Actualizacion de ecuacion V desconocida

Se utiliza la base de datos **freeny**.

Actualización de ecuaciones

Se parte de la distribución posterior en $t = 19$ y se obtiene la distribución a priori en $t = 20$.

```
#A. Posterior en t=19
m19 <- c(1.5, 1.8, -0.7)
C19 <- matrix(c(0.00002, 0.00001, -0.00002, 0.00001, 0.00003, -0.00001, -0.00002,
                -0.00001, 0.00002), ncol = 3)

#Valores conocidos de G20 y W20
G20 <- matrix(c(1.001, 0, 0, 0, 1, 0, 0, 0, 1), ncol=3)
W20 <- matrix(c(0.00001, 0, 0, 0, 0.00001, -0.00001, 0, -0.00001, 0.00005), ncol=3)

#B. Priori de parámetros en t=20
a20 <- G20 %*% m19
R20 <- G20 %*% C19 %*% t(G20) + W20
a20
```

```
##        [,1]
## [1,]  1.501
## [2,]  1.800
## [3,] -0.700
```

```
R20
```

```
##            [,1]       [,2]       [,3]
## [1,]  3.004e-05  1.001e-05 -2.002e-05
## [2,]  1.001e-05  4.000e-05 -2.000e-05
## [3,] -2.002e-05 -2.000e-05  7.000e-05
```

Con los valores de las variables explciativas se calculan el pronóstico y su varianza.

```
freeny[20,]
```

```
##          y lag.quarterly.revenue price.index income.level market.potential
## 1967 9.314                 9.284        4.51        6.061            13.07
```

```
F20 <- c(1, 6.06093, 4.51018) #Variables explicativas en t=20. El 1 es para
                              #agregar el intercepto
S19 <- 0.00005 # Estimación de V en T=19
n19 <- 19.5 # Grados de libertad
```

```r
#C. Pronóstico a un periodo.
f20 <- as.numeric(t(F20) %*% a20)
Q20 <- as.numeric(t(F20) %*% R20 %*% F20 + S19)
f20
```

```
## [1] 9.254
```

```r
Q20
```

```
## [1] 0.001821
```

Intervalo del pronóstico.En este caso las distribuciones son t de Student.

```r
c(qst(0.025, nu = n19, mu = f20, sigma = sqrt(Q20)),
  qst(0.975, nu = n19, mu = f20, sigma = sqrt(Q20)))
```

```
## [1] 9.165 9.343
```

Una vez que se observa el valor real, se calcula la distribución posterior en t=20 y se reestima el valor de la varianza de observación desconocida.

```r
#Valor observado de Y20:
Y20 <- 9.31378
#D. Posterior en t=20
A20 <- R20 %*% F20 / Q20
e20 <- Y20-f20
m20 <- a20 + A20 %*% e20
n20 <- n19 + 1
S20 <- S19 + (S19/n20)*(e20^2/Q20-1)
C20 <- (S20/S19)*(R20-A20 %*% t(A20) * Q20)
m20
```

```
##         [,1]
## [1,]  1.5015
## [2,]  1.8053
## [3,] -0.6943
```

```r
C20
```

```
##            [,1]       [,2]       [,3]
## [1,]  3.145e-05  1.044e-05 -2.100e-05
## [2,]  1.044e-05  2.674e-05 -3.721e-05
## [3,] -2.100e-05 -3.721e-05  5.577e-05
```

Función para actualizar más de un periodo cuando la varianza de observación es desconocida.

```r
datos_ej <- freeny %>%
  mutate(intercept = 1) %>%
  slice(20:n()) %>%
  dplyr::select(y, intercept, income.level, price.index)
```

```r
#Se asume que Gt, Vt y Wt son ctes conocidas para toda t.
actualizacion_V_desc <- function(datos, m0, C0, G, W, S0, n0){
  mt_menos_1 <- m0
  Ct_menos_1 <- C0
  St_menos_1 <- S0
  nt_menos_1 <- n0
  lista_at <- list()
  lista_Rt <- list()
  lista_ft <- list()
  lista_Qt <- list()
  lista_mt <- list()
  lista_Ct <- list()
  lista_CI <- list()
  lista_CI_inf <- list()
  lista_CI_sup <- list()
  lista_St <- list()
  for(t in 1:length(datos$y)){

    at <- G %*% mt_menos_1
    Rt <- G %*% Ct_menos_1 %*% t(G) + W
    Ft <- as.numeric(datos[t, 2:4])
    ft <- t(Ft) %*% at
    Qt <- t(Ft) %*% Rt %*% Ft + St_menos_1
    CI <- c(qst(0.025, nu = nt_menos_1, mu = ft, sigma = sqrt(Qt)),
            qst(0.975, nu = nt_menos_1, mu = ft, sigma = sqrt(Qt)))
    CI_inf <- CI[1]
    CI_sup <- CI[2]
    Yt <- datos[t,1]
    At = Rt %*% Ft %*% solve(Qt)
    et = Yt-ft
    mt = at + At %*% et
    nt <- nt_menos_1 + 1
    St <- as.numeric(St_menos_1 + (St_menos_1/nt) * (et^2 %*% solve(Qt) - 1))
    Ct <- (St/St_menos_1)*(Rt - At %*% Qt %*% t(At))
    lista_at[[t]] <- at
    lista_Rt[[t]] <- Rt
    lista_ft[[t]] <- ft
    lista_Qt[[t]] <- Qt
    lista_CI[[t]] <- CI
    lista_CI_inf[[t]] <- CI_inf
    lista_CI_sup[[t]] <- CI_sup
    lista_mt[[t]] <- mt
    lista_Ct[[t]] <- Ct
    lista_St[[t]] <- St
    mt_menos_1 <- mt
    Ct_menos_1 <- Ct
    nt_menos_1 <- nt
    St_menos_1 <- St
  }
  return(list("at" = lista_at, "Rt" = lista_Rt, "ft" = lista_ft,
              "Qt" = lista_Qt, "CI" = lista_CI, "CI_inf" = lista_CI_inf,
              "CI_sup" = lista_CI_sup, "mt" = lista_mt, "Ct" = lista_Ct,
              "St" = lista_St))
```

```
}
```

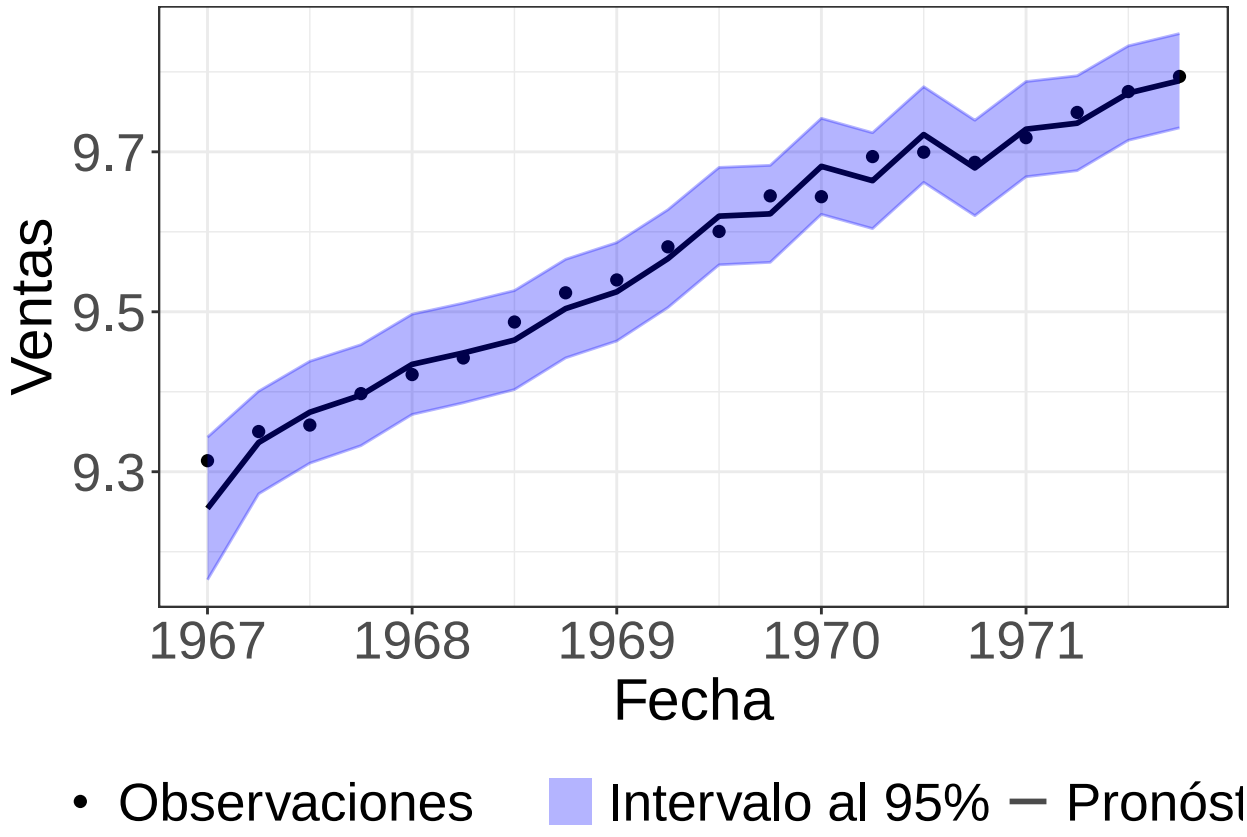
Se aplica la función.

```
res_dlm <- actualizacion_V_desc(datos_ej, m19, C19, G20, W20, S19, n19)
```

Se grafican los pronósticos a un paso y sus intervalos.

```
df_graficas <- data.frame("fecha" = datos_ej %>% row.names(), "y_real" = datos_ej$y,
           "y_pronostico" = res_dlm$ft %>% unlist(), "CI_inf" = res_dlm$CI_inf %>% unlist(),
           "CI_sup" = res_dlm$CI_sup %>% unlist()) %>%
  mutate(fecha = as.numeric(fecha))
```

```
ggplot(data = df_graficas, aes(x = fecha)) +
  geom_point(aes(y = y_real, shape = "Observaciones"), size = 2) +
  geom_line(aes(y = y_pronostico, color = 'Pronósticos'), size = 1) +
  geom_line(aes(y = CI_inf), color = "blue", alpha = 0.3) +
  geom_line(aes(y = CI_sup), color = "blue", alpha = 0.3) +
  geom_ribbon(aes(ymax = CI_sup, ymin = CI_inf, fill = 'Intervalo al 95%'), alpha = 0.3) +
  theme_bw() +
  scale_colour_manual(
    name = "", values = c("Intervalo al 95%" = "transparent",
                          "Pronósticos" = "black")) +
  scale_fill_manual(
    name = "",  values = c("Intervalo al 95%" = "blue",
                           "Pronósticos" = "transparent")) +
  theme(legend.position = "bottom") +
  labs(shape = "") +
  ylab('Ventas') +
  xlab('Fecha') +
  theme(axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20),
        axis.title = element_text(size = 22),
        legend.text = element_text(size=20))
```

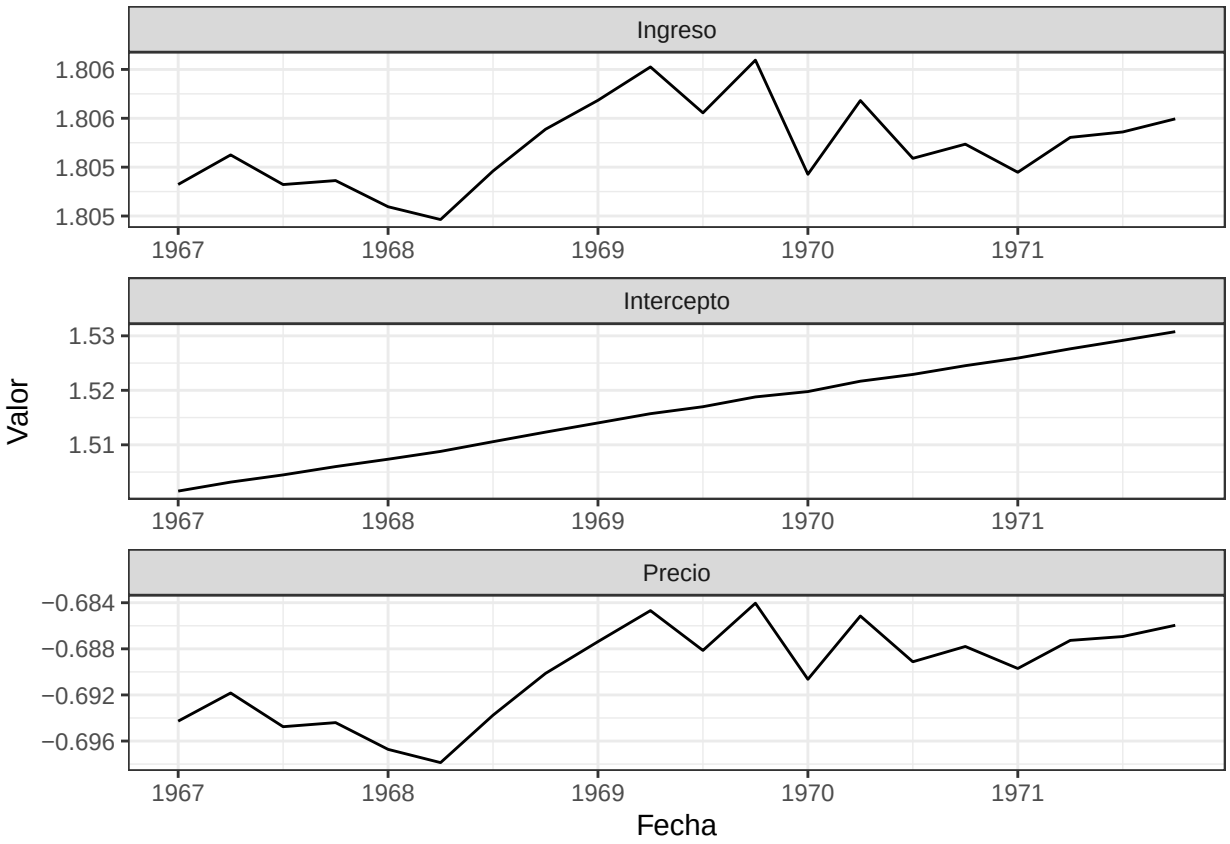



```
ggsave(filename = "graphs/teoria/V desc/actualizacion.png", width = 11.7, height = 6)
```

Se grafican la evolución de los parámetros.

```
df_params <- data.frame(reduce(res_dlm$mt, cbind) %>% t(), fecha = df_graficas$fecha)  %>%
  rename(Intercepto = X1, Ingreso = X2, Precio = X3) %>%
  pivot_longer(names_to = "parametro", values_to = "valor",
               cols = c(Intercepto, Ingreso, Precio))
```

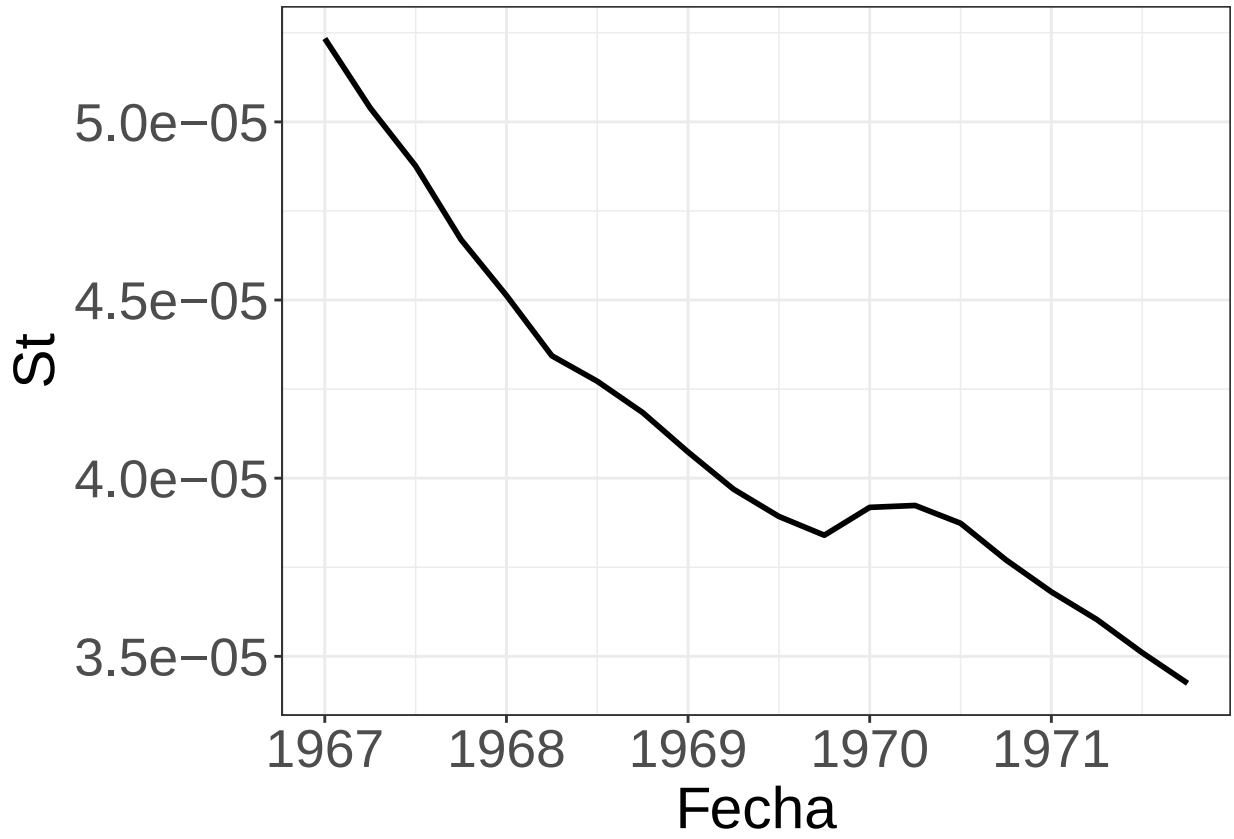
```
ggplot(df_params, aes(x=fecha, y = valor)) +
  geom_line() +
  facet_wrap(~parametro, nrow = 3, scales = "free") +
  theme_bw()  +
  ylab("Valor") +
  xlab("Fecha")
```



Se grafica la evolución de la estimación de la varianza de observación.

```
df_St <- data.frame(res_dlm$St %>% unlist(), fecha = df_graficas$fecha) %>%
  rename(St = 1)
```

```
ggplot(df_St, aes(x=fecha, y = St)) +
  geom_line(size = 1) +
  theme_bw()  +
  ylab("St") +
  xlab("Fecha") +
  theme(axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20),
        axis.title = element_text(size = 22),
        legend.text = element_text(size=20))
```




```
ggsave(filename = "graphs/teoria/V desc/St.png", width = 11.7, height = 6)
```

Distribuciones filtradas

Se supone que pasaron los años y nos encontramos en $t = 39$. El objetivo es obtener las distribuciones filtradas de los estados en $t = 38$ dada la información en $t = 39$.

```
#Se supondrá que estaremos en el observacion t=39
C38 <- res_dlm$Ct[[19]]
C39 <- res_dlm$Ct[[20]]
R39 <- res_dlm$Rt[[20]]
G39 <- G20
m39 <- res_dlm$mt[[20]]
m38 <- res_dlm$mt[[19]]
a39 <- res_dlm$at[[20]]
S39 <- res_dlm$St[[20]]
S38 <- res_dlm$St[[19]]

B38 <- C38 %*% t(G39) %*% solve(R39)
a39_menos_1 <- m38 + B38 %*% (m39 - a39)
R39_menos_1 <- C38 + B38 %*% (C39 - R39) %*% t(B38)
a39_menos_1
```

```
##          [,1]
## [1,]  1.5292
```

```
## [2,]  1.8059
## [3,] -0.6869
```

```
S39/S38 *R39_menos_1
```

```
##             [,1]       [,2]       [,3]
## [1,]  1.660e-04  2.918e-06 -4.293e-05
## [2,]  2.918e-06  1.545e-04 -2.242e-04
## [3,] -4.293e-05 -2.242e-04  3.359e-04
```

Ahora se obtiene la respuesta media en $t = 38$ dada la información en $t = 39$.

```
freeny[38,]
```

```
##            y lag.quarterly.revenue price.index income.level market.potential
## 1971.5 9.775                 9.749       4.278        6.194            13.16
```

```
F38 <- c(1, 6.19377, 4.27839)
t(F38) %*% a39_menos_1
```

```
##       [,1]
## [1,] 9.775
```

```
S39/S38 * (t(F38) %*% R39_menos_1 %*% F38)
```

```
##           [,1]
## [1,] 3.149e-05
```

Se define una función para calcular dsitribuciones filtradas más atrás de un periodo pasado.

```
suavizamiento_V_desc <- function(datos, G, nt){

  lista_ft_k_filt <- list()
  lista_at_k_filt <- list()
  lista_resp_med_esc <- list()
  lista_CI_inf <- list()
  lista_CI_sup <- list()

  #Se calculan las distribuciones filtradas para k = T
  at_k_filt_mas_1 <- res_dlm$mt[[length(datos$y)]]
  Rt_k_filt_mas_1 <- res_dlm$Ct[[length(datos$y)]]
  St <- res_dlm$St[[length(datos$y)]]

  Ft <- as.numeric(datos[length(datos$y), 2:4])
  lista_ft_k_filt[[length(datos$y)]] <- t(Ft) %*% at_k_filt_mas_1
  lista_at_k_filt[[length(datos$y)]] <- at_k_filt_mas_1
  lista_resp_med_esc[[length(datos$y)]] <- t(Ft) %*% Rt_k_filt_mas_1 %*% Ft
  lista_CI_inf[[length(datos$y)]] <- qst(0.025, nu = nt, mu = lista_ft_k_filt[[length(datos$y)]],
                                         sigma = sqrt(lista_resp_med_esc[[length(datos$y)]]))
  lista_CI_sup[[length(datos$y)]] <- qst(0.975, nu = nt, mu = lista_ft_k_filt[[length(datos$y)]],
```

```
                                        sigma = sqrt(lista_resp_med_esc[[length(datos$y)]]))

  for(i in length(datos$y):2){

    #Se calculan las distribuciones filtradas para k = i-1
    Ct_k <- res_dlm$Ct[[i-1]]
    Rt_k_mas_1 <- res_dlm$Rt[[i]]
    mt_k <- res_dlm$mt[[i-1]]
    at_k_mas_1 <- res_dlm$at[[i]]
    St_k_mas_1 <- res_dlm$St[[i]]
    St_k <- res_dlm$St[[i-1]]

    Bt_k <- Ct_k %*% t(G) %*% solve(Rt_k_mas_1)
    at_k_filt <- mt_k + Bt_k %*% (at_k_filt_mas_1 - at_k_mas_1)
    Rt_k_filt <- Ct_k + Bt_k %*% (Rt_k_filt_mas_1 - Rt_k_mas_1) %*% t(Bt_k)
    params_esc <- (St/St_k)*Rt_k_filt

    Ft_k <- as.numeric(datos[i-1, 2:4])
    ft_k_filt <- t(Ft_k) %*% at_k_filt
    resp_med_esc <- (St/St_k)*(t(Ft_k) %*% Rt_k_filt %*% Ft_k)
    CI <- c(qst(0.025, nu = nt, mu = ft_k_filt, sigma = sqrt(resp_med_esc)),
            qst(0.975, nu = nt, mu = ft_k_filt, sigma = sqrt(resp_med_esc)))

    lista_ft_k_filt[[i-1]] <- ft_k_filt
    lista_at_k_filt[[i-1]] <- at_k_filt
    lista_resp_med_esc[[i-1]] <- resp_med_esc
    lista_CI_inf[[i-1]] <- CI[1]
    lista_CI_sup[[i-1]] <- CI[2]

    at_k_filt_mas_1 <- at_k_filt
    Rt_k_filt_mas_1 <- Rt_k_filt

  }

  return(list("ft_k_filt" = lista_ft_k_filt, "at_k_filt" = lista_at_k_filt,
              "resp_media_esc" = lista_resp_med_esc, "CI_inf" = lista_CI_inf,
              "CI_sup" = lista_CI_sup))
}
```

Se aplica la función dada la información en $t = 39$.

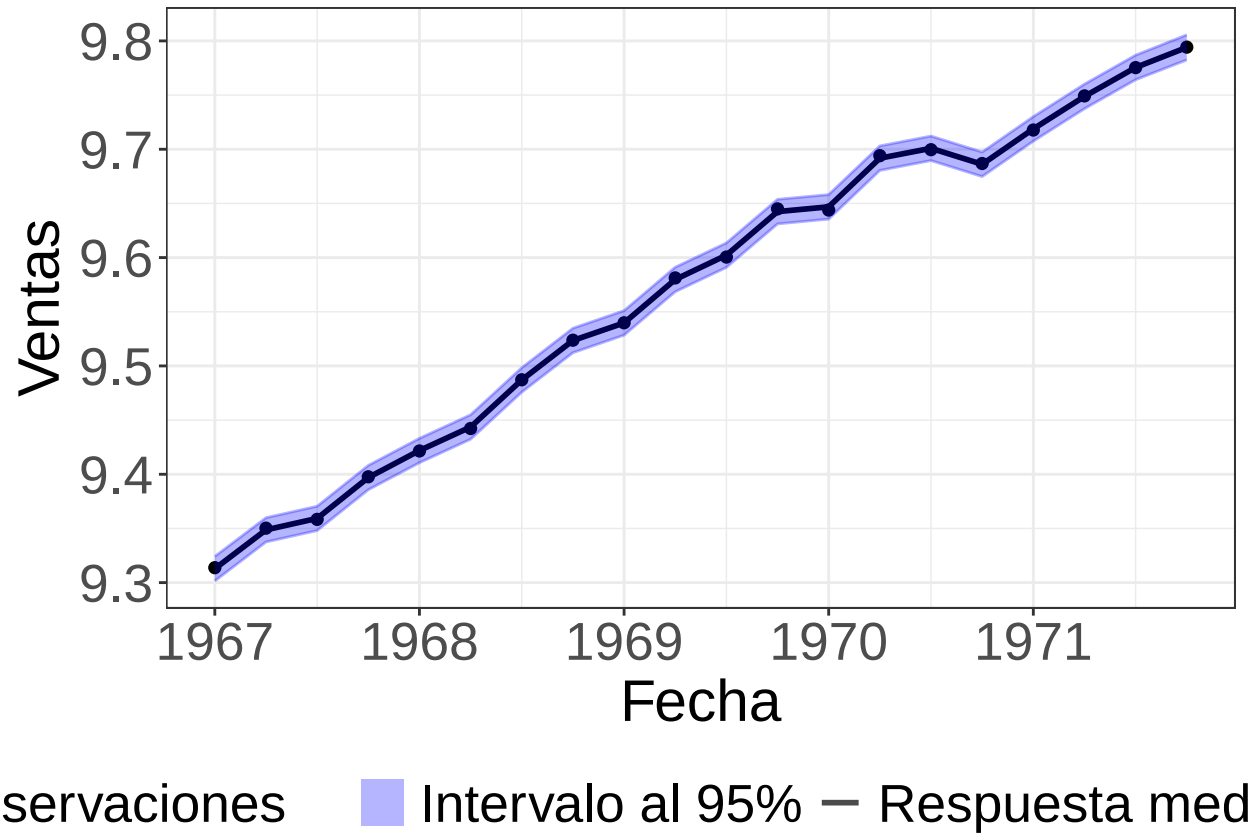
```
suav_dlm <- suavizamiento_V_desc(datos_ej, G39, 39.5)
```

Se grafican los resultados.

```
df_graficas_suav <- data.frame("fecha" = datos_ej %>% row.names(), "y_real" = datos_ej$y,
            "respuesta_media" = suav_dlm$ft_k_filt %>% unlist(), "CI_inf" = suav_dlm$CI_inf %>% unlist()
            "CI_sup" = suav_dlm$CI_sup %>% unlist()) %>%
  mutate(fecha = as.numeric(fecha))
```

```
ggplot(data = df_graficas_suav, aes(x = fecha)) +
  geom_point(aes(y = y_real, shape = "Observaciones"), size = 2) +
```

```
  geom_line(aes(y = respuesta_media, color = 'Respuesta media suavizada'), size = 1) +
  geom_line(aes(y = CI_inf), color = "blue", alpha = 0.3) +
  geom_line(aes(y = CI_sup), color = "blue", alpha = 0.3) +
  geom_ribbon(aes(ymax = CI_sup, ymin = CI_inf, fill = 'Intervalo al 95%'), alpha = 0.3) +
  theme_bw() +
  scale_colour_manual(
    name = "", values = c("Intervalo al 95%" = "transparent",
                          "Respuesta media suavizada" = "black")) +
  scale_fill_manual(
    name = "",  values = c("Intervalo al 95%" = "blue",
                           "Respuesta media suavizada" = "transparent")) +
  theme(legend.position = "bottom") +
  labs(shape = "") +
  ylab('Ventas') +
  xlab('Fecha') +
  theme(axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20),
        axis.title = element_text(size = 22),
        legend.text = element_text(size=20))
```




```
ggsave(filename = "graphs/teoria/suavizamiento/suavizamiento.png", width = 11.7, height = 6)
```

```
df_params_suav <- data.frame(reduce(suav_dlm$at_k_filt, cbind) %>% t(), fecha = df_graficas_suav$fecha) %>%
  rename(Intercepto = X1, Ingreso = X2, Precio = X3) %>%
  pivot_longer(names_to = "parametro", values_to = "valor",
               cols = c(Intercepto, Ingreso, Precio))
```

```r
ggplot(df_params_suav, aes(x=fecha, y = valor)) +
  geom_line() +
  facet_wrap(~parametro, nrow = 3, scales = "free") +
  theme_bw()  +
  ylab("Valor") +
  xlab("Fecha")
```