Streaming Data Management and Time Series Analsis

Contents

Exponential Smoothing

La serie storica può essere essere pensata come somma o prodotto di 4 componenti:

$$y_t = T_t + C_t + S_t + A_t \quad \text{oppure} \quad y_t = T_t \cdot C_t \cdot S_t \cdot A_t$$

dove T_t è il trend che descrive la serie a lungo termine, C_t è il ciclo che è definto come componente di oscillazione periodica, S_t è la componente stagionale e cattura i movimenti ripetitivi e infine la parte A_t è parte accidentale. Oss: Il modello moltiplicativo può essere riportato al modello additivo attraverso un logaritmo.

Se le serie sono molto corte un modo molto semplice e spesso efficace è quello di effetuare una regressione usando come variabile esplicativa il tempo o dummy stagionali oppure una loro combinazione:

$$y_t = \beta_0 + \beta_1 t + ... + \beta_n t^n + \beta_{n+1} D^{(1)} + ... + \beta_{m+n} D^{(m)} + \varepsilon_t$$

Assumiamo la serie come come somma di componente trend che varia accidentalmente e una componenete casuale non autocorrelata, in questo caso dato un valore il valore successivo ha la stessa probabilità di essere sia sopra che sotto il valore attuale, quindi l'idea migliore sembrerebbe di dire che il valore successivo è il valore attuale. Ma si il valore successivo osservato è la somma del valore attuale con una componente casuale a media nulla, per elimianare tale valore si dovrebbe fare la media dei valori osservati fino all'istante attuale, ma siccome è una componente casuale non osservabile, si cerca un compromesso che è raggiunto dal modello *exponential smoothing* che pesa più gli ultimi valori che i valori più remoti, in particolare usa pesi esponenzialmente decrescenti più ci si allontana dall'istante attuale. Di solito si presenta in questa forma:

$$\hat{y}_{t|t} = \alpha y_t + (1-\alpha)\hat{y}_{t-1|t-1}$$

dove $\alpha \in [-1, +1]$ è un coefficiente da stimare, espandendo ricorsivamente la formula sopra abbiamo:

$$\begin{aligned}
\hat{y}_{t|t} &= \alpha y_t + (1-\alpha)[\alpha y_{t-1} + (1-\alpha)\hat{y}_{t-2|t-2}] \\
&= \alpha \sum_{i=0}^{m} (1-\alpha)^i y_{t-i} + (1-\alpha)^{m+1}\hat{y}_{t-m-1|t-m-1} \\
&\to \alpha \sum_{i=0}^{\infty} (1-\alpha)^i y_{t-i}
\end{aligned}$$

Osserviamo che quello sopra è esattamente la media ponderata delle y_i inoltre i pesi sommano ad uno poiché la serie geometrica $\sum_{i=0}^{\infty}(1-\alpha)^i \to \frac{1}{\alpha}$ moltiplicati per il α (fuori dalla serie) fa proprio uno.

Il coefficiente α viene stimato usando il metodo dei minimi quadrati degli errori di previsione un periodo in avanti $\hat{\alpha} = \arg\min_\alpha \sum (y_t - \hat{y}_{t|t-1})^2$.

Ma spesso questo modello da risultati deludenti poiché non tiene conto del fatto che spesso le serie temporali possono rallentare o accelerare, l'evoluzione del exponential smoothing semplice è il modello exponential smoothing di Holt-Winters. In questo caso indichiamo con T_t la stima del trend a tempo t e b_t la stima dell'incremento del trend al tempo t basando sui dati fino al tempo t, le equazioni di HOLT quindi sono:

$$\begin{aligned}
T_t &= \alpha y_t + (1-\alpha)(T_{t-1} + b_{t-1}) \\
b_t &= \beta(T_t - T_{t-1}) + (1-\beta)b_{t-1}
\end{aligned}$$

mentre le previsioni si ottengono con la seguente proiezione:

$$\hat{y}_{t+k|t} = T_t + b_t \cdot k \quad \text{con qualche } k \in \mathbb{N}^+$$

si può aggiungere anche al parte stagionale a questo modello nel modo seguente nel caso il modello sia addittivo:

$$\begin{aligned}
T_t &= \alpha(y_t - S_{t-s}) + (1-\alpha)(T_{t-1} + b_{t-1}) \\
b_t &= \beta(T_t - T_{t-1}) + (1-\beta)b_{t-1} \\
S_t &= \gamma(y_t - T_t) + (1-\gamma)S_{t-s}
\end{aligned}$$

nel caso sia moltiplicativo abbiamo:

$$
\begin{aligned}
T_t &= \alpha \frac{y_t}{S_{t-s}} + (1-\alpha)(T_{t-1} + b_{t-1}) \\
b_t &= \beta(T_t - T_{t-1}) + (1-\beta)b_{t-1} \\
S_t &= \gamma \frac{y_t}{T_t} + (1-\gamma)S_{t-s}
\end{aligned}
$$

con $0 \leq \alpha \leq 1, 0 \leq \beta \leq 1$ e $0 \leq \gamma \leq 1-\alpha$, e la proiezione risulta essere nel caso addittivo e moltiplicativo rispettivamente:

$$
\begin{aligned}
\hat{y}_{t+k|t} &= T_t + b_t \cdot k + S_{t+k-s\cdot\lceil k/s \rceil} \quad \text{con qualche } k \in \mathbb{N} \\
\hat{y}_{t+k|t} &= (T_t + b_t \cdot k) \cdot S_{t+k-s\cdot\lceil k/s \rceil} \quad \text{con qualche } k \in \mathbb{N}
\end{aligned}
$$

Gli exponential smoothing non essendo modello stocastici né pemettono inferenza sui coefficienti stimati né forniscono dei intevalli di confidenza per le previsioni.

Modelli ARIMA

$X_1, X_2, ..., X_n$ variabili casuali, tra loro dipendenti altrimenti non avrebbe senso la serie storica, e se tale dipendenza rimane costante nel tempo posso prevedere la serie storica. La stazionarietà è la omogeinità nella dipendenza tra queste variabili.

Stazionarietà debole o in covarianza

Una serie storica X_t è stazionaria in senso debole (o in covarianza) se $\forall t$

$$
\begin{aligned}
E[X_t] &= \mu < \infty \\
Var(X_t) &= \sigma^2 = \gamma_0 < \infty \\
Cov(X_t, X_{t-k}) &= \gamma_k < \infty
\end{aligned}
$$

cioè ho omogenieità temporale nella serie, la dipendenza è relativa rispetto al t scelto ma indipendete da tale t quindi posso proiettare ciò che ho nel passato nel futuro.

Stazionarietà forte

Una serie storia equispaziata è stazionaria in senso forte $\forall h-upla\ t_1, t_2, ..., t_h$, $\forall k$

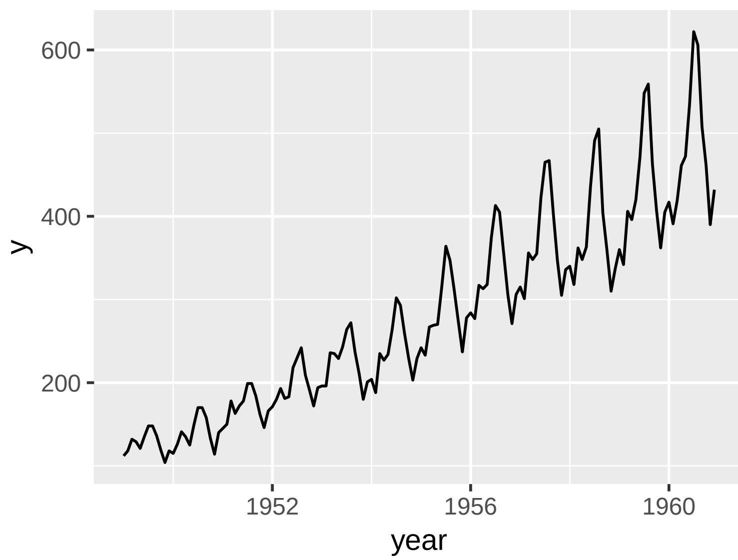
$$
X_{t_1}, X_{t_2}, ..., X_{t_h} =^{(d)} X_{t_1+k}, X_{t_2+k}, ..., X_{t_h+k}
$$

cioè la distribuzione è invariante rispetto alle traslazioni della serie storica.

Oss: Nel caso particolare di processo Gaussiani la stazionarietà forte e debole coincidono, poiché la gaussiana è completamente caratterizata dai primi 2 momenti, la media e la covarianza.

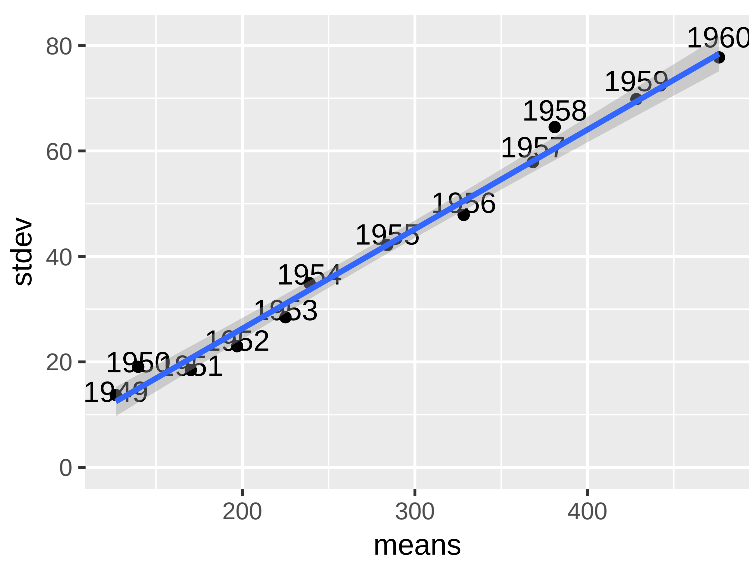
Spesso nelle serie storiche le ipotesi di stazionarietà vengono violate, in tal caso bisogna usare delle trasformazioni reversibili in modo da far rispettare le ipotesi e dopo la previsioni invertire la trasformazione (es. Transformazione Box-Cox).

```r
library(ggplot2)
y <- AirPassengers
ggplot( data = data.frame(y=as.numeric(y),
                          year=as.numeric(time(y))),
        aes(y=y, x=year))+
  geom_line()
```



Osserva come la varianza cresce col passare degli anni, anche la media non è costante nel tempo, quindi viola praticamente tutte le ipotesi di stazionarietà.

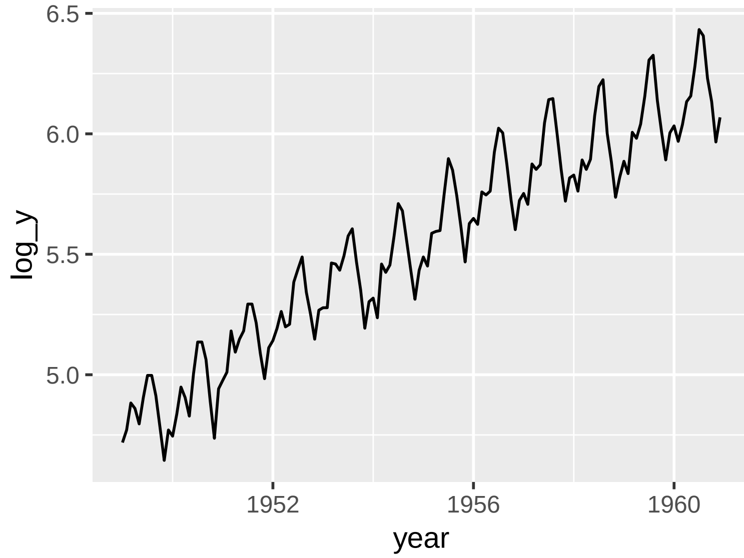
```
means <- tapply(y, floor(time(y)) , mean)
stdev <- tapply(y, floor(time(y)) , sd)
df <- data.frame(means=means, stdev=stdev)

ggplot(aes(x=means, y=stdev), data = df) +
  geom_point() +
  geom_text(label=rownames(df), position = position_stack(vjust = 1.05)) +
  geom_smooth(method=lm)
```



Ragruppando per ogni anni la media e la deviazione standard vedo che cresce linearmente, questo comportamento indica l'utilizzo di un trasformazione logaritmica:

```
ggplot(aes(y=log_y, x=year),
       data = data.frame(log_y=log(as.numeric(y)),
                         year=as.numeric(time(y))))+
```

```r
  geom_line()
```



Questo appiatisce la varianza negli negli anni, adesso dovrò solo aggiustare il trend e la stagionalità, poi la serie sarà stazionaria.

Il Random Walk

Supponiamo $X_t = X_{t-1} + \varepsilon_t$ con ε_t $i.i.d.(0, \sigma^2)$, questo è un Random Walk, il RW non è stazionaria:

$X_0 = x_0 \Rightarrow X_t = X_0 + \sum_{i=1}^{t} \varepsilon_i$ quindi abbiamo che

$$E[X_t] = E[X_0] + \sum_i E[\varepsilon_i] = E[X_0] = x_0$$

$$Var(X_t) = E[(X_t - X_0)^2] = E[(\sum_i \varepsilon_i)^2] = \sum_i Var(\varepsilon_i) = t\sigma^2$$

e quindi la varianza non è costante quindi non è stazionoria, ma è stazionario $X_t - X_{t-1} = \varepsilon_t$. Anzi noto che la varianza cresce col tempo, quindi cresce anche l'incertezza sulla stima che faccio sulla serie.

Un processo X_t non stazionario, ma tale che la sua differenza $X_t - X_{t-1}$ è stazionario si dice integrato di ordine 1: $X_t \sim I(1)$. Definisco nomeinclatura degli operatori: B l'operatore tale che $BX_t = X_{t-1}$ e $\Delta = \mathbb{I} - B$ e ottengo un altro operatore: $\Delta X_t = X_t - X_{t-1}$

$X_t \sim I(d)$ se X_t non è stazionario $\Delta^k X_t$ per $k = 1, ..., d-1$ non è stazionario ma $\Delta^d X_t$ è stazionario. Oss: $\Delta^2 X_t \neq X_t - X_{t-2}$ bensì $\Delta^2 X_t = X_t - 2X_{t-1} + X_{t-2}$.

Predittore Ottimale

Il predittore minimizza il valore atteso della funzione di perdita $l()$, in funzione di $p(X_1, ..., X_m)$ che è una funzione misurabile:

$$\min_p \mathbb{E}(l(Y - p(X_1, ..., X_m)))$$

Integrazione Stagionale

Differenza stagionale Δ_k è così definita:

$$\Delta_k = \mathbb{I} - B^k \rightarrow \Delta_k X_t = X_t - X_{t-k}$$

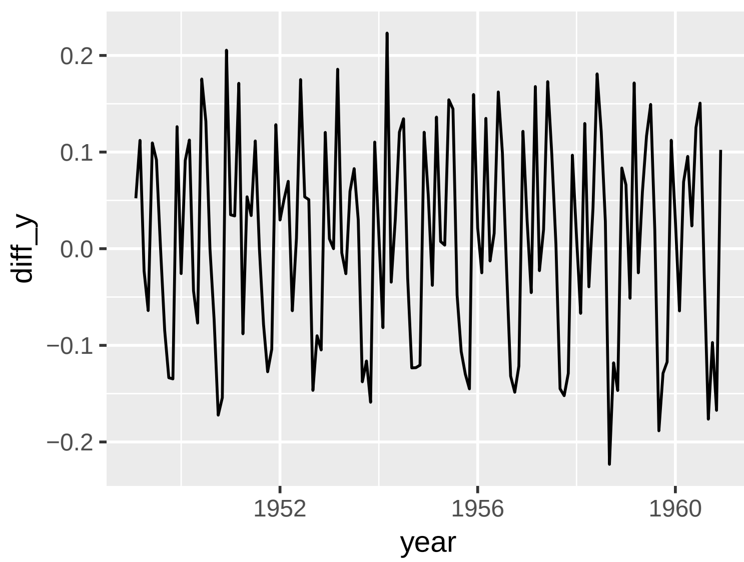
Data la differenza stagionale Δ_k, X_t è stagionale ordinato di ordine 1 se X_t è non stazionario e $X_t - X_{t-k}$ è stazionario. E' utile quando vi è una ripetitività stagionale, cioè un comportamento che si ripete dopo k tempi, eliminandolo elimino questa stagionalità, che non potevo ottenere solo con la differenza Δ^k.

X_t è stagionalmente integrato (di stagione k) di ordine 1 se X_t è non stazionario e $X_t - X_{t-k}$ è stazionario. Generalizzando X_t è stagionalmente integrato di ordine h se X_t non è stazionario $\Delta_{kj} X_t$ è non stazionario per $j = 1, ..., h-1$ è non stazionario e $\Delta_{kh} X_t$ è stazionario.

Oss: $\Delta_k = \mathbb{I} - B^k = (\mathbb{I} - B)(\mathbb{I} + B^1 + ... + B^{k-1})$, quindi la differenza stagionale k contiene anche la differenza prima.

L'Integrazione elimina il trend ma non riesce ad eliminare la stagionalità, per eliminare la stagionalità bisogna usare la differenza stazionale.

```
library(ggplot2)
ly <- log(AirPassengers)
dly <- diff(ly)
ggplot(data=data.frame(diff_y=as.numeric(dly), year=as.numeric(time(dly))),
       aes(x=year, y=diff_y))+
  geom_line()
```



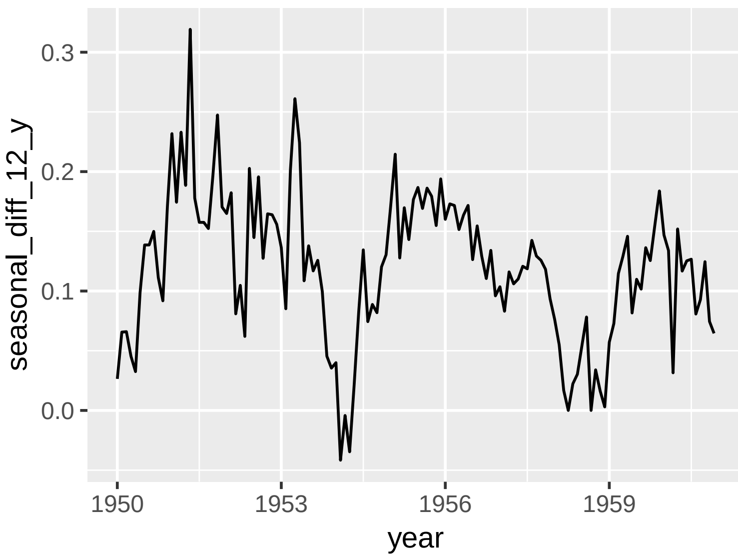
Si vede che il trend non c'è più ma la stagionalità è ancora evidente, inoltre perdo anche l'informazione dell'intercetta in questo modo, quindi vorrei che il modelli trovi e salvi l'intercetta (drift) in qualche modo.

White noise

ε_t è white noise se $\mathbb{E}[\varepsilon_\approx] = 0$, $Var(\varepsilon_t) = \sigma^2$ e $Var(\varepsilon_t, \varepsilon_{t-k}) = 0 \quad \forall k \neq 0$. White noise è un processo stazionario e per rendere il white noise a media non nulla basta aggiungere una costante, quindi ho $X_t = X_{t-1} + \varepsilon_t + c$ detto Random walk con drift, si osserva che $X_t = X_0 + ct + \sum_{i=1}^t \varepsilon_i$, il pezzo $c \cdot t$ è il drift.

Oss: Nel breve periodo di previsione la previsione è il drift, mentre a lungo periodo la maggior parte della varianza è spiegata dal white noise (che aumenta quindi il rumore e quindi l'intervallo di confidenza della previsione).

```
sdly <- diff(ly,12) #seasonal difference
ggplot(data=data.frame(seasonal_diff_12_y=as.numeric(sdly),
                       year=as.numeric(time(sdly))),
       aes(x=year, y=seasonal_diff_12_y))+
  geom_line()
```



E adesso sembra non avere più la stagionalità, chiaramente un po' di memoria stagionale rimane, lo si vede usando ACF e PACF. Ma i movimenti netti sulla stagionalità non ci sono più. Osserva siccome siamo sul logaritmo, il dato adesso è in percentuali.

Spesso non si è sicuro se la serie dopo la trasformazione è ancora stazionario o meno, vi sono dei test ma uno potrebbe applicare 2 modelli uno considerandolo come stazionario e l'altro come non stazionario e poi si prende il modello migliore.

Modello Autoregressivo

Un processo autoregressivo $AR(p)$ è $X_t = C + \phi_1 X_{t-1} + ... + \phi_p X_{t-p} + \varepsilon_t$.

OSS: Random Walk è $AR(1)$ quindi un $AR(p)$ non è necessariamente stazionario, poiché il caso particolare Random Walk non lo è.

Notazione compatta per la funzione con operatore ritardo è:

$$\phi_p(B) = \mathbb{I} - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p$$

quindi ho che $\phi_p(B)X_t = c + \varepsilon_t$, infatti

$$\phi_p(B)X_t = (\mathbb{I} - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p)X_t = X_t - \phi_1 X_{t-1} - ... - \phi_p X_{t-p} = c + \varepsilon_t$$

Teorema: Se le soluzioni di

$$\phi_p(Z) = 0 \to \mathbb{I} - \phi_1 Z - ... - \phi_p Z^p = 0$$

sono tali che le soluzioni per $i = 1, ..., p$, $|Z^{(i)}| > 1 \Rightarrow$ il processo è stazionario.

Es. prendiamo $AR(1) : X_t = \phi X_{t-1} + \varepsilon_t + c$ da cui

$$(1 - \phi B)X_t = \varepsilon_t + c$$

Quindi le soluzioni dell'equazione $1 - \phi Z = 0$ sono $Z = \frac{1}{\phi}$ quindi ha modulo > 1 sse $\phi < 1$ quindi il random walk che è con $\phi = 1$ non è stazionario.

Se una delle soluzioni di $\phi_p(Z) = 0$ è pari a 1 e le altre sono in modulo maggiore di 1 allore il processo $AR(p)$ è integrato di ordine 1.

Dimostrazione nel caso di 1 radice, (per k radici è analogo): Date le soluzioni il polinomio sopra si può fattorizzare come:

$$\phi_p(B) = (1 - \frac{1}{Z^{(1)}}B)(1 - \frac{1}{Z^{(2)}}B) \cdot ... \cdot (1 - \frac{1}{Z^{(p)}}B)$$

Se $Z^{(1)} = 1$ allora $\phi_p(B) = (1 - B)\varphi_{p-1}(B)$ quindi il processo AR(p) $\phi_p(B)X_t = c + \varepsilon_t$ posso riscriverlo come

$$\varphi_{p-1}(B)(1 - B)X_t = \varphi_{p-1}(B)\Delta X_t = c + \varepsilon_t$$

quindi ho dimostrato che il processo era integrato di ordine 1 (ΔX_t).

Si generalizza, ottendendo se ho più radici unitarie l'ordine del processo aumenta con il numero delle radici, quindi se ho k radici il processo sarà integrato di ordine k.

Oss: Se abbiamo una radice unitaria allora dal $1 - \phi_1 Z - ... - \phi_p Z^p = 1 - \sum_i \phi_i Z^i = 0$ deduciamo che la somma di ϕ_i è 1.

Oss: nell'equazione di AR(1) $X_t = c + \phi X_{t-1} + \varepsilon_t$, c non è la media, infatti sia μ la media allora

$$\mu \mathbb{E}[X_t] = \mathbb{E}[c] + \mathbb{E}[\phi X_{t-1}] = c + \phi\mu \to \mu = \frac{c}{1 - \phi}$$

Quindi se $\phi = 1$ la media esplode. Si generalizza per AR(p) ottentendo:

$$\mu = \frac{c}{1 - \sum_{i=1}^{p} \phi_i}$$

In un processo stazionario abbiamo: $\gamma_k = Cov(X_t, X_{t-k})$ abbiamo che $\rho_k = \frac{\gamma_k}{\gamma_0}$ è la correlazione. Il lag di autocorrelazione parziale è

$$\alpha_k = Cor(X_t - \mathbb{P}(X_t | X_{t-1}, ..., X_{t-k+1}), X_{t-k} - \mathbb{P}(X_{t-k} | X_{t-1}, ..., X_{t-k+1}))$$

da cui deriva la ACF e PACF:

In caso di un processo AR(p) abbiamo p ritardi di PACF non nulli e dopo p ritardi vanno a zero, mentre ACF tende a tornare a zero geometricamente (questo perché si scorda dei shock dopo infiniti periodi).

Supponiamo una periodicità stagionale S, SAR(P) è molto simile a AR(p) ma lavora su multipli di s, quindi ho

$$X_t = c + \Phi_1 X_{t-S} + ... + \Phi_P X_{t-P \cdot S} + \varepsilon_t$$

Usando gli operatori:

$$(1 - \Phi_1 B^S - ... - \Phi_P B^{P \cdot S})X_t = c + \varepsilon_t$$

chiamiamo il polinomio per SAR(P) $\Phi_P(B)$, un modello quindi AR(p)(P)$_S$ è:

$$\phi_p(B)\Phi_P(B)X_t = c + \varepsilon_t$$

è come se si applicassero 2 filtri, uno che si occupa del white noise di memoria recente e l'altro con memoria stagionale.

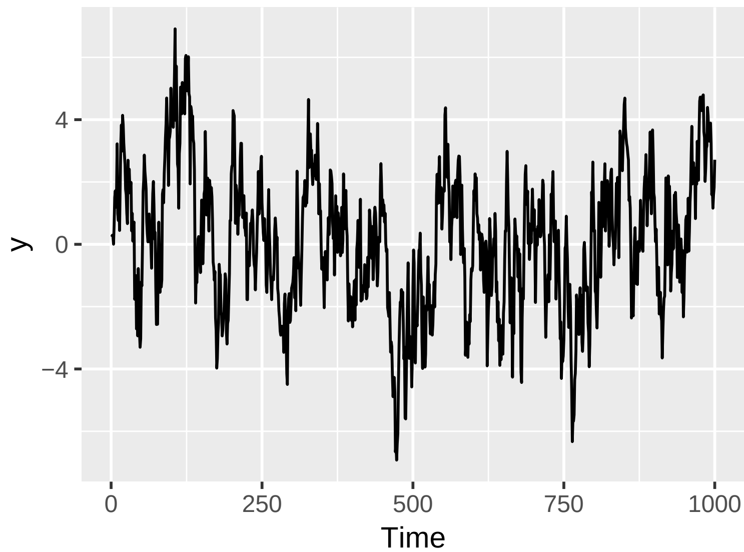
ad es. AR(1)(1)$_4$ è $(1 - \phi B)(1 - \Phi B^4)X_t = (1 - \phi B - \Phi B^4 + \phi\Phi B^5)X_t = c + \varepsilon_t$

per riconoscere SAR(1)$_P$ abbiamo il primo e l'unico ritardo di PACF al ritardo P, mentre nel ACF abbiamo il ritardo sui multipli di P che decade geometricamente.

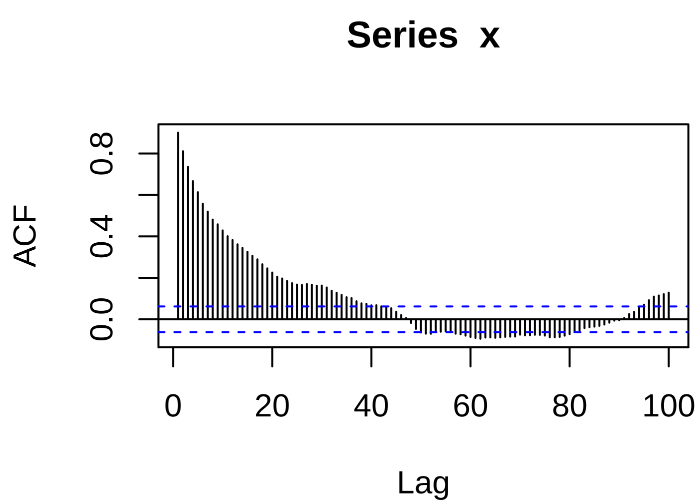
Simulazione modelli ARIMA in R:

```r
#~~ AR(1)
library(ggplot2)
n <- 1000
phi <- 0.9 #OSS: è < 1, se fosse >1 esplode la serie poiché viene respinto da 0
#se phi=1 non è più attratto da 0 ma non esplode ancora (è un random walk)
```

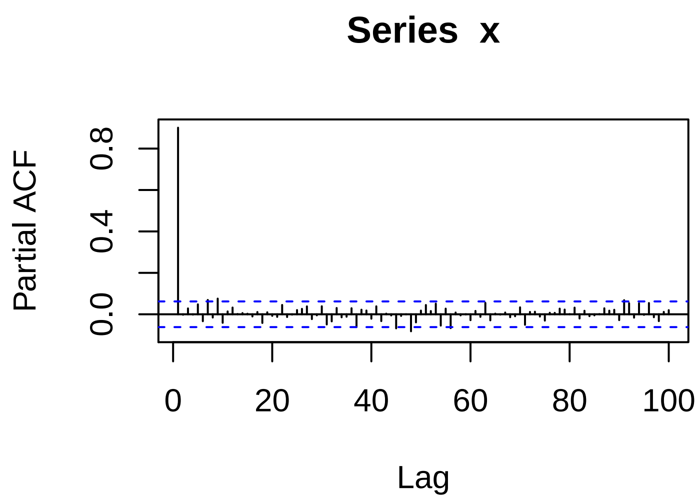
```r
eps <- rnorm(n) #White Noise
x <- stats::filter(eps, c(phi), "recursive", init=0)

ggplot(aes(x=Time, y=y),
       data = data.frame(y=as.numeric(x),
                         Time=1:length(x)))+
  geom_line()
```




```r
#~~ ACF e PACF plot
library(forecast)
Acf(x, lag.max = 100)
```



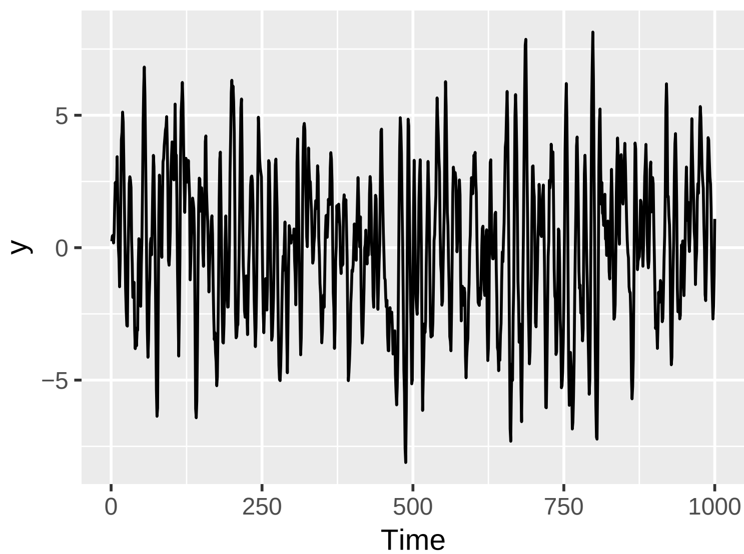

```r
Pacf(x, lag.max = 100)
```



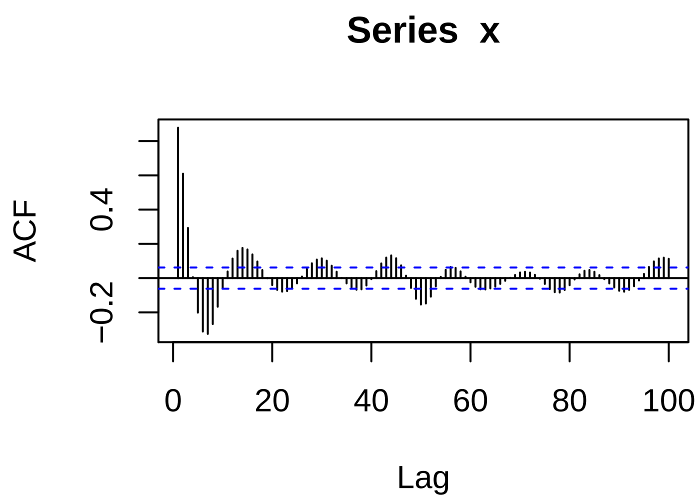

```r
#~~ AR(2)
phi <- c(1.5, -0.7)
#Le radici sono in modulo > 1, poiché il processo è stazionario
Mod(polyroot(c(1, -phi)))
```

```
## [1] 1.195229 1.195229
```

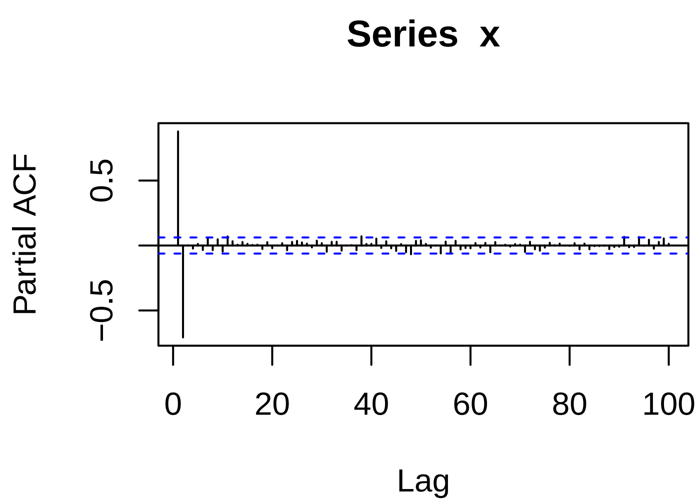
```r
x <- stats::filter(eps, c(phi), "recursive")
ggplot(aes(x=Time, y=y),
       data = data.frame(y=as.numeric(x),
                         Time=1:length(x)))+
  geom_line()
```



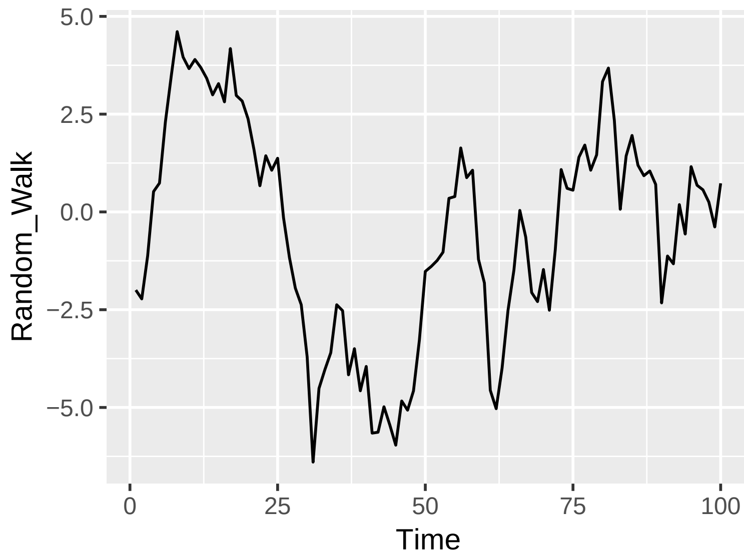

```r
Acf(x, lag.max = 100)
```




```
Pacf(x, lag.max = 100)
```




```
#~~ Stima AR(1) su un processo RandomWalk (RW)
n <- 100
eps <- rnorm(n)
rw <- cumsum(eps)
ggplot(aes(x=Time, y=Random_Walk),
       data = data.frame(Random_Walk=rw,
                         Time=as.numeric(1:length(rw))))+
  geom_line()
```




```
mod1 <- Arima(rw, order = c(1,0,0)) #oss: il coef ar1 non è 1, quindi la stima è "distorta"
summary(mod1)
```

```
## Series: rw
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##          ar1     mean
##       0.9163  -0.5312
## s.e.  0.0368   1.1547
##
## sigma^2 estimated as 1.163:  log likelihood=-149.33
## AIC=304.66   AICc=304.91   BIC=312.47
##
## Training set error measures:
##                      ME     RMSE       MAE      MPE     MAPE      MASE
## Training set 0.02222957 1.067373 0.8331625 43.91917 131.5218 0.9532721
##                    ACF1
## Training set 0.08868984
```

Vi sono dei test (di Dickey-Fuller) per evitare questi problemi e vedere se vi è una radice unitaria nella serie:

Nel processo AR(p), ho che H_0 : almeno una radice caratteristica è $= 1$, e H_1 : tutte le radici sono in modulo > 1 (stazionarietà).

in realtà Dickey-Fuller ha $H_0 : RW$ e $H_1 : AR(1)$ stazionaria mentre il test ADF che è basato su Dickey-Fuller, che testa se la somma delle radici è $= 1$, in caso affermativo sono sicuro di avere una radice unitaria, vedi lezione 2 per l'equazione da risolvere per le radici del polinomio fa cui deriva questa idea della somma delle radici $= 1$

```
urca::summary(urca::ur.df(rw, type="drift", lags=10, "AIC"))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
```

```
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.88842 -0.61767  0.04788  0.70887  2.10648
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.10671    0.11956  -0.893   0.3746
## z.lag.1     -0.09953    0.04288  -2.321   0.0226 *
## z.diff.lag   0.04744    0.10668   0.445   0.6577
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.089 on 86 degrees of freedom
## Multiple R-squared:  0.05899,    Adjusted R-squared:  0.03711
## F-statistic: 2.696 on 2 and 86 DF,  p-value: 0.0732
##
##
## Value of test-statistic is: -2.3211 2.7404
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.51 -2.89 -2.58
## phi1  6.70  4.71  3.86
```

Il valore del test è a doppia coda quindi accetto l'ipotesi nulla se il valore della statistica (t-value di z.lag.1) sta a destra di 5pct tau (-2.89) e rifiuto nel caso sta a sinistra di tale valore, ad esempio sulla x di prima che sappiamo non avere radice unitaria H_0 viene rifiutata:

```r
#non stampo tutto il summary infondo interessa solo la statistica,
#per il confronto guardare i treshold sopra.
print(urca::ur.df(x, type="drift", lags=10, "AIC"))
```

```
##
## ###############################################################
## # Augmented Dickey-Fuller Test Unit Root / Cointegration Test #
## ###############################################################
##
## The value of the test statistic is: -18.8414 177.5021
```

Moving Average

Un processo Moving Average a q ritardi, MA(q), è data da una combinazione lineare mobile dei white noise:

$$X_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + ... + \theta_q \varepsilon_{t-q}$$

Oss: a questo punto X_t e X_{t-k} con k>q sono incorrelate, esempio com MA(1)

$$X_t = \varepsilon_t + \theta_1 \varepsilon_{t-1}$$

$$X_{t-1} = \varepsilon_{t-1} + \theta_1 \varepsilon_{t-2}$$

e quindi

$$Cov(X_t, X_{t-1}) = \mathbb{E}[X_t \cdot X_{t-1}] = \mathbb{E}[(\varepsilon_t + \theta_1 \varepsilon_{t-1}) \cdot (\varepsilon_{t-1} + \theta_1 \varepsilon_{t-2})] = \theta_1 \sigma_\varepsilon^2$$

Quindi il processo MA(q) si ricorda al più ciò che è successo fino a q ritardi prima, mentre un processo AR si porta dietro una memoria molto lunga. Quindi nel grafico ACF fino a q ritardi è non nullo il valore e vale zero oltre q ritardi, mentre nel caso di PACF rientra a zero a velocità geometrica (sono possibili delle oscillazioni). Anche qua si possono definire dei polinomi:

$$\theta_q(B) = 1 + \theta_1 B + ... + \theta_q B^q$$

e il polinomio caratteristico per quella stagionale è:

$$\Theta_Q(B)_S = 1 + \Theta_1 B^S + ... + \Theta_q B^{S \cdot Q}$$

esempio di modelli misti: MA(1)(1)$_4$: $(1 - \theta_1 B)(1 - \Theta_1 B^4) X_t$

Oss: in questo caso c è la media effetiva di X_t.

Modello ARIMA completo

si può mettere tutto insieme in un modello ARIMA(p,d,q)(P,D,Q)$_S$:

$$\phi_p(B)\Phi_P(B)\Delta^d \Delta_S^D X_t = c + \theta_q(B)\Theta_Q(B)\varepsilon_t$$

Il Metodo di Box&Jenkins

- Faccio grafico della serie, media-sd e mi chiedo
  - è stazionaria in varianza?
    - Se sì vado avanti
    - Se no prendo la trasformazione più opportuna (con Box-Cox $\frac{X^\lambda - 1}{\lambda}$), Oss:

$$\lim_{\lambda \to 0} \frac{X^\lambda - 1}{\lambda} = \log(x)$$

  - è stazionaria in media per stagionalità?
    - Se no prendo la differenza stagionale
    - Se sì vado avanti
  - è stazionaria in media?
    - Se no prendo la differenza semplice
    - Se sì vado avanti
- Faccio ACF e PACF e tento il primo modello ARMA, stimo il modello e calcolo i residui $r_t = y_t - \hat{y}_{t|t-1}$.
- i residui sono WN (White Noise)?
  - Se sì ho finito
  - Se no aggiusto il modello ARMA iniziale affinché i residui siano WN.

La funzione di verosimiglianza: $L(\theta) = f_\theta(X_1, ..., X_n) = f(X_1)f(X_2|X_1)...f(X_n|X_{n-1}, ..., X_1)$ che va massimizzata in θ.

Oss: $X_t|X_{t-1}...X_1 \sim N(\hat{X}_{t|t-1}, \sigma^2)$ e quindi $r_t = X_t - \hat{X}_{t|t-1} \sim N(0, \sigma^2)$

Un modello ARMA si può esprimere come un modello AR puramente infinto o MA infinito, un modello MA è invertibili se le radici del polinomio caratterstico di MA abbiano modulo > 1 (stesse condizioni della stazionarietà dal modello AR).

Es. Prendiamo un modello MA(1): $y_t = \varepsilon_t + \theta\varepsilon_{t-1}$, e assumiamo il modello MA invertibile quindi cerco $\Pi(B) \quad t.c. \quad \Pi(B)(\mathbb{I} - \theta B) = \mathbb{I}$, sotto ipotesi di invertibilità abbiamo che l'operatore inverso è (dove si vede perché chiediamo $|\theta| < 1$):

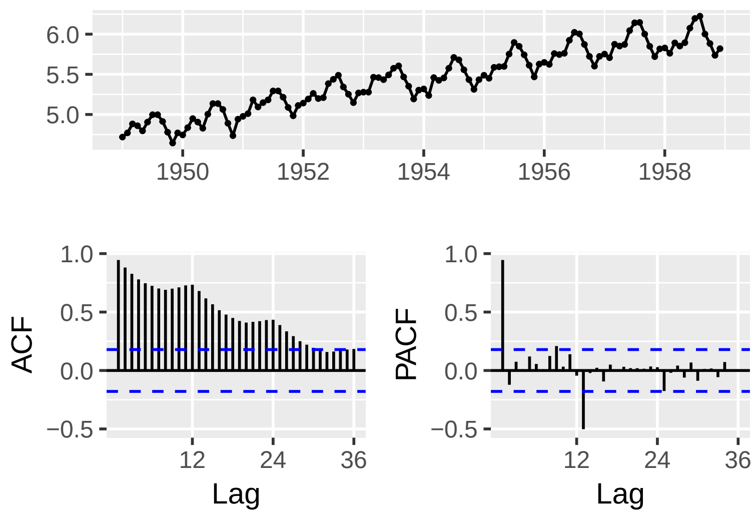
$$\Pi(B) = 1 - \sum_{i=1}^{\infty} (-1)^n \theta^n B^n$$

applicando il modello inverso otteniamo:

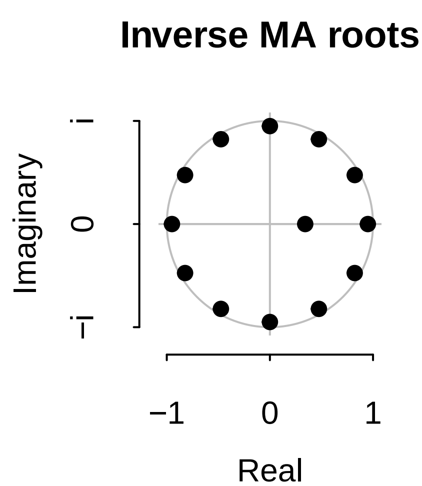
$$(1-\theta B)^{-1}y_t = \varepsilon_t(1-\theta B+\theta^2B^2-...)y_t = \varepsilon_t y_t = \varepsilon_t + \theta y_{t-1} - \theta^2 y_{t-2} + ...$$

Come si può vedere è un AR(∞). Analogamente si può scrivere MA(∞) al posto di un processo AR finito.

```
library(forecast)
data("AirPassengers")
y <- window(AirPassengers, end=c(1958,12))
ggtsdisplay(log(y))
```




```
mod1 <- Arima(y, c(0, 1, 1), c(0, 1, 1), lambda = 0) #con lambda = 0 diventa una serie logaritmica
plot(mod1)
```




```
#print(mod1)
```

```
pre1 <- matrix(NA_real_, 24, 12)
colnames(pre1) <- paste0("h", 1:12)
err1 <- pre1
```

```r
yna <- c(AirPassengers, rep(NA, 12))

for (i in 1:24){
  pre1[i, ] <- forecast(Arima(yna[1:(119+i)], model = mod1),
                        h=12, biasadj = TRUE)$mean
  err1[i,] <- yna[(120+i):(131+i)] - pre1[i,]
}

meanloss <- cbind(RMSE = sqrt(colMeans(err1^2, na.rm = T)),
                  MAE  = colMeans(err1, na.rm = T),
                  MAPE = colMeans(abs(err1)/(err1+pre1), na.rm = T)*100)
t(meanloss)
```

```
##             h1        h2        h3        h4        h5        h6        h7
## RMSE 15.264185 15.733193 16.131737 18.107846 17.001679 17.797812 21.447382
## MAE   1.090497  1.780475  3.239936  4.023566  4.709915  5.016538  6.042383
## MAPE  2.546494  2.708195  2.652833  2.972683  2.944106  3.153833  3.900381
##             h8        h9       h10       h11       h12
## RMSE 20.146664 18.417459 20.702773 20.056498 21.260001
## MAE   6.336393  6.431376  5.369597  6.244222  7.400860
## MAPE  3.670042  3.568156  4.168495  4.109114  4.245629
```

ARIMAX

ARMAX è il modello ARMA con dei regressori (X) ed è della forma:

$$y_t = X_t^T\beta + \eta \quad \text{dove} \quad \eta \sim ARMA(p, q)$$

η può essere anche stagionale.

La parte I è più complessa, alcuni software (R) applicano la differenza sia alla y anche alla X (ed è giusto così). Le trasformazioni invece vengono applicate solo alla y e non a X, per le trasformazioni alla X bisogna procedere manualmente. ARIMAX di stagione S in R è implemenetato così:

$$\Delta^d \Delta_S^D y_t = \Delta^d \Delta_S^D X_t^T\beta + \eta \quad \text{dove} \quad \eta \sim ARMA(p, q)(P, Q)_S$$

espandendo la formula totale abbiamo:

$$\phi(B)\Phi(B)\Delta^d \Delta_S^D(y_t - X_t^T\beta) = \theta(B)\Theta(B)\varepsilon_t$$

Modelli UCM

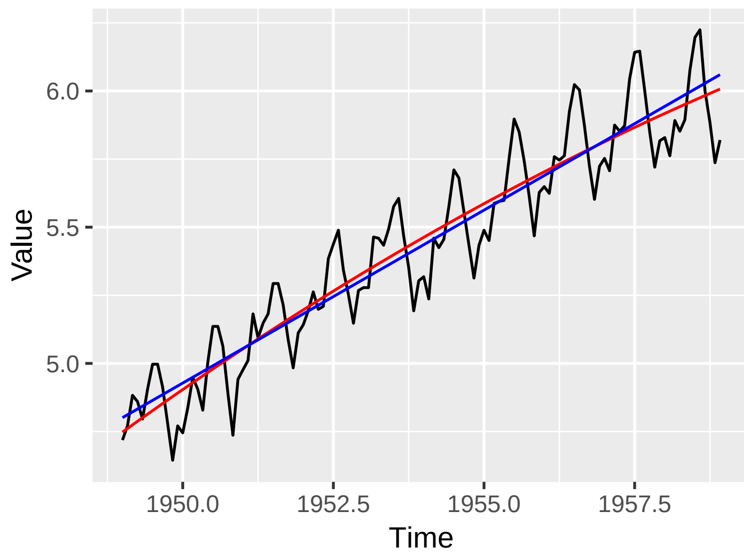
Ragioniamo in termini di trend, stagionalità, componenti cicliche e scordiamo l'esistenza dei modelli ARIMA. Per risolvere un problema di serie storica si può pensare di risolverla come una regressione, i miei regressori possono essere il tempo:

$$y_t = \beta_0 + \beta_1 t(+\beta_2 t^2)$$

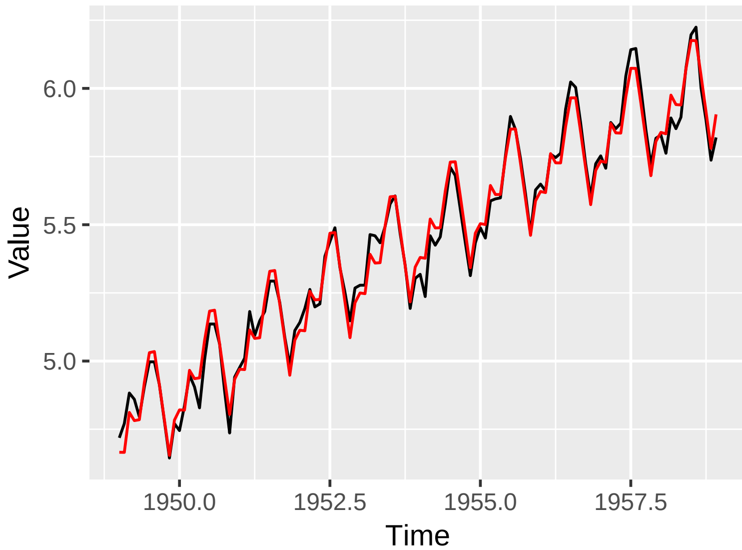
in questo modo cogliamo sicuramente il trend crescente mentre per la stagionalità si possono usare come regressori i dummy dei mesi (oppure un'altra base con le frequenze di seno e coseno).

```r
library(forecast)
library(ggplot2)
data("AirPassengers")
y <- window(AirPassengers, end=c(1958,12))
```

```r
mod1 <- Arima(y, c(0, 1, 1), c(0, 1, 1), lambda = 0) #con lambda = 0 diventa una serie logaritmica
trend <- 1:144
reg1 <- lm(log(y)~trend[1:120])

reg <- lm(log(y)~trend[1:120] + I(trend[1:120]^2))

ggplot()+
  geom_line(aes(y=as.numeric(log(y)), x=time(y))) +
  geom_line(aes(y=reg$fitted.values, x=time(y)), colour='red') +
  geom_line(aes(y=reg1$fitted.values, x=time(y)), colour='blue') +
  xlab("Time") + ylab("Value")
```




```r
mesi <- months(as.Date(time(timeSeries::as.timeSeries(y))))

reg3 <- lm(log(y)~trend[1:120] + I(trend[1:120]^2) + mesi)

ggplot()+
  geom_line(aes(y=as.numeric(log(y)), x=time(y))) +
  geom_line(aes(y=reg3$fitted.values, x=time(y)), colour='red') +
  xlab("Time") + ylab("Value")
```

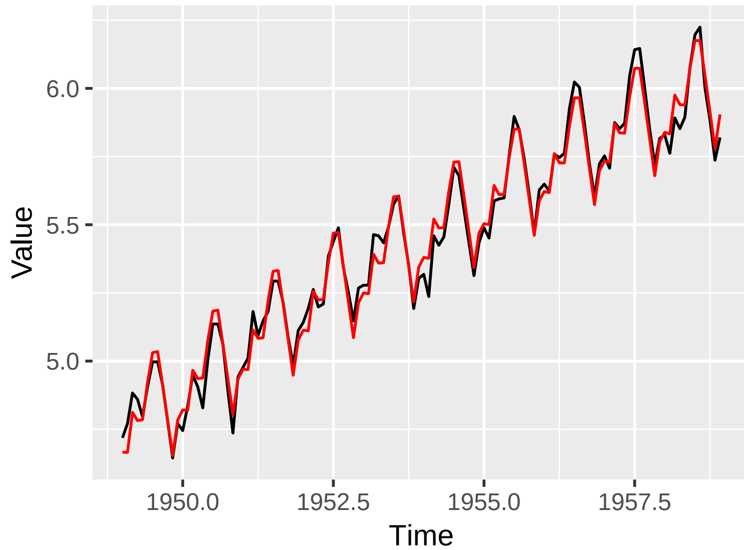


Sia γ_t la stagionalità di periodo s quindi $\gamma_t = \gamma_{t-s}$, se sommo $\sum_{i=0}^{s-1} \gamma_{t-i}$ questa somma va a 0, inoltre vista la stagionalità ho che:

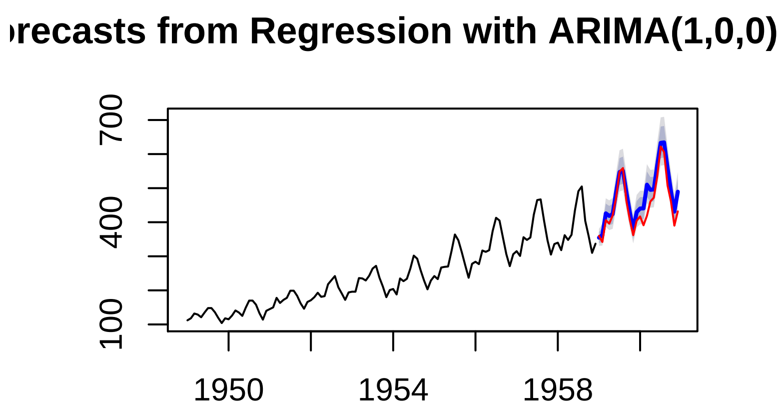
$$\gamma_t = \sum_{j=1}^{\lfloor \frac{s}{2} \rfloor} \alpha_j \cos(\frac{2\pi}{s} jt) + \beta_j \sin(\frac{2\pi}{s} jt)$$

Oss: se s pari per $j = \frac{s}{2}$ $\sin(\frac{2\pi}{s} \frac{s}{2} t) = 0$, quindi ho un regressore in meno.

```
#~~ costruisco i regressori sinusoidi e cosinusiodi
fr <- 2*pi*outer(trend, 1:6) / 12
co <- cos(fr)
si <- sin(fr[,1:5]) #la sesta non serve dopo l'osservazione sopra

colnames(co) <- paste0("cos",1:6)
colnames(si) <- paste0("sin",1:5)

reg_cosi <- lm(log(y)~trend[1:120] + I(trend[1:120]^2) + co[1:120,] + si[1:120,])

ggplot()+
  geom_line(aes(y=as.numeric(log(y)), x=time(y))) +
  geom_line(aes(y=reg_cosi$fitted.values, x=time(y)), colour='red') +
  xlab("Time") + ylab("Value")
```




```
arimax <- Arima(y, c(1, 0, 0), lambda = 0,
                xreg = cbind(trend, co, si)[1:120,])

#prev <- predict(arimax, newxreg = cbind(trend, co, si)[121:144,])

plot(forecast(arimax, xreg = cbind(trend, co, si)[121:144,]))
lines(window(AirPassengers, start=c(1959,1) ,end=c(1960, 12)),col='red')
```



Il trend per $t = 0, 1, 2, ...$ lo possiamo definire dato $\gamma_0 = \alpha$ come trend iniziale:

$$\gamma_t = \alpha + \beta t$$

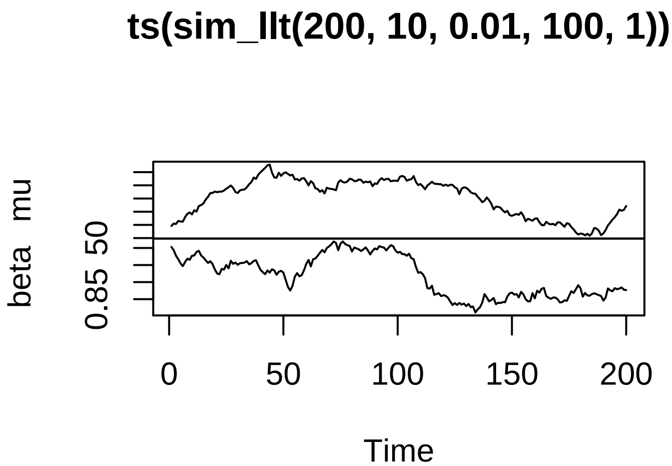
ma vorrei che sia l'intercetta α che β dipenda dal tempo poiché cambia col tempo:

$$\gamma_{t+1} = \gamma_t + \beta_t + \eta_t, \quad \eta_t \sim WN(0, \sigma_\eta^2)$$

dove $\beta_{t+1} = \beta_t + \zeta_t, \quad \zeta_t \sim WN(0, \sigma_\zeta^2)$

OSS: γ_t è integrato 2 volte. μ è detto level e β è detto slope

```r
#~~ implementazione in R di trend
sim_llt <- function(n, sd_level, sd_slope, level0, slope0){
  beta <- cumsum(rnorm(n, sd = sd_slope)) + slope0
  mu <- cumsum(beta + rnorm(n, sd = sd_level)) + level0
  cbind(mu = mu, beta = beta)
}

plot(ts(sim_llt(200, 10, 0.01, 100, 1)))
```



Il ciclo stocastico

Indica che c'è un ciclo ma non è deterministico quindi non basta usare seno e coseno come per il ciclo di prima, userò seno e coseno stocastico.

$$\begin{bmatrix}\psi_{t+1}\\ \psi^*_{t+1}\end{bmatrix} = \rho\begin{bmatrix}\cos(\lambda) & \sin(\lambda)\\ -\sin(\lambda) & \cos(\lambda)\end{bmatrix}\begin{bmatrix}\psi_t\\ \psi^*_t\end{bmatrix} + \begin{bmatrix}k_t\\ k^*_t\end{bmatrix}, \quad \begin{bmatrix}k_t\\ k^*_t\end{bmatrix} \sim WN(0,\sigma^2_k\mathbb{I}) \text{ e } 0 \le \rho \le 1$$

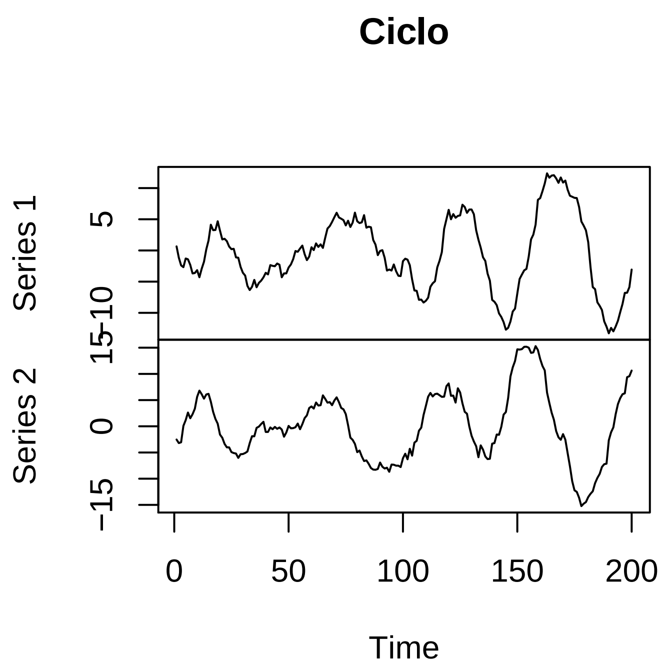
Osserviamo che la matrice è la matrice di Rotazione $R(\lambda)$ che ruota il vettore di angolo λ. k_t e k^*_t sono WN tra loro incorrelati ma a varianza costante. Tipicamente si prende solo ψ_t l'altro ψ^*_t è ortogonale al primo e serve solo per la sua costruzione. Il primo pezzo dell'equazione non rappresenta altro che un ciclo e la parte di WN aggiunge il rumore per renderlo stocastico. Il ρ serve per far assorbile il ciclo a lungo andare (geometricamente tende a far portare a 0), se $\rho = 0$ allora rimane solo il WN mentre se $\rho = 1$, il ciclo diventa non stazionario. Se è stazionario abbiamo che la varianza $\mathbb{E}(\psi\psi^t) = \dfrac{\sigma^2}{1-\rho^2}\mathbb{I}$.

Osservazione: La formula della varianza è identica al caso di AR(1) con $\rho = \phi$.

Se il ciclo stocastico ha stagionalità S e il ciclo ha una durata di $k \cdot S$ allora la frequenza λ è data dal rapporto: $\lambda = \dfrac{\pi}{kS}$.

```r
#~~ In R
library(ggplot2)
sim_cyc <- function(n, rho, lambda, sd_kappa, psi0){
  psi <- matrix(0, n, 2)
  R <- matrix(c(cos(lambda),-sin(lambda), sin(lambda), cos(lambda)), 2, 2)
  Rt <- t(R) #traspongo perché in R voglio un vettore colonna e non riga come in teoria
```

```r
  psi[1, ] <- rho*R %*% psi0 + rnorm(2, sd=sd_kappa)
  for (t in 2:n){
    psi[t, ] <- rho*R %*% psi[t-1, ] + rnorm(2, sd=sd_kappa)
  }
  psi
}
#il rho dà la persistenza del ciclo, più è vicino a zero, per più tempo lo vedremo esistere
ciclo = sim_cyc(n = 200, rho = 0.99, lambda = 2*pi/48, sd_kappa = 1, rnorm(2))
plot(ts(ciclo), main = 'Ciclo')
```



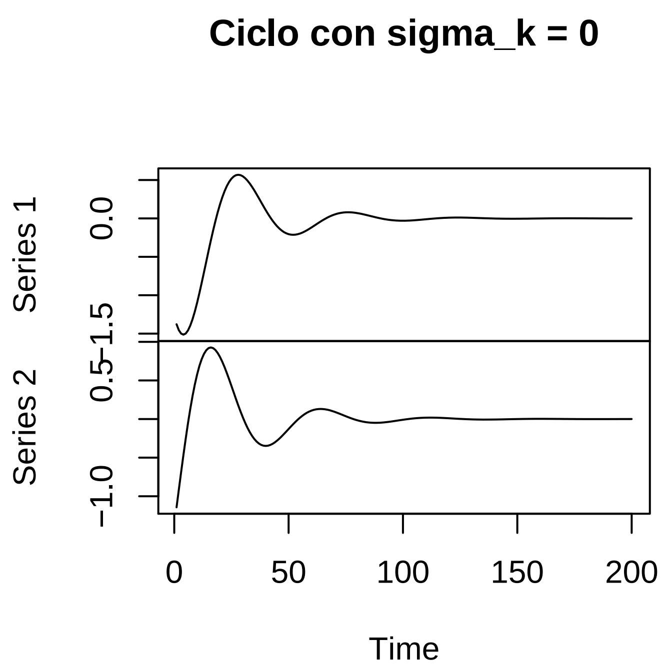
Nei 2 cicli generati sopra se uno prende la loro correlazione, dovrebbe essere circa 0 (almeno prendendo serie infinita quindi dipende da n):

```r
cor(ciclo[,1], ciclo[,2])
```

```
## [1] -0.004667041
```

Se σ_k la prendiamo $= 0$ cioè elimino la stocasticità in teoria la serie diventa un seno che piano piano scompare, in base al coefficiente ρ:

```r
ciclo = sim_cyc(n = 200, rho = 0.96, lambda = 2*pi/48, sd_kappa = 0, rnorm(2))
plot(ts(ciclo), main = 'Ciclo con sigma_k = 0')
```



In particolare se $\sigma_k = 0$ e $\rho = 1$, diventa un ciclo perfetto, mentre se $\rho = 1$ e $\sigma_k \neq 0$ allora abbiamo una non stazionarietà (non c'è un ciclo).

Sappiamo che possiamo scrivere la stagionalità come:

$$\gamma_t = \sum_{j=1}^{\lfloor \frac{s}{2} \rfloor} \alpha_j \cos(\frac{2\pi}{s} jt) + \beta_j \sin(\frac{2\pi}{s} jt)$$

il ciclo stocastico stagionale lo possiamo ottenere come:

$$\gamma_t = \sum_{j=1}^{\lfloor \frac{s}{2} \rfloor} \psi_t^{(j)}$$

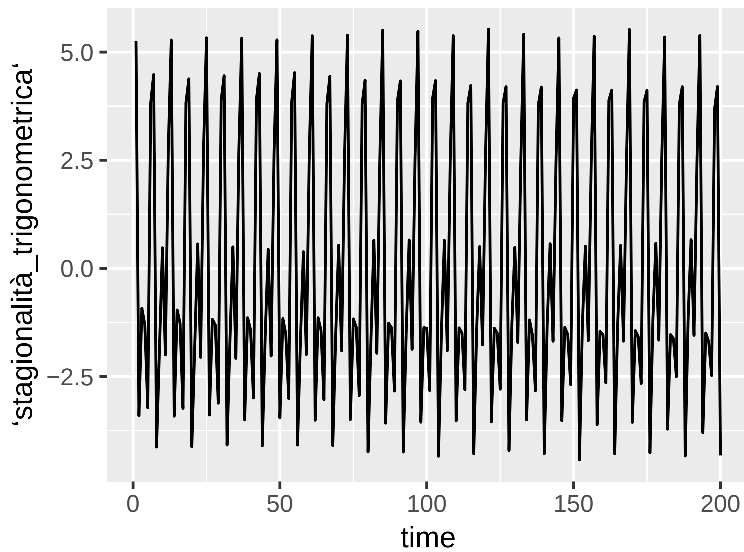
dove

$$\begin{bmatrix} \psi_{t+1}^{(j)} \\ \psi_{t+1}^{(j)*} \end{bmatrix} = \rho \begin{bmatrix} \cos(\frac{2\pi}{s} j) & \sin(\frac{2\pi}{s} j) \\ -\sin(\frac{2\pi}{s} j) & \cos(\frac{2\pi}{s} j) \end{bmatrix} \begin{bmatrix} \psi_t^{(j)} \\ \psi_t^{(j)*} \end{bmatrix} + \begin{bmatrix} \omega_t^{(j)} \\ \omega_t^{(j)*} \end{bmatrix}, \quad \begin{bmatrix} \omega_t^{(j)} \\ \omega_t^{(j)*} \end{bmatrix} \sim WN(0, \sigma_\omega^2 \cdot \mathbb{I})$$

Anche qua il numero delle componenti quando s è pari sono s-1.

```
sim_trig_seas <- function(n, s, sd_omega, PSI0){
  gamma <- numeric(n)
  frb <- 2*pi/s #frequenza base
  for (j in 1:floor(s/2)) {
    gamma <- gamma + sim_cyc(n, 1, frb*j, sd_omega, PSI0[j,])[,1]
  }
  gamma
}

stagionalità_trigonometrica <- sim_trig_seas(n = 200, s = 12,
```

```
                                       sd_omega = 0.01, matrix(rnorm(12),6,2))
time = 1:length(stagionalità_trigonometrica)
ggplot() +
  geom_line(aes(y=stagionalità_trigonometrica, x=time))
```



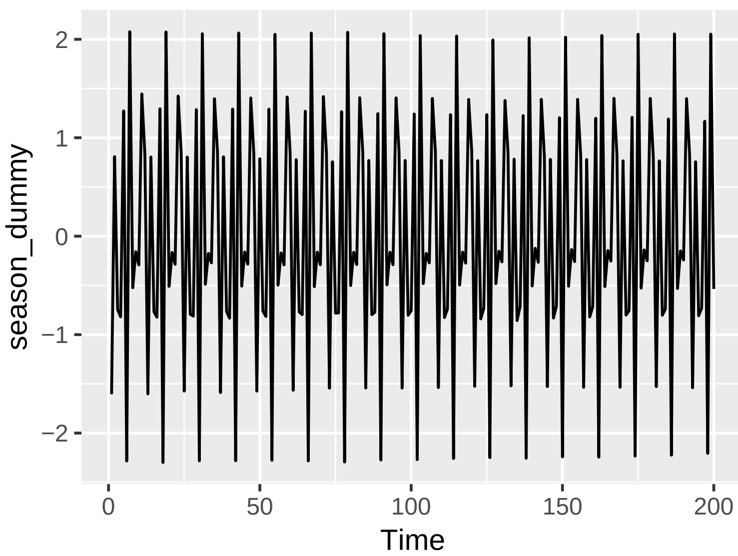
Quando si costruiranno i modello State Space, sapremo come stimare i parametri delle componenti UCM.

Stagionalità a dummy stocastiche

Ho la stagionalità fatta in modo che $\gamma_t = \gamma_{t-s}$ tale che $\sum_{i=0}^{s} \gamma_{t-i} = 0$, cioè se la sommo deve darmi zero → riscrivendo l'equazione sopra ho che $\gamma_t = -\gamma_{t-1} - ... - \gamma_{t-s+1}$ che è già un equazione alle differenze, per renderla stocastica aggiungo come sempre un WN: $\gamma_t = -\gamma_{t-1} - ... - \gamma_{t-s+1} + \omega_t$ dove $\omega \sim WN(0, \sigma^2_\omega)$. Matricialmente (VAR) la posso riscrivere come:

$$\begin{bmatrix} \gamma_t \\ \gamma_t^{(1)} \\ \gamma_t^{(2)} \\ \vdots \\ \gamma_t^{(s-2)} \end{bmatrix}_t = \begin{bmatrix} -1 & -1 & \dots & -1 & -1 \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix} \begin{bmatrix} \gamma_{t-1} \\ \gamma_{t-1}^{(1)} \\ \gamma_{t-1}^{(2)} \\ \vdots \\ \gamma_{t-1}^{(s-2)} \end{bmatrix}_{t-1} + \begin{bmatrix} \omega_t \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \omega_t \sim WN(0, \sigma^2_\omega \mathbb{I})$$

```
sis_dummy_seas <- function(n, s, sd_omega, gamma_init){
  filter(rnorm(n, sd=sd_omega), rep(-1, s-1), "recursive", init=gamma_init)
}

season_dummy <- as.numeric(sis_dummy_seas(200, 12, 0.01, rnorm(11)))
Time <- 1:length(season_dummy)
ggplot() +
  geom_line(aes(y=season_dummy, x=Time))
```



Forma StateSpace

Sostanzialemente sono due set di equazioni: equazione di trasizione (stato) e equazioni di osservazioni(misurazione). L'equazione di **osservazione** ha la sequente forma, indico con (t) componenti che potrebbe essere o non essere variabili nel tempo:

$$y_t = c_{(t)} + Z_{(t)}\alpha_t + \varepsilon_t \quad \varepsilon_t \sim WN(0, H_{(t)})$$

dove y_y è la variabile osservabile e α_t è il vettore che contiene informazioni non osservabili direttamente ma che combinati linearmente mi formano la serie es. il trend, lo slope ecc. L'equazione di **transizione** è della forma:

$$\alpha_{t+1} = d_{(t)} + T_{(t)}\alpha_t + \eta_t \quad \eta_t \sim WN(0, Q_{(t)})$$

Inoltre supponiamo che η_t e ε_t siano tra loro incorrelate (questo per semplicità in generale quest'ipotesi può essere rilassata). I valori iniziali vengono attribuiti in base a una distribuzione (di solito Gaussiano): $\alpha_1 \sim D(a_{1|0}, P_{1|0})$, se voglio esprimere la mia ignoranza posso dire che $P_{1|0} = \infty$ così sono sicuro che il valore iniziale è incluso lì dentro e α_1 è incorrelato da η_t e ε_t.

Oss: α_t, y_t devono essere distribuiti come una normale perché possono essere espressi come combinazioni lineari di η_t, ε_t e α_0 che sono normali.

ATTENZIONE KFAS (il pacchetto di R), cambia un po' le equazioni aggiungendo una matrice davanti a η_t e rimuovendo le costanti $d_{(t)}$:

$$\alpha_{t+1} = T_{(t)}\alpha_t + R_{(t)}\eta_t$$

Possiamo per semplicità assumere $T, R, Z, d, c, Q = \mathbb{E}[\eta_t\eta_t'], H = \mathbb{E}[\varepsilon_t\varepsilon_t']$ costanti nel tempo (in generale possono dipendere dal tempo), questo perché è una condizione necessaria (ma non sufficiente) per la stazionarità nel tempo.

Es 1. con la regressione lineare:

$$\begin{cases} y_t = Z_t(= X_t^T)\alpha_t(= \beta) + \varepsilon_t \\ \alpha_{t+1} = \alpha_t \end{cases}$$

Es 2. Regressione lineare con i coefficienti che evolvono come RW: Identico a prima ma $\alpha_{t+1} = \alpha_t + \eta_t$.

Es 3. AR(2) $y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \varepsilon_t$

$$\begin{cases} y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha_t^{(1)} \\ \alpha_t^{(2)} \end{bmatrix} \\ \begin{bmatrix} \alpha_{t+1}^{(1)} \\ \alpha_{t+1}^{(2)} \end{bmatrix} = \begin{bmatrix} \phi_1 & \phi_2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha_t^{(1)} \\ \alpha_t^{(2)} \end{bmatrix} + \begin{bmatrix} \varepsilon_t \\ 0 \end{bmatrix} \end{cases}$$

quindi in particolare $H = 0$ e $Z = \begin{bmatrix} 1 & 0 \end{bmatrix}$,

$$T = \begin{bmatrix} \phi_1 & \phi_2 \\ 1 & 0 \end{bmatrix} \quad Q = \begin{bmatrix} \sigma_t^2 & 0 \\ 0 & 0 \end{bmatrix}$$

Siccome è un AR(2) a media 0 i valori iniziali sono tali che $a_{1|0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ mentre $P_{1|0} = \begin{bmatrix} \gamma_0 & \gamma_1 \\ \gamma_1 & \gamma_0 \end{bmatrix}$ poiché è la varianza covarianza tra il primo *alpha* e il suo primo ritardo.

Es 4. MA(1) $y_t = \varepsilon_t + \theta \varepsilon_{t-1}$

$$\begin{cases} y_t = \begin{bmatrix} 1 & \theta \end{bmatrix} \begin{bmatrix} \alpha_t^{(1)} \\ \alpha_t^{(2)} \end{bmatrix} \\ \begin{bmatrix} \alpha_{t+1}^{(1)} \\ \alpha_{t+1}^{(2)} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha_t^{(1)} \\ \alpha_t^{(2)} \end{bmatrix} + \begin{bmatrix} \varepsilon_t \\ 0 \end{bmatrix} \end{cases}$$

i valori iniziali sono: $a_{1|0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ mentre $P_{1|0} = \begin{bmatrix} \sigma_\omega^2 & 0 \\ 0 & \sigma_\omega^2 \end{bmatrix}$.

Omettiamo le costanti e usiamo la forma che usa KFAS d'ora in poi:

$$\begin{cases} \alpha_{t+1} = T\alpha_t + R\eta_t \\ y_t = Z\alpha_t + \varepsilon_t \end{cases}$$

Supponiamo di avere LLT (Local Linear Trend) + noise:

$$\mu_{t+1} = \mu_t + \beta_t + \eta_t \beta_{t+1} = \beta_t + \zeta_t$$

supponiamo di osservarlo con del rumore:

$$y_t = \mu_t + \varepsilon_t$$

Osserva che sembra già essere in state space. basta prendere

$$\alpha_t = \begin{bmatrix} \mu_{t+1} \\ \beta_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mu_t \\ \beta_t \end{bmatrix} + \begin{bmatrix} \eta_t \\ \zeta_t \end{bmatrix}$$

quindi in particolare abbiamo $T = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$ e $R = \mathbb{I}_2$ con $Q = \begin{bmatrix} \sigma_\eta^2 & 0 \\ 0 & \sigma_\zeta^2 \end{bmatrix}$ e

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \mu_t \\ \beta_t \end{bmatrix} + \varepsilon_t$$

con $H = \sigma_\varepsilon^2$ e non conoscendo nulla del RW metto $a_{1|0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ e $P_{1|0} = \begin{bmatrix} \infty & 0 \\ 0 & \infty \end{bmatrix}$ Un altro esempio LLT con la componente ciclica stocastico:

$$y_t = \mu_t + \psi_t + \varepsilon$$

$$\begin{bmatrix} \psi_t \\ \psi_{t+1}^* \end{bmatrix} = \rho \begin{bmatrix} \cos\lambda & \sin\lambda \\ -\sin\lambda & \cos\lambda \end{bmatrix} \begin{bmatrix} \psi_t \\ \psi_t^* \end{bmatrix} + \begin{bmatrix} k_t \\ k_t^* \end{bmatrix}$$

per la forma state space applichiamo la formula LEGO:

$$\alpha_t = \begin{bmatrix} \mu_{t+1} \\ \beta_{t+1} \\ \psi_{t+1} \\ \psi^*_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \rho\cos\lambda & \rho\sin\lambda \\ 0 & 0 & -\rho\sin\lambda & \rho\cos\lambda \end{bmatrix} \begin{bmatrix} \mu_t \\ \beta_t \\ \psi_t \\ \psi^*_t \end{bmatrix} + \begin{bmatrix} \eta_t \\ \zeta_t \\ k_t \\ k^*_t \end{bmatrix}$$

e

$$y_t = \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix} \alpha_t + \varepsilon_t$$

Per la formula LEGO basta concatenare i vettori mettendo uno sopra l'altro, mentre le matrici vanno concatenate diagonalmente.

Filtro si Kalman

La forma state space serve per il filtro di Kalman, che è un algoritmo ricorsivo che fa 2 passi: $a_{t|s} = \mathbb{P}[\alpha_t|y_1, ..., y_s]$ sotto iptosi di Gaussianità è anche il valore atteso condizionato (predittore ottimale) quindi il mio valore atteso $\alpha_{t|s}$ diventa la proiezione del vettore di stato al tempo t con dati i noti fino ad s (oss: può essere sia nel presente che nel futuro). Abbiamo inoltre l'incertezza data dalla matrice di varianza covarianza degli errori $P_{t|s} = \mathbb{E}[(\alpha_t - a_{t|s})(\alpha_t - a_{t|s})^T]$.\

- Se $t > s$ è caso di forecast
- Se $t = s$ è una stima a tempo reale è chiamato filter
- Se $t < s$ è lo smoother

KF a due passi funziona così, nota che è ricorsivo:

$$(a_{t|t-1}, P_{t|t-1}) \to (a_{t|t}, P_{t|t}) \to (a_{t+1|t}, P_{t+1|t})$$

e all'inzio del modello state space si fornisce $(a_{1|0}, P_{1|0})$. Quindi KF ci calcola il Filter (quando t=s) e restituisce le previsioni un passo in avanti: $\hat{y}_{t|t-1} = Z_{(t)} a_{t|t-1}$ e posso definire l'innvoazione: $i = y_t - \hat{y}_{t|t-1}$ e si defisce anche la sua varianza $F_t = \mathbb{E}[i \cdot i^T]$ Se assumi la gaussianità cioé $y_t|y_1, ..., y_{t-1} \sim N(\hat{y}_{t|t+1}, F_t)$ allora possiamo costruire la funzione di massima verosimglianza:

$$L(\theta) = \sum_{t=1}^{n} -\frac{1}{2}\Big(\log(\det(F_t)) + (y_t - \hat{y}_{t|t-1})^T F^{-1} (y_t - \hat{y}_{t|t-1}) \Big)$$

Se i dati non sono Gaussiana posso considerarla comunque come una funzione di perdita generale (come MSE), quindi comunque tutto funziona bene. In realtà nella funzione di massima verosomiglianza abbiamo che sia F_t che $\hat{y}$ variano al variare dei parametri θ, quindi la ottimizzazione è numerica e non analitica inoltre il KF ci trova sia $\hat{y}_{t|t-1}$ che la sua matrice di varianza covarianza F_t. Esistono anche funzioni smoother per fare queste stime.

```r
library(KFAS)
y <- log(AirPassengers)
y[121:144] <- NA

#SSMseasonal crea solo cicli stazionari (non stocastici)
mod1 <- SSModel(y ~ 0 + SSMtrend(2, list(NA, NA)) + # 2 per LLT(local linear trend)
                  SSMseasonal(12, NA, "dummy"), #NA nella varianza indica che non la so
                H = NA)
```

```r
#non stampo usa troppo spazio per niente
mod1$T #la matrice T di state space (la sua terza
# dimesione indica la possibile variazione nel tempo) (13x13)

mod1$Q #3 errori quindi una matrice 3x3

mod1$R #matrice R che distribuisce i 3 errori sulle variabili quindi deve essere 13x3
```

```r
vary <- var(y, na.rm = TRUE) #la varianza di y che serve da stima per i parametri iniziali
pars <- numeric(4) #parametri iniziali
#si usano log varianze perché siccome le varianze sono sempre positive ma il computer può
#portarle ad essere negative e ciò crea casini e passando all'exp diventa positivo.
pars[1] <- log(vary/10) #log varianza del livello
pars[2] <- log(0.1) # per la percentuale dello slope metto un valore piccolo
pars[3] <- log(0.1) # anche per la stagionalità
pars[4] <- log(vary/10)# errore di osservazione

fit1 <- fitSSM(mod1, inits = pars)
fit1$optim.out$convergence #se è 0 vi è convergenza
```

```
## [1] 0
```

```r
fit1$model$Q #osserva non sono più NA come nel mod1$Q o meglio ha stimato i valori
```

```
## , , 1
##
##              [,1]         [,2]         [,3]
## [1,] 3.830231e-24 0.000000e+00 0.000000e+00
## [2,] 0.000000e+00 2.114698e-69 0.000000e+00
## [3,] 0.000000e+00 0.000000e+00 7.300983e-12
```

```r
#la update function suppone che gli unici parametrici sono le varianze
# e che siano le log varianze

#la update function deve essere così:
updatefn(pars, model, ...){
  #pars=parametri da stimare
  transform(pars) #la mappa da applicare ai paramtri (es. Logistic, exp)
  putInModelMatrix(pars) #metti nelle matrici giuste i vari parametri
  model #return model
}
```

```r
library(KFAS)
y <- log(AirPassengers)
y[121:144] <- NA

mod1 <- SSModel(y ~ 0 + SSMtrend(2, list(NA, NA)) + # 2 per LLT(local linear trend)
                  SSMseasonal(12, NA, "dummy"), #NA nella varianza indica che non la so
                H = NA)
vary <- var(y, na.rm = TRUE)
pars <- numeric(4)
pars[1] <- log(vary/10)
pars[2] <- log(0.01) #metto un valore piccolo
pars[3] <- log(0.01)
pars[4] <- log(vary/10)

fit1 <- fitSSM(mod1, inits = pars, control=list(maxit=1000))
fit1$optim.out$convergence #se è 0 vi è convergenza
```

```
## [1] 0
```

```r
fit1$optim.out
```

```
## $par
```

```
## [1]  -7.135179 -29.414042 -10.213653  -8.867935
##
## $value
## [1] -185.4886
##
## $counts
## function gradient
##      501       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

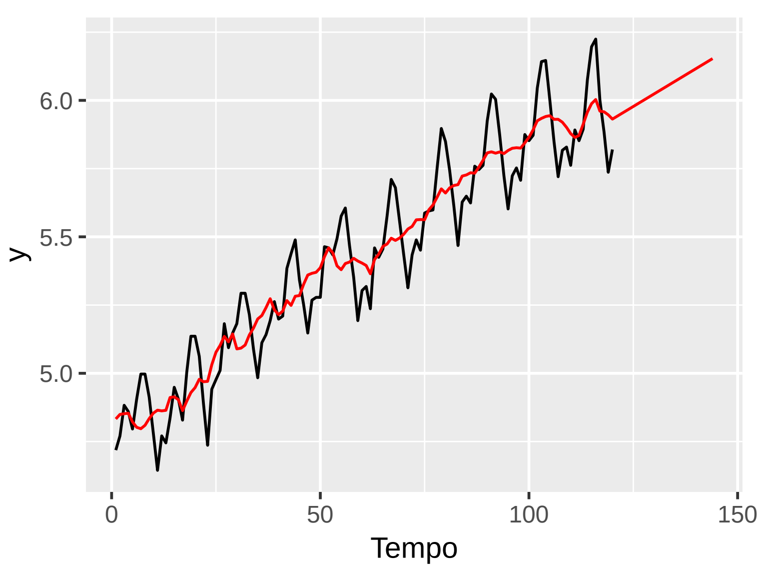
```
fit1$model$Q #osserva non sono più NA come nel mod1$Q
```

```
## , , 1
##
##              [,1]         [,2]         [,3]
## [1,] 0.0007965835 0.000000e+00 0.000000e+00
## [2,] 0.0000000000 1.681295e-13 0.000000e+00
## [3,] 0.0000000000 0.000000e+00 3.666626e-05
```

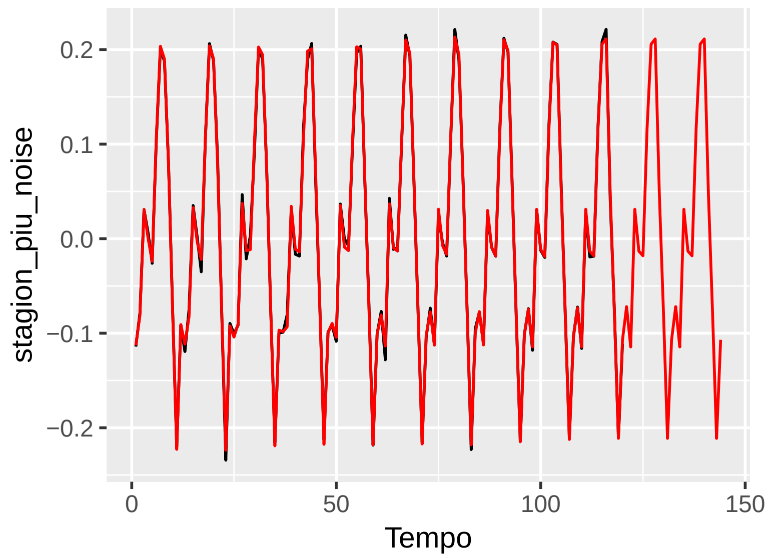
```
library(ggplot2)
smo1 <- KFS(fit1$model, smoothing = c("state","disturbance","signal"))
colnames(smo1$alphahat)
```

```
##  [1] "level"       "slope"       "sea_dummy1"  "sea_dummy2"  "sea_dummy3"
##  [6] "sea_dummy4"  "sea_dummy5"  "sea_dummy6"  "sea_dummy7"  "sea_dummy8"
## [11] "sea_dummy9"  "sea_dummy10" "sea_dummy11"
```

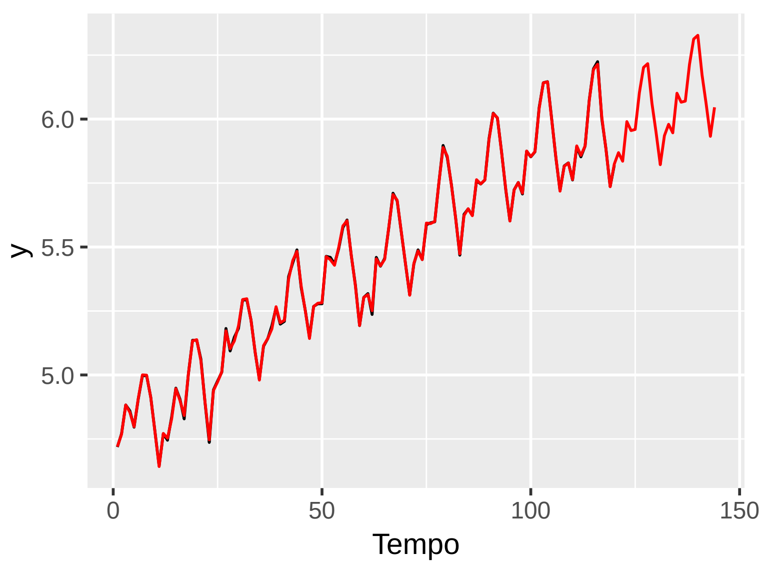
```
Tempo = 1:length(y)
y = as.numeric(y)
ggplot() +
  geom_line(aes(y=y, x=Tempo)) +
  geom_line(aes(y=smo1$alphahat[, "level"], x=Tempo), colour='red')
```



```r
stagion_piu_noise <- as.numeric(y-smo1$alphahat[, "level"])
ggplot()+
  geom_line(aes(y=stagion_piu_noise, x=Tempo)) +
  geom_line(aes(y=smo1$alphahat[, "sea_dummy1"], x=Tempo), colour='red') #è la stagionalità
```

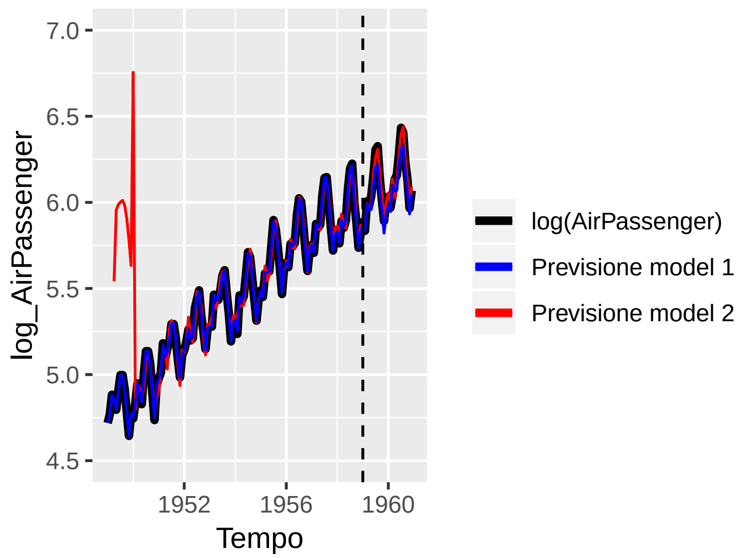



```r
#muhat è serie - noise ma il noise è basso quindi coincide praticamente con la serie
ggplot() +
  geom_line(aes(y=y, x=Tempo)) +
  geom_line(aes(y=smo1$muhat, x=Tempo), colour='red')
```

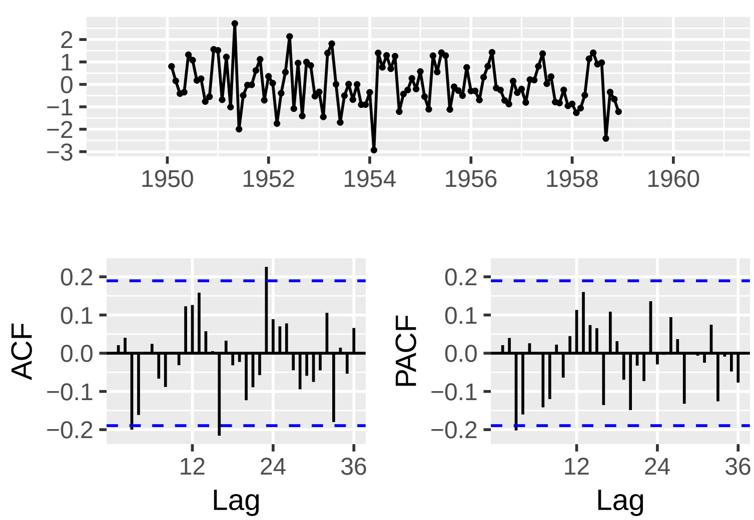



```r
#per previsioni 1step ahead
mod11 <- fit1$model #rubo il modello da fit1
mod11$y[] <- log(AirPassengers) #passo tutta la serie (è illegale ma si fa lo stesso)

smo11 <- KFS(mod11, filtering = "signal") #filtro solo il segnale

Tempo = time(log(AirPassengers))
```

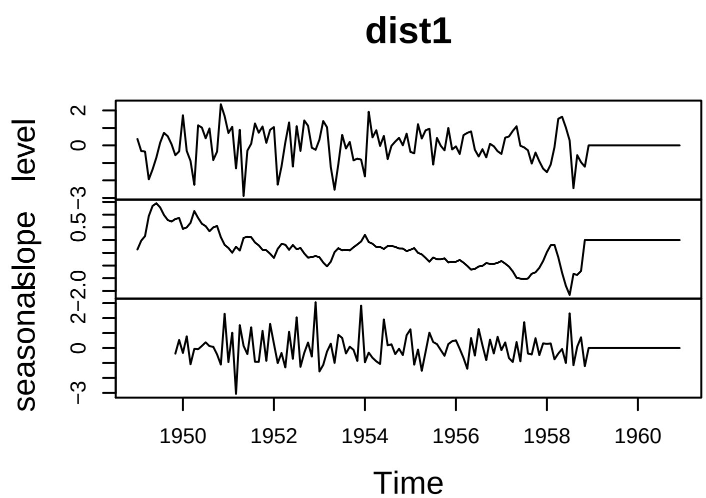
```r
df <- data.frame(as.numeric(Tempo), "log_AirPassenger" = as.numeric(log(AirPassengers)),
                 "Prev_11" = as.numeric(smo11$m), "Prev_1" = as.numeric(smo1$muhat))
ggplot(data=df, aes(x=Tempo)) +
  geom_line(aes(y=log_AirPassenger, color="log(AirPassenger)"), lwd=1.5) +
  geom_line(aes(y=Prev_11, color="Previsione model 2"))+
  geom_line(aes(y=Prev_1, color="Previsione model 1"))+
  scale_color_manual(values = c("log(AirPassenger)" = "black",
                                "Previsione model 2" = "red",
                                "Previsione model 1" = "blue"),
                     name = "") +
  geom_vline(xintercept = 1959, lty=2) +
  ylim(4.5, 7)
```




```r
stdresid <- rstandard(smo1, "recursive")
# dà diversi tipi di residui standardizzati
forecast::ggtsdisplay(stdresid)
```



```r
dist1 <- rstandard(smo1, "state") #disturbance
#tendenzialmente se sta tra -2 e 2 va bene, altrimenti forse c'è stato un evento
# che ha causato una botta
plot(dist1)
```




```r
osserr1 <- rstandard(smo1, "pearson")
ggplot() +
  geom_line(aes(x=Tempo, y=osserr1))
```