ALGORITHMS FOR FORECASTING

Recall that the optimal linear forecast for a stationary process is found by solving a least-squares problem: we minimize the mean square (forecast or prediction) error

$$\text{MSE}_n(h) = \mathbb{E}\left[\left(X_{n+h} - \widehat{X}_{n+h}\right)^2\right] \tag{1}$$

for a specified n and $h \geq 1$. We first note the decomposition

$$\begin{aligned}\mathbb{E}\left[\left(X_{n+h} - \widehat{X}_{n+h}\right)^2\right] &= \mathbb{E}\left[\left(X_{n+h} - m_n(h) + m_n(h) - \widehat{X}_{n+h}\right)^2\right] \\ &= \mathbb{E}\left[(X_{n+h} - m_n(h))^2\right] + 2\mathbb{E}\left[(X_{n+h} - m_n(h))(m_n(h) - \widehat{X}_{n+h})\right] + \mathbb{E}\left[(m_n(h) - \widehat{X}_{n+h})^2\right]\end{aligned} \tag{2}$$

where $m_n(h) = \mathbb{E}[X_{n+h}|X_1, \ldots, X_n]$ is the conditional expectation of X_{n+h} given $X_1, \ldots, X_n$. In these three terms, the first is some constant that does not depend on the choice of $\widehat{X}_{n+h}$; the second is zero, as, by iterated expectation

$$\mathbb{E}\left[(X_{n+h} - m_n(h))(m_n(h) - \widehat{X}_{n+h})\right] = \mathbb{E}_{X_{1:n}}\left[\mathbb{E}_{X_{n+h}|X_{1:n}}\left[(X_{n+h} - m_n(h))(m_n(h) - \widehat{X}_{n+h})|X_{1:n}\right]\right]$$

if $X_{1:n} \equiv (X_1, \ldots, X_n)$ – for the inner expectation, we have

$$\mathbb{E}_{X_{n+h}|X_{1:n}}\left[(X_{n+h} - m_n(h))(m_n(h) - \widehat{X}_{n+h})|X_{1:n}\right] = (m_n(h) - \widehat{X}_{n+h})\mathbb{E}_{X_{n+h}|X_{1:n}}\left[(X_{n+h} - m_n(h))|X_{1:n}\right]$$

as both $m_n(h)$ and $\widehat{X}_{n+h}$ are functions of $X_{1:n}$, and $\mathbb{E}_{X_{n+h}|X_{1:n}}\left[(X_{n+h} - m_n(h))|X_{1:n}\right] = 0$. Therefore we mininmize (1) by minimizing the third term in (2): this can be achieved by setting

$$\widehat{X}_{n+h} = m_n(h) = \mathbb{E}[X_{n+h}|X_1, \ldots, X_n]$$

in which case

$$\text{MSE}_n(h) = \mathbb{E}\left[(X_{n+h} - \mathbb{E}[X_{n+h}|X_1, \ldots, X_n])^2\right]. \tag{3}$$

Using a linear representation for $m_n(h)$, we can get a more explicit result

$$\widehat{X}_{n+h} = a_0 + \sum_{i=1}^{n} a_i X_{n-i+1}.$$

This reduces to choosing the coefficients $a = (a_1, \ldots, a_n)^\top$. We minimize using calculus: we have

$$\frac{\partial \text{MSE}_n(h)}{\partial a_0} = \mathbb{E}\left[\left(X_{n+h} - a_0 - \sum_{i=1}^{n} a_i X_{n-i+1}\right)\right] \tag{4}$$

$$\frac{\partial \text{MSE}_n(h)}{\partial a_j} = \mathbb{E}\left[\left(X_{n+h} - a_0 - \sum_{i=1}^{n} a_i X_{n-i+1}\right) X_{n-j+1}\right], j = 1, \ldots, n. \tag{5}$$

allowing for differentiation under the expectation. We must solve after setting (4) and (5) equal to zero simultaneously. From equation (4), we have that

$$\mathbb{E}[X_{n+h}] - a_0 - \sum_{i=1}^{n} a_i \mathbb{E}[X_{n-i+1}] = 0$$

and if $\{X_t\}$ is stationary with mean μ, we may conclude that the optimal choice must satisfy

$$a_0 = \mu\left(1 - \sum_{i=1}^{n} a_i\right).$$

From (5), we have for each $j = 1, \ldots, n$,

$$\mathbb{E}[X_{n+h}X_{n-j+1}] - a_0\mathbb{E}[X_{n-j+1}] - \sum_{i=1}^{n} a_i \mathbb{E}[X_{n-i+1}X_{n-j+1}] = 0.$$

Noting that from the earlier result, we have

$$a_0\mathbb{E}[X_{n-j+1}] = \mu^2 \left(1 - \sum_{i=1}^{n} a_i\right)$$

and hence, rearranging

$$\mathbb{E}[X_{n+h}X_{n-j+1}] - \mu^2 - \sum_{i=1}^{n} a_i(\mathbb{E}[X_{n-i+1}X_{n-j+1}] - \mu^2) = 0.$$

so that

$$\sum_{i=1}^{n} a_i \text{Cov}[X_{n-i+1}, X_{n-j+1}] = \text{Cov}[X_{n+h}, X_{n-j+1}]$$

or equivalently

$$\sum_{i=1}^{n} a_i \gamma_X(i-j) = \gamma_X(j+h-1) \qquad j = 1, \ldots, n.$$

Stacking these n equations, we are left with

$$\begin{bmatrix} \gamma_X(0) & \gamma_X(1) & \gamma_X(2) & \cdots & \gamma_X(n-1) \\ \gamma_X(1) & \gamma_X(0) & \gamma_X(1) & \cdots & \gamma_X(n-2) \\ \gamma_X(2) & \gamma_X(1) & \gamma_X(0) & \cdots & \gamma_X(n-2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_X(n-1) & \gamma_X(n-2) & \gamma_X(n-3) & \cdots & \gamma_X(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \gamma_X(h) \\ \gamma_X(h+1) \\ \gamma_X(h+2) \\ \vdots \\ \gamma_X(h+n-1) \end{bmatrix}$$

or equivalently

$$\Gamma_X(n)\boldsymbol{a} = \boldsymbol{\gamma}_X(n,h) \tag{6}$$

say, where

$$\boldsymbol{\gamma}_X(n,h) = (\gamma_X(h), \ldots, \gamma_X(h+n-1))^\top.$$

Thus provided $\Gamma_X(n)$ is non-singular, we have that the optimal $\boldsymbol{a}$ is the solution to (6)

$$\boldsymbol{a}(n,h) = \{\Gamma_X(n)\}^{-1}\boldsymbol{\gamma}_X(n,h)$$

which requires us to compute $\{\Gamma_X(n)\}^{-1}$.

The Levinson-Durbin Algorithm: Without loss of generality, take $\mathbb{E}[X_t] = 0$. Consider forecasting for $h = 1$, and write

$$\widehat{X}_{n+1} = \sum_{i=1}^{n} \phi_{n,i} X_{n-i+1} = \boldsymbol{\phi}_n^\top X_{n:1}$$

where $\boldsymbol{\phi}_n = (\phi_{n,1}, \cdots, \phi_{n,n})^\top$, which corresponds in our prediction forecast to $\boldsymbol{a}(n, 1)$. The optimal choice solves the linear system

$$\boldsymbol{\Gamma}_X(n)\boldsymbol{\phi}_n = \boldsymbol{\gamma}_X(n, 1),$$

which is an $n \times 1$ system of equations, yielding minimum MSE

$$v_n = \gamma_X(0) - \boldsymbol{\phi}_n^\top \boldsymbol{\gamma}_X(n, 1)$$

The Levinson-Durbin recursion solves the equation without matrix inversion, in order n^2 operations (most matrix inversion procedures are order n^3) by exploiting the Toeplitz structure of $\Gamma_X(n)$. One can also use singular value, eigenvalue decomposition or Cholesky decomposition; these method have special implementation for positive-definite Toeplitz structure matrices.

Initialization: For $n = 1$, set

$$\phi_{1,1} = \frac{\gamma_X(1)}{\gamma_X(0)} = \rho_X(1) \qquad v_0 = \gamma_X(0) \qquad v_1 = \gamma_X(0)(1 - \phi_{1,1}^2).$$

Recursion: Define $\boldsymbol{\phi}_n$ in terms of $\boldsymbol{\phi}_{n-1}$ and v_{n-1} as follows:

$$\phi_{n,n} = \frac{1}{v_{n-1}} \left[\gamma_X(n) - \boldsymbol{\phi}_{n-1}^\top \boldsymbol{\gamma}_X^{\mathrm{R}}(n-1, 1)\right]$$

where $\boldsymbol{\gamma}_X^{\mathrm{R}}(n-1, 1) = (\gamma_X(n-1), \gamma_X(n-1), \ldots, \gamma_X(1))^\top$ (the superscript 'R' means 'reverse') and

$$\boldsymbol{\phi}_{n,1:(n-1)} = \boldsymbol{\phi}_{n-1} - \phi_{n,n}\boldsymbol{\phi}_{n-1}^{\mathrm{R}}$$

where $\boldsymbol{\phi}_{n-1}$ and $\boldsymbol{\phi}_{n-1}^{\mathrm{R}}$ are $(n-1) \times 1$ vectors with $\boldsymbol{\phi}_{n-1}^{\mathrm{R}} = (\phi_{n-1,n-1}, \phi_{n-1,n-2}, \cdots, \phi_{n-1,1})^\top$, and $\phi_{n,n}$ is a scalar Then set

$$v_n = v_{n-1}(1 - \phi_{n,n}^2).$$

Proof (NOT EXAMINABLE)
To see why this works, let

$$\mathbf{P}_n = \frac{1}{\gamma_X(0)} \boldsymbol{\Gamma}_X(n)$$

be the autocorrelation matrix and let

$$\boldsymbol{\rho}_{1:n} = \frac{1}{\gamma_X(0)} (\gamma_X(1), \ldots, \gamma_X(n))^\top \qquad \boldsymbol{\rho}_{n:1} = \boldsymbol{\rho}_{1:n}^{\mathrm{R}} = (\rho_X(n), \cdots, \rho_X(1))^\top.$$

The recursion computes the inverse of $\boldsymbol{P}_{n+1}$ from the inverse of $\boldsymbol{P}_n$ using rank-one operations. Note first that

$$\boldsymbol{\Gamma}_X(n)\boldsymbol{\phi}_n = \boldsymbol{\gamma}_X(n, 1) \Longrightarrow \mathbf{P}_n\boldsymbol{\phi}_n = \boldsymbol{\rho}_{1:n}.$$

We need to verify that

$$\mathbf{P}_n\boldsymbol{\phi}_n = \boldsymbol{\rho}_{1:n} \Longrightarrow \mathbf{P}_{n+1}\boldsymbol{\phi}_{n+1} = \boldsymbol{\rho}_{1:(n+1)}.$$

For $n = 1$, $\mathbf{P}_1 = 1, \boldsymbol{\phi}_1 = \phi_{1,1} = \rho(1) = \boldsymbol{\rho}_{1,1}$. We establish the result by induction on n. Assume that $\mathbf{P}_k\boldsymbol{\phi}_k = \boldsymbol{\rho}_{1:k}$. Now

$$\mathbf{P}_{k+1} = \begin{bmatrix} \mathbf{P}_k & \boldsymbol{\rho}_{k:1} \\ \boldsymbol{\rho}_{k:1}^\top & 1 \end{bmatrix}$$

and $\mathbf{P}_k\boldsymbol{\phi}_{\boldsymbol{k}} = \boldsymbol{\rho}_{1:k}$, therefore

$$\mathbf{P}_k\boldsymbol{\phi}_k^{\mathrm{R}} = \boldsymbol{\rho}_{k:1}$$

using the symmetry of $\mathbf{P}_k$. In the proposed recursion, we have that

$$\mathbf{P}_{k+1}\boldsymbol{\phi}_{k+1} = \begin{bmatrix} \mathbf{P}_k & \boldsymbol{\rho}_{k:1} \\ \boldsymbol{\rho}_{k:1}^\top & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi}_k - \phi_{k+1,k+1}\boldsymbol{\phi}_k^{\mathrm{R}} \\ \phi_{k+1,k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}$$

where $\mathbf{a}$ is equal to a $k \times 1$ system of the form

$$\mathbf{a} = \mathbf{P}_k\boldsymbol{\phi}_k - \phi_{k+1,k+1}\mathbf{P}_k\boldsymbol{\phi}_k^{\mathrm{R}} + \phi_{k+1,k+1}\boldsymbol{\rho}_{k:1} = \mathbf{P}_k\boldsymbol{\phi}_k = \boldsymbol{\rho}_{1:k}$$

by the inductive hypothesis. For $\mathbf{b}$, we have

$$\mathbf{b} = \boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k - \phi_{k+1,k+1}\boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k^{\mathrm{R}} + \phi_{k+1,k+1} = \boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k - \phi_{k+1,k+1}(1 - \boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k^{\mathrm{R}})$$

We have now

$$\phi_{k+1,k+1} = \frac{1}{v_k}\left[\gamma_X(k+1) - \boldsymbol{\phi}_k^\top\boldsymbol{\gamma}_X^{\mathrm{R}}(k,1)\right]$$

where

$$v_k = \gamma_X(0) - \boldsymbol{\phi}_k^\top\boldsymbol{\gamma}_X(k,1) = \gamma_X(0)\left(1 - \boldsymbol{\phi}_k^\top\boldsymbol{\rho}_{1:k}\right) = \gamma_X(0)(1 - \boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k^{\mathrm{R}})$$

From the above formula, we have

$$\phi_{k+1,k+1} = \frac{1}{\gamma_X(0)(1 - \boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k^{\mathrm{R}})}\left[\gamma_X(k+1) - \boldsymbol{\phi}_k^\top\gamma_X^{\mathrm{R}}(k,1)\right] \tag{7}$$

changing the order of the inner product and from $\mathbf{b}$, we have

$$\begin{aligned} &\boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k + \frac{\left[\gamma_X(k+1) - \boldsymbol{\phi}_k^\top\gamma_X^{\mathrm{R}}(k,1)\right]}{\gamma_X(0)(1 - \boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k^{\mathrm{R}})}(1 - \boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k^{\mathrm{R}}) \\ &= \boldsymbol{\rho}_{k:1}^\top\boldsymbol{\phi}_k + \rho_X(k+1) - \boldsymbol{\phi}_k^\top\boldsymbol{\rho}_{k:1} \\ &= \rho_X(k+1) \end{aligned}$$

as required. Therefore the required recursion is valid by the inductive argument. ■

We have effectively performed a matrix inversion by considering a block decomposition

$$\mathbf{P}_{k+1} = \begin{bmatrix} \mathbf{P}_k & \boldsymbol{\rho}_{k:1} \\ \boldsymbol{\rho}_{k:1}^\top & 1 \end{bmatrix}$$

and then computed $\mathbf{P}_{k+1}^{-1}$ using $\mathbf{P}_k^{-1}$. This works because of the nature of the nature of the matrix (Toeplitz structure). For v_n, we have

$$\begin{aligned} v_n = \mathbb{E}[(X_{n+1} - \boldsymbol{\phi}_n^\top X_{n:1})^2] &= \gamma_X(0) - \boldsymbol{\phi}_n^\top\gamma_X(n-1,1) \\ &= \gamma_X(0) - \boldsymbol{\phi}_{n-1}^\top\boldsymbol{\gamma}_X(n-1,1) + \phi_{n,n}(\boldsymbol{\phi}_{n-1}^{\mathrm{R}})^\top\boldsymbol{\gamma}_X(n-1,1) - \phi_{n,n}\gamma_X(n) \\ &= v_{n-1} + \phi_{n,n}\big((\boldsymbol{\phi}_{n-1}^{\mathrm{R}})^\top\boldsymbol{\gamma}_X(n-1,1) - \gamma_X(n)\big) \\ &= v_{n-1} - \phi_{n,n}^2\big(\gamma_X(0) - (\boldsymbol{\phi}_{n-1}^{\mathrm{R}})^\top\boldsymbol{\gamma}_X(n-1,1)\big) \\ &= v_{n-1} - \phi_{n,n}^2 v_{n-1} \\ &= v_{n-1}(1 - \phi_{n,n}^2) \end{aligned}$$

plugging in (7). That is, $v_n = v_{n-1}(1 - \phi_{n,n}^2)$ and as $v_n \geq 0$, imply $\phi_{n,n}^2 < 1, |\phi_{n,n}| < 1$ thus $v_n \leq v_{n-1}$, *i.e.* the $\{v_n\}$ form a non-increasing sequence.

Partial autocorrelation: Using the Levinson-Durbin algorithm, define the function $\pi_X(h)$ by

$$\pi_X(h) = \begin{cases} 1 & \text{if } h = 0 \\ \phi_{h,h} & \text{if } h \geq 1 \end{cases}$$

for $h = 0, 1, 2, \ldots$ This $\pi_X(h)$ is the *partial autocorrelation function* (pacf). For random variables X, Y, Z, the *partial correlation* is defined by

$$\text{Corr}[(X - \mathbb{E}[X|Z]), (Y - \mathbb{E}[Y|Z])];$$

this is the correlation between residuals obtained by regressing in turn X and on Z and Y on Z. The pacf computes the autocorrelation between

$$X_t - \mathbb{E}[X_t|X_{(t+1):(t+h-1)}] \qquad \text{and} \qquad X_{t+h} - \mathbb{E}[X_{t+h}|X_{(t+1):(t+h-1)}]$$

Consider a causal stationary AR(p) process; for $h > p$,

$$\mathbb{E}[X_{t+h}|X_{(t+1):(t+h-1)}] = \phi_1 X_{t+h-1} + \cdots + \phi_p X_{t+h-p} \qquad \mathbb{E}[X_t|X_{(t+1):(t+h-1)}] = f(X_{t+1}, \ldots, X_{t+h-1})$$

for some function $f()$, where by definition $X_{t+h} - \mathbb{E}[X_{t+h}|X_{(t+1):(t+h-1)}] = Z_{t+h}$, therefore

$$\text{Corr}[X_t - \mathbb{E}[X_t|X_{(t+1):(t+h-1)}], X_{t+h} - \mathbb{E}[X_{t+h}|X_{(t+1):(t+h-1)}]] = \text{Corr}[X_t - \mathbb{E}[X_t|X_{(t+1):(t+h-1)}], Z_{t+h}] = 0$$

as $\{X_t\}$ is causal so $X_t - \mathbb{E}[X_t|X_{(t+1):(t+h-1)}]$ depends on $\{Z_s, s \leq t + h - 1\}$ only. Thus $\pi_X(h) = 0$ for $h > p$, while for $h \leq p, \pi_X(h) \neq 0$. That is, we can diagnose an AR(p) structure by inspecting whether the partial autocorrelation drops to zero at some finite lag.

EXAMPLE Consider the AR(3) process

$$X_t = -1.4X_{t-1} - 0.48X_{t-2} + 0.16X_{t-3} + Z_t$$

or equivalently

$$(1 + 1.4\text{B} + 0.48\text{B}^2 - 0.16\text{B}^3)X_t = Z_t.$$

or $\Phi(B)X_t = Z_t$ with $\Phi(z) = (1 + 1.4z + 0.48z^2 - 0.16z^3)$. We have the factorization

$$\Phi(z) = (1 - 0.2z)(1 + 1.6z + 0.8z^2) = (1 - \alpha_1 z)(1 - \alpha_2 z)(1 - \alpha_3 z)$$

and first reciprocal root is $\alpha_1 = 0.2$. The two remaining roots are

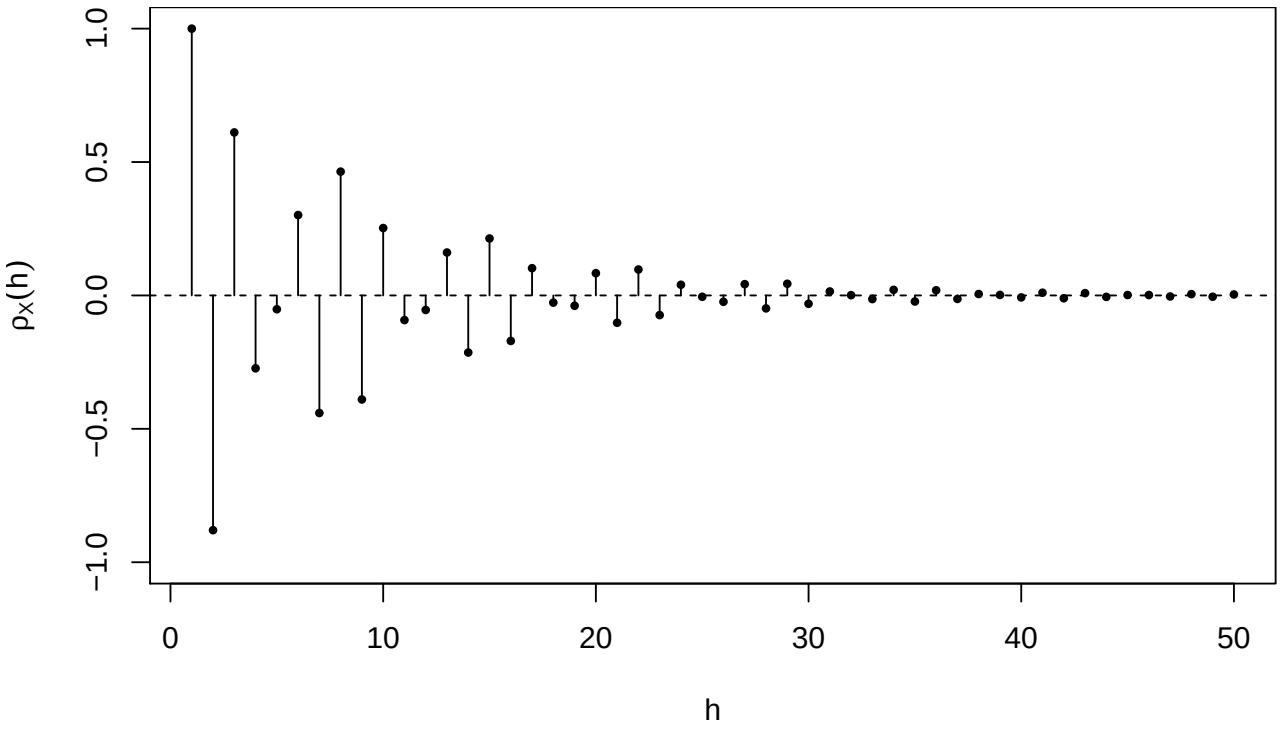
$$\frac{-1.6 \pm \sqrt{2.56 - 3.2}}{2} = \frac{-1.6 \pm \sqrt{-0.64}}{2} = -0.8 \pm 0.4i$$

so the reciprocal roots are $\alpha_2 = -0.8 + 0.4i$ and $\alpha_3 = -0.8 - 0.4i$. All the reciprocal roots are less than one in modulus, so the process is stationary and causal.

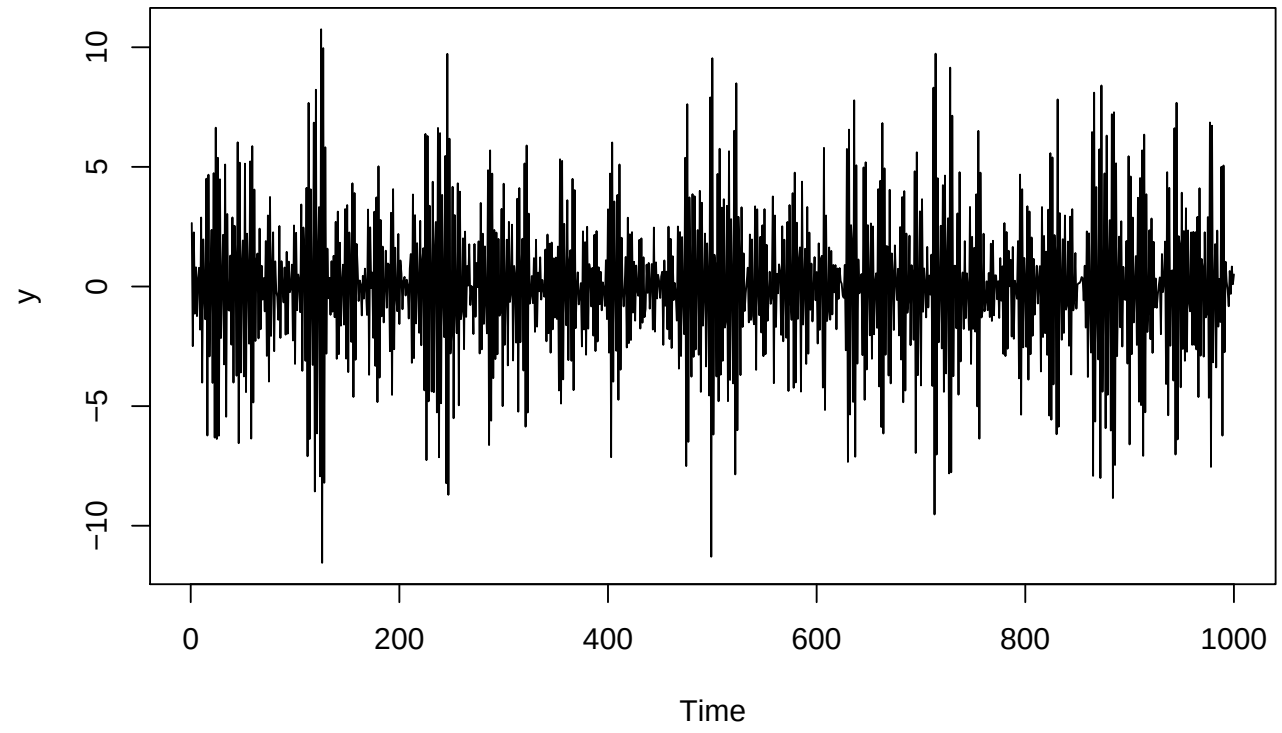
```
alpha.vals<-c(0.2,-0.8+0.4i,-0.8-0.4i)
ar.coefs<-rep(1,length(alpha.vals)+1)
for(j in 1:length(alpha.vals)){
        ar.coefs.old<-ar.coefs
        if(j > 1){
                for(k in 2:j){
                        ar.coefs[k]<-ar.coefs.old[k]-alpha.vals[j]*ar.coefs.old[k-1]
                }
        }
        ar.coefs[j+1]<--alpha.vals[j]*ar.coefs.old[j]
}
ar.coefs   #These are the AR coefficients

+ [1]  1.00+0i  1.40+0i  0.48+0i -0.16+0i

n<-200
rho.fn1<-ARMAacf(ar=Re(-ar.coefs[2:4]), lag.max = n+1)  #Simulate data
par(mar=c(4,4,0,1))
plot(rho.fn1[1:50],type="n",ylim=range(-1,1),xlab="h",ylab=expression(rho[X](h)))
abline(h=0,lty=2)
for(h in 1:50){lines(c(h,h),c(0,rho.fn1[h]))}
points(rho.fn1[1:50],pch=19,cex=0.5)
```




```
set.seed(26)
y<-arima.sim(n=1000,model=list(ar=Re(-ar.coefs[2:4])))
par(mar=c(4,4,0,1))
plot(y)
```




```
#####################################################
#Levinson-Durbin
psi.Vals<-ARMAtoMA(ar=Re(-ar.coefs[2:4]),lag.max=10000)
Variance<-1+sum(psi.Vals^2)
```

```
gamma.fn1<-Variance*rho.fn1[-1]
Phi.mat<-matrix(0,nrow=n,ncol=n)
v<-rep(0,n+1)
v[0+1]<-Variance
Phi.mat[1,1]<-gamma.fn1[1]/Variance  #Initialize
v[1+1]<-Variance*(1-Phi.mat[1,1]^2)
for(j in 2:n){   #Recursion
        Phi.mat[j,j]<-(gamma.fn1[j]-sum(Phi.mat[j-1,1:(j-1)] * rev(gamma.fn1[1:(j-1)])))/v[(j-1)+1]
        Phi.mat[j,1:(j-1)]<- Phi.mat[j-1,1:(j-1)] - Phi.mat[j,j] * rev(Phi.mat[j-1,1:(j-1)])
        v[j+1]<-v[(j-1)+1]*(1-Phi.mat[j,j]^2)
}
round(Phi.mat[1:8,1:8],10)

+            [,1]       [,2] [,3] [,4] [,5] [,6] [,7] [,8]
+ [1,] -0.8798856  0.0000000 0.00    0    0    0    0    0
+ [2,] -1.5155993 -0.7224959 0.00    0    0    0    0    0
+ [3,] -1.4000000 -0.4800000 0.16    0    0    0    0    0
+ [4,] -1.4000000 -0.4800000 0.16    0    0    0    0    0
+ [5,] -1.4000000 -0.4800000 0.16    0    0    0    0    0
+ [6,] -1.4000000 -0.4800000 0.16    0    0    0    0    0
+ [7,] -1.4000000 -0.4800000 0.16    0    0    0    0    0
+ [8,] -1.4000000 -0.4800000 0.16    0    0    0    0    0

diag(Phi.mat)[1:10]   #pacfs

+  [1] -8.798856e-01 -7.224959e-01  1.600000e-01  1.776357e-15  4.440892e-16
+  [6] -8.881784e-16 -8.881784e-16 -8.881784e-16 -4.440892e-16  0.000000e+00

ARMAacf(ar=Re(-ar.coefs[2:4]), lag.max = 10, pacf=TRUE) #Check

+  [1] -8.798856e-01 -7.224959e-01  1.600000e-01  1.847383e-15  1.696718e-17
+  [6] -6.351014e-16 -1.863144e-16 -8.370233e-16 -1.126903e-15 -2.665510e-16

v[1:20]

+  [1] 9.508426 2.147015 1.026273 1.000000 1.000000 1.000000 1.000000
+  [8] 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
+ [15] 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
```

The matrix `Phi.mat` contains the results of the Levinson-Durbin algorithm, and its diagonal elements are the pacf values match those computed by the `ARMAacf` function. Note that the pacf values are

$$\pi_X(h) = \begin{cases} 1 & h = 0 \\ -0.8799 & h = 1 \\ -0.7225 & h = 2 \\ 0.16 & h = 3 \\ 0 & h \geq 4 \end{cases}$$

Also, note that the `Phi.mat` matrix has identical rows after the third. This is because for an AR(3) model, the optimal prediction is always the conditional mean. Note also that the prediction variance converges to a constant.

The Innovations Algorithm: Let $\{X_t\}$ be zero-mean process with $\mathbb{E}[X_t^2] < \infty$ and let $\gamma_X(t,s) = \mathbb{E}[X_t X_s]$. Let

$$\widehat{X}_n = \begin{cases} 0 & \text{if } n = 1 \\ \mathbb{E}[X_n | \mathbf{X}_{1:(n-1)}] & \text{if } n \geq 2 \end{cases}$$

Let $U_n = X_n - \widehat{X}_n$ be the error in prediction and let $\boldsymbol{U}_{1:n}$ be the vector of prediction errors (computed in a one-step ahead fashion). We assume that $\mathbb{E}[X_n | \mathbf{X}_{1:(n-1)}]$ is linear in $X_1, \ldots, X_{n-1}$, and write $\boldsymbol{U}_{1:n} = \boldsymbol{A}_n \mathbf{X}_{1:n}$ where $\boldsymbol{A}_n$ is given by

$$\boldsymbol{A}_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ a_{1,1} & 1 & 0 & 0 & 0 \\ a_{2,2} & a_{2,1} & 1 & 0 & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n-1,n-1} & \cdots & \cdots & a_{n-1,1} & 1 \end{bmatrix}$$

as

$$\begin{aligned} U_1 &= X_1 \\ U_2 &= X_2 - \widehat{X}_2 = X_2 + a_{1,1} X_1 \\ U_3 &= X_3 - \widehat{X}_3 = X_3 + a_{2,2} X_2 + a_{2,1} X_1 \\ &\cdots \end{aligned}$$

Let $\boldsymbol{C}_n = \boldsymbol{A}_n^{-1}$, again a lower triangular matrix, where

$$\boldsymbol{C}_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \theta_{1,1} & 1 & 0 & 0 & 0 \\ \theta_{2,2} & \theta_{2,1} & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \theta_{n-1,n-1} & \cdots & \cdots & \theta_{n-1,1} & 1 \end{bmatrix}$$

Then $\mathbf{X}_{1:n} = \boldsymbol{C}_n \boldsymbol{U}_{1:n} = \boldsymbol{C}_n (\mathbf{X}_{1:n} - \widehat{\mathbf{X}}_{1:n})$, and

$$\widehat{\mathbf{X}}_{1:n} = \mathbf{X}_{1:n} - \boldsymbol{U}_{1:n} = \boldsymbol{C}_n \boldsymbol{U}_{1:n} - \boldsymbol{U}_{1:n} = (\boldsymbol{C}_n - \mathbf{I}_n) \boldsymbol{U}_{1:n} = (\boldsymbol{C}_n - \mathbf{I}_n)(\mathbf{X}_{1:n} - \widehat{\mathbf{X}}_{1:n}) = \boldsymbol{\Theta}_n (\mathbf{X}_{1:n} - \widehat{\mathbf{X}}_{1:n})$$

where

$$\boldsymbol{\Theta}_n = \boldsymbol{C}_n - \mathbf{I}_n = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ \theta_{1,1} & 0 & 0 & 0 & 0 \\ \theta_{2,2} & \theta_{2,1} & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \theta_{n-1,n-1} & \cdots & \cdots & \theta_{n-1,1} & 0 \end{bmatrix}$$

and so

$$\widehat{X}_{n+1} = \begin{cases} 0 & \text{if } n = 1 \\ \sum_{i=1}^{n} \theta_{n,i} (X_{n+1-i} - \widehat{X}_{n+1-i}) & \text{if } n \geq 2 \end{cases}$$

i.e. $\widehat{X}_{n+1}$ is a linear combination of $X_1 - \widehat{X}_1$, $X_2 - \widehat{X}_2$, …, $X_n - \widehat{X}_n$, the one-step ahead prediction errors which are *uncorrelated.*.

For the general (non-stationary) case:

- Initialize

$$v_0 = \widetilde{\gamma}_X(1,1) = \mathbb{E}[X_1^2]$$

- Recursion at step $i = 1, 2, 3, \ldots$:

$$\theta_{i,i-k} = \frac{1}{v_k}\left[\widetilde{\gamma}_X(i+1,k+1) - \sum_{j=0}^{k-1}\theta_{k,k-j}\theta_{i,i-j}v_j\right]$$

for $0 \leq k < i$

$$v_i = \widetilde{\gamma}_X(i+1,i+1) - \sum_{j=0}^{i-1}\theta_{i,i-j}^2 v_j$$

Compute: $v_0, \theta_{1,1}, v_1, \theta_{2,2}, \theta_{2,1}, v_2, \theta_{3,3}, \theta_{3,2}, \theta_{3,1}, v_3, \ldots$.

For the stationary case:

- Initialize

$$v_0 = \gamma_X(0) = \mathbb{E}[X_1^2]$$

- Recursion at step $i = 1, 2, 3, \ldots$:

$$\theta_{i,i-k} = \frac{1}{v_k}\left[\gamma_X(k-i) - \sum_{j=0}^{k-1}\theta_{k,k-j}\theta_{i,i-j}v_j\right]$$

for $0 \leq k < i$

$$v_i = \gamma_X(0) - \sum_{j=0}^{i-1}\theta_{i,i-j}^2 v_j$$

Compute: $v_0, \theta_{1,1}, v_1, \theta_{2,2}, \theta_{2,1}, v_2, \theta_{3,3}, \theta_{3,2}, \theta_{3,1}, v_3, \ldots$.

```
Gamma.X<-Variance*ARMAacf(ar=Re(-ar.coefs[2:4]), lag.max = n+1)
v<-rep(0,n+1)
v[0+1]<-Gamma.X[0+1]                                    #Index 0
Theta.mat<-matrix(0,n+1,n+1)
Theta.mat[1,1]<-Gamma.X[1+1]/v[0+1]                     #Index i=1
v[1+1]<-Gamma.X[0+1]-Theta.mat[1,1]^2*v[0+1]
for(i in 2:n){                                              #Index i=2,3,...
    for(k in 0:(i-1)){
        jvals<-0:(k-1)
        sumth<-sum(Theta.mat[k,k-jvals]*Theta.mat[i,i-jvals]*v[jvals+1])
        Theta.mat[i,i-k] = (Gamma.X[(i-k)+1] - sumth)/v[k+1]
     }
     jvals<-0:(i-1)
     v[i+1]<-(Gamma.X[0+1] - sum(Theta.mat[i,i-jvals]*Theta.mat[i,i-jvals]*v[jvals+1]))
}
v[1:20]

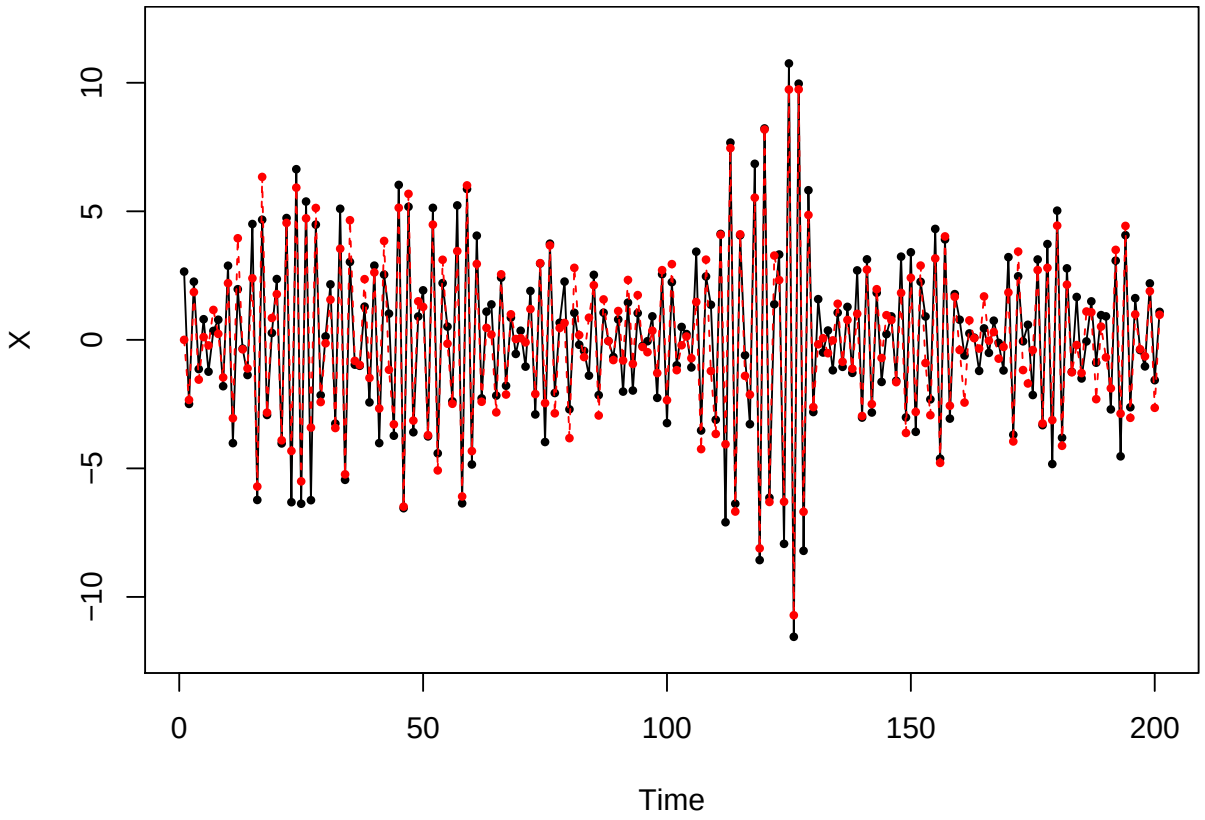
+  [1] 9.508426 2.147015 1.026273 1.000000 1.000000 1.000000 1.000000
+  [8] 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
+ [15] 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000

Theta.mat[1:5,1:5]

+             [,1]      [,2]       [,3]        [,4]     [,5]
+ [1,] -0.8798856 0.0000000  0.0000000  0.00000000 0.000000
+ [2,] -1.5155993 0.6110582  0.0000000  0.00000000 0.000000
+ [3,] -1.4000000 1.6418391 -0.2731363  0.00000000 0.000000
+ [4,] -1.4000000 1.4800000 -1.4110870 -0.05169876 0.000000
+ [5,] -1.4000000 1.4800000 -1.2400000  0.94494319 0.301253
```

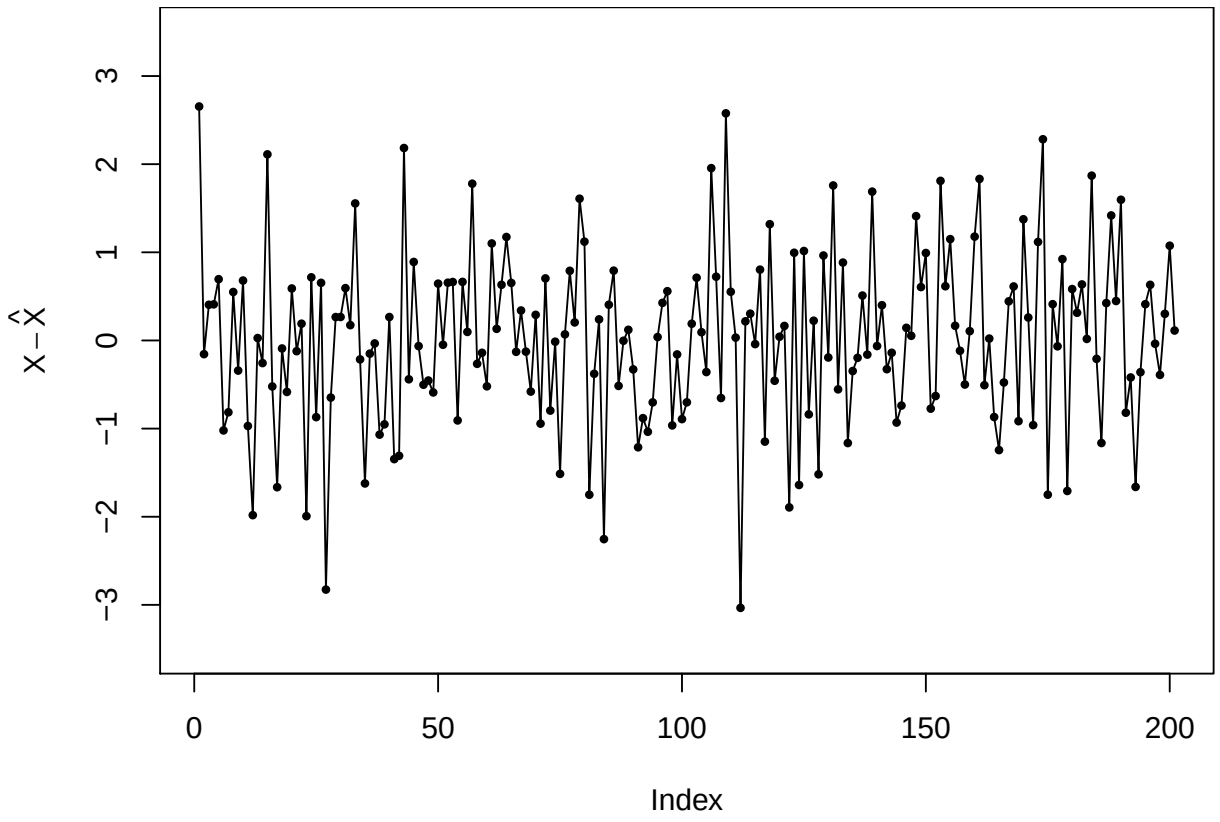
For a real series, we can see how the forecast behaves:

```
set.seed(26)
n<-200
X<-arima.sim(n=n+1,model=list(ar=Re(-ar.coefs[2:4])))
Xhat<-rep(0,n+1)
for(ni in 1:n){
    Xhat[ni+1]<-Theta.mat[ni,1:ni] %*% rev(X[1:ni] - Xhat[1:ni])
}
par(mar=c(4,6,0,1))
plot(X,type='l',ylim=range(-12,12))
points(X,pch=19,cex=0.5)
lines(Xhat,type='l',lty=2,col='red')
points(Xhat,pch=19,cex=0.5,col='red')
```



The one-step ahead forecasts seem largely to be accurate.

```
par(mar=c(4,6,0,1))
plot(as.numeric(X-Xhat),type='l',ylim=range(-3.5,3.5),ylab=expression(X-hat(X)))
points(as.numeric(X-Xhat),pch=19,cex=0.5)
```




```
acf(as.numeric(X-Xhat),lag.max=20,ylim=range(-1,1))
```

