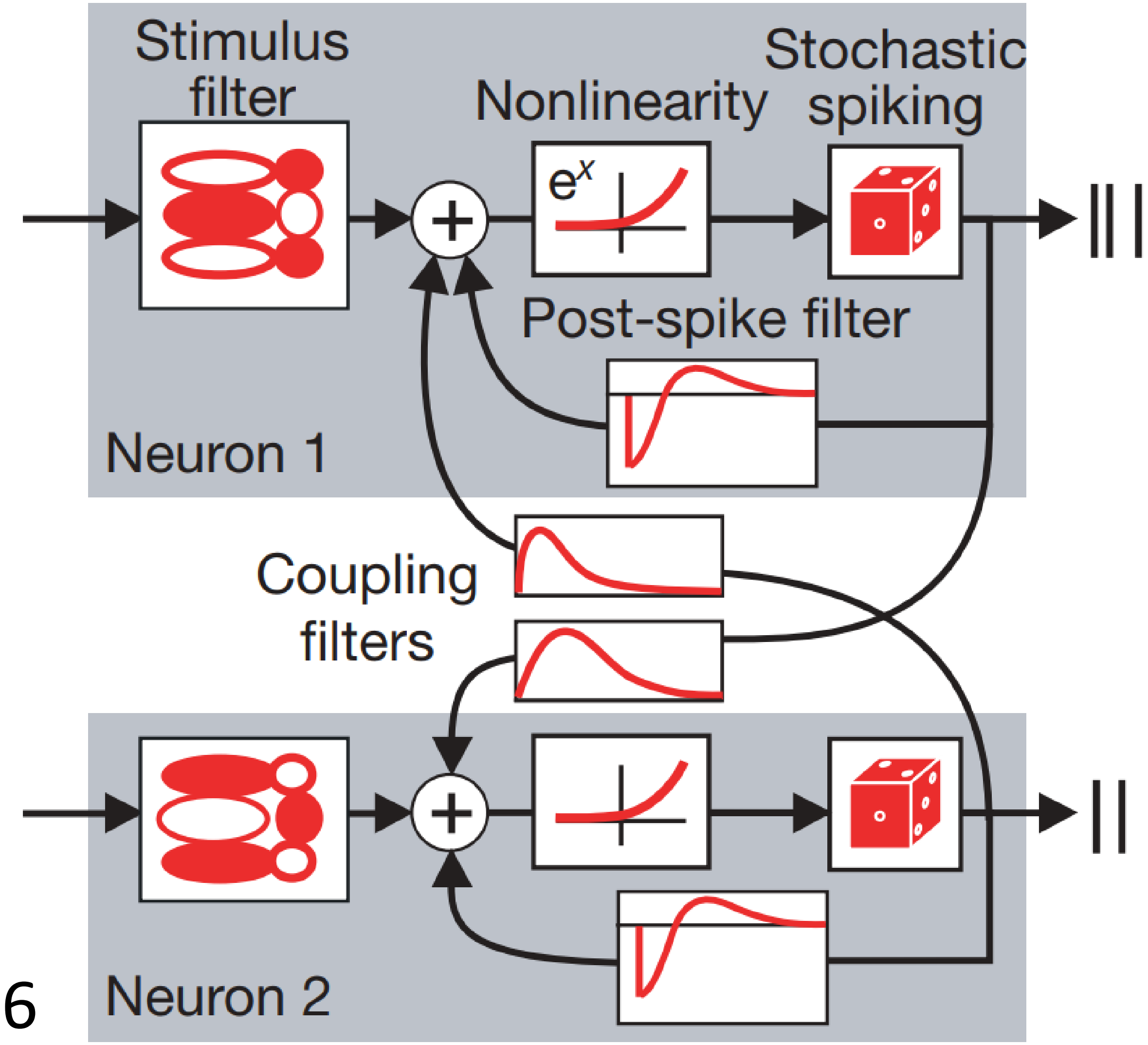
- just another way to regularize a linear prediction
- works for ND as well (just a bigger regression)

Autoregressive filter (1D and ND)

- add bells and whistles to model your favorite data
- Example: multineuron recordings

$$y_t = A_0 + A_{-1}\, x_{t-1} + A_{-2} x_{t-2} + \ \dots$$

$$x_t = \text{Poisson}\ (f(y_t))$$

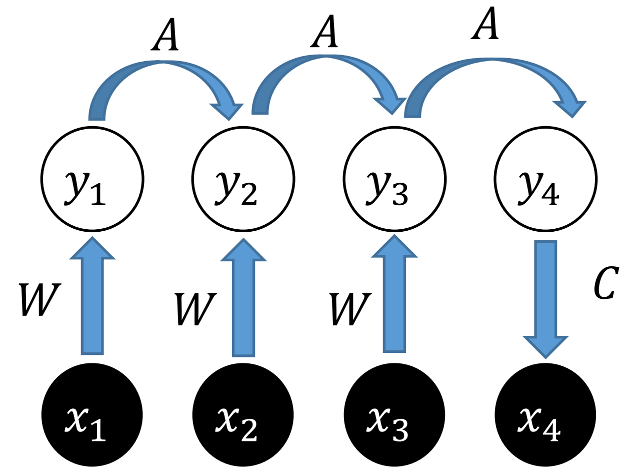


Pillow et al, 2006

Kalman filter

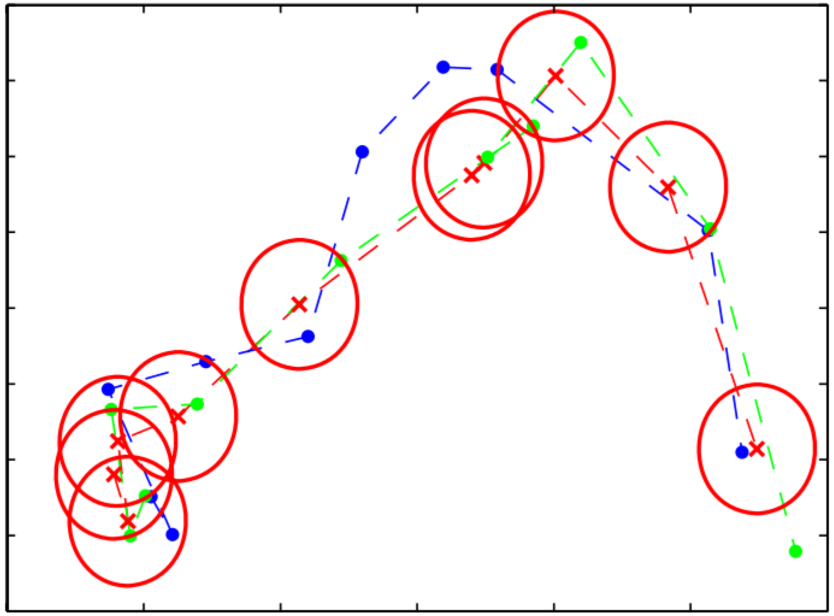
$$\hat{x}_t = C y_t$$
$$y_{t+1} = A\, y_t + W\,(x_t - \hat{x}_t)$$

- these are just the mean equations
- additional equations for confidence prediction



- just another way to regularize a linear regression
- IIR vs FIR: infinite impulse response vs finite response filter
- can capture rotational dynamics in high-D

historically used for tracking



Bishop book. 2006

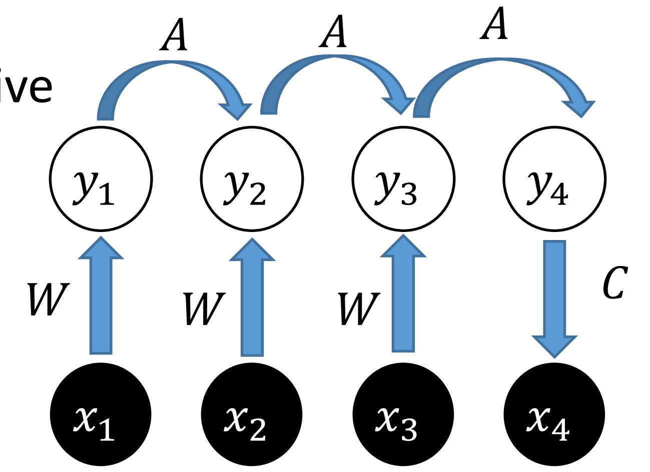
Dynamical systems

- the “generative model” for Kalman filters
  - allows to learn parameters from data
  - allows us to generate data from the model

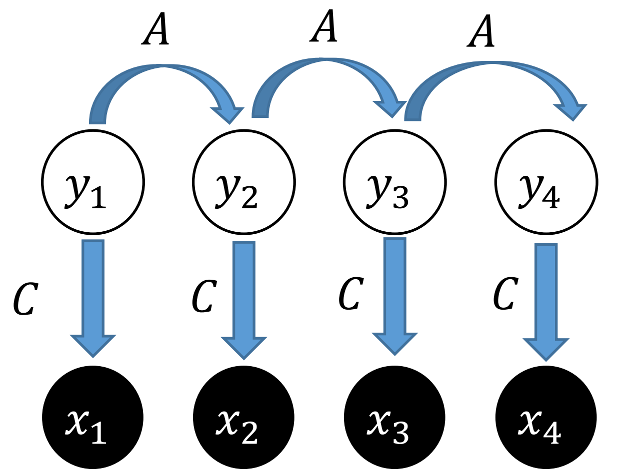
- linear dynamical systems have well understood dynamics:

$$y_{t+1} = A\, y_t + \epsilon_t$$

discriminative



generative



Overview of methods

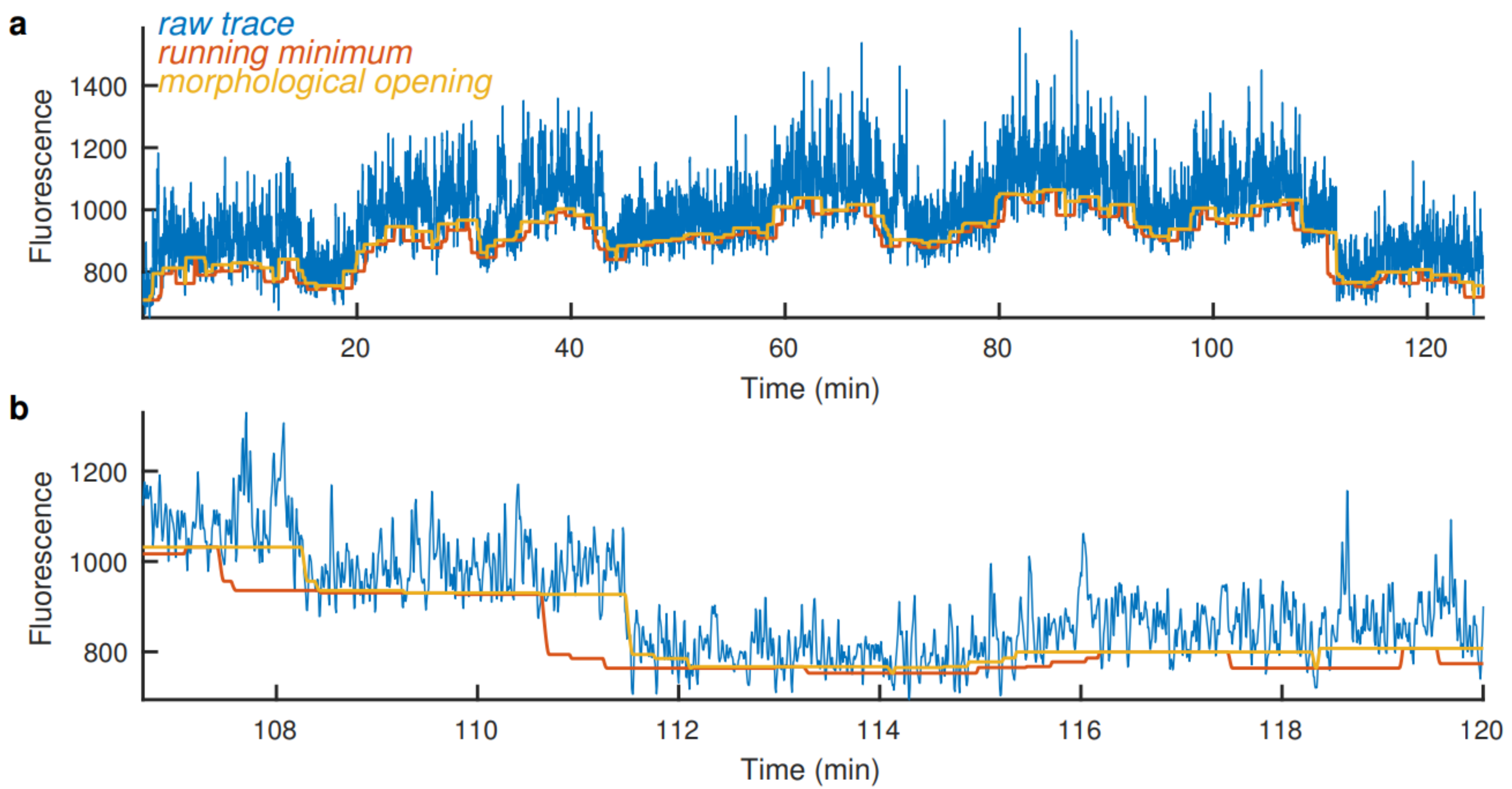
- Linear models (or “filtering”)
  - Gaussian process (or kriging)
  - Fourier-domain
  - Wiener (aka regression)
  - autoregressive filter
  - Kalman filter
- Nonlinear models
  - median filtering
  - deconvolution
  - Wavelets
  - HMM (discrete)
  - blackbox: RNN, seq-to-seq

Overview of methods

- Linear models (or "filtering")
  - Gaussian process (or kriging)
  - Fourier-domain
  - Wiener (aka regression)
  - autoregressive filter
  - Kalman filter
- Nonlinear models
  - median filtering
  - deconvolution
  - Wavelets
  - HMM (discrete)
  - blackbox: RNN, seq-to-seq

Median filtering

- some related filters: running percentile (minimum, maximum)
- “morphological opening”

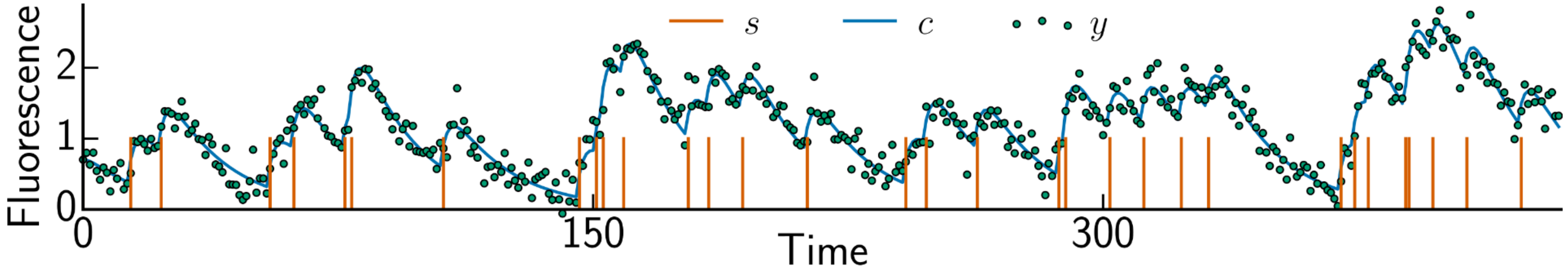


Constrained deconvolution

- the generative model is linear, but observations are noisy and there are constraints. For example:

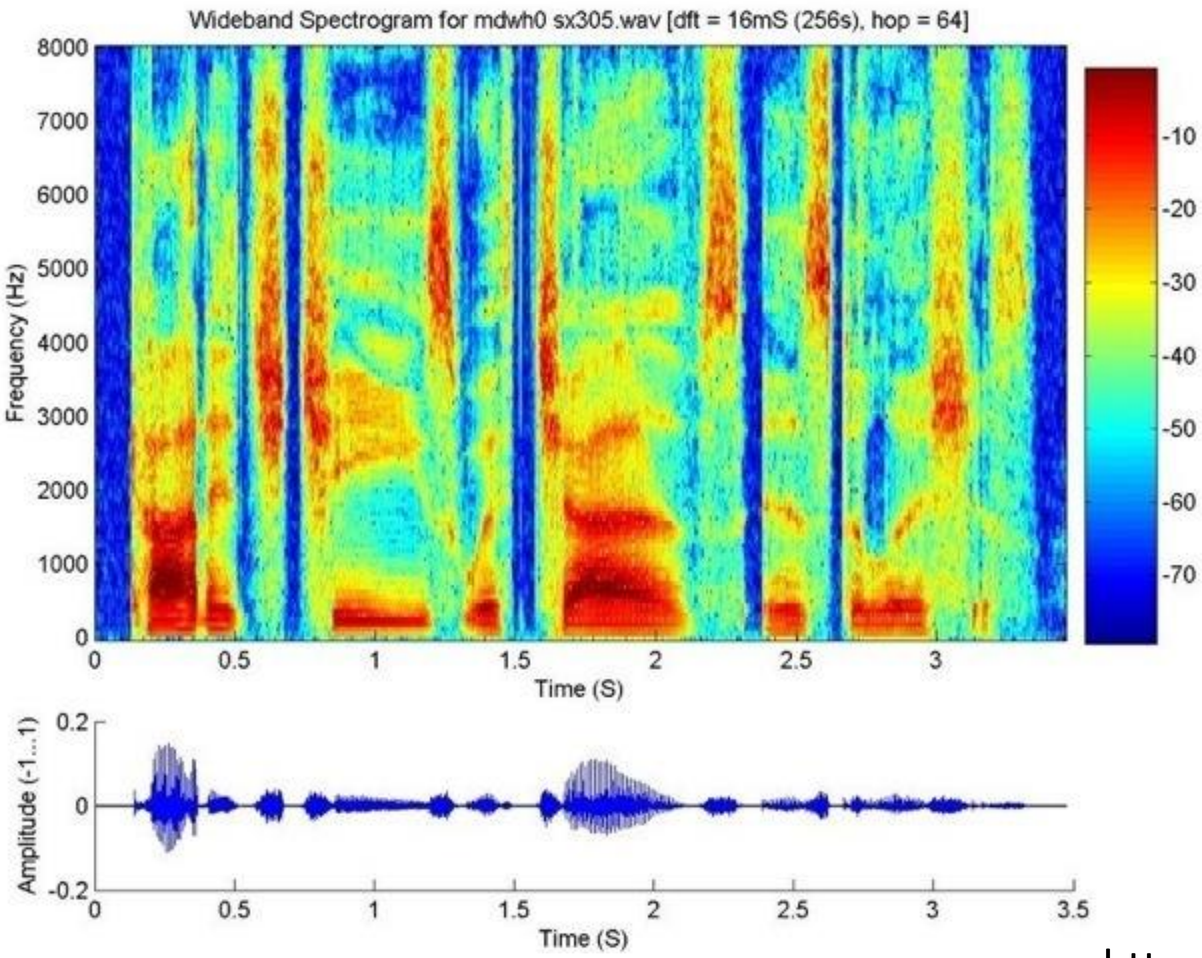
$$y_{t+1} = a\, y_t + z_t \text{ where } z_t > 0$$
$$x_t = y_t + \epsilon_t$$

- Surprisingly, this one can be solved easily with OASIS (Friederich et al, 2017)



- Other constraints on z_t : sparsity (L1), discreteness (L0)
  - L0 constraint can be approx. solved with “matching pursuit” or “wavelet decomposition”

Wavelets for coding human speech



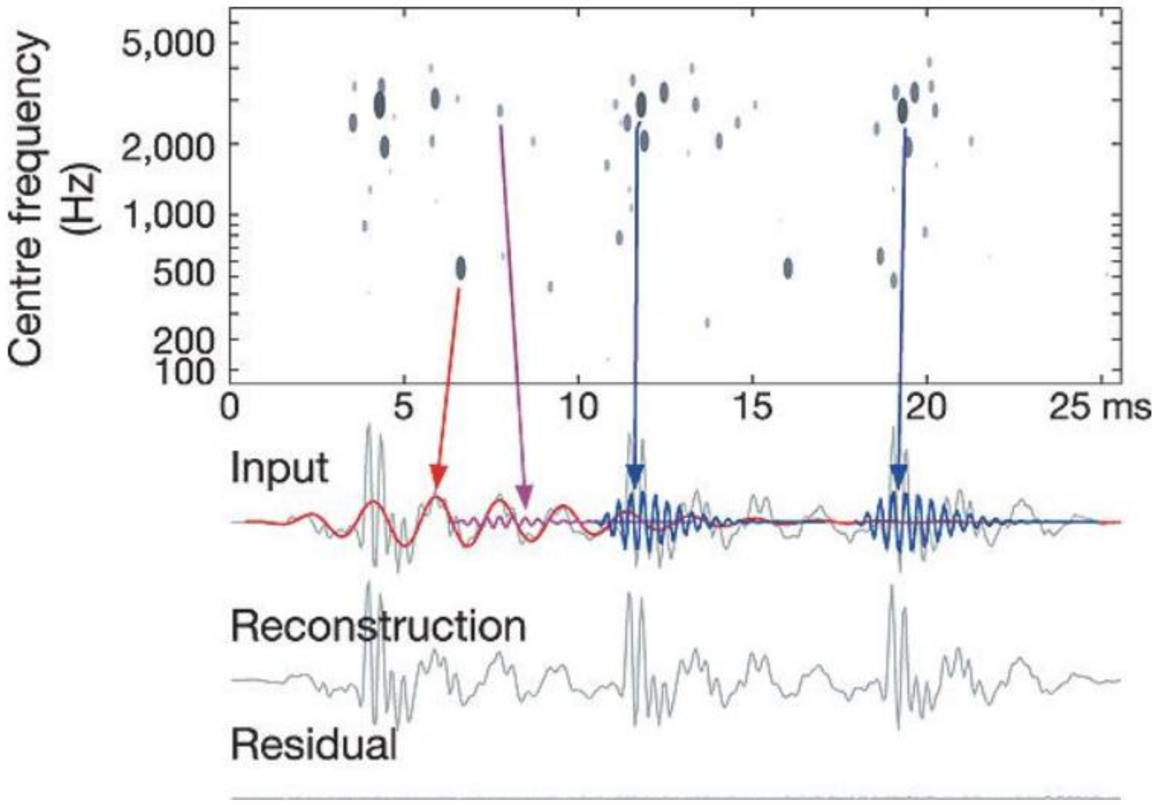
Cottage cheese with chives is delicious.

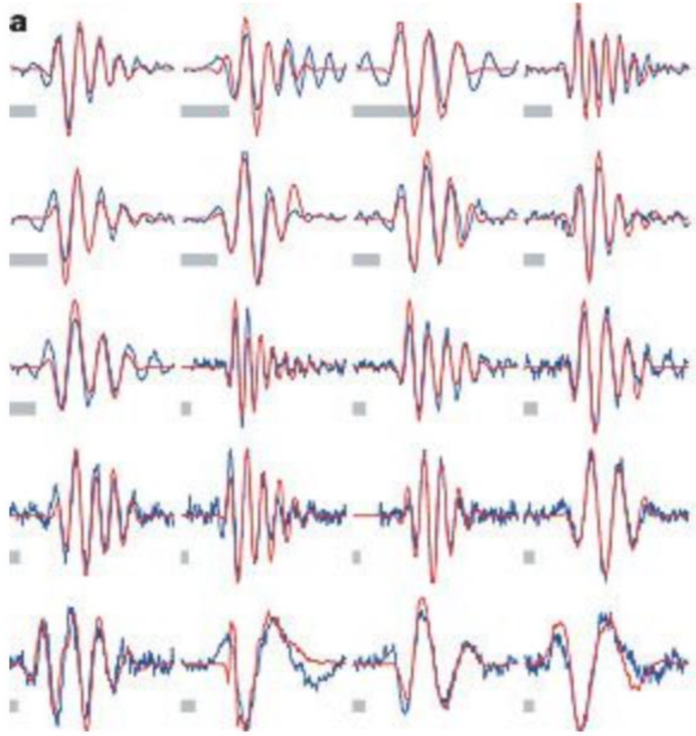
This looks “dense” but it can be encoded by a few overlapping “wavelets”.

http://www.columbia.edu/~djg2138/Dan_Gillespie_%40_Columbia/Assignments/Entries/2009/1/31_Assignment_1.html

Learn the wavelets from speech data

- Lewicki et al, 2006



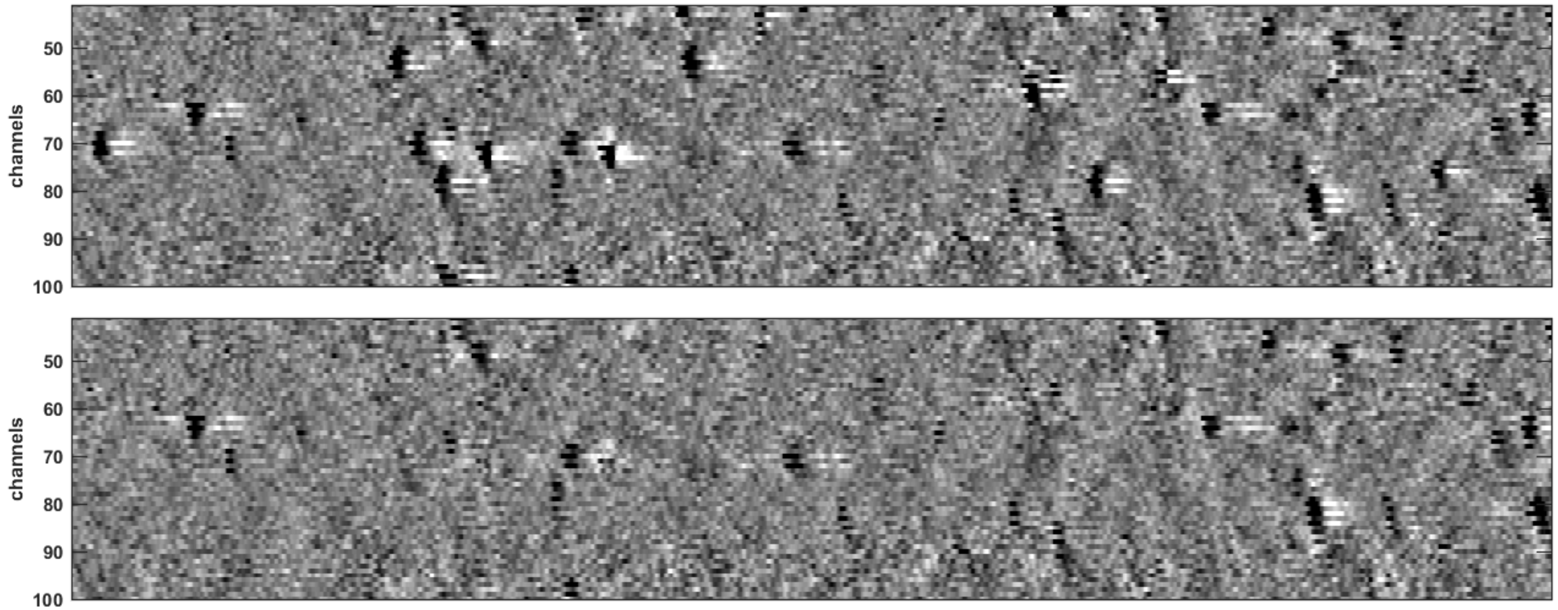


Wavelet decomposition

- Decompose a signal into discrete “packets” with matching pursuit (Mallat et al, 1993)

original

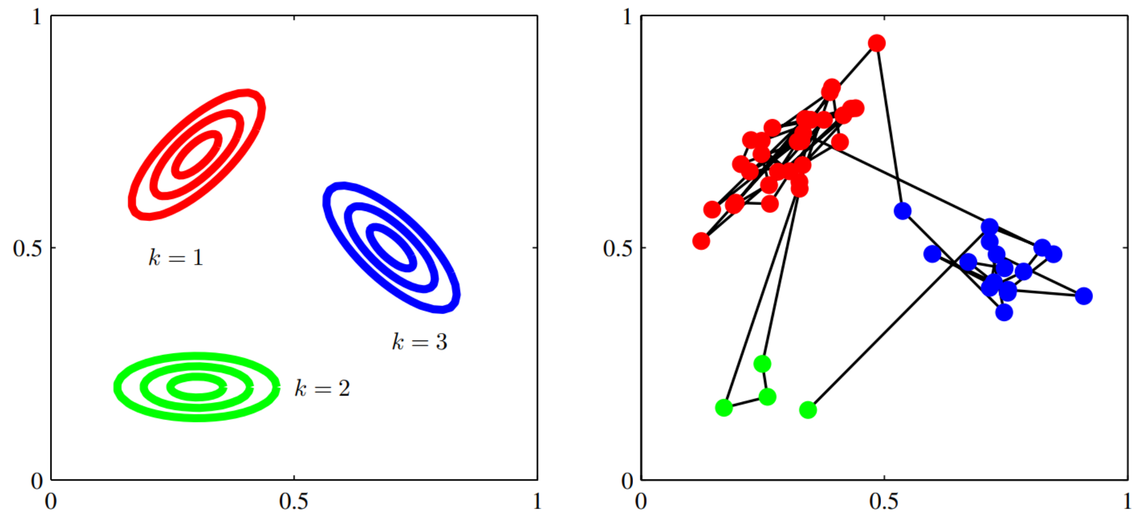
subtract off templates



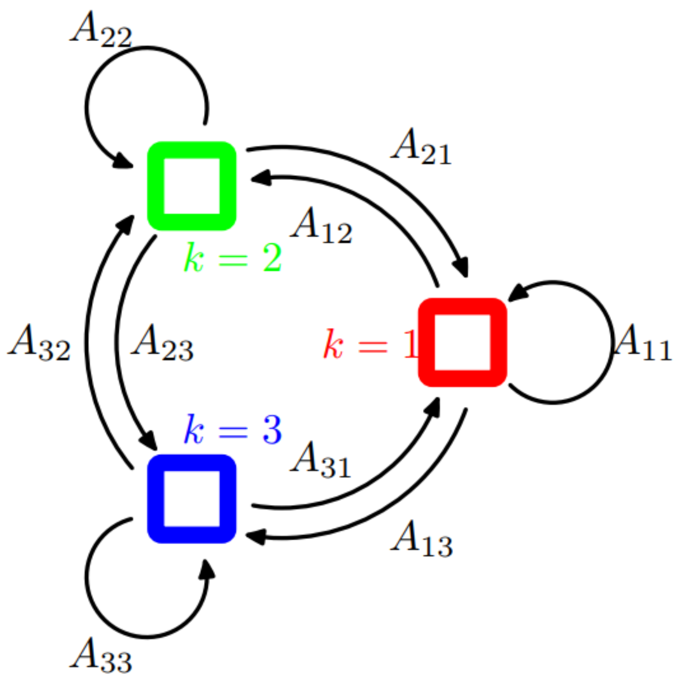
Hidden Markov Model (discrete)

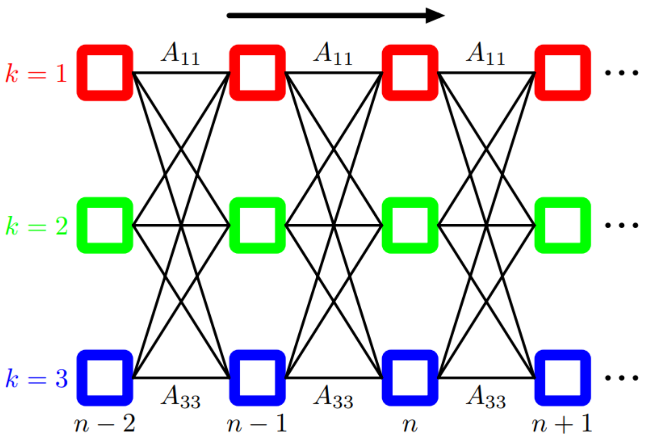
- N states, with transition probability matrix between them

- each state produces a different output + noise



- Surprisingly, inference algorithm is exact: dynamic programming / Viterbi



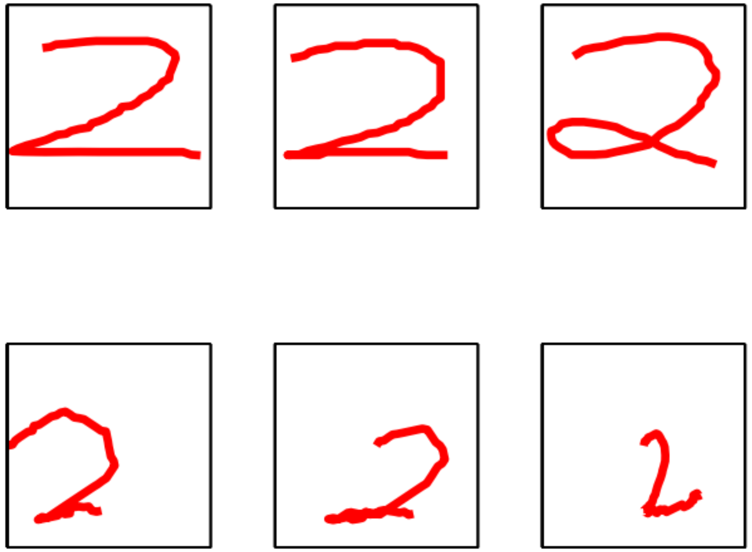


Bishop book, 2006

Hidden Markov Model (discrete)

Real digits

Sampled
from a model



Weaknesses

- discrete variables carry less information than continuous ones
- does not have distributed representations -> less ability to carry information

Most powerful when used in conjunction with other models
(see switching linear dynamical system)

Bishop book, 2006

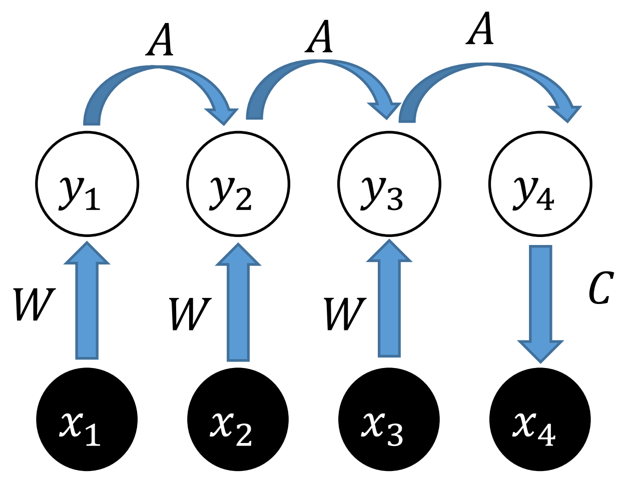
Black box prediction: recurrent neural networks

Kalman filter

$$\hat{x}_t \quad = Cy_t$$
$$y_{t+1} = fA\, y_t + W\, (x_t - \hat{x}_t)$$

Recurrent neural networks

$$\hat{x}_t \quad = Cy_t$$
$$y_{t+1} = f(A\, y_t + W\, x_t)$$



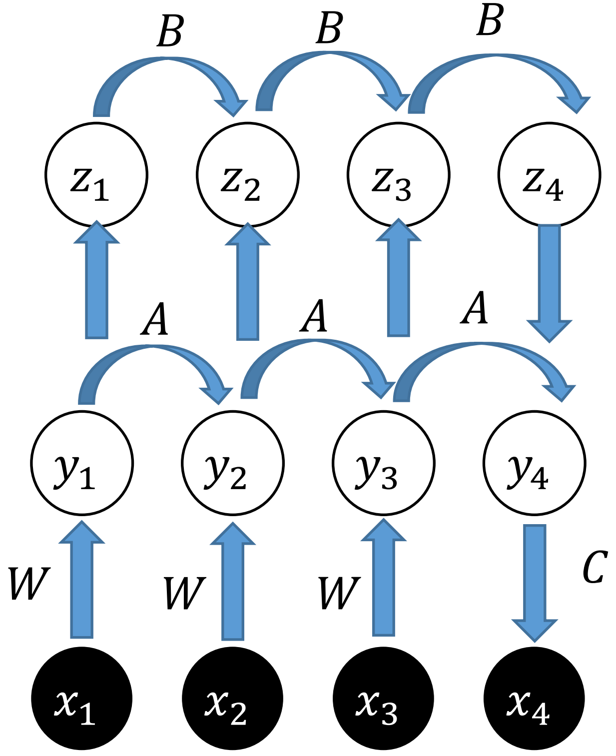
Black box prediction: recurrent neural networks

Kalman filter

$$\hat{x}_t \quad = C y_t$$
$$y_{t+1} = f A\, y_t + W\,(x_t - \hat{x}_t)$$

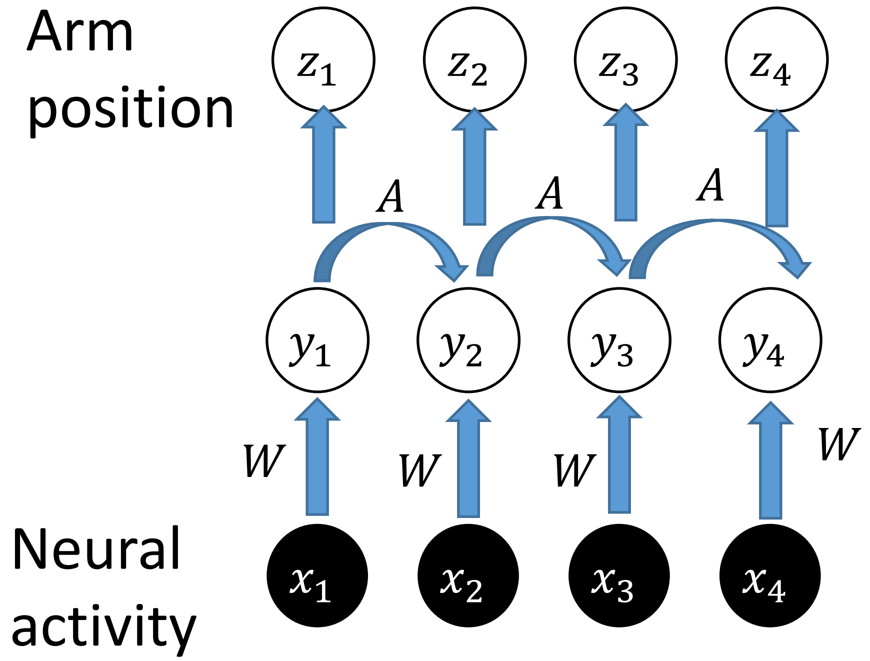
Recurrent neural networks

$$\hat{x}_t \quad = C y_t$$
$$y_{t+1} = f(A\, y_t + W\, x_t)$$



Seq to seq prediction

- Language translation
- Brain-machine interface



Tips & tricks

- non-stationarity, i.e. changes in statistics
  - between training and testing data
  - solution: interleave train and test blocks, preprocess
- real-world data often has 1/f spectrum
- separation of timescales
  - easy to predict from slow timescales, but that may be uninteresting
  - a slow timescale may look like a non-stationarity

Conclusions

- most timeseries models are linear filters
- some linear filters give estimates of confidence, which can be useful
- nonlinear models can capture interesting spatio-temporal patterns
  - best to use a dedicated framework for this: pytorch, tensorflow etc