ARIMA-Model-Based Decomposition of Time Series. The `tsdecomp` R Package

Javier López-de-Lacalle
https://jalobe.com

DRAFT VERSION: January 28, 2017
PACKAGE VERSION: 0.3 (devel)

Abstract

Given an autoregressive integrated moving average (ARIMA) model fitted to an observed time series, the `tsdecomp` R package extracts a signal for the trend and seasonal components. The package is mainly intended for annual, quarterly and monthly time series.

Key words: ARIMA, R, seasonal adjustment, signal extraction, time series.

1 Introduction

This document introduces the `tsdecomp` package of R (R Development Core Team, 2016), which implements the methodology developed and described, among others, in Box *et al.* (1978), Burman (1980), Hillmer and Tiao (1982) and Maravall and Pierce (1987). An extensive review of the methodology is given in Planas (1997) and Gómez and Maravall (2001b).

The procedure relies on the decomposition of the pseudo-spectrum of an ARIMA model fitted to an observed time series. The methodology has long been studied in the literature and is used by many statistical offices for the compilation of seasonally adjusted data. The software `TRAMO` and `SEATS` (Gómez and Maravall, 2001a) is the reference tool for the methodology described in the references above. For the development of `tsdecomp`, the code of the `SSMMATLAB` package (Gómez, 2015) has been illuminating for the understanding of some points of the methodology.

The development of the package `tsdecomp` began as a pedagogical exercise. The current version has become relatively mature and provides interfaces that can be used by users not necessarily familiar with the methodology. The package has been tested and developed for annual, quarterly and monthly time series.

The remaining of this document is organised as follows. Notation and some preliminary concepts are given in Section 2 The methodology and implementation is introduced in Section 3. Some examples are shown in Section 4.

2 Notation and concepts

2.1 ARIMA process

An **autoregressive moving-average process** (ARMA), x_t, can be defined as follows:

$$x_t = \sum_{i=1}^{p} \phi_i x_{t-i} + \sum_{i=0}^{q} \theta_i \varepsilon_{t-i}\,, \qquad \varepsilon_t \sim \mathrm{NID}(0, \sigma^2)\,,$$

for $t = 1, 2, \ldots, T$ and θ_0 is normalized to unity, $\theta_0 = 1$; p and q are respectively the orders of the AR and MA parts of the model. The AR and MA structures can be conveniently represented as a polynomial in the lag operator:

$$(1 - \phi_1 L - \cdots - \phi_p L^p) x_t = (1 + \theta_1 L + \cdots + \theta_q L^q)\varepsilon_t\,,$$
$$\phi(L) x_t = \theta(L)\varepsilon_t\,,$$

where L is the lag operator such that $L^i x_t = x_{t-i}$.

It is sometimes helpful to represent an ARMA process as a moving average of infinite order (this requires the roots of $\phi(L)$ to lie outside the unit circle):

$$x_t = \frac{\theta(L)}{\phi(L)}\varepsilon_t = \psi(L)\varepsilon_t = \sum_{i=0}^{\infty} \psi_i \varepsilon_{t-i}\,. \tag{1}$$

The coefficients ψ_i can be obtained by rewriting the above as $\psi(L)\phi(L) = \theta(L)$. Then, multiplying the polynomials in the left-hand-side (LHS) and equating the coefficients of the same powers L^i in both sides of the expression yields a system than can be solved recursively.

If the roots of $\phi(L)$ lie outside the unit circle (the inverse roots inside the unit circle), then process is stationary, which roughly involves that the mean and variance remain constant over time. The class of ARMA models can be extended to non-stationary data by means of ARIMA models. This is the case when the data can be rendered stationary by taking differences to the data (typically once or twice). We denote the regular differencing operator as $(1 - L) \equiv \nabla$.

We will also consider seasonal ARIMA models, which consist of a regular ARIMA structure times another ARIMA structure of seasonal order. We will denote the polynomials related to the seasonal part of the model in capital letters. For a series of period S (e.g., $S = 4$ in quarterly data), a seasonal AR(2) process is represented by: $\Phi(L)x_t = x_t - \Phi_1 x_{t-S} - \Phi_1 x_{t-2\times S}$. Similarly, the seasonal differencing operator is defined as $(1 - L^S) \equiv \nabla_S$. Thus, the general seasonal ARIMA(p,d,q)(P,D,Q) model is given by:

$$\phi(L)\Phi(L)\nabla^d x_t \nabla_S^D x_t = \theta(L)\Theta(L)\varepsilon_t\,.$$

2.2 Theoretical functions of an ARMA process

The **autocovariance function** (ACF) of an ARMA process is given by:

$$\gamma_\tau = \mathrm{Cov}(x_t, x_{t-\tau}) = E\left[\sum_{i=0}^{\infty} \psi_i \varepsilon_{t-i} \sum_{i=0}^{\infty} \psi_i \varepsilon_{t-\tau-i}\right] = \sigma^2 \sum_{i=0}^{\infty} \psi_i \psi_{i+\tau}\,. \tag{2}$$

(The last part can be checked by making taking into account that ε_t are independently distributed and, hence, $E(\varepsilon_i \varepsilon_j) = 0$ for $i \neq j$ and $E(\varepsilon_i \varepsilon_j) = \sigma^2$ if $i = j$.) As the above implies an infinite

sum, the following approach can be followed to compute it: multiply both sides of the equation $\phi(L)x_t = \theta(L)\varepsilon_t$ by x_{t-k} and take expectations; this will give a system of equations for the first $p+1$ autocovariances; the remaining autocovariances can then be obtained recursively.

The **autocovariance generating function** (ACGF) is defined as a power series whose coefficients are the autocovariances:

$$\gamma(L) = \sum_{\tau=-\infty}^{\infty} \gamma_\tau L^\tau \,.$$

Substituting the expression of γ_τ given in equation (2), the ACGF can be expressed as:

$$\begin{aligned}\gamma(L) &= \sigma^2 \sum_{\tau=-\infty}^{\infty} \sum_{i=0}^{\infty} \psi_i \psi_{i+\tau} L^\tau = \sigma^2 \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \psi_i \psi_j L^{j-i} \\ &= \sigma^2 \sum_{i=0}^{\infty} \psi_i L^{-i} \sum_{i=0}^{\infty} \psi_i L^i = \sigma^2 \psi(L^{-1})\psi(L)\,.\end{aligned}$$

By inspection of equation (1), it can be deduced that the ACGF can be expressed as:

$$\gamma(L) = \sigma^2 \frac{\theta(L)\theta(L^{-1})}{\phi(L)\phi(L^{-1})}\,. \tag{3}$$

The ACGF is sometimes introduced as the z-transform of the autocovariance function:

$$\gamma(z) = \gamma_0 + \gamma_1(z + z^{-1}) + \gamma_2(z^2 + z^{-2}) + \cdots = \sum_{\tau=-\infty}^{\infty} \gamma_\tau z^\tau\,, \tag{4}$$

$$\gamma(z) = \sigma^2 \frac{\theta(z)\theta(z^{-1})}{\phi(z)\phi(z^{-1})} = \sigma^2 \psi(z)\psi(z^{-1})\,,$$

where z is a complex variable.

The **spectral density** is defined by:

$$f(\omega) = \frac{\sigma^2}{2\pi} \frac{\theta(e^{-i\omega})\theta(e^{i\omega})}{\phi(e^{-i\omega})\phi(e^{i\omega})} = \frac{\sigma^2}{2\pi} \frac{|\theta(e^{-i\omega})|^2}{|\phi(e^{-i\omega})|^2} = \frac{\sigma^2}{2\pi} \frac{|1 + \sum_{j=1}^{q} \theta_j e^{-ij\omega}|^2}{|1 - \sum_{j=1}^{p} \phi_j e^{-ij\omega}|^2}\,,$$

with $\omega \in [0, 2\pi]$.

Replacing z by $e^{-i\omega}$ in the expression of the ACGF given in equation (4) yields the spectral density multiplied by 2π. In fact, the spectral density is the frequency domain representation of the autocovariances. In particular, the spectral density is the Fourier transform of the autocovariance function, and vice versa.

$$f(\omega) = \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} \gamma_\tau e^{-i\omega\tau}\,. \tag{5}$$

The decomposition pursued in the procedure relies on the partial fraction expansion of the spectrum given in equation (4) with $z = e^{-i\omega}$. Note that the variable in this equation is $z^j + z^{-j} = \cos(\omega j) + i\sin(\omega j) + \cos(\omega j) - i\sin(\omega j) = 2\cos(\omega j)$ (for $z = e^{-iw}$, which has unit modulus, the inverse $1/z$ is the complex-conjugate of z). Thus, the expression given in equation (4) is not a polynomial since the variable changes with the order of each element, j. As a consequence, the standard techniques for partial fraction expansion cannot be applied. As we shall see, a change of variable can be adopted so that the expression

$$A(2\cos(\omega j)) = a_0 + a_1 2\cos(\omega) + a_2 2\cos(2\omega) + \cdots + a_n 2\cos(n\omega)$$

can be transformed into the polynomial:

$$\begin{aligned} B(2\cos(\omega)) &= b_0 + b_1 2\cos(\omega) + b_2(2\cos(\omega))^2 + \cdots + b_n(2\cos(\omega))^n \\ &= b_0 + b_1 x + b_2 x^2 + \cdots + b_n x^n\,, \quad \text{with } x = 2\cos(\omega)\,, \end{aligned}$$

to be decomposed in partial fractions.

2.3 Change of variable in the pseudo-spectrum

Upon the structure derived in Appendix A, the code below builds the matrix that gives the mapping from the coefficients of the ACGF for the variable $z = e^{-i\omega}$ to a polynomial in the variable $2\cos\omega$, for a given order `n`. Note that each non-zero diagonal is the cumulative sum of the precedent non-zero diagonal in absolute value, alternating negative and positive signs in each diagonal.

```
n <- 50
m <- diag(1, n, n)
n2 <- n - 2
j <- -1
tmp <- as.numeric(seq.int(2, n-1))
for (i in seq.int(3, n-2, 2))
{
  id <- cbind(seq_len(n2), seq.int(i,n))
  m[id] <- j * tmp
  n2 <- n2 - 2
  j <- -1 * j
  tmp <- cumsum(tmp[seq_len(n2)])
}
if (n%%2 == 0) {
  id <- cbind(seq_len(n2), seq.int(n-1,n))
  m[id] <- j * tmp
} else
  m[1,n] <- j * tmp
mat.acgf2poly <- m
mat.acgf2poly[1:10,1:10]
```

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    1    0   -2    0    2    0   -2    0    2     0
 [2,]    0    1    0   -3    0    5    0   -7    0     9
 [3,]    0    0    1    0   -4    0    9    0  -16     0
 [4,]    0    0    0    1    0   -5    0   14    0   -30
 [5,]    0    0    0    0    1    0   -6    0   20     0
 [6,]    0    0    0    0    0    1    0   -7    0    27
 [7,]    0    0    0    0    0    0    1    0   -8     0
 [8,]    0    0    0    0    0    0    0    1    0    -9
 [9,]    0    0    0    0    0    0    0    0    1     0
[10,]    0    0    0    0    0    0    0    0    0     1
```

There are other alternatives to implement this change of variable. The appeal of this approach is that we can create a matrix of a relatively large order, say 50, and store it as internal data in the package, so that it does not need to be built each time it is required; also, the matrix product is a relatively quick operation. In addition, the transformation can be undone easily by means of the inverse of this matrix (we will use it below).

In order to understand the equivalence of equation (4) for the the variables $z = e^{-i\omega}$ and $x = 2\cos\omega$, it may be helpful to fiddle around with the equations and concepts introduced above. Let's take the following ARMA(2,1) model:

$$x_t - 0.8x_{t-1} + 0.6x_{t-2} = 0.4\varepsilon_{t-1} + \varepsilon_t\,, \quad \text{with}\,\sigma_\varepsilon^2 = 1\,.$$

It can be checked that complex-conjugate roots of this AR polynomial generates cycles of duration around 6 periods. The spectral density has a peak at frequency $\omega = 1.028$.

```
r <- polyroot(c(1,-0.8,0.6))
cbind(roots=r, w=Arg(r), period=2*pi/Arg(r))
                    roots            w       period
[1,] 0.666667+1.105542i  1.028157+0i  6.111113+0i
[2,] 0.666667-1.105542i -1.028157+0i -6.111113+0i
```

Below, the theoretical autocovariances of the ARMA model are computed and stored in `gamma`. For simplicity, they are computed based on the ψ_i coefficients of the MA representation of the ARMA process given in equation (1). The spectral density is then computed by replacing $z = e^{-i\omega}$ in the ACGF defined in equation (4). Note that, for this z of unit modulus, $z^j + z^{-j} = \cos(\omega j) + i\sin(\omega j) + \cos(\omega j) - i\sin(\omega j)$ collapses to $2\cos(\omega j)$. The result is rescaled by the factor $1/2\pi$.

```
psi <- c(1, ARMAtoMA(ar=c(0.8,-0.6), ma=0.4, lag.max=49))
gamma <- c(sum(psi^2), rep(0, length(psi)-1))
for (tau in seq_along(gamma[-1]))
  gamma[tau+1] <- sum(psi[seq_len(length(psi)-tau)] * psi[-seq_len(tau)])
w <- seq(0, pi, len=length(gamma))
spec1 <- rep(gamma[1], length(w))
for (i in seq_along(w))
{
  for (j in seq_len(length(gamma)-1))
  {
    z <- 2*cos(w[i]*j)
    spec1[i] <- spec1[i] + gamma[j+1] * z
  }
}
spec1 <- spec1/(2*pi)
```

Displaying the spectral density stored in `spec1`

```
plot(w, spec1)
```

shows a curve with a peak at around `w=1.028`, as mentioned above. Alternatively, the spectral density can be computed as the the above matches the Fourier transform of the autocovariances,

gamma, as defined in equation (5):

```
spec2 <- rep(NA, length(spec1))
for (i in seq_along(w))
{
  tmp <- c(gamma * exp(-1i * w[i] * seq.int(0, length(gamma)-1)))
  spec2[i] = (tmp[1] + sum(tmp[-1] + Conj(tmp[-1]))) / (2*pi)
}
print(all(Im(spec2) == 0))
[1] TRUE
print(all.equal(spec1, Re(spec2)))
[1] TRUE
```

The polynomial in equation (4) is transformed according to the change of variable explained above by premultuplying the vector of autocovariances, gamma, by the transformation matrix mat.acgf2poly created before. We can check that, up to a rounding error, these new coefficients yield the same values of the spectrum for the new variable (2*cos(w[i])). In this way, we will be able to apply a partial fraction expasion on this polynomial.

```
newcoefs <- mat.acgf2poly %*% gamma
spec3 <- rep(newcoefs[1], length(w))
for (i in seq_along(w))
{
  for (j in seq_along(newcoefs[-1]))
  {
    x <- (2*cos(w[i]))^j
    spec3[i] <- spec3[i] + newcoefs[j+1] * x
  }
}
spec3 <- spec3/(2*pi)
all.equal(spec1, spec3)
[1] "Mean relative difference: 7.027414e-05"
```

3 ARIMA-model-based decomposition

This section introduces the implementation of each step in the ARIMA-model-based decomposition method. The methodology has been broadly described and developed in the references cited in the introduction above and others. Here, I will focus on showing the implementation. Theoretical concepts required for the understanding of the overall procedure will be introduced, emphasising those points that I found more critical for the understanding and implementation of the methodology.

The bottom-line idea is to fit an ARIMA model and assume that the unobserved components (i.e., trend-cycle and seasonal components) follow an ARIMA model. A partial fraction decomposition is performed on the frequency domain of the ACGF, upon the know coefficients of the fitted

model and the roots allocated to each component.

Let x_t be a series consists of the sum of a trend, T_t, and a seasonal component, S_t, as well as some noise $e_t \sim \text{NID}(0, \sigma_e^2)$:

$$x_t = T_t + S_t + e_t \,.$$

The methodology assumes that the unobserved components (trend, seasonal) follow an ARIMA model:

$$\begin{aligned} \phi_T(L)T_t &= \theta_T(L)a_{1,t} \,, \quad a_{1,t} \sim \text{NID}(0, \sigma_1^2) \\ \phi_S(L)S_t &= \theta_S(L)a_{2,t} \,, \quad a_{2,t} \sim \text{NID}(0, \sigma_2^2) \,. \end{aligned} \tag{6}$$

An ARIMA model is chosen and fitted to the observed data x_t. The coefficients of the models for the components are obtained from the relationship:

$$\sigma_a^2 \frac{\theta(B)\theta(F)}{\phi(B)\phi(F)} = \sigma_1^2 \frac{\theta_T(B)\theta_T(F)}{\phi_T(B)\phi_T(F)} + \sigma_2^2 \frac{\theta_S(B)\theta_S(F)}{\phi_S(B)\phi_S(F)} + \sigma_e^2 \,,$$

where B is the backshift operator (the same as the lag operator L) and F is the forward operator $F = B^{-1}$. The polynomials in the left-hand-side are known, they come from the fitted model. The polynomials in the right-hand-side are unknown and are determined as follows: the AR polynomials (denominators) are obtained upon the roots of the AR polynomial in the fitted model that are allocated to each component; the MA polynomials (numerators) are then obtained by means of partial fractial expansion. This is the overall strategy of the procedure. Next, we will see each step in more detail.

3.1 Fit an ARIMA model to the observed data

Let's start by generating some data and fitting an ARIMA(0,0,1)(1,0,0) model to it:

```
set.seed(125)
y <- arima.sim(n=200, model=list(ar=c(0,0,0,0.8), ma=0.5))
y <- ts(round(y, 2), frequency=4)
fit <- arima(y, order=c(0,0,1), seasonal=list(order=c(1,0,0)),
  include.mean=FALSE)
c(coef(fit), sigma2 = fit$sigma2)
      ma1      sar1    sigma2
0.5475396 0.8567436 1.0395498
```

Fitted model:

$$(1 - 0.86L^4)y_t = (1 + 0.55L)\varepsilon_t \,, \quad \sigma_\varepsilon^2 = 1.04 \,.$$

3.2 Allocate the roots of the AR polynomial

The first step is to factorize the AR polynomial of the fitted model and allocate the roots to the trend and cycle components. We know that an AR polynomial generates cycles where the period is determined by the angular frequencies of the roots of the polynomial.

Table 1 summarizes long run and seasonal cycles in quarterly and monthly of the seasonal differencing operators, which are factorised as follows: $(1 - L^4) = (1 - L)(1 + L^2)(1 + L)$ and

$(1-L^S) = (1-L)(1-\sqrt{3}L+L^2)(1-L+L^2)(1+L^2)(1+L+L^2)(1+\sqrt{3}L+L^2)(1+L)$. The modulus of these roots is equal to unity. In this case the corresponding cycle is shaped as an accumulation of shocks, ε_t, rendering the cycle non-stationary. When the inverse of the root lies inside the unit circle, then the effect of the shocks have a transient effect. Explosive patterns are generated by roots with modulus larger than unity (inverse root's modulus lower than unity), but this is not generally observed in the kind of series intended to be analysed with the package (macroeconomic indicators, climatological data,...).

Table 1: Long run and seasonal unit roots

Frecuency		Period	Cycles/ year	Root	Filter
Monthly series					
0	Long run	∞	0	1	$(1-L)$
$\frac{\pi}{6}, \frac{11\pi}{6}$	Annual	$12; 1.09$	$1; 11$	$\frac{1}{2}\left(\sqrt{3} \pm i\right)$	$(1-\sqrt{3}L+L^2)$
$\frac{\pi}{3}, \frac{5\pi}{3}$	Biannual	$6; 1.2$	$2; 10$	$\frac{1}{2}\left(1 \pm \sqrt{3}i\right)$	$(1-L+L^2)$
$\frac{\pi}{2}, \frac{3\pi}{2}$		$4; \frac{4}{3}$	$3; 9$	$\pm i$	$(1+L^2)$
$\frac{2\pi}{3}, \frac{4\pi}{3}$	Quarterly	$3; 1.5$	$4; 8$	$-\frac{1}{2}\left(1 \pm \sqrt{3}i\right)$	$(1+L+L^2)$
$\frac{5\pi}{6}, \frac{7\pi}{6}$		$2, 4; 1.7$	$5; 7$	$-\frac{1}{2}\left(\sqrt{3} \pm i\right)$	$(1+\sqrt{3}L+L^2)$
π	Bimonthly	2	6	-1	$(1+L)$
Quarterly series					
0	Long run	∞	0	1	$(1-L)$
$\frac{\pi}{2}, \frac{3\pi}{2}$	Annual	$4; \frac{4}{3}$	$1; 3$	$\pm i$	$(1+L^2)$
π	Biannual	2	2	-1	$(1+L)$

The factorisation of the AR polynomial in the model fitted above, as well as the a summary of the corresponding inverse roots is returned by `roots.allocation`:

```
p <- roots.allocation(fit)
print(p, units="pi")
Roots of AR polynomial
----------------------
(1 - 0.86L^4) = (1 - 0.962L)(1 + 0.962L + 0.926L^2 + 0.891L^3)

  Component            Root Modulus Argument Period Cycles.per.Year
1     trend  0.9621-0.0000i  0.9621        0    Inf               0
2  seasonal  0.0000+0.9621i  0.9621     pi/2  4.000               1
3  seasonal -0.9621+0.0000i  0.9621       pi  2.000               2
4  seasonal  0.0000-0.9621i  0.9621    3pi/2  1.333               3
Warning messages:
1: In polynomial(p) : imaginary parts discarded in coercion
2: In polynomial(p) : imaginary parts discarded in coercion
```

The roots can be obtained by means of `polyroot`. Then the angular frequency, ω, is returned by `Arg`, the period is given by $\frac{2\pi}{\omega}$ and the number of cycles completed in a year is the number of

seasons (in the example $S = 4$) divided by the period. In order to reproduce above, we first get the roots `r <- polyroot(c(1,0,0,0,-coef(fit)[2]))` (the roots reported above are the inverse roots), then the angular frequencies are returned by `Arg(r)`.

Upon this information `roots.allocation` allocates each root to each component; then, the coefficients of the AR polynomial is built for each component based on the roots allocated to each of them. For example, the polynomial related to the seasonal roots can be obtained as folows:

```
roots2poly(1/p[["roots.stationary"]][["seasonal"]])
```

The AR polynomials for each component are returned by `roots.allocation`:

```
print(polystring(p$trend, ndec=4))
[1] "1 - 0.9621x"
print(polystring(p$seasonal, ndec=4))
[1] "1 + 0.9621x + 0.9256x^2 + 0.8905x^3"
```

3.3 Pseudo-spectrum

The building block of the procedure is the following relationship (partial fraction decomposition of the pseudo-spectrum):

$$\sigma^2 \frac{\theta(B)\theta(F)}{\phi(B)\phi(F)} = \sigma_a^2 \frac{\theta_T(B)\theta_T(F)}{\phi_T(B)\phi_T(F)} + \sigma_b^2 \frac{\theta_S(B)\theta_S(F)}{\phi_S(B)\phi_S(F)} + \sigma_e^2 \,, \tag{7}$$

The polynomials in the left-hand-side are known (they are the estimates of the AR and MA parts of the model fitted to the observed data) and σ^2 is the estimated variance of the residuals of the fitted model. The AR polynomials in the right-hand-side, $\phi(L)$, were determined according to the allocation of roots explained above in Section 3.2. The MA polynomials, $\theta(L)$, as well as the variances of the models for each component remain unknown at this point. After expressing the known polynomials in terms of the variable $x = 2\cos\omega$ as explained in Section 2.3, the MA polynomials can be obtained from the relationship (7).

We will proceed as follows. First, the product of the known polynomials in the backshift and forward operators are obtained and are transformed in terms of the variable discussed before. Then, the orders of the polynomials in the numerators of the right-hand-side are determined; the coefficients related to the variables of the same order from both sides of the relationship are equated. This will yield a system of equations. Solving the system gives the coefficients of the numerators of the pseudo-spectrum $\theta(B)\theta(F)$ in terms of the transformed variable.

In Section 3.2, the AR polynomials for the trend and seasonal components ($\phi_T(B)$ and $\phi_S(B)$) were stored in `p$trend` and `p$seasonal`, respectively. As the roots of the polynomial $\phi(B^{-1}) \equiv \phi(F)$ are the inverse of the roots of the polynomial $\phi(B)$, we could get the coefficients of the polynomial $\phi(F)$ from the inverse roots of $\phi(B)$ A less demanding approach is to multiply the $\phi(B)$ by the itself but with the coefficients in reverse order. This is what `convolve(type="open")` does. The result is a symmetric polynomial; we need only one side of it (that's why the first elements are removed). After extracting the MA coefficients from the fitted model, we do this operation for it and for the AR polynomials of the components.

```
ma.total <- c(1, fit$model$theta[seq_len(fit$arma[2]+fit$arma[4]*fit$arma[5])])
tmp <- convolve(ma.total, ma.total, type="open")
ma.total.bf <- tmp[-seq_along(ma.total[-1])]
tmp <- convolve(p$trend, p$trend, type="open")
den.trend.bf <- tmp[-seq_along(p$trend[-1])]
tmp <- convolve(p$seasonal, p$seasonal, type="open")
den.seas.bf <- tmp[-seq_along(p$seasonal[-1])]
```

The polynomial $\theta(B)\theta(F)$ of the left-hand-side is stored in `ma.total.bf`; $\phi_T(B)\phi_T(F)$ and $\phi_S(B)\phi_S(F)$ in `den.trend.bf` and `den.seas.bf`, respectively for each component.

The transformation discussed above is now performed for each one of these elements so that we have the coefficients of a polynomial liable to be decomposed into partial fractions. Here, instead of the multiplication by the matrix `mat.acgf2poly` created before, we use the interface `acgf2poly`.

```
num.psp.total <- acgf2poly(ma.total.bf)
den.psp.trend <- acgf2poly(den.trend.bf)
den.psp.seas <- acgf2poly(den.seas.bf)
print(num.psp.total)
[1] 1.2997996 0.5475396
print(den.psp.trend)
[1]  1.9256045 -0.9620834
print(den.psp.seas)
[1] 0.010657631 0.005324837 1.782348127 0.890508707
```

As we shall see, the denominator of the left-hand-side will not be needed, but we can compute it for illustration and debuggin of the code. It can computed doing the same steps as for the other elements (i.e., `convolve` and `acgf2poly`) or simply as the product of the denominators in the right-hand-side.

```
ar.total <- c(1, -fit$model$phi)
tmp <- convolve(ar.total, ar.total, type="open")
ar.total.bf <- tmp[-seq_along(ar.total[-1])]
den.psp.total <- acgf2poly(ar.total.bf)
all.equal(den.psp.total, polyprod(den.psp.trend, den.psp.seas))
[1] TRUE
```

3.4 Partial fraction decomposition

At this point, we have all the necessary elements to carry out the partial fraction decomposition: `num.psp.total` contains the coefficients of $\theta(B)\theta(F)$; `den.psp.trend` those of $\phi_T(B)\phi_T(F)$ and

$\phi_S(B)\phi_S(F)$ is stored in `den.psp.seas`. The relationship in equation (7):

$$\frac{1.2998 + 0.5475x}{0.0205 - 0x + 3.427x^2 - 0x^3 - 0.8567x^4} =$$

$$\frac{A(x)}{1.9256 - 0.9621x} + \frac{B(x)}{0.0107 + 0.0053x + 1.7823x^2 + 0.8905x^3}$$

Now we set $A(x) = a_0$ and $B(x) = b_0 + b_1x + b_2x^2$ (these are the highest order polynomials that we will be able to solve, i.e., same number of equations and variables). Multiplying the denominator of the left-hand-side by each term in the right-hand-side yields:

$$1.2998 + 0.5475x =$$

$$(0.0107 + 0.0053x + 1.7823x^2 + 0.8905x^3)A(x) + (1.9256 - 0.9621x)B(x)$$

Note that as the denominator of the LHS is the product of the denominators in the RHS, the former vanishes and the actual coefficients are not actually needed. Equating the coefficients related to elements of the same order in both sides of the equation yields the following system of equations:

```
lhs <- cbind(den.psp.seas,
  matrix(c(rep(c(den.psp.trend, rep(0, 3)), 2), den.psp.trend), nrow=4))
rhs <- c(num.psp.total, rep(0, nrow(lhs)-length(num.psp.total)))
res <- solve(lhs, rhs)
num.psp.trend <- res[1]
num.psp.seas <- res[2:4]
as.vector(res)
[1] 0.1675147 0.6740815 0.6206727 0.1550523
```

$$\begin{bmatrix} 0.0107 & 1.9256 & 0 & 0 \\ 0.0053 & -0.9621 & 1.9256 & 0 \\ 1.7823 & 0 & -0.9621 & 1.9256 \\ 0.8905 & 0 & 0 & -0.9621 \end{bmatrix} \times \begin{bmatrix} a_0 \\ b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 1.2998 \\ 0.5475 \\ 0 \\ 0 \end{bmatrix}$$

Solving the system gives the coefficients of $A(x)$ and $B(x)$.

$$A(x) = 0.1675\,,$$
$$B(x) = 0.6741 + 0.6207x + 0.1551x^2\,.$$

The coefficients of these polynomials are in terms of the variable $x = 2\cos\omega$, therefore, in order to get the coefficients of the corresponding MA polynomials we will need to undo the transformation.

3.5 Canonical decomposition

An issue remains to be addressed. How to allocate the the variance across the components? Actually, the relationship solved above can be met for several definitions of the components. Given an admissible decomposition, we can get another valid one by simply removing constant fraction of one of the components and assigning it to another component. The solution adopted in practice

is to choose the so-called canonical decomposition. This solution defines the components (except for the irregular) clean of white noise, i.e., it is not possible to extract a white noise signal (a fraction of the variance) and assign it to other component. Despite being motivated by the need to choose a single decomposition among all the possible ones, this solution seems sensible from a practical point of view, since it implies that the estimated trend and seasonal components will not be contaminated by noise.

In other words, the canonical decomposition consists in maximising the variance of the irregular subject to the polynomials that we have obtained so far. To do so, the terms $\frac{\theta(B)\theta(F)}{\phi(B)\phi(F)}$ related to each component are in the range $[-\pi, \pi]$; and the minimum value within this range is from the corresponding component and assigned to the variance of the irregular component. This is done in the code below. Remember that we are still working with the polynomials in terms of the variable $2\cos\omega$, therefore the range $[-\pi, \pi]$ becomes $[-2, 2]$. The miminum is found by means of `optimize`, the relevant end points for each component are evaluated separately.

```
fobj <- function(x, p1, p2) polyeval(p1, x) / polyeval(p2, x)
res <- optimize(f=fobj, interval=c(-2, 2), p1=num.psp.trend, p2=den.psp.trend)
trend.minval <- min(res$obj, fobj(-2, num.psp.trend, den.psp.trend))
num.trend <- c(num.psp.trend, 0) - den.psp.trend * trend.minval
res <- optimize(f=fobj, interval=c(-2, 2), p1=num.psp.seas, p2=den.psp.seas)
seas.minval <- min(res$obj, fobj(2, num.psp.seas, den.psp.seas))
num.seas <- c(num.psp.seas, 0) - den.psp.seas * seas.minval
sigma2.irregular <- trend.minval + seas.minval
print(polystring(num.trend))
[1] "0.08 + 0.04x"
print(polystring(num.seas))
[1] "0.67 + 0.62x - 0x^2 - 0.08x^3"
print(sigma2.irregular)
[1] 0.1278762
```

3.6 Recovery of MA coefficients

The vector `num.trend` and `num.seas` contain the coefficients of the numerators $\theta_T(B)\theta_T(F)$ and $\theta_S(B)\theta_S(F)$ in terms of the variable $2\cos\omega$. From these vectors we can get the coefficients of the MA polynomials and the variances of the ARIMA models for each component. A possible approach is to first undo the change of variable (premultiplying by the inverse of `mat.acgf2poly`); this will give as the coefficients of the ACGF of the MA process, i.e., the autocovariances. Then, we can apply an algorithm that maps the autocovariances to the coefficients and variance of a MA process. If we were dealing with an AR process, this algorithm would be the well-known Yule-Walker equations. For a MA process, the algorithm is not so straightforward. The functions `acov2ma.init` and `acov2ma` in the package tsdecomp implement the algorithms described in (Pollock, 1999, Chapter 17).

```
ma.trend.acovs <- solve(mat.acgf2poly[1:2,1:2]) %*% num.trend
tmp <- acov2ma(x=ma.trend.acovs, init=acov2ma.init(ma.trend.acovs)$coef)
```

```
ma.trend.coefs <- tmp$coef
sigma2.trend <- tmp$sigma2
ma.seas.acovs <- solve(mat.acgf2poly[1:4,1:4]) %*% num.seas
tmp <- acov2ma(x=ma.seas.acovs, init=acov2ma.init(ma.seas.acovs)$coef)
ma.seas.coefs <- tmp$coef
sigma2.seas <- tmp$sigma2
print(polystring(ma.trend.coefs))
[1] "1 + x"
print(polystring(ma.seas.coefs))
[1] "1 + 1.31x + 0.46x^2 - 0.33x^3"
c(irregular = sigma2.irregular, trend = sigma2.trend, seas = sigma2.seas)
 irregular      trend       seas
0.12787616 0.04186303 0.22584534
```

In some cases, I observed that the algorithm `acov2ma` strugges to reach a stable solution. It may be due to the presence of several roots of modulus greater than unity. Inspecting the code of the function `pu2ma` in the package `SSMMATLAB` (Gómez, 2015), I found an alternative procedure to get the coefficients of the MA polynomial from the partial fraction in the variable $2\cos\omega$. This is how I understand this alternative procedure: first, the roots of `num.seas` are obtained; `num.seas` is one side of the numerator of the symmetric polynomial $\theta_S(B)\theta_S(F)$. We can get the roots of `num.seas` as usual by means of `polyroot`. The roots of $\theta_S(B)$ are required, nonetheless. As the roots of $\theta_S(B)$ are the inverse of the roots of $\theta_S(B)$, we know that a root λ of the symmetric polynomial $\theta_S(B)\theta_S(F)$ will be of the form $(x-\delta)(x-1/\delta) = 1 - ((\delta^2+1)/\delta)x + x^2$. Given the roots $\lambda = (\delta^2+1)/\delta$ of $\theta_S(B)\theta_S(F)$, we can get the roots δ of $\theta_S(B)$ by solving the quadratic equation $ax^2 + bx + c$ with $a = c = 1$ and $b = \lambda$.

```
r <- polyroot(num.seas)
tmp <- sqrt(r^2 - 4)
#r <- c((r + tmp)/2, (r - tmp)/2)
r <- (r + tmp)/2
roots2poly(r)
[1]  1.0000000  1.3055754  0.4550511 -0.3326465
```

Summing up, the models for the components are:

$$\begin{aligned}
\text{Trend:} \quad & (1 - 0.9621L)T_t = (1 + L)a_t\,, \quad \sigma_a^2 = 0.0419\,. \\
\text{Seasonal:} \quad & (1 + 0.9621L + 0.9256L^2 + 0.8905L^3)S_t = \\
& (1 + 1.3056L + 0.455L^2 - 0.3326L^3)b_t\,, \quad \sigma_b^2 = 0.2258\,.
\end{aligned}$$

3.7 Filtering

After having found the models for the components, we are now interested on an estimate of those components. A possible approach is to put the models in state-space form and run the Kalman filter and smoother. Alternatively, we can use the following result.

The minimum mean squared error estimator of the component s_t –in the observed series x_t– is given by the following filtering operation:

$$\begin{aligned} \mathrm{E}(s_t|X) \equiv \hat{s}_t &= \left(v_0 + \sum_{i=1}^{\infty} v_i (B^i + F^i) \right) x_t \\ &= v_0 + v_1(x_{t-1} + x_{t+1}) + v_2(x_{t-2} + x_{t+2}) + \cdots \\ &= v(B,F)x_t \,, \end{aligned} \tag{8}$$

where $X = \{x_{-\infty}, \ldots, x_t, \ldots, x_{\infty}\}$ and $v(B,F)$ is the Wiener-Kolmogorov filter, a centered and symmetric whose weights v_i are given by the ratio of the MA representation of the models for the signal and for the observed series in the backward and forward operators:

$$v(B,F) = \sigma_s^2 \frac{\psi_s(B)\psi_s(F)}{\psi(B)\psi(F)} \,.$$

Replacing the polynomials ψ by the AR and MA terms –remember from equation (1) $\psi(L) = \frac{\theta(L)}{\phi(L)}$– and canceling roots in the ϕ polynomials, the filter can be expressed as:

$$v(B,F) = \sigma_s^2 \frac{\phi_n(B)\theta_s(B)\phi_n(F)\theta_s(F)}{\theta(B)\theta(F)} \,,$$

where $\phi_n()$ are the AR polynomials of the model for the non-signal component, i.e., the component other than s_t. Thus, the centered and symmetric filter $v(B,F)$ will also be convergent if the MA polynomial of the model fitted to the observed series is invertible.

Looking at the expression of the ACGF in equation (3), it can be deduced that the weights of the filter $v(B,F)$ are given by the ACGF of the model:

$$\theta(L)x_t = \phi_n(L)\theta_s(L)b_t \,, \quad b_t \sim \mathrm{NID}(0, \sigma_s^2) \,,$$

where $\theta(L)$ is the MA of the model fitted to the observed data, $\theta_s(L)$ is the MA of the component (signal) we want to estimate and $\phi_n(L)$ is the AR polynomial of the remaining components.

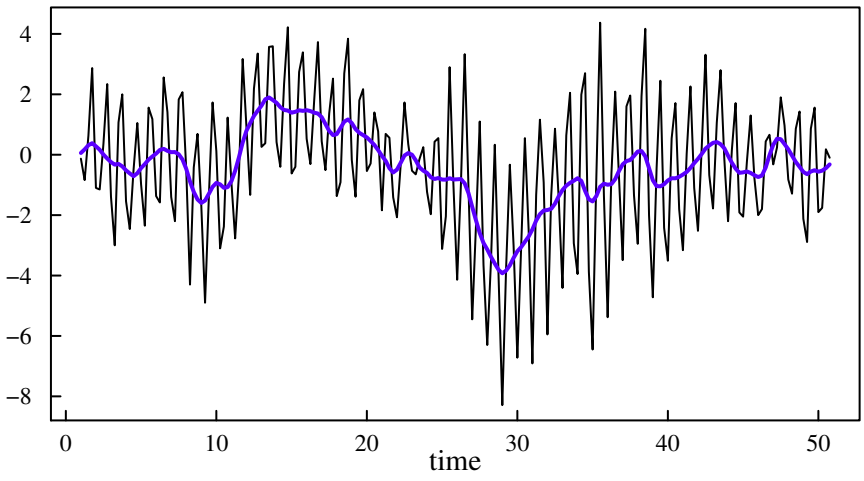
Based on the above result, an estimate of the trend is obtained below. The function `ARMAacov` is similar to `stats::ARMAacf`, but instead of the autocorrelations returns the theoretical autocovariances of an ARMA model.

```
n <- length(y)
nm1x2 <- 2*(n-1)
y2 <- c(rep(0, n-1), y, rep(0, n-1))
wtrend <- ARMAacov(ar=-ma.total[-1],
  ma=polyprod(p$seasonal, ma.trend.coefs)[-1],
  lag.max=n-1, sigma2=sigma2.trend)
trend <- filter(y2, filter=c(rev(wtrend[-1]), wtrend), method="conv", sides=1)
trend <- ts(trend[-seq_len(nm1x2)])
tsp(trend) <- tsp(y)
#plot(y)
#lines(trend)
```

According to the definition of the ARMA model used by `ARMAacov` (the AR coefficients are defined in the right-hand-side), the sign of $\theta(L)$ must be changed, because here it plays the role of the AR polynomial. Also, the first coefficient in the AR and MA polynomials $\phi_0 = 1$ and $\theta_0 = 1$ are not required by `ARMAacov`. $\phi_n(L)$ is the AR polynomial of the seasonal component, `p[["seasonal"]]`, and $\theta_s(L)$ the MA coefficients of the `ma.trend.coefs` component that is being estimated (trend). The estimated trend is shown in Figure 1. Similarly, an estimate of the seasonal component can be obtained as follows:

```
wseas <- ARMAacov(ar=-ma.total[-1], ma=polyprod(p$trend, ma.seas.coefs)[-1],
  lag.max=n-1, sigma2=sigma2.seas)
seas <- filter(y2, filter=c(rev(wseas[-1]), wseas), method="conv", sides=1)
seas <- ts(seas[-seq_len(nm1x2)])
tsp(seas) <- tsp(y)
```

Figure 1: Original series and estimated trend



3.8 Theoretical component, estimator and empirical signal

Recap: Two ARMA models are involved in the description of the methodology given above.

1) Theoretical ARMA models assumed for the components, given in equation (6):

$$\phi_s(L)s_t = \theta_s(L)b_t\,, \quad a_{s,t} \sim \text{NID}(0, \sigma_s^2)\,.$$

2) ARMA model implied by the estimator $\hat{s}_t$. Replacing the ARMA model for the observed series $x_t = \theta(B)/\phi(B)a_t$ in the expression of the estimator $\hat{s}_t = v(B, F)x_t$ and canceling AR factors yields:

$$\hat{s}_t = \sigma_s^2 \frac{\theta_s(B)\theta_s(F)\phi_n(F)}{\phi_s(B)\theta(F)} a_t\,, \quad a_t \sim \text{NID}(0, \sigma_a^2)\,.$$

The ACF of the ARMA models for the theoretical component and for the estimator, as well as the empirical ACF of the estimated signal can be compared in order to assess the reliability of the decomposition.

The ACF of the theoretical ARMA models for the trend and the seasonal components are obtained as follows. The stationary transformation of the model is considered, hence, only the stationary roots in the corresponding AR polynomials are taken:

```
acf1.trend <- ARMAacf(ar=-p$polys.stationary$trend[-1],
  ma=ma.trend.coefs[-1], lag.max=10)[-1]
acf1.seas <- ARMAacf(ar=-p$polys.stationary$seasonal[-1],
  ma=ma.seas.coefs[-1], lag.max=10)[-1]
head(cbind(acf1.trend, acf1.seas))
  acf1.trend   acf1.seas
1  0.9810417 -0.04226707
2  0.9438439 -0.81712549
3  0.9080566 -0.09469756
4  0.8736262  0.88508115
5  0.8405012 -0.03621205
6  0.8086323 -0.70006707
```

The ACF of the corresponding estimators can be computed as follows (again considering only the stationary roots of the AR polynomials; also, as the root of the MA polynomial `theta` turns out to be non-invertible and it belongs to the AR part of the ARMA model implied by the estimator, it is removed and the MA polynomial collapses to 1):

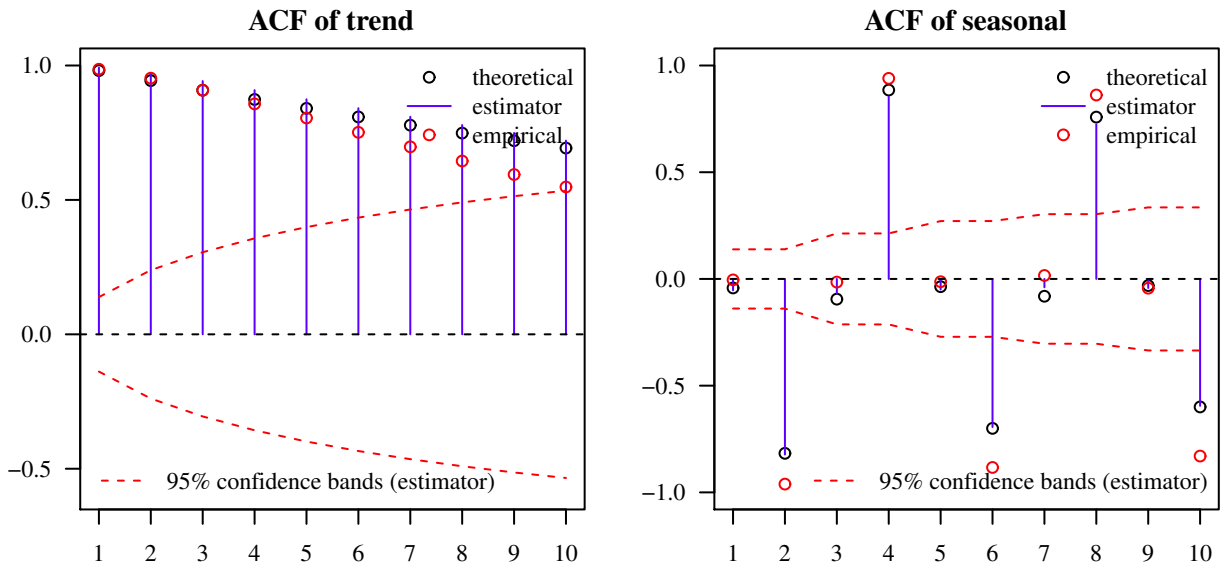
```
theta <- c(1, fit$model$theta[seq_len(fit$arma[2]+fit$arma[4]*fit$arma[5])])
print(Mod(polyroot(theta)))
[1] 1.826352

theta <- 1
ar.coefs <- -polyprod(p$polys.stationary$trend, theta)[-1]
ma.coefs <- convolve(ma.trend.coefs, ma.trend.coefs, type="open")
ma.coefs <- polyprod(ma.coefs, roots2poly(1/polyroot(p$transitory)))
ma.coefs <- polyprod(ma.coefs, roots2poly(1/polyroot(p$seasonal)))
acf2.trend <- ARMAacf(ar=ar.coefs, ma=ma.coefs[-1], lag.max=10)[-1]
ar.coefs <- -polyprod(p$polys.stationary$seasonal, theta)[-1]
ma.coefs <- convolve(ma.seas.coefs, ma.seas.coefs, type="open")
ma.coefs <- polyprod(ma.coefs, roots2poly(1/polyroot(p$trend)))
ma.coefs <- polyprod(ma.coefs, roots2poly(1/polyroot(p$transitory)))
acf2.seas <- ARMAacf(ar=ar.coefs, ma=ma.coefs[-1], lag.max=10)[-1]
head(cbind(acf2.trend, acf2.seas))
  acf2.trend   acf2.seas
1  0.9924311 -0.05339670
2  0.9716376 -0.82280267
3  0.9420733 -0.06789316
4  0.9084901  0.85370512
5  0.8743501 -0.04959640
6  0.8411978 -0.69403219
```

Finally, it is straightforward to obtain the ACF of the empirical estimates by means of `stats::acf`:

```
acf3.trend <- acf(trend, lag.max=10, plot=FALSE)$acf[-1,,1]
acf3.seas <- acf(seas, lag.max=10, plot=FALSE)$acf[-1,,1]
head(cbind(acf3.trend, acf3.seas))
     acf3.trend    acf3.seas
[1,]  0.9878892 -0.004427556
[2,]  0.9560306 -0.963391413
[3,]  0.9115250 -0.015458158
[4,]  0.8609166  0.940839480
[5,]  0.8084744 -0.011884538
[6,]  0.7552822 -0.884066935
```

Figure 2 displays the ACF of the three elements, along with 95% confidence bands (the standard deviation of the ACF is based on a Bartlett's approximation, similar to `stats::plot.acf(ci.type="ma")`).

Figure 2: Autocorrelation functions



4 Examples

4.1 Seasonal autoregressive model

Generate data and fit model:

```
set.seed(125)
y <- arima.sim(n=200, model=list(ar=c(0,0,0,0.6)))
y <- ts(round(y, 2), frequency=4)
fit <- arima(y, seasonal=list(order=c(1,0,0)), include.mean=FALSE)
c(coef(fit), sigma2 = fit$sigma2)
```

```
     sar1    sigma2
0.6841932 1.0959102
```

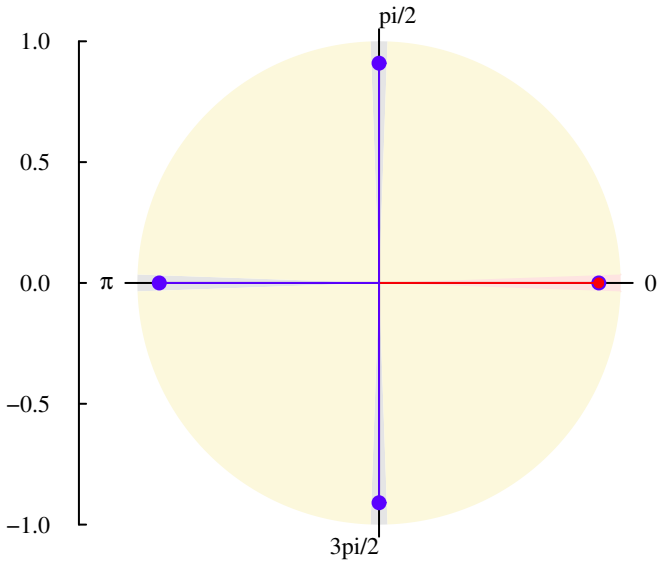
Fitted model:

$$(1 - 0.684L^4)y_t = \varepsilon_t\,, \quad \sigma_\varepsilon^2 = 1.096\,.$$

Roots allocation

```
p <- roots.allocation(fit)
print(p, units="pi")
Roots of AR polynomial
----------------------
(1 - 0.68L^4) = (1 - 0.909L)(1 + 0.909L + 0.827L^2 + 0.752L^3)

  Component             Root Modulus Argument Period Cycles.per.Year
1     trend  0.9095+0.0000i  0.9095        0    Inf               0
2  seasonal  0.0000+0.9095i  0.9095     pi/2  4.000               1
3  seasonal -0.9095+0.0000i  0.9095       pi  2.000               2
4  seasonal  0.0000-0.9095i  0.9095    3pi/2  1.333               3
Warning message:
In polynomial(p) : imaginary parts discarded in coercion
```

Figure 3: Allocation of roots in the fitted seasonal AR(1) model



Not all seasonal AR models generate cycles of seasonal periodicity. If the sign of the coefficient in the seasonal AR(1) is positive (when moved to the left-hand-side of the ARMA model), the factorisation does not involve roots related to seasonal frequencies. Example:

```
fit2 <- arima(y, seasonal=list(order=c(1,0,0)), include.mean=FALSE, fixed=-0.6)
print(coef(fit2))
sar1
-0.6
roots.allocation(fit2)
Roots of AR polynomial
----------------------
(1 + 0.6L^4) = (1 - 0L - 0L^2 - 0L^3 + 0.6L^4)

  Component            Root Modulus Argument Period Cycles.per.Year
1     trend  0.6223+0.6223i  0.8801   0.7854  8.000             0.5
2     trend -0.6223+0.6223i  0.8801   2.3562  2.667             1.5
3     trend -0.6223-0.6223i  0.8801   3.9270  1.600             2.5
4     trend  0.6223-0.6223i  0.8801   5.4978  1.143             3.5
Warning message:
In polynomial(p) : imaginary parts discarded in coercion
```

Pseudo-spectrum After some operations, we arrive to the following expression for the pseudo-spectrum (in the variable $x = 2\cos(\omega)$).
Polynomial division is not required.
Pseudo-spectrum (reference relationship):

$$\frac{1}{0.1 - 0x + 2.737x^2 - 0.684x^4} =$$

$$\frac{A(x)}{1.827 - 0.909x} + \frac{B(x)}{0.055 + 0.027x + 1.511x^2 + 0.752x^3}$$

Partial fraction decomposition The numerators of the partial fractions are obtained here. Actually, they are already returned by `pseudo.spectrum`, which calls `partial.fraction`. Here, we show this stage in more detail.

```
pfd <- partial.fraction(psp$total.numerator, psp$den$trend,
  psp$den$trans, psp$den$seas)
print(pfd$num.trend)
[1] 0.08124031
print(pfd$num.seasonal)
[1] 0.54487057 0.27000561 0.06719871
```

Set $A(x) = a_0$, $B(x) = b_0 + b_1x + b_2x^2$ and multiply the denominator of the LHS (products of the denominators in the RHS) by each term in the RHS.

$$1 = (0.055 + 0.027x + 1.511x^2 + 0.752x^3)A(x) + (1.827 - 0.909x)B(x)$$

Equating the coefficients related to elements of the same order in both sides of the equation yields the following system of equations:

$$\begin{bmatrix} 0.055 & 1.827 & 0 & 0 \\ 0.027 & -0.909 & 1.827 & 0 \\ 1.511 & 0 & -0.909 & 1.827 \\ 0.752 & 0 & 0 & -0.909 \end{bmatrix} \times \begin{bmatrix} a_0 \\ c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Solving the system gives the coefficients of the polynomials in the numerators of the partial fractions:

$$A(x) = 0.081\,,$$
$$B(x) = 0.545 + 0.27x + 0.067x^2\,.$$

Canonical decomposition

```
cd <- canonical.decomposition(pfd$num.trend, psp$den$trend,
  pfd$num.trans, psp$den$trans,
  pfd$num.seas, psp$den$seas, psp$quotient)
cd
MA polynomials
--------------

  Trend:
  (1 + L)a_t,  a_t ~ IID(0, 0.0203)
  Seasonal:
  (1 - 0.184L - 0.475L^2 - 0.341L^3)c_t,  c_t ~ IID(0, 0.245)

Variances
---------

    trend  seasonal irregular
  0.02026   0.24502   0.13349

Roots
-----

  Component           Root Modulus  Argument    Period
1     trend -1.000+0.0000i  1.0000 3.142e+00 2.000e+00
2  seasonal  1.000+0.0000i  1.0000 1.490e-08 4.217e+08
3  seasonal -0.408+0.4183i  0.5843 2.344e+00 2.681e+00
4  seasonal -0.408-0.4183i  0.5843 3.939e+00 1.595e+00
```

Models for the components. The denominators (AR) of the corresponding ARMA models are those obtained in the allocation of the AR roots; the numerators (MA) are those polynomials obtained

in the canonical decomposition.

$$\begin{aligned} \text{Trend:} \quad & (1-0.909L)T_t = (1+L)a_t\,, \quad \sigma_a^2 = 0.02\,. \\ \text{Seasonal:} \quad & (1+0.909L+0.827L^2+0.752L^3)S_t = \\ & (1-0.184L-0.475L^2-0.341L^3)c_t\,, \quad \sigma_c^2 = 0.245\,. \end{aligned}$$

Filtering The Wiener-Kolmogorov filter for the component s_t is given by the ACGF of the model:

$$\theta(L)z_t = \phi_n(L)\theta_s(L)a_t\,.$$

```
comp <- filtering(x=y, mod=fit,
  trend=list(ar=p$trend, ma=cd$trend$coef, sigma2=cd$trend$sigma2),
  transitory=list(ar=p$trans, ma=cd$trans$coef, sigma2=cd$trans$sigma2),
  seasonal=list(ar=p$seas, ma=cd$seas$coef, sigma2=cd$seas$sigma2),
  irregular.sigma2=cd$irregular.sigma2)
#plot(comp, overlap.trend = TRUE, args.trend = list(col="red"),
# set.pars = list(las=1))
```

4.2 The Airlines model and data

The Airlines model, ARIMA(0,1,1)(0,1,1), accommodates fairly well the dynamic of many macroeconomic time series data and is often used in practice. Here, we use this model with the classic Box & Jenkins airline data (monthly totals of international airline passengers, 1949 to 1960, in logarithms).

```
y <- round(log(AirPassengers),2)
fit <- arima(y, order=c(0,1,1), seasonal=list(order=c(0,1,1)))
c(coef(fit), sigma2 = fit$sigma2)
```

```
         ma1         sma1       sigma2
-0.405322943 -0.559778840  0.001340141
```

Fitted model:

$$(1-L)(1-L^{12})y_t = (1-0.405L)(1-0.56L^{12})\varepsilon_t\,, \quad \sigma_\varepsilon^2 = 0.001\,.$$

Roots allocation

```
p <- roots.allocation(fit)
print(polystring(p$trend))
```

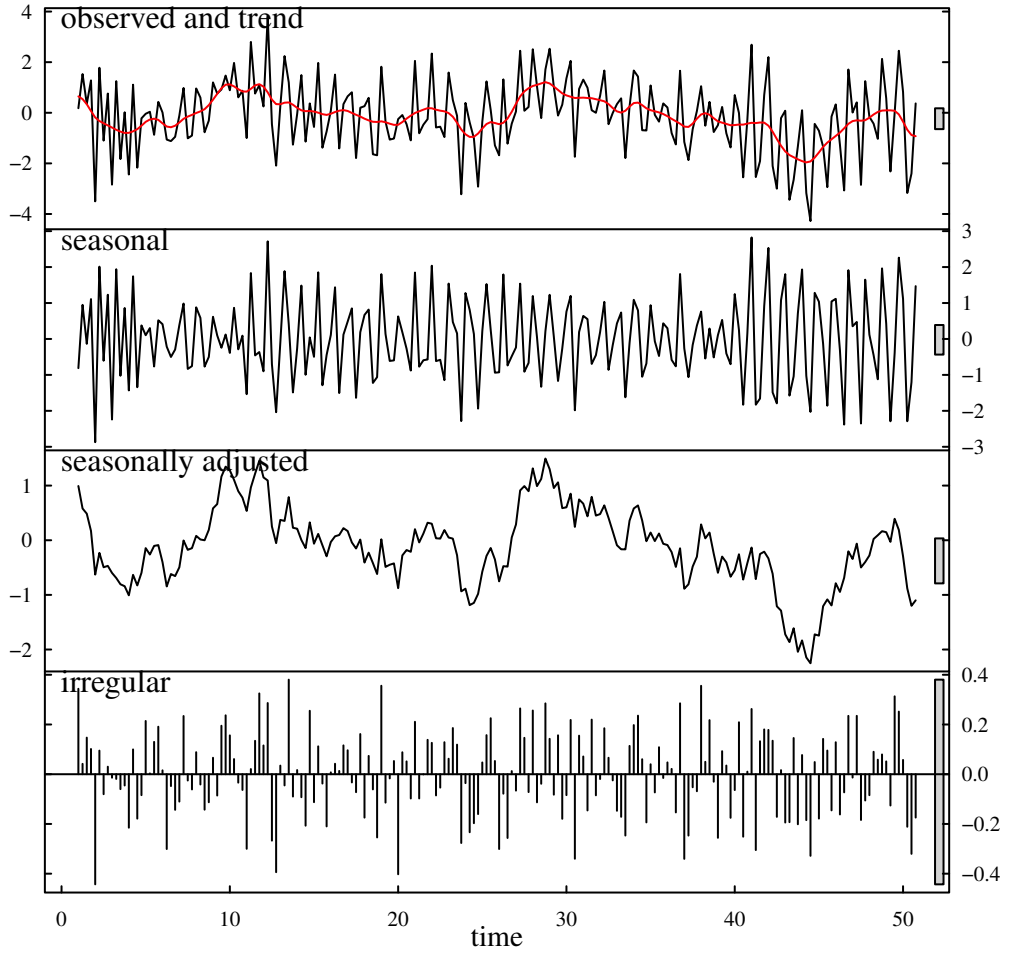
```
[1] "1 - 2x + x^2"
```

```
print(polystring(p$seas))
```

```
[1] "1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^10 + x^11"
```

```
print(p, units="pi")
```

Figure 4: Filtered components




```
Roots of AR polynomial
----------------------
(1 - L)(1 - L^12) = (1 - 2L + L^2)(1 + L + L^2 + L^3 + L^4 + L^5 + L^6 + L^7 + L^8 + L^9
```

	Component	Root	Modulus	Argument	Period	Cycles.per.Year
1	trend	1.000+0.000i	1	0	Inf	0
2	trend	1.000+0.000i	1	0	Inf	0
3	seasonal	0.866+0.500i	1	pi/6	12.000	1
4	seasonal	0.500+0.866i	1	2pi/6	6.000	2
5	seasonal	0.000+1.000i	1	3pi/6	4.000	3
6	seasonal	-0.500+0.866i	1	4pi/6	3.000	4
7	seasonal	-0.866+0.500i	1	5pi/6	2.400	5
8	seasonal	-1.000+0.000i	1	pi	2.000	6
9	seasonal	-0.866-0.500i	1	7pi/6	1.714	7
10	seasonal	-0.500-0.866i	1	8pi/6	1.500	8
11	seasonal	0.000-1.000i	1	9pi/6	1.333	9
12	seasonal	0.500-0.866i	1	10pi/6	1.200	10
13	seasonal	0.866-0.500i	1	11pi/6	1.091	11

Pseudo-spectrum After some operations, we arrive to the following expression for the pseudo-spectrum (in the variable $x = 2\cos(\omega)$).
Polynomial division:

$$\frac{0.23 - 0.08x + 23.46x^2 - 8.17x^3 - 68.43x^4 + 23.82x^5 + 73x^6 - 25.41x^7 - 35.19x^8 + 12.25x^9 + 7.82x^{10} - 2.72x^{11}}{72x^2 - 36x^3 - 210x^4 + 105x^5 + 224x^6 - 112x^7 - 108x^8 + 54x^9 + 24x^{10} - 12x^{11} - 2x^{12} + x^{13}}$$

$$\underbrace{0.226 - 0.079x + 7.127x^2 - 20.786x^4 + 22.172x^6 - 10.69x^8 + 2.376x^{10} - 0.198x^{12}}_{\text{Remainder}} + \underbrace{0.227}_{\text{Quotient}} .$$

Pseudo-spectrum (reference relationship):

$$\frac{0.226 - 0.079x + 7.127x^2 - 20.786x^4 + 22.172x^6 - 10.69x^8 + 2.376x^{10} - 0.198x^{12}}{72x^2 - 36x^3 - 210x^4 + 105x^5 + 224x^6 - 112x^7 - 108x^8 + 54x^9 + 24x^{10} - 12x^{11} - 2x^{12} + x^{13}} =$$

$$\frac{A(x)}{4 - 4x + x^2} + \frac{B(x)}{18x^2 + 9x^3 - 48x^4 - 24x^5 + 44x^6 + 22x^7 - 16x^8 - 8x^9 + 2x^{10} + x^{11}} .$$

Partial fraction decomposition The numerators of the partial fractions are obtained here. Actually, they are already returned by `pseudo.spectrum`, which calls `partial.fraction`. Here, we show this stage in more detail.

```
pfd <- partial.fraction(psp$total.numerator, psp$den$trend,
  psp$den$trans, psp$den$seas)
print(pfd$num.trend)
[1]  0.4088311 -0.2041776
print(pfd$num.seasonal)
 [1]  0.056408138  0.036770782 -0.035425600 -0.045689135  0.132073693
 [6]  0.146351548 -0.065977061 -0.105182554 -0.002855571  0.024391925
[11]  0.006216963
```

Set $A(x) = a_0 + a_1 x$, $B(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 + b_5 x^5 + b_6 x^6 + b_7 x^7 + b_8 x^8 + b_9 x^9 + b_{10} x^{10}$ and multiply the denominator of the LHS (products of the denominators in the RHS) by each term in the RHS.

$$0.226 - 0.079x + 7.127x^2 - 20.786x^4 + 22.172x^6 - 10.69x^8 + 2.376x^{10} - 0.198x^{12} =$$

$$(18x^2 + 9x^3 - 48x^4 - 24x^5 + 44x^6 + 22x^7 - 16x^8 - 8x^9 + 2x^{10} + x^{11})A(x) + (4 - 4x + x^2)B(x) .$$

Equating the coefficients related to elements of the same order in both sides of the equation yields the following system of equations:

$$\begin{bmatrix} 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 18 & 0 & 1 & -4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 9 & 18 & 0 & 1 & -4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -48 & 9 & 0 & 0 & 1 & -4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ -24 & -48 & 0 & 0 & 0 & 1 & -4 & 4 & 0 & 0 & 0 & 0 & 0 \\ 44 & -24 & 0 & 0 & 0 & 0 & 1 & -4 & 4 & 0 & 0 & 0 & 0 \\ 22 & 44 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 4 & 0 & 0 & 0 \\ -16 & 22 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 4 & 0 & 0 \\ -8 & -16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 4 & 0 \\ 2 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 4 \\ 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} a_0 \\ a_1 \\ b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \\ b_{10} \end{bmatrix} = \begin{bmatrix} 0.226 \\ -0.079 \\ 7.127 \\ 0 \\ -20.786 \\ 0 \\ 22.172 \\ 0 \\ -10.69 \\ 0 \\ 2.376 \\ 0 \\ -0.198 \end{bmatrix}.$$

Solving the system gives the coefficients of the polynomials in the numerators of the partial fractions:

$$\begin{aligned} A(x) &= 0.409 - 0.204x\,, \\ B(x) &= 0.056 + 0.037x - 0.035x^2 - 0.046x^3 + 0.132x^4 + 0.146x^5 \\ &\quad - 0.066x^6 - 0.105x^7 - 0.003x^8 + 0.024x^9 + 0.006x^{10}\,. \end{aligned}$$

Canonical decomposition As explained in Section 3.5, among all the solutions that meet the relationship of the pseudo-spectrum, the canonical decomposition chooses the solution that maximises the variance of the irregular component. To do so, the value (variance) at which the pseudo-spectrum defined for each component reaches a minimum is removed from the component and assigned to the irregular.

For the trend, the minimum is reached at 0.051:

```
fobj <- function(x, p1, p2) polyeval(p1, x) / polyeval(p2, x)
res <- optimize(f=fobj, interval=c(-2, 2), p1=psp$num$trend, p2=psp$den$trend)
trend.minval <- min(res$obj, fobj(-2, psp$num$trend, psp$den$trend))
print(trend.minval)
[1] 0.05107415
num.trend <- c(psp$num$trend, 0) - psp$den$trend * trend.minval
irregular.sigma2 <- trend.minval
polystring(num.trend)
[1] "0.2 - 0x - 0.05x^2"
```

This also gives us the final polynomial `num.trend`, which is related to the MA polynomial of the model obtained for the trend component.

```
ma.trend.acovs <- solve(mat.acgf2poly[1:3,1:3]) %*% num.trend
tmp <- acov2ma(x=ma.trend.acovs, init=acov2ma.init(ma.trend.acovs)$coef)
print(tmp$coef)
```

```
            [,1]
[1,]  1.00000000
[2,]  0.04711517
[3,] -0.95288488
```

```
print(tmp$sigma2)
```

```
[1] 0.05359949
```

For the seasonal component, we proceed in the same way and obtain the autocovariances of the MA related to the model obtained for the seasonal component.

```
res <- optimize(f=fobj, interval=c(-2, 2), p1=psp$num$seas, p2=psp$den$seas)
seas.minval <- min(res$obj, fobj(-2, psp$num$seas, psp$den$seas))
print(seas.minval)
```

```
[1] 0.02389556
```

```
num.seas <- c(psp$num$seas, 0) - psp$den$seas * seas.minval
irregular.sigma2 <- irregular.sigma2 + seas.minval
polystring(num.seas)
ma.seas.acovs <- solve(mat.acgf2poly[1:12,1:12]) %*% num.seas
print(ma.seas.acovs)
```

```
              [,1]
 [1,]  0.538495912
 [2,]  0.492360767
 [3,]  0.409907951
 [4,]  0.311096568
 [5,]  0.211126435
 [6,]  0.120914015
 [7,]  0.047568343
 [8,] -0.005133046
 [9,] -0.036268189
[10,] -0.047294763
[11,] -0.041574162
[12,] -0.023895563
```

Here, the algorithm that computes numerically the mapping from the autocovariances to the coefficients of a MA model, failed to converge to a stable or reliable solution: `acov2ma.init` converged to a zero variance and `maxiter` needed to be limited to 4; `acov2ma` did not converge even for a relatively large number of iterations.

```
tmp1 <- acov2ma(x=ma.seas.acovs, maxiter=120,
  init=acov2ma.init(ma.seas.acovs, maxiter=4)$coef)
tmp2 <- acov2ma(x=ma.seas.acovs, maxiter=150,
  init=acov2ma.init(ma.seas.acovs, maxiter=4)$coef)
print(cbind(tmp1$coef, tmp2$coef))
```

```
             [,1]        [,2]
 [1,]  1.00000000  1.00000000
```

```
 [2,]  1.36260374  1.22683436
 [3,]  1.49662749  1.23755060
 [4,]  1.39583895  1.47400757
 [5,]  2.01534420  1.63553658
 [6,]  1.95775615  1.86603511
 [7,]  1.85072027  1.69490622
 [8,]  1.48905271  1.13149947
 [9,]  0.43106355  0.55519604
[10,]  0.08802272 -0.08150671
[11,] -0.44760418 -0.40217737
[12,] -0.92720037 -0.77939083
```

```
print(c(tmp1$sigma2, tmp2$sigma2))
```

```
[1] 0.02466157 0.03059028
```

This problem may come from the presence of several roots of modulus greater than unity. Below, the other approach discussed in Section 3.6 is applied:

```
r <- polyroot(num.seas)
tmp <- sqrt(r^2 - 4)
r <- c((r + tmp)/2, (r - tmp)/2)
r <- r[order(Mod(r))]
r <- r[seq_len(length(r)/2)]
mapoly <- roots2poly(1/r)
sigma2 <- num.seas[length(num.seas)]/Re(prod(-r))
print(mapoly)
```

```
 [1]  1.00000000  1.42676056  1.57667998  1.47559517  1.25346821  1.01161355
 [7]  0.74316608  0.44023586  0.18941725  0.03266987 -0.16646163 -0.50043696
```

```
print(sigma2)
```

```
[1] 0.0477494
```

We can now build the coefficients of the MA polynomial from these roots. The interesting point of this approach is that we can now choose the roots to be included in the MA polynomial, in particular, roots with modulus larger than unity can be discarded.

```
cd <- canonical.decomposition(pfd$num.trend, psp$den$trend,
  pfd$num.trans, psp$den$trans,
  pfd$num.seas, psp$den$seas, psp$quotient)
cd
```

```
MA polynomials
--------------

  Trend:
  (1 + 0.047L - 0.953L^2)a_t,  a_t ~ IID(0, 0.0536)
  Seasonal:
  (1 + 1.427L + 1.577L^2 + 1.476L^3 + 1.253L^4 + 1.012L^5 + 0.743L^6 + 0.44L^7 + 0.189L^8
```

```
Variances
---------

    trend  seasonal irregular
  0.05360   0.04775   0.30186

Roots
-----

   Component            Root Modulus Argument Period
1      trend  0.9529+0.0000i  0.9529   0.0000    Inf
2      trend -1.0000+0.0000i  1.0000   3.1416  2.000
3   seasonal  0.6784+0.0000i  0.6784   0.0000    Inf
4   seasonal  0.6381+0.6488i  0.9100   0.7938  7.916
5   seasonal  0.2422+0.9178i  0.9493   1.3128  4.786
6   seasonal -0.2604+0.9655i  1.0000   1.8343  3.425
7   seasonal -0.7317+0.6816i  1.0000   2.3916  2.627
8   seasonal -0.9758+0.2188i  1.0000   2.9210  2.151
9   seasonal -0.9546-0.2980i  1.0000   3.4442  1.824
10  seasonal -0.6827-0.7307i  1.0000   3.9610  1.586
11  seasonal -0.2604-0.9655i  1.0000   4.4489  1.412
12  seasonal  0.2422-0.9178i  0.9493   4.9703  1.264
13  seasonal  0.6381-0.6488i  0.9100   5.4894  1.145
```

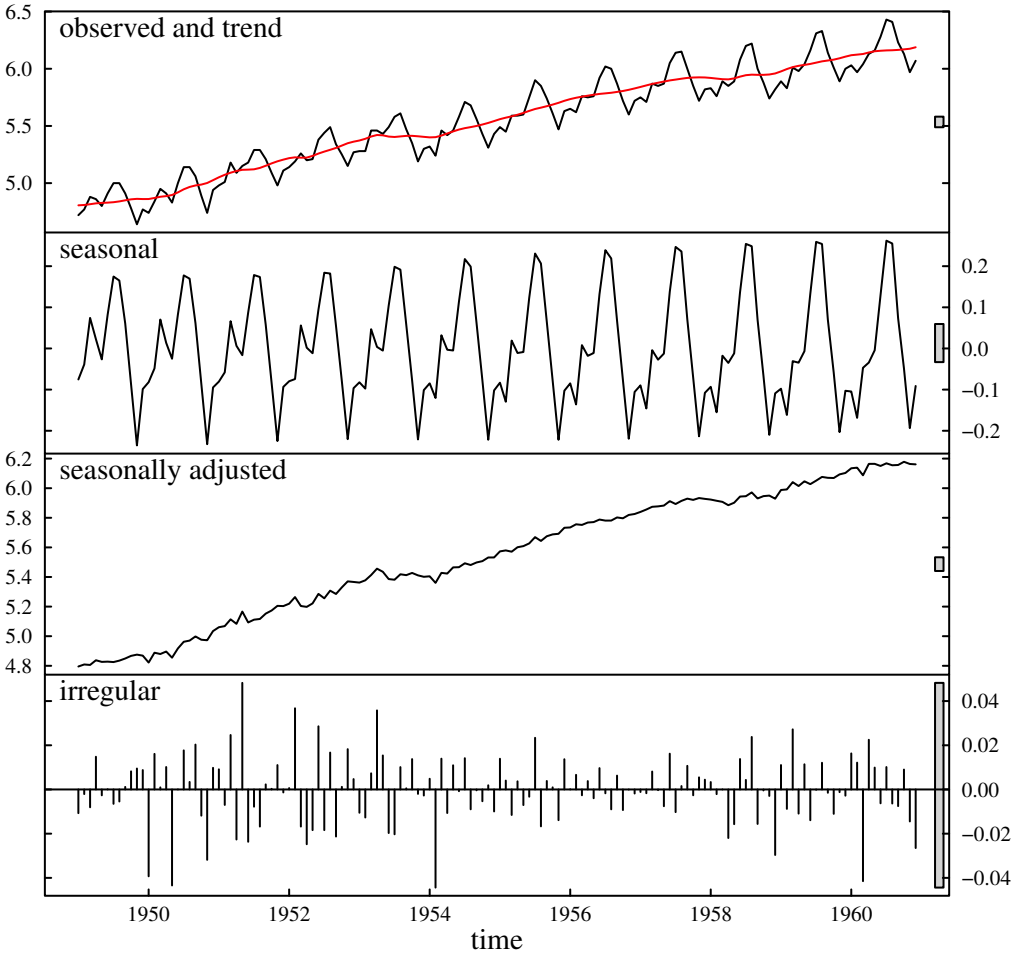
Models for the components. The denominators (AR) of the corresponding ARMA models are those obtained in the allocation of the AR roots; the numerators (MA) are those polynomials obtained in the canonical decomposition.

$$\begin{aligned}
\text{Trend:}\quad & (1-2L+L^2)T_t = (1+0.047L-0.953L^2)a_t\,, \quad \sigma_a^2 = 0.054\,. \\
\text{Seasonal:}\quad & (1+L+L^2+L^3+L^4+L^5+L^6+L^7+L^8+L^9+L^{10}+L^{11})S_t = \\
& (1+1.427L+1.577L^2+1.476L^3+1.253L^4+1.012L^5+ \\
& 0.743L^6+0.44L^7+0.189L^8+0.033L^9-0.166L^{10}-0.5L^{11})c_t\,, \quad \sigma_c^2 = 0.048\,.
\end{aligned}$$

The components and seasonally adjusted series are obtained as follows:

```
comp <- filtering(x=y, mod=fit,
  trend=list(ar=p$trend, ma=cd$trend$coef, sigma2=cd$trend$sigma2),
  transitory=list(ar=p$trans, ma=cd$trans$coef, sigma2=cd$trans$sigma2),
  seasonal=list(ar=p$seas, ma=cd$seas$coef, sigma2=cd$seas$sigma2),
  irregular.sigma2=cd$irregular.sigma2, extend=72)
#plot(comp, overlap.trend = TRUE, args.trend = list(col="red"))
```

Figure 5: Airlines data (logs.). Filtered components



4.3 Polynomial division and transitory component

This example shows a case (an ARIMA(0,1,2) model) where a transitory component which is not set in the allocation of the AR roots, shows up when performing polynomial division in the pseudospectrum.

Generate data and fit model:

```
set.seed(123)
y <- arima.sim(n=200, model=list(order=c(0,1,2), ma=c(-0.6, 0.2)))
y <- ts(round(y, 2))
fit <- arima(y, order=c(0,1,2), include.mean=FALSE)
c(coef(fit), sigma2 = fit$sigma2)
       ma1        ma2     sigma2
-0.6764766  0.1931481  0.9051298
```

Fitted model:

$$(1-L)y_t = (1 - 0.676L + 0.193L^2)\varepsilon_t\,, \quad \sigma_\varepsilon^2 = 0.905\,.$$

Roots allocation

```
p <- roots.allocation(fit)
print(p, units="pi")
Roots of AR polynomial
----------------------
(1 - L) = (1 - L)

  Component Root Modulus Argument Period Cycles.per.Year
1     trend    1       1        0    Inf               0
```

Pseudo-spectrum After some operations, we arrive to the following expression for the pseudo-spectrum (in the variable $x = 2\cos(\omega)$).
Polynomial division:

$$\frac{1.109 - 0.807x + 0.193x^2}{2 - x} = \underbrace{0.267}_{\text{Remainder}} + \underbrace{0.421 - 0.193x}_{\text{Quotient}} .$$

Pseudo-spectrum (reference relationship):

$$\frac{0.267}{2 - x} = \frac{A(x)}{2 - x} .$$

Partial fraction decomposition The numerators of the partial fractions are obtained here. Actually, they are already returned by `pseudo.spectrum`, which calls `partial.fraction`. Here, we show this stage in more detail.

```
pfd <- partial.fraction(psp$total.numerator, psp$den$trend,
  psp$den$trans, psp$den$seas)
print(pfd$num.trend)
[1] 0.2669495
print(pfd$num.seasonal)
NULL
```

Set $A(x) = a_0$ and multiply the denominator of the LHS (products of the denominators in the RHS) by each term in the RHS.

$$0.267 = (1)A(x) .$$

Equating the coefficients related to elements of the same order in both sides of the equation yields the following system of equations:

$$\left[\ 1\ \right] \times \left[\ a_0\ \right] = \left[\ 0.267\ \right] .$$

Solving the system gives the coefficients of the polynomials in the numerators of the partial fractions:

$$A(x) = 0.267 .$$

Canonical decomposition

```
cd <- canonical.decomposition(pfd$num.trend, psp$den$trend,
  pfd$num.trans, psp$den$trans,
  pfd$num.seas, psp$den$seas, psp$quotient)
cd
MA polynomials
--------------

  Trend:
  (1 + L)a_t,  a_t ~ IID(0, 0.0667)
  Transitory:
  (1 - L)b_t,  b_t ~ IID(0, 0.1931)

Variances
---------

     trend transitory  irregular
   0.06674    0.19315    0.10128

Roots
-----

   Component  Root Modulus Argument Period
1      trend -1+0i       1    3.142      2
2 transitory  1+0i       1    0.000    Inf
```

Models for the components. The denominators (AR) of the corresponding ARMA models are those obtained in the allocation of the AR roots; the numerators (MA) are those polynomials obtained in the canonical decomposition.

$$\begin{aligned} \text{Trend:} \quad & (1-L)T_t = (1+L)a_t\,, \quad \sigma_a^2 = 0.067\,. \\ \text{Transitory:} \quad & (1)C_t = (1-L)b_t\,, \quad \sigma_b^2 = 0.193\,. \end{aligned}$$

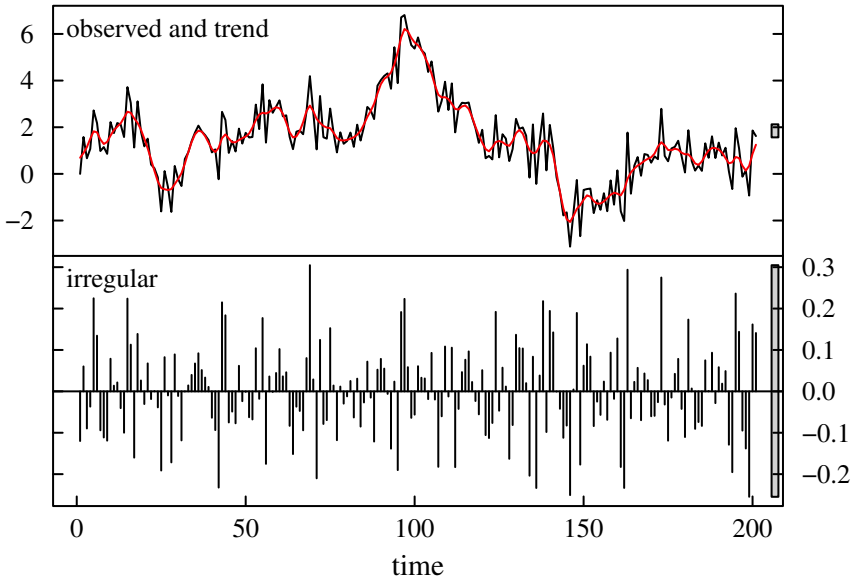
Filtering The Wiener-Kolmogorov filter for the component s_t is given by the ACGF of the model:

$$\theta(L)z_t = \phi_n(L)\theta_s(L)a_t\,.$$

```
comp <- filtering(x=y, mod=fit,
  trend=list(ar=p$trend, ma=cd$trend$coef, sigma2=cd$trend$sigma2),
  transitory=list(ar=p$trans, ma=cd$trans$coef, sigma2=cd$trans$sigma2),
  seasonal=list(ar=p$seas, ma=cd$seas$coef, sigma2=cd$seas$sigma2),
  irregular.sigma2=cd$irregular.sigma2)
```

```
#plot(comp, select = c("observed", "trend", "irregular"),
# overlap.trend = TRUE, args.trend = list(col="red"))
```

Figure 6: Filtered components



4.4 Three components

Generate data and fit model:

```
set.seed(123)
y <- arima.sim(n=200, model=list(order=c(1,1,0), ar=c(0.6)))[-1]
y <- diffinv(y, lag=4)[-seq_len(4)]
y <- round(ts(y, frequency=4), 2)
fit <- arima(y, order=c(1,1,1), seasonal=list(order=c(0,1,0)))
c(coef(fit), sigma2 = fit$sigma2)
       ar1         ma1      sigma2
0.52252573 0.01226773 0.87379835
```

Fitted model:

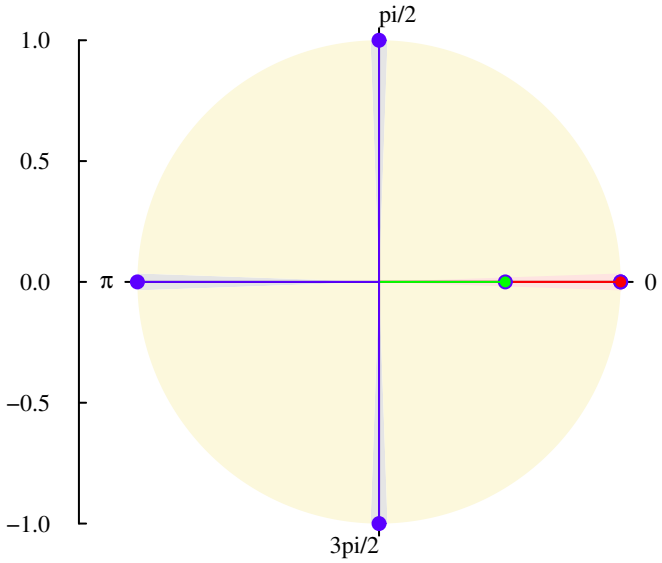
$$(-0.523)(1-L)(1-L^4)y_t = (1+0.012L)\varepsilon_t\,, \quad \sigma_\varepsilon^2 = 0.874\,.$$

Roots allocation

```
p <- roots.allocation(fit, min.modulus = 0.6)
print(p, units="pi")
Roots of AR polynomial
----------------------
(1 - 0.523L)(1 - L)(1 - L^4) = (1 - 2L + L^2)(1 - 0.523L)(1 + L + L^2 + L^3)
```

```
   Component       Root Modulus Argument Period Cycles.per.Year
1      trend  1.0000+0i  1.0000        0    Inf               0
2      trend  1.0000+0i  1.0000        0    Inf               0
3 transitory  0.5225+0i  0.5225        0    Inf               0
4   seasonal  0.0000+1i  1.0000     pi/2  4.000               1
5   seasonal -1.0000+0i  1.0000       pi  2.000               2
6   seasonal  0.0000-1i  1.0000    3pi/2  1.333               3
```

Figure 7: Allocation of roots in the three components model



Pseudo-spectrum After some operations, we arrive to the following expression for the pseudo-spectrum (in the variable $x = 2\cos(\omega)$).
Polynomial division is not required.
Pseudo-spectrum (reference relationship):

$$\frac{1 + 0.012x}{10.184x^2 - 9.272x^3 - 0.456x^4 + 2.318x^5 - 0.523x^6} =$$

$$\frac{A(x)}{4 - 4x + x^2} + \frac{B(x)}{1.273 - 0.523x} + \frac{C(x)}{2x^2 + x^3}\,.$$

Partial fraction decomposition The numerators of the partial fractions are obtained here. Actually, they are already returned by `pseudo.spectrum`, which calls `partial.fraction`. Here, we show this stage in more detail.

```
pfd <- partial.fraction(psp$total.numerator, psp$den$trend,
  psp$den$trans, psp$den$seas)
print(pfd$num.trend)
```

```
[1] -0.3112149  0.2960636
print(pfd$num.seasonal)
[1] 0.19641093 0.27943838 0.09719257
```

Set $A(x) = a_0 + a_1 x$, $B(x) = b_0$ and $C(x) = c_0 + c_1 x + c_2 x^2$ and multiply the denominator of the LHS (products of the denominators in the RHS) by each term in the RHS.

$$\begin{aligned} 1 + 0.012x = (2.546x^2 + 0.228x^3 - 0.523x^4)A(x) + (8x^2 - 4x^3 - 2x^4 + x^5)B(x) \\ + (5.092 - 7.182x + 3.363x^2 - 0.523x^3)C(x)\,. \end{aligned}$$

Equating the coefficients related to elements of the same order in both sides of the equation yields the following system of equations:

$$\begin{bmatrix} 0 & 0 & 0 & 5.092 & 0 & 0 \\ 0 & 0 & 0 & -7.182 & 5.092 & 0 \\ 2.546 & 0 & 8 & 3.363 & -7.182 & 5.092 \\ 0.228 & 2.546 & -4 & -0.523 & 3.363 & -7.182 \\ -0.523 & 0.228 & -2 & 0 & -0.523 & 3.363 \\ 0 & -0.523 & 1 & 0 & 0 & -0.523 \end{bmatrix} \times \begin{bmatrix} a_0 \\ a_1 \\ b_0 \\ c_0 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.012 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Solving the system gives the coefficients of polynomials in the numerators of the partial fractions:

$$\begin{aligned} A(x) &= -0.311 + 0.296x\,, \\ B(x) &= 0.205\,, \\ C(x) &= 0.196 + 0.279x + 0.097x^2\,. \end{aligned}$$

Canonical decomposition

```
cd <- canonical.decomposition(pfd$num.trend, psp$den$trend,
  pfd$num.trans, psp$den$trans,
  pfd$num.seas, psp$den$seas, psp$quotient)
cd
MA polynomials
--------------

  Trend:
  (1 - 0.102L + L^2)a_t,  a_t ~ IID(0, 0.078)
  Transitory:
  (1 + L)b_t,  b_t ~ IID(0, 0.0463)
  Seasonal:
  (1 + 1.488L + 1.059L^2 + 0.041L^3)c_t,  c_t ~ IID(0, 0.0936)

Variances
---------

     trend transitory   seasonal  irregular
```

```
  0.078008   0.046321   0.093610   0.006808

Roots
-----

   Component             Root Modulus Argument Period
1      trend  0.05118+0.9987i 1.00000    1.520  4.135
2      trend  0.05117-0.9987i 1.00000    4.764  1.319
3 transitory -0.99998+0.0061i 1.00000    3.135  2.004
4   seasonal -0.72364+0.6902i 1.00000    2.380  2.640
5   seasonal -0.04091+0.0000i 0.04091    3.142  2.000
6   seasonal -0.72365-0.6902i 1.00000    3.903  1.610
```

Models for the components. The denominators (AR) of the corresponding ARMA models are those obtained in the allocation of the AR roots; the numerators (MA) are those polynomials obtained in the canonical decomposition.

$$\begin{aligned}
\text{Trend:} \quad & (1 - 2L + L^2)T_t = (1 - 0.102L + L^2)a_t\,, \quad \sigma_a^2 = 0.078\,. \\
\text{Transitory:} \quad & (1 - 0.523L)C_t = (1 + L)b_t\,, \quad \sigma_b^2 = 0.046\,. \\
\text{Seasonal:} \quad & (1 + L + L^2 + L^3)S_t = \\
& (1 + 1.488L + 1.059L^2 + 0.041L^3)c_t\,, \quad \sigma_c^2 = 0.094\,.
\end{aligned}$$

4.5 Economic cycle

Clark (1987) decomposed the real Gross Domestic Product (GDP) into a stochastic trend and a transitory component, where the latter captures a signal that resembles the economic cycle. Here, we will get an approximation to the economic cycle by means of an ARIMA(1,1,0) with a deterministic drift. The model is fitted as follows:

```
y <- log(gdp4795)
fit1 <- arima(y, order=c(1,1,0), xreg=cbind(drift=seq_along(y)))
fit1

Call:
arima(x = y, order = c(1, 1, 0), xreg = cbind(drift = seq_along(y)))

Coefficients:
         ar1   drift
      0.3673  0.0077
s.e.  0.0665  0.0010

sigma^2 estimated as 8.394e-05:  log likelihood = 635.04,  aic = -1264.08
```

The root if the differencing filter is assigned to the trend, while the root if the stationary AR polynomial is assigned to the transitory (due to a low modulus):

```
p1 <- roots.allocation(fit1)
p1
Roots of AR polynomial
----------------------
(1 - 0.367L)(1 - L) = (1 - L)(1 - 0.367L)

   Component      Root Modulus Argument Period Cycles.per.Year
1      trend 1.0000+0i  1.0000        0    Inf               0
2 transitory 0.3673+0i  0.3673        0    Inf               0
```

The canonical decomposition of this model is not admissible since it leads to a negative variance in the irregular component. In particular, the amount of noise that is extracted from the trend to be assigned to the irregular is large and negative (is not compensated with transitory):

```
psp <- pseudo.spectrum(fit1, p1)
fobj <- function(x, p1, p2) polyeval(p1, x) / polyeval(p2, x)
tmp <- optimize(f=fobj, interval=c(-2, 2),
  p1=psp$num$trend, p2=psp$den$trend)
print(tmp)
$minimum
[1] -1.999959

$objective
[1] 0.6245433

trend.minval <- tmp$obj
trans.minval <- optimize(f=fobj, interval=c(-2, 2),
  p1=psp$num$trans, p2=psp$den$trans)$obj
irregular.sigma2 <- psp$quotient + trend.minval + trans.minval
print(irregular.sigma2)
[1] -1.667617
```

The alternative that we adopt here is to fit an AR(1) model with mean for the differenced series. The model is exactly the same as the model fitted above (the first differences of `drift=seq_along(y)` become a constant).

```
fit2 <- arima(diff(y), order=c(1,0,0), include.mean=TRUE)
fit2

Call:
arima(x = diff(y), order = c(1, 0, 0), include.mean = TRUE)

Coefficients:
         ar1  intercept
      0.3673     0.0077
```

```
s.e.  0.0665     0.0010

sigma^2 estimated as 8.394e-05:  log likelihood = 635.04,  aic = -1264.08
```

With this formulation of the model, the unit root of the differencing filter is not considered and the decomposition is now admissible.

```
p2 <- roots.allocation(fit2)
print(p2, units="pi")
Roots of AR polynomial
----------------------
(1 - 0.367L) = (1 - 0.367L)

   Component      Root Modulus Argument Period Cycles.per.Year
1 transitory 0.3673+0i  0.3673        0    Inf               0
psp <- pseudo.spectrum(fit2, p2)
pfd <- partial.fraction(psp$total.numerator, psp$den$trend,
  psp$den$trans, psp$den$seas)
cd <- canonical.decomposition(pfd$num.trend, psp$den$trend,
  pfd$num.trans, psp$den$trans,
  pfd$num.seas, psp$den$seas, psp$quotient)
cd
MA polynomials
--------------

  Transitory:
  (1 + L)b_t,  b_t ~ IID(0, 0.1965)

Variances
---------

transitory  irregular
    0.1965     0.5349

Roots
-----

   Component       Root Modulus Argument Period
1 transitory -1+0.0079i       1    3.134  2.005
```

Before filtering the data for obtaining the estimate of the cycle, we must be aware that the mean in the differenced series is actually a linear trend in the original series. The linear trend is first removed as follows.

```
linear.trend <- coef(fit2)["intercept"]*seq_along(y)
y2 <- y - linear.trend
```

The estimate of the component will be a smoothed version of the detrended series, `y2`. Before applying the filter, we extend the series with forecasts:

```
extend <- 16
fitaux <- arima(y2, order=c(1,1,0),
  include.mean=FALSE, fixed=coef(fit2)[1])
pred.right <- predict(fitaux, n.ahead=extend, se.fit=FALSE)
revy2 <- ts(rev(y2))
tsp(revy2) <- tsp(y2)
fitaux <- arima(revy2, order=c(1,1,0), include.mean=FALSE)
pred.left <- predict(fitaux, n.ahead=extend, se.fit=FALSE)
yext <- ts(c(rev(pred.left), y2, pred.right),
  frequency=frequency(y), start=time(y)[1]-extend/frequency(y))
```

Now the cycle (transitory component) can be estimated:

```
n <- length(yext)
nm1x2 <- 2*(n-1)
wtrans <- ARMAacov(ma=cd$transitory$coef[-1], lag.max=n-1,
  sigma2=cd$transitory$sigma2)
trans <- ts(filter(c(rep(0, n-1), yext, rep(0, n-1)),
  filter=c(rev(wtrans[-1]), wtrans), method="conv", sides=1)[-seq_len(nm1x2)])
tsp(trans) <- tsp(yext)
trans <- window(trans, start=start(y), end=end(y))
#plot(trans)
```

Despite the unit root related to the differencing filter has not been assigned to any component, the original series is recovered from this decomposition. It can be checked below, where the linear trend and the irregular component added up to the transitory component obtained above:

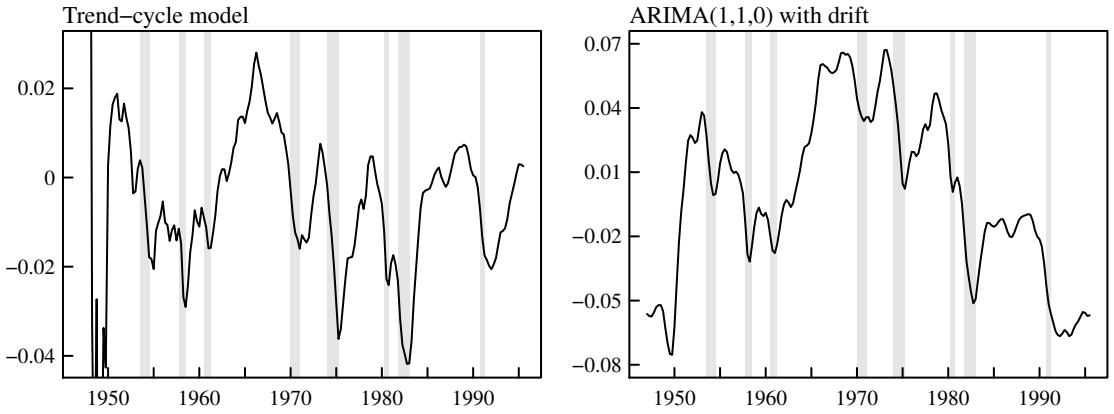
```
fres <- filtering(x=y, mod=fit2, drift=TRUE,
  transitory=list(ar=p2$trans, ma=cd$trans$coef, sigma2=cd$trans$sigma2),
  irregular.sigma2=cd$irregular.sigma2, extend=16)
print(all.equal(trans, fres$comp[,"transitory"]))
[1] TRUE
tmp <- rowSums(fres$comp[,c("trend", "transitory", "irregular")])
print(all.equal(tmp, c(y)))
[1] "Mean relative difference: 1.109554e-05"
```

The estimated economic cycle obtained by means of Clark's model and the ARIMA(1,1,0) with drift discussed here is shown in Figure 8 (recession periods dated by the NBER are shaded in gray.)

4.6 UK consumption

Fit model to the logarithms of the UK consumption and detect possible outliers:

Figure 8: Estimated cyclical component




```
require("tsoutliers")
y <- log(UKconsumption)
fitxreg <- tso(y, cval=3.8, tsmethod="arima", args.tsmethod=list(order=c(1,0,0),
  seasonal=list(order=c(0,1,1)), xreg=cbind(drift=seq_along(y))))
fitxreg$fit$call <- NULL
fit <- fitxreg$fit
fitxreg
Loading required package: tsoutliers

Call:
structure(list(method = NULL), .Names = "method")

Coefficients:
         ar1     sma1   drift    AO53    TC73    AO98
      0.8807  -0.3787  0.0061  0.0403  0.0447  0.0543
s.e.  0.0386   0.0806  0.0012  0.0076  0.0096  0.0076

sigma^2 estimated as 0.0001466:  log likelihood = 466.39,  aic = -918.79

Outliers:
  type ind    time coefhat tstat
1   AO  53 1968:01 0.04028 5.334
2   TC  73 1973:01 0.04474 4.639
3   AO  98 1979:02 0.05428 7.097

dec <- ARIMAdec(y, fit, drift=TRUE, extend=16)
dec$ar
Roots of AR polynomial
----------------------
(1 - 0.881L)(1 - L^4) = (1 - 1.881L + 0.881L^2)(1 + L + L^2 + L^3)
```

```
  Component       Root Modulus Argument Period Cycles.per.Year
1     trend  1.0000+0i  1.0000    0.000    Inf               0
2     trend  0.8807+0i  0.8807    0.000    Inf               0
3  seasonal  0.0000+1i  1.0000    1.571  4.000               1
4  seasonal -1.0000+0i  1.0000    3.142  2.000               2
5  seasonal  0.0000-1i  1.0000    4.712  1.333               3
```

```
dec$ma
```

```
MA polynomials
--------------

  Trend:
  (1 + 0.214L - 0.786L^2)a_t,  a_t ~ IID(0, 0.1314)
  Seasonal:
  (1 + 0.989L + 0.327L^2 - 0.463L^3)c_t,  c_t ~ IID(0, 0.0424)

Variances
---------

    trend  seasonal irregular
  0.13135   0.04245   0.13684

Roots
-----

  Component            Root Modulus Argument Period
1     trend  0.7857-0.0000i  0.7857    0.000    Inf
2     trend -1.0000+0.0000i  1.0000    3.142  2.000
3  seasonal  0.4632+0.0000i  0.4632    0.000    Inf
4  seasonal -0.7260+0.6877i  1.0000    2.383  2.636
5  seasonal -0.7260-0.6877i  1.0000    3.900  1.611
```

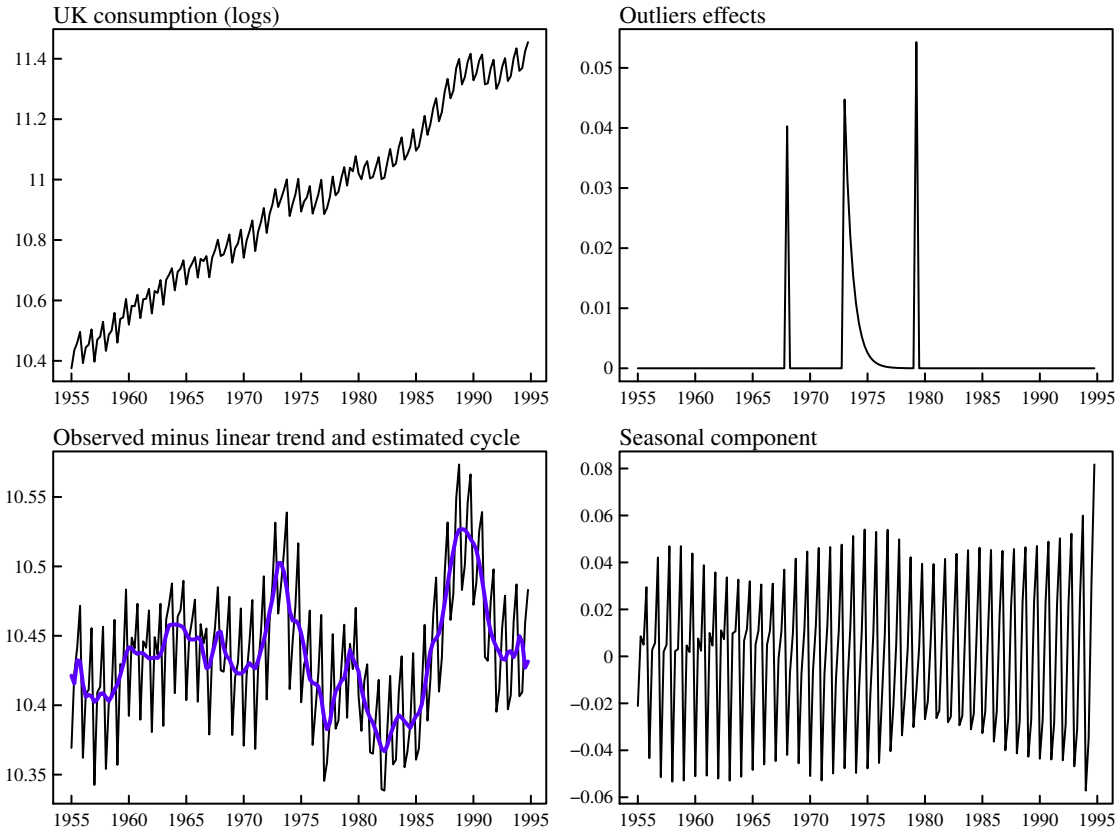
A Change of variable in the ACGF

Setting $z = e^{-i\omega}$ in the expression of the Autocovariance generating function given in equation (4) yields the theoretical spectrum of an ARMA process. As mentioned above, the resulting equation of the ACGF is not a polynomial since $z^j + z^{-j} = 2\cos(\omega j)$ (for $z = e^{-iw}$, which has unit modulus, the inverse $1/z$ is the complex-conjugate of z) and, hence, the variable varies with the order j. Here, I give some details about the change of variable that is applied in order to express the ACGF as a standard polynomial.

Remember that the product of cosines is performed as follows:

$$\cos(a)\cos(b) = \frac{1}{2}\left(\cos(a+b) + \cos(a-b)\right) .$$

Figure 9: Decomposition of the UK consumption time series (logs)



The so-called double-angle formula can be obtained form the product of cosines, taken $a = b = \omega$:

$$\cos(\omega)\cos(\omega) = \frac{1}{2}(\cos(2\omega) + \cos(0)) = \frac{1}{2}(\cos(2\omega) + 1)$$
$$2\cos^2(\omega) = \cos(2\omega) + 1\,.$$

Multiplying this expression by 2, we will use the following result:

$$2\cos(2\omega) = (2\cos(\omega))^2 - 2\,.$$

Next, I will show the transformation that is pursued for different orders of j. Then we will be able to generalize it for any order.

Polynomial of order 2: The goal is to transform the expression:

$$A(2\cos(j\omega)) = a_0 + a_1 2\cos(\omega) + a_2 2\cos(2w)$$

into the polynomial:

$$B(2\cos(\omega)) = b_0 + b_1 2\cos(\omega) + b_2(2\cos(\omega))^2\,.$$

As they are equivalent, we have $B(\cdot) = A(\cdot)$:

$$b_0 + b_1 2\cos(\omega) + b_2(2\cos(\omega))^2 = a_0 + a_1 2\cos(\omega) + a_2 \underbrace{2\cos(2w)}_{(2\cos(\omega))^2 - 2}$$
$$b_0 + b_1 2\cos(\omega) + b_2(2\cos(\omega))^2 = (a_0 - 2a_2) + a_1 2\cos(\omega) + a_2(2\cos(\omega))^2$$

Equating the coefficients related to the same terms gives the mapping of the coefficents from the original polynomial $A(\cdot)$ to the polynomial $B(\cdot)$ in the variable $2\cos(\omega)$:

$$\mathbf{b_0 = a_0 - 2a_2}\,; \quad \mathbf{b_1 = a_1}\,; \quad \mathbf{b_2 = a_2}\,.$$

Polynomial of order 3: Let's denote $x = 2\cos(\omega)$, for short. Transform the expression:

$$A(\cdot) = a_0 + a_1 2\cos(\omega) + a_2 2\cos(2w) + +a_3 2\cos(3w)$$

into the polynomial in the variable $x = 2\cos(\omega)$:

$$B(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3\,.$$

As done before, upon the double-angle formula, we can write $2\cos(2w)$ in terms of $x = 2\cos(\omega)$: $2\cos(2w) = x^2 - 2$. Working upon the expression of $\cos(a)\cos(b)$, we choose a and b so that $\cos(3w)$ shows up on it and then solve for it and multiply the final expression by 2. Taking $a = 2w$ and $b = w$:

$$\begin{aligned}
\cos(2w)\cos(\omega) &= \frac{1}{2}\left(\cos(3w) + \cos(\omega)\right) \\
\left(2\cos^2(w) - 1\right)\cos(\omega) &= \frac{1}{2}\left(\cos(3w) + \cos(\omega)\right) \\
2\cos^3(w) - \cos(\omega) &= \frac{1}{2}\left(\cos(3w) + \cos(\omega)\right) \\
\cos(3w) &= 4\cos^3(w) - 3\cos(\omega)\,.
\end{aligned}$$

Multiplying the expression above by 2 and substituting $x = 2\cos(\omega)$, we arrive to the same result:

$$2\cos(3w) = x^3 - 3x\,.$$

Equating $B(x) = A(\cdot)$:

$$b_0 + b_1 x + b_2 x^2 + b_3 x^3 = a_0 + a_1 \underbrace{2\cos(\omega)}_{x} + a_2 \underbrace{2\cos(2w)}_{x^2-2} + a_3 \underbrace{2\cos(3w)}_{x^3-3x}$$

$$b_0 + b_1 x + b_2 x^2 + b_3 x^3 = (a_0 - 2a_2) + (a_1 - 3a_3)x + a_2 x^2 + a_3 x^3$$

$$\mathbf{b_0 = a_0 - 2a_2}\,; \quad \mathbf{b_1 = a_1 - 3a_3}\,; \quad \mathbf{b_2 = a_2}\,; \quad \mathbf{b_3 = a_3}\,.$$

Polynomial of order 4: Set $a = 3w$ and $b = w$ in $\cos(a)\cos(b)$:

$$\begin{aligned}
\cos(3w)\cos(\omega) &= \frac{1}{2}\left(\cos(4w) + \cos(2w)\right) \\
\left(4\cos^3(w) - 3\cos(\omega)\right)\cos(\omega) &= \frac{1}{2}\left(\cos(4w) + \cos(2w)\right) \\
8\cos^4(w) - 6\cos^2(w) &= \cos(4w) + 2\cos^2(w) - 1 \\
\cos(4w) &= 8\cos^4(w) - 8\cos^2(w) + 1\,.
\end{aligned}$$

Multiplying by 2 and replacing $x = 2\cos(\omega)$:

$$2\cos(4w) = x^4 - 4x^2 + 2\,.$$

Equating $B(x) = A(\cdot)$ and solving fot b_i:

$$b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 = a_0 + a_1 \underbrace{2\cos(\omega)}_{x} + a_2 \underbrace{2\cos(2w)}_{x^2-2} + a_3 \underbrace{2\cos(3w)}_{x^3-3x} + a_4 \underbrace{2\cos(4w)}_{x^4-4x^2+2}$$

$$\mathbf{b_0 = a_0 - 2a_2 + 2a_4}\,; \quad \mathbf{b_1 = a_1 - 3a_3}\,; \quad \mathbf{b_2 = a_2 - 4a_4}\,; \quad \mathbf{b_3 = a_3}\,; \quad \mathbf{b_4 = a_4}\,.$$

Polynomial of order 5: Set $a = 4w$ and $b = w$ in $\cos(a)\cos(b)$:

$$\begin{aligned}
\cos(4w)\cos(\omega) &= \frac{1}{2}\left(\cos(5w) + \cos(3w)\right) \\
\left(8\cos^4(w) - 8\cos^2(w) + 1\right)\cos(\omega) &= \frac{1}{2}\left(\cos(5w) + \cos(3w)\right) \\
16\cos^5(w) - 16\cos^3(w) + 2\cos(\omega) &= \cos(5w) + 4\cos^3(w) - 3\cos(\omega) \\
\cos(5w) &= 16\cos^5(w) - 20\cos^3(w) + 5\cos(\omega)
\end{aligned}$$

Multiplying by 2 and replacing $x = 2\cos(\omega)$:

$$2\cos(5w) = x^5 - 5x^3 + 5x\,.$$

Equating $B(x) = A(\cdot)$ and solving fot b_i:

$$\begin{aligned}
b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 + b_5 x^5 = a_0 + a_1 \underbrace{2\cos(\omega)}_{x} + a_2 \underbrace{2\cos(2w)}_{x^2-2} + a_3 \underbrace{2\cos(3w)}_{x^3-3x} \\
+ a_4 \underbrace{2\cos(4w)}_{x^4-4x^2+2} + a_5 \underbrace{2\cos(5w)}_{x^5-5x^3+5x}
\end{aligned}$$

$$\begin{aligned}
&\mathbf{b_0 = a_0 - 2a_2 + 2a_4}\,; \quad \mathbf{b_1 = a_1 - 3a_3 + 5a_5}\,; \quad \mathbf{b_2 = a_2 - 4a_4}\,; \\
&\mathbf{b_3 = a_3 - 5a_5}\,; \quad \mathbf{b_4 = a_4}\,; \quad \mathbf{b_5 = a_5}\,.
\end{aligned}$$

The following table summarizes the mappings from the coefficients in $A(\cdot)$ to those in $B(\cdot)$ that we found so far:

	degree				
	1	2	3	4	5
$b_0 =$	a_0	$a_0 - 2a_2$	$a_0 - 2a_2$	$a_0 - 2a_2 + 2a_4$	$a_0 - 2a_2 + 2a_4$
$b_1 =$	–	a_1	$a_1 - 3a_3$	$a_1 - 3a_3$	$a_1 - 3a_3 + 5a_5$
$b_2 =$	–	a_2	a_2	$a_2 - 4a_4$	$a_2 - 4a_4$
$b_3 =$	–	–	a_3	a_3	$a_3 - 5a_5$
$b_4 =$	–	–	–	a_4	a_4
$b_5 =$	–	–	–	–	a_5

This generalizes as follows:

$$
\begin{aligned}
b_0 &= a_0 - 2a_2 + 2a_4 - 2a_6 + 2a_8 + \cdots \\
b_1 &= a_1 - 3a_3 + 5a_5 - 7a_7 + 9a_9 + \cdots \\
b_2 &= a_2 - 4a_4 + 9a_6 - 16a_8 + \cdots \\
b_3 &= a_3 - 5a_5 + 14a_7 - 30a_9 + \cdots \\
b_4 &= a_4 - 6a_6 + 20a_8 + \cdots \\
b_5 &= a_5 - 7a_7 + 27a_9 + \cdots \\
b_6 &= a_6 - 8a_8 + \cdots \\
b_7 &= a_7 - 9a_9 + \cdots \\
b_8 &= a_8 + \cdots \\
b_9 &= a_9 + \cdots
\end{aligned}
$$

The following product gives the coefficients of the pursued polynomial in the the variable $x = 2\cos\omega$:

$$
\begin{bmatrix}
1 & 0 & -2 & 0 & 2 & 0 & -2 & 0 & 2 & 0 \\
0 & 1 & 0 & -3 & 0 & 5 & 0 & -7 & 0 & 9 \\
0 & 0 & 1 & 0 & -4 & 0 & 9 & 0 & -16 & 0 \\
0 & 0 & 0 & 1 & 0 & -5 & 0 & 14 & 0 & 30 \\
0 & 0 & 0 & 0 & 1 & 0 & -6 & 0 & 20 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & -7 & 0 & 27 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -8 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -9 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\times
\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \end{bmatrix}
=
\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \end{bmatrix}.
$$

References

Box GEP, Hillmer SC, Tiao GC (1978). "Analysis and Modeling of Seasonal Time Series." In A Zellner (ed.), *Seasonal Analysis of Economic Time Series*, pp. 309–334. U.S. Dept. of Commerce - Bureau of the Census, Washington, D.C. URL http://www.nber.org/chapters/c3904.pdf.

Burman JP (1980). "Seasonal Adjustment by Signal Extraction." *Journal of the Royal Statistical Society. Series A (General)*, **143**(3), 321–337. doi:10.2307/2982132.

Clark PK (1987). "The Cyclical Component of U.S. Economic Activity." *The Quarterly Journal of Economics*, **102**(4), 797–814. doi:10.2307/1884282.

Gómez V (2015). "SSMMATLAB: A Set of MATLAB Programs for the Statistical Analysis of State Space Models." *Journal of Statistical Software*, **66**(1), 1–37. ISSN 1548-7660. doi:10.18637/jss.v066.i09.

Gómez V, Maravall A (2001a). "Programs TRAMO and SEATS. Instructions for the User (Beta Version: June 1997)." *Technical Report SGAPE-97001*, Ministerio de Economía y Hacienda. Dirección General de Análisis y Programación Presupuestaria. URL http://www.bde.es/f/webbde/SES/servicio/Programas_estadisticos_y_econometricos/Programas/ficheros/manualdos.pdf.

Gómez V, Maravall A (2001b). "Seasonal Adjustment and Signal Extraction in Economic Time Series." In D Peña, GC Tiao, RS Tsay (eds.), *A Course in Time Series Analysis*, chapter 8. John Wiley & Sons, Inc. doi:10.1002/9781118032978.ch8.

Hillmer SC, Tiao GC (1982). "An ARIMA-Model-Based Approach to Seasonal Adjustment." *Journal of the American Statistical Association*, **77**(377), 63–70. doi:10.1080/01621459.1982.10477767.

Maravall A, Pierce DA (1987). "A Prototypical Seasonal Adjustment Model." *Journal of Time Series Analysis*, **8**(2), 177–193. ISSN 1467-9892. doi:10.1111/j.1467-9892.1987.tb00431.x.

Planas C (1997). *Applied Time Series Analysis: Modelling, Forecasting, Unobserved Components Analysis and the Wiener-Kolmogorov Filter*. Eurostat: Series E, Methods. Office for Official Publications of the European Communities. ISBN 9789282815724. URL https://bookshop.europa.eu/en/applied-time-series-analysis-pbCA0897484/.

Pollock DSG (1999). *A Handbook of Time-Series Analysis Signal Processing and Dynamics*. Academic Press, London. ISBN 0-12-560990-6. doi:10.1016/B978-012560990-6/50002-6.

R Development Core Team (2016). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.