Modelos de Estado-Espacio

Gustavo A. Caffaro

Instituto Tecnológico de Santo Domingo

Nov 2021 - Ene 2022

Introducción

Contenido:

- Las matemáticas de la modelación Estado-Espacio
- Filtrado y predicción
- Proyecciones

Introducción

- Los modelos de estado-espacio son modelos dinámicos lineales que consisten en dos ecuaciones.
- Estos modelos son útiles para trabajar series estructurales y dinámicas.
- Las variables no observadas reflejan el estado del sistema que evoluciona a través del tiempo
- Algunos ejemplos:
  - Análisis macroeconómico
  - Otras variables no observadas, como expectativas, salarios de reserva, ingreso permanente, etc.
  - Modelos de equilibrio general (DSGE)
  - Modelos de volatilidad estocástica

Introducción

- El estado del sistema está determinado por un vector no observado $\alpha = \{\alpha_1, \ldots, \alpha_n\}$, el cual está asociado con las observaciones $y = \{y_1, \ldots, y_n\}$
- La relación entre los a_ts y los y_ts está especificada por el modelo de estado espacio

Introducción

- Digamos que estamos interesados en medir en nivel del agua del río Nilo
- Cada año, vamos a un lugar específico del río y tomas una muestra
- Puede ser que ese día llovió, o había sequía, o el instrumento de medición era impreciso, o cualquier otra razón que no nos permitiera medir exactamente el nivel del agua
  - La idea es que no pudimos tomar una muestra **precisa** del nivel del agua por factores (aleatorios) que no podemos controlar

$$NivelAguaObservada_t = NivelAguaReal_t + Error_t$$

Introducción

- Sabemos entonces que cada año, nuestra muestra es una función del nivel real y un error de medición que siempre se encuentra presente (pero es aleatorio)
- Sin embargo, hace sentido asumir que el nivel real del río Nilo varía en el tiempo
  - Por ej. la evolución de la cuenca hidrográfica, el uso del agua del río, la construcción de presas, etc.
- Entonces hace sentido que incluyamos la siguiente ecuación:

$$NivelAguaReal_t = NivelAguaReal_{t-1} + ErrorAditivo_t$$

El verdadero (y no observado) nivel de agua hoy depende del nivel del año pasado y otro componente, también aleatorio.

Modelo básico de estado espacio

Una serie de tiempo y_t podría estar expresada en forma aditiva

$$y_t = \mu_t + \gamma_t + \epsilon_t, \quad t = 1, \ldots, T$$

donde

- μ_t es el componente tendencia (variable)
- γ_t es el componente periódico (estacional)
- ϵ_t es el componente error

Para modelar μ_t y γ_t suponemos que la serie a_t sigue un proceso de caminata aleatoria

$$\alpha_{t+1} = \alpha_t + w_t, \quad w_t \sim iid.N(0, W_t)$$

Modelo básico de estado espacio

Si asumimos que no tenemos componente estacional ($\gamma_t = 0$), y que $\mu_t = \alpha_t$ (donde α_t es un RW), podemos reescribir las ecuaciones anteriores como

$$y_t = \alpha_t + S_t\epsilon_t, \quad \epsilon_t \sim iid.N(0, V)$$
$$\alpha_{t+1} = \alpha_t + R_t w_t, \quad w \sim iid.N(0, W)$$

Este modelo tiene características de estado espacio ya que tenemos

- una **ecuación de transición de estados**, la cual refleja la dinámica de las variables estado no observadas, $\{\alpha_1, \ldots, \alpha_n\}$
- una **ecuación de observaciones**, la cual describe la relación entre las variables observadas $\{y_1, \ldots, y_n\}$ y las variables estado no observadas, α_t

Modelo básico de estado espacio

- El objetivo de la modelación de estado espacio es inferir las propiedades de los α_t (no observados) a partir de los y_t (observados)
- Para llevar a cabo este proceso, usamos la técnica del **Filtro de Kalman**.

Exposición General

- La forma de estado espacio provee una representación unificada de
  - modelos ARIMA, modelos con componentes no observados
  - series de tiempo dinámicas, con parámetros que varían en el tiempo
  - regresiones no paramétricas, como regresiones spline

Ecuación de observaciones

- Consideremos m variables estado que están sujetas a distorciones y ruido (ej. choques)
- Estas variables están contenidas en α_t, un vector $(m \times 1)$
- Las n variables observadas están definidas por el vector $(N \times 1)$, y_t

Entonces, la ecuación de observaciones se define como

$$y_t = F_t \alpha_t + S_t \epsilon_t,$$

donde

- F_t es una matriz de orden $N \times m$
- r es la dimensión del vector de errores de la ecuación de observaciones
- ϵ_t es un vector $r \times 1$ con media cero y matriz de varianza-covarianza V
- S_t es una matriz de orden $N \times r$

Ecuación estado

- La ecuación estado entonces, biene definida por

$$\alpha_{t+1} = G_t \alpha_t + R_t w_t,$$

donde

- G_t y R_t son matrices de orden $m \times m$ y $m \times g$, respectivamente
- g se refiere a la dimensión del vector de errores de la ecuación de estados
- w_t es un vector $g \times 1$ con media cero y matriz de varianza-covarianza W

Errores

Se asume que los errores en las ecuaciones de observaciones y de estado no están correlacionados:

$$\begin{bmatrix} \epsilon_t \\ w_t \end{bmatrix} \sim iid.N\left(0, \begin{bmatrix} V & 0 \\ 0 & W \end{bmatrix}\right)$$

Y también no están correlacionados con el vector inicial α_0

$$E(\alpha_0 \epsilon_t') = 0$$

y

$$E(\alpha_0 w_t') = 0$$

Modelo de estado espacio

Entonces, un modelo de estado espacio consiste en la ecuación de observaciones y la ecuación de estados:

$$y_t = F_t \alpha_t + S_t \epsilon_t, \quad \epsilon_t \sim iid.N(0, V)$$
$$\alpha_{t+1} = G_t \alpha_t + w_t, \quad w \sim iid.N(0, W),$$

con condiciones iniciales

$$\alpha_0 \sim N(m_0, C_0)$$

Modelos estado espacio en R

Existen numerosos paquetes que nos facilitan la modelación de modelos de estado espacio.

Algunos de ellos son

- `dlm`
- `sspir`
- `StructTS`
- `RWinBugs`
- `MASS`
- `RStan`

Ejemplo: Modelo de nivel local (local level)

- Es el modelo de estado espacio más sencillo
- También se conoce como el random walk plus noise model y first order polynomial model
- Es un modelo univariado, por lo que el vector estado es unidimensional
- Componente nivel varía a través del tiempo
- Las ecuaciones de observaciones y estados son

$$y_t = \mu_t + \epsilon_t, \quad \epsilon_t \sim iid.N(0, V)$$
$$\mu_{t+1} = \mu_t + w_t, \quad w \sim iid.N(0, W)$$

- $F_t = G_t = [1]$
- Los únicos parámetros del modelo son las varianzas V y W, los cuales estimamos mediante maximum likelihood o técnicas bayesianas.

Paquete dlm en R

https://cran.r-project.org/web/packages/dlm/vignettes/dlm.pdf

A DLM is specified by the following equations:

$$\begin{cases} y_t = F_t\theta_t + v_t, & v_t \sim \mathcal{N}(0, V_t) \\ \theta_t = G_t\theta_{t-1} + w_t, & w_t \sim \mathcal{N}(0, W_t) \end{cases}$$

for $t = 1, \ldots, n$, together with a *prior* distribution for θ_0:

$$\theta_0 \sim \mathcal{N}(m_0, C_0).$$

Here y_t is an m-dimensional vector, representing the observation at time t, while θ_t is a generally unobservable p-dimensional vector representing the state of the system at time t. The v_t's are observation errors and the w_t's evolution errors. The matrices F_t and G_t have dimension m by p and p by p, respectively, while V_t and W_t are variance matrices of the appropriate dimension.

Ejemplo: Modelo de nivel local (local level)

```
library(dlm)

# Definamos un modelo RW plus drift con
# V=0.8, W=0.1, m_0=0, y C_0=100
model1_a <- dlm(FF=1,V=0.8,GG=1,W=0.1,m0=0,C0=100)
```

También podemos usar la función `dlmModPoly()`, que es bastante útil porque viene con varios parámetros predeterminados. Como esto es un sistema de 1er orden:

```
# Definamos un modelo RW plus drift con
# V=0.8, W=0.1, m_0=0, y C_0=100
model1_b <- dlmModPoly(order=1,dV=0.8,dW=0.1,C0=100)
```

Modelo de nivel local con tendencia (local level trend)

- El modelo de nivel local con tendencia (local level trend model) tiene dos ecuaciones estado para incluir un componente pendiente, v_t
- Se puede derivar de la especificación general al definir

$$\begin{aligned} y_t &= \mu_t + \epsilon_t, \quad \epsilon_t \sim iid.N(0, V) \\ \mu_{t+1} &= \mu_t + v_t + \xi_t, \quad \xi \sim iid.N(0, W_\xi) \\ v_{t+1} &= v_t + \psi_t, \quad \psi \sim iid.N(0, W_\psi) \end{aligned}$$

o, de manera alternativa

$$\alpha_t = \begin{bmatrix} \mu_t \\ v_t \end{bmatrix}, w_t = \begin{bmatrix} \xi_t \\ \psi_t \end{bmatrix}, G_t = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, F_t = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, S_t = 1,$$

$$W = \begin{bmatrix} W_\xi & 0 \\ 0 & W_\psi \end{bmatrix}, R_t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Modelo de nivel local con tendencia (local level trend)

Podemos acceder y/o modificar los componentes del modelo utilizando las funciones correspondientes:

```
# Definimos el dlm de orden 2
model2 <- dlmModPoly(order=2)

W(model2)
```

```
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    1
```

```
W(model2) <- matrix(c(10,0,
                      0,1/10),ncol=2,byrow=T)
W(model2)
```

```
##      [,1] [,2]
## [1,]   10  0.0
## [2,]    0  0.1
```

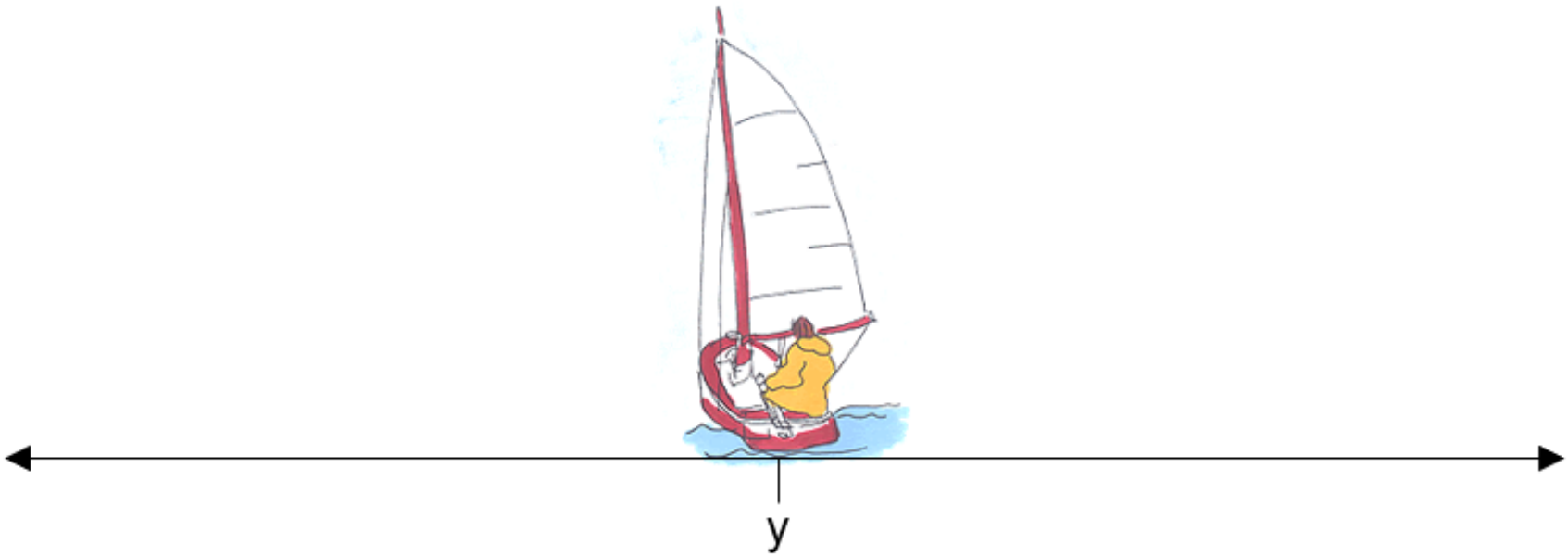
Filtro de Kalman

- Los modelos de estado espacio son estimados con Filtros de Kalman.
  - El Filtro de Kalman fue desarrollado en 1960 por Rudolph Kalman
  - El propósito original del filtro de Kalman era tomar mediciones observadas a través del tiempo, que contienen ruidos o errores, y producir valores más cercanos a los valores reales de las mediciones
- El filtro de Kalman es un algoritmo de procesamiento recursivo
  - Es esencialmente un conjunto de reglas de estimación que se aplican cada vez que surge nueva información
  - El filtro de Kalman no requiere que se almacenen mediciones anteriores, ni volver a procesar cada data en cada período de tiempo

Filtro de Kalman

- El Filtro de Kalman genera estimaciones óptimas de las cantidades en cuestión, dadas las observaciones
- Por óptimo nos referimos a
  - Para un sistema lineal y con errores Gaussianos, el filtro de Kalman es la "mejor" estimación basada en las observaciones anteriores
  - Para sistemas no lineales, que el sistema es "calificado"

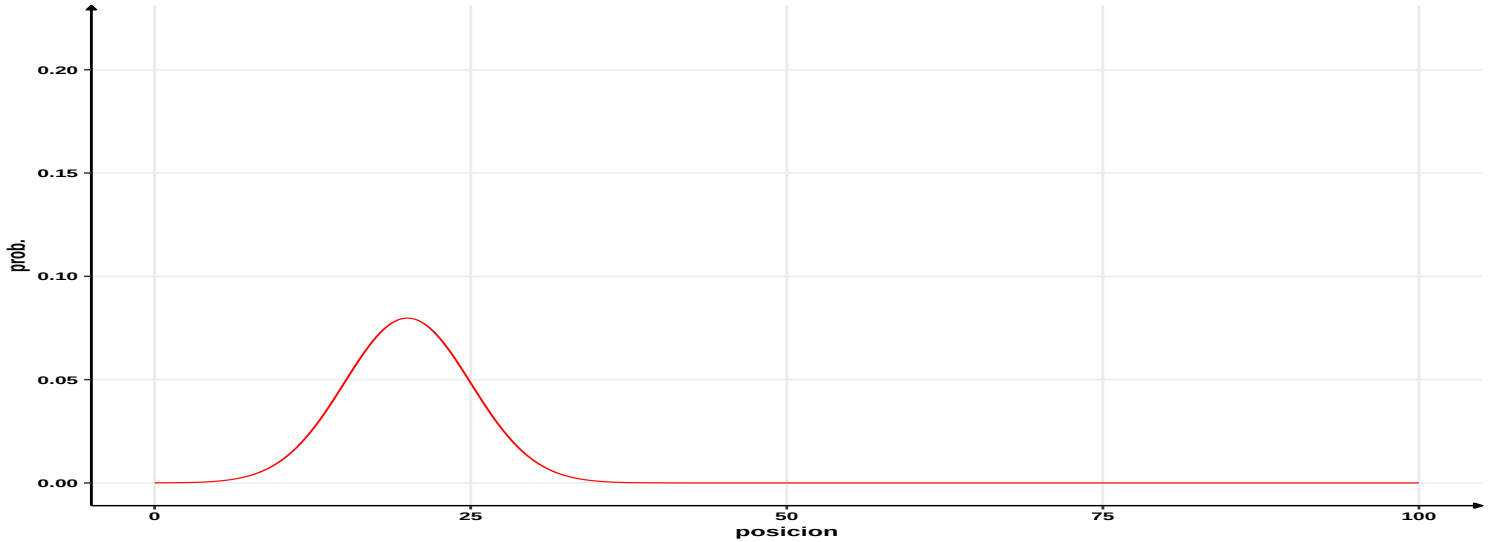
Introducción Conceptual



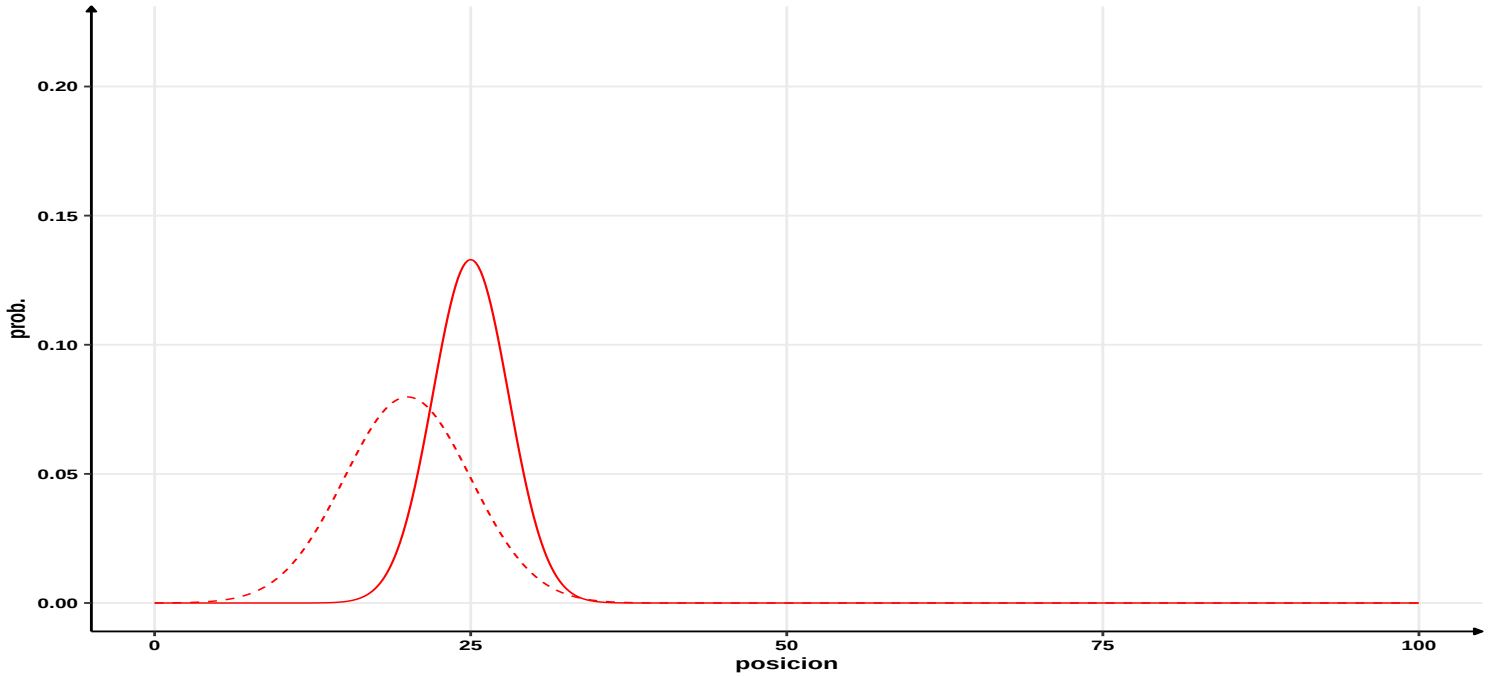
- Digamos que estamos perdidos en el mar en una línea unidimensional
- Nuestra posición es $y(t)$
- Asumamos mediciones distribuídas de forma Gaussiana

Introducción Conceptual

- Utilizando un sextante, la medición (imprecisa) de nuestra posición en el período t_1 es $N(z_1, \sigma^2_{z_1})$
  - La posición estimada es $\hat{y}(t_1) = z_1$
  - El error de esta estimación tiene varianza $\sigma^2_x(t_1) = \sigma^2_{z_1}$
- La proyección de nuestra posición en t_2 es z_1 (el bote se mantiene en el mismo lugar)

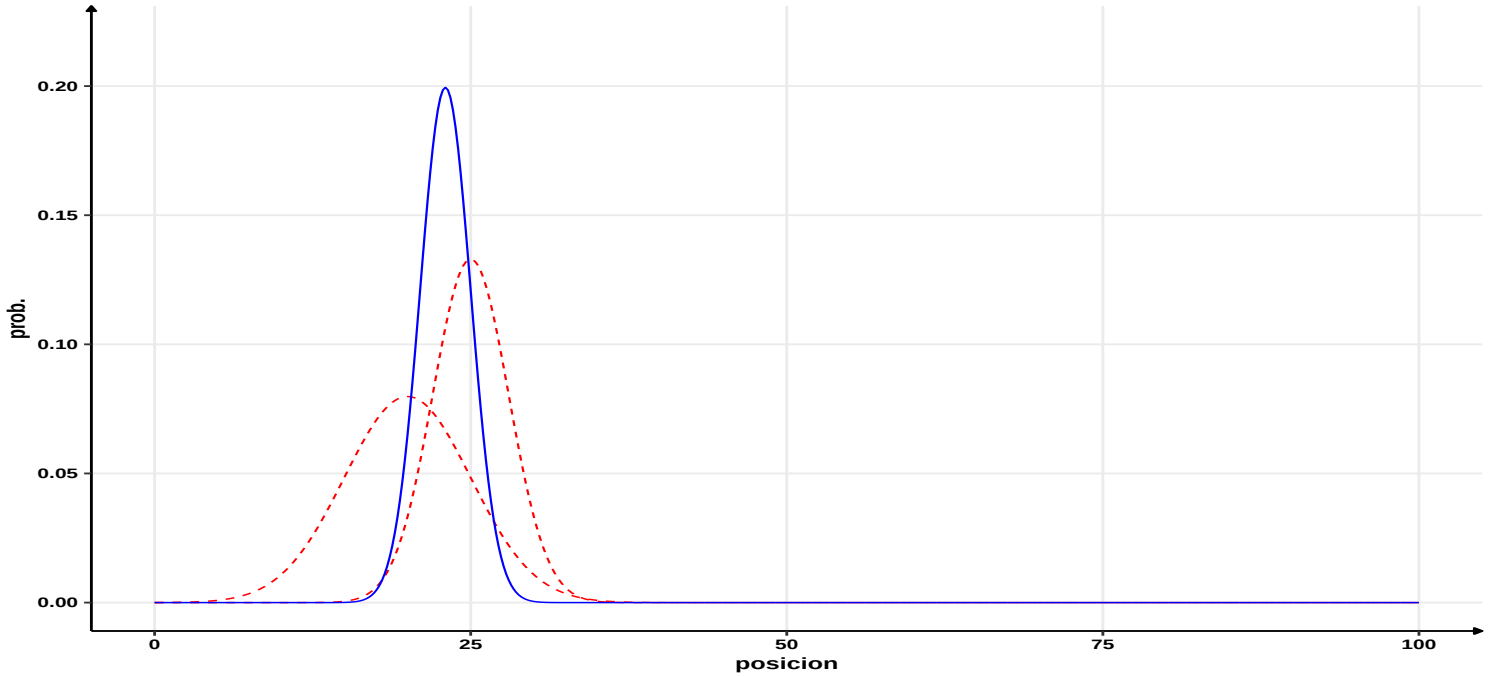


Introducción Conceptual



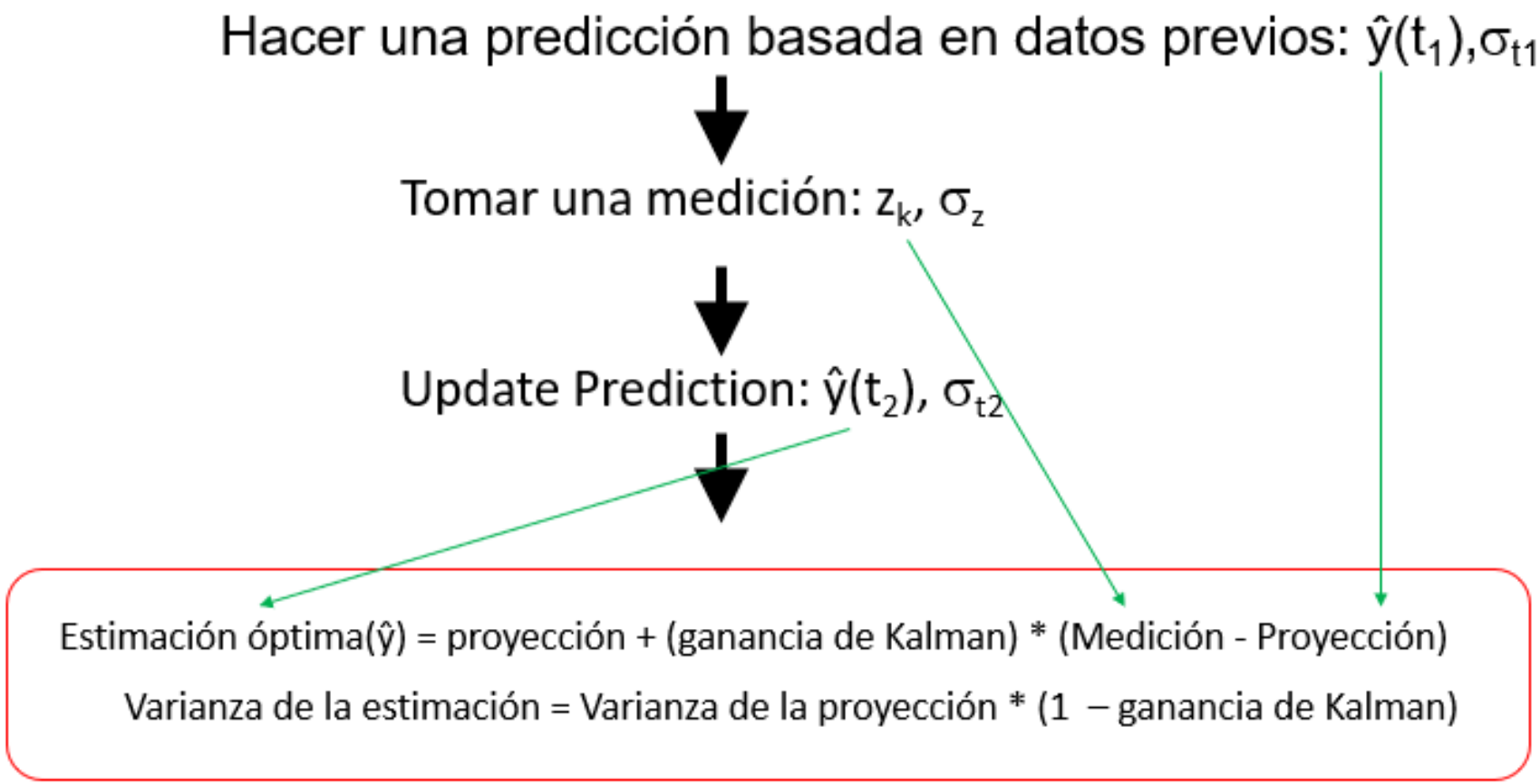
- Si usamos un GPS (más preciso), estimaríamos que nuestra posición en t_2 sería $N(z_2, \sigma^2_{z_2})$
- Podríamos corregir la predicción de nuestra posición utilizando una nueva medición de $\hat{y}(t_2)$

Introducción Conceptual

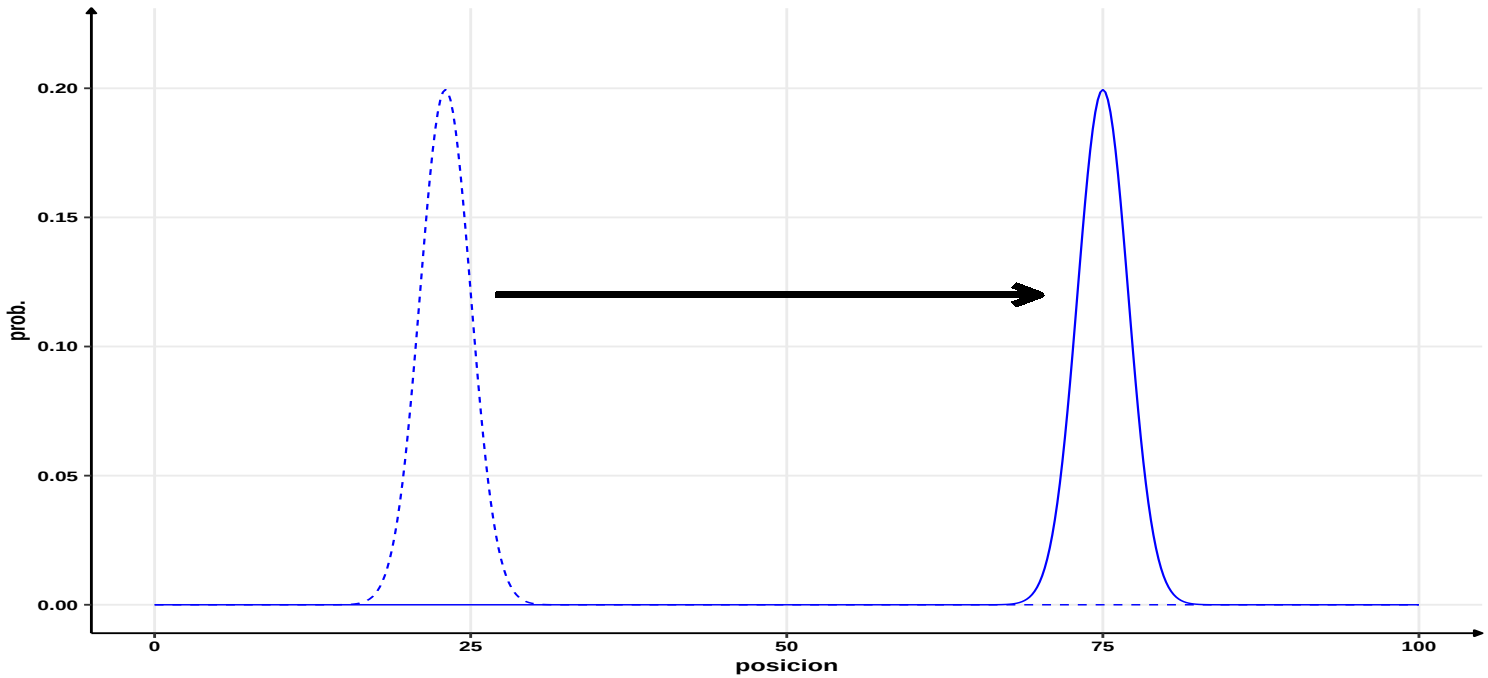


- El filtro de Kalman nos ayuda a "fusionar" las mediciones y proyecciones en base a cuánto queremos confiar en cada instrumento.
- La nueva media "corregida" es la nueva estimación de nuestra posición
  - La posición predecida por el sextante se actualizó utilizando una medición GPS
- La nueva varianza es menor que la de las dos varianzas previas

Introducción Conceptual

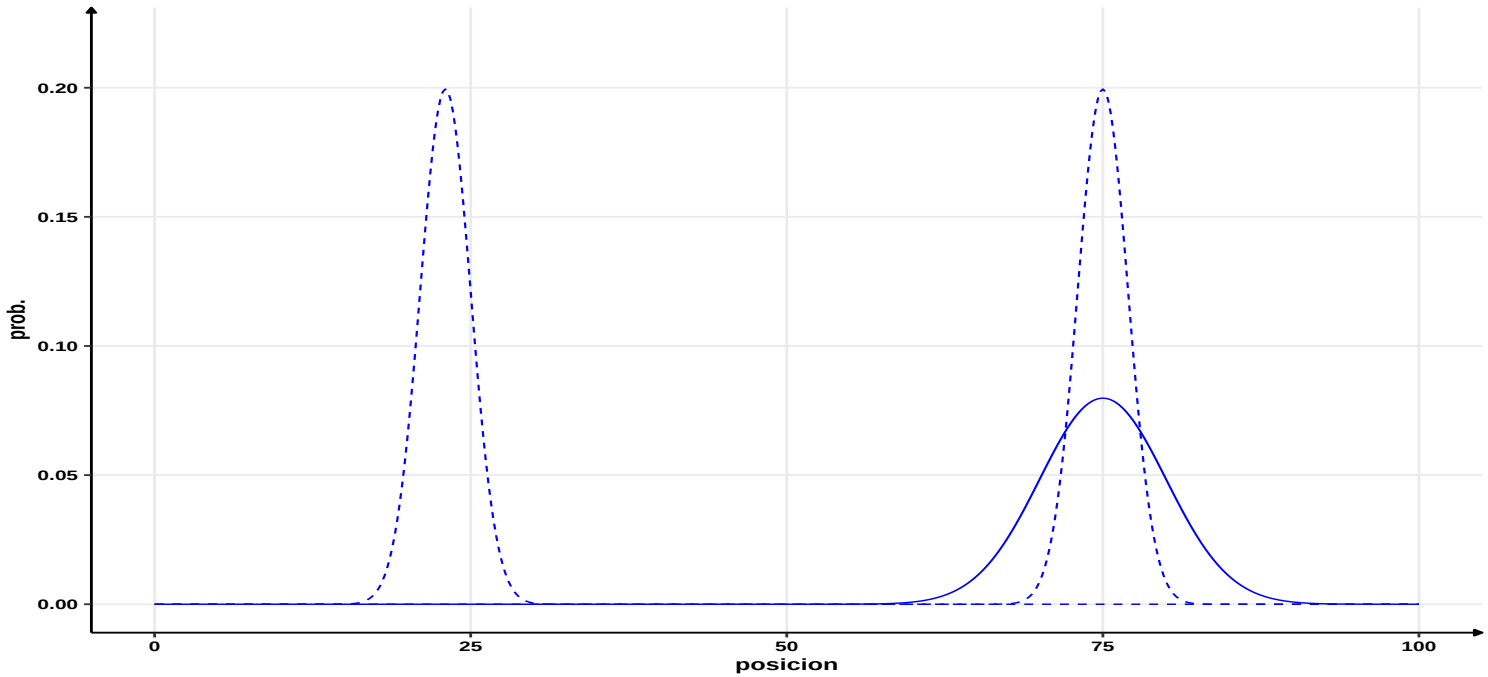


Introducción Conceptual



- Ahora digamos que el bote se empieza a desplazar en el período t_2 con velocidad $dy/dt = u$
- Un approach ingenuo sería predecir la posición en t_3 como un desplazamiento de la curva hacia la derecha
  - Esto funcionaría si conocieramos la velocidad exacta (un modelo perfecto)

Introducción Conceptual



- Sin embargo, si la velocidad exacta es desconocida, es mejor asumir un modelo imperfecto agregando ruido Gaussiano: $dy/dt = u + w$
- La distribución de la predicción **se desplaza y se expande**.

Introducción Conceptual

Recapitulando:

- Condiciones iniciales ($\hat{y}_{k-1}$ y σ_{k-1})
- Proyecciones ($\hat{y}'_k$ y σ'_k)
  - Utiliza las condiciones iniciales y el modelo (por ej. velocidad constante) para hacer una proyección
- Medición (z_k)
- Corrección ($\hat{y}_k$ y σ_k)
  - Utiliza la medición para corregir la proyección

Entonces, la estimación óptima tendrá una varianza menor.

El Filtro de Kalman

Dado el modelo de estado espacio

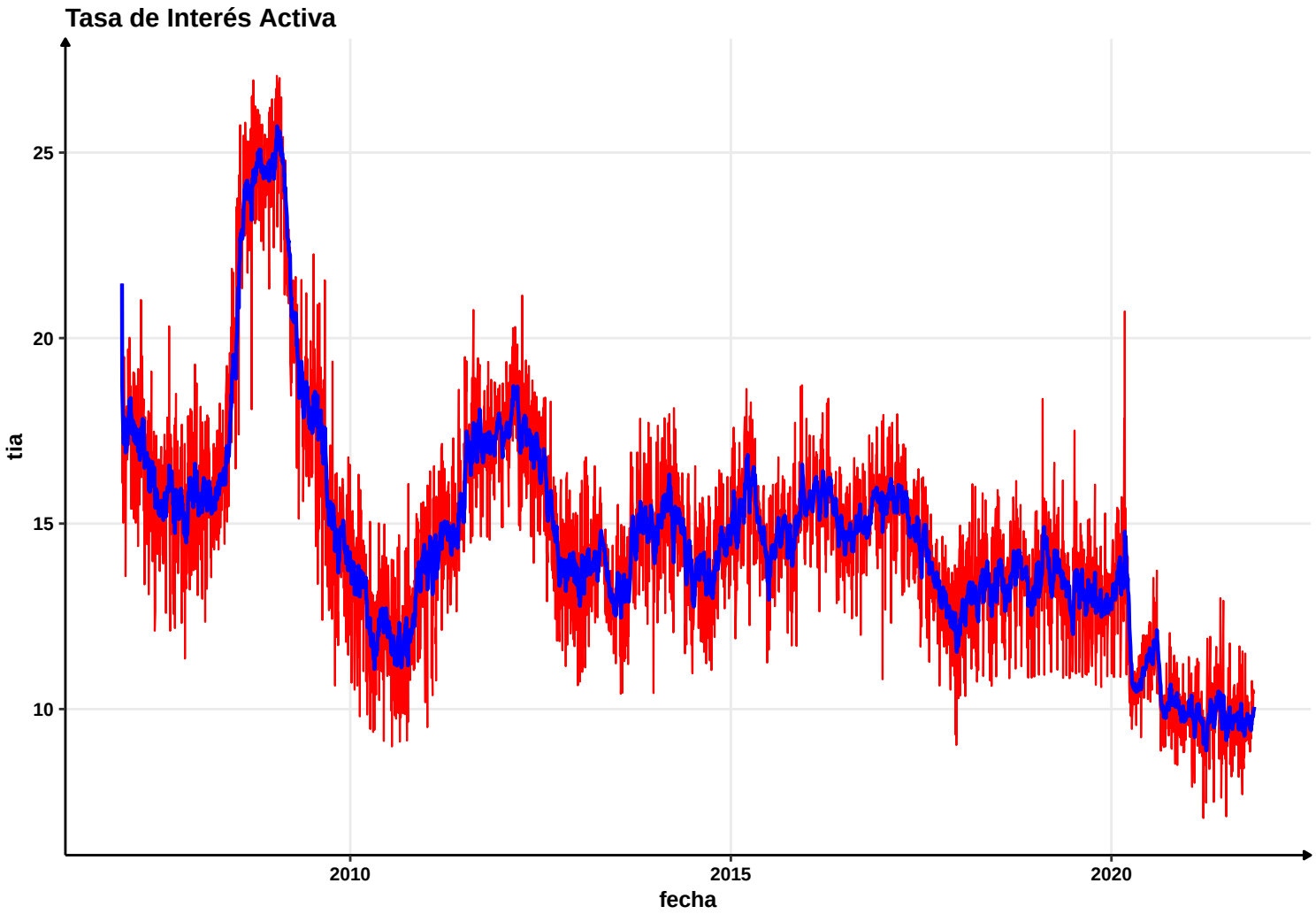
$$y_t = F_t\alpha_t + S_t\epsilon_t, \quad \epsilon_t \sim iid.N(0, V)$$
$$\alpha_{t+1} = G_t\alpha_t + R_t w_t, \quad w \sim iid.N(0, W),$$

la expresión del estado obtenido a partir del Filtro de Kalman viene dada por

$$\alpha_{t+1} = \alpha_t + K_t(y_t - F_t'\alpha_t),$$

donde K_t es la ganancia de Kalman.

Ej. TIA



El filtro de Kalman en R

La función `dlmFilter()` utiliza un filtro de Kalman para computar valores filtrados de los vectores estado, en adición a las matrices de varianza-covarianza.

```
tia <- readxl::read_xlsx("../data/tia.xlsx")

# Definimos un DLM de orden 1
buildFun <- function(x){
  dlmModPoly(1,dV=exp(x[1]),dW=exp(x[2]))
}

# Estimamos los desconocidos por MLE
fit <- dlmMLE(tia$tia,
              parm = c(0,0),
              build = buildFun)

# construimos nuestro modelo
dlmTIA <- buildFun(fit$par)
dlmTIA

# aplicmos el filtro de kalman
tiaFilt <- dlmFilter(tia$tia,dlmTIA)
```

El suavizado de Kalman

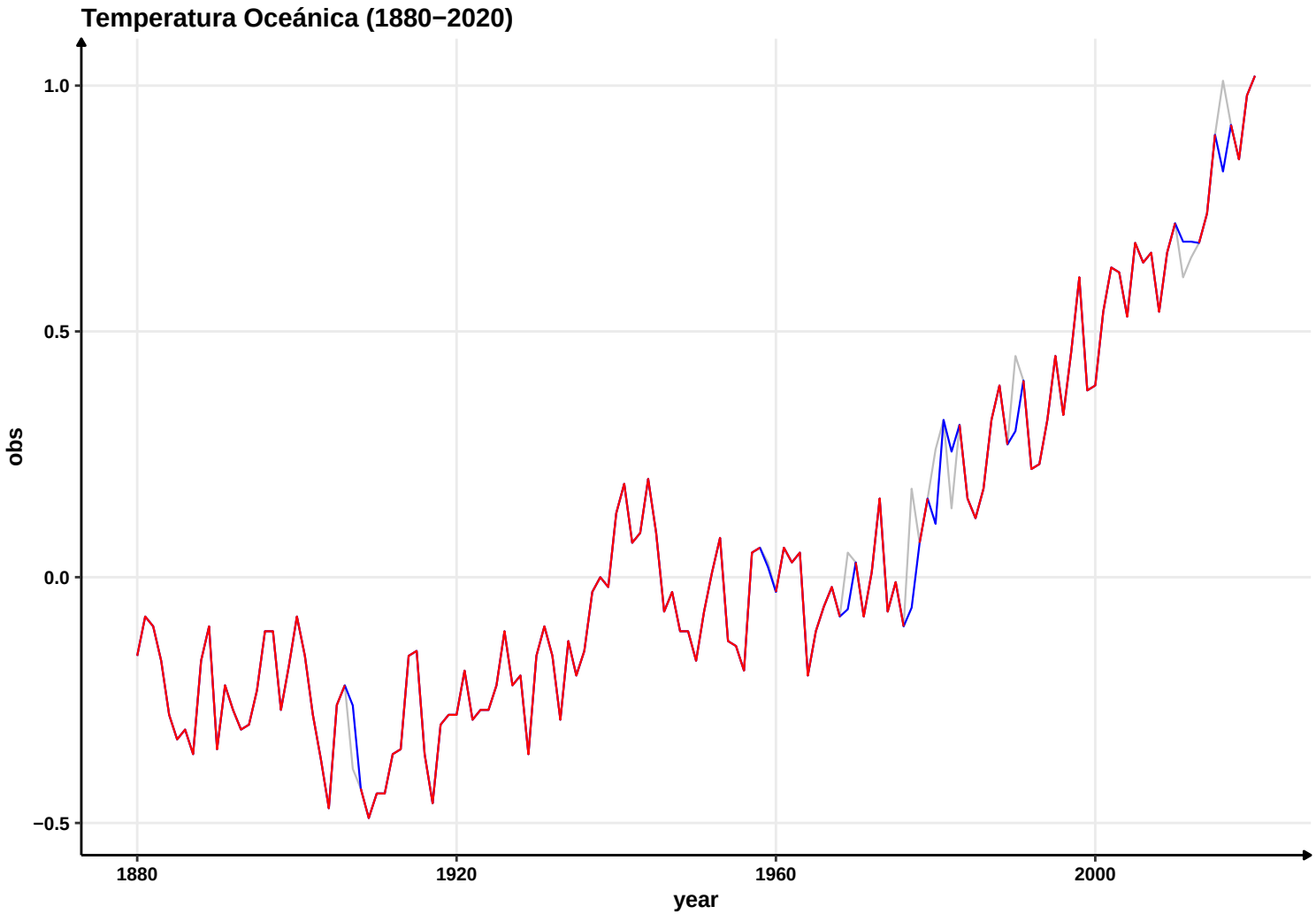
- El filtro de Kalman utiliza valores de y_t y α_t para estimar los valores de α_{t+1}
- Por lo tanto, los valores del filtro de Kalman parece que proyectan y_t por un período
- Si vamos del último valor hasta el primero, utilizamos el suavizado de Kalman

Llenando los NAs

Una de las ventajas del filtro de Kalman es que nos permite hacer interpolaciones para llenar datos no disponibles.

Por ejemplo, tomemos los datos de temperatura oceánica que vimos en la práctica 1 y eliminemos 10 observaciones de manera aleatoria.

Llenando los NAs



Llenando los NAs

```
# Importamos la data
d_temp <- read.csv("../data/global_temp.csv")
x <- d_temp$ocean_temp_index

set.seed(1234) # para reproducibilidad
inx <- sample(1:length(x),10) # eliminamos 10 obs. aleatorias
x[inx] <- NA

# Definimos un DLM de orden 1
buildFun <- function(x){
  dlmModPoly(1,dV=exp(x[1]),dW=exp(x[2]))
}

# Estimamos los desconocidos por MLE
fit <- dlmMLE(x,
              parm = c(0,0),
              build = buildFun)

# construimos nuestro modelo
dlmTemp <- buildFun(fit$par)

# aplicmos el filtro de kalman
tempFilt3 <- dlmFilter(x,dlmTemp)



# graficamos
plot(x, type="l")
lines(dropFirst(tempFilt3$m),type="l",col="red")
```