Time Series Analysis

Time series (explain also random process)

$x_1, x_2, \ldots, x_t, \ldots$

$\{x\}_t$ or $\{x_t\}$

Multivariate Time series

$x_1, x_2, \ldots, x_t, \ldots$

$y_1, y_2, \ldots, y_t, \ldots$

$z_1, z_2, \ldots, z_t, \ldots$

Example 1: Currency Time series

$(EUR/USD)_1, (EUR/USD)_2, \ldots, (EUR/USD)_t, \ldots$

$(GBP/USD)_1, (GBP/USD)_2, \ldots, (GBP/USD)_t, \ldots$

Example 2: Currency Time series including bid-ask spread

$(EUR/USD)_{\text{ask},1}, (EUR/USD)_{\text{ask},2}, \ldots, (EUR/USD)_{\text{ask},t}, \ldots$

$(EUR/USD)_{\text{bid},1}, (EUR/USD)_{\text{bid},2}, \ldots, (EUR/USD)_{\text{bid},t}, \ldots$

$(GBP/USD)_{\text{ask},1}, (GBP/USD)_{\text{ask},2}, \ldots, (GBP/USD)_{\text{ask},t}, \ldots$

$(GBP/USD)_{\text{bid},1}, (GBP/USD)_{\text{bid},2}, \ldots, (GBP/USD)_{\text{bid},t}, \ldots$

Time: multiple understanding: physical time (GMT), alternatives: trading time (number of trades, volume), number of quotes (high frequency trading, algotrading), problem of weekends

Descriptive statistics of time series

Mean: $\mathrm{E}(x_t) = \mu_x = \text{const}$

Standard deviation and variance: $\mathrm{var}(x_t) = \sigma_x^2$

Autocorrelation function (ACF):

$$\gamma_k = \mathrm{cov}(x_t, x_{t-k}) = \mathrm{E}[(x_t - \mu)(x_{t-k} - \mu)] \tag{1}$$

$$\rho_k = \frac{\gamma_k}{\gamma_0} = \frac{\gamma_k}{\sigma_x^2} = \mathrm{corr}(x_t, x_{t-k})$$

Cross correlation function

$$\tilde{\rho}_k = \mathrm{corr}(x_t, y_{t-k})$$

Parcial correlations

Nonlinear type statistics: higher order correlation functions,
$\mathrm{E}[(x_t - \mu)^2(x_{t-k} - \mu)]$
$\mathrm{E}[(x_t - \mu)^2(x_{t-k} - \mu)^2]$
$\mathrm{E}[|x_t - \mu|^q|x_{t-k} - \mu|^r]$

Structure functions, Hurst exponents, correlation dimensions ...

Stationary process

In mathematics, a stationary process (or *strictly stationary process* or strongly stationary process) is a stochastic process whose *joint probability distribution* does not change when shifted in time or space. Consequently, parameters such as the mean and variance, if they are present, also do not change over time or position.

Let $\{x_t\}$ be a stochastic process and let $F_x(x_{t_1+\tau}, \ldots, x_{t_k+\tau})$ represents cumulative distribution function of $\{x_t\}$ at times $t_1+\tau, \ldots, t_k+\tau$. Then $\{x_t\}$ is said to be stationary if for all k, for all τ and for all t_1, $t_2, \ldots, t_k$

$$F_x(x_{t_1}, x_{t_2}, \ldots, x_{t_k}) = F_x(x_{t_1+\tau}, x_{t_2+\tau}, \ldots, x_{t_k+\tau})\,. \tag{2}$$

Since τ does not affect $F_x(.)$, F_x is not a function of time.

Weak or wide sense stationary

Weak or wide sense (WSS) stationary random processes only require that 1st moment and covariance do not vary with respect to time. [Any strictly stationary process which has a mean and a covariance is also WSS].

The restriction on the mean

$$\mu_x = \mathrm{E}[x_t] = \mathrm{E}[x_{t+\tau}]\,, \qquad \forall \tau \in R$$

restriction on the autocovariance

$$\gamma_{t_1,t_2} = \mathrm{E}[(x_{t_1} - \mu_x)(x_{t_2} - \mu_x)]$$

is

$$\gamma_{t_1,t_2} = \gamma_{t_1+\tau,t_2+\tau}$$

As $t_2 = -\tau$ we obtain

$$\gamma_{t_1,t_2} = \gamma_{t_1-t_2,0}$$

Other terminology:

- stationary up to order m
- second-order stationarity
- trend stationary

Trend stationary

Stochastic process is *trend stationary* if an underlying trend (function solely of time) can be removed, leaving a stationary process. A process $\{x_t\}$ is said to be trend stationary if

$$x_t = f(t) + e_t \tag{3}$$

where $f(t)$ is any function mapping from the reals to the reals, and $\{e_t\}$ is a stationary process.

Example:

Many economic time series are characterized by exponential growth. The example is

$$GDP_t = B\, \exp(at)\, U_t$$

$$\underbrace{\ln(GDP_t)}_{x_t} = \underbrace{\ln(B) + at}_{f(t)} + \underbrace{\ln U_t}_{e_t}$$

Data filtering and data standardization:

The raw data are often transformed to become stationary.

The procedures may include:

- data standardization (e.g. z-score)
- noise suppression methods (**convolutions**)
- time series decomposition :

  (a) identification of **seasonal** regularities

  (b) trend identification/elimination procedures

  (c) identification of the **periodicity** (DFT - discrete fourier transform)

Statistical data tests:

- tests of **stationarity**
- tests of randomness (runs test)
- tests of the Gaussian vs. non-Gaussian behaviour (Lévy flight) **fat tailed behaviour** fluctuations

linear in parameters autoregression: (Box-Jenkins methodology):
models:
MA, AR, ARMA, ARIMA, ARCH, GARCH, VAR, VECM

MA - moving average model
AR - autoregressive model
ARMA - autoregressive moving average model
ARIMA - autoregressive integrated moving average models
ARCH - autoregressive conditional heteroskedasticity models
GARCH - generalized autoregressive conditional heteroskedasticity models
VAR - vector autoregression
VECM - vector error correction model

Data standardization and normalization (standardizing raw data series)

Motivation

A. *(standardization- normalization of the data step is very important when dealing with parameters of different units and scales. For example, some data mining techniques use the Euclidean distance. Therefore, all parameters should have the same scale for a fair comparison between them.*

B*: The importance of standardizing variables for multivariate analysis. Without standardization, the variables measured at different scales do not contribute equally to the analysis.*

1a. $z_t = \frac{x_t - \mathrm{E}(x_t)}{\hat{\sigma}_x}$ (z-score scaling)

$\mathrm{E}(z) = 0$, $\hat{\sigma}_z = 1$

1b. $z_t = \frac{x_t - \mathrm{median}(x_t)}{\mathrm{E}(|x_t - \mathrm{median}(x_t)|)}$

---

2. $z_t = \frac{x_t}{\mathrm{E}(x_t)}$, $\hat{E}(z) = 1$ (transform into dimensionless case)

3. 0-1 scaling since $z_t \in < 0, 1 >$, This method allows variables to have differing means and standard deviations but equal ranges.

$$z_t = \frac{x_t - \min_{\tau \in [t-T,t]} x_\tau}{\max_{\tau \in [t-T,t]} x_\tau - \min_{\tau \in [t-T,t]} x_\tau}$$

4.a. $z_t \equiv z_{t,T} = x_t - x_{t-T}$ (returns, stationarity)

4.b. $z_t = \frac{x_t - x_{t-T}}{x_t}$

4.c. $z_t = \frac{x_t - x_{t-T}}{x_{t-T}}$

4.d. $z_t = \frac{2(x_t - x_{t-T})}{x_t + x_{t-T}}$

4.e. higher order differences; the differencing is applied to the series before estimating models. Differencing is necessary when trends are present (series with trends are typically nonstationary and e.g. ARIMA modeling assumes stationarity) and is used to remove their effect.

5.a. $z_t = \log(x_t)$ (log-transform transformation, size effect)

5.b. $z_t = \frac{(x_t + c)^\lambda - 1}{\lambda}$ (Box-Cox transform, size)

5.c. $z_t = \log(x_t) - \log(x_{t-T})$ (log returns)

6.a. $z_t = x_t - \hat{x}_t|\text{history})$ (deviations from the predictions)

6.b. $z_t = x_t - \hat{\text{MA}}_{[t-T,t]}(x_\tau)$

6.c. $z_t = x_t - \hat{\text{EMA}}_{[t-T,t]}(x_\tau)$

6.d. $z_t = x_t - \hat{\text{Trend}}_{[t-T,t]}(x_\tau)$

```
**********************************
Example of (0,1) scaling in R:
**********************************

> x=rnorm(10,0,2)
> x
 [1] -3.6009900 -0.5465578  2.9292530  0.3562723 -0.6894619  1.1027583
 [7] -4.7105555  3.0792583  3.9019479 -4.3988261
> X=rnorm(10,0,20)
> X
 [1]  14.286801   9.420125  10.252706  38.964826 -16.660802  -9.404258
 [7]  -9.292123  -4.677444   7.710518 -15.567768
> xmin=min(x)
> xmin
[1] -4.710556
> Xmin=min(X)
> Xmin
[1] -16.6608
> xmax=max(x)
> xmax
[1] 3.901948
> Xmax=max(X)
> Xmax
[1] 38.96483
> z=(x-xmin)/(xmax-xmin)
> Z=(X-Xmin)/(Xmax-Xmin)
> z
 [1] 0.12883194 0.48348285 0.88706014 0.58831068 0.46689022 0.67498537
 [7] 0.00000000 0.90447730 1.00000000 0.03619498
> Z
 [1] 0.55635512 0.46886531 0.48383288 1.00000000 0.00000000 0.13045324
 [7] 0.13246913 0.21542871 0.43813113 0.01964983
**********************************************************************
```

```
*************************
Example: z score scaling
*************************
  z1=(x-mean(x))/sd(x)
> z1
 [1] -1.06169658 -0.09173266  1.01204403  0.19496958 -0.13711320  0.43202331
 [7] -1.41404964  1.05967965  1.32093253 -1.31505702
> sd(z1)
[1] 1
> mean(z1)
[1] 5.561957e-18
*************************
```

Autocorrelation function, Partial Correlation function Cross Correlation function

implementation in R

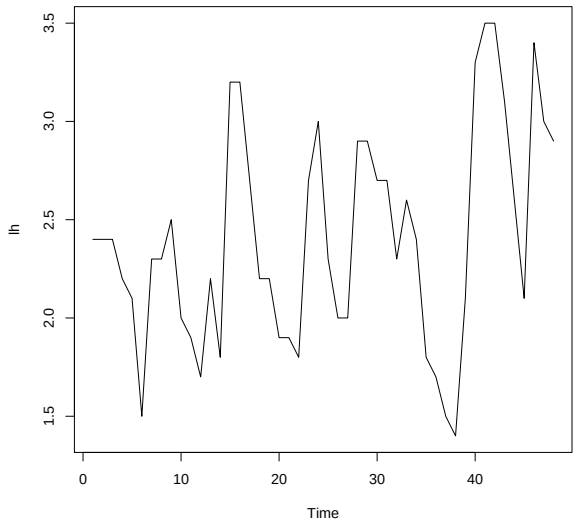
Partial Autocorrelation function of lag k is another way to measure the connection between variables x_t and $x_{t+\tau}$; the purpose is to filter out the linear influence of the random variables $x_{t+1}, \ldots, x_{t+\tau-1}$ that lie in between x_t and $x_{t+\tau}$ and then calculate the correlation of the transformed random variables:

$$\begin{aligned} \text{pacf}_1 &= \text{cor}(x_t, x_{t+1})\,, \\ \text{pacf}_k &= \text{cor}\Big(x_{t+k} - \text{Proj}(x_{t+k}|x_{t+1}, \ldots, x_{t+k-1}), \\ &\qquad\qquad x_t - \text{Proj}(x_t|x_{t+1}, \ldots, x_{t+k-1})\ \Big) \end{aligned}$$

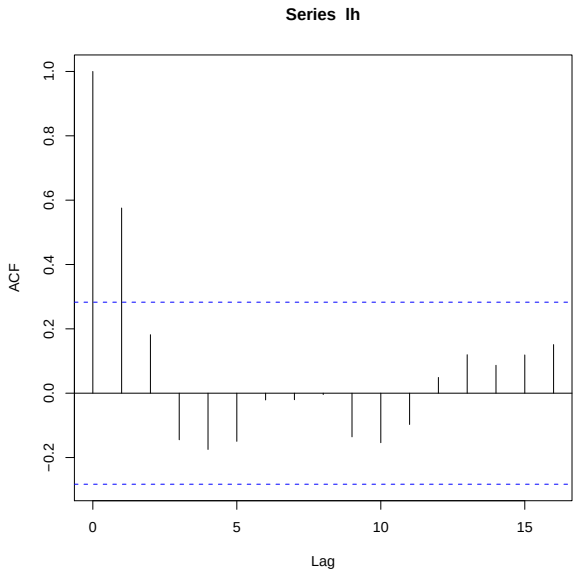
```
R documentation

acf {stats}

Auto- and Cross- Covariance and -Correlation Function Estimation

Description

The function acf computes (and by default plots)

estimates of the autocovariance or autocorrelation function.

Function pacf is the function used for the partial autocorrelations.

Function ccf computes the cross-correlation or cross-covariance
of two univariate series.
```

```
acf(x, lag.max = NULL,
    type = c("correlation", "covariance", "partial"),
    plot = TRUE,
    na.action = na.fail, demean = TRUE, ...)

pacf(x, lag.max, plot, na.action, ...)
```

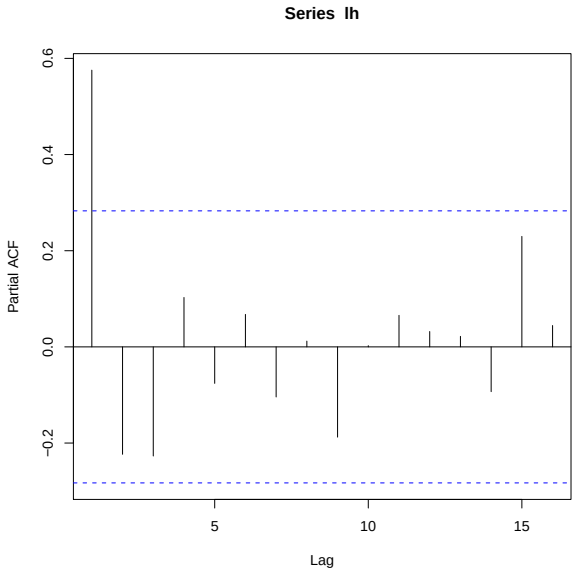
```
**********************************************************************
data lh in R
## Examples from Venables & Ripley
lh data luteinizing Hormone in Blood Samples
Description: A regular time series giving the luteinizing hormone in blood
samples at 10 mins intervals from a human female, 48 samples
**********************************************************************

lh
Time Series:
Start = 1
End = 48
Frequency = 1
 [1] 2.4 2.4 2.4 2.2 2.1 1.5 2.3 2.3 2.5 2.0 1.9 1.7 2.2 1.8 3.2 3.2 2.7 2.2
2.2
[20] 1.9 1.9 1.8 2.7 3.0 2.3 2.0 2.0 2.9 2.9 2.7 2.7 2.3 2.6 2.4 1.8 1.7 1.5
1.4
[39] 2.1 3.3 3.5 3.5 3.1 2.6 2.1 3.4 3.0 2.9
> ?lh
```



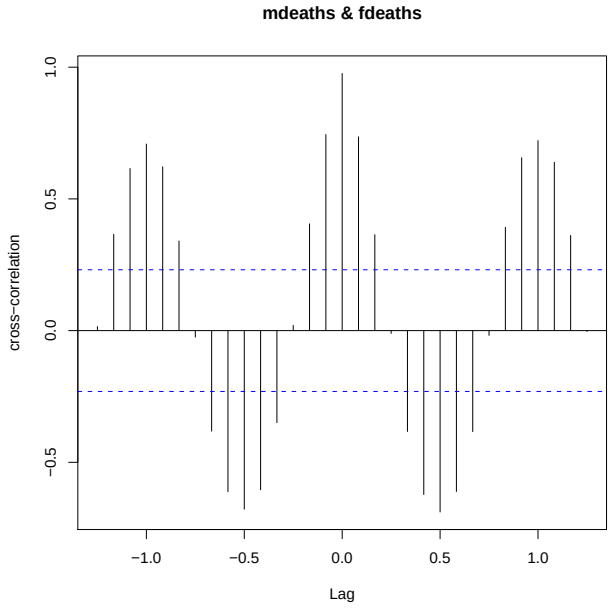

```
> plot(acf(lh))
```




```
> plot(pacf(lh))
```


Series lh
Partial ACF
0.6
0.4
0.2
0.0
−0.2
5
10
15
Lag

```
data ldeaths, mdeaths, fdeaths
**********************************************************************

UKLungDeaths              package:datasets              R Documentation

Monthly Deaths from Lung Diseases in the UK

Description:

     Three time series giving the monthly deaths from bronchitis,
     emphysema and asthma in the UK, 1974-1979, both sexes (ldeaths),
     males (mdeaths) and females (fdeaths).
**********************************************************************

ccf(mdeaths, fdeaths, ylab = "cross-correlation")
```



Linear filtering

Filtering - one of the most basic tools for transforming time series (e.g. for eliminating seasonality) is linear filtering.
original time series: $\{X\}_t$
transformed - filtered time series: $\{\hat{x}\}_t$

As an important class of linear filters are finite moving averages, transformations that replace the original raw data X_t by the weighted sum

$$\hat{x}_t = \sum_{j=-r}^{j=s} a_j X_{t+j}\,, \qquad t = r+1, \ldots, n-s$$

If $s = r$, the filter is called symmetric.

Example: $s = r = 2$, symmetric filter determined by 5-tuple of coefficients $(a_{-2}, a_{-1}, a_0, a_1, a_2)$

$$\hat{x}_t = a_{-2}X_{t-2} + a_{-1}X_{t-1} + a_0 X_t + a_1 X_{t+1} + a_2 X_{t+2} \tag{1}$$

Try setting: $(1/4, 1/2, 1, 1/2, 1/4)$

$$\hat{x}_t = (1/4)X_{t-2} + (1/2)X_{t-1} + X_t + (1/2)X_{t+1} + (1/4)X_{t+2} \tag{2}$$

In R the function filter() permits of fairly general filters, its argument filter takes a vector containing the coefficients a_j.

Illustration - Rewrite the aforementioned sum

$$\hat{x}_{r+1} = a_{-r}X_1 + a_{-r+1}X_2 + \ldots + a_s X_{r+1+s}$$

$$\hat{x}_{r+2} = a_{-r}X_2 + a_{-r+1}X_3 + \ldots + a_s X_{r+2+s}$$

$$\hat{x}_{r+3} = a_{-r}X_3 + a_{-r+1}X_4 + \ldots + a_s X_{r+3+s}$$

$$\hat{x}_{n-s-1} = a_{-r}X_{n-s-1-r} + a_{-r+1}X_{n-s-r} + \ldots + a_s X_{n-1}$$

$$\hat{x}_{n-s} = a_{-r}X_{n-s-r} + a_{-r+1}X_{n-s-r+1} + \ldots + a_s X_n$$

Illustration: $n = 10$, $s = 2$, $r = 1$

$\hat{x}_2 = a_{-1}X_1 + a_0X_2 + a_1X_3 + a_2X_4 = (a_{-1}, a_0, a_1, a_2) \cdot (X_1, X_2, X_3, X_4)$

$\hat{x}_3 = a_{-1}X_2 + a_0X_3 + a_1X_4 + a_2X_5 = (a_{-1}, a_0, a_1, a_2) \cdot (X_2, X_3, X_4, X_5)$

$\hat{x}_4 = a_{-1}X_3 + a_0X_4 + a_1X_5 + a_2X_6 = (a_{-1}, a_0, a_1, a_2) \cdot (X_3, X_4, X_5, X_6)$

$\hat{x}_5 = a_{-1}X_4 + a_0X_5 + a_1X_6 + a_2X_7 = (a_{-1}, a_0, a_1, a_2) \cdot (X_4, X_5, X_6, X_7)$

$\hat{x}_6 = a_{-1}X_5 + a_0X_6 + a_1X_7 + a_2X_8 = (a_{-1}, a_0, a_1, a_2) \cdot (X_5, X_6, X_7, X_8)$

$\hat{x}_7 = a_{-1}X_6 + a_0X_7 + a_1X_8 + a_2X_9 = (a_{-1}, a_0, a_1, a_2) \cdot (X_6, X_7, X_8, X_9)$

$\hat{x}_8 = a_{-1}X_7 + a_0X_8 + a_1X_9 + a_2X_{10} =$
$(a_{-1}, a_0, a_1, a_2) \cdot (X_7, X_8, X_9, X_{10})$

Illustration MA: (Moving average - technical analysis)
(Only the past values are known!)

MA:

$r = 1$, $s = 0$

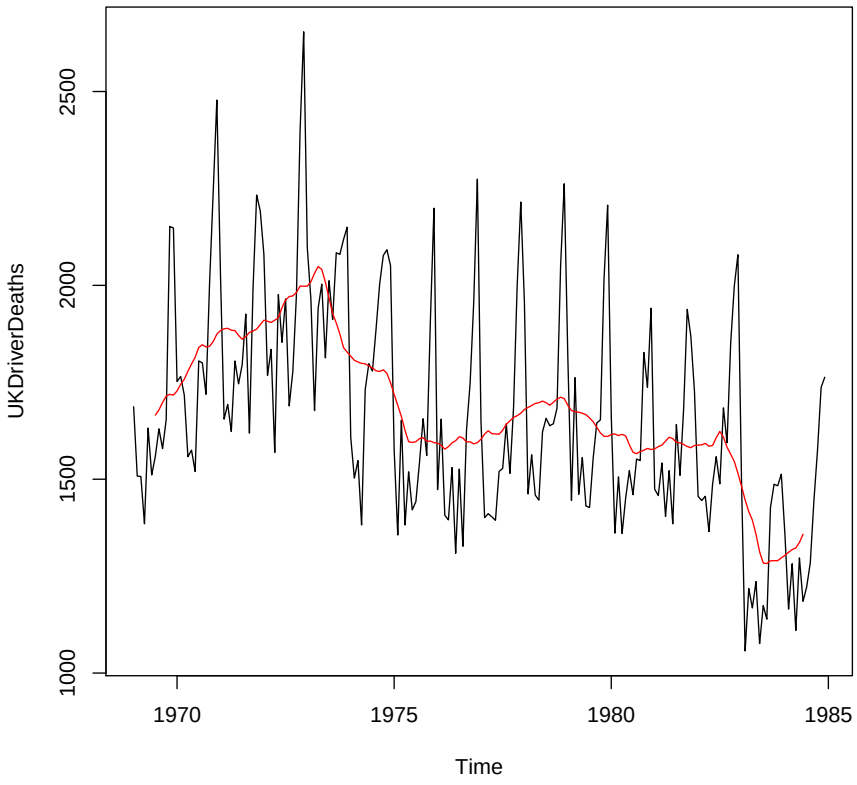
$(a_{-1}, a_0) = (1/2, 1/2)$

$\hat{x}_t = a_{-1}X_{t-1} + a_0X_t = (X_t + X_{t-1})/2$

```
R application

filter          package:stats     R Documentation

Linear Filtering on a Time Series

Description:

     Applies linear filtering to a univariate time series or to each
     series separately of a multivariate time series.

Usage:

     filter(x, filter, method = c("convolution", "recursive"),
            sides = 2, circular = FALSE, init)

  method: Either "convolution" or "recursive" (and can be
          abbreviated). If "convolution" a moving average is used: if
          "recursive" an autoregression is used.

   sides: for convolution filters only. If sides=1 the filter
          coefficients are for past values only; if sides=2 they are
          centred around lag 0. In this case the length of the filter
          should be odd, but if it is even, more of the filter is
          forward in time than backward.
```

The filter c(1/2,rep(1,11),1/2)/12) eliminates seasonability

```
c(1/2,rep(1,11),1/2)/12) =
[1]  0.04166667 0.08333333 0.08333333 0.08333333 0.08333333 0.08333333
[7]  0.08333333 0.08333333 0.08333333 0.08333333 0.08333333 0.08333333
[13] 0.04166667

> plot(UKDriverDeaths)
> lines(filter(UKDriverDeaths,c(1/2,rep(1,11),1/2)/12),col=2)
```


UKDriverDeaths
1000
1500
2000
2500
1970
1975
1980
1985
Time

Decomposition

Filtering with moving averages can also be used for additive or multiplicative decomposition into seasonal, trend, and irregular components. The classical approach to this task, implemented in the function decompose(), is to take simple symmetric filter as illustrated above for extracting and derive seasonal component by averaging the trend-adjusted observations from the corresponding periods.

```
Description:

     Decompose a time series into seasonal, trend and irregular
     components using moving averages.  Deals with additive or
     multiplicative seasonal component.

Usage:

     decompose(x, type = c("additive", "multiplicative"), filter = NULL)

Arguments:

       x: A time series.

    type: The type of seasonal component. Can be abbreviated.

  filter: A vector of filter coefficients in reverse time order (as for
          AR or MA coefficients), used for filtering out the seasonal
          component. If NULL, a moving average with symmetric window
          is performed.
```

The additive model used (R):

$$x_t = T_t + S_t + e_t$$

The multiplicative model (R):

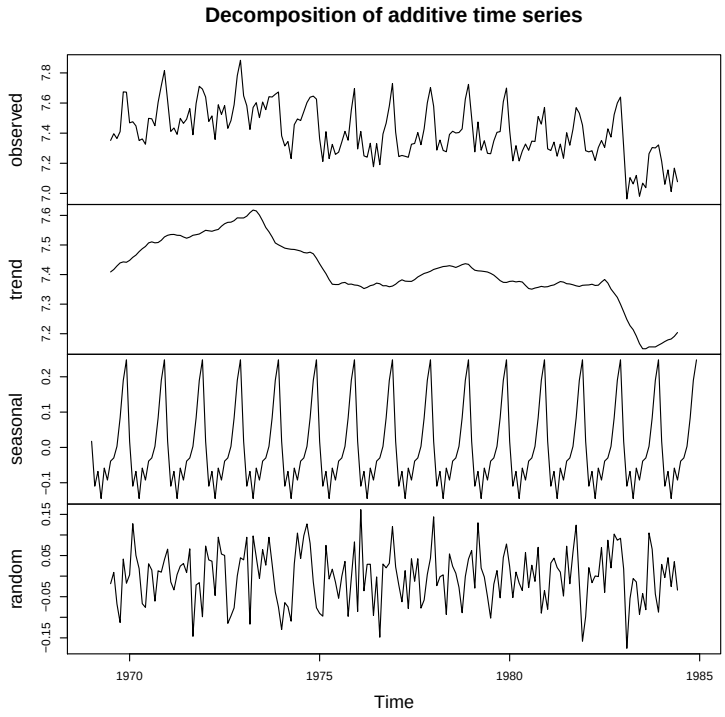
$$x_t = T_t S_t e_t$$

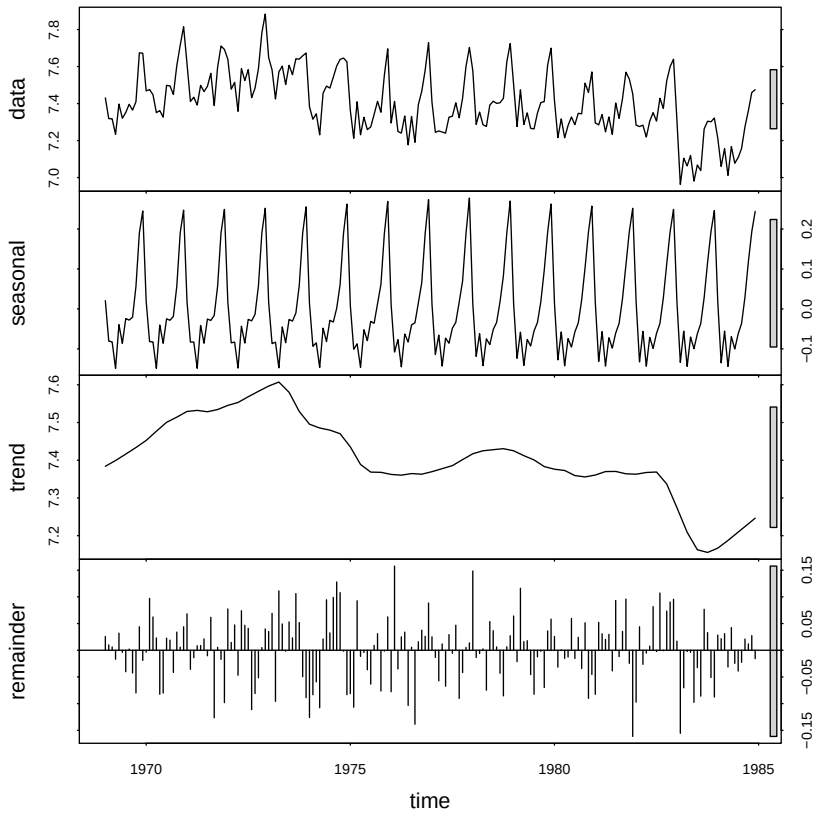
1. the trend component T_t is determined using a moving average (if filter is NULL, a symmetric window with equal weights is used), and removes it from the time series.

2. the seasonal component S_t is computed by averaging, for each time unit, over all periods. The seasonal figure is then centered.

3. the error component e_t is determined by removing trend and seasonal figure (recycled as needed) from the original time series.

```
> dd_dec<-decompose(log(UKDriverDeaths))
> dd_stl<-stl(log(UKDriverDeaths),s.window=13)
> plot(dd_stl)
> plot(dd_dec)
```





the access to signal components can be done by means of

dd_dec$trend

dd_dec$figure (seasonal)

dd_dec$random (remainder)

Grounds of linear autoreggression
Moving-average (MA) model

In time series analysis, the moving-average (MA) model is a common approach for modeling univariate time series models. The notation MA(q) refers to the moving average model, the value of q is called the order of the MA model.

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

Alternative description in terms of lag operators

$X_t - \mu = \theta(\hat{L})\epsilon_t$

$\theta(\hat{L}) = +1 + \theta_1 \hat{L} + \cdots + \theta_q \hat{L}^q$

μ is the mean of the series, the

θ_1, ..., θ_q are the parameters of the model and the

ϵ_t, ϵ_{t-1},...ϵ_{t-q} are white noise error terms with the properties

$\mathrm{E}(\epsilon_t) = 0$
$\mathrm{E}(\epsilon_t^2) = \sigma_\epsilon^2$,
$\mathrm{E}(\epsilon_t \epsilon_{t'}) = 0$ (as $t \neq t'$)

Consequences:
$\mathrm{E}(X_t) = \mu$

$\sigma_X^2 = \left(1 + \theta_1^2 + \cdots + \theta_q^2\right)\sigma_\epsilon^2$
$\rho_k = 0$, as $k > q$

Special case MA(2), $q = 2$

$$X_t = \mu + \epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2}$$

$$\sigma_X^2 = \left(1 + \theta_1^2 + \theta_2^2\right)\sigma_\epsilon^2$$

$$\rho_1 = \mathrm{corr}(X_t, X_{t-1}) = \frac{\theta_1(1+\theta_2)}{1+\theta_1^2+\theta_2^2}$$

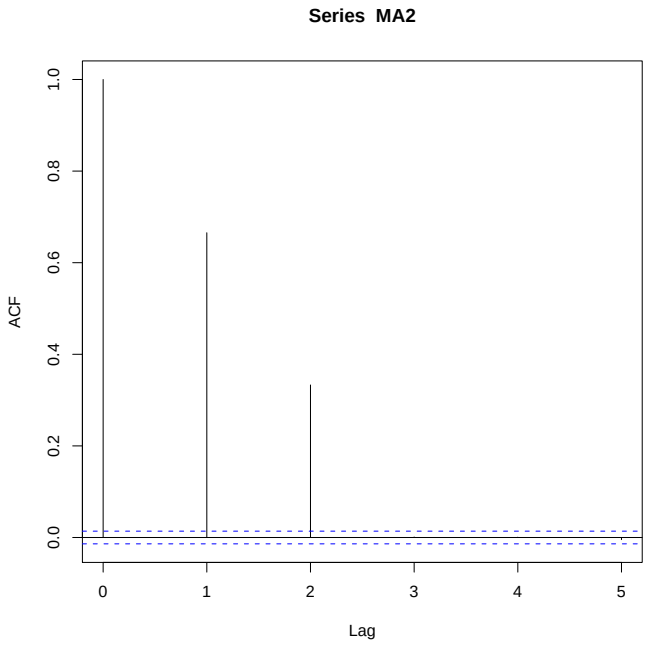
$$\rho_2 = \mathrm{corr}(X_t, X_{t-2}) = \frac{\theta_2}{1+\theta_1^2+\theta_2^2}$$

$$\rho_3 = 0$$

Example: $\theta_1 = 1$, $\theta_2 = 1$, $\sigma_X^2 = 3\sigma_\epsilon^2 = 3$

```
> MA2<-arima.sim(model=list(ma=c(1,1)),n=20000) # sd=1
> mean(MA2)
[1] -0.0007542627
> sd(MA2)^2
[1] 3.003547
```

```
# verify relation acf,
> th1=1
> th2=1
> th1*(1+th2)/(1+th1*th1+th2*th2)
[1] 0.6666667
> th2/(1+th1*th1+th2*th2)
[1] 0.3333333

  pdf("fig_acfMA2lag5.pdf")
> plot(acf(MA2, lag.max=5))
```




```
# when set sd=2 we obtain different dispersion

MA2<-arima.sim(model=list(ma=c(1,1)),n=200,sd=2)
> sd(MA2)^2
[1] 12.65155
```

AR(p) model

$$X_t = \mu + \sum_{i=1}^{p} \varphi_i X_{t-i} + \epsilon_t$$

$\varphi_1, \ldots, \varphi_p$ are the parameters of the model,
μ is a constant (often omitted for simplicity) mostly called drift
ϵ_t is white noise (uncorrelated, gaussian)

constraints on the model to remain wide-sense stationary

- AR(1) model with $|\varphi_1| \geq 1$ is not stationary
- generally, for an AR(p) model to be wide-sense stationary, the roots of the polynomial

  $z^p - \sum_{i=1}^{p} \varphi_i z^{p-i}$

  must lie within the unit circle i.e.,

  each root z_i must satisfy $|z_i| < 1$

AR(1) process

$$X_t = \mu + \varphi_1 X_{t-1} + \epsilon_t$$

wide-sense stationary if

$$|\varphi_1| < 1$$

since it is obtained as the output of a stable filter whose input is the white noise. (If $\varphi_1 = 1$ then X_t has infinite variance, and is therefore not wide sense stationary.) Consequently, assuming $|\varphi_1| < 1$, the mean $\mathrm{E}(X_t)$ is identical for all values of t

Mean

$$\mathrm{E}(X_t) = \mathrm{E}(\mu) + \varphi_1 \mathrm{E}(X_{t-1}) + \mathrm{E}(\epsilon_t)$$

$$\mathrm{E}(X_t) = \frac{\mu}{1 - \varphi_1}\,.$$

Variance

$$\mathrm{var}(X_t) = \frac{\sigma_\epsilon^2}{1 - \varphi_1^2}$$

Autocovariance

$$\mathrm{E}(X_{t-\tau} X_t) = \frac{\sigma_\epsilon^2 \varphi^{|\tau|}}{1 - \varphi_1^2}$$

ar {stats} R Documentation (econometry)

Fit Autoregressive Models to Time Series

Description

Fit an autoregressive time series model to
    the data, by default selecting
    the complexity by AIC.

Usage

```
ar(x, aic = TRUE, order.max = NULL,
   method=c("yule-walker",
   "burg", "ols", "mle", "yw"),
   na.action, series, ...)
```

Auto-Regressive Integrated Moving Average, ARIMA

ARIMA(p, d, q): ARIMA models are, in theory, the most general class of models for forecasting a time series which can be stationarized by transformations such as differencing and logging.
A nonseasonal ARIMA model is classified as an ARIMA(p, d, q) model:

$$\left(1 - \sum_{j=1}^{p} \phi_j L^j\right) (1 - L)^d \, X_t = \left(1 + \sum_{j=1}^{q} b_j L^j\right) \epsilon_t$$

p is the number of autoregressive terms (AR)
d is the number of nonseasonal differences, deterministic trend component of the model
q is the number of lagged forecast errors in the prediction equation (MA)
ϵ_t are generally assumed to be independent, identically distributed (iid) variables sampled from a normal distribution with zero mean

Example 1: **Integration process**:

1. $p = 0, d = 0, q = 0$ ARIMA(0,0,0)
$(1 - L)^0 X_t = \epsilon_t$
$X_t = \epsilon_t$

2. $p = 0, d = 1, q = 0$ ARIMA(0,1,0)
$(1 - L)^1 X_t = \epsilon_t$
$X_t - X_{t-1} = \epsilon_t$
$X_t = X_{t-1} + \epsilon_t$ (integrated of 1-st order, random walk)

3. $p = 0, d = 2, q = 0$ ARIMA(0,2,0)
$(1 - L)^2 X_t = \epsilon_t$
$X_t - 2X_{t-1} + X_{t-2} = \epsilon_t$
$(X_t - X_{t-1}) - (X_{t-1} - X_{t-2}) = \epsilon_t$
(integrated of 2nd order)
(differences of first differences are stationary)

Example 2: for $p = 1, \underline{d = 1}, q = 1$,
ARIMA(1,1,1) to ARMA relations
$LX_t = X_{t-1}$, $L^2 X_t = X_{t-2}$

$$(1 - \phi_1 L)(1 - L)X_t = \epsilon_t + b_1 L\epsilon_t$$

$$(1 - L - \phi_1 L + \phi_1 L^2)X_t = \epsilon_t + b_1 L\epsilon_t$$

$$X_t - X_{t-1}(1 + \phi_1) + \phi_1 X_{t-2} = \epsilon_t + b_1\epsilon_{t-1}$$

$$X_t = X_{t-1}\underbrace{(1 + \phi_1)}_{a_1} + \underbrace{-\phi_1}_{a_2} X_{t-2} + \epsilon_t + b_1\epsilon_{t-1}$$

$$X_t = a_1 X_{t-1} + a_2 X_{t-2} + \epsilon_t + b_1\epsilon_{t-1}$$

is of ARMA$(2, 1)$ type

Auto regressive Integrated Moving Average (ARIMA)

Including Seasonality features.

Notation:

A very useful notation to describe the orders of various components in the multiplicative seasonal ARIMA model is given by

$$(p, d, q) \times (P, D, Q)$$

p order of autoregressive AR part of the ARIMA model at non-seasonal level

d order of differencing at non-seasonal level

q order of the moving average (MA) part of the ARIMA model at non-seasonal level

P order of autoregressive AR part of the ARIMA model at seasonal level

D order of differencing at seasonal level

Q order of the moving average (MA) part of the ARIMA model at seasonal level

ARIMA seasonal

```
arima(USAccDeaths, order = c(0,1,1),
      seasonal = list(order=c(0,1,1)))

Call:
arima(x = USAccDeaths, order = c(0, 1, 1),
          seasonal = list(order = c(0, 1, 1)))

Coefficients:
          ma1     sma1
      -0.4303  -0.5528
s.e.   0.1228   0.1784

sigma^2 estimated as 99347:
log likelihood = -425.44,  aic = 856.88
```

ARIMA $(p, d, q) \times (P, D, Q)_s$ Box Jenkins

$$\phi(L)\Phi(L)(1-L)^d(1-L^s)^D Y_t = \mu + \theta(L)\Theta(L)\epsilon_t$$

$$\phi(L) = 1 - \phi_1 L - \phi_2 L^2 - \ldots \phi_p L^p$$

AR(p) operator non-seasonal

$$\Phi(L) = 1 - \Phi_1 L^s - \Phi_2 L^{2s} - \ldots \Phi_P L^{sP}$$

AR(sP) operator seasonal

$$\theta(L) = 1 - \theta_1 L - \theta_2 L^2 - \ldots \theta_q L^q$$

MA(q) operator non-seasonal

$$\Theta(L) = 1 - \Theta_1 L^s - \Theta_2 L^{2s} - \ldots \Theta_Q L^{sQ}$$

MA(sQ) operator seasonal

```
ARIMA Modelling of Time Series

Description:
     Fit an ARIMA model to a univariate time series.

Usage:
     arima(x, order = c(0, 0, 0),
           seasonal = list(order = c(0, 0, 0), period = NA),
           xreg = NULL, include.mean = TRUE,
           transform.pars = TRUE,
           fixed = NULL, init = NULL,
           method = c("CSS-ML", "ML", "CSS"),
           n.cond, optim.method = "BFGS",
           optim.control = list(), kappa = 1e6)

Arguments:
       x: a univariate time series
   order: A specification of the non-seasonal part of the ARIMA model:
          the three components (p, d, q) are the AR order, the degree
          of differencing, and the MA order.

Examples  arima(lh, order = c(1,0,0))
          arima(USAccDeaths, order = c(0,1,1),
                              seasonal = list(order=c(0,1,1)))
```

Wold decomposition

define ARMA(p, q)

$$X_t = \mu + \epsilon_t + \sum_{i=1}^{p} \phi_i X_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j}$$

Reminder: MA(q)

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

Conversion of ARMA$(1, 0)$

$$X_t = \epsilon_t + 0.1 X_{t-1}$$

, $\mu = 0$
Begin: try to substitute MA(1) into ARMA$(1, 0)$

$$X_t \simeq \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2}$$

$X_{t-1} \simeq \epsilon_{t-1} + \theta_1 \epsilon_{t-2} + \theta_2 \epsilon_{t-3}$

$$\epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} \simeq \epsilon_t + 0.1(\epsilon_{t-1} + \theta_1\epsilon_{t-2} + \theta_2 \underbrace{\epsilon_{t-3}}_{exclude})$$

$$\theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} \simeq 0.1(\epsilon_{t-1} + \theta_1\epsilon_{t-2} + \theta_2\epsilon_{t-3})$$
$$\theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} \simeq 0.1(\epsilon_{t-1} + \theta_1\epsilon_{t-2})$$
$$\theta_1 = 0.1,\ \theta_2 = 0.1\theta_1 = 0.01$$
$$X_t \simeq \epsilon_t + 0.1\epsilon_{t-1} + 0.01\epsilon_{t-2}$$
$$X_t = \epsilon_t + 0.1\epsilon_{t-1} + 0.01\epsilon_{t-2} + \ldots$$

```
ARMAtoMA {stats} R Documentation

Convert ARMA Process to Infinite MA Process

Usage: ARMAtoMA(ar = numeric(), ma = numeric(), lag.max)


  Let:

  c(0.1)    ar coefficient phi_1

  0         zero mean
  40        terms approximation
  ARMAtoMA(ar=c(0.1),0,40)
```

```
  resulting MA coefficients:

  [1] 1e-01 1e-02 1e-03 1e-04 1e-05 1e-06 1e-07 1e-08 1e-09 1e-10 1e-11 1e-12
 [13] 1e-13 1e-14 1e-15 1e-16 1e-17 1e-18 1e-19 1e-20 1e-21 1e-22 1e-23 1e-24
 [25] 1e-25 1e-26 1e-27 1e-28 1e-29 1e-30 1e-31 1e-32 1e-33 1e-34 1e-35 1e-36
 [37] 1e-37 1e-38 1e-39 1e-40
```

Example of the transition from the stationary to the non-stationary behaviour

```
> ARMAtoMA(ar=c(0.9),0,40)
 [1] 0.90000000 0.81000000 0.72900000 0.65610000 0.59049000 0.53144100
 [7] 0.47829690 0.43046721 0.38742049 0.34867844 0.31381060 0.28242954
[13] 0.25418658 0.22876792 0.20589113 0.18530202 0.16677182 0.15009464
[19] 0.13508517 0.12157665 0.10941899 0.09847709 0.08862938 0.07976644
[25] 0.07178980 0.06461082 0.05814974 0.05233476 0.04710129 0.04239116
[31] 0.03815204 0.03433684 0.03090315 0.02781284 0.02503156 0.02252840
[37] 0.02027556 0.01824800 0.01642320 0.01478088
\end{verbatim
}

\vspace*{3mm}
```

approach to non-stationarity

{\scriptsize
\begin{verbatim}
> ARMAtoMA(ar=c(0.99),0,40)

 [1] 0.9900000 0.9801000 0.9702990 0.9605960 0.9509900 0.9414801 0.9320653
 [8] 0.9227447 0.9135172 0.9043821 0.8953383 0.8863849 0.8775210 0.8687458
[15] 0.8600584 0.8514578 0.8429432 0.8345138 0.8261686 0.8179069 0.8097279
[22] 0.8016306 0.7936143 0.7856781 0.7778214 0.7700431 0.7623427 0.7547193
[29] 0.7471721 0.7397004 0.7323034 0.7249803 0.7177305 0.7105532 0.7034477
[36] 0.6964132 0.6894491 0.6825546 0.6757290 0.6689718

> ARMAtoMA(ar=c(1),0,40)

 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
> 1 1
[39] 1 1
> ARMAtoMA(ar=c(1.1),0,40)
 [1] 1.100000 1.210000 1.331000 1.464100 1.610510 1.771561 1.948717
 [8] 2.143589 2.357948 2.593742 2.853117 3.138428 3.452271 3.797498
[15] 4.177248 4.594973 5.054470 5.559917 6.115909 6.727500 7.400250
[22] 8.140275 8.954302 9.849733 10.834706 11.918177 13.109994 14.420994
[29] 15.863093 17.449402 19.194342 21.113777 23.225154 25.547670 28.102437
[36] 30.912681 34.003949 37.404343 41.144778 45.259256