

# Analysis of Time Series

## Chapter 10: Introduction to partially observed Markov process models

Edward L. Ionides

# Outline

- 1 Stochastic dynamic systems observed with noise
  - Latent process models
  - The Markov property
  - The measurement model
- 2 Prediction, filtering, smoothing and likelihood
  - Prediction and filtering recursions
  - Calculating the likelihood
  - Smoothing
- 3 Linear Gaussian POMP models
  - ARMA models as LG-POMP models
  - The basic structural model
  - Spline smoothing represented as an LG-POMP
  - The Kalman filter

# Latent process models

- Uncertainty and variability are common features biological and social systems. Complex physical systems can also be unpredictable: we can only forecast weather reliably in the near future.
- Time series models of deterministic trend plus colored noise imply perfect predictability if the trend function enables extrapolation.
- To model variability and unpredictability in a dynamic system, we can specify a stochastic (i.e., random) model for the system.
- Often times, the full dynamic system is unobserved. We have only noisy or incomplete measurements.
- We model measurements as random variables conditional on the trajectory of the **latent process**. The latent process is also called a **state process** or **hidden process**.

# The Markov property

- A model for a stochastic dynamic system has the **Markov property** if the future evolution of the system depends only on the current state, plus randomness introduced in future.
- A models with the Markov property may be called a **Markov chain** of a **Markov process**.
- We use the term Markov process since the term chain is often reserved for situations where either time or the latent state (or both) take discrete values.
- The Markov property is often used to model the latent process in a time series model.

## Notation for discrete time Markov processes

- A time series model  $X_{0:N}$  is a **Markov process** model if the conditional densities satisfy the **Markov property** [P1] that

$$[\text{P1}] \quad f_{X_n|X_{1:n-1}}(x_n | x_{1:n-1}) = f_{X_n|X_{n-1}}(x_n | x_{n-1}).$$

for all  $n \in 1 : N$

- We may suppose there is an underlying continuous time,  $t$ , such that  $X_n$  occurs at time  $t_n$ .
- We write  $X(t)$  for the continuous time model, setting  $X_n = X(t_n)$ .
- $t_{1:N}$  are **measurement times**
- $t_0$  is the **initialization time**

## Initial conditions

- We **initialize** the Markov process model at a time  $t_0$ , although data are collected only at times  $t_{1:N}$ .
- The initialization model could be deterministic (a fixed value) or a random variable.
- We model  $X_0 = X(t_0)$  as a draw from a probability density function

$$f_{X_0}(x_0). \quad (1)$$

- A fixed initial value is a special case of a density corresponding to a point mass with probability one at the fixed value.
- A discrete probability mass function is a special case of a density corresponding to a collection of point masses.

# The process model

- The probability density function  $f_{X_n|X_{n-1}}(x_n | x_{n-1})$  is called the **one-step transition density** of the Markov process.
- The Markov property asserts that the next step taken by a Markov process follows the one-step transition density based on the current state, whatever the previous history of the process.
- For a Markov model, the full joint distribution of the latent process is entirely specified by the one-step transition densities, given the initial value.
- Therefore, we also call  $f_{X_n|X_{n-1}}(x_n | x_{n-1})$  the **process model**.

# The joint distribution in terms of one-step transition densities

**Exercise 10.1.** Use [P1] to derive an expression for the joint distribution of a Markov process as a product of the one-step transition densities. In other words, derive

$$[\text{P2}] \quad f_{X_{0:N}}(x_{0:N}) = f_{X_0}(x_0) \prod_{n=1}^N f_{X_n|X_{n-1}}(x_n | x_{n-1}).$$

**Hint:** This involves elementary rules for manipulation of joint and conditional densities, together with application of the Markov property. It is a good exercise to work through by hand to build familiarity with the model class.



**Question 10.1.** Explain why a causal Gaussian AR(1) process is a Markov process. Gaussian AR(1):  $Y_n = \phi Y_{n-1} + \varepsilon_n$ ,  $\varepsilon_n \sim \text{iid normal}(0, \sigma^2)$ .

Here,  $\varepsilon_n$  is independent of  $Y_{1:n-1}$  by construction, because  $Y_{1:n-1}$  is a function of  $\varepsilon_{0:n-1}$ . Thus,  $Y_n$  given  $Y_{1:n-1}$  has a distribution depending only on  $Y_{n-1}$ , not  $Y_{0:n-2}$ , i.e.,  $\{Y_n\}$  is Markov.

Note for a general white noise process,  $\{\varepsilon_n\}$  are uncorrelated but not independent, so  $\{Y_n\}$  is not necessarily Markov.

## Time-homogeneous transitions and stationarity

- The one step transition density  $f_{X_n|X_{n-1}}$  for a Markov process  $X_{0:N}$  can depend on  $n$ .
- $X_{0:N}$  is **time-homogeneous** if  $f_{X_n|X_{n-1}}$  does not depend on  $n$ , so there is a conditional density  $f(\cdot|\cdot)$  such that, for all  $n \in 1:N$ ,

$$f_{X_n|X_{n-1}}(x_n | x_{n-1}) = f(x_n | x_{n-1}). \quad (2)$$

**Question 10.2.** If  $X_{0:N}$  is strict stationary, it is time-homogeneous. Why?

*strict stationarity implies all joint & conditional densities are shift invariant, so*

$$f_{X_n|X_{n-1}}(x_n | x_{n-1}) = f_{X_1|X_0}(x_n | x_{n-1}) \text{ for all } n.$$

**Question 10.3.** Time-homogeneity does not necessarily imply stationarity.

Find a counter-example. *Consider a random walk, with or without drift:  $X_n = X_{n-1} + \mu + \varepsilon_n$ ,  $\varepsilon_n \sim \text{iid normal}(0, \sigma^2)$ . This is non-stationary, but has time-homogeneous transitions.*

## Partially observed Markov process (POMP) models

- **Partial observation** may mean either or both of (i) measurement noise; (ii) entirely unmeasured latent variables.
- These features are present in many systems.
- A **partially observed Markov process (POMP)** model is defined by putting together a Markov latent process model and a **measurement model**.
- POMP models are a general class, covering many models designed for specific applications.
- Statistical methods for to this general class give us flexibility to develop specific POMP models appropriate to a range of applications.

# The measurement model

- The **measurement process** is a collection of random variables  $Y_{1:N}$  which models the data  $y_{1:N}^*$ .
- $Y_n$  is assumed to depend on the latent process only through its value  $X_n$  at the time of the measurement. Formally, this assumption is:

$$[\text{P3}] \quad f_{Y_n|X_{0:N}, Y_{1:n-1}, Y_{n+1:N}}(y_n | x_{0:N}, y_{1:n-1}, y_{n+1:N}) = f_{Y_n|X_n}(y_n | x_n).$$

- We call  $f_{Y_n|X_n}(y_n | x_n)$  the **measurement model**.

# Time-homogeneous measurement models

- In general, the measurement model can depend on  $n$  or on any covariate time series.
- The measurement model is **time-homogeneous** if there is a conditional probability density function  $g(\cdot | \cdot)$  such that, for all  $n \in 1 : N$ ,

$$f_{Y_n|X_n}(y_n | x_n) = g(y_n | x_n). \quad (3)$$

- Time-inhomogeneous process and measurement models are sufficiently common that we benefit from the extra generality of writing  $f_{X_n|X_{n-1}}(x_n|x_{n-1})$  and  $f_{Y_n|X_n}(y_n|x_n)$  versus  $f(x_n|x_{n-1})$  and  $g(y_n|x_n)$ .

## Four basic calculations for working with POMP models

Many time series models in science, engineering and industry can be written as POMP models. A reason that POMP models form a useful tool for statistical work is that there are convenient recursive formulas to carry out four basic calculations:

- 1 Prediction
- 2 Filtering
- 3 Smoothing
- 4 Likelihood calculation

# Prediction

- **One-step prediction** (also called forecasting) of the latent process at time  $t_{n+1}$  given data up to time  $t_n$  involves finding

$$f_{X_{n+1}|Y_{1:n}}(x_{n+1} | y_{1:n}^*). \quad (4)$$

- We may want to predict more than one time step ahead. However, one-step prediction turns out to be closely related to computing the likelihood function, and therefore central to statistical inference.
- Our prediction is a conditional probability density, not a point estimate. In the context of forecasting, this is called a **probabilistic forecast**. What are the advantages of a probabilistic forecast over a point forecast? Are there any disadvantages?

*Advantage: see the full range of possibilities, weighted by their probability.*

*Disadvantage: it may be harder for some people to understand.*

# Filtering

- The **filtering** calculation at time  $t_n$  is to find the conditional distribution of the latent process  $X_n$  given data  $y_{1:n}^*$  available at time  $t_n$ .
- Filtering involves calculating

$$f_{X_n|Y_{1:n}}(x_n | y_{1:n}^*). \quad (5)$$

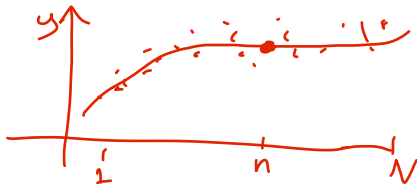
- This can be evaluated numerically or algebraically. We will see that Monte Carlo methods can be a good tool.
- The name “filtering” comes from the history of signal processing. A noisy received signal was filtered through capacitors and resistors to estimate the source signal.



# Smoothing

- In the context of a POMP model, smoothing involves finding the conditional distribution of  $X_n$  given all the data,  $y_{1:N}^*$ .
- So, the smoothing calculation is to find

$$f_{X_n|Y_{1:N}}(x_n | y_{1:N}^*). \quad (6)$$



# The likelihood

- The likelihood is the joint density of  $Y_{1:N}$  evaluated at the data,

$$f_{Y_{1:N}}(y_{1:N}^*). \quad (7)$$

- The model may depend on a parameter vector  $\theta$ . We can include  $\theta$  in all the joint and conditional densities above. Then, the **likelihood function** is the likelihood viewed as a function of  $\theta$ . We write

$$\mathcal{L}(\theta) = f_{Y_{1:N}}(y_{1:N}^*; \theta) \quad (8)$$

- If we can compute  $\mathcal{L}(\theta)$  then we can perform numerical optimization to get a maximum likelihood estimate
- Likelihood evaluation and maximization lets us compute profile likelihood confidence intervals, carry out likelihood ratio tests, and make AIC model comparisons.

## The prediction formula

- One-step prediction of the latent process at time  $t_n$  given data up to time  $t_{n-1}$  can be computed recursively in terms of the filtering problem at time  $t_{n-1}$ , via the **prediction formula** for  $n \in 1 : N$ ,

$$\begin{aligned}
 \text{[P4]} \quad f_{X_n|Y_{1:n-1}}(x_n | y_{1:n-1}^*) = \\
 \int f_{X_{n-1}|Y_{1:n-1}}(x_{n-1} | y_{1:n-1}^*) f_{X_n|X_{n-1}}(x_n | x_{n-1}) dx_{n-1}.
 \end{aligned}$$

- For the case  $n = 1$ , we let  $1 : k$  is the empty set when  $k = 0$ , so that  $f_{X_0|Y_{1:0}}(x_0 | y_{1:0}^*)$  means  $f_{X_0}(x_0)$ . In other words, the filter distribution at time  $t_0$  is the initial density for the latent process, since at time  $t_0$  we have no data to condition on.

**Exercise 10.2.** Derive [P4] using the definition of a POMP model with elementary properties of joint and conditional densities.

## Hints for deriving the recursion formulas

Any general identity holding for densities must also hold when we condition everything on a new variable.

**Example 1.** From

$$f_{XY}(x, y) = f_X(x) f_{Y|X}(y | x) \quad (9)$$

we can condition on  $Z$  to obtain

$$f_{XY|Z}(x, y | z) = f_{X|Z}(x | z) f_{Y|XZ}(y | x, z). \quad (10)$$

**Example 2.** The prediction formula is a special case of the identity

$$f_{X|Y}(x | y) = \int f_{XZ|Y}(x, z | y) dz. \quad (11)$$

**Example 3.** A conditional form of Bayes' identity is

$$f_{X|YZ}(x | y, z) = \frac{f_{Y|XZ}(y | x, z) f_{X|Z}(x | z)}{f_{Y|Z}(y | z)}. \quad (12)$$

# The filtering formula

- Filtering at time  $t_n$  can be computed by combining the new information in the datapoint  $y_n^*$  with the calculation of the one-step prediction of the latent process at time  $t_n$  given data up to time  $t_{n-1}$ .
- This is carried out via the **filtering formula** for  $n \in 1 : N$ ,

$$[\text{P5}] \quad f_{X_n|Y_{1:n}}(x_n | y_{1:n}^*) = \frac{f_{X_n|Y_{1:n-1}}(x_n | y_{1:n-1}^*) f_{Y_n|X_n}(y_n^* | x_n)}{f_{Y_n|Y_{1:n-1}}(y_n^* | y_{1:n-1}^*)}.$$

**Exercise 10.3.** Derive [P5] using the definition of a POMP model with elementary properties of joint and conditional densities.

- The prediction and filtering formulas are **recursive**. If they can be computed for time  $t_n$  then they enable the computation at time  $t_{n+1}$ .

## The conditional likelihood formula

- The denominator in the filtering formula [P5] is the **conditional likelihood** of  $y_n^*$  given  $y_{1:n-1}^*$ .
- It can be computed in terms of the one-step prediction density, via the **conditional likelihood formula**,

$$[P6] \quad f_{Y_n|Y_{1:n-1}}(y_n^* | y_{1:n-1}^*) = \int f_{X_n|Y_{1:n-1}}(x_n | y_{1:n-1}^*) f_{Y_n|X_n}(y_n^* | x_n) dx_n.$$

- To make this formula work for  $n = 1$ , we take advantage of the convention that  $1 : k$  is the empty set when  $k = 0$ .

## Computation of the likelihood and log likelihood

- The likelihood of the entire dataset,  $y_{1:N}^*$  can be found from [P6], using the identity

$$f_{Y_{1:N}}(y_{1:N}^*) = \prod_{n=1}^N f_{Y_n|Y_{1:n-1}}(y_n^* | y_{1:n-1}^*). \quad (13)$$

- Equation (13) uses the convention that  $1:k$  is the empty set when  $k=0$ , so the first term in the product is

$$f_{Y_1|Y_{1:0}}(y_1^* | y_{1:0}^*) = f_{Y_1}(y_1^*) \quad (14)$$

- If our model has an unknown parameter  $\theta$  then (13) gives the **log likelihood function** as a sum of conditional log likelihoods,

$$\ell(\theta) = \log \mathcal{L}(\theta) = \log f_{Y_{1:N}}(y_{1:N}^*; \theta) = \sum_{n=1}^N \log f_{Y_n|Y_{1:n-1}}(y_n^* | y_{1:n-1}^*; \theta).$$

# The smoothing recursions

- Smoothing is less fundamental for likelihood-based inference than filtering and one-step prediction.
- Nevertheless, sometimes we want to compute the smoothing density, so we develop some necessary formulas.
- The filtering and prediction formulas are recursions forward in time: a solution at time  $t_{n-1}$  is used for the computation at  $t_n$ .
- For smoothing, we have **backwards smoothing recursion formulas**,

$$[\text{P7}] \quad f_{Y_{n:N}|X_n}(y_{n:N}^* | x_n) = f_{Y_n|X_n}(y_n^* | x_n) f_{Y_{n+1:N}|X_n}(y_{n+1:N}^* | x_n).$$

$$[\text{P8}] \quad f_{Y_{n+1:N}|X_n}(y_{n+1:N}^* | x_n) \\ = \int f_{Y_{n+1:N}|X_{n+1}}(y_{n+1:N}^* | x_{n+1}) f_{X_{n+1}|X_n}(x_{n+1} | x_n) dx_{n+1}.$$



# Combining recursions to find the smoothing distribution

The forwards and backwards recursion formulas together allow us to compute the **smoothing formula**,

$$[\text{P9}] \quad f_{X_n|Y_{1:N}}(x_n | y_{1:N}^*) = \frac{f_{X_n|Y_{1:n-1}}(x_n | y_{1:n-1}^*) f_{Y_{n:N}|X_n}(y_{n:N}^* | x_n)}{f_{Y_{n:N}|Y_{1:n-1}}(y_{n:N}^* | y_{1:n-1}^*)}.$$

**Exercise 10.4.** Show how [P7], [P8] and [P9] follow from the basic properties of conditional densities combined with the Markov property.

**Hint:** you can write the left hand side of [P9] as  $f_{X|YZ}$  with  $X = X_n$ ,  $Y = Y_{1:n-1}$ ,  $Z = Y_{n:N}$ .

# Linear Gaussian POMP (LG-POMP) models

- Linear Gaussian partially observed Markov process (LG-POMP) models have many applications across science and engineering.
- Gaussian ARMA models are LG-POMP models. The POMP recursion formulas give a computationally efficient way to obtain the likelihood of a Gaussian ARMA model.
- Smoothing splines (including the Hodrick-Prescott filter, which is a smoothing spline) can be written as an LG-POMP model.
- The **Basic Structural Model** is an LG-POMP used for econometric forecasting. It models a stochastic trend, seasonality, and measurement error, in a framework with econometrically interpretable parameters. This is more interpretable than fitting SARIMA.
- If an LG-POMP model is appropriate, you avoid Monte Carlo computations used for inference in general nonlinear POMP models.

## The general LG-POMP model

Suppose the latent process,  $X_{0:N}$ , and the observation process  $\{Y_n\}$ , takes vector values with dimension  $d_X$  and  $d_Y$ . A general mean zero LG-POMP model is specified by

- A sequence of  $d_X \times d_X$  matrices,  $\mathbb{A}_{1:N}$ ,
- A sequence of  $d_X \times d_X$  covariance matrices,  $\mathbb{U}_{0:N}$ ,
- A sequence of  $d_Y \times d_X$  matrices,  $\mathbb{B}_{1:N}$
- A sequence of  $d_Y \times d_Y$  covariance matrices,  $\mathbb{V}_{1:N}$ .

We initialize with  $X_0 \sim N[0, \mathbb{U}_0]$  and then define the entire LG-POMP model by a recursion for  $n \in 1 : N$ ,

$$\text{[LG1]} \quad X_n = \mathbb{A}_n X_{n-1} + \epsilon_n, \quad \epsilon_n \sim N[0, \mathbb{U}_n],$$

$$\text{[LG2]} \quad Y_n = \mathbb{B}_n X_n + \eta_n, \quad \eta_n \sim N[0, \mathbb{V}_n].$$

Often, but not always, we will have a **time-homogeneous** LG-POMP model, with  $\mathbb{A}_n = \mathbb{A}$ ,  $\mathbb{B}_n = \mathbb{B}$ ,  $\mathbb{U}_n = \mathbb{U}$  and  $\mathbb{V}_n = \mathbb{V}$  for  $n \in 1 : N$ .

# The LG-POMP representation of a Gaussian ARMA

- Let  $\{Y_n\}$  be a Gaussian ARMA( $p, q$ ) model with noise process  $\omega_n \sim \text{normal}[0, \sigma^2]$ , defined by

$$Y_n = \sum_{j=1}^p \phi_j Y_{n-j} + \omega_n + \sum_{k=1}^q \psi_k \omega_{n-k}. \quad (15)$$

- We look for a time-homogeneous LG-POMP defined by [LG1] and [LG2] where  $Y_n$  is the first component of  $X_n$  with no measurement error.
- To do this, we define  $d_X = r = \max(p, q + 1)$  and

$$\mathbb{B} = (1, 0, 0, \dots, 0), \quad (16)$$

$$\mathbb{V} = 0. \quad (17)$$

- We require  $\mathbb{A}$  and  $\mathbb{U}$  such that  $Y_n$  satisfies equation (15).

We state a solution and see if it works out. Consider

$$X_n = \begin{pmatrix} Y_n \\ \phi_2 Y_{n-1} + \cdots + \phi_r Y_{n-r+1} + \psi_1 \omega_n + \cdots + \psi_{r-1} \omega_{n-r+2} \\ \phi_3 Y_{n-1} + \cdots + \phi_r Y_{n-r+1} + \psi_2 \omega_n + \cdots + \psi_{r-1} \omega_{n-r+3} \\ \vdots \\ \phi_r Y_{n-1} + \psi_{r-1} \omega_n \end{pmatrix}$$

We can check that the ARMA equation (15) matches the matrix equation

$$X_n = \mathbb{A} X_{n-1} + \begin{pmatrix} 1 \\ \psi_1 \\ \psi_2 \\ \vdots \\ \psi_{r-1} \end{pmatrix} \omega_n. \text{ where } \mathbb{A} = \begin{pmatrix} \phi_1 & 1 & 0 & \dots & 0 \\ \phi_2 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \phi_{r-1} & 0 & \dots & 0 & 1 \\ \phi_r & 0 & \dots & 0 & 0 \end{pmatrix}$$

This is a time-homogenous LG-POMP, with  $\mathbb{A}$ ,  $\mathbb{B}$  and  $\mathbb{V}$  as above and

$$\mathbb{U} = \sigma^2 (1, \psi_1, \psi_2, \dots, \psi_{r-1})^\top (1, \psi_1, \psi_2, \dots, \psi_{r-1}).$$

## Different POMP models can give the same model for $Y_{1:N}$

- There are other LG-POMP representations giving rise to the same ARMA model.
- When only one component of a latent process is observed, any model giving rise to the same observed component is indistinguishable from the data.
- Here, the LG-POMP model has order  $d_X^2 = r^2 = \max(p, q + 1)^2$  parameters. The ARMA model has order  $r$  parameters, so we expect many ways to parameterize the ARMA model as a special case of the much larger LG-POMP model.
- This unidentifiability can also arise for non-Gaussian POMP models, but it is easier to see in the Gaussian case.

# The basic structural model

- The **basic structural model** was developed for econometric analysis.
- It decomposes an observable process  $Y_{1:N}$  as the sum of a **level** ( $L_n$ ), a **trend** ( $T_n$ ) describing the rate of change of the level, and a monthly **seasonal component** ( $S_n$ ).
- The model supposes that the level, trend and seasonality are perturbed with Gaussian white noise at each time point,

$$[\text{BSM1}] \quad Y_n = L_n + S_n + \epsilon_n$$

$$[\text{BSM2}] \quad L_n = L_{n-1} + T_{n-1} + \xi_n$$

$$[\text{BSM3}] \quad T_n = T_{n-1} + \zeta_n$$

$$[\text{BSM4}] \quad S_n = -\sum_{k=1}^{11} S_{n-k} + \eta_n$$

where  $\epsilon_n \sim \text{normal}[0, \sigma_\epsilon^2]$ ,  $\xi_n \sim \text{normal}[0, \sigma_\xi^2]$ ,  $\zeta_n \sim \text{normal}[0, \sigma_\zeta^2]$ ,  
and  $\eta_n \sim \text{normal}[0, \sigma_\eta^2]$ .

## Two common special cases of the basic structural model

- The **local linear trend** model is the basic structural model without the seasonal component,  $\{S_n\}$
- The **local level model** is the basic structural model without either the seasonal component,  $\{S_n\}$ , or the trend component,  $\{T_n\}$ . The local level model is therefore a random walk observed with measurement error.



## Initial values for the basic structural model

- To complete the model, we need to specify initial values.
- We have an example of the common problem of failing to specify initial values: these are not explained in the documentation of the R implementation of the basic structural model, `StructTS`. We could go through the source code to find out what it does.
- Incidentally, `?StructTS` does give some advice which resonates with our experience earlier in the course that optimization for ARMA models is often imperfect.

“Optimization of structural models is a lot harder than many of the references admit. For example, the ‘AirPassengers’ data are considered in Brockwell & Davis (1996): their solution appears to be a local maximum, but nowhere near as good a fit as that produced by ‘StructTS’. It is quite common to find fits with one or more variances zero, and this can include  $\sigma_{\text{eps}}^2$ .”

# The basic structural model is an LG-POMP model

[BSM1–4] can be put in matrix form,

$$\begin{pmatrix} L_n \\ T_n \\ S_n \\ S_{n-1} \\ S_{n-2} \\ \vdots \\ S_{n-10} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & -1 & -1 & -1 & \dots & -1 \\ 0 & 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & & & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} L_{n-1} \\ T_{n-1} \\ S_{n-1} \\ S_{n-2} \\ S_{n-3} \\ \vdots \\ S_{n-11} \end{pmatrix} + \begin{pmatrix} \xi_n \\ \zeta_n \\ \eta_n \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Now, set

$$X_n = (L_n, T_n, S_n, S_{n-1}, S_{n-2}, \dots, S_{n-10})^T, \quad (18)$$

$$Y_n = (1, 0, 1, 0, 0, \dots, 0)X_n + \epsilon_n. \quad (19)$$

We can identify matrices  $\mathbb{A}$ ,  $\mathbb{B}$ ,  $\mathbb{U}$  and  $\mathbb{V}$  giving a time-homogeneous LG-POMP representation [LG1, LG2] for the basic structural model.

# Spline smoothing and its LG-POMP representation

- Spline smoothing is a standard method to smooth scatter plots and time plots. For example, `smooth.spline` and `hpfiler` in R.
- A **smoothing spline** for an equally spaced time series  $y_{1:N}^*$  collected at times  $t_{1:N}$  is the sequence  $x_{1:N}$  minimizing the **penalized sum of squares (PSS)**, which is defined as

$$[\text{SS1}] \quad \text{PSS}(x_{1:N}; \lambda) = \sum_{n=1}^N (y_n^* - x_n)^2 + \lambda \sum_{n=3}^N (\Delta^2 x_n)^2.$$

- The spline is defined for all times, but here we are only concerned with its value at the times  $t_{1:N}$ .
- Here,  $\Delta x_n = (1 - B)x_n = x_n - x_{n-1}$ .

- The **smoothing parameter**,  $\lambda$ , penalizes  $x_{1:N}$  to prevent the spline from interpolating the data.
- If  $\lambda = 0$ , the spline will go through each data point, i.e,  $x_{1:N}$  will interpolate  $y_{1:N}^*$ .
- If  $\lambda = \infty$ , the spline will be the ordinary least squares regression fit,

$$x_n = \alpha + \beta n, \quad (20)$$

since  $\Delta^2(\alpha + \beta n) = 0$ .

- Now consider the linear Gaussian model,

$$[\text{SS2}] \quad X_n = 2X_{n-1} - X_{n-2} + \epsilon_n, \quad \epsilon_n \sim \text{iid } N[0, \sigma^2/\lambda]$$

$$[\text{SS3}] \quad Y_n = X_n + \eta_n, \quad \eta_n \sim \text{iid } N[0, \sigma^2]$$

- Note that  $\Delta^2 X_n = \epsilon_n$ .
- We will show that [SS1] is equivalent to [SS2,SS3].

# Constructing a linear Gaussian POMP (LG-POMP) model matching [SS2] and [SS3]

**Question 10.4.**  $\{X_n, Y_n\}$  defined in [SS2] and [SS3] is not quite an LG-POMP model. However, we can use  $\{X_n\}$  and  $\{Y_n\}$  to build an LG-POMP model. How?

$$\text{Set } \tilde{X}_n = \begin{pmatrix} X_n \\ X_{n-1} \end{pmatrix}, \text{ so } \tilde{X}_n = \begin{pmatrix} 2 & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} X_{n-1} \\ X_{n-2} \end{pmatrix} + \begin{pmatrix} \epsilon_n \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 2 & -1 \\ 1 & 0 \end{pmatrix} \tilde{X}_{n-1} + \tilde{\epsilon}_n$$

for  $\tilde{\epsilon}_n = \begin{pmatrix} \epsilon_n \\ 0 \end{pmatrix}$ .

Strategy: put all necessary lags into the latent process.

# Deriving the penalized spline from the LG-POMP

- The joint density of  $X_{1:N}$  and  $Y_{1:N}$  in [SS2,SS3] is

$$f_{X_{1:N}Y_{1:N}}(x_{1:N}, y_{1:N}) = f_{X_{1:N}}(x_{1:N}) f_{Y_{1:N}|X_{1:N}}(y_{1:N} | x_{1:N}). \quad (21)$$

Taking logs of (21) we get

$$\log f_{X_{1:N}Y_{1:N}}(x_{1:N}, y_{1:N}) = \log f_{X_{1:N}}(x_{1:N}) + \log f_{Y_{1:N}|X_{1:N}}(y_{1:N} | x_{1:N}).$$

- [SS2,SS3] tell us that  $\{\Delta^2 X_n, n \in 1 : N\}$  and  $\{Y_n - X_n, n \in 1 : N\}$  are independent normal $[0, \sigma^2/\lambda]$  and normal $[0, \sigma^2]$ . Thus,

$$\begin{aligned} \log f_{X_{1:N}Y_{1:N}}(x_{1:N}, y_{1:N}; \sigma, \lambda) = \\ \frac{-1}{2\sigma^2} \sum_{n=1}^N (y_n - x_n)^2 + \frac{-\lambda}{2\sigma^2} \sum_{n=3}^N (\Delta^2 x_n)^2 + C. \end{aligned} \quad (22)$$

- Here,  $C$  depends on  $\sigma$  and  $\lambda$  but not on  $y_{1:N}$ .  $C$  depends on the initial terms  $x_0$  and  $x_{-1}$ , but we suppose these can be ignored, for example by modeling them with an improper uniform density.

- Comparing (22) with [SS1], we see that maximizing the density  $f_{X_{1:N}Y_{1:N}}(x_{1:N}, y_{1:N}^*; \sigma, \lambda)$  as a function of  $x_{1:N}$  is the same problem as finding the smoothing spline by minimizing the penalized sum of squares.
- For a Gaussian density, the mode (i.e., the maximum of the density) is equal to the expected value. Therefore, we have

$$\begin{aligned}
 \arg \min_{x_{1:N}} \text{PSS}(x_{1:N}; \lambda), &= \arg \max_{x_{1:N}} f_{X_{1:N}Y_{1:N}}(x_{1:N}, y_{1:N}^*; \sigma, \lambda), \\
 &= \arg \max_{x_{1:N}} \frac{f_{X_{1:N}Y_{1:N}}(x_{1:N}, y_{1:N}^*; \sigma, \lambda)}{f_{Y_{1:N}}(y_{1:N}^*; \sigma, \lambda)}, \\
 &= \arg \max_{x_{1:N}} f_{X_{1:N}|Y_{1:N}}(x_{1:N} | y_{1:N}^*; \sigma, \lambda), \\
 &= \mathbb{E}[X_{1:N} | Y_{1:N} = y_{1:N}^*; \sigma, \lambda].
 \end{aligned}$$

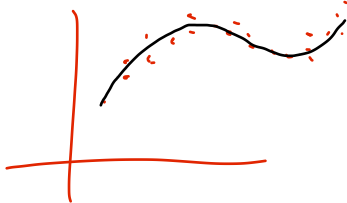
- Because a (conditional) normal distribution is characterized by its (conditional) mean and variance, the smoothing calculation for an LG-POMP model involves finding the conditional mean and variance of  $X_n$  given  $Y_{1:N} = y_{1:N}^*$ .
- We conclude that the smoothing problem for this LG-POMP model is the same as the spline smoothing problem defined by [SS1].
- If you have experience using smoothing splines, this connection may help you transfer that experience to POMP models.
- Once you have experience with POMP models, this connection helps you understand spline smoothers that are commonly used in many applications.
- For example, the smoothing parameter  $\lambda$  could be selected by maximum likelihood for the POMP model.



# Why do we penalize by $\sum_n (\Delta^2 X_n)^2$ when smoothing?

**Question 10.5.** We found that the smoothing spline corresponds to a particular choice of LG-POMP model given by [SS2, SS3], Why do we choose that penalty, rather than the equivalent penalty from some other LG-POMP model?

The spline penalty happens to match the physics of a thin metal plate bent to match data.  
 A "spline" is a metal plate used as a technical drawing tool for making curves. This is not a statistical justification: We are constrained by software.



**Note:** This LG-POMP model is sometimes reasonable, but presumably there are other occasions when a different LG-POMP model would lead to superior performance.

# The Kalman filter

- The **Kalman filter** is the name given to the prediction, filtering and smoothing formulas [P4–P9] for the linear Gaussian partially observed Markov process (LG-POMP) model.
- Linear Gaussian models have Gaussian conditional distributions.
- The integrals in the general POMP formulas can be found exactly for the Gaussian distribution, leading to linear algebra calculations of conditional means and variances.
- The R function `arma()` uses a Kalman filter to evaluate the likelihood of an ARMA model (or ARIMA, SARMA, SARIMA).

## Review of the multivariate normal distribution

- A random variable  $X$  taking values in  $\mathbb{R}^{d_X}$  is **multivariate normal** with mean  $\mu_X$  and variance  $\Sigma_X$  if we can write

$$X = \mathbb{H}Z + \mu_X,$$

where  $Z$  is a vector of  $d_X$  independent identically distributed  $\text{normal}[0, 1]$  random variables and  $\mathbb{H}$  is a  $d_X \times d_X$  matrix square root of  $\Sigma_X$ , i.e.,

$$\mathbb{H}\mathbb{H}^T = \Sigma_X.$$

- A matrix square root of this type exists for any covariance matrix, though the choice of  $\mathbb{H}$  is not unique.
- We write  $X \sim \text{normal}[\mu_X, \Sigma_X]$ . If  $\Sigma_X$  is invertible,  $X$  has a probability density function, *determinant of  $\Sigma_X$*

$$f_X(x) = \frac{1}{(2\pi)^{d_X/2} |\Sigma_X|} \exp \left\{ -\frac{(x - \mu_X) [\Sigma_X]^{-1} (x - \mu_X)^T}{2} \right\}.$$

## Joint multivariate normal vectors

$X$  and  $Y$  are **joint multivariate normal** if the combined vector

$$Z = \begin{pmatrix} X \\ Y \end{pmatrix}$$

is multivariate normal. In this case, we write

$$\mu_Z = \begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \quad \Sigma_Z = \begin{pmatrix} \Sigma_X & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_Y \end{pmatrix},$$

where

$$\Sigma_{XY} = \text{Cov}(X, Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)^T].$$

- For joint multivariate normal random variables  $X$  and  $Y$ , we have the useful property that the conditional distribution of  $X$  given  $Y = y$  is multivariate normal, with conditional mean and variance

$$[\text{KF1}] \quad \mu_{X|Y}(y) = \mu_X + \Sigma_{XY} \Sigma_Y^{-1} (y - \mu_Y),$$

$$[\text{KF2}] \quad \Sigma_{X|Y} = \Sigma_X - \Sigma_{XY} \Sigma_Y^{-1} \Sigma_{YX}.$$

- We write this as

$$X | Y = y \sim \text{normal}[\mu_{X|Y}(y), \Sigma_{X|Y}].$$

- The joint multivariate normal has a special property that the conditional variance of  $X$  given  $Y = y$  does not depend on the value of  $y$ . In non-Gaussian situations, it will usually depend on  $y$ .
- If  $\Sigma_Y$  is not invertible, we can interpret  $\Sigma_Y^{-1}$  as a generalized inverse.

## Notation for the Kalman filter recursions

We define the conditional means and variances for the filtering, prediction and smoothing distributions:

$$[\text{KF3}] \quad X_n \mid Y_{1:n-1} = y_{1:n-1} \sim \text{normal}[\mu_n^P(y_{1:n-1}), \Sigma_n^P],$$

$$[\text{KF4}] \quad X_n \mid Y_{1:n} = y_{1:n} \sim \text{normal}[\mu_n^F(y_{1:n}), \Sigma_n^F],$$

$$[\text{KF5}] \quad X_n \mid Y_{1:N} = y_{1:N} \sim \text{normal}[\mu_n^S(y_{1:N}), \Sigma_n^S].$$

- For data  $y_{1:N}^*$ , we call  $\mu_n^P = \mu_n^P(y_{1:n-1}^*) = \mathbb{E}[X_n \mid Y_{1:n-1} = y_{1:n-1}^*]$  the **prediction mean**, and  $\Sigma_n^P$  the **prediction variance**.
- $\mu_n^F = \mu_n^F(y_{1:n}^*) = \mathbb{E}[X_n \mid Y_{1:n} = y_{1:n}^*]$  is the **filter mean** and  $\Sigma_n^F$  the **filter variance**.
- $\mu_n^S = \mu_n^S(y_{1:N}^*) = \mathbb{E}[X_n \mid Y_{1:N} = y_{1:N}^*]$  is the **smoothed mean** and  $\Sigma_n^S$  the **smoothed variance**.

# The Kalman matrix recursions

- Applying the properties of linear combinations of Normal random variables, we get the Kalman filter and prediction recursions:

$$[\text{KF6}] \quad \mu_{n+1}^P(y_{1:n}) = \mathbb{A}_{n+1} \mu_n^F(y_{1:n})$$

$$[\text{KF7}] \quad \Sigma_{n+1}^P = \mathbb{A}_{n+1} \Sigma_n^F \mathbb{A}_{n+1}^T + \mathbb{U}_{n+1},$$

$$[\text{KF8}] \quad \Sigma_n^F = ([\Sigma_n^P]^{-1} + \mathbb{B}_n^T \mathbb{V}_n^{-1} \mathbb{B}_n)^{-1},$$

$$[\text{KF6}] \quad \mu_n^F(y_{1:n}) = \mu_n^P(y_{1:n-1}) + \Sigma_n^F \mathbb{B}_n^T \mathbb{V}_n^{-1} \{y_n - \mathbb{B}_n \mu_n^P(y_{1:n-1})\}.$$

## Outline of a derivation of the Kalman matrix recursions

- The prediction recursions [KF6] and [KF7] follow from the property that if  $X$  is a  $d$ -dimensional multivariate normal,  $X \sim \text{normal}(\mu, \Sigma)$ , then  $\mathbb{A}X + b \sim \text{normal}(\mathbb{A}\mu + b, \mathbb{A}\Sigma\mathbb{A}^T)$ .
- Note that the multivariate normal identities [KF1,KF2] also hold when all variables are conditioned on some additional joint Gaussian variable, in this case  $Y_{1:n-1}$ .
- [KF8] and [KF9] can be deduced by writing out the joint density,

$$f_{X_n Y_n | Y_{1:n-1}}(x_n, y_n | y_{1:n-1}) \quad (23)$$

and completing the square in the exponent. The conditional density of  $X_n$  given  $Y_{1:n}$  is proportional to this joint density, with proportionality constant allowing integration to one.

**Exercise 10.5.** The derivation of the Kalman algebra is not central to this course. However, working through the algebra to your own satisfaction is a good exercise.



- The Kalman filter matrix equations are easy to code, and quick to compute unless the dimension of the latent space is very large.
- In numerical weather forecasting, with careful programming, they are solved with latent variables having dimension  $d_X \approx 10^7$ .
- A similar computation gives backward Kalman recursions. Putting the forward and backward Kalman recursions together, as in [P9], is called **Kalman smoothing**.

## Further reading

- The approach in this chapter is aligned with King *et al.* (2016)
- Chapter 6 of Shumway and Stoffer (2017) gives an approach emphasizing linear Gaussian state space models.

## License, acknowledgments, and links

- Licensed under the [Creative Commons Attribution-NonCommercial license](#). Please share and remix non-commercially, mentioning its origin.
- The materials builds on [previous courses](#).
- Compiled on March 6, 2021 using R version 4.0.4.



[Back to course homepage](#)

## References

- King AA, Nguyen D, Ionides EL (2016). “Statistical Inference for Partially Observed Markov Processes via the R Package pomp.” *Journal of Statistical Software*, **69**(12), 1–43. doi: [10.18637/jss.v069.i12](https://doi.org/10.18637/jss.v069.i12).
- Shumway RH, Stoffer DS (2017). *Time Series Analysis and its Applications: With R Examples*. Springer. URL <http://www.stat.pitt.edu/stoffer/tsa4/tsa4.pdf>.