Time Series: Homework 5

Martín Prado

November 4, 2024

Universidad de los Andes – Bogotá Colombia

Exercise 5.8.

The values **.644**, **– .442**, **– .919**, **– 1.573**, **.852**, **– .907**, **.686**, **– .753**, **– .954**, **.576**, are simulated values of $X_1, \ldots, X_{10}$ where $\{X_t\}$ is the ARMA(2, 1) process,

$$X_t - .1X_{t-1} - .12X_{t-2} = Z_t - .7Z_{t-1}, \qquad \{Z_t\} \sim \text{WN}(0, 1).$$

(a) Compute the forecasts $P_{10}X_{11}$, $P_{10}X_{12}$ and $P_{10}X_{13}$ and the corresponding mean squared errors.

(b) Assuming that $Z_t \sim N(0, 1)$, construct 95 % prediction bounds for X_{11}, X_{12} and X_{13}. $\tilde{X}_{11}^T$, $\tilde{X}_{12}^T$ and $\tilde{X}_{13}^T$ and compare

(c) Using the method of Problem 5.1S, compute these values with those obtained in (a). [The simulated values of X_{11}, X_{12} and X_{13}, were in fact .074, 1.097 and -.187 respectively.]

Preliminaries

For every causal **ARMA**(p, q) process with $\phi(B)X_t = \theta(B)Z_t$ we have that the autocovariance function is

$$\gamma(k) = \sigma^2 \sum_{n=0}^{\infty} \psi_n \psi_{n+k},$$

where ψ_k is the k-th coefficient of the Taylor series expansion of $\theta(z)/phi(z)$. Since θ and ϕ are polynomials with no common roots on $|z| < 1$, we can decompose ψ in partial fractions

$$\psi(z) = \frac{a_1}{(1 - r_1 z)^{k_1}} + \cdots + \frac{a_m}{(1 - r_m z)^{k_m}}$$

with $|r_j| < 1$, and thus, there's a Taylor series expansion for each partial fraction with radius of convergence greater or equal than 1:

$$\begin{aligned}
\frac{a}{(1-rz)^k} &= \frac{a}{(k-1)!r^{k-1}} \frac{d^{k-1}}{dz^{k-1}} \left(\frac{1}{1-rz} \right) \\
(|z|<1) &= \frac{a}{(k-1)!r^{k-1}} \sum_{n=k-1}^{\infty} \frac{n!}{(n-k+1)!} r^n z^{n-k+1} \\
&= \sum_{n=k-1}^{\infty} \binom{n}{k-1} a(rz)^{n-k+1} \\
&= \sum_{n=0}^{\infty} \binom{n+k-1}{k-1} a r^n z^n.
\end{aligned}$$

Hence,

$$\psi_n = \sum_{j=1}^{m} \binom{n+k_j-1}{k_j-1} a_j r_j^n.$$

For our case, the partial fraction expansion of ψ is the following

$$\psi(z) = \frac{1-.7z}{1-.1z-.12z^2} = \frac{-3/7}{1-2z/5} + \frac{10/7}{1+3z/10}.$$

Therefore,

$$\psi_n = -\frac{3}{7}\left(\frac{2}{5}\right)^n + \frac{10}{7}\left(\frac{-3}{10}\right)^n = \frac{-3\cdot 4^n + 10\cdot(-3)^n}{7\cdot 10^n},$$

so it follows that

$$\gamma(k) = \sigma^2 \sum_{n=0}^{\infty} \psi_n \psi_{n+|k|}$$

$$\begin{aligned}
\gamma(k) &= \sum_{n=0}^{\infty} \frac{-3\cdot 4^n + 10\cdot(-3)^n}{7\cdot 10^n} \cdot \frac{-3\cdot 4^{n+k} + 10\cdot(-3)^{n+k}}{7\cdot 10^{n+k}} \\
&= \sum_{n=0}^{\infty} -\frac{30\left(-\frac{3}{10}\right)^k\left(-\frac{3}{25}\right)^n}{49} + \frac{100\left(-\frac{3}{10}\right)^k\left(\frac{100}{9}\right)^{-n}}{49} - \frac{30\left(-\frac{3}{25}\right)^n\left(\frac{5}{2}\right)^{-k}}{49} + \frac{9\left(\frac{25}{4}\right)^{-n}\left(\frac{5}{2}\right)^{-k}}{49} \\
&= \sum_{n=0}^{\infty} -\frac{30\left(-\frac{3}{10}\right)^k\left(-\frac{3}{25}\right)^n}{49} + \sum_{n=0}^{\infty} \frac{100\left(-\frac{3}{10}\right)^k\left(\frac{100}{9}\right)^{-n}}{49} + \sum_{n=0}^{\infty} -\frac{30\left(-\frac{3}{25}\right)^n\left(\frac{5}{2}\right)^{-k}}{49} + \sum_{n=0}^{\infty} \frac{9\left(\frac{25}{4}\right)^{-n}\left(\frac{5}{2}\right)^{-k}}{49} \\
&= -\frac{375\left(-\frac{3}{10}\right)^k}{686} + \frac{10000\left(-\frac{3}{10}\right)^k}{4459} + -\frac{375\cdot 2^k 5^{-k}}{686} + \frac{75\cdot 2^k 5^{-k}}{343} \\
&= \frac{25\cdot 50^{-k}\left(605\,(-15)^k - 117\cdot 20^k\right)}{8918}
\end{aligned}$$

The first 10 values of this analytic method are:

```
[1.368019735366674, -0.6399977573446961, 0.10016259250953127, -0.0667834716304104,
↪   0.005341163938102713, -0.007479900201838976,-0.0001070503476115721,
↪   -0.0009082930589818344, -0.00010367534761157211, -0.00011936270183897734]
```

Similarly, following (3.3.4) we can also obtain ψ_n from the following linear equation

$$\psi_j - \sum_{0<k\leq p} \phi_k \psi_{j-k} = 0, \qquad j \geq \max(p, q+1).$$

Then, we obtain the autocovariance function solving the linear system in (3.3.8) and (3.3.9)

$$\gamma(-k) = \gamma(k)$$

$$\gamma(k) - \phi_1\gamma(k-1) - \cdots - \phi_p\gamma(k-p) = \sigma^2 \sum_{k\leq j\leq q} \theta_j \psi_{j-k}, \qquad 0 \leq k < \max(p, q+1),$$

$$\gamma(k) - \phi_1\gamma(k-1) - \cdots - \phi_p\gamma(k-p) = 0, \qquad k \geq \max(p, q+1).$$

This can be computed as follows:

In the first place, we import the packages we're going to use for the symbolic calculations

```
import sympy as sp
import numpy as np
from IPython.display import display, Math
import matplotlib.pyplot as plt

z = sp.Symbol("z", complex = True)
n, k, j = sp.symbols("n k j", integer = True)
N = 20
```

Also, we define ϕ, θ,σ and X_t:

```
phi = 1 - 0.1 * z - 0.12 * z**2
theta = 1 - 7/10 * z
sigma = 1 # standard deviation
X = [0, .644, -.442, -.919, -1.573, - 0.852, -.907, .686, -.753, -.954, .576] # X_0 = 0
```

Now, we define p, q and the coefficients for ϕ and θ

```
phi = sp.Poly(phi)
theta = sp.Poly(theta)
p = phi.degree()
q = theta.degree()
m = max(p,q)
center = max(p,q+1)
```

```
phi_coeff = lambda k: -phi.coeffs()[-1-k] if 0< k <= p else 1 if k == 0 else 0
theta_coeff = lambda k: theta.coeffs()[-1-k] if 0< k <= q else 1 if k == 0 else 0
```

Now the method for the autocovariance:

```
def get_autocovariance(phi, theta,N = N):
    psi = [theta_coeff(0)]
    for j in range(1, max(p,q+1)):
        psi_j = theta_coeff(j) + sum([phi_coeff(k) * psi[j-k] for k in range(1,j+1)])
        psi.append(psi_j)

    for j in range(max(p,q+1), 2*p+2*q):
        psi_j = sum([phi_coeff(k) * psi[j-k] for k in range(1,j+1)])
        psi.append(psi_j)

    gamma_symmetry_matrix = np.zeros((center, 2*center+1))
    gamma_symmetry_vector = np.zeros((center,))
    for j in range(1, max(p,q+1) + 1):
        gamma_symmetry_matrix[j-1,center+j] = 1
        gamma_symmetry_matrix[j-1,center-j] = -1

    gamma_boundary_matrix = np.zeros((center+1, 2*center+1))
    gamma_boundary_vector = np.zeros((center+1,))
    for k in range(0 , center+1):
        for j in range(0,p+1):
            gamma_boundary_matrix[k,center+k-j] = 1 if j == 0 else -phi_coeff(j)
        gamma_boundary_vector[k] = sum([theta_coeff(j)*psi[j-k] for j in range(k,q+1)])



    gamma_solution = np.linalg.solve(np.vstack([gamma_symmetry_matrix,
    ↪   gamma_boundary_matrix]),  np.hstack([gamma_symmetry_vector,
    ↪   gamma_boundary_vector]))[center:]

    for k in range(center+1,N):
        gamma_k = sum([phi_coeff(j) * gamma_solution[k-j] for j in range(1,p+1)])
        gamma_solution = np.append(gamma_solution, gamma_k)
    return gamma_solution
```

The first 10 values of the recursive method `get_autocovariance(phi,theta)` are

```
[1.368019735366674, -0.6399977573446961, 0.10016259250953129, -0.0667834716304104,
↪   0.00534116393810271, -0.00747990020183898, -0.0001070503476 11572,
↪   -0.000908293058981834,-0.000103675347611572, -0.000119362701838977]
```

which luckily coincides with the analytical result so we're going through the right path.

Item (a) Innovations Algorithm

Now that we have the exact autocovariance function of this process, the innovations algoritm can be computed to obtain $\theta_{i,j}$ and r_k.

In the first place, we need to calculate κ

$$\kappa(i,j) = \begin{cases} \sigma^{-1}\gamma(i-j) & 1 \le i,j \le m, \\ \sigma^{-2}\left[\gamma(i-j) - \sum_{r=1}^{p}\phi_r\gamma_X(r-|i-j|)\right] & \min(i,j) \le m < \max(i,j) \le 2m, \\ \sum_{r=0}^{q}\theta_r\theta_{r+|i-j|}, & \min(i,j) > m, \\ 0, & \text{otherwise} \end{cases}$$

Then, using the Innovations Algorithm, we obtain

$$v_0 = \kappa(1,1),$$
$$\theta_{n,n-k} = v_k^{-1}\left(\kappa(n+1,k+1) - \sum_{j=0}^{k-1}\theta_{k,k-j}\theta_{n,n-j}v_j\right), \quad k = 0,\dots,n-1,$$
$$v_n = \kappa(n+1,n+1) - \sum_{j=0}^{n-1}\theta_{n,n-j}^2 v_j.$$

This can be done using the following code,

```python
def innovations_algorithm(phi,theta,sigma,N = N):
    gamma_solution = get_autocovariance(phi,theta)
    gamma = lambda h: gamma_solution[abs(h)]
    def kappa(i,j):
        if 1 <= min(i, j) and max(i, j) <= m:
            return sigma**(-2) * gamma(i - j)
        if min(i, j) <= m < max(i, j) <= 2 * m:
            return sigma**(-2) * (gamma(i - j) - sum([phi_coeff(r) * gamma(r - abs(i -
            ↪ j)) for r in range(1, p + 1)]))
        if min(i, j) > m:
            return sum([theta_coeff(r) * theta_coeff(r + abs(i - j)) for r in range(0, q
            ↪ + 1)])
        else:
            return 0

    kappa_matrix = np.array([[kappa(i,j) for i in range(0,N+3)]for j in
    ↪ range(0,N+3)]).astype(float)

    v_0 = kappa_matrix[1,1]
    v = np.array([v_0])
    O = np.empty((N+1, N+1))
    for n in range(1,N+1):
        for k in range(0,n):
            O[n,n-k] = v[k]**(-1) * (kappa_matrix[n+1,k+1] - sum([
            ↪ O[k,k-j]*O[n,n-j]*v[j] for j in range(0,k) ]))
        v_n = kappa_matrix[n+1,n+1] - sum([ O[n,n-j]**2 * v[j] for j in range(0,n)])
        v = np.append(v,v_n)

    return O, v/sigma**2
```

This way, we can calculate $\hat{X}_k$ using (5.2.15)

$$\hat{X}_{n+1} = \begin{cases} 0 & if\ n = 0, \\ \sum_{j=1}^{n} \theta_{nj}(X_{n+1-j} - \hat{X}_{n+1-j}) & if\ n \geq 1, \end{cases}$$

```
def prediction(phi,theta,sigma,X):
    O, r = innovations_algorithm(phi,theta,sigma,N=len(X))
    Xhat = np.array([0])
    for n in range(0, len(X)):
        if 0 <= n < m:
            Xhat_n = sum([O[n,j] * ( X[n+1-j] - Xhat[n+1-j] ) for j in range(1, n+1)])
            Xhat = np.append(Xhat, Xhat_n)
        elif n >= m:
            Xhat_n = sum([phi_coeff(i)*X[n+1-i] for i in range(1,p+1)]) + sum([O[n,j] *
            ↪  ( X[n+1-j] - Xhat[n+1-j] ) for j in range(1, q+1)])
            Xhat = np.append(Xhat, Xhat_n)
    return Xhat
```

We obtain the following results from the output of `prediction(phi,theta,sigma,X)`

```
array([0.0, 0.0, -0.3012811475409836, 0.125258726843133,
   0.563745296173031, 1.20611643606984, 1.15640073456291, 1.24638808741670,
   ↪  0.351360452344656, 0.779424709348685, 1.02713941765458, 0.258854112159841],
   ↪  dtype=object)
```

That is

$$\hat{X}_0 = \hat{X}_1 = 0,\ \hat{X}_2 = -0.3012,\ \ldots,\ \hat{X}_{10} = 1.0271,\ \hat{X}_{11} = 0.2588.$$

Making a slight modification to the formula we can obtain the h-step prediction given by (5.3.15)

$$P_n X_{n+h} = \begin{cases} \displaystyle\sum_{j=h}^{n+h-1} \theta_{n+h-1,j}(X_{n+h-j} - \hat{X}_{n+h-j}), & 1 \leq h \leq m-n, \\ \displaystyle\sum_{n}^{p} \phi_i P_n X_{n+h-i} + \sum_{h \leq j \leq q} \theta_{n+h-1,j}(X_{n+h-j} - \hat{X}_{n+h-j}), & h > m-n. \end{cases}$$

```
def h_step_prediction(phi,theta,sigma,X,h):
    O, r = innovations_algorithm(phi,theta,sigma,N=len(X)+h)
    Xhat = prediction(phi,theta,sigma,X)

    P_nX = np.append(X, Xhat[-1])
    for k in range(n+2, n+h+1):
        if k <= m:
            P_nX_k = sum([phi_coeff(i)*P_nX[k-i] for i in range(1,p+1)]) + sum([O[k-1,j]
            ↪  * ( X[k-j] - Xhat[k-n-j] ) for j in range(k-n, q+1)])
            P_nX = np.append(P_nX, P_nX_k)
        else:
            P_nX_k = sum([phi_coeff(i)*P_nX[k-i] for i in range(1,p+1)]) + sum([O[k-1,j]
            ↪  * ( X[k-j] - Xhat[k-j] ) for j in range(k-n, q+1)])
```

```
            P_nX = np.append(P_nX, P_nX_k)
    return P_nX
```

We obtain the following results from the output of `h_step_prediction(phi,theta,sigma,X,3)[11:]`

```
array([0.258854112159841, 0.0950054112159841, 0.0405630345807793],
dtype=object)
```

That is,

$$P_{10}X_{11} = \hat{X}_{11} = 0.2588,\ P_{10}X_{12} = 0.0950,\ P_{10}X_{13} = 0.040$$

Finally, for the mean squared error we calculate the power series expansion of $\chi(z) = \phi(z)^{-1}$ using a similar argument as the calculation of ψ to obtain

$$\chi(z) = -\frac{4}{7\left(\frac{2z}{5} - 1\right)} + \frac{3}{7\left(\frac{3z}{10} + 1\right)}$$

$$\implies \chi_n = \frac{50^{-n}\left(3\left(-15\right)^n + 4 \cdot 20^n\right)}{7}.$$

In fact, every symbolic calculation of power series expansions was done with the following code:

```
partial_fraction_expansion = sp.apart(1/phi, full=True).nsimplify(tolerance=1e-10)
display(Math("\\chi(z) = \\phi(z)^{-1} =" + sp.latex(partial_fraction_expansion)))
chi_n = 0
for partial_fraction in partial_fraction_expansion.args:
    poly = sp.Poly(partial_fraction**(-1), z)
    coeffs = (poly).coeffs()
    degree = sp.degree(poly, z)
    a = coeffs[-1]**(-1)
    r = sp.root((a*(-1)**degree) * coeffs[0],degree)
    chi_n += a*sp.binomial(n,degree-1)*r**n

def chi(i):
    return (chi_n).subs({n:i})
display(Math("\\chi_n =" + sp.latex(chi(n).simplify())))
```

However, the formula (5.3.21) yields the same results and is more efficient: $\chi_0 = 1$, then

$$\chi_j = \sum_{k=1}^{\min(p,j)} \phi_k \chi_{j-k}, \qquad j = 1, 2, \ldots.$$

```
chi_list = [1]
for j in range(1,2*N):
    chi_list.append(sum([phi_coeff(k)*chi_list[j-k] for k in range(1,min(p,j)+1)]))

chi = lambda k:  chi_list[k]
```

Therefore, using the formula (5.3.22), we calculate the mean squared error for our projections:

$$\sigma_n^2(h) := E(X_{n+h} - P_n X_{n+h})^2 = \sum_{j=0}^{h-1} \left(\sum_{r=0}^{j} \chi_r \theta_{n+h-r-1,j-r} \right)^2 v_{n+h-j-1}.$$

$$\approx \sigma^2 \sum_{j=0}^{h-1} \left(\sum_{r=0}^{j} \chi_r \theta_{j-r} \right)^2$$

```
n = len(X)-1
O, R = innovations_algorithm(phi,theta,sigma,N=N+2)
v = sigma**2 * R
mean_square_error = lambda h: sigma**2 * sum([sum([chi(r) * theta_coeff(j-r) for r in
↪   range(0,j+1)])**2 for j in range(0,h)])

mean_square_error(1), mean_square_error(2), mean_square_error(3)
```

To have the following output

```
(1.0, 1.36000000000000, 1.36360000000000)
```

Item (b): Prediction Bounds

If $Z_t \sim N(0,1)$, then

$$X_{n+h} \quad \sim \quad N(P_n X_{n+h}, \sigma_n^2(h))$$

Thus, X_{n+h} lies with probability $(1 - 0.05)$ in the interval $[P_n X_{n+h} + \Phi_{0.025}, P_n X_{n+h} + \Phi_{1-0.025}]$.

h	$P_{10}X_{10+h}$	$\sigma_n^2(h)$
1	0.2588	1
2	0.0950	1.36
3	0.04056	1.3636

With these values, we obtain

h	$P_nX_{n+h} + \Phi_{0.025}$	$P_nX_{n+h} + \Phi_{0.975}$
1	-1.11291244	1.63062067
2	-1.27690181	1.46691264
3	-1.33141311	1.41253918

Exercise 5.11.

Use the model defined in Problem 4.12 to find the best linear predictors of the Wolfer sunspot numbers $X_{101}, \dots, X_{105}$ (being careful to take into account the non-zero mean of the series). Assuming that the series is Gaussian, ind 95% prediction bounds for each value. (The observed values of $X_{101}, \dots,$ X_{105} are in fact 139, 111, 102, 66, 45.) How do the predicted values $P_{100}X_{100+h}$ and their mean squared errors behave for large h ?

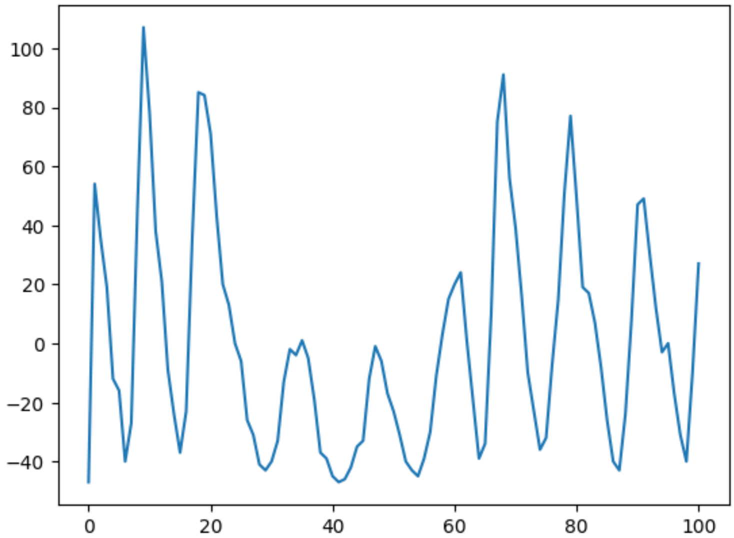
Solution

We insert the numbers from problem 4.12 and the sample numbers in the algorithm:

$$Y_t = X_t - \underbrace{46.93}_{=\mu}$$

$$Y_t - 1.317Y_{t-1} + .634\,Y_{t-2} = Z_t, \qquad \{Z_t\} \sim \mathrm{WN}(0, 289.3).$$

```
mu = 46.93
phi = 1 - 1.317 * z + 0.634 * z**2
theta = 1
sigma = np.sqrt(289.3) # standard deviation
X = np.array([
    0, 101, 82, 66, 35, 31, 7, 20, 92, 154, 125,
    85, 68, 38, 23, 10, 24, 83, 132, 131, 118,
    90, 67, 60, 47, 41, 21, 16, 6, 4, 7,
    14, 34, 45, 43, 48, 42, 28, 10, 8, 2,
    0, 1, 5, 12, 14, 35, 46, 41, 30, 24,
    16, 7, 4, 2, 8, 17, 36, 50, 62, 67,
    71, 48, 28, 8, 13, 57, 122, 138, 103, 86,
    63, 37, 24, 11, 15, 40, 62, 98, 124, 96,
    66, 64, 54, 39, 21, 7, 4, 23, 55, 94,
    96, 77, 59, 44, 47, 30, 16, 7, 37, 74
])
Y = X - mu
plt.plot(Y)
```



Then, using the same algorithm for the previous method,

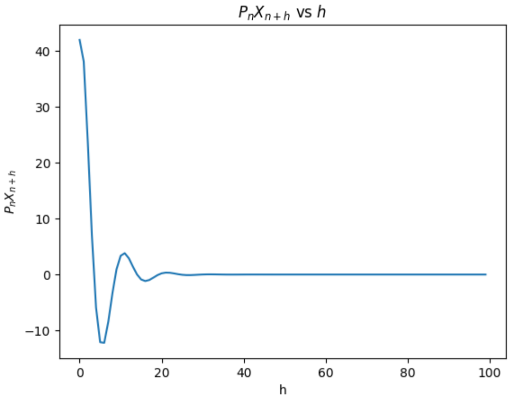
```
mu_n = h_step_prediction(phi,theta,sigma,Y,5)[101:].astype(float)
sigma_n = np.sqrt(np.array([mean_square_error(i) for i in range(1,6)]).astype(float))
```

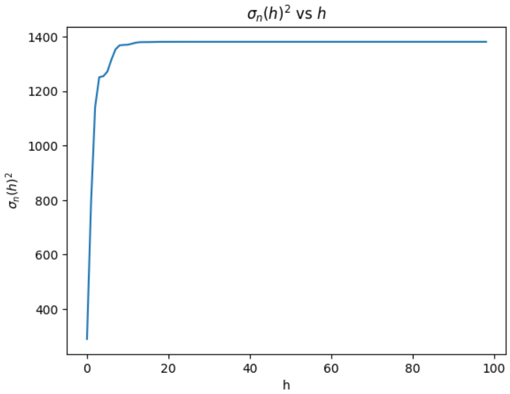
to obtain the following results we recenter again to the original expected value

$$P_n X_{n+h} = P_n Y_{n+h} + \mu$$

h	$P_{100}X_{100+h}$	$\sigma_n^2(h)$
1	88.87681	289.3
2	85.01156877	791.0876677
3	70.48914853	1141.45196582
4	53.81368401	1250.6469773
5	41.05931168	1254.23782416

Also, it seems that after some iterations, $P_n X_{n+h}$ converges to 0 and σ_n^2 converges to 1380.67330749 when h gets larger:





Finally, the bounds for the confidence interval are:

h	$P_nX_{n+h} + \Phi_{0.025}$	$P_nX_{n+h} + \Phi_{0.975}$
1	55.540	122.213
2	29.885	140.138
3	4.271	136.707
4	-15.499	123.127
5	-28.353	110.472

Exercise 7.1.

If $\{X_t\}$ is a causal $\mathbf{AR}(1)$ process with mean μ, show that $\bar{X}_n$ is AN$(\mu,\, \sigma^2(1-\phi)^{-2}n^{-1})$. In a sample of size 100 fbrom an $\mathbf{AR}(1)$ process with $\phi = .6$ and $\sigma^2 = 2$, we obtain $\bar{X}_n = .271$. Construct an approximate 95% confidence interval for the mean μ. Does the data suggest that $\mu = 0$?

Solution

For the causal $\mathbf{AR}(1)$ process with mean μ in this exercise we have that for some $\{Z_t\} \sim \text{IID}(0, \sigma^2)$ (otherwise, if Z_t is not IID, then I don't know how to use Theorem 7.1.2),

$$Y_t = X_t - \mu,$$

$$Y_t - \phi Y_{t-1} = Z_t.$$

Then, since $\{X_t\}$ is causal, $|\phi| < 1$, so it follows that

$$\psi(z) = \frac{\Theta(z)}{\Phi(z)} = \frac{1}{1-\phi z} = \sum_{k=0}^{\infty} \phi^k z^k,$$

$$\implies \psi_n = \phi^n$$

and thus,

$$\begin{aligned} v &= \sigma^2 \left(\sum_{n=0} \psi_n \right)^2 \\ &= \sigma^2 \left(\sum_{n=0} \phi^n \right)^2 \\ &= \frac{\sigma^2}{(1-\phi)^2}. \end{aligned}$$

Then, using Theorem 7.1.2, we conclude that

$$\overline{X}_n = n^{-1}(X_1 + \cdots, X_n) \sim \mathrm{AN}(\mu, \sigma^2(1-\phi)^2 n^{-1})$$

In order to find the bounds for μ, note that

$$n^{1/2}(\overline{X}_n - \mu) \sim W \sim \mathrm{N}\left(0, \sum_{|h|<n} \left(1 - \frac{|h|}{n}\right) \gamma(h)\right),$$

In our case, $\sigma^2 = 4$, $\phi = 0.6$, and therefore,

$$\begin{aligned} \gamma(h) &= \sum_{n=0}^{\infty} \psi_n \psi_{n+|k|} \\ &= \left(\frac{3}{5}\right)^{|k|} \sum_{n=0}^{\infty} \left(\frac{3}{5}\right)^{2n} \\ &= \left(\frac{3}{5}\right)^{|k|} \cdot \frac{25}{16} \end{aligned}$$

After putting the numbers in the calculator, we obtain,

$$\sum_{h=-99}^{99} \left(1 - \frac{|h|}{100}\right) \gamma(h) = 6.1328125.$$

Finally, the bounds of the 95% confidence interval are

$$\Phi_{\alpha/2} = -4.85375593, \qquad \Phi_{1-\alpha/2} = 4.85375593$$

Therefore, $1 - \alpha = 95\%$ confidence interval for $\mu = 100^{-1/2} W - \overline{X}_n = W/10 - 0.271$ is

$$I_{\alpha}^{(\mu)} = [-0.75637559, 0.21437559],$$

so is somewhat implausible for μ to be 0 because is beyond a 95% confidence interval.

Exercise 7.3.

Show that for any series $\{x_1, \dots, x_n\}$, the sample autocovariances satisfy $\sum_{|h|<n} \hat{\gamma}(h) = 0$.

Solution

Let $\overline{x}_n = n^{-1} \sum_{t=1}^{n} x_t$ and let $y_t := x_t - \overline{x}_n$ with $\sum_{t=1}^{n} y_t = 0$. Then,

$$\sum_{h=1}^{n-1} \hat{\gamma}(-h) = \sum_{h=1}^{n-1} \hat{\gamma}(h) = \sum_{h=0}^{n-1} \sum_{t=1}^{n-h} (x_t - \overline{x}_n)(x_{t+h} - \overline{x}_n)$$

$$= \sum_{h=0}^{n-1} \sum_{t=1}^{n-h} y_t \, y_{t+h}.$$

$$\begin{array}{|cccccccccc|c|}
\hline
= & & y_1 \, y_2 & + & y_2 \, y_3 & + \;\cdots\; + & y_{n-2} \, y_{n-1} & + & y_{n-1} \, y_n & & (h=1) \\
 & + & y_1 \, y_3 & + & y_2 \, y_4 & + \;\cdots\; + & y_{n-2} \, y_n & & & & (h=2) \\
 & + & \vdots & + & \vdots & + \;\ddots\; + & \vdots & & & & \vdots \\
 & + & y_1 \, y_{n-1} & + & y_2 \, y_n & & & & & & (h=n-2) \\
 & + & y_1 \, y_n & & & & & & & & (h=n-1) \\
\hline
= & & y_1 \sum_{t=2}^{n} y_t & + & y_2 \sum_{t=3}^{n} y_t & + \;\cdots\; + & y_{n-2} \sum_{t=n-1}^{n} y_t & + & y_{n-1} \sum_{t=n}^{n} y_t & & \\
\hline
\end{array}$$

$$= \sum_{k=1}^{n} y_k \sum_{t=k+1}^{n} y_t$$

$$= \sum_{k=1}^{n} y_k \left(\cancelto{0}{\sum_{t=1}^{n} y_t} - \sum_{t=1}^{k} y_t \right)$$

$$= - \sum_{k=1}^{n} \sum_{t=1}^{k} y_k \, y_t$$

$$\begin{array}{|ccccccccccc|c|}
\hline
= & & & & & & & & - & y_1 \, y_1 & & (k=1) \\
 & & & & & & - & y_2 \, y_1 & - & y_2 \, y_2 & & (k=2) \\
 & & & & - & \cdot^{\cdot^{\cdot}} & - & \vdots & - & \vdots & & \vdots \\
 & & - & y_{n-1} \, y_1 & - & \cdots & - & y_{n-1} \, y_{n-2} & - & y_{n-1} \, y_{n-1} & & (k=n-1) \\
 & - y_n \, y_1 & - & y_n \, y_2 & - & \cdots & - & y_n \, y_{n-1} & - & y_n \, y_n & & (k=n) \\
\hline
= & - \sum_{t=1}^{n-(n-1)} y_{t+n-1} \, y_t & - & \sum_{t=1}^{n-(n-2)} y_{t+n-2} \, y_t & - & \cdots & - & \sum_{t=1}^{n-(1)} y_{t+1} y_t & - & \sum_{t=1}^{n-(0)} y_{t+0} y_t & & \\
\hline
= & -\hat{\gamma}(n-1) & - & \hat{\gamma}(n-2) & - & \cdots & - & \hat{\gamma}(1) & - & \hat{\gamma}(0). & & \\
\hline
\end{array}$$

Therefore,

$$\sum_{h=-n+1}^{-1} \hat{\gamma}(h) = -\sum_{h=0}^{n-1} \hat{\gamma}(h)$$

Finally,

$$\sum_{|h|<n} \hat{\gamma}(h) = \sum_{h=0}^{n-1} \hat{\gamma}(h) + \sum_{h=-n+1}^{-1} \hat{\gamma}(h) = \sum_{h=0}^{n-1} \hat{\gamma}(h) - \sum_{h=0}^{n-1} \hat{\gamma}(h) = 0.$$

Exercise 7.4.

Use formula (7.2.5) to compute the asymptotic covariance matrix of $\hat{\rho}(1), \ldots, \hat{\rho}(h)$ for an $\mathbf{MA}(1)$ process. For which values of j and k in $\{1, 2, \ldots\}$ are $\hat{\rho}(j)$ and $\hat{\rho}(k)$ asymptotically independent?

Solution

If $\{X_t\}$ is a $\mathbf{MA}(1)$ process, then there exists a white noise process $\{Z_t\} \sim \mathrm{WN}(0, \sigma^2)$ and $\theta \neq 0$ such that

$$X_t = Z_t + \theta Z_{t-1}.$$

Also, note that the autocovariance function satisfies $\gamma(1) \neq 0$ and $\gamma(h) = 0$ for $|h| > 1$ and $\mathbf{E}\, X_t = 0$ for every $t = 0, 1, \ldots$. Then, as previous calculations showed,

$$\gamma(0) = \sigma^2(1+\theta^2), \qquad \gamma(1) = \gamma(-1) = \sigma^2\theta$$

$$\implies \rho(0) = 1, \qquad \rho(1) = \rho(-1) = \frac{\theta}{1+\theta^2}$$

So, by following equation (7.2.5)

$$w_{ij} = \sum_{k=1}^{\infty} \{\rho(k+i) + \rho(k-i) - 2\rho(i)\rho(k)\} \times \{\rho(k+j) + \rho(k-j) - 2\rho(j)\rho(k)\}$$

If $|k-i| > 1$, $|k+i| > 1$ and either $|k| > 1$ or $|i| > 1$ that implies

$$\rho(k+i) + \rho(k-i) - 2\rho(i)\rho(k) = 0.$$

Therefore,

Claim 1: The matrix W is a 5 band matrix, that is if $|i-j| > 2$, then $w_{ij} = 0$.

Proof: If $|i-j| > 2$, then $j = i + a$ with $|a| > 2$. We then have 2 cases for k:

- If k that satisfies $|k-i| \leq 1$, by the inverse triangle inequality,

$$|k-j| = |k-i-a| \geq |a| - |k-i| > 2-1 = 1,$$

and thus, $\rho(k-j) = 0$. Also

$$|k+j| \geq |k-j| - |j-j| > 1-0 = 1,$$

and thus, $\rho(k+j) = 0$. Finally, if $|k| \leq 1$, then $k = 1$ because the summation starts at 1. Then, i is either 0 or 2. In both cases, since $|i-j| > 2$ and $j \geq 0$, it must be the case that $|j| > 1$. If $|j| \leq 1$, then $|i| > 2$ so

$$|k| \geq |i| - |k-i| > 2-1 = 1,$$

so either $|j| > 1$ or $|k| > 1$. Therefore, with these 3 inequalities, we conclude

$$\rho(k+j) + \rho(k-j) - 2\rho(j)\rho(k) = 0.$$

- If k that satisfies $|k-j| \leq 1$, we use the same argument to conclude

$$\rho(k+i) + \rho(k-i) - 2\rho(i)\rho(k) = 0.$$

In the summation any of these 2 cases must happen, so if $|i-j| > 2$, then

$$\sum_{k=1}^{\infty} (\rho(k+i) + \rho(k-i) - 2\rho(i)\rho(k)) \times (\rho(k+j) + \rho(k-j) - 2\rho(j)\rho(k)) = 0.$$

The number of calculations in this matrix can also be reduced by the following claim.

Claim 2: if $|i+j| > 5$, then $w_{i,j} = w_{i+h,j+h}$ for every $h = 0, 1, \ldots$.

Proof: In the previous claim we showed that the only way to find non-zero entries is for $|i-j| \leq 2$. Therefore, in that case,

$$\begin{aligned} 2|i| = |i+j+i-j| &\geq |i+j| - |i-j| > 3 \\ &\implies i \geq 2 \end{aligned}$$

The same argument applies to conclude that $j \geq 2$. Then, since $k \geq 1$, it follows that $k+i \geq 2$ and $k+j = 2$. Both inequalities imply that

$$\rho(k+i) = \rho(k+j) = \rho(i) = \rho(j) = 0.$$

Therefore, since $k-h-i \leq 2$ ($\implies \rho(k-h-i) = 0$) for every $k < h$, it follows that

$$\begin{aligned} w_{ij} &= \sum_{k=1}^{\infty} \rho(k-i)\rho(k-j) \\ &= \sum_{k=h}^{\infty} \rho(k-h-i)\rho(k-h-j) \qquad . \\ &= \sum_{k=1}^{\infty} \rho(k-h-i)\rho(k-h-j) = w_{i+h,j+h}. \end{aligned}$$

Claim 3: $w_{ij} = w_{j,i}$, which is elemental.

Finally, we calculate the first 6×6 entries of the matrix and the rest follows from the previous 3 claims.

$$\begin{bmatrix} \frac{16\theta^2}{(\theta^2+1)^2} & \frac{8\theta^3}{(\theta^2+1)^3} + \frac{4\theta}{\theta^2+1} & \frac{4\theta^2}{(\theta^2+1)^2} & 0 & 0 & 0 \\ \frac{8\theta^3}{(\theta^2+1)^3} + \frac{4\theta}{\theta^2+1} & \frac{\theta^2\left(\theta^2+1\right)^2+\left(2\theta^2+\left(\theta^2+1\right)^2\right)^2}{(\theta^2+1)^4} & \frac{2\theta\left(\theta^2+\left(\theta^2+1\right)^2\right)}{(\theta^2+1)^3} & \frac{\theta^2}{(\theta^2+1)^2} & 0 & 0 \\ \frac{4\theta^2}{(\theta^2+1)^2} & \frac{2\theta\left(\theta^2+\left(\theta^2+1\right)^2\right)}{(\theta^2+1)^3} & \frac{2\theta^2}{(\theta^2+1)^2}+1 & \frac{2\theta}{\theta^2+1} & \frac{\theta^2}{(\theta^2+1)^2} & 0 \\ 0 & \frac{\theta^2}{(\theta^2+1)^2} & \frac{2\theta}{\theta^2+1} & \frac{2\theta^2}{(\theta^2+1)^2}+1 & \frac{2\theta}{\theta^2+1} & \frac{\theta^2}{(\theta^2+1)^2} \\ 0 & 0 & \frac{\theta^2}{(\theta^2+1)^2} & \frac{2\theta}{\theta^2+1} & \frac{2\theta^2}{(\theta^2+1)^2}+1 & \frac{2\theta}{\theta^2+1} \\ 0 & 0 & 0 & \frac{\theta^2}{(\theta^2+1)^2} & \frac{2\theta}{\theta^2+1} & \frac{2\theta^2}{(\theta^2+1)^2}+1 \end{bmatrix}$$

Finally, note that from the matrix is clear that $\hat{\rho}(i)$ is asymptotically independent from $\hat{\rho}(j)$ if $|i-j| > 2$. The calculations were a bit long, so I made them symbolically using the following code

```
import sympy as sp
from IPython.display import display, Math

theta = sp.Symbol("theta", complex = True)
m = sp.Symbol("m", integer = True)

rho = lambda m: sp.Piecewise((1, sp.Eq(m,0)), (theta/(1+theta**2), sp.Eq(m,1) |
    sp.Eq(m,-1)),(0,True))
w = lambda i,j: sp.Sum( ( rho(k+i) + rho(k-i) + 2* rho(i)*rho(k) ) * ( rho(k+j) +
    rho(k-j) + 2* rho(j)*rho(k) ) , (k,1,40) )
W = sp.Matrix([[w(i,j).doit() for i in range(10)] for j in range(10)])
display(sp.simplify(W)[:6,:6])
```