Efficient Filtering and Fitting of Models Derived from Integro-Difference Equations

Evan Tate Paterson Hughes

Table of contents

1 Integro-difference Based Dynamics **2**

2 Process Decomposition **3**
2.1 Process Noise . . . 4
2.2 Kernel Decomposition . . . 5
2.3 IDEM as a linear dynamical system . . . 5

3 Filtering, Forecasting and Maximum Likelihood Estimation **7**
3.1 The Kalman Filter . . . 7
3.2 The Information Filter . . . 9
3.3 Kalman Smoothers . . . 11

4 EM Algorithm (NEEDS A LOT OF WORK, PROBABLY IGNORE FOR NOW) **12**

5 Algorithm for Maximum Complete-data Likelihood estimation **14**

A Appendix **15**
A.1 Woodbury's identity . . . 15
A.2 Proof of Theorem 3.2.1 . . . 15
A.3 Truly Vague Prior with the Kalman Filter . . . 16

Chapter 1

Integro-difference Based Dynamics

As common and widespread as the problem is, spatio-temporal modelling still presents a great deal of difficulty. Inherently, Spatio-Temporal datasets are almost always high-dimensional, and repeated observations are usually not possible.

Traditionally, the problem has been tackled by the moments (usually the means and covariances) of the process in order to make inference (Wikle, Zammit-Mangion, and Cressie (2019), for example, call this 'descriptive' modelling). While this method can be sufficient for many problems, there are many cases where we are underutilizing some knowledge of the underlying dynamical systems involved. For instance, in temperature models, we know that temperature has movement (convection) and spread (diffusion), and that the state at any given time will depend on its state at previous times [1]. We call models which make use of this 'dynamical' models. Of focus here is the Integro-Difference Equation Model (IDEM), which models diffusion and convection in discrete time by using convolution-like integral equations to model the relation between the process and it's previous state. We use a hierarchical model to represent this type of system;

$$\begin{aligned} Z_t(\mathbf{s}) &= Y_t(\mathbf{s}) + X(\mathbf{s})^\intercal + \epsilon_t(\mathbf{s}) \\ Y_{t+1}(\mathbf{s}) &= \int_{\mathscr{D}_s} \kappa(s,r) Y_t(r) d\mathbf{r} + \omega_t(\mathbf{s}). \end{aligned} \tag{1.1}$$

Where $\omega_t(\mathbf{s})$ is a small scale gaussian variation with no temporal dynamics (Cressie and Wikle 2015 call this a 'spatially descriptive' component), $\mathbf{X}(\mathbf{s})$ are spatially varying covariates (for example, in a large-scale climate scenario, this might simply be latitude), Z is observed data, Y is an unobserved dynamic process, κ is the driving 'kernel' function, and ϵ_t is a gaussian white noise 'measurement error' term.

[1] at least, in a discrete-time scenario. Integro-difference based mechanics can be derived from continuous-time convection-diffusion processes, see Liu, Yeo, and Lu (2022)

Chapter 2

Process Decomposition

In order to work with the process, we likely want to consider the spectral decomposition of it. That is, choose a complete class of spatial spectral basis functions, $\phi_i(\mathbf{s})$, and decompose;

$$Y_t(\mathbf{s}) \approx \sum_{i=1}^{r} \alpha_{i,t}\phi_i(\mathbf{s}). \tag{2.1}$$

where we truncate the expansion at some $r \in \mathbb{N}$. Notice that we can write this in vector/matrix form; considering times $t = 1, 2, \dots, T$, we set

$$\begin{aligned} (\mathbf{s}) &= (\phi_1(\mathbf{s}), \phi_2(\mathbf{s}), \dots, \phi_r(\mathbf{s}))^\intercal \\ {}_t &= (\alpha_{1,t}, \alpha_{2,t}, \dots, \alpha_{r,t})^\intercal \end{aligned} \tag{2.2}$$

Now, (Equation 2.1) gives us

$$Y(\mathbf{s}; t) = {}^\intercal(\mathbf{s})\alpha(t) \tag{2.3}$$

We now want to find the equation defining the evolution of the spectral coefficients, ${}_t$.

Theorem 2.0.1 (Basis form of the state evolution). Define the Gram matrix;

$$\Psi := \int_{\mathcal{D}_s} (\mathbf{s})(\mathbf{s})^\intercal d\mathbf{s} \tag{2.4}$$

Then, the basis coefficients evolve by the equation

$$(t+1) = M(t) + {}_t,$$

where $M = \Psi^{-1} \int\int (\mathbf{s})\kappa(\mathbf{s}, \mathbf{r})(\mathbf{r})^\intercal d\mathbf{r} d\mathbf{s}$ and ${}_t = \Psi^{-1} \int (\mathbf{s})\omega_t(s) d\mathbf{s}$.

Proof 1. (Adapting from Dewar, Scerri, and Kadirkamanathan 2008), write out the process equation, (Equation 1.1), using the first equation of (Equation 2.3);

$$Y(\mathbf{s}; t+1) = (\mathbf{s})\alpha(t+1) = \int_{\mathscr{D}_s} \kappa(\mathbf{s},\mathbf{r})(\mathbf{r})^\intercal(t)d\mathbf{r} + \omega_t(\mathbf{s}),$$

We then multiply both sides by (s) and integrate over $\mathbf{s}$

$$\int_{\mathscr{D}_s} (\mathbf{s})(\mathbf{s})d\mathbf{s}(t+1) = \int (\mathbf{s}) \int \kappa(\mathbf{s},\mathbf{r})(\mathbf{r})^\intercal d\mathbf{r}d\mathbf{s}\ (t) + \int (\mathbf{s})\omega_t(s)d\mathbf{s}$$
$$\Psi(t+1) = \int \int (\mathbf{s})\kappa(\mathbf{s},\mathbf{r})(\mathbf{r})^\intercal d\mathbf{r}d\mathbf{s}\ (t) + \int (\mathbf{s})\omega_t(s)d\mathbf{s}.$$

So, finally, pre-multipling by the inverse of the gram matrix, Ψ^{-1} (Equation 2.4), we arrive at the result. □

2.1 Process Noise

We still have to set out what the process noise, $\omega_t(\mathbf{s})$, and it's spectral couterpart, $_t$, are. Dewar (Dewar, Scerri, and Kadirkamanathan 2008) fixes the variance of $\omega_t(\mathbf{s})$ to be uniform and uncorrelated across space and time, with $\omega_t(\mathbf{s}) \sim \mathcal{N}(0, \sigma^2)$ It is then easily shown that $_t$ is also normal, with $_t \sim \mathcal{N}(0, \sigma^2\Psi^{-1})$.

However, in practice, we simulate in the spectral domain; that is, if we want to keep things simple, it would make sense to specify (and fit) the distribution of $_t$, and compute the variance of $\omega_t(\mathbf{s})$ if needed. This is exactly what the IDE package (Zammit-Mangion 2022) in R does, and, correspondingly, what this JAX project does.

Lemma 2.1.1. Let $_t \sim \mathcal{N}(0, \Sigma_\eta)$, and $\mathbb{C}\mathrm{ov}[_t, _{t+\tau}] = 0, \forall\tau > 0$. Then $\omega_t(\mathbf{s})$ has covariance

$$\mathbb{C}\mathrm{ov}[\omega_t(\mathbf{s}), \omega_{t+\tau}(\mathbf{r})] = \begin{cases} (\mathbf{s})^\intercal\Sigma_\eta(\mathbf{r}) & \text{if } \tau = 0 \\ 0 & \text{else} \end{cases}$$

Proof 2. Consider Ψ_t. It is clearly normal, with expectation zero and variance (using (Equation 2.4)),

$$\begin{aligned} \mathbb{V}\mathrm{ar}[\Psi_t] &= \Psi\,\mathbb{V}\mathrm{ar}[_t]\Psi^\intercal = \Psi\Sigma_\eta\Psi^\intercal, \\ &= \int_{\mathscr{D}_s} (\mathbf{s})(\mathbf{s})^\intercal d\mathbf{s}\ \Sigma_\eta \int_{\mathscr{D}_s} (\mathbf{r})(\mathbf{r})^\intercal d\mathbf{r} \\ &= \int \int_{\mathscr{D}_s^2} (\mathbf{s})(\mathbf{s})^\intercal\ \Sigma_\eta\ (\mathbf{r})(\mathbf{r})^\intercal d\mathbf{r}d\mathbf{s} \end{aligned} \tag{2.5}$$

Since it has zero expectation, we also have

$$\begin{aligned} \mathbb{V}\mathrm{ar}[\Psi_t] &= \mathbb{E}[(\Psi_t)(\Psi_t)^\intercal] = \mathbb{E}[\Psi_{t\,t}^\intercal\Psi^\intercal] \\ &= \mathbb{E}\left[\int_{\mathscr{D}_s} (\mathbf{s})\omega_t(\mathbf{s})d\mathbf{s} \int_{\mathscr{D}_s} (\mathbf{r})^\intercal\omega_t(\mathbf{r})d\mathbf{r}\right] \\ &= \int \int_{\mathscr{D}_s^2} (\mathbf{s})\ \mathbb{E}[\omega_t(\mathbf{s})\omega_t(\mathbf{r})]\ (\mathbf{r})^\intercal d\mathbf{s}d\mathbf{r}. \end{aligned} \tag{2.6}$$

We can see that, comparing (Equation 2.5) and (Equation 2.6), we have

$$\mathbb{C}\mathrm{ov}[\omega_t(\mathbf{s}), \omega_t(\mathbf{r})] = \mathbb{E}[\omega_t(\mathbf{s})\omega_t(\mathbf{r})] = (\mathbf{s})^\intercal\Sigma_\eta(\mathbf{r}).$$

□

2.2 Kernel Decomposition

Next is the key part od the system, which defines the dynamics; the kernelf function, κ. There are a few ways to handle the kernel. One of the most obvious is to expand it out into a spectral decomposition as well;

$$\kappa \approx \sum_i \beta_i \psi(\mathbf{s}, \mathbf{r}).$$

This can allow for a wide range of interestingly shaped kernel functions, but see how these basis functions must now act on $\mathbb{R}^2 \times \mathbb{R}^2$; to get a wide enough space of possible functions, we would likely need many terms of the basis expansion.

A much simpler approach would be to simply parameterise the kernel function, to $\kappa(\mathbf{s}, \mathbf{r}, {}_\kappa)$. We then establish a simple shape for the kernel (e.g. Gaussian) and rely on very few parameters (for example, scale, shape, offsets). The example kernel used in the program is aGaussian kernel;

$$\kappa(\mathbf{s}, \mathbf{r}; \mathbf{m}, a, b) = a \exp\left(-\frac{1}{b}|\mathbf{s} - \mathbf{r} + \mathbf{m}|^2\right)$$

Of course, this kernel lacks spatial dependance. We can add spatial variance back in in a nice way by adding dependance on $\mathbf{s}$ to the parameters, for example, variyng the offset term as $\mathbf{m}(\mathbf{s})$. Of course, now we are back to having entire functions as parameters, but taking the spectral decomposition of the parameters we actually want to be spatially variant seems like a reasonable middle ground (Cressie and Wikle 2015). The actual parameters of such a spatially-variant kernel are then the basis coefficients for the expansion of any spatially variant parameters, as well as any constant parameters.

2.3 IDEM as a linear dynamical system

To recap, we have discretized space in such a way that the Integro-difference model is of a more traditional linear dynamical system form. All that is left is to include our observations in our system.

Lets assume that at each time t there are n_t observations at locations $\mathbf{s}_{1,t}, \ldots, \mathbf{s}_{n_t,t}$. We write the vector of the process at these points as $\mathbf{Y}(t) = (Y(s_{1,t}; t), \ldots, Y(s_{n_t,t}; t))^\intercal$, and, in it's expanded form $\mathbf{Y}_t = \Phi_{tt}$, where $\Phi \in \mathbb{R}^{r \times n_t}$ is

$$\{\Phi_t\}_{i,j} = \phi_i(s_{j,t}).$$

For the covariates, we write the matrix $X_t = (\mathbf{X}(\mathbf{s}_{1,t}), \ldots, \mathbf{X}(\mathbf{s}_{1=n_t,t})^\intercal$. We then have

$$\begin{aligned}
\mathbf{Z}_t &= \Phi\alpha_t + X_t + {}_t, \quad t = 0, 1, \ldots, T, \\
{}_{t+1} &= M_t + {}_t, \quad t = 1, 2, \ldots, T, \\
M &= \int_{\mathscr{D}_s} (\mathbf{s})(\mathbf{s})^\intercal d\mathbf{s} \int_{\mathscr{D}_s^2} (\mathbf{s})\kappa(\mathbf{s}, \mathbf{r}; {}_\kappa)(\mathbf{r})^\intercal d\mathbf{r} d\mathbf{s},
\end{aligned}$$

Writing $\tilde{\mathbf{Z}}_t = \mathbf{Z}_t - X_t$,

$$\begin{aligned}
\tilde{\mathbf{Z}}_t &= \Phi_{tt} + {}_t, \qquad t = 1, 2, \ldots, T, \\
{}_{t+1} &= M_t + {}_t, \qquad t = 0, 1, \ldots, T.
\end{aligned} \tag{2.7}$$

We should also initialise $_0 \sim \mathcal{N}^r(\mathbf{m}_0, \Sigma_0)$, and fix simple distrubtions to the noise terms,

$$\epsilon_{t,i} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\epsilon^2),$$
$$\eta_{t,i} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\eta^2),$$

which are (also) independant in time.

As in, for example, (Wikle and Cressie 1999), this is now in a traditional enough form that the Kalman filter can be applied to filter and compute many necessary quantities for inference, including the marginal likelihood. We can use these quantities in either an EM algorithm or a Bayesian approach. Firstly, we cover the EM algorithm applied to this system.

At most, the parameters to be estimated are

$$= \left({}_\kappa^\intercal, {}^\intercal, \mathbf{m}_0^\intercal, \sigma_\epsilon^2, \sigma_\eta^2, \text{vec}[\Sigma_0] \right),$$

where the vec$[\cdot]$ operator gives the elements of the matrix in a column vector. Of course, in practice, some of these may be estimated outside of the algorithm, fixed, given a much simpler form (e.g. $\Sigma_\eta = \sigma_\eta^2 I_d$), etc.

There are two approaches we can make from here; directly maximising the marginal data likelihood using only the Kalman filter, or maximising the full likelihood with the EM algorithm.

Now (Equation 2.7) is of the very familar linear dynamical system (LDS) type. This is a well-understood problem, and optimal state estimation can be done using the kalman filter and (RTS) smoother.

Chapter 3

Filtering, Forecasting and Maximum Likelihood Estimation

The Kalman filter gives us linear estimates for the distribution of $_r \mid \{Z_t\}_{t=0,\dots,r}$ in any dynamical system like Equation 2.7. For full discussions and proofs of the Kalman filter, see, for example, (Shumway, Stoffer, and Stoffer 2000).

3.1 The Kalman Filter

Firstly, we should establish some notation. Write

$$\begin{aligned} m_{r|s} &= \mathbb{E}[_r \mid \{Z_t\}_{t=0,\dots,s}] \\ P_{r|s} &= \mathbb{V}\text{ar}[_r \mid \{Z_t\}_{t=0,\dots,s}] \\ P_{r,q|s} &= \mathbb{C}\text{ov}[_r, _q \mid \{Z_t\}_{t=0,\dots,s}]. \end{aligned}$$

For the initial terms, $m_{0|0} = m_0$ and $P_{0|0} = \Sigma_0$. For convenience and generality, we write Σ_η and Σ_ϵ for the variance matrices of the process and observations. Note that, if the number of observations change at each time point (for example, due to missing data), then Σ_ϵ should be time variyng; we could either always keep it as uncorrelated so that $\Sigma_\epsilon = \text{diag}(\sigma_\epsilon^2)$, or perhaps put some kind of distance-dependant covariance function to it.

To move the filter forward, that is, given $m_{r|s}$ and $P_{r|s}$, to get $m_{t+1|t+1}$ and $P_{t+1|t+1}$, we first *predict*

$$\begin{aligned} \mathbf{m}_{t+1|t} &= M\mathbf{m}_{t|t} \\ P_{t+1|t} &= MP_{t|t}M^\intercal + \Sigma_\eta, \end{aligned} \tag{3.1}$$

then we add our new information, z_t, adjusted for the *Kalman gain*;

$$\begin{aligned} \mathbf{m}_{t+1|t+1} &= \mathbf{m}_{t+1|t} + K_{t+1}\mathbf{e}_{t+1} \\ P_{t+1|t+1} &= [I - K_{t+1}\Phi_{t+1}]P_{t+1|t} \end{aligned} \tag{3.2}$$

where K_{t+1} is the *Kalman gain*;

$$K_{t+1} = P_{t+1|t}\Phi_{t+1}^\intercal[\Phi_{t+1}P_{t+1|t}\Phi_{t+1}^\intercal + \Sigma_\epsilon]^{-1}, \quad t = 0, \dots, T-1$$

and $\mathbf{e}_{t+1}$ are the *prediction errors*

$$\mathbf{e}_{t+1} = \tilde{\mathbf{z}}_{t+1} - \Phi_{t+1}\mathbf{m}_{t+1|t}, \quad t = 1, \ldots, T$$

Starting with $m_{0|0} = m_0$ and $P_{0|0} = \Sigma_0$, we can then iteratively move across the data to eventually compute $m_{T|T}$ and $P_{T|T}$.

Assuming Gaussian all random variables here are Guassian, this is the optimal mean-square estimators for these quantities, but even outside of the Gaussian case, these are optimal for the class of *linear* operators.

We can compute the marginal data likelihood alongside the kalman fiter using the prediciton errors $\mathbf{e}_t$. These, under the assumptions we have made about η and ϵ being normal, are also normal with zero mean and variance

$$\mathbb{V}\text{ar}[\mathbf{e}_t] = \Sigma_t = \Phi_t P_{t|t-1} \Phi_t^\intercal + \Sigma_\epsilon.$$

Therefore, the log-likelihood at each time is

$$\mathscr{L}(Z \mid) = -\frac{1}{2}\sum \log\det(\Sigma_t()) - \frac{1}{2}\sum \mathbf{e}_t()^\intercal \Sigma_t()^{-1} \mathbf{e}_t() - \frac{n_t}{2}\log(2 * \pi).$$

Summing these across time, we get the log likelihood for all the data.

A simplified example of the kalman filter function, written to be jax compatible, used in the package is this;

```
# TODO: Replace this with a simpler 'naive' implementation
@jax.jit
def kalman_filter(
    m_0: ArrayLike,
    P_0: ArrayLike,
    M: ArrayLike,
    PHI_obs: ArrayLike,
    Sigma_eta: ArrayLike,
    Sigma_eps: ArrayLike,
    ztildes: ArrayLike,  # data matrix, with time across columns
) -> tuple:
    nbasis = m_0.shape[0]
    nobs = ztildes.shape[0]

    @jax.jit
    def step(carry, z_t):
        m_tt, P_tt, _, _, ll, _ = carry

        # predict
        m_pred = M @ m_tt
        P_pred = M @ P_tt @ M.T + Sigma_eta

        # Update

        # Prediction Errors
        eps_t = z_t - PHI_obs @ m_pred
```

```
        Sigma_t = PHI_obs @ P_pred @ PHI_obs.T + Sigma_eps

        # Kalman Gain
        K_t = (
            jnp.linalg.solve(Sigma_t, PHI_obs)
            @ P_pred.T
        ).T

        m_up = m_pred + K_t @ eps_t

        P_up = (jnp.eye(nbasis) - K_t @ PHI_obs) @ P_pred

        # likelihood of epsilon, using cholesky decomposition
        chol_Sigma_t = jnp.linalg.cholesky(Sigma_t)
        z = jax.scipy.linalg.solve_triangular(chol_Sigma_t, eps_t)
        ll_new = ll - jnp.sum(jnp.log(jnp.diag(chol_Sigma_t))
                              ) - 0.5 * jnp.dot(z, z)

        return (m_up, P_up, m_pred, P_pred, ll_new, K_t), (
            m_up,
            P_up,
            m_pred,
            P_pred,
            ll_new,
            K_t,
        )

    carry, seq = jl.scan(
        step,
        (m_0, P_0, m_0, P_0, 0, jnp.zeros((nbasis, nobs))),
        ztildes.T,
    )

    return (carry[4], seq[0], seq[1], seq[2][1:], seq[3][1:], seq[5][1:])
```

For the documentation of the method proveded by the package, see [WORK OUT HOW TO LINK DO PAGES]

3.2 The Information Filter

In some computational scenarios, it is beneficial to work with vectors of consistent dimension. In python jax, the efficient `scan` and `map` operations work only with such operations; JAX has no support for jagged arrays, and traditional for loops with have long compile times when jit-compiled. Although there are some tools in JAX to get around this problem (namely the `jax.tree` functions which allow mapping over PyTrees), scan is still a large problem; since the Kalman filter is, at it's core, a scan-type operation, this causes a large problem when the observation dimension is changing, as is frequent with many spatio-temporal data.

But it is possible to re-write the kalman filter in a way which is compatible with this kind of data. the sometimes called 'information filter' involves transforming the data into a kind of 'information form', which will always have consistent dimension.

The information filter is simply the kalman filter re-written to use the Gaussian distribution's canonical parameters, those being the information vector and the information matrix. If a Gaussian distribution has mean and variance matrix Σ, then the corresponding *information vector* and *information matrix* is $\nu = \Sigma^{-1}\mu$ and $Q = \Sigma^{-1}$, correspondingly.

Theorem 3.2.1 (The Information Filter). The Kalman filter can be rewritten in information form as follows (for example, Khan 2005). Write

$$\begin{aligned} Q_{i|j} &= P_{i|j} \\ {}_{i|j} &= Q_{i|j}\mathbf{m}_{i|j} \end{aligned}$$

and transform the observations into their 'information form', for $t = 1, \ldots, T$

$$\begin{aligned} I_t &= \Phi_t^\intercal \Sigma_\epsilon^{-1} \Phi_t, \\ i_t &= \Phi_t^\intercal \Sigma_\epsilon^{-1} \mathbf{z}_t. \end{aligned} \tag{3.3}$$

The prediction step now becomes

$$\begin{aligned} {}_{t+1|t} &= (I - J_t) M^{-1}{}_{t|t} \\ Q_{t+1|t} &= (I - J_t) S_t \end{aligned}$$

where $S_t = M^{-\intercal} Q_{t|t} M^{-1}$ and $J_t = S_t [S_t + \Sigma_\eta^{-1}]^{-1}$.
Updating is now as simple as adding the information-form observations;

$$\begin{aligned} {}_{t+1|t+1} &= {}_{t+1|t} + i_{t+1} \\ Q_{t+1|t+1} &= Q_{t+1|t} + I_{t+1}. \end{aligned}$$

Proof in Appendix (Section A.2.)

We can see that the information form of the observations (Equation 3.3) will always have the same dimension (that being the process dimension, previously labelled r, the number of basis functions used in the expansion). For our purposes, this means that `jax.lax.scan` will work after we 'informationify' the data, which can be done using `jax.tree.map`. This is implemented in the functions `information_filter` and `information_filter_indep` (for uncorrelated errors).

There are other often cited advantages to filtering in this form. It can be quicker that the traditional form in certain cases, especially when the observation dimension is bigger than the state dimension (since you solve a smaller system of equations with $[S_t + \Sigma_\eta]^{-1}$ in the process dimesnion instead of $[\Phi_t P_{t+1|t} \Phi_t^\intercal + \Sigma_\epsilon]^{-1}$ in the observation dimension) (Assimakis, Adam, and Douladiris 2012).

The other often mentioned advantage is the ability to use a truly vague prior for α_0; that is, we can set Q_0 as the zero matrix, without worriying about an infinite variance matrix. While this is indeed true, it is actually possible to do the same with the Kalman filter by doing the first step analytically, see (Section A.3).

As with the kalman filter, it is also possible to get the data likelihood in-line as well. Again, we would like to stick with things in the state dimension, so working direclty with the prediction errors $\mathbf{e}_t$ should be avoided. Luckily, by multipliying the errors by $\Phi_t^\intercal \Sigma_\epsilon^{-1}$, we can define the 'information errors' ${}_t$;

$$\begin{aligned} {}_t &= \Phi_t^\intercal \Sigma_\epsilon^{-1} \mathbf{e}_t = \Phi_t^\intercal \Sigma_\epsilon^{-1} \tilde{\mathbf{z}}_t - \Phi_t^\intercal \Sigma_\epsilon^{-1} \Phi_t m_{t|t-1} \\ &= i_t - I_t Q^{-1}_{t|t-1} {}_{t|t-1}. \end{aligned}$$

The variance of this quantity is also easy to find;

$$\begin{aligned}
\mathbb{V}\text{ar}[\,_t] &= \Phi_t^\intercal \Sigma_\epsilon^{-1}\, \mathbb{V}\text{ar}[\mathbf{e}_t]\Sigma_\epsilon^{-1}\Phi_t \\
&= \Phi_t^\intercal \Sigma_\epsilon^{-1}[\Phi_t P_{t|t-1}\Phi_t^\intercal + \Sigma_\epsilon]\Sigma_\epsilon^{-1}\Phi_t \\
&= \Phi_t^\intercal \Sigma_\epsilon^{-1}\Phi_t Q_{t|t-1}^{-1}\Phi_t^\intercal \Sigma_\epsilon^{-1}\Phi_t \Phi_t^\intercal \Sigma_\epsilon^{-1}\Phi_t \\
&= I_t Q_{t|t-1}^{-1} I_t^\intercal + I_t =: \Sigma_{\iota,t}.
\end{aligned}$$

Noting that clearly still has mean zero, this allows us once again to compute the log likelihood, this time through

$$\mathscr{L}(z_t \mid) = -\frac{1}{2}\sum \log\det(\Sigma_{\iota,t}()) - \frac{1}{2}\sum {}_t()^\intercal \Sigma_{\iota,t}()^{-1}{}_t() - \frac{r}{2}\log(2 * \pi).$$

3.3 Kalman Smoothers

Beyond the Kalman filters, we can also do Kalman smoothers. That is, filters estimate $\mathbf{m}_{T|T}$ and $P_{T|T}$, but there is use for estimating $\mathbf{m}_t \mid T$ and $P_{t|T}$ for all $t = 0, \ldots, T$.

We can then work backwards from these values using what is known as the *Rauch-Tung-Striebel (RTS) smoother*;

$$\begin{aligned}
\mathbf{m}_{t-1|T} &= \mathbf{m}_{t-1|t-1} + J_{t-1}(\mathbf{m}_{t|T} - \mathbf{m}_{t|t-1}), \\
P_{t-1|T} &= P_{t-1|t-1} + J_{t-1}(P_{t|T} - P_{t|t-1})J_{t-1}^\intercal,
\end{aligned} \tag{3.4}$$

where,

$$J_{t-1} = P_{t-1|t-1}M^\intercal[P_{t|t-1}]^{-1}.$$

We can clearly see, then, that it is crucial to keep the values in Equation 3.1.

We can then also compute the lag-one cross-covariance matrices $P_{t,t-1|T}$ using the *Lag-One Covariance Smoother* (is this what they call the RTS smoother?) From

$$P_{T,T-1|T} = (I - K_T\Phi_T)M\,P_{T-1|T-1},$$

we can compute the lag-one covariances

$$P_{t,t-1|T} = P_{t|t}J_{t-1}^\intercal + J_t[P_{t+1,t|T} - M\,P_{t-1|t-1}]J_{t-1}^\intercal \tag{3.5}$$

These values can be used to implement the expectation-maximisation (EM) algorithm which will be introduced later.

Chapter 4

EM Algorithm (NEEDS A LOT OF WORK, PROBABLY IGNORE FOR NOW)

Instead of the marginal data likelihood, we may instead want to work with the 'full' likelihood, including the unobserved process, $l(\mathbf{z}(1), \dots, \mathbf{z}(T), \mathbf{Y}(1), \dots, \mathbf{Y}(T) \mid)$, or, equivalently, $l(\mathbf{z}(1), \dots, \mathbf{z}(t), (1), \dots, (T) \mid)$. This is difficult to maximise directly, but can be done with the EM algorithm, consisting of two steps, which can be shown to always increase the full likelihood.

Firstly, the E step is to find the function

$$\mathcal{Q}(;') = \mathbb{E}_{\mathbf{Z}(t)\sim p(Z|(t),)}[\log p(Z^{(T)}, A^{(T)}) \mid Z^{(T)}], \tag{4.1}$$

where $Z^{(T)} = \{\mathbf{z}_t\}_{t=0,\dots,T}$, $A^{(T)} = \{_t\}_{t=0,\dots,T}$ and $A^{(T-1)} = \{_t\}_{t=0,\dots,T-1}$. This approximates $\log p_\theta(Z^{(T)}, A^{(T)})$.

Proposition 4.0.1. *We have [NOTE: This may well be wrong in places…]*

$$\begin{aligned}
-2\mathcal{Q}(;') &= \mathbb{E}_{Z^{(T)}\sim p(Z|A^{(T)},')}[\log p(Z^{(T)}, A^{(T)} \mid Z^{(T)} = z^{(T)})] \\
&\overset{c}{=} \sigma_\epsilon^2[\sum_{t=0}^{T} \mathbf{z}_t^\intercal z_t - 2\Phi_t(\sum_{t=1}^{T} \mathbf{z}_t^\intercal \mathbf{m}_{t|T}) - 2(\sum_{t=0}^{T} \mathbf{z}_t^T)X_t \\
&\qquad + \Phi_t^\intercal(\sum_{t=0}^{T} \mathrm{tr}\{P_{t|T} - \mathbf{m}_{t|T}\mathbf{m}_{t|T}^\intercal\})\Phi_t + 2X_t\Phi_t(\sum_{t=0}^{T} \mathbf{m}_{t|T}) + (\sum_{t=1}^{T} X_t^{\intercal\intercal} X_t)] \\
&\quad + \mathrm{tr}\{\Sigma_\eta^{-1}[(\sum_{t=1}^{T} P_{t|T} - m_{t|T}) - 2M(\sum_{t=1}^{T} P_{t,t-1|T} - \mathbf{m}_{t-1,T}\mathbf{m}_{t|T}^\intercal) \\
&\qquad + M(\sum_{t=1}^{T} P_{t-1|T} - \mathbf{m}_{t-1|T}\mathbf{m}_{t-1|T}^\intercal)M^\intercal]\} \\
&\quad + \mathrm{tr}\{\Sigma_0^{-1}[P_{0|T} - m_{0|T}m_{0|T}^\intercal - 2\mathbf{m}_{0|T}\mathbf{m}_0 + \mathbf{m}_0\mathbf{m}_0^\intercal]\} \\
&\quad + \log(\det(\sigma_\epsilon^{2T}\Sigma_\eta^{T+1}\Sigma_0))
\end{aligned} \tag{4.2}$$

Proof 3. See appendix.

In the EM algorithm, we maximise the full likelihood by changing in order to increase (Equation 4.2), which can be shown to guarantee that the Likelihood $L()$ also increases. The idea is then that repeatedly alternating between adjusting to increase Equation 4.2, and then doing the filters and smoothers to obtain new values for $\mathbf{m}_{t|T}$, $\boldsymbol{P}_{t|T}$, and $\boldsymbol{P}_{t,t-1|T}$.

Chapter 5

Algorithm for Maximum Complete-data Likelihood estimation

Overall, our algorithm for Maximum Likelihood estimation is:

1. Set $i = 0$ and take an initial guess for the parameters we are considering, $_0 = _i$
2. Starting from $\mathbf{m}_{0|0} = \mathbf{m}_0$, $P_{0|0} = \Sigma_0$, run the **Kalman Filter** to get $\mathbf{m}_{t|t}$, $P_{t|t}$, and K_t for all t Equation 3.2,
3. Starting from $\mathbf{m}_{T|T}$, $P_{T|T}$, run the **Kalman Smoother** to get $\mathbf{m}_{t|T}$, $P_{t|T}$, and J_t for all t (Equation 3.4),
4. Starting from $P_{T,T-1|T} = (I - K_n A_n) M P_{T-1|T-1}$, run the **Lag-One Smoother** to get $\mathbf{m}_{t,t-1|T}$ and $P_{t,t-1|T}$ for all t Equation 3.5,
5. Use the above values to construct $\mathcal{Q}(;')$ in Equation 4.2,
6. Maximise the function $\mathcal{Q}(;')$ to get a new guess $_{i+1}$, then return to step 2,
7. Stop once a certain criteria is met.

Appendix A

Appendix

A.1 Woodbury's identity

The following two sections will make heavy use of the Woodbury identity.

Lemma A.1.1 (Woodbury's Identity). We have, for conformable matrices A, U, C, V,

$$(A+UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1}+VA^{-1}U)^{-1}VA^{-1}. \tag{A.1}$$

Additionally, we have the variant

$$(A+UCV)^{-1}UC = A^{-1}U(C^{-1}+VA^{-1}U)^{-1}. \tag{A.2}$$

Proof 4. We only prove (Equation A.2), since various proofs of (Equation A.1) are well known (see, for example, the wikipedia page).
Simply multipliying (Equation A.1) by CU, (similar to Khan 2005, although there is an error in their proof)

$$\begin{aligned}(A+UCV)^{-1}UC &= A^{-1}UC - A^{-1}U(C^{-1}+VA^{-1}U)^{-1}VA^{-1}UC \\ &= A^{-1}UC - A^{-1}U(C^{-1}+VA^{-1}U)[(C^{-1}+VA^{-1}U)C - I] \\ &= A^{-1}U(C^{-1}+VA^{-1}U)\end{aligned}$$

as needed. □

A.2 Proof of Theorem 3.2.1

Proof 5. Firstly, for the prediction step, using $S_t = M^{-\intercal}Q_{t|t}M^{-1}$ and $J_t = S_t(\Sigma_\eta^{-1}+S_t)^{-1}$ and the identities Equation A.1 and Equation A.2,

$$\begin{aligned}Q_{t+1|t} = P_{t+1|t}^{-1} &= (MQ_{t|t}^{-1}M^\intercal + \Sigma_\eta)^{-1} \\ &= S_t - J_tS_t = (I-J_t)S_t,\end{aligned}$$

where we used $A = MQ_{t|t}^{-1}M^\intercal$, $C = \Sigma_\eta$ and $U = C = I$ in Equation A.1. Thurthermore,

$$
\begin{aligned}
{}_{t+1|t} &= Q_{t+1|t}\mathbf{m}_{t+1|t} \\
&= Q_{t+1|t}MQ^{-1}_{t|t}\,{}_{t|t} = Q_{t+1|t}(MQ^{-1}_{t|t})_{t|t} \\
&= (I - J_t)M^{-\intercal}Q_{t|t}M^{-1}(MQ^{-1}_{t|t})_{t|t} \\
&= (I - J_t)M^{-\intercal}{}_{t|t}.
\end{aligned}
$$

For the update step,

$$
\begin{aligned}
Q_{t+1|t+1} &= P^{-1}_{t+1|t+1} \\
&= (Q^{-1}_{t+1} - Q^{-1}_{t+1|t}\Phi^{\intercal}_{t+1}[\Phi_{t+1}\Sigma_\epsilon\Phi^{\intercal}_{t+1} + \Sigma_\epsilon]^{-1}\Phi_{t+1}Q^{-1}_{t+1|t})^{-1} \\
&= ((Q_{t+1|t} + \Phi^{\intercal}_{t+1}\Sigma^{-1}_\epsilon\Phi_{t+1})^{-1})^{-1} = Q_{t+1|t} + \Phi^{\intercal}_{t+1}\Sigma^{-1}_\epsilon\Phi_{t+1} \\
&= Q_{t+1|t} + I_{t+1}.
\end{aligned}
$$

Then, writing $\mathbf{m}_{t+1|t+1}$ in terms of $Q_{t+1|t}$ and ${}_{t+1|t}$

$$
\begin{aligned}
\mathbf{m}_{t+1|t+1} &= Q^{-1}_{t+1|t}{}_{t+1|t} - Q^{-1}_{t+1|t}\Phi^{\intercal}_{t+1}[\Phi_{t+1}Q^{-1}_{t+1|t}\Phi^{\intercal}_{t+1} + \Sigma_\epsilon]^{-1}[\tilde{\mathbf{z}}_{t+1} - \Phi_{t+1}Q^{-1}_{t+1|t}{}_{t+1|t}] \\
&= (Q^{-1}_{t+1|t} - Q^{-1}_{t+1|t}\Phi^{\intercal}_{t+1}[\Phi_{t+1}Q^{-1}_{t+1|t}\Phi^{\intercal}_{t+1} + \Sigma_\epsilon]^{-1}\Phi_{t+1}Q^{-1}_{t+1|t})_{t+1|t} \\
&\quad + Q^{-1}_{t+1|t}\Phi^{\intercal}_{t+1}[\Phi_{t+1}Q^{-1}_{t+1|t}\Phi^{\intercal}_{t+1} + \Sigma_\epsilon]^{-1}\tilde{\mathbf{z}}_{t+1} \\
&= [Q_{t+1|t} + I_{t+1}]^{-1}{}_{t+1|t} \\
&\quad + [Q_{t+1|t} + I_{t+1}]^{-1}\Phi_{t+1}\Sigma^{-1}_\epsilon\tilde{\mathbf{z}}_{t+1},
\end{aligned}
$$

and now noting that ${}_{t+1|t+1} = (Q_{t+1|t} + I_{t+1})\mathbf{m}_{t+1|t+1}$, we complete the proof. □

A.3 Truly Vague Prior with the Kalman Filter

It has been stated before that one of the large advantages of the information filter is the ability to use a completely vague prior $Q_0 = 0$. While this is true, it is actually possible to do this in the Kalman filter by 'skipping' the first step (contrary to some sources, such as the wikipedia page as of January 2025).

Theorem A.3.1. In the Kalman Filter (Section 3.1), if we allow $P_0^{-1} = 0$, effectively setting infinite variance, and assuming the propegator matrix M is invertible, we have

$$
\begin{aligned}
\mathbf{m}_{1|1} &= (\Phi^{\intercal}_1\Sigma^{-1}_\epsilon\Phi_1)^{-1}\Phi_1\Sigma^{-1}_\epsilon\tilde{\mathbf{z}}_1, \\
P_{1|1} &= (\Phi^{\intercal}_1\Sigma^{-1}_\epsilon\Phi_1)^{-1}.
\end{aligned}
\tag{A.3}
$$

Therefore, starting with these values then continuing the filter as normal, we can perform the kalman filter with 'infinite' prior variance.
[NOTE: The requirement that M be invertible should be droppable, see the proof below]

Proof 6. Unsuprisingly, the proof is effectively equivalent to proving the information filter and setting $Q_0 = P_0^{-1} = 0$.
For the first predict step (Equation 3.1),

$$\begin{aligned}\mathbf{m}_{1|0} &= M\mathbf{m}_0,\\ P_{1|0} &= MP_0M^\intercal + \Sigma_\eta.\end{aligned}$$

By (Equation A.1),

$$\begin{aligned}P_{1|0}^{-1} &= \Sigma_\eta^{-1} - \Sigma_\eta^{-1}M(P_0^{-1} + M^\intercal\Sigma_\eta^{-1}M)^{-1}M^\intercal\Sigma_\eta^{-1}\\ &= \Sigma_\eta^{-1} - \Sigma_\eta^{-1}M(M^\intercal\Sigma_\eta^{-1}M)^{-1}M^\intercal\Sigma_\eta^{-1}\\ &= \Sigma_\eta^{-1} - \Sigma_\eta^{-1} = 0.\end{aligned}$$

So, moving to the update step (Equation 3.2),

$$\mathbf{m}_{1|1} = M\mathbf{m}_0 + P_{1|0}\Phi_1[\Phi_1P_{1|0}\Phi_1^\intercal + \Sigma_\epsilon]^{-1}(\tilde{\mathbf{z}}_1 - \Phi M\mathbf{m}_0).$$

Applying (Equation A.2) with $A = P_{1|0}^{-1}, U = \Phi_1, V = \Phi_1^\intercal, C = \Sigma_\epsilon^{-1}$,

$$\begin{aligned}\mathbf{m}_{1|1} &= M\mathbf{m}_0 + (P_{1|0}^{-1} + \Phi_1^\intercal\Sigma_\epsilon^{-1}\Phi_1)^{-1}\Phi_1^\intercal\Sigma_\epsilon^{-1}(\tilde{\mathbf{z}}_1 - \Phi_1M\mathbf{m}_0)\\ &= M\mathbf{m}_0 + (\Phi_1^\intercal\Sigma_\epsilon^{-1}\Phi_1)^{-1}\Phi_1^\intercal\Sigma_\epsilon^{-1}\tilde{\mathbf{z}}_1 - (\Phi_1^\intercal\Sigma_\epsilon^{-1}\Phi_1)^{-1}\Phi_1^\intercal\Sigma_\epsilon^{-1}\Phi_1M\mathbf{m}_0\\ &= (\Phi_1^\intercal\Sigma_\epsilon^{-1}\Phi_1)^{-1}\Phi_1^\intercal\Sigma_\epsilon^{-1}\tilde{\mathbf{z}}_1.\end{aligned}$$

For the variance, we apply the (Equation A.1) with $A = P_{1|0}^{-1}, U = \Phi_1^\intercal, V = \Phi_1, C = \Sigma_\epsilon^{-1}$,

$$\begin{aligned}P_{1|1} &= (I - P_{1|0}\Phi_1^\intercal[\Sigma_\epsilon + \Phi_1^\intercal P_{1|0}\Phi_1]^{-1}\Phi_1)P_{1|0}\\ &= (P_{1|0}^{-1} + \Phi_1^\intercal\Sigma_\epsilon^{-1}\Phi_1)^{-1}\\ &= (\Phi_1^\intercal\Sigma_\epsilon^{-1}\Phi_1)^{-1},\end{aligned}$$

as needed. □

It is worth noting that (Equation A.3) seems to make a lot of sense; namely, we expect the estimate for $\mathbf{m}_0$ to look like a correlated least squares-type estimator like this.

Assimakis, Nicholas, Maria Adam, and Anargyros Douladiris. 2012. "Information Filter and Kalman Filter Comparison: Selection of the Faster Filter." In *Information Engineering*, 2:1–5. 1.

Cressie, Noel, and Christopher K Wikle. 2015. *Statistics for Spatio-Temporal Data*. John Wiley & Sons.

Dewar, Michael, Kenneth Scerri, and Visakan Kadirkamanathan. 2008. "Data-Driven Spatio-Temporal Modeling Using the Integro-Difference Equation." *IEEE Transactions on Signal Processing* 57 (1): 83–91.

Khan, Mohammad Emtiyaz. 2005. "Matrix Inversion Lemma and Information Filter." *Honeywell Techonology Solutions Lab, Bangalore, India*.

Liu, Xiao, Kyongmin Yeo, and Siyuan Lu. 2022. "Statistical Modeling for Spatio-Temporal Data from Stochastic Convection-Diffusion Processes." *Journal of the American Statistical Association* 117 (539): 1482–99.

Shumway, Robert H, David S Stoffer, and David S Stoffer. 2000. *Time Series Analysis and Its Applications*. Vol. 3. Springer.

Wikle, Christopher K, and Noel Cressie. 1999. "A Dimension-Reduced Approach to Space-Time Kalman Filtering." *Biometrika* 86 (4): 815–29.

Wikle, Christopher K, Andrew Zammit-Mangion, and Noel Cressie. 2019. *Spatio-Temporal Statistics with r*. CRC Press.

Zammit-Mangion, Andrew. 2022. *IDE: Integro-Difference Equation Spatio-Temporal Models*. https://CRAN.R-project.org/package=IDE.