Chapter 9. Introduction to partially observed Markov process models

Objectives

1. Develop a framework for thinking about models that consist of a stochastic dynamic system observed with noise.
2. In the linear Gaussian case, develop matrix operations to find an exact and computationally fast algorithm for the likelihood function. This algorithm is called the **Kalman filter**.
3. Understand how the Kalman filter is used to compute the likelihood for ARMA models.
4. See how the Kalman filter also facilitates forecasting and estimation of the state of the unobserved process.
5. Start to investigate the general nonlinear filtering equations.

Partially observed Markov processes (POMP) models

- Uncertainty and variability are ubiquitous features of processes in the biological and social sciences. A physical system obeying Newton's laws is fully predictable, but complex systems are in practice not perfectly predictable—we can only forecast weather reliably in the near future.
- Basic time series models of deterministic trend plus colored noise imply perfect predictability if the trend function enables extrapolation.
- To model variability and unpredictability in low frequency components, we may wish to specify a random process model for how the system evolves. We could call this a "stochastic trend" approach, though that is an oxymoron since we've defined trend to be expected value for the model.
- As in the deterministic signal plus noise model, we will model the observations as random variables conditional on the trajectory of a **latent process**. It can also be called a **state process** or a **hidden process**.

The Markov property

- A standard class of latent process models is characterized by the requirement that the future evolution of the system depends only on the current state, plus randomness introduced in future.
- A model of this type is called a **Markov chain** for a discrete time model or a **Markov process** in continuous time.
- We use the term Markov process for both discrete and continous time.
- **Partial observations** here mean either or both of (i) measurement noise; (ii) entirely unmeasured latent variables. Both these features are present in many systems.
- A **partially observed Markov process** (POMP) model is defined by putting together a latent process model and an observation model.

- Often, much of the scientific interest is in understanding what models for the behavior of this latent process are consistent with the data.
- A good model for the underlying, but imperfectly observed, dynamics of a system can also lead to a skillful forecast.
- We are going to introduce a general framework for specifying POMP models. This generality will give us the flexibility to develop models and methods appropriate to a range of applications.

Discrete time Markov processes

A time series model $X_{0:N}$ is a **Markov process** model if the conditional densities satisfy the **Markov property** that, for all $n \in 1:N$.

[MP1] $$f_{X_n|X_{1:n-1}}(x_n \,|\, x_{1:n-1}) = f_{X_n|X_{n-1}}(x_n \,|\, x_{n-1}),$$

We suppose that the random process X_n occurs at time t_n for $n \in 0:N$, so the discrete time process corresponds to time points in continuous time.

Initial conditions

- We have **initialized** the Markov process model at a time t_0, although we will suppose that data are collected only at times $t_{1:N}$.
- The initialization model could be deterministic (a fixed value) or a random variable.
- Formally, a fixed initial value is a special case of a discrete distribution having a point mass with probability one at the fixed value. Therefore, fixed initial values are covered in our framework since we use probability density functions to describe both discrete and continuous probability distributions.
- Mathematically, a probability mass function (for discrete distributions) is a probability density on a discrete space. We avoid getting sidetracked on to that topic, but it is worth noting that there is a proper mathematical justification for treating a probability mass function as a type of probability density function.
- It is not important whether to adopt the convention that the Markov process model is intialized at time t_1 or at some previous time t_0. Here, we follow the choice to use t_0.

The process nodel and one-step predictions

- 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.
- In words, the Markov property says 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 POMP model, the full joint distribution of the latent process is entirely specified by the one-step transition densities, given the initial value. We show this below.
- Therefore, we also call $f_{X_n|X_{n-1}}(x_n \,|\, x_{n-1})$ the **process model**.

Question 9.1. Use [MP1] 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

[MP2] $$f_{X_{0:N}}(x_{0:N}) = f_{X_0}(x_0)\prod_{n=1}^{N} f_{X_n \mid X_{n-1}}(x_n \mid x_{n-1}).$$

$$f_{X_{0:N}}(x_{0:N}) = f_{X_0}(x_0)\prod_{n=1}^{N} f_{X_n \mid X_{0:n-1}}(x_n \mid x_{0:n-1})$$

the general factorization of a joint density.

$$= f_{X_0}(x_0)\prod_{n=1}^{N} f_{X_n \mid X_{n-1}}(x_n \mid x_{n-1})$$

using MP1.

Question 9.2. Show that a causal Gaussian AR(1) process is a Markov process.

AR(1) Model: $Y_n = \phi Y_{n-1} + \epsilon_n$, $\epsilon_n \sim$ iid $N[0, \sigma^2]$.

Now, ϵ_n is independent of $Y_{1:n-1}$ by construction.

So, Y_n given $Y_{1:n-1}$ has a distribution that just depends on Y_{n-1}, i.e. $\{Y_n\}$ is Markov.

note: for a general AR(1), the errors are uncorrelated but not independent, so the process is not necessarily Markov.

Time homogeneous transitions and stationarity

In general, the one step transition probability density in a POMP model can depend on n. A latent process model $X_{0:N}$ is **time-homogeneous** if the one step transition probability density does not depend on n, so there is a conditional density $f(y\,|\,x)$ such that, for all $n \in 1:N$,

in general, $f_{X_n|X_{n-1}}$ can be a different function to $f_{X_{n-1}|X_{n-2}}$

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

Question 9.3. If $X_{0:N}$ is strict stationary then it is time homogeneous. Why?

Strict stationarity implies all joint and conditional densities are shift invariant, so

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

Question 9.4. Time homogeneity does not necessarily imply stationarity. Find a counter-example. What has to be added to time homogeneity to get stationarity? Random walk (with or without drift). $X_0 = 0$, $X_n = X_{n-1} + \mu + \varepsilon_n$, $\varepsilon_n \sim$ iid $N[0,\sigma^2]$. Then, the distribution of X_n given $X_{n-1} = x$ doesn't depend on n. But, $E[X_n] = n\mu$, $\text{Var}[X_n] = n\sigma^2$ so $\{X_n\}$ is not stationary.

The measurement model

- We model the observation process random variables $Y_{1:N}$.
- For state space models, we will generally write the data as $y_{1:N}$.
- We model the measurement at time t_n to depend only on the value of the latent process at time t_n, conditionally independent of all other latent process and observation process variables. Formally, this assumption is,

$$\text{[MP3]} \quad f_{Y_n|X_{0:N},Y_{1:n-1},Y_{n+1:N}}(y_n \,|\, x_{0:N}, y_{1:n-1}, y_{n+1:N}) = f_{Y_n|X_n}(y_n \,|\, x_n).$$

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

Time-inhomoegeneous 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 $g(y \,|\, x)$ such that, for all $n \in 1:N$,
$$f_{Y_n|X_n}(y_n \,|\, x_n) = g(y_n \,|\, x_n).$$

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 of the latent process at time t_{n+1} given data up to time t_n involves finding

it is usually good to quantify uncertainty.

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

- We may want to carry out prediction (also called forecasting) more than one time step ahead. However, unless specified otherwise, the prediction calculation will be one-step prediction.
- One-step prediction turns out to be closely related to computing the likelihood function, and therefore central to statistical inference.
- We have required our prediction to be a conditional probability density, not a point estimate. In the context of forecasting, this is called a **probabilistic forecast**, and has advantages over a point estimate forecast. What are they? Are there any disadvantages to probabilistic forecasting?

Users might not understand a probabilistic forecast?

Filtering

name comes from the history of signal processing: a noisy signal was "filtered" through capacitors & resistors to estimate the signal being sent.

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

$$f_{X_n|Y_{1:n}}(x_n \,|\, y_{1:n}).$$

- This can be calculated numerically or algebraically. We will also see that Monte Carlo methods can be a good tool.

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

$$f_{X_n|Y_{1:N}}(x_n \,|\, y_{1:N}).$$

The likelihood

- The model may depend on a parameter vector θ.
- Since we have not explicitly written this dependence above, the likelihood calculation is to evaluate the joint density of $Y_{1:N}$ at the data,
$$f_{Y_{1:N}}(y_{1:N}).$$
- If we can compute this at any value of θ we choose, 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 and filtering formulas

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

prediction at time n

[MP4]
$$
\begin{aligned}
& f_{X_n|Y_{1:n-1}}(x_n \,|\, y_{1:n-1}) \\
&= \int f_{X_{n-1}|Y_{1:n-1}}(x_{n-1} \,|\, y_{1:n-1}) f_{X_n|X_{n-1}}(x_n \,|\, x_{n-1})\, dx_{n-1}.
\end{aligned}
$$

one-step transition density

filtering at time n-1.

- To make this formula work for $n = 1$, we need the convention that $1 : k$ is the empty set when $k = 0$. Conditioning on an empty collection of random variables is the same as not conditioning at all! In this case, we have by definition that

$$f_{X_0|Y_{1:0}}(x_0 \,|\, y_{1:0}) = f_{X_0}(x_0).$$

- In other words, the filtering calcuation at time t_0 is the initial density for the latent process. This makes sense, since at time t_0 we have no data to condition on.

Hints for homework: deriving the recursion formulae

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

Example 1. From

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

we can condition on Z to obtain

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

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

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

Bayes formula conditioned on Z.

Question 9.5. Why is the following identity true?

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

assimilating the new data at time n.

prediction at time n

- 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 $n \in 1:N$,

[MP5] $$f_{X_n|Y_{1:n}}(x_n\,|\,y_{1:n}) = \frac{f_{X_n|Y_{1:n-1}}(x_n\,|\,y_{1:n-1})\, f_{Y_n|X_n}(y_n\,|\,x_n)}{f_{Y_n|Y_{1:n-1}}(y_n\,|\,y_{1:n-1})}.$$

normalizing constant

This can be seen as a conditional form of Bayes theorem.

- The denominator in the filtering formula [MP5] 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**,

[MP6]
$$f_{Y_n|Y_{1:n-1}}(y_n \,|\, y_{1:n-1}) = \int f_{X_n|Y_{1:n-1}}(x_n \,|\, y_{1:n-1})\, f_{Y_n|X_n}(y_n \,|\, x_n)\, dx_n.$$

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

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

Question 9.6. Give a detailed derivation of [MP4], [MP5] and [MP6], being careful to note when you use the Markov property [MP1].

See HW 5

Computation of the likelihood

- The likelihood of the entire dataset, $y_{1:N}$ can be found from [MP6], using the identity

$$\text{[MP7]} \qquad f_{Y_{1:N}}(y_{1:N}) = \prod_{n=1}^{N} f_{Y_n|Y_{1:n-1}}(y_n \,|\, y_{1:n-1}).$$

- As above, this formula [MP7] requires the convention that $1:k$ is the empty set when $k=0$, so the first term in the product is

$$f_{Y_1|Y_{1:0}}(y_1 \,|\, y_{1:0}) = f_{Y_1}(y_1).$$

- If our model has an unknown parameter θ, the likelihood identity [MP7] lets us evaluate the log likelihood function,

$$\ell(\theta) = \log f_{Y_{1:N}}(y_{1:N}\,;\theta).$$

The smoothing formulas

- 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 forwards in time (we use the solution at time t_{n-1} to carry out the computation at time t_n).
- There are similar **backwards recursion formulas**,

[MP8] $$f_{Y_{n:N}|X_n}(y_{n:N} \,|\, x_n) = f_{Y_n|X_n}(y_n \,|\, x_n) f_{Y_{n+1:N}|X_n}(y_{n+1:N} \,|\, x_n).$$

[MP9] $$\begin{aligned} & f_{Y_{n+1:N}|X_n}(y_{n+1:N} \,|\, x_n) \\ & \quad = \int f_{Y_{n+1:N}|X_{n+1}}(y_{n+1:N} \,|\, x_{n+1}) \, f_{X_{n+1}|X_n}(x_{n+1} \,|\, x_n) \, dx_{n+1}. \end{aligned}$$

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

$$\text{[MP10]} \quad f_{X_n|Y_{1:N}}(x_n \,|\, y_{1:N}) = \frac{f_{X_n|Y_{1:n-1}}(x_n \,|\, y_{1:n-1})\, f_{Y_{n:N}|X_n}(y_{n:N} \,|\, x_n)}{f_{Y_{n:N}|Y_{1:n-1}}(y_{n:N} \,|\, y_{1:n-1})}.$$

Question 9.7. Show how [MP8], [MP9] and [MP10] follow from the basic properties of conditional densities combined with the Markov property.

See HW 5

An algebraic trick: Using un-normalized identities

- Sometimes we can avoid calculating a normalizing constant for a density, since it can be worked out later using the property that the probability density function must integrate to 1.
- The denominators $f_{Y_n|Y_{1:n-1}}(y_n \,|\, y_{1:n-1})$ and $f_{Y_{n:N}|Y_{1:n-1}}(y_{n:N} \,|\, y_{1:n-1})$, in equations [MP5] and [MP10] respectively, may sometimes be hard to compute.
- We can simplify [MP5] and [MP10] using the proportionality relationship $\propto$. This gives,

[MP5′] $f_{X_n|Y_{1:n}}(x_n \,|\, y_{1:n}) \propto f_{X_n|Y_{1:n-1}}(x_n \,|\, y_{1:n-1}) \, f_{Y_n|X_n}(y_n \,|\, x_n)$,

[MP10′] $f_{X_n|Y_{1:N}}(x_n \,|\, y_{1:N}) \propto f_{X_n|Y_{1:n-1}}(x_n \,|\, y_{1:n-1}) \, f_{Y_{n:N}|X_n}(y_{n:N} \,|\, x_n)$.

- The normalizing "constant" avoided in equations [MP5'] and [MP10'] does depend on $y_{1:N}$. However, the data are fixed constants. The variable in these equations is x_n.

Linear Gaussian POMP (LG-POMP) models

- Linear Gaussian partially observed Markov process (LG-POMP) models have many applications
- Gassian ARMA models are LG-POMP models. The POMP recursion formulas give a computationally efficient way to obtain the likelihood of a Gaussian ARMA model.
- The computations for smoothing splines can be written as an LG-POMP model, enabling computationally efficient spline smooothing.
- 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.
- LG-POMP models are widely used in engineering, especially for control applications. If a scientific and engineering application is not too far from linear and Gaussian, you save a lot of effort if an LG-POMP model is appropriate. General nonlinear POMP models usually involve intensive Monte Carlo computation.

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 $n \in 1:N$,

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

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

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

Suppose $\{Y_n\}$ is a Gaussian ARMA(p,q) model with noise process $\omega_n \sim N[0, \sigma^2]$ and specification

[LG3] $$Y_n = \sum_{j=1}^{p} \phi_j Y_{n-j} + \omega_n + \sum_{k=1}^{q} \psi_q \omega_{n-k}.$$

Set $r = \max(p, q+1)$ so that $\{Y_n\}$ is also ARMA(r,r-1). Our LG-POMP representation has $d_X = r$, with

$$\mathbb{B}_n = \mathbb{B} = (1, 0, 0, \ldots, 0)$$

and

$$\mathbb{V}_n = \mathbb{V} = 0.$$

Therefore, Y_n is the first component of X_n, observed without measurement error.

Now, define

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

We can check that the ARMA equation [LG3] corresponds to 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 in the form of a time-homogenous LG-POMP, with $\mathbb{A}$, $\mathbb{B}$ and $\mathbb{V}$ defined above, and

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

Different POMPs 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 r^2 parameters and the ARMA model has order r parameters, so we might expect there are many ways to parameterize the ARMA model as a special case of the much larger LG-POMP model.
- The same can be true of non-Gaussian POMPs, but it is easier to see in the Gaussian case.

The basic structural model and its LG-POMP representation

- The **basic structural model** is an econometric model used for forecasting.
- The basic stuctural model supposes that the observation process $Y_{1:N}$ is 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 all these quantities are perturbed with Gaussian white noise at each time point. So, we have the following model equations

$$\begin{array}{lrcl}
\text{[BSM1]} & Y_n &=& L_n + S_n + \epsilon_n \\
\text{[BSM2]} & L_n &=& L_{n-1} + T_{n-1} + \xi_n \\
\text{[BSM3]} & T_n &=& T_{n-1} + \zeta_n \\
\text{[BSM4]} & S_n &=& -\sum_{k=1}^{11} S_{n-k} + \eta_n
\end{array}$$

- We suppose $\epsilon_n \sim N[0,\sigma^2_\epsilon]$, $\xi_n \sim N[0,\sigma^2_\xi]$, $\zeta_n \sim N[0,\sigma^2_\zeta]$, and $\eta_n \sim N[0,\sigma^2_\eta]$.

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$^2_{\mathrm{eps}}$.”

To put [BSM1-4] in the form of an LG-POMP model, we set

[BSM5] $$X_n = (L_n, T_n, S_n, S_{n-1}, S_{n-2}, \dots, S_{n-10})^{\mathrm{T}}.$$

Then, we have

[BSM6] $$Y_n = (1, 0, 1, 0, 0, \dots, 0) X_n + \epsilon_n,$$

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

From [BSM5] and [BSM6], we can read off the matrices $\mathbb{A}$, $\mathbb{B}$, $\mathbb{U}$ and $\mathbb{V}$ in the LG-POMP representation of 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` 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

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

- 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**, λ, 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,

$$x_n = \alpha + \beta n,$$

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

- Now consider the model,

$$\text{[SS2]} \qquad \begin{array}{rcll} X_n & = & 2X_{n-1} - X_{n-2} + \epsilon_n, & \epsilon_n \sim \text{iid } N[0, \sigma^2/\lambda] \\ Y_n & = & X_n + \eta_n & \eta_n \sim \text{iid } N[0, \sigma^2]. \end{array}$$

- Note that $\Delta^2 X_n = \epsilon_n$.

Constructing a linear Gaussian POMP (LG-POMP) model from [SS2]

Question 9.8. $\{X_n, Y_n\}$ defined in [SS2] is not quite an LG-POMP model. However, we can use $\{X_n\}$ and $\{Y_n\}$ to build an LG-POMP model. How?

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

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

Strategy: put into POMP format by including all necessary lags in the latent state.

- The joint density of $X_{1:N}$ and $Y_{1:N}$ in [SS2] can be written as

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

- Taking logs,

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

- Suppose the initial conditions are irrelevant (either unknown parameters or an improper Gaussian distribution with infinite variance). Noting that $\{\Delta^2 X_n, n\in 1:N\}$ and $\{Y_n - X_n, n \in 1:N\}$ are collections of independent Normal random variables with mean zero and variances σ^2/λ and σ^2 respectively, we have

[SS3] $$\log f_{X_{1:N}Y_{1:N}}(x_{1:N},y_{1:N}\,;\sigma,\lambda) = \frac{-1}{2\sigma^2}\sum_{n=1}^N (y_n - x_n)^2 + \frac{-\lambda}{2\sigma^2}\sum_{n=3}^N (\Delta^2 x_n)^2 + C.$$

- In [SS3], C is a constant depending on σ and λ but not $x_{1:N}$ or $y_{1:N}$.
- Comparing [SS3] 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 in [SS1].

- For a Gaussian density, the mode (i.e., the maximum of the density) is equal to the expected value. Therefore, we have

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

Smoothing problem for a POMP:
find $f_{X_{1:N}|Y_{1:N}}$

For a Gaussian problem, $f_{X_{1:N}Y_{1:N}}$ is defined by the mean & covariance matrix, so (*) follows from the smoothing.

- The smoothing calculation for an LG-POMP model involves finding the 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, we might propose that the smoothing parameter λ could be selected by maximum likelihood for the POMP model.

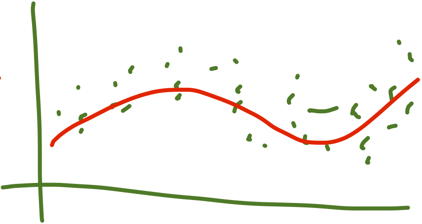
Question 9.9. Why do we use $\Delta^2 X_n = \epsilon_n$ for our smoothing model?

- Seeing that the smoothing spline arrives from the particular choice of LG-POMP model in equation [SS2] could make you wonder why we choose that model. Any ideas?

smoothing spline has same intuition, in terms of 2^{nd} derivative.

smoothing spline, loess are provided in software.

In practice, we are often constrained by software.



- Even if this LG-POMP model is sometimes reasonable, presumably there are other occasions when a different LG-POMP model would be a superior choice for smoothing.

The Kalman filter

- We find exact versions of the prediction, filtering and smoothing formulas [MP4–10] for the linear Gaussian partially observed Markov process (LG-POMP) model [LG1,LG2].

- In the linear Gaussian case, the conditional probability density functions in [MP4–10] are specified by the conditional mean and conditional variance.

Review of the multivariate normal distribution

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

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

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

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

- The choice of $\mathbb{H}$ is not unique, and a matrix square root of this type exists for any covariance matrix because covariance matrices are positive semi-definite.
- We write $X \sim N\left[\mu_X, \Sigma_X\right]$.
- $X \sim N\left[\mu_X, \Sigma_X\right]$ has a probability density function if and only if Σ_X is invertible. This density is given by

determinant of Σ_X

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

Join‸tly multivariate normal vectors

X and Y are **jointly multivariate normal** if the combined vector

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

is multivariate normal. In this case, we write

$$\mu_W = \begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \qquad \Sigma_W = \begin{pmatrix} \Sigma_X & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_Y \end{pmatrix},$$

where

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

- For jointly 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

[KF1]
$$\begin{aligned} \mu_{X|Y}(y) &= \mu_X + \Sigma_{XY}\Sigma_Y^{-1}\big(y-\mu_Y\big), \\ \Sigma_{X|Y} &= \Sigma_X - \Sigma_{XY}\Sigma_Y^{-1}\Sigma_{YX}. \end{aligned}$$

- We write this as

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

- In general, the conditional variance of X given $Y=y$ will depend on y (remind yourself of the definition of conditional variance). In the special case where X and Y are jointly multivariate normal, this conditional variance happens not to depend on the value of y.
- If Σ_Y is not invertible, to make [KF1] work we have to interpret Σ_Y^{-1} as a generalized inverse.

Notation for the Kalman filter recursions

To write the Kalman filter, we define the following notation, Since the system is Gaussian, the filtering and prediction distributions are defined by their mean and variance.

prediction

$$\text{[KF2]} \qquad \begin{array}{lcll} X_n \,|\, Y_{1:n-1} = y_{1:n-1} & \sim & N\big[\,\mu_n^P(y_{1:n-1}),\, \Sigma_n^P\,\big], & \\ X_n \,|\, Y_{1:n} = y_{1:n} & \sim & N\big[\,\mu_n^F(y_{1:n}),\, \Sigma_n^F\,\big], & \leftarrow \text{Filtering} \\ X_n \,|\, Y_{1:N} = y_{1:N} & \sim & N\big[\,\mu_n^S(y_{1:N}),\, \Sigma_n^S\,\big]. & \leftarrow \text{smoothing.} \end{array}$$

To relate this notation to the general POMP recursion formulas, given data $y_{1:N}$, we define the following terminology:

$\mu_n^P(y_{1:n-1}) = \mathbb{E}\big[X_n \,|\, Y_{1:n-1} = y_{1:n-1}\big]$ is the **one-step prediction mean** for time t_n. It is an arbitrary decision we have made to call this the prediction for time t_n (the time for which the prediction is being made) rather than for time t_{n-1} (the time at which the prediction for time t_n becomes available).

$\Sigma_n^P(y_{1:n-1}) = \mathrm{Var}\big(X_n \,|\, Y_{1:n-1} = y_{1:n-1}\big)$ is the **one-step prediction variance** for time t_n.

For a Gaussian model, this conditional variance does not depend on $y_{1:n}$. To make this terminology work for general POMP models we see later, we include possible dependence on $y_{1:n-1}$.

Other related quantities use the same notation:

- $\mu_n^F(y_{1:n}) = \mathbb{E}\big[X_n \,|\, Y_{1:n} = y_{1:n}\big]$ is the **filter mean** for time t_n.
- $\Sigma_n^F(y_{1:n}) = \mathrm{Var}\big(X_n \,|\, Y_{1:n} = y_{1:n}\big)$ is the **filter variance** for time t_n.
- $\mu_n^S(y_{1:N}) = \mathbb{E}\big[X_n \,|\, Y_{1:N} = y_{1:N}\big]$ is the **smoothing mean** for time t_n.
- $\Sigma_n^S(y_{1:N}) = \mathrm{Var}\big(X_n \,|\, Y_{1:N} = y_{1:N}\big)$ is the **smoothing variance** for time t_n.

The Kalman matrix recursions

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

$X_n = \mathbb{A}_n X_{n-1} + \varepsilon_n$

$Y_n = \mathbb{B}_n X_n + \eta_n$

$\varepsilon_n \sim N[0, \mathbb{U}_n]$

$\eta_n \sim N[0, \mathbb{V}_n]$

[KF3] $$\mu^P_{n+1}(y_{1:n}) = \mathbb{A}_{n+1}\mu^F_n(y_{1:n}),$$

[KF4] $$\Sigma^P_{n+1} = \mathbb{A}_{n+1}\Sigma^F_n\mathbb{A}^{\mathrm{T}}_{n+1} + \mathbb{U}_{n+1}.$$

[KF5] $$\Sigma^F_n = \big([\Sigma^P_n]^{-1} + \mathbb{B}^{\mathrm{T}}_n\mathbb{V}^{-1}_n\mathbb{B}_n\big)^{-1}.$$

[KF6] $$\mu^F_n(y_{1:n}) = \mu^P_n(y_{1:n-1}) + \Sigma^F_n\mathbb{B}^{\mathrm{T}}_n\mathbb{V}^{-1}_n\big\{y_n - \mathbb{B}_n\mu^P_n(y_{1:n-1})\big\}.$$

- The prediction recursions [KF3–4] are relatively easy to demonstrate, but it is a good exercise to go through the algebra to your own satisfaction.
- A useful trick for the algebra is to notice that the conditioning identities [KF1] for joint Gaussian random variables continue to hold if left and right are both conditioned on some additional jointly Gaussian variable, such as $Y_{1:n-1}$.

KF5–6 can be deduced by completing the square in an expression for the joint density $f(x_n, y_n \mid y_{1:n-1})$ and noticing that the

Question 9.10. Derive some or all of these equations.

Optional exercise.

- These 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 [MP10], is called **Kalman smoothing**.

Acknowledgments and License

- These notes build on previous versions at `ionides.github.io/531w16` and `ionides.github.io/531w18`.