Rob J Hyndman

State space models

3: ARIMA and RegARMA models, and dlm

Outline

Linear Gaussian SS models

$$
\begin{aligned}
\text{Observation equation} \quad & y_t = \boldsymbol{f}'\boldsymbol{x}_t + \varepsilon_t \\
\text{State equation} \quad & \boldsymbol{x}_t = \boldsymbol{G}\boldsymbol{x}_{t-1} + \boldsymbol{w}_t
\end{aligned}
$$

- State vector $\boldsymbol{x}_t$ of length p
- $\boldsymbol{G}$ a $p \times p$ matrix, $\boldsymbol{f}$ a vector of length p
- $\varepsilon_t \sim \text{NID}(0, \sigma^2)$, $\quad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W})$.

Outline

1. **ARIMA models in state space form**
2. RegARMA models in state space form
3. The dlm package for R
4. MLE using the dlm package
5. Filtering, smoothing and forecasting using the dlm package
6. Final remarks

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

Then

$$y_t = [1 \;\; 0]\boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & \phi_2 \\ 1 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Now in state space form

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \mathrm{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

Then

$$y_t = [1 \;\; 0]\boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & \phi_2 \\ 1 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Now in state space form
- We can use Kalman filter to compute likelihood and forecasts.

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

Then

$$y_t = [1 \;\; 0]\boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & \phi_2 \\ 1 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Now in state space form
- We can use Kalman filter to compute likelihood and forecasts.

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

Then

$$y_t = [1 \;\; 0]\boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & \phi_2 \\ 1 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Now in state space form
- We can use Kalman filter to compute likelihood and forecasts.

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

Then

$$y_t = [1 \ \ 0]\boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & \phi_2 \\ 1 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Now in state space form
- We can use Kalman filter to compute likelihood and forecasts.

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Alternative formulation

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ \phi_2 y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 \\ \phi_2 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Alternative state space form

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Alternative formulation

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ \phi_2 y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 \\ \phi_2 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Alternative state space form
- We can use Kalman filter to compute likelihood and forecasts.

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Alternative formulation

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ \phi_2 y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 \\ \phi_2 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Alternative state space form
- We can use Kalman filter to compute likelihood and forecasts.

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Alternative formulation

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ \phi_2 y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 \\ \phi_2 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Alternative state space form
- We can use Kalman filter to compute likelihood and forecasts.

ARMA models in state space form

AR(2) model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Alternative formulation

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ \phi_2 y_{t-1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 \\ \phi_2 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

- Alternative state space form
- We can use Kalman filter to compute likelihood and forecasts.

ARMA models in state space form

AR(p) model

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-p+1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \\ \vdots \\ 0 \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & \phi_2 & \dots & \phi_{p-1} & \phi_p \\ 1 & 0 & \dots & 0 & 0 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

ARMA models in state space form

AR(p) model

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ y_{t-1} \\ \vdots \\ y_{t-p+1} \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ 0 \\ \vdots \\ 0 \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & \phi_2 & \dots & \phi_{p-1} & \phi_p \\ 1 & 0 & \dots & 0 & 0 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

ARMA models in state space form

ARMA(1, 1) model

$$y_t = \phi y_{t-1} + \theta e_{t-1} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ \theta e_t \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ \theta e_t \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi & 1 \\ 0 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

ARMA models in state space form

ARMA(1, 1) model

$$y_t = \phi y_{t-1} + \theta e_{t-1} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Let $\boldsymbol{x}_t = \begin{bmatrix} y_t \\ \theta e_t \end{bmatrix}$ and $\boldsymbol{w}_t = \begin{bmatrix} e_t \\ \theta e_t \end{bmatrix}$.

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi & 1 \\ 0 & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$

ARMA models in state space form

ARMA(p,q) model

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t$$

Let $r = \max(p, q+1)$, $\theta_i = 0, q < i \le r$, $\phi_j = 0, p < j \le r$.

$$y_t = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 & 0 & \dots & 0 \\ \phi_2 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \phi_{r-1} & 0 & \dots & 0 & 1 \\ \phi_r & 0 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} 1 \\ \theta_1 \\ \vdots \\ \theta_{r-1} \end{bmatrix} e_t$$

The `arima` function in R is implemented using this formulation.

ARMA models in state space form

ARMA(p,q) model

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t$$

Let $r = \max(p, q+1)$, $\quad \theta_i = 0,\, q < i \le r, \quad \phi_j = 0,\, p < j \le r$.

$$y_t = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 & 0 & \dots & 0 \\ \phi_2 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \phi_{r-1} & 0 & \dots & 0 & 1 \\ \phi_r & 0 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} 1 \\ \theta_1 \\ \vdots \\ \theta_{r-1} \end{bmatrix} e_t$$

The `arima` function in R is implemented using this formulation.

ARMA models in state space form

ARMA(p, q) model

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t$$

Let $r = \max(p, q+1)$, $\theta_i = 0, q < i \le r$, $\phi_j = 0, p < j \le r$.

$$y_t = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 & 0 & \dots & 0 \\ \phi_2 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \phi_{r-1} & 0 & \dots & 0 & 1 \\ \phi_r & 0 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} 1 \\ \theta_1 \\ \vdots \\ \theta_{r-1} \end{bmatrix} e_t$$

The `arima` function in R is implemented using this formulation.

ARMA models in state space form

ARMA(p, q) model

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t$$

Let $r = \max(p, q+1)$, $\quad \theta_i = 0, q < i \leq r, \quad \phi_j = 0, p < j \leq r.$

$$y_t = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_t$$

$$\boldsymbol{x}_t = \begin{bmatrix} \phi_1 & 1 & 0 & \dots & 0 \\ \phi_2 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \phi_{r-1} & 0 & \dots & 0 & 1 \\ \phi_r & 0 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x}_{t-1} + \begin{bmatrix} 1 \\ \theta_1 \\ \vdots \\ \theta_{r-1} \end{bmatrix} e_t$$

The `arima` function in R is implemented using this formulation.

Outline

RegARMA models in state space form

Linear regression with AR(2) error

$$y_t = \alpha + \beta z_t + n_t$$

$$n_t = \phi_1 n_{t-1} + \phi_2 n_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Regression model

$$y_t = [1,\ z_t]\boldsymbol{x}_t + n_t \qquad \boldsymbol{x}_t = [\alpha,\ \beta]'$$

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1}$$

AR(2) model

$$n_t = [1,\ 0]\boldsymbol{f}_t \qquad \boldsymbol{f}_t = [n_t,\ \phi_2 n_{t-1}]'$$

$$\boldsymbol{f}_t = \begin{bmatrix} \phi_1 & 1 \\ \phi_2 & 0 \end{bmatrix} \boldsymbol{f}_{t-1} + \begin{bmatrix} e_t \\ 0 \end{bmatrix}$$

RegARMA models in state space form

Linear regression with AR(2) error

$$y_t = \alpha + \beta z_t + n_t$$

$$n_t = \phi_1 n_{t-1} + \phi_2 n_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Regression model

$$y_t = [1,\ z_t]\boldsymbol{x}_t + n_t \qquad \boldsymbol{x}_t = [\alpha,\ \beta]'$$

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1}$$

AR(2) model

$$n_t = [1,\ 0]\boldsymbol{f}_t \qquad \boldsymbol{f}_t = [n_t,\ \phi_2 n_{t-1}]'$$

$$\boldsymbol{f}_t = \begin{bmatrix} \phi_1 & 1 \\ \phi_2 & 0 \end{bmatrix} \boldsymbol{f}_{t-1} + \begin{bmatrix} e_t \\ 0 \end{bmatrix}$$

RegARMA models in state space form

Linear regression with AR(2) error

$$y_t = \alpha + \beta z_t + n_t$$

$$n_t = \phi_1 n_{t-1} + \phi_2 n_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, \sigma^2)$$

Combined state space model

$$y_t = [1,\ z_t,\ 1,\ 0]\boldsymbol{x}_t \qquad\qquad \boldsymbol{x}_t = [\alpha,\ \beta,\ n_t,\ \phi_2 n_{t-1}]'$$

$$\boldsymbol{x}_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \phi_1 & 1 \\ 0 & 0 & \phi_2 & 0 \end{bmatrix} \boldsymbol{f}_{t-1} + \begin{bmatrix} 0 \\ 0 \\ e_t \\ 0 \end{bmatrix}$$

RegARMA models in state space form

- Any two state models can be combined:

$$y_t = \boldsymbol{f}_1'\boldsymbol{x}_t + \boldsymbol{f}_2'\boldsymbol{z}_t + \varepsilon_t$$
$$\boldsymbol{x}_t = \boldsymbol{G}_1\boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$
$$\boldsymbol{z}_t = \boldsymbol{G}_2\boldsymbol{z}_{t-1} + \boldsymbol{u}_t$$

$$y_t = \begin{bmatrix}\boldsymbol{f}_1' & \boldsymbol{f}_2'\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_t \\ \boldsymbol{z}_t\end{bmatrix} + \varepsilon_t$$
$$\begin{bmatrix}\boldsymbol{x}_t \\ \boldsymbol{z}_t\end{bmatrix} = \begin{bmatrix}\boldsymbol{G}_1 & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{G}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_{t-1} \\ \boldsymbol{z}_{t-1}\end{bmatrix} + \begin{bmatrix}\boldsymbol{w}_t \\ \boldsymbol{u}_t\end{bmatrix}$$

- So we can take a modular approach to defining state space models.
- This is implemented in the **dlm** package in R.

RegARMA models in state space form

- Any two state models can be combined:

$$y_t = \boldsymbol{f}_1'\boldsymbol{x}_t + \boldsymbol{f}_2'\boldsymbol{z}_t + \varepsilon_t$$
$$\boldsymbol{x}_t = \boldsymbol{G}_1\boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$
$$\boldsymbol{z}_t = \boldsymbol{G}_2\boldsymbol{z}_{t-1} + \boldsymbol{u}_t$$

$$y_t = \begin{bmatrix}\boldsymbol{f}_1' & \boldsymbol{f}_2'\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_t\\ \boldsymbol{z}_t\end{bmatrix} + \varepsilon_t$$
$$\begin{bmatrix}\boldsymbol{x}_t\\ \boldsymbol{z}_t\end{bmatrix} = \begin{bmatrix}\boldsymbol{G}_1 & \boldsymbol{O}\\ \boldsymbol{O} & \boldsymbol{G}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_{t-1}\\ \boldsymbol{z}_{t-1}\end{bmatrix} + \begin{bmatrix}\boldsymbol{w}_t\\ \boldsymbol{u}_t\end{bmatrix}$$

- So we can take a modular approach to defining state space models.
- This is implemented in the **dlm** package in R.

RegARMA models in state space form

- Any two state models can be combined:

$$y_t = \boldsymbol{f}_1'\boldsymbol{x}_t + \boldsymbol{f}_2'\boldsymbol{z}_t + \varepsilon_t$$
$$\boldsymbol{x}_t = \boldsymbol{G}_1\boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$
$$\boldsymbol{z}_t = \boldsymbol{G}_2\boldsymbol{z}_{t-1} + \boldsymbol{u}_t$$

$$y_t = \begin{bmatrix}\boldsymbol{f}_1' & \boldsymbol{f}_2'\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_t \\ \boldsymbol{z}_t\end{bmatrix} + \varepsilon_t$$
$$\begin{bmatrix}\boldsymbol{x}_t \\ \boldsymbol{z}_t\end{bmatrix} = \begin{bmatrix}\boldsymbol{G}_1 & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{G}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_{t-1} \\ \boldsymbol{z}_{t-1}\end{bmatrix} + \begin{bmatrix}\boldsymbol{w}_t \\ \boldsymbol{u}_t\end{bmatrix}$$

- So we can take a modular approach to defining state space models.
- This is implemented in the **dlm** package in R.

RegARMA models in state space form

- Any two state models can be combined:

$$y_t = \boldsymbol{f}_1'\boldsymbol{x}_t + \boldsymbol{f}_2'\boldsymbol{z}_t + \varepsilon_t$$
$$\boldsymbol{x}_t = \boldsymbol{G}_1\boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$
$$\boldsymbol{z}_t = \boldsymbol{G}_2\boldsymbol{z}_{t-1} + \boldsymbol{u}_t$$

$$y_t = \begin{bmatrix}\boldsymbol{f}_1' & \boldsymbol{f}_2'\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_t \\ \boldsymbol{z}_t\end{bmatrix} + \varepsilon_t$$
$$\begin{bmatrix}\boldsymbol{x}_t \\ \boldsymbol{z}_t\end{bmatrix} = \begin{bmatrix}\boldsymbol{G}_1 & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{G}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_{t-1} \\ \boldsymbol{z}_{t-1}\end{bmatrix} + \begin{bmatrix}\boldsymbol{w}_t \\ \boldsymbol{u}_t\end{bmatrix}$$

- So we can take a modular approach to defining state space models.
- This is implemented in the **dlm** package in R.

RegARMA models in state space form

- Any two state models can be combined:

$$y_t = \boldsymbol{f}_1'\boldsymbol{x}_t + \boldsymbol{f}_2'\boldsymbol{z}_t + \varepsilon_t$$
$$\boldsymbol{x}_t = \boldsymbol{G}_1\boldsymbol{x}_{t-1} + \boldsymbol{w}_t$$
$$\boldsymbol{z}_t = \boldsymbol{G}_2\boldsymbol{z}_{t-1} + \boldsymbol{u}_t$$

$$y_t = \begin{bmatrix}\boldsymbol{f}_1' & \boldsymbol{f}_2'\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_t \\ \boldsymbol{z}_t\end{bmatrix} + \varepsilon_t$$
$$\begin{bmatrix}\boldsymbol{x}_t \\ \boldsymbol{z}_t\end{bmatrix} = \begin{bmatrix}\boldsymbol{G}_1 & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{G}_2\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_{t-1} \\ \boldsymbol{z}_{t-1}\end{bmatrix} + \begin{bmatrix}\boldsymbol{w}_t \\ \boldsymbol{u}_t\end{bmatrix}$$

- So we can take a modular approach to defining state space models.
- This is implemented in the **dlm** package in R.

Outline

1 ARIMA models in state space form

2 RegARMA models in state space form

3 **The dlm package for R**

4 MLE using the dlm package

5 Filtering, smoothing and forecasting using the dlm package

6 Final remarks

State space models in dlm

$$y_t = \boldsymbol{f}_t' \boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$

$$\boldsymbol{x}_t = \boldsymbol{G}_t \boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$

$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

Model Parameter	List Name	Time Varying Name
$\boldsymbol{f}$	`FF`	`JFF`
$\boldsymbol{G}$	`GG`	`JGG`
σ^2	`V`	`JV`
$\boldsymbol{W}$	`W`	`JW`
$\boldsymbol{m}_0$	`m0`	
$\boldsymbol{C}_0$	`C0`	

State space models in dlm

Functions to create dlm objects

Function	Model
`dlm`	generic DLM
`dlmModARMA`	ARMA process
`dlmModPoly`	nth order polynomial DLM
`dlmModReg`	Linear regression
`dlmModSeas`	Periodic — Seasonal factors
`dlmModTrig`	Periodic — Trigonometric form

Local level model

$$y_t = \boldsymbol{f}_t'\boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$
$$\boldsymbol{x}_t = \boldsymbol{G}_t\boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$
$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$\boldsymbol{x}_t = \ell_t$, $\boldsymbol{f}_t = 1$, $\boldsymbol{G}_t = 1$.
Suppose $\sigma^2 = 0.8$, $\boldsymbol{W}_t = 0.1$, $\boldsymbol{m}_0 = 0$, $\boldsymbol{C}_0 = 10^7$.

`dlm()` function specification

```
dlm(FF=1, GG=1, V=0.8, W=0.1, m0=0, C0=1e7)
```

Local level model

$$y_t = \boldsymbol{f}_t' \boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$

$$\boldsymbol{x}_t = \boldsymbol{G}_t \boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$

$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$\boldsymbol{x}_t = \ell_t$, $\boldsymbol{f}_t = 1$, $\boldsymbol{G}_t = 1$.
Suppose $\sigma^2 = 0.8$, $\boldsymbol{W}_t = 0.1$, $\boldsymbol{m}_0 = 0$, $\boldsymbol{C}_0 = 10^7$.

`dlm()` function specification

```
dlm(FF=1, GG=1, V=0.8, W=0.1, m0=0, C0=1e7)
```

Local level model

$$y_t = \boldsymbol{f}_t' \boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$
$$\boldsymbol{x}_t = \boldsymbol{G}_t \boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$
$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$\boldsymbol{x}_t = \ell_t$, $\boldsymbol{f}_t = 1$, $\boldsymbol{G}_t = 1$.
Suppose $\sigma^2 = 0.8$, $\boldsymbol{W}_t = 0.1$, $\boldsymbol{m}_0 = 0$, $\boldsymbol{C}_0 = 10^7$.

`dlm()` function specification

```
dlm(FF=1, GG=1, V=0.8, W=0.1, m0=0, C0=1e7)
```

`dlmModPoly()` function specification

```
dlmModPoly(order=1, dV=0.8, dW=0.1)
```

Local level model

```
> mod <- dlmModPoly(order=1, dV=.8, dW=.1)
> names(mod)
 [1] "m0" "C0" "FF" "V" "GG" "W" "JFF" "JV" "JGG" "JW"
> FF(mod)
     [,1]
[1,]    1
> GG(mod)
     [,1]
[1,]    1
> class(mod)
[1] "dlm"
```

Local trend model

$$y_t = \boldsymbol{f}_t'\boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$
$$\boldsymbol{x}_t = \boldsymbol{G}_t\boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$
$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$\boldsymbol{x}_t = \begin{bmatrix} \ell_t \\ b_t \end{bmatrix}$, $\boldsymbol{f}_t = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $\boldsymbol{G}_t = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$.

Suppose $\sigma^2 = 0.8$, $\boldsymbol{W}_t = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.1 \end{bmatrix}$, $\boldsymbol{m}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\boldsymbol{C}_0 = \begin{bmatrix} 10^7 & 0 \\ 0 & 10^7 \end{bmatrix}$

dlm() function specification

```
dlm(FF=matrix(c(1,0),nrow=1),
  GG=matrix(c(1,0,1,1),ncol=2),
  V=0.8, W=diag(c(0.2,0.1)),
  m0=c(0,0), C0=diag(c(1e7,1e7)))
```

Local trend model

$$y_t = \boldsymbol{f}_t'\boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$
$$\boldsymbol{x}_t = \boldsymbol{G}_t\boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$
$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$$\boldsymbol{x}_t = \begin{bmatrix}\ell_t \\ b_t\end{bmatrix}, \boldsymbol{f}_t = \begin{bmatrix}1 & 0\end{bmatrix}, \boldsymbol{G}_t = \begin{bmatrix}1 & 1 \\ 0 & 1\end{bmatrix}.$$

Suppose $\sigma^2 = 0.8$, $\boldsymbol{W}_t = \begin{bmatrix}0.2 & 0 \\ 0 & 0.1\end{bmatrix}$, $\boldsymbol{m}_0 = \begin{bmatrix}0 \\ 0\end{bmatrix}$, $\boldsymbol{C}_0 = \begin{bmatrix}10^7 & 0 \\ 0 & 10^7\end{bmatrix}$

`dlm()` function specification

```
dlm(FF=matrix(c(1,0),nrow=1),
  GG=matrix(c(1,0,1,1),ncol=2),
  V=0.8, W=diag(c(0.2,0.1)),
  m0=c(0,0), C0=diag(c(1e7,1e7)))
```

Local trend model

$$y_t = \boldsymbol{f}_t' \boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$
$$\boldsymbol{x}_t = \boldsymbol{G}_t \boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$
$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$\boldsymbol{x}_t = \begin{bmatrix} \ell_t \\ b_t \end{bmatrix}$, $\boldsymbol{f}_t = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $\boldsymbol{G}_t = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$.

Suppose $\sigma^2 = 0.8$, $\boldsymbol{W}_t = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.1 \end{bmatrix}$, $\boldsymbol{m}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\boldsymbol{C}_0 = \begin{bmatrix} 10^7 & 0 \\ 0 & 10^7 \end{bmatrix}$

`dlmModPoly()` function specification

```
dlmModPoly(order=2, dV=0.8, dW=c(0.2,0.1))
```

Time varying regression model

$$y_t = \boldsymbol{f}_t'\boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$
$$\boldsymbol{x}_t = \boldsymbol{G}_t\boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$
$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$$\boldsymbol{x}_t = \begin{bmatrix}\alpha_t \\ \beta_t\end{bmatrix}, \boldsymbol{f}_t = \begin{bmatrix}1 & z_t\end{bmatrix}, \boldsymbol{G}_t = \begin{bmatrix}1 & 0 \\ 0 & 1\end{bmatrix}.$$

Suppose $\sigma^2 = 15$, $\boldsymbol{W}_t = \begin{bmatrix}1 & 0 \\ 0 & 2\end{bmatrix}$, $\boldsymbol{m}_0 = \begin{bmatrix}0 \\ 0\end{bmatrix}$, $\boldsymbol{C}_0 = \begin{bmatrix}10^7 & 0 \\ 0 & 10^7\end{bmatrix}$

dlmModReg() function specification

```
dlmModReg(z, dV=15, dW=c(1,2))
```

Time varying regression model

$$y_t = \boldsymbol{f}_t' \boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$

$$\boldsymbol{x}_t = \boldsymbol{G}_t \boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$

$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$$\boldsymbol{x}_t = \begin{bmatrix} \alpha_t \\ \beta_t \end{bmatrix}, \boldsymbol{f}_t = \begin{bmatrix} 1 & z_t \end{bmatrix}, \boldsymbol{G}_t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Suppose $\sigma^2 = 15$, $\boldsymbol{W}_t = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$, $\boldsymbol{m}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\boldsymbol{C}_0 = \begin{bmatrix} 10^7 & 0 \\ 0 & 10^7 \end{bmatrix}$

`dlmModReg()` function specification

```
dlmModReg(z, dV=15, dW=c(1,2))
```

Linear regression with AR(2) errors

$$y_t = \alpha + \beta z_t + n_t$$

$$n_t = \phi_1 n_{t-1} + \phi_2 n_{t-2} + e_t, \qquad e_t \sim \text{NID}(0, u)$$

Linear regression with AR(2) errors

$$y_t = \boldsymbol{f}_t'\boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$
$$\boldsymbol{x}_t = \boldsymbol{G}_t\boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$
$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$$\boldsymbol{x}_t = \begin{bmatrix} \alpha \\ \beta \\ n_t \\ \phi_2 n_{t-1} \end{bmatrix}, \boldsymbol{f}_t = \begin{bmatrix} 1 & z_t & 1 & 0 \end{bmatrix}, \boldsymbol{G}_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \phi_1 & 1 \\ 0 & 0 & \phi_2 & 0 \end{bmatrix}.$$

Set $\sigma^2 = 0$, $\boldsymbol{W}_t = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & u & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$, $\boldsymbol{m}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $\boldsymbol{C}_0 = \begin{bmatrix} 10^7 & 0 & 0 & 0 \\ 0 & 10^7 & 0 & 0 \\ 0 & 0 & 10^7 & 0 \\ 0 & 0 & 0 & 10^7 \end{bmatrix}$

R (dlm) specification

```
dlmModReg(z, dV=0, dW=c(0,0)) +
  dlmModARMA(ar=c(phi1,phi2), sigma=u)
```

Linear regression with AR(2) errors

$$y_t = \boldsymbol{f}_t' \boldsymbol{x}_t + \varepsilon_t \qquad \varepsilon_t \sim \text{NID}(0, \sigma^2)$$

$$\boldsymbol{x}_t = \boldsymbol{G}_t \boldsymbol{x}_{t-1} + \boldsymbol{w}_t \qquad \boldsymbol{w}_t \sim \text{NID}(\boldsymbol{0}, \boldsymbol{W}_t)$$

$$\boldsymbol{x}_0 \sim \text{NID}(\boldsymbol{m}_0, \boldsymbol{C}_0)$$

$$\boldsymbol{x}_t = \begin{bmatrix} \alpha \\ \beta \\ n_t \\ \phi_2 n_{t-1} \end{bmatrix}, \boldsymbol{f}_t = \begin{bmatrix} 1 & z_t & 1 & 0 \end{bmatrix}, \boldsymbol{G}_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \phi_1 & 1 \\ 0 & 0 & \phi_2 & 0 \end{bmatrix}.$$

Set $\sigma^2 = 0$, $\boldsymbol{W}_t = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & u & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{m}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \boldsymbol{C}_0 = \begin{bmatrix} 10^7 & 0 & 0 & 0 \\ 0 & 10^7 & 0 & 0 \\ 0 & 0 & 10^7 & 0 \\ 0 & 0 & 0 & 10^7 \end{bmatrix}$

R (dlm) specification

```
dlmModReg(z, dV=0, dW=c(0,0)) +
  dlmModARMA(ar=c(phi1,phi2), sigma=u)
```

Outline

1. ARIMA models in state space form
2. RegARMA models in state space form
3. The dlm package for R
4. **MLE using the dlm package**
5. Filtering, smoothing and forecasting using the dlm package
6. Final remarks

Local level model estimation

Requirements

- A “build” function that takes possible parameter values and returns the model.
- Initial values for the parameters.

Example

```
loclvl <- function(p) {
  dlmModPoly(1, dV=exp(p[1]), dW=exp(p[2]))
}

fit <- dlmMLE(oil, parm=c(0,0), build=loclvl)
mod  <- loclvl(fit$par)
```

Local level model estimation

Requirements

- A “build” function that takes possible parameter values and returns the model.
- Initial values for the parameters.

Example

```
loclvl <- function(p) {
  dlmModPoly(1, dV=exp(p[1]), dW=exp(p[2]))
}

fit <- dlmMLE(oil, parm=c(0,0), build=loclvl)
mod  <- loclvl(fit$par)
```

Local level model estimation

Requirements

- A "build" function that takes possible parameter values and returns the model.
- Initial values for the parameters.

Example

```
loclvl <- function(p) {
  dlmModPoly(1, dV=exp(p[1]), dW=exp(p[2]))
}

fit <- dlmMLE(oil, parm=c(0,0), build=loclvl)
mod  <- loclvl(fit$par)
```

Local level model estimation

Requirements

- A “build” function that takes possible parameter values and returns the model.
- Initial values for the parameters.

Example

```
loclvl <- function(p) {
   dlmModPoly(1, dV=exp(p[1]), dW=exp(p[2]))
}

fit <- dlmMLE(oil, parm=c(0,0), build=loclvl)
mod  <- loclvl(fit$par)
```

Local level model estimation

```
> loclvl <- function(p) {
+   dlmModPoly(1, dV=exp(p[1]), dW=exp(p[2]))
+ }
> fit <- dlmMLE(oil, parm=c(0,0), build=loclvl)
> mod  <- loclvl(fit$par)
> V(oil.fit)
            [,1]
[1,] 0.0002563
> W(oil.fit)
      [,1]
[1,] 2427
> StructTS(oil, type="level")
Variances:
  level  epsilon
   2427        0
```

Local trend model estimation

```
> loctrend <- function(p) {
+   dlmModPoly(2, dV=exp(p[1]), dW=exp(p[2:3]))
+ }
> fit <- dlmMLE(ausair, parm=c(0,0,0), build=loctrend)
> ausair.fit <- loctrend(fit$par)
> V(ausair.fit)
           [,1]
[1,] 1.6563e-06
> W(ausair.fit)
      [,1]     [,2]
[1,] 2.327 0.000000
[2,] 0.000 0.021735
> StructTS(ausair, type="trend")
Variances:
  level    slope  epsilon
 2.2827   0.0265   0.0000
```

Local trend model estimation

```
> loctrend <- function(p) {
+   dlmModPoly(2, dV=exp(p[1]), dW=exp(p[2:3]))
+ }
> fit <- dlmMLE(ausair, parm=c(0,0,0), build=loctrend)
> ausair.fit <- loctrend(fit$par)
> V(ausair.fit)
             [,1]
[1,] 1.6563e-06
> W(ausair.fit)
      [,1]     [,2]
[1,] 2.327 0.000000
[2,] 0.000 0.021735
> StructTS(ausair, type="trend")
Variances:
  level    slope  epsilon
 2.2827   0.0265   0.0000
```

Different initialization and optimization choices often given different parameter estimates.

Linear regression with AR(2) errors

R (dlm) specification

```
dlmModReg(z, dV=0, dW=c(0,0)) +
  dlmModARMA(ar=c(phi1,phi2), sigma=u)
```

Linear regression with AR(2) errors

R (dlm) specification

```
dlmModReg(z, dV=0, dW=c(0,0)) +
  dlmModARMA(ar=c(phi1,phi2), sigma=u)
```

MLE

```
regar2 <- function(p) {
  dlmModReg(z, dV=.0001, dW=c(0,0)) +
    dlmModARMA(ar=c(p[1],p[2]), sigma=exp(p[3]))
}
z <- usconsumption[,1]
fit <- dlmMLE(usconsumption[,2], parm=c(0,0,0),
  build=regar2)
mod <- regar2(fit$par)
```

Linear regression with AR(2) errors

R (dlm) specification

```
dlmModReg(z, dV=0, dW=c(0,0)) +
  dlmModARMA(ar=c(phi1,phi2), sigma=u)
```

MLE

V must be positive.

```
regar2 <- function(p) {
  dlmModReg(z, dV=.0001, dW=c(0,0)) +
    dlmModARMA(ar=c(p[1],p[2]), sigma=exp(p[3]))
}
z <- usconsumption[,1]
fit <- dlmMLE(usconsumption[,2], parm=c(0,0,0),
  build=regar2)
mod <- regar2(fit$par)
```

BSM model

```
bsm <- function(p) {
  mod <- dlmModPoly() + dlmModSeas(4)
  V(mod) <- exp(p[1])
  diag(W(mod))[1:3] <- exp(p[2:4])
  return(mod)
}
fit <- dlmMLE(austourists, parm=c(0,0,0,0),
    build=bsm)
ausbsm <- bsm(fit$par)
```

Outline

More dlm functions

- `dlmFilter`: Kalman filter. Returns filtered values of state vectors.
- `dlmSmooth`: Kalman smoother. Returns smoothed values of state vectors.
- `dlmForecast`: Means and variances of future observations and states.

More dlm functions

- `dlmFilter`: Kalman filter. Returns filtered values of state vectors.
- `dlmSmooth`: Kalman smoother. Returns smoothed values of state vectors.
- `dlmForecast`: Means and variances of future observations and states.

More dlm functions

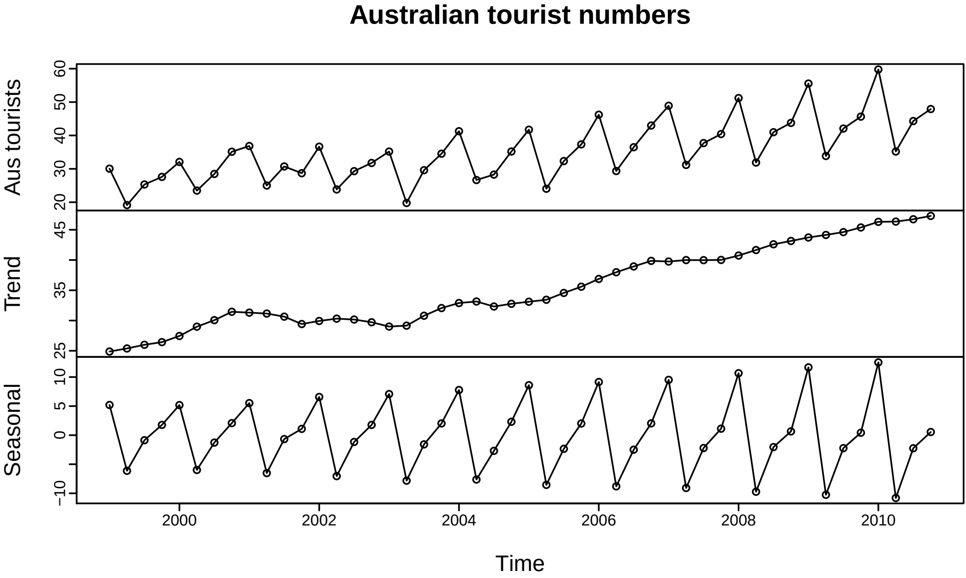
- `dlmFilter`: Kalman filter. Returns filtered values of state vectors.
- `dlmSmooth`: Kalman smoother. Returns smoothed values of state vectors.
- `dlmForecast`: Means and variances of future observations and states.

BSM model

Decomposition by Kalman smoothing

```
ausSmooth <- dlmSmooth(austourists, mod = ausbsm)
x <- cbind(austourists, dropFirst(ausSmooth$s[,c(1,3)]))
colnames(x) <- c("Aus tourists", "Trend", "Seasonal")
plot(x, type = 'o', main = "Australian tourist numbers")
```

BSM model

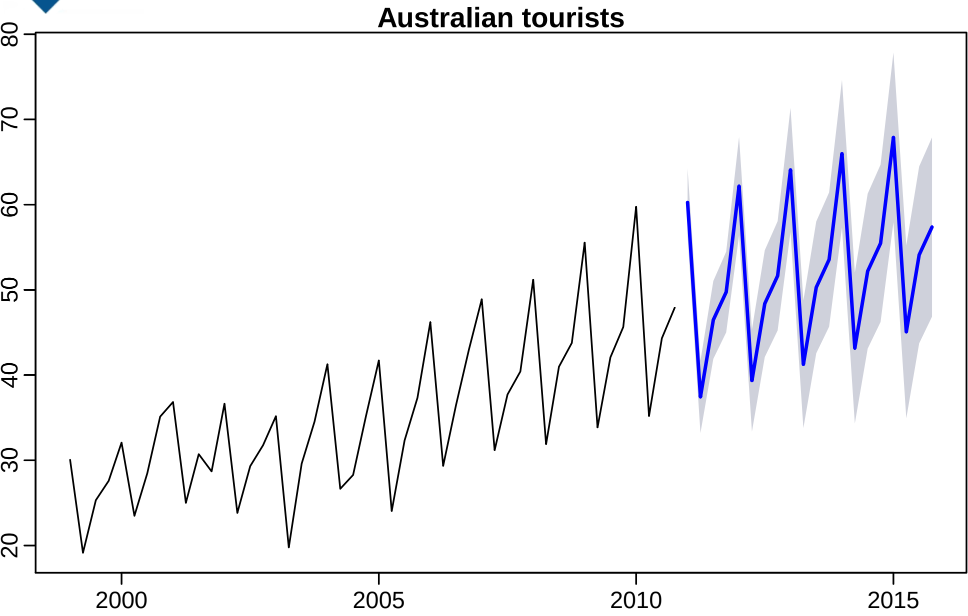

Australian tourist numbers
Aus tourists
Trend
Seasonal
Time
2000
2002
2004
2006
2008
2010
20
30
40
50
60
25
35
45
−10
0
5
10

BSM model

Forecasting by Kalman filter

```
Filt <- dlmFilter(austourists, mod = ausbsm)
Fore <- dlmForecast(Filt, nAhead = 20)
fsd <- sqrt(unlist(Fore$Q))
pl <- Fore$f + qnorm(0.05, sd = fsd)
pu <- Fore$f + qnorm(0.95, sd = fsd)
fc <- list(mean=Fore$f, lower=pl, upper=pu,
  x=austourists, level=90)
plot.forecast(fc,  main="Australian tourists")
```

BSM model



Outline

Final remarks

- State space models come in lots of flavours. We have only touched the surface.
- We haven't even mentioned the Bayesian flavours.
- State space models are a flexible way of handling lots of time series models and provide a framework for handling missing values, likelihood estimation, smoothing, forecasting, etc.

Final remarks

- State space models come in lots of flavours. We have only touched the surface.
- We haven't even mentioned the Bayesian flavours.
- State space models are a flexible way of handling lots of time series models and provide a framework for handling missing values, likelihood estimation, smoothing, forecasting, etc.

Final remarks

- State space models come in lots of flavours. We have only touched the surface.
- We haven’t even mentioned the Bayesian flavours.
- State space models are a flexible way of handling lots of time series models and provide a framework for handling missing values, likelihood estimation, smoothing, forecasting, etc.

Recommended References

1. RJ Hyndman, AB Koehler, J Keith Ord, and RD Snyder (2008). ***Forecasting with exponential smoothing: the state space approach***. Springer
2. AC Harvey (1989). ***Forecasting, structural time series models and the Kalman filter***. Cambridge University Press
3. J Durbin and SJ Koopman (2001). ***Time series analysis by state space methods***. Oxford University Press
4. G Petris, S Petrone, and P Campagnoli (2009). ***Dynamic Linear Models with R***. Springer

Contact details

www.robjhyndman.com

Rob.Hyndman@monash.edu