Simulación

2. Generación de variables aleatorias.
2.3 Procesos Estocásticos III: Series de tiempo

Jorge de la Vega Góngora

Departamento de Estadística,
Instituto Tecnológico Autónomo de México

Clase 8

Series de tiempo con modelos ARIMA

Ruido blanco I

Haremos un breve repaso de la notación necesaria para simular series de tiempo tipo ARIMA.

- Una serie de tiempo $\{y_t\}$ es **estacionaria** si cumple lo siguiente:
  - tiene media constante μ (no depende del tiempo t):

$$E(y_t) = \mu \quad \forall t$$

  - Si las autocovarianzas de la serie (y por lo tanto las autocorrelaciones) no dependen de t, sino solo del número de rezagos en la diferencia:

$$Cov(y_t, y_{t-j}) = \gamma_j.$$

- Podemos definir la autocorrelación (poblacional) en términos de las autocovarianzas como $\rho_j = \frac{\gamma_j}{\gamma_0}$.
- En términos prácticos, una serie es estacionaria si no hay crecimiento o decrementos en los datos. Los datos deben fluctuar alrededor de una media constante, y la varianza no debe de aumentar o disminuir con el tiempo.

Ruido blanco II

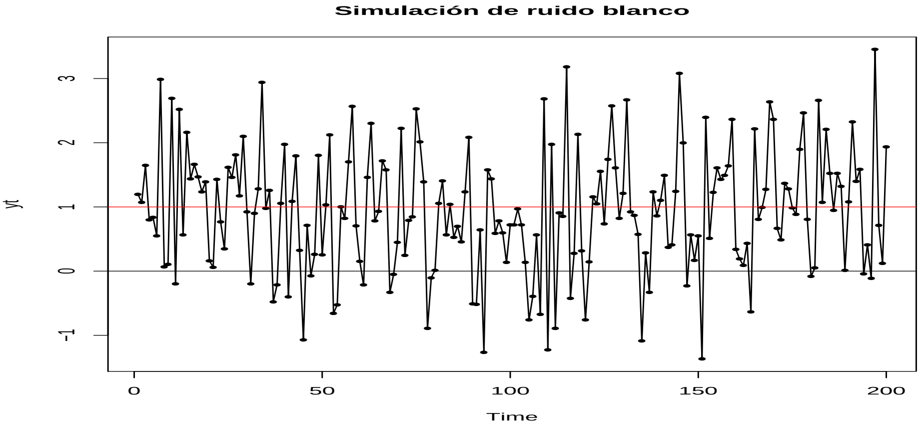
- Un ejemplo de un proceso estacionario es el *ruido blanco* que es un modelo en el que cada observación se compone de dos partes: un nivel constante c y un componente de error ϵ_t que es independiente de cualquier periodo,

$$y_t = c + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}\left(0, \sigma^2\right)$$

Se simula usando la función `rnorm`.

```
n <- 200 #longitud de la serie
yt <- 1 + rnorm(n)
plot.ts(yt, type = "o", main = "Simulación de ruido blanco", pch = 16, cex = 0.7)
abline(h = c(0,1), col = c("black", "red"))
```

Ruido blanco III



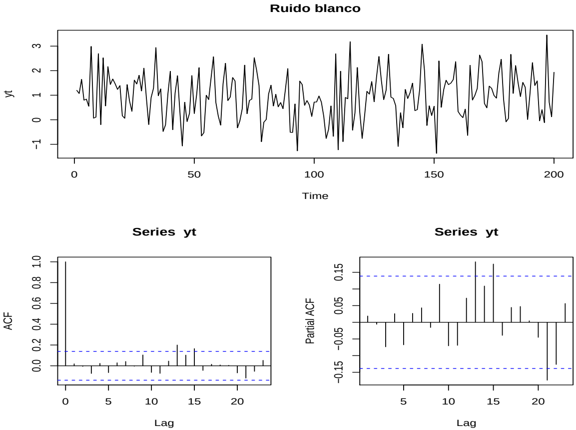
- Cuando la serie de tiempo es un ruido blanco, las propiedades de la función de autocorrelación son bien conocidas, y esto nos puede ayudar a identificarlo.

Ruido blanco IV

- Si $\{\epsilon_t\}$ es ruido blanco, entonces $\hat{\rho}_k \sim N(0, 1/n)$ para $k > 0$, donde n es el número de observaciones en la serie.
- Así que 95 % de los coeficientes de autocorrelación $\hat{\rho}_k$ deben estar entre $\pm 1.96/\sqrt{n}$, que son los límites críticos incluídos en las gráficas.
- También las autocorrelaciones parciales deben ser cercanas a 0 cuando el modelo es un modelo de ruido blanco.

```
layout(matrix(c(1, 1, 2, 3), nrow = 2, byrow = T)) #Acomodo de las gráficass
plot.ts(yt, main = "Ruido blanco")
acf(yt)
pacf(yt)
```

Ruido blanco V

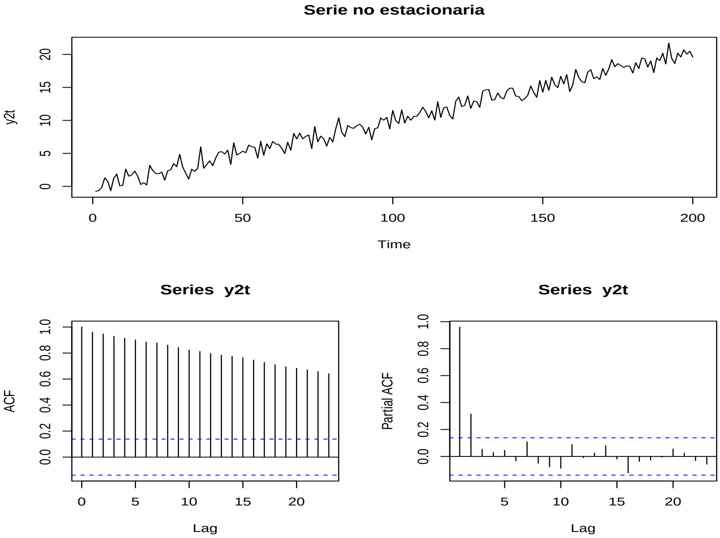


Identificando estacionariedad I

- Necesitamos un mecanismo para verificar si una serie es estacionaria, que es a través de las funciones que definimos: `acf` y `pacf`
- Las autocorrelaciones $\hat{\rho}_k$ de datos estacionarios tienden a 0 relativamente rápido, mientras que para series no estacionarias, los coeficientes son significativamente diferentes de 0 para varios rezagos.
- Por ejemplo, la siguiente serie no es estacionaria, y lo podemos ver en sus funciones de autocorrelación:

```
layout(matrix(c(1, 1, 2, 3), nrow = 2, byrow = T))
y2t <- 1:n/10 + rnorm(n)
plot.ts(y2t, main = "Serie no estacionaria")
acf(y2t)
pacf(y2t)
```

Identificando estacionariedad II



Series no estacionarias

- Si una serie es no estacionaria, lo que es relativamente más fácil de identificar, se puede transformar en estacionaria tomando diferencias de distintos órdenes de la serie original:
  - 1a. dif: $\Delta y_t = y_t - y_{t-1}$
  - 2a. dif: $\Delta^2 y_t = \Delta(\Delta y_t) = \Delta y_t - \Delta y_{t-1} = y_t - 2y_{t-1} + y_{t-2}$
  - 3a. dif: $\Delta^2 y_t = \Delta(\Delta^2 y_t) = \Delta(y_t - 2y_{t-1} + y_{t-2}) = \ldots$
- A continuación veremos los componentes típicos de las series en los modelos de Box-Jenkins.

Modelos de promedios móviles (MA) I

- Un *modelo de promedios móviles* de orden q es un modelo de la forma:

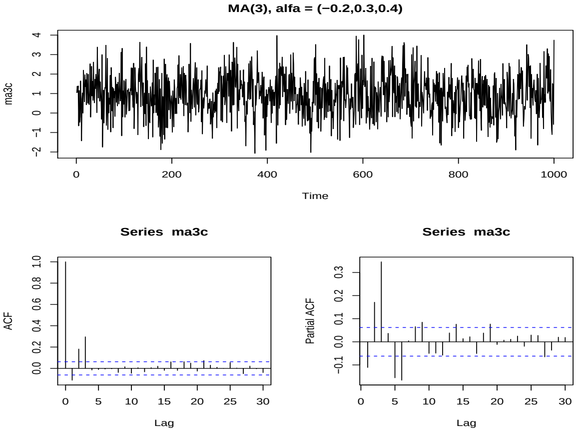
$$y_t = \mu + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t$$

  donde $\{\epsilon_t\}$ es una serie de ruido blanco.
- Cuando $q = \infty$, a la serie se le llama un *filtro lineal*. A este tipo de procesos se le llama promedio móvil porque es una especie de promedio móvil del ruido blanco $\{\epsilon_t\}$, con pesos dados por los coeficientes $\theta_1, \theta_2, \ldots$.
- Para simular un proceso MA(3), por ejemplo, de longitud n, generamos una serie de ruido blanco y consideramos una fórmula recursiva:

```
n <- 1000                      # longitud del proceso
theta <- c(-0.2, 0.3, 0.4)     # vector de coeficientes del promedio móvil
mu <- 1                        # media
eps <- rnorm(n+3)              # ruido blanco
ma3 <- NULL
for(i in 4:(n+3))              # empezamos en 4 para considerar los rezagos
                ma3[i] <- mu + theta[1]*eps[i-1] + theta[2]*eps[i-2] + theta[3]*eps[i-3] + eps[i]

ma3c <- ma3[4:(n+3)] #corrige los índices del proceso
layout(matrix(c(1, 1, 2, 3), nrow = 2, byrow = T))
plot.ts(ma3c, main = paste0("MA(", length(theta), "), alfa = (", paste(theta, collapse = ","),")"))
acf(ma3c)
pacf(ma3c)
```

Modelos de promedios móviles (MA) II



Modelos autoregresivos (AR) I

- Un modelo autorregresivo de orden p, que se denota por $AR(p)$ es un modelo de la forma:

$$y_t = \mu + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

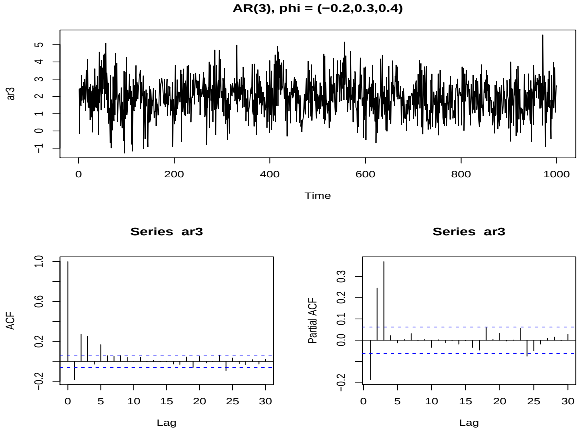
  donde $\epsilon_t \sim N(0, \sigma_\epsilon^2)$, y los errores son independientes. En este modelo los predictores de la observación y_t son sus propios rezagos en el tiempo.
- Los modelos AR pueden ser estacionarios o no estacionarios, dependiendo de las restricciones que se impongan sobre los pesos del modelo.
- La función de autocorrelación de un $AR(1)$ es $\rho_k = \phi_1^k, \quad k = 0, 1, 2, \ldots$.
- Por ejemplo, consideremos simular un proceso $AR(3)$ a continuación

Modelos autoregresivos (AR) II

```
n <- 1000                 # longitud del proceso
phi <- c(-0.2,0.3,0.4)    # vector de coeficientes del promedio móvil
mu <- 1                   # media
eps <- rnorm(n+3)         # ruido blanco
ar3    <- mu + eps[1]     # inicializa el proceso
ar3[2] <- mu + phi[1]*ar3[1] + eps[2]
ar3[3] <- mu + phi[1]*ar3[2] + phi[2]*ar3[1] + eps[3]
for(i in 4:n) ar3[i] <- mu + phi[1]*ar3[i-1] + phi[2]*ar3[i-2] + phi[3]*ar3[i-3] + eps[i]

layout(matrix(c(1,1,2,3), nrow = 2, byrow = T))
plot.ts(ar3, main = paste0("AR(", length(phi),"), phi = (",paste(phi, collapse = ","),")"))
acf(ar3)
pacf(ar3)
```

Modelos autoregresivos (AR) III



Operador rezago I

- Definan al operador rezago como la función L (de lag) tal que
$$La_t = a_{t-1}$$
En general,
$$L^j a_t = a_{t-j}.$$
- Los modelos AR y MA pueden simplificarse usando esta notación, ya que podemos escribirlos como *polinomios* en el operador L.
- Un polinomio en L de orden k es de la forma
$$\Phi_k(L) = \phi_0 L^0 + \phi_1 L^1 + \cdots \phi_k L^k,$$
donde $L^0 = 1$.
- Simplificando notación:
    - Un $MA(q)$, se puede escribir como $y_t = \mu + \Theta_q(L)\epsilon_t$, donde
$$\Theta_q(L) = \theta_0 L^0 + \theta_1 L^1 + \theta_2 L^2 + \cdots + \theta_q L^q \text{ y } \theta_0 = 1.$$

Operador rezago II

- Un $AR(p)$ se puede escribir como $\Phi_p(L)y_t = \alpha_0 + \epsilon_t$ donde

$$\Phi_p(L) = 1 - \phi_1 L^1 - \phi_2 L^2 - \cdots - \phi_p L^p.$$

Modelos ARMA y ARIMA I

- Los modelos ARMA combinan modelos autorregresivos con modelos de promedios móviles para llegar a modelos más generales.
- Un modelo $ARMA(p, q)$ se escribe, usando notación de polinomios, como

$$\Phi_p(L)y_t = \mu + \Theta_q(L)\epsilon_t$$

- Los modelos ARMA sólo se pueden aplicar a series estacionarias.
- Para extender los modelos a series no estacionarias, se requiere diferenciar la serie. Esto da origen a los modelos ARIMA.
- Un modelo $ARIMA(p, d, q)$ es un modelo $ARMA(p, q)$ en donde la serie original y_t se reemplaza por la serie diferenciada $\Delta^d y_t$.
- En la práctica, d usualmente toma valores en $\{0, 1, 2\}$ y p y q toman valores no mayores a 4, aunque esto no es una regla.
- Los modelos AR y MA son casos particulares de modelos ARIMA, ya que por ejemplo, un $AR(p)$ es lo mismo que un $ARIMA(p, 0, 0)$, o un $MA(q)$ es un $ARIMA(0, 0, q)$.

Simulación de modelos ARIMA con la función `arima.sim` I

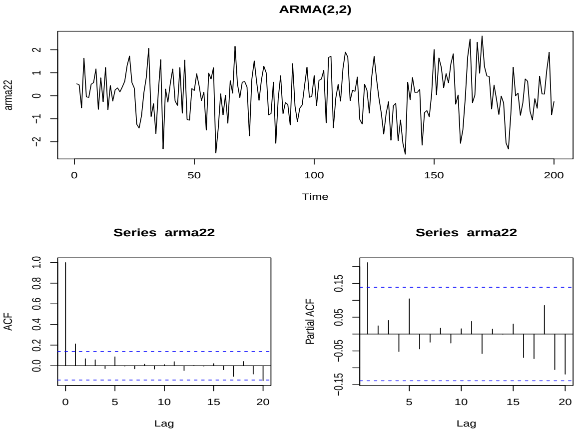
- Vemos que simular procesos ARIMA puede ser complejo, ya que hay que considerar ecuaciones recursivas, rezagos y diferencias. Para simplificar el procedimiento de simulación, se puede usar la función `arima.sim`. Esta función supone que no hay constante en el modelo ($\mu = 0$).
- Por ejemplo, para simular un proceso $ARMA(2,2)$, se usa la siguiente sintaxis:

```r
n <- 200
orden  <- c(2,2)
phis   <- c( 0.9, -0.2)  # coeficientes parte AR
thetas <- c(-0.7,  0.1)  # coeficientes parte MA

arma22 <- arima.sim(model = list(ar = phis, ma = thetas), n = n)

layout(matrix(c(1,1,2,3),nrow=2,byrow=T))
plot.ts(arma22, main = paste0("ARMA(", paste(orden, collapse = ","),")"))
acf(arma22, lag.max = 20)
pacf(arma22, lag.max = 20)
```

Simulación de modelos ARIMA con la función `arima.sim` II

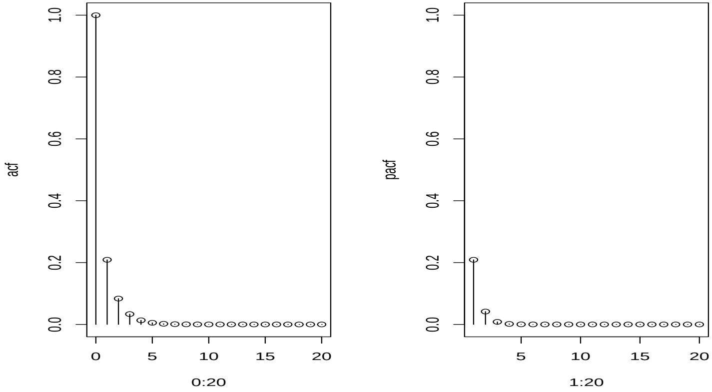


Simulación de modelos ARIMA con la función `arima.sim` III

- ¿Cómo debería ser la función *teórica* de un acf y pacf de un proceso $ARMA(2,2)$? Podemos verla con la función **`ARMAacf`** (no hay una función equivalente para ARIMAs; habría que integrar la serie):

```
par(mfrow=c(1,2))
plot(0:20, ARMAacf(ar = phis, ma= thetas, lag.max = 20, pacf = F),ylab="acf")
segments(x0 = 0:20, y0 = rep(0,21), y1 = ARMAacf(ar = phis, ma= thetas, lag.max = 20, pacf = F))
plot(1:20, ARMAacf(ar = phis, ma= thetas, lag.max = 20, pacf = T), ylab="pacf",ylim=c(0,1))
segments(x0=1:20, y0 =rep(0,20), y1 = ARMAacf(ar = phis, ma= thetas, lag.max = 20, pacf = T))
```

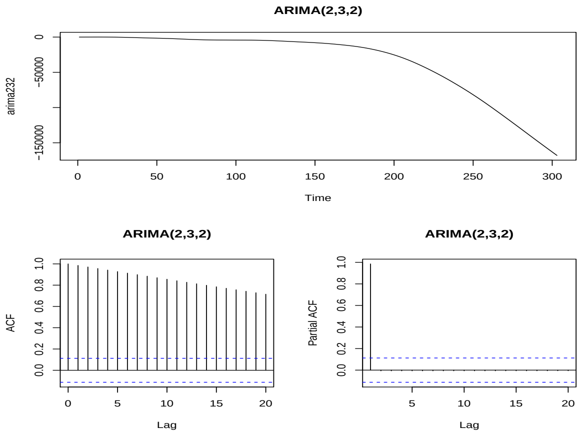
Simulación de modelos ARIMA con la función `arima.sim` IV



- Un $ARIMA(2,3,2)$ se puede simular de la siguiente manera:

```
n <- 300
orden <- c(2,3,2) # vector con los coeficientes (p, d, q)
arima232 <- arima.sim(model = list(order = orden, ar = c(0.8, -0.2), ma = c(-0.7, 0.1)),  n = n)

layout(matrix(c(1,1,2,3),nrow=2,byrow=T))
plot.ts(arima232, main = paste0("ARIMA(",paste(orden,collapse=","),")"))
acf(arima232, main = paste0("ARIMA(",paste(orden,collapse=","),")"), lag.max = 20)
pacf(arima232, main = paste0("ARIMA(",paste(orden,collapse=","),")"), lag.max = 20)
```



- Un ejemplo dinámico de un proceso ARIMA a partir de primeros principios se puede visualizar en este link.

Simulación de modelos ARIMA con la función `arima.sim` VII

- Ahora consideraremos algunos ejemplos con más detalle de varios de estos procesos y de sus funciones de autocorrelación y autocorrelación parcial.

Casos particulares I

Es posible identificar los parámetros de un modelo ARIMA a través de conocer sus funciones `acf` y `pacf` teóricas. Aplicaremos lo que veamos a series de tiempo reales para ver cómo se procedería, a grandes rasgos.

- $AR(1)$
  - El modelo es $y_t = \mu + \phi_1 y_{t-1} + \epsilon_t$.
  - En lo que sigue, supondremos que la media de y_t es $E(y_t) = \xi$.
  - Para que el proceso sea estacionario, se requiere que $|\phi_1| < 1$.
  - La media, autocovarianza y autocorrelación son, respectivamente:

$$\begin{aligned}
\xi &= E(y_t) = \frac{\mu}{1-\phi_1}, \\
\gamma_k &= \phi_1^k \frac{\sigma_\epsilon^2}{1-\phi_1^2} \text{ para } k = 0, 1, 2, \ldots \\
\rho_k &= \phi_1^k \text{ para } k = 0, 1, 2, \ldots.
\end{aligned}$$

  - La acf decrece exponencialmente cuando $\phi_1 > 0$ y oscila decayendo exponencialmente cuando $\phi_1 < 0$.

Casos particulares II

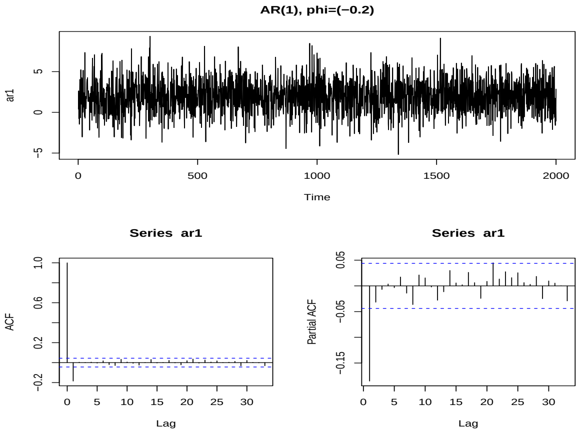
- La pacf tiene un pico en el rezago 1, luego es 0; el pico es positivo si $\phi_1 > 0$, negativo si $\phi_1 < 0$.

Ejemplo. [$y_t = 2 + 0.2y_{t-1} + \epsilon_t$ con $\sigma_\epsilon = 2$]

```
phi <- -0.2
ar1 <- arima.sim(model = list(ar = phi),n = 2000, sd = 2) + 2

layout(matrix(c(1,1,2,3), nrow = 2, byrow = T))
plot.ts(ar1, main=paste0("AR(",length(phi),"), phi=(",paste(phi,collapse = ","),")"))
acf(ar1)
pacf(ar1)
```

Casos particulares III




```
#Evaluamos los parámetros del modelo:
mean(ar1)
```

```
[1] 1.995581
```

Casos particulares IV

- $AR(2)$
  - El modelo es de la forma: $y_t = \mu + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \epsilon_t$.
  - Para que el proceso sea estacionario, se requiere que se cumplan estas ecuaciones simultáneamente:

$$\begin{aligned} \phi_1 + \phi_2 &< 1 \\ \phi_2 - \phi_1 &< 1 \\ |\phi_2| &< 1. \end{aligned}$$

  - La media del proceso es $\xi = E(y_t) = \frac{\mu}{1-\phi_1-\phi_2}$.
  - El cálculo de la autocovarianza teórica es complicado. Las fórmulas son recursivas (se conocen como ecuaciones de Yule-Walker):

$$\gamma_k = \begin{cases} \phi_1\gamma_1 + \phi_2\gamma_2 + \sigma_\epsilon^2 & \text{si } k = 0 \\ \phi_1\gamma_{k-1} + \phi_2\gamma_{k-2} & \text{si } k > 0 \end{cases}$$

  - Las autocorrelaciones se obtienen usando la ecuación $\rho_k = \phi_1\rho_{k-1} + \phi_2\rho_{k-2}$, para $k \geq 3$.

Casos particulares V

- Los valores de ρ_1 y ρ_2 se obtienen de resolver el sistema de ecuaciones (una vez que se conocen los valores de ϕ_1 y ϕ_2):

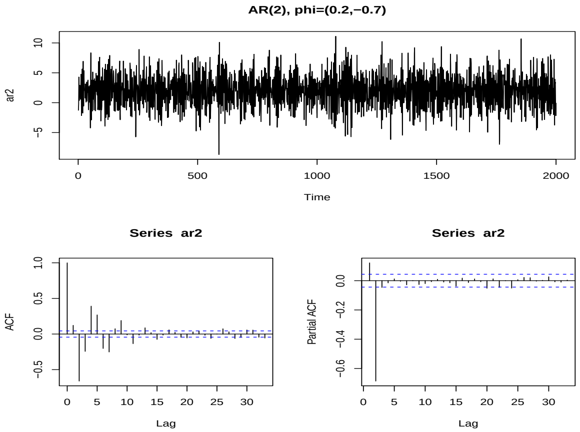
$$\begin{aligned}\rho_1 &= \phi_1 + \phi_2\rho_1 \\ \rho_2 &= \phi_1\rho_1 + \phi_2\end{aligned}$$

- Si $\phi_1^2 + 4\phi_2 \geq 0$, la acf es una mezcla de exponenciales decrecientes, y si $\phi_1^2 + 4\phi_2 < 0$, la autocorrelación es una onda sinusoidal decreciente.
- la pacf tiene picos en los rezagos 1 y 2, luego son ceros.

Ejemplo. [** $y_t = 2 + 0.2y_{t-1} - 0.7y_{t-2} + \epsilon_t$ **con** $\sigma_\epsilon = 2$**]

```
phi <- c(0.2,-0.7)
ar2 <- 2 + arima.sim(model = list(ar = phi),n = 2000, sd = 2)
layout(matrix(c(1,1,2,3), nrow = 2, byrow = T))
plot.ts(ar2, main=paste0("AR(",length(phi),"), phi=(",paste(phi,collapse = ","),")"))
acf(ar2)
pacf(ar2)
```

Casos particulares VI




```r
#Evaluamos los parámetros del modelo:
mean(ar2)
```

```
[1] 2.006581
```

Casos particulares VII

- $MA(1)$
  - El modelo es $y_t = \mu + \epsilon_t - \theta_1 \epsilon_{t-1}$.
  - El proceso es estacionario para cualquier valor de θ_1.
  - La media, autocovarianza y autocorrelación son, respectivamente:

$$\begin{aligned} E(y_t) &= \mu \\ \gamma_0 &= \sigma_\epsilon^2(1+\theta_1^2) \\ \rho_k &= \begin{cases} -\frac{\theta_1}{1+\theta_1^2} & \text{si } k = 1 \\ 0 & \text{si } k > 1 \end{cases} \end{aligned}$$

  - La acf tiene un pico en $k = 1$ y después es 0, positivo si $\theta_1 > 0$, negativo si $\theta_1 < 0$

Casos particulares VIII

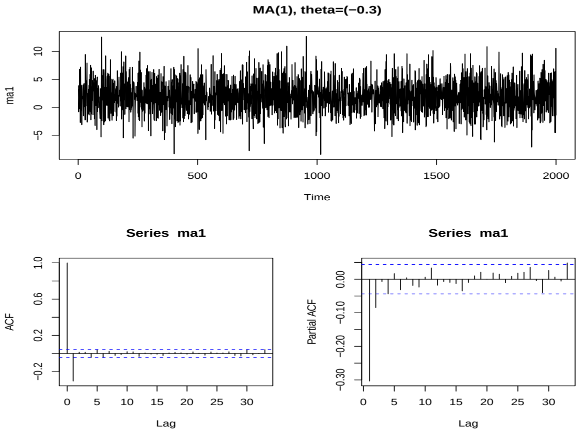
- La pacf tiene Decae exponencialmente: del lado negativo si $\theta_1 < 0$ y alternando en signo empezando en el lado positivo si $\theta_1 > 0$.

Ejemplo. [Ejemplo:** $y_t = 2 + \epsilon_t + 0.3\epsilon_{t-1}$ **y** $\sigma_\epsilon = 3$**]

```
theta <- c(-0.3)
ma1 <- 2 + arima.sim(model = list(ma = theta),n = 2000, sd = 3)

layout(matrix(c(1,1,2,3), nrow = 2, byrow = T))
plot.ts(ma1, main=paste0("MA(",length(theta),"), theta=(",paste(theta,collapse = ","),")"))
acf(ma1)
pacf(ma1)
```

Casos particulares IX




```r
#Evaluamos los parámetros del modelo:
mean(ma1)
```

```
[1] 1.971331
```

Casos particulares X

- $MA(2)$
  - El modelo es $y_t = \mu + \epsilon_t - \theta_1\epsilon_{t-1} - \theta_2\epsilon_{t-2}$.
  - El proceso es estacionario para cualquier valor de θ_1 y θ_2.
  - La media, autocovarianza y autocorrelación son, respectivamente:

$$
\begin{aligned}
E(y_t) &= \mu \\
\gamma_0 &= \sigma_\epsilon^2(1+\theta_1^2+\theta_2^2) \\
\rho_k &= \begin{cases} -\frac{\theta_1(1-\theta_2)}{1+\theta_1^2+\theta_1^2} & \text{si } k=1 \\ -\frac{\theta_2}{1+\theta_1^2+\theta_1^2} & \text{si } k=2 \\ 0 & \text{si } k>2 \end{cases}
\end{aligned}
$$

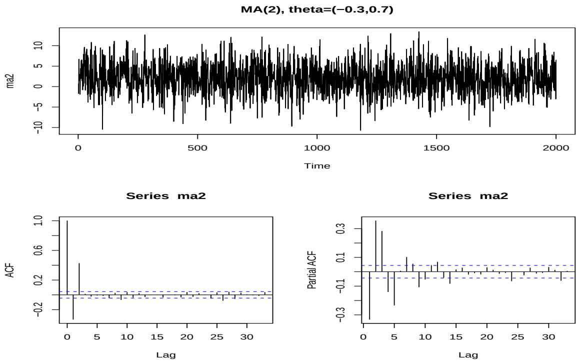
  - La acf tiene un picos en los rezagos 1 y 2 y después es 0.

- Decae exponencialmente en forma de función sinoidal decreciente. El patrón exacto depende de los signos y tamaños de θ_1 y θ_2.

***Ejemplo.* [$y_t = 2 + \epsilon_t + 0.3\epsilon_{t-1} - 0.7\epsilon_{t-2}$ *y* $\sigma_\epsilon = 3$]**

```
theta <- c(-0.3,0.7)
ma2 <- 2 + arima.sim(model = list(ma = theta),n = 2000, sd = 3)
layout(matrix(c(1,1,2,3), nrow = 2, byrow = T))
plot.ts(ma2, main = paste0("MA(",length(theta),"), theta=(",paste(theta,collapse = ","),")"))
acf(ma2)
pacf(ma2)
```

Casos particulares XII




```r
#Evaluamos los parámetros del modelo:
mean(ma2)
```

```
[1] 2.021978
```

Modelos de Markov ocultos (HMM)

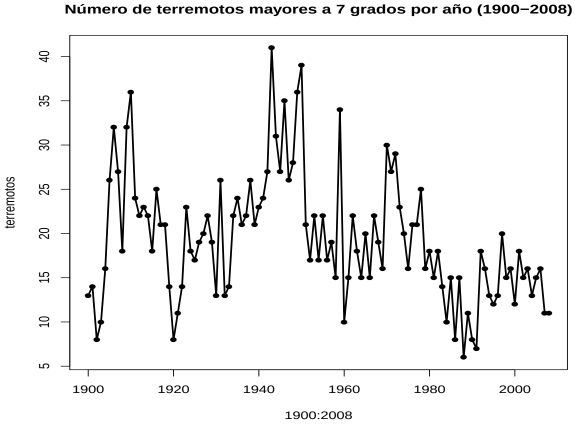
Conceptos generales de HMM

- Los modelos de Markov ocultos (HMMs) son modelos en los cuales la distribución que genera una observación depende del estado de un proceso de Markov subyacente no observable.
- La distribución marginal de un HMM es una distribución de mezclas. Los modelos de mezclas permiten acomodar heterogeneidad no observada en una población, cuando la población consiste en grupos no observados que tienen distintas distribuciones.
- Los modelos ARMA no son apropiados para todos los tipos de datos, porque se basan en la distribución normal.

Ejemplo 1 I

- La siguiente serie corresponde al número de terremotos de magnitud 7 o mayor en el mundo, de 1900 a 2008

```
terremotos <- c(13, 14, 8, 10, 16, 26, 32, 27, 18, 32, 36, 24, 22, 23, 22, 18, 25, 21, 21, 14, 8, 11, 14, 23, 18, 17, 19, 20, 22, 19, 13, 26, 13,
                14, 22, 24, 21, 22, 26, 21, 23, 24, 27, 41, 31, 27, 35, 26, 28, 36, 39, 21, 17, 22, 17, 22, 17, 19, 15, 34, 10, 15, 22, 18, 15, 20,
                15, 22, 19, 16, 30, 27, 29, 23, 20, 16, 21, 21, 25, 16, 18, 15, 18, 14, 10, 15, 8, 15, 6, 11, 8, 7, 18, 16, 13, 12, 13, 20, 15, 16,
                12, 18, 15, 16, 13, 15, 16, 11, 11)
plot(1900:2008,terremotos,type="o",lwd=2, pch=16, main= "Número de terremotos mayores a 7 grados por año (1900-2008)")
```


Número de terremotos mayores a 7 grados por año (1900–2008)
terremotos
5
10
15
20
25
30
35
40
1900
1920
1940
1960
1980
2000
1900:2008

Ejemplo 1 III

- Supongan que cada conteo de terremotos es generado por uno de dos posibles distribuciones Poisson con medias λ_1 y λ_2, en donde la elección de la media depende de un mecanismo aleatorio.
- Por ejemplo, se selecciona λ_1 con probabilidad δ_1 y λ_2 con probabilidad δ_2. En general, si se tienen m grupos, podemos considerar:

$$P(X = x) = \sum_{i=1}^{m} P(X = x|L = i)P(L = i) = \sum_{i=1}^{m} f_i(x)\delta_i$$

Ejemplo 2 I

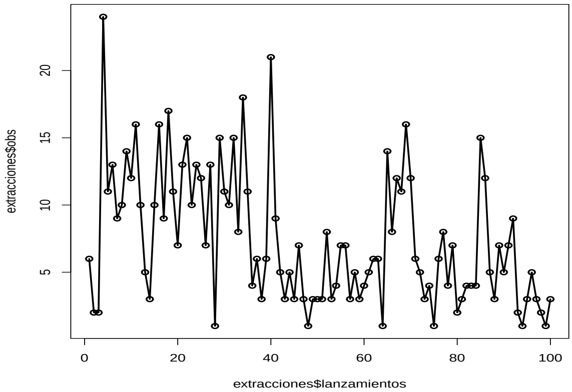
Consultar:
http://gradientdescending.com/hidden-markov-model-example-in-r-with-the-depmixs4-package/

- Consideren el siguiente ejemplo: se tienen dos dados y un jarrón lleno de monedas de 10 y 5 pesos. A tira los dados, y si la suma es mayor que 4, toma un puñado de monedas y lanza de nuevo los dados. Si la suma en el segundo tiro es igual a 2, toma otro puñado de monedas y le pasa los dados a B. en el turno de B, se aplican las mismas reglas, pero si los dados suman más de 4, toma un puñado de monedas, pero sólo se queda con las monedas de 10 pesos y las demás las regresa al jarrón. Se repite así hasta que se vacía el jarrón.
- Se espera que A tome más monedas que B
- Supongamos que no podemos ver quién está tirando los dados, sólo sabemos cuántas monedas son tomadas después del lanzamiento. Tampoco sabemos la denominación, sólo el número final de monedas que se sacaron del jarrón en ese turno. ¿Cómo podemos saber quién tiró los dados?

Ejemplo 2 II

- El estado es la persona que tiró los dados: A o B. La observación es cuántas monedas se extrajeron en ese turno. El lanzamiento de los dados y la condición de pasar los dados si el valor es menor que 4 es la probabilidad de transición. En este ejemplo, sabemos que la probabilidad de transición es $\frac{1}{12}$.

Simulación del Ejemplo 2 I



Estimación del HMM I

```
fit.hmm <- function(draws){

# HMM with depmix
mod <- depmix(obs ~ 1, data = draws, nstates = 2, family = poisson()) # use gaussian() for normally distributed data

fit.mod <- fit(mod)

# predict the states by estimating the posterior
est.states <- posterior(fit.mod)
head(est.states)

# results
tbl <- table(est.states$state, draws$estado)
draws$est.state.labels <- c(colnames(tbl)[which.max(tbl[1,])], colnames(tbl)[which.max(tbl[2,])])[est.states$state]
est.states$roll <- 1:100
colnames(est.states)[2:3] <- c(colnames(tbl)[which.max(tbl[1,])], colnames(tbl)[which.max(tbl[2,])])
hmm.post.df <- melt(est.states, measure.vars = c("B", "A"))

# print the table
print(table(draws[,c("estado", "est.state.labels")]))

# return it
return(list(draws = draws, hmm.post.df = hmm.post.df))
}
hmm1 <- fit.hmm(extracciones)
```

```
converged at iteration 8 with logLik: -265.4128
      est.state.labels
estado  A  B
     A 38  1
     B  1 60
```

Gráfica

```r
# plot output
plot.hmm.output <- function(model.output){
        g0 <- (ggplot(model.output$draws, aes(x = lanzamientos, y = obs)) + geom_line() +
                  theme(axis.ticks = element_blank(), axis.title.y = element_blank())) %>% ggplotGrob

        g1 <- (ggplot(model.output$draws, aes(x = lanzamientos, y = estado, fill = estado, col = estado)) +
                  geom_bar(stat = "identity", alpha = I(0.7)) +
                  scale_fill_manual(values = mycols, name = "Estado:\nPersona que \nlanzó los\ndados", labels = c("B", "A")) +
                  scale_color_manual(values = mycols, name = "Estado:\nPersona que \nlanzó los\ndados", labels = c("B", "A")) +
                  theme(axis.ticks = element_blank(), axis.text.y = element_blank()) +
                  labs(y = "Estado Real")) %>% ggplotGrob

        g2 <- (ggplot(model.output$draws, aes(x = lanzamientos, y = est.state.labels, fill = est.state.labels,
                                  col = est.state.labels)) +
                   geom_bar(stat = "identity", alpha = I(0.7)) +
                   scale_fill_manual(values = mycols, name = "Estado:\nPersona que \nlanzó los\ndados", labels = c("B", "A")) +
                   scale_color_manual(values = mycols, name = "Estado:\nPersona que \nlanzó los\ndados", labels = c("B", "A")) +
                   theme(axis.ticks = element_blank(), axis.text.y = element_blank()) +
                   labs(y = "Estado Estimado")) %>% ggplotGrob

        g3 <- (ggplot(model.output$hmm.post.df, aes(x = roll, y = value, col = variable)) + geom_line() +
                   scale_color_manual(values = mycols, name = "Estado:\nPersona que \nlanzó los\ndados", labels = c("B", "A")) +
                   theme(axis.ticks = element_blank(), axis.text.y = element_blank()) +
           labs(y = "Prob Posterior")) %>%
               ggplotGrob()

g0$widths <- g1$widths

return(grid.arrange(g0, g1, g2, g3, widths = 1, nrow = 4))
}
```

Resultado

```r
plot.hmm.output(hmm1)
```