Notes on computing the Fisher Information matrix for MARSS models. Part IV Implementing the Recursion in Harvey 1989

EE Holmes

2017-05-31

This is part of a series on computing the Fisher Information for Multivariate Autoregressive State-Space Models. Part I: Background, Part II: Louis 1982, Part III: Harvey 1989, Background, Part IV: Harvey 1989, Implementation.

Part III Introduced the approach of Harvey (1989) for computing the expected and observed Fisher Information matrices by using the prediction error form of the log-likelihood function. Here I show the Harvey (1989) recursion on page 143 for computing the derivatives in his equations.

Derivatives needed for the 2nd derivative of the conditional log-likelihood

Equations 3.4.66 and 3.4.69 in Harvey (1989) have first and second derivatives of v_t and F_t with respect to θ_i and θ_j. These in turn involve derivatives of the parameter matrices and of $\tilde{x}_{t|t}$ and $\tilde{V}_{t|t}$. Harvey shows all the first derivatives, and it is easy to compute the second derivatives by taking the derivatives of the first.

The basic idea of the recursion is simple, if a bit tedious.

- First we set up matrices for all the first derivatives of the parameters.

- Then starting from t=1 and working forward, we will do the recursion (described below) for all θ_i and we store the first derivatives of v_t, F_t, $\tilde{x}_{t|t}$ and $\tilde{V}_{t|t}$ with respect to θ_i.

- Then we go through the parameter vector a second time, to get all the second derivatives with respect to θ_i and θ_j.

- We input the first and second derivatives of v_t and F_t into equations 3.4.66 and 3.4.69 to get the observed Fisher Information at time t and add to the Fisher Information from the previous time step. The Fisher Information matrix is symmetric, so we can use an outer loop from θ_1 to θ_p (p is the number of parameters) and an inner loop from θ_i to θ_p. That will be $p(p-1)/2$ loops for each time step.

The end result with be the observed Fisher Information matrix using equation 3.4.66 and using 3.4.69.

Outline of the loops in the recursion

This is a forward recursion starting at t=1. We will save the previous time step's $\partial v_t / \theta_i$ and $\partial F_t / \theta_i$. That will be $p \times 2$ ($n \times 1$) vectors and $n \times 2$ ($n \times n$) matrices. We do not need to store all the previous time steps since this is a one-pass recursion unlike the Kalman smoother, which is forward-backward.

Set-up

Number of parameters = p.

- Create Iijt and oIijt which are p x p matrices.
- Create dvit which is a n x p matrix. n Innovations and p θ_i.

- Create d2vijt which is a n x p x p array. n Innovations and p θ_i.
- Create dFit which is a n x n x p array. n x n Sigma matrix and p θ_i.
- Create d2Fijt which is a n x n x p x p array. n x n Sigma matrix and p θ_i.

Outer loop from t=1 to t=T.

Inner loop over all MARSS parameters: x0, V0, Z, a, R, B, u, Q. This is `par$Z`, e.g., and is a vector of the estimated parameters elements in Z.

Inner loop over parameters in parameter matrix, so, e.g. over the rows in the column vector `par$Z`.

Keep track of what parameter element I am on via p counter.

The form of the parameter derivatives

Within the recursion, we have terms like, $\partial M/\partial \theta_i$, where M means some parameter matrix. We can write M as $vec(M) = f + D\theta_m$, where θ_m is the vector of parameters that appear in M. This is the way that matrices are written in Holmes (2010). So

$$\begin{bmatrix} 2a+c & b \\ b & a+1 \end{bmatrix} \tag{1}$$

is written in vec form as

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 2 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{2}$$

The derivative of this with respect to $\theta_i = a$ is

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 2 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{3}$$

So in MARSS, $\partial M/\partial \theta_i$ would be

```
dthetai=matrix(0,ip,1); dthetai[i,]=1 #set up the d theta_i bit.
dM=unvec(f+D%*%dthetai,dim(M)) #only needed if M is matrix
```

The reason is that MARSS allows any linear constraint of the form $\alpha + \beta a + \beta_2 b$, etc. The vec form allows me to work with a generic linear constraint without having to know the exact form of that constraint. The model and parameters are all specified in vec form with f, D, and p matrices (lower case = column vector).

The second derivative of a parameter matrix with respect to θ_j is always 0 since 3 has no parameters in it, only constants.

Derivatives of the innovations and variance of innovations

Equation 3.4.71b in Harvey shows $\partial v_t/\partial \theta_i$. Store result in dvit[,p].

$$\frac{\partial v_t}{\partial \theta_i} = -Z_t \frac{\partial \tilde{x}_{t|t-1}}{\partial \theta_i} - \frac{Z_t}{\partial \theta_i} \tilde{x}_{t|t-1} - \frac{\partial a_t}{\partial \theta_i} \tag{4}$$

$\tilde{x}_{t|t-1}$ is the one-step ahead prediction covariance output from the Kalman filter, and in MARSSkf is `xtt1[,t]`. Next, use equation 3.4.73, to get $\partial F_t/\partial\theta_i$. Store result in `dFit[„p]`.

$$\frac{\partial F_t}{\partial\theta_i} = \frac{\partial Z_t}{\partial\theta_i}\tilde{V}_{t|t-1}Z_t^\top + Z_t\frac{\partial\tilde{V}_{t|t-1}}{\partial\theta_i}Z_t^\top + Z_t\tilde{V}_{t|t-1}\frac{\partial Z_t^\top}{\partial\theta_i} + \frac{\partial(H_t R_t H_t^\top)}{\partial\theta_i} \tag{5}$$

$\tilde{V}_{t|t-1}$ is the one-step ahead prediction covariance output from the Kalman filter, and in MARSSkf is denoted `Vtt1[„t]`.

Recursion for derivatives of states and variance of states

If t=1

- **Case 1**. $\pi = x_0$ is treated as a parameter and $V_0 = 0$. For any θ_i that is not in π, Z or a, $\partial v_1/\partial\theta_i = 0$. For any θ_i that is not in Z or R, $\partial F_1/\partial\theta_i = 0$ (a n x n matrix of zeros).

  From equation 3.4.73a:

  $$\frac{\partial\tilde{x}_{1|0}}{\partial\theta_i} = \frac{\partial B_1}{\partial\theta_i}\pi + B_1\frac{\partial\pi}{\partial\theta_i} + \frac{\partial u_t}{\partial\theta_i} \tag{6}$$

  From equation 3.4.73b and using $V_0 = 0$:

  $$\frac{\partial\tilde{V}_{1|0}}{\partial\theta_i} = \frac{\partial B_1}{\partial\theta_i}V_0 B_1^\top + B_1\frac{\partial V_0}{\partial\theta_i}B_1^\top + B_1 V_0\frac{\partial B_1^\top}{\partial\theta_i} + \frac{\partial(G_t Q_t G_t^\top)}{\partial\theta_i} = \frac{\partial(G_t Q_t G_t^\top)}{\partial\theta_i} \tag{7}$$

- **Case 2**. $\pi = x_{1|0}$ is treated as a parameter and $V_{1|0} = 0$.

  $$\frac{\partial\tilde{x}_{1|0}}{\partial\theta_i} = \frac{\partial\pi}{\partial\theta_i} \text{ and } \partial V_{1|0}/\partial\theta_i = 0. \tag{8}$$

- **Case 3**. x_0 is specified by a fixed prior. $x_0 = \pi$ and $V_0 = \Lambda$. The derivatives of these are 0, because they are fixed.

  From equation 3.4.73a and using $x_0 = \pi$ and $\partial\pi/\partial\theta_i = 0$:

  $$\frac{\partial\tilde{x}_{1|0}}{\partial\theta_i} = \frac{\partial B_1}{\partial\theta_i}\pi + B_1\frac{\partial\pi}{\partial\theta_i} + \frac{\partial u_t}{\partial\theta_i} = \frac{\partial B_1}{\partial\theta_i}\pi + \frac{\partial u_t}{\partial\theta_i} \tag{9}$$

  From equation 3.4.73b and using $V_0 = \Lambda$ and $\partial\Lambda/\partial\theta_i = 0$:

  $$\frac{\partial\tilde{V}_{1|0}}{\partial\theta_i} = \frac{\partial B_1}{\partial\theta_i}V_0 B_1^\top + B_1\frac{\partial V_0}{\partial\theta_i}B_1^\top + B_1 V_0\frac{\partial B_1^\top}{\partial\theta_i} + \frac{\partial(G_t Q_t G_t^\top)}{\partial\theta_i} = \frac{\partial B_1}{\partial\theta_i}\Lambda B_1^\top + B_1\Lambda\frac{\partial B_1^\top}{\partial\theta_i} + \frac{\partial(G_t Q_t G_t^\top)}{\partial\theta_i} \tag{10}$$

- **Case 4**. $x_{1|0}$ is specified by a fixed prior. $x_{1|0} = \pi$ and $V_{1|0} = \Lambda$. $\partial V_{1|0}/\partial\theta_i = 0$ and $\partial x_{1|0}/\partial\theta_i = 0$.

- **Case 5**. Estimate V_0 or $V_{1|0}$. That is unstable (per Harvey 1989, somewhere). I do not allow that in the MARSS package.

When coding this recursion, I will loop though the MARSS parameters (x0, V, Z, a, R, B, u, Q) and within that loop, loop through the individual parameters within the parameter vector. So say Q is diagonal and unequal. It has m variance parameters, and I'll loop through each.

Now we have $\frac{\partial\tilde{x}_{1|0}}{\partial\theta_i}$ and $\frac{\partial\tilde{V}_{1|0}}{\partial\theta_i}$ for $t = 1$ and we can proceed.

If t>1

The derivative of $\tilde{x}_{t|t-1}$ is (3.4.73a in Harvey)

$$\frac{\partial \tilde{x}_{t|t-1}}{\partial \theta_i} = \frac{\partial B_t}{\partial \theta_i} \tilde{x}_{t-1|t-1} + B_t \frac{\partial \tilde{x}_{t-1|t-1}}{\partial \theta_i} + \frac{\partial u_t}{\partial \theta_i} \tag{11}$$

Then we take the derivative of this to get the second partial derivative.

$$\frac{\partial^2 \tilde{x}_{t|t-1}}{\partial \theta_i \partial \theta_j} = \frac{\partial^2 B_t}{\partial \theta_i \partial \theta_j} \tilde{x}_{t-1|t-1} + \frac{\partial B_t}{\partial \theta_i} \frac{\partial \tilde{x}_{t-1|t-1}}{\partial \theta_j} + \frac{\partial B_t}{\partial \theta_j} \frac{\partial \tilde{x}_{t-1|t-1}}{\partial \theta_i} + B_t \frac{\partial^2 \tilde{x}_{t-1|t-1}}{\partial \theta_i \partial \theta_j} + \frac{\partial^2 u_t}{\partial \theta_i \partial \theta_j} \tag{12}$$

$$= \frac{\partial B_t}{\partial \theta_i} \frac{\partial \tilde{x}_{t-1|t-1}}{\partial \theta_j} + \frac{\partial B_t}{\partial \theta_j} \frac{\partial \tilde{x}_{t-1|t-1}}{\partial \theta_i} + B_t \frac{\partial^2 \tilde{x}_{t-1|t-1}}{\partial \theta_i \partial \theta_j} \tag{13}$$

In the equations, $\tilde{x}_{t|t}$ is output by the Kalman filter. In MARSSkf, it is called xtt[,t]. $\tilde{x}_{t-1|t-1}$ would be called xtt[,t-1]. The derivatives of $\tilde{x}_{t-1|t-1}$ is from the next part of the recursion (below).

The derivative of $\tilde{V}_{t|t-1}$ is (3.4.73b in Harvey)

$$\frac{\partial \tilde{V}_{t|t-1}}{\partial \theta_i} = \frac{\partial B_t}{\partial \theta_i} \tilde{V}_{t-1|t-1} B_t^\top + B_t \frac{\partial \tilde{V}_{t-1|t-1}}{\partial \theta_i} B_t^\top + B_t \tilde{V}_{t-1|t-1} \frac{\partial B_t^\top}{\partial \theta_i} + \frac{\partial (G_t Q_t G_t^\top)}{\partial \theta_i} \tag{14}$$

The second derivative of $\tilde{V}_{t|t-1}$ is obtained by taking the derivative of 14 and eliminating any second derivatives of parameters:

$$\frac{\partial^2 \tilde{V}_{t|t-1}}{\partial \theta_i \partial \theta_j} = \frac{\partial B_t}{\partial \theta_i} \frac{\tilde{V}_{t-1|t-1}}{\partial \theta_j} B_t^\top + \frac{\partial B_t}{\partial \theta_i} \tilde{V}_{t-1|t-1} \frac{\partial B_t^\top}{\partial \theta_j} + \frac{\partial B_t}{\partial \theta_j} \frac{\partial \tilde{V}_{t-1|t-1}}{\partial \theta_i} B_t^\top + B_t \frac{\partial^2 \tilde{V}_{t-1|t-1}}{\partial \theta_i \partial \theta_j} B_t^\top + \tag{15}$$

$$B_t \frac{\partial \tilde{V}_{t-1|t-1}}{\partial \theta_i} \frac{\partial B_t^\top}{\partial \theta_j} + \frac{\partial B_t}{\partial \theta_j} \tilde{V}_{t-1|t-1} \frac{\partial B_t^\top}{\partial \theta_i} + B_t \frac{\tilde{V}_{t-1|t-1}}{\partial \theta_j} \frac{\partial B_t^\top}{\partial \theta_i} \tag{16}$$

In the derivatives, $\tilde{V}_{t|t}$ is output by the Kalman filter. In MARSSkf, it is called Vtt[,t]. $\tilde{V}_{t-1|t-1}$ would be called Vtt[,t-1]. The derivatives of $\tilde{V}_{t-1|t-1}$ is from the rest of the recursion (below).

Rest of the recursion equations are the same for all t.

From equation 3.4.74a:

$$\frac{\partial \tilde{x}_{t|t}}{\partial \theta_i} = \frac{\partial \tilde{x}_{t|t-1}}{\partial \theta_i} + \frac{\partial \tilde{V}_{t|t-1}}{\partial \theta_i} Z_t^\top F_t^{-1} v_t + \tilde{V}_{t|t-1} \frac{\partial Z_t^\top}{\partial \theta_i} F_t^{-1} v_t - \tilde{V}_{t|t-1} Z_t^\top F_t^{-1} \frac{\partial F_t}{\partial \theta_i} F_t^{-1} v_t + \tilde{V}_{t|t-1} Z_t^\top F_t^{-1} \frac{\partial v_t}{\partial \theta_i} \tag{17}$$

$\tilde{V}_{t|t-1}$ is output by the Kalman filter. In MARSSkf, it is called Vtt1[,t]. v_t are the innovations. In MARSSkf, they are called Innov[,t].

From equation 3.4.74b:

$$\frac{\partial \tilde{V}_{t|t}}{\partial \theta_i} = \frac{\partial \tilde{V}_{t|t-1}}{\partial \theta_i} - \frac{\partial \tilde{V}_{t|t-1}}{\partial \theta_i} Z_t^\top F_t^{-1} Z_t \tilde{V}_{t|t-1} - \tilde{V}_{t|t-1} \frac{\partial Z_t^\top}{\partial \theta_i} F_t^{-1} Z_t \tilde{V}_{t|t-1} + \tilde{V}_{t|t-1} Z_t^\top F_t^{-1} \frac{\partial F_t}{\partial \theta_i} F_t^{-1} Z_t \tilde{V}_{t|t-1} -$$
$$\tilde{V}_{t|t-1} Z_t^\top F_t^{-1} \frac{\partial Z_t}{\partial \theta_i} \tilde{V}_{t|t-1} - \tilde{V}_{t|t-1} Z_t^\top F_t^{-1} Z_t \frac{\partial \tilde{V}_{t|t-1}}{\partial \theta_i}$$

(18)

- Repeat for next element in parameter matrix.
- Repeat for parameter matrix at time t.
    - Loop over i = 1 to p.
    - Loop over j = i to p.
        * Compute $I_{ij}(\theta)$ and add to previous time step. This is equation 3.4.69 with expectation dropped. Store in `Iij[i,j]` and `Iij[j,i]`.

$$I_{ij}(\theta)_t = I_{ji}(\theta)_t = \frac{1}{2} \left[tr \left[F_t^{-1} \frac{\partial F_t}{\partial \theta_i} F_t^{-1} \frac{\partial F_t}{\partial \theta_j} \right] \right] + \left(\frac{\partial v_t}{\partial \theta_i} \right)^\top F_t^{-1} \frac{\partial v_t}{\partial \theta_j}$$

(19)

        * Add this on to previous one to get new $I_{ij}(\theta)$:

$$I_{ij}(\theta) = I_{ij}(\theta) + I_{ij}(\theta)_t$$

(20)

    - Repeat for next j.
    - Repeat for next i.
- Repeat for next t.

At the end, $I_{ij}(\theta)$ is the observed Fisher Information Matrix.

Note that Q and R do not appear in $\partial v_t / \partial \theta_i$, but all the other parameters do appear. So the second term in $I_{ij}(\theta)$ is always zero between Q and R and any other parameters. In the second term, u and a do not appear, but every other terms do appear. So the first term in $I_{ij}(\theta)$ is always zero between u and a and any other parameters. This means that there is always zero covariance between u or a and Q or R. But this will not be the case between Q or R and B or Z.

Part of the motivation of implementing the Harvey (1989) recursion is that currently in MARSS, I use a numerical estimate of the Fisher Information matrix by using one of R's functions to return the Hessian. But it often returns errors. I might improve it if I constrained it. If I am only estimating u, a, Q and R, I could do a two-step process. Get the Hessian holding the variances at the MLEs and then repeat with u and a at the MLEs.