Time Series: A First Course with Bootstrap Starter

Contents

Lesson 4-1: Vector Space Geometry

- Euclidean geometry and linear algebra are tools for analyzing n-dimensional space.
- We can adapt these tools to studying time series random vectors.

My add-ons, summing-up

3 applications of projections - forecasting (intuition = residuals to zero, justification: normal equations) - projection of signals - concept of latent process - signal and noise, independent - signal extraction (proj signal onto data, signal = linear filter of data) - interpolation

Concepts and questions

- intuition for inner product, correlation

Tucker questions

TP / questions

- apply to RF3030

Insert, flesh out

- theorems from Pradel

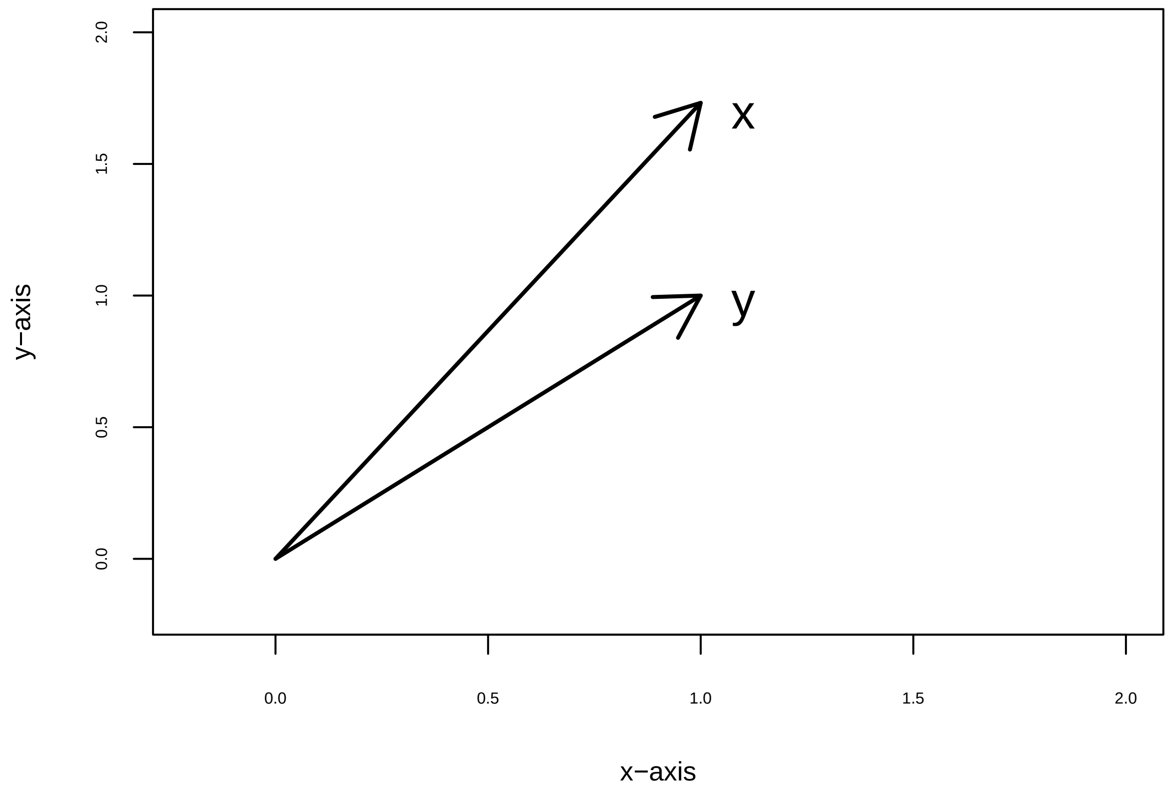
Code skills

Options Set xaxt = "n" and yaxt = "n" to remove the tick labels of the plot

Example 4.1.1. Angle Between Two Vectors

- Let $\underline{x} = [1, \sqrt{3}]$ and $\underline{y} = [1, 1]$
- Let θ be the angle between them.
- Their angles with the x-axis are $\pi/3$ and $\pi/4$ respectively. So $\theta = \pi/12$.

```
x <- c(1,1)
y <- c(1,sqrt(3))

par(mar=c(4,4,2,2)+0.1,cex.lab=.8)
plot(NA,xlim=c(-.2,2),ylim=c(-.2,2),xlab="x-axis",ylab="y-axis",yaxt="n",xaxt="n")
axis(1,cex.axis=.5)
axis(2,cex.axis=.5)
x0 <- c(0,0,1,sqrt(3))
y0 <- c(0,0,1,1)
arrows(x0[1],x0[2],x0[3],x0[4],col=1,lwd=2)
arrows(y0[1],y0[2],y0[3],y0[4],col=1,lwd=2)
text(1.1,sqrt(3)-.05,"x",cex=1.5)
text(1.1,.95,"y",cex=1.5)
```



Inner Product

- Measure a degree of similarity of two vectors via the *inner product*.
- For vectors $\underline{x}, \underline{y} \in \mathbb{R}^n$, their inner product is

$$\langle \underline{x}, \underline{y} \rangle = \sum_{i=1}^{n} x_i y_i.$$

- Also, $\|\underline{x}\| = \sqrt{\langle \underline{x}, \underline{x} \rangle}$ is the norm of $\underline{x}$.
- The angle θ between these two vectors satisfies

$$\cos(\theta) = \frac{\langle \underline{x}, \underline{y} \rangle}{\|\underline{x}\| \|\underline{y}\|}.$$

Theorem 4.1.7. Cauchy-Schwarz Inequality

- For $\underline{x}, \underline{y}$ in a vector space with inner product,

$$|\langle \underline{x}, \underline{y} \rangle| \leq \|\underline{x}\| \|\underline{y}\|.$$

- Equality occurs if and only if the vectors are a scalar multiple of one another.

Lesson 4-2: The L2 Space

- We use *Hilbert Spaces* to think about time series prediction (i.e., forecasting).
- A Hilbert Space is a vector space with inner product, where Cauchy sequences converge.

The Space $\mathbb{L}_2$

- For a given probability space, let $\mathbb{L}_2$ denote all random variables with finite second moment.
- Define an inner product on $\mathbb{L}_2$ as follows:

$$\langle X, Y \rangle = \mathbb{E}[XY]$$

  for $X, Y \in \mathbb{L}_2$.
- The *norm* is $\|X\| = \sqrt{\langle X, X \rangle}$.
- $\mathbb{L}_2$ is a Hilbert Space.

Cauchy-Schwarz

- The Cauchy-Schwarz inequality holds. If the random variables are mean zero, it says that

$$|\text{Cov}[X, Y]| \leq \sqrt{\text{Var}[X]\text{Var}[Y]}.$$

- This is equivalent to $|\text{Corr}[X, Y]| \leq 1$.

Angle Between Random Variables

- Heuristically we can think of θ as the angle between $X, Y \in \mathbb{L}_2$, with

$$\cos(\theta) = \frac{\langle X, Y \rangle}{\|X\|\|Y\|}.$$

- Hence the inner product is zero if $\theta = \pi/2$, i.e., the random variables are *orthogonal*.
- So when mean zero random variables have zero covariance (or correlation), they are orthogonal. We say they are *collinear* if their correlation is ± 1.

Paradigm 4.2.5. Projection

- We can project Y onto X by finding a scalar a such that X is orthogonal to $Y - aX$.
- So $0 = \langle X, Y - aX \rangle$, or $\langle X, Y \rangle = a\|X\|^2$, yielding

$$a = \frac{\langle X, Y \rangle}{\|X\|^2}.$$

- In summary, the projection of Y onto X is

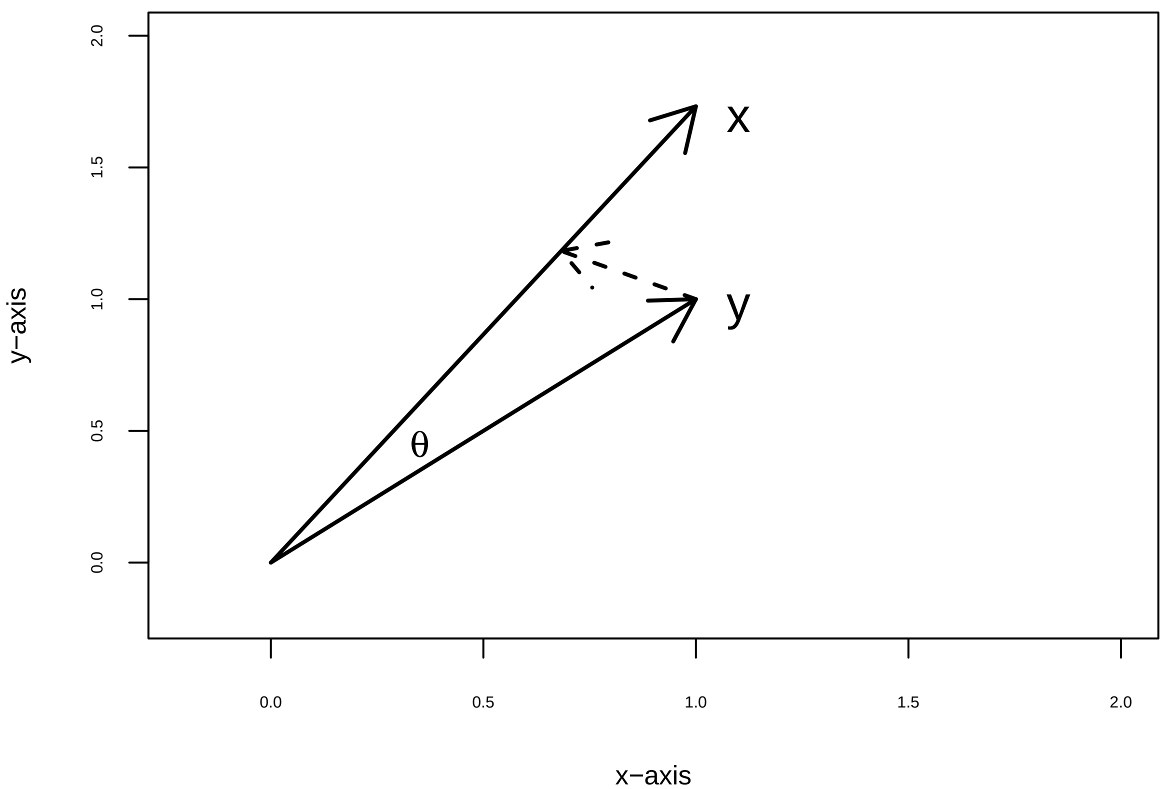
$$\widehat{Y} = \frac{\langle X, Y \rangle}{\|X\|^2} X.$$

- If the random variables are mean zero, this is

$$\widehat{Y} = \frac{\text{Cov}[X, Y]}{\text{Var}[X]} X.$$

```
par(mar=c(4,4,2,2)+0.1,cex.lab=.8)
plot(NA, xlim=c(-.2,2), ylim=c(-.2,2),xlab="x-axis",ylab="y-axis",yaxt="n",xaxt="n")
axis(1,cex.axis=.5)
axis(2,cex.axis=.5)
x0 <- c(0,0,1,sqrt(3))
y0 <- c(0,0,1,1)
arrows(x0[1],x0[2],x0[3],x0[4],col=1,lwd=2)
arrows(y0[1],y0[2],y0[3],y0[4],col=1,lwd=2)
text(1.1,sqrt(3)-.05,"x",cex=1.5)
text(1.1,.95,"y",cex=1.5)
```

```
x <- c(1,sqrt(3))
y <- c(1,1)
dot <- sum(x*y)
proj <- (dot/sum(x^2))*x
z0 <- c(0,0,proj[1],proj[2])
#arrows(z0[1],z0[2],z0[3],z0[4],col=1,lwd=2)
w0 <- c(1,1,proj[1],proj[2])
arrows(w0[1],w0[2],w0[3],w0[4],lwd=2,lty=2)
text(.35,.45,expression(theta),cex=1.2,col=1)
```



Lesson 4-4: Projection in Hilbert Space

- We further examine projections in Hilbert Spaces.

Projection on a Linear Space

- We can project one vector onto a linear space spanned by many vectors.
- Let $\mathcal{M} = \text{span}\{\underset{\sim}{x}_1, \ldots, \underset{\sim}{x}_p\}$, which is all linear combinations of the p spanning vectors.
- To project $\underset{\sim}{y}$ onto $\mathcal{M}$, we seek a linear combination $\widehat{\underset{\sim}{y}}$ of the p spanning vectors, such that $\underset{\sim}{y} - \widehat{\underset{\sim}{y}}$ is orthogonal to $\mathcal{M}$, i.e., to each $\underset{\sim}{x}_i$.

Projection in $\mathbb{L}_2$

- Suppose we want to project $Y \in \mathbb{L}_2$ onto a subspace $\mathcal{M} \subset \mathbb{L}_2$.
- Suppose the subspace is the span of random variables $X_1, \ldots, X_p$.

- Let $\widehat{Y} \in \mathcal{M}$ be the projection. Then $Y - \widehat{Y}$ is orthogonal to each X_i.

Fact 4.4.2. Orthogonality Principle

- Consider projection in $\mathbb{L}_2$. The distance to the projection $\widehat{Y}$ is $\|Y - \widehat{Y}\|$.
- The *orthogonality principle* states that the distance is minimized if and only if $Y - \widehat{Y}$ is orthogonal to all elements of $\mathcal{M}$.
- So the projection onto a subspace actually minimizes the norm distance to that space.

Normal Equations

- The condition for projection says that $0 = \langle Y - \widehat{Y}, X_i \rangle$ for $1 \leq i \leq p$.
- These p equations are called the *normal equations*, because they ensure that the error vector $\epsilon = Y - \widehat{Y}$ is orthogonal (i.e., normal) to the subspace.
- So we have to solve $\langle Y, X_i \rangle = \langle \widehat{Y}, X_i \rangle$ for $1 \leq i \leq p$.
- The distance from Y to the subspace is $\|\epsilon\|$.
- In $\mathbb{L}_2$, $\|\epsilon\|^2 = \mathbb{E}[(Y - \widehat{Y})^2]$ is the *Mean Squared Error* (MSE).

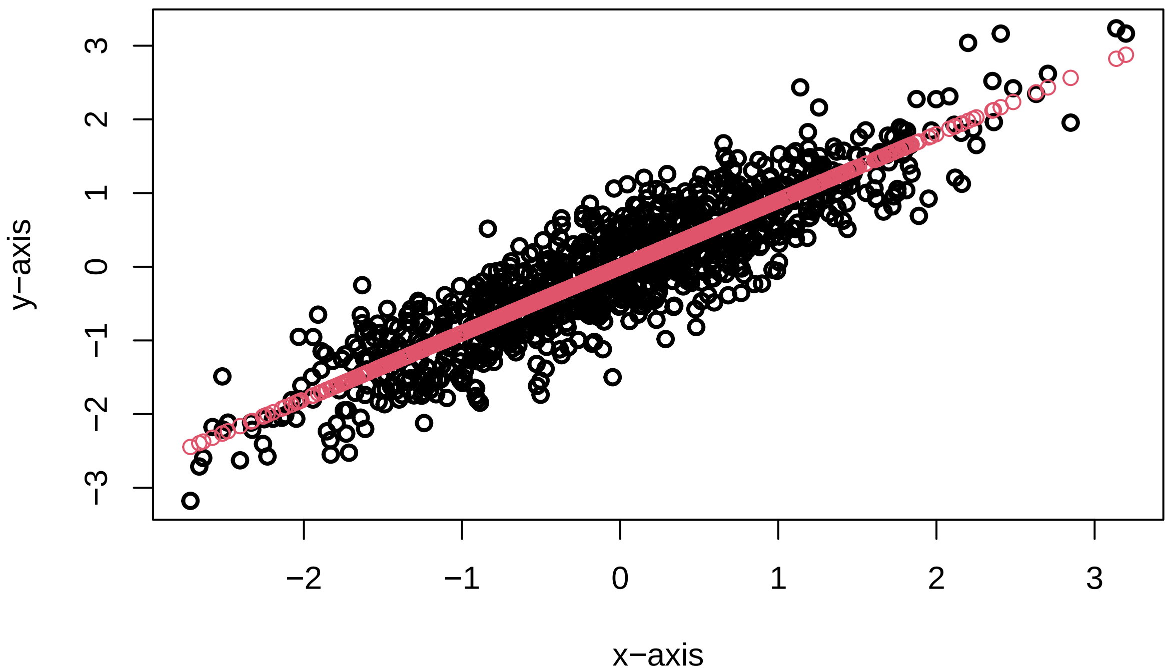
Example of Linear Projection in $\mathbb{L}_2$

- We simulate bivariate Gaussian random variables with correlation ρ and variance 1.
- We can do this by using

$$\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} 1 & \rho \\ 0 & \sqrt{1-\rho^2} \end{bmatrix}.$$

- From prior results, the projection of the second variable onto the first is ρ times the first random variable.
- We compute the projection, and plot.

```r
rho <- .9
mat <- matrix(c(1,rho,0,sqrt(1-rho^2)),2,2)
z <- rnorm(2000)
x <- mat %*% matrix(z,2,1000)
plot(x=x[1,],y=x[2,],xlab="x-axis",ylab="y-axis",axes=TRUE,lwd=2)
proj <- rho*x[1,]
points(x=x[1,],y=proj,col=2)
```



- The projection MSE is $\|X_2 - \rho X_1\|^2 = 1 - \rho^2$.
- We compute the sample variance of the projection errors, and compare to the projection MSE.

```
print(var(proj-x[2,]))
```

```
## [1] 0.1905949
```

```
print(1 - rho^2)
```

```
## [1] 0.19
```

Lesson 4-5: Time Series Prediction

- We apply projection techniques to predict (or forecast) time series.

Paradigm 4.5.1. The Conditional Expectation

- Let $\{X_t\}$ be a weakly stationary time series in $\mathbb{L}_2$.
- Suppose for some $t > n$ we wish to predict X_t from $X_1, \ldots, X_n$. The predictor is denoted $\widehat{X}_t$.
- We want the prediction error to have minimal mean square:

$$\mathbb{E}[(\widehat{X}_t - X_t)^2]$$

  is the Mean Squared Error (MSE).

- Theorem 4.5.2. The minimal MSE predictor is the conditional expectation:

$$\widehat{X}_t = \mathbb{E}[X_t | X_1, \ldots, X_n].$$

Example 4.5.6. Order One Autoregression

- Let $\{X_t\}$ be an AR(1), i.e., $X_t = \phi X_{t-1} + Z_t$ with $\{Z_t\}$ i.i.d. $(0, \sigma^2)$.
- Assume Z_t is independent of X_s for all $s < t$.

One-step ahead prediction

- Consider predicting one-step ahead: we want $\widehat{X}_{n+1}$, given $X_1, \ldots, X_n$.
- We calculate the conditional expectation:

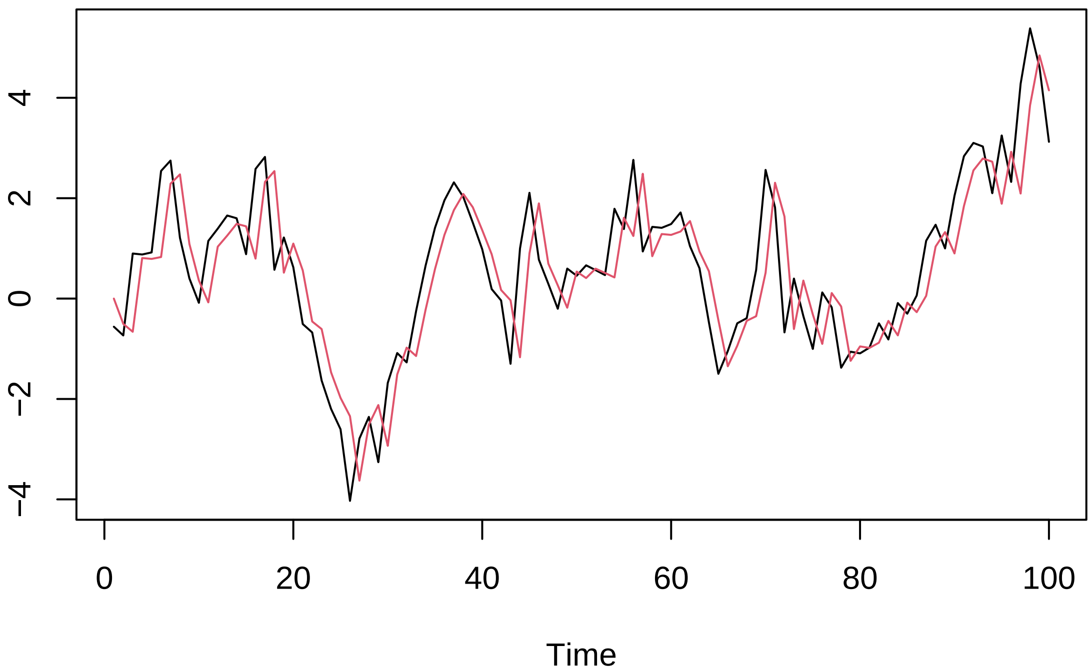
$$\begin{aligned}
\mathbf{E}[X_{n+1}|X_1, \ldots, X_n] &= \mathbf{E}[\phi X_n + Z_{n+1}|X_1, \ldots, X_n] \\
&= \phi\, \mathbf{E}[X_n|X_1, \ldots, X_n] + \mathbf{E}[Z_{n+1}|X_1, \ldots, X_n] \\
&= \phi\, X_n + 0.
\end{aligned}$$

  This uses linearity of conditional expectation, and independence of Z_{n+1} from $X_1, \ldots, X_n$.
- The prediction error is then

$$X_{n+1} - \widehat{X}_{n+1} = X_{n+1} - \phi\, X_n = Z_{n+1},$$

  so that the MSE is $\mathbb{E}[Z_{n+1}^2] = \sigma^2$.

```r
set.seed(123)
n <- 100
z <- rnorm(n)
x <- rep(0,n)
xhat <- rep(0,n)
phi <- .9
x0 <- 0
x[1] <- x0 + z[1]
for(t in 2:n)
{
  x[t] <- phi*x[t-1] + z[t]
  xhat[t] <- phi*x[t-1]
}
plot(ts(x),xlab="Time",ylab="")
lines(ts(xhat),col=2)
```



Two-step ahead prediction

- Consider predicting two steps ahead: we want $\widehat{X}_{n+2}$, given $X_1, \ldots, X_n$.
- Note that by applying the AR(1) recursion twice we can write

$$X_{n+2} = \phi^2 X_n + \phi Z_{n+1} + Z_{n+2}.$$

- Hence the conditional expectation is

$$\begin{aligned}\mathbf{E}[X_{n+2}|X_1,\ldots,X_n] &= \mathbf{E}[\phi^2 X_n + \phi Z_{n+1} + Z_{n+2}|X_1,\ldots,X_n]\\ &= \phi^2\,\mathbf{E}[X_n|X_1,\ldots,X_n] + \phi\,\mathbf{E}[Z_{n+1}|X_1,\ldots,X_n] + \mathbf{E}[Z_{n+2}|X_1,\ldots,X_n]\\ &= \phi^2\,X_n + 0.\end{aligned}$$

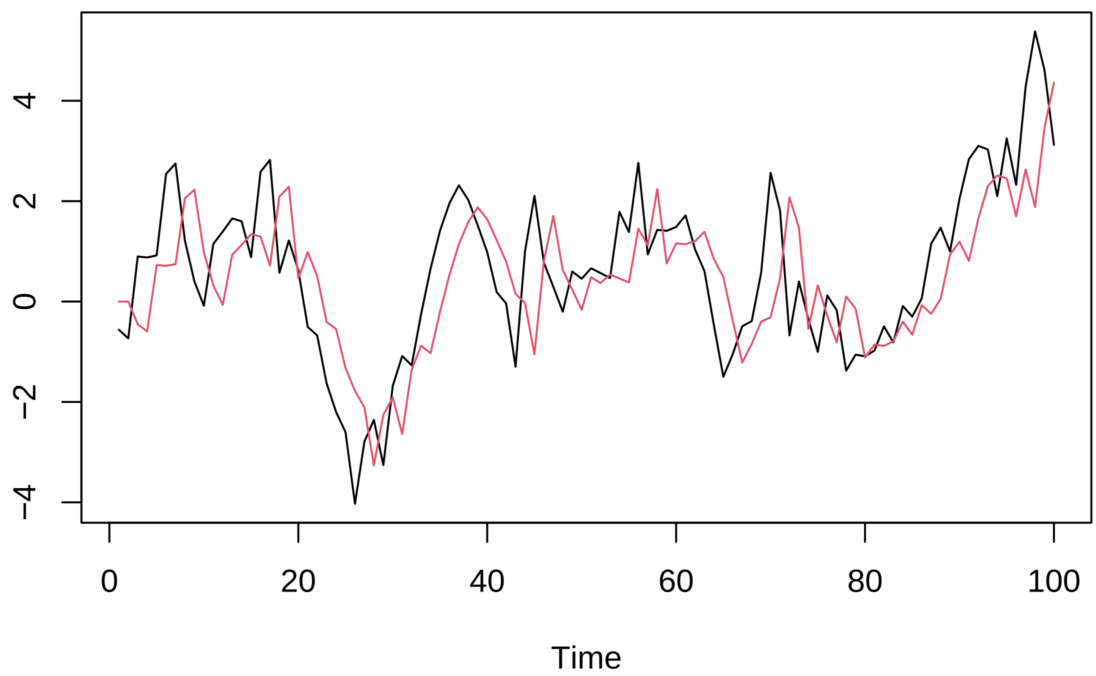
- The prediction error is

$$X_{n+2} - \widehat{X}_{n+2} = \phi^2 X_n + \phi Z_{n+1} + Z_{n+2} - \phi^2 X_n = \phi Z_{n+1} + Z_{n+2}.$$

Hence the prediction MSE is $(1+\phi^2)\sigma^2$.

```
set.seed(123)
n <- 100
z <- rnorm(n)
x <- rep(0,n)
xhat <- rep(0,n)
phi <- .9
x0 <- 0
x[1] <- x0 + z[1]
```

```r
x[2] <- phi*x[1] + z[2]
for(t in 3:n)
{
  x[t] <- phi*x[t-1] + z[t]
  xhat[t] <- phi^2*x[t-2]
}
plot(ts(x),xlab="Time",ylab="")
lines(ts(xhat),col=2)
```



Lesson 4-6: Linear Prediction

- Now we focus on linear prediction. This is the same as the conditional expectation when the distribution is Gaussian, or in the case of a linear process (like the AR(1)).

Paradigm 4.6.1. Linear Prediction and the Yule-Walker Equations

- Let $\{X_t\}$ be a mean zero weakly stationary time series in $\mathbb{L}_2$.
- Say $\mathcal{M}$ is the linear span of the random variables $X_1, \ldots, X_n$.
- Suppose we wish to predict Y from $X_1, \ldots, X_n$. Then the minimal MSE *linear* predictor $\widehat{Y}$ is obtained by projection onto $\mathcal{M}$.
- The orthogonality principle says that

$$0 = \langle Y - \widehat{Y}, X_t \rangle$$

for $t = 1, \ldots, n$. These are the normal equations. They can be rewritten as

$$\langle \widehat{Y}, X_t \rangle = \langle Y, X_t \rangle.$$

One-step Ahead Forecasting

- Suppose $Y = X_{n+1}$.
- Because $\widehat{X}_{n+1} \in \mathcal{M}$, there exist constants $\phi_1, \ldots, \phi_n$ such that

$$\widehat{X}_{n+1} = \phi_1 X_n + \ldots + \phi_n X_1 = \sum_{j=1}^{n} \phi_j X_{n+1-j}.$$

- Then the normal equations imply that for any $1 \leq t \leq n$,

$$\begin{aligned} \langle \widehat{X}_{n+1}, X_t \rangle &= \langle X_{n+1}, X_t \rangle \\ \sum_{j=1}^{n} \phi_j \langle X_{n+1-j}, X_t \rangle &= \langle X_{n+1}, X_t \rangle \\ \sum_{j=1}^{n} \phi_j \gamma(n+1-j-t) &= \gamma(n+1-t). \end{aligned}$$

- This is now linear algebra! Let $\underline{\phi}$ and $\underline{\gamma}_n$ be vectors

$$\underline{\phi} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix} \qquad \underline{\gamma}_n = \begin{bmatrix} \gamma(1) \\ \gamma(2) \\ \vdots \\ \gamma(n) \end{bmatrix}.$$

  And recall that Γ_n is the n-dimensional Toeplitz matrix of autocovariances.
- Now our normal equations are

$$\Gamma_n \, \underline{\phi} = \underline{\gamma}_n.$$

  These are called the *Yule-Walker* equations (i.e., normal equations associated with one-step ahead prediction).
- The solution is

$$\underline{\phi} = \Gamma_n^{-1} \, \underline{\gamma}_n.$$

- The prediction MSE can be derived:

$$\left\| X_{n+1} - \widehat{X}_{n+1} \right\|^2 = \gamma(0) - \underline{\gamma}_n' \, \Gamma_n^{-1} \, \underline{\gamma}_n.$$

Example 4.6.4. Order One Moving Average

- Consider an MA(1) process $\{X_t\}$ given by $X_t = Z_t + \theta Z_{t-1}$, for a white noise $\{Z_t\}$ with variance σ^2.
- Suppose we want to forecast one-step ahead with sample size $n = 2$.
- The Yule-Walker equations are

$$\underline{\phi} = \begin{bmatrix} (1+\theta^2)\sigma^2 & \theta\sigma^2 \\ \theta\sigma^2 & (1+\theta^2)\sigma^2 \end{bmatrix}^{-1} \begin{bmatrix} \theta\sigma^2 \\ 0 \end{bmatrix} = (1+\theta^2+\theta^4)^{-1} \begin{bmatrix} (1+\theta^2)\theta \\ -\theta^2 \end{bmatrix}.$$

- This means that the forecast is

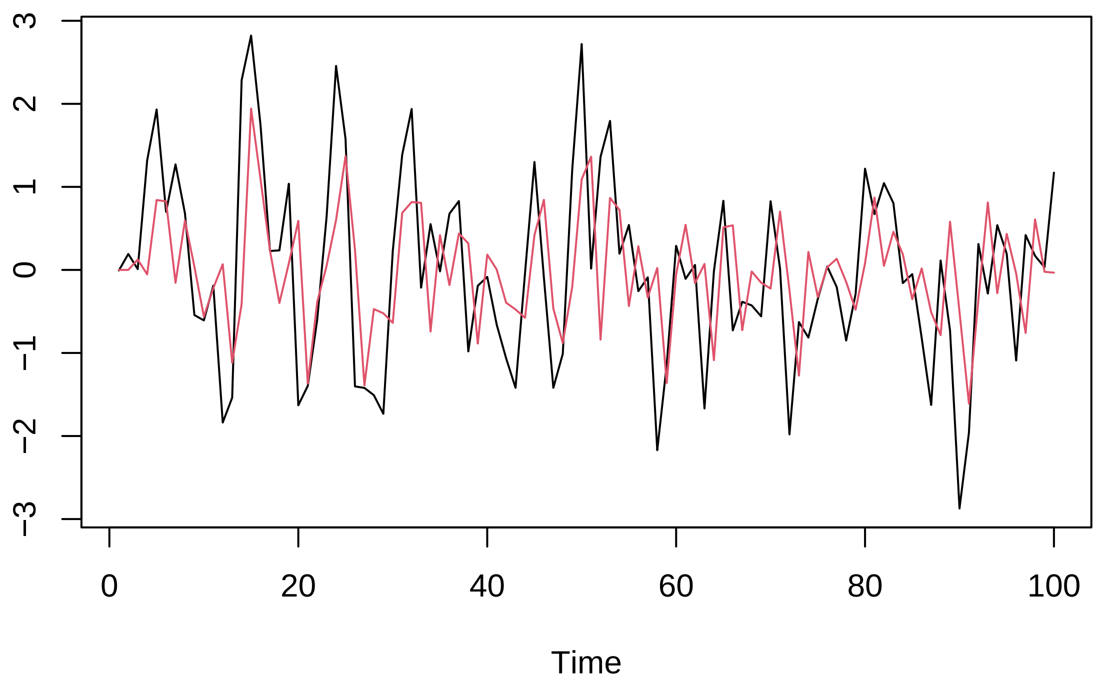
$$\widehat{X}_3 = \frac{(1+\theta^2)\theta}{1+\theta^2+\theta^4} X_2 + \frac{-\theta^2}{1+\theta^2+\theta^4} X_1.$$

- The prediction MSE is

$$\sigma^2 \frac{(1+\theta^2)(1+\theta^4)}{1+\theta^2+\theta^4}.$$

- This formula also applies if we want to predict X_{n+1} only using X_n and X_{n-1}.

```r
set.seed(777)
n <- 100
z <- rnorm(n+1)      # Gaussian input
theta <- .8
x <- z[-1] + theta*z[-(n+1)]
xhat <- rep(0,n)
phi1 <- (1+theta^2)*theta/(1+theta^2+theta^4)
phi2 <- -theta^2/(1+theta^2+theta^4)
for(t in 3:n)
{
  xhat[t] <- phi1*x[t-1] + phi2*x[t-2]
}
plot(ts(x),xlab="Time",ylab="",main="")
lines(ts(xhat),col=2)
```



Lesson 4-7: Orthonormal Sets

- We extend our discussion to sub-spaces that are linear combinations of infinitely many random variables.
- This is so we can project onto the past of a time series (for forecasting), or onto an entire time series (for imputation or signal extraction).

Orthonormal Set

- A collection $\{e_t\}$ where the index set can be $\mathbb{Z}$, has the property that

$$\langle e_s, e_t \rangle = \begin{cases} 1 & s = t, \\ 0 & s \neq t \end{cases}$$

Examples

- The unit vectors in Euclidean space are orthonormal.
- In $\mathbb{L}_2$, a collection of i.i.d. random variables with variance 1 are orthonormal.

Closed Linear Span

- We can take the span of a countable collection of random variables, by considering linear combinations.
- If we also include the limits of sequences of such, it is called the *closed linear span*, denoted

$$\overline{\text{sp}}\{e_t\}$$

- If the basis of the span is finite (i.e., finitely many variables generate the space), then closure is automatic.

Infinite Projection

- Now we can project onto an infinite set.
- For forecasting, we project X_{n+1} onto $\overline{\text{sp}}\{X_t, t \leq n\}$. This is the orthonormal set of random variables X_t for any $t \leq n$, and then we take the closure.
- For index generation, we project one variable Y_t onto an entire time series $\overline{\text{sp}}\{X_t, t \in \mathbb{Z}\}$.
- For imputation, where the value at time t is missing (an NA), we project X_t onto $\overline{\text{sp}}\{X_s, s \neq t\}$.
- In each case, the unknown target (either a forecast, index, missing value, etc.) is projected onto the information we do have.

Example 4.7.8. Order Two Autoregression

- Consider an order 2 autoregressive (or AR(2)) process:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + Z_t$$

  with $Z_t \sim i.i.d.(0, \sigma^2)$. Suppose the recursion is intialized such that the process is stationary.
- We see that Z_t is independent of X_s for all $s < t$.
- The one-step ahead forecast based on the infinite past is denoted $\widehat{X}_{n+1} = P_{\overline{\text{sp}}\{X_t, t \leq n\}}[X_{n+1}]$.
- Its formula is

$$\widehat{X}_{n+1} = \phi_1 X_n + \phi_2 X_{n-1},$$

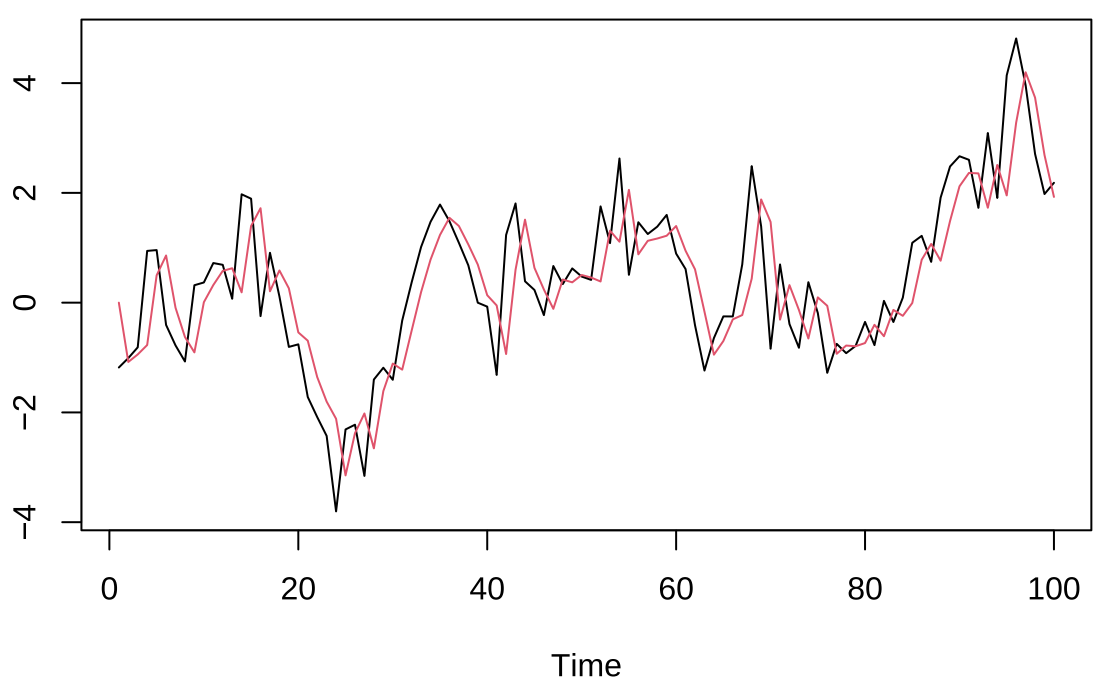
  which is established by verifying the normal equations:

$$\langle \widehat{X}_{n+1} - X_{n+1}, X_t \rangle = \langle Z_{n+1}, X_t \rangle = 0$$

  for $t \leq n$.
- We look at an example with $\phi_1 = .7$ and $\phi_2 = .2$, and $\sigma^2 = 1$.

```
set.seed(123)
n <- 100
phi <- c(.7,.2)
sigma <- 1
Phi <- rbind(phi,c(1,0))
Sigma <- rbind(c(sigma^2,0),c(0,0))
```

```r
Gam.0 <- solve(diag(4) - Phi %x% Phi,matrix(Sigma,ncol=1))
Gam.0 <- matrix(Gam.0,nrow=2)
Gam.half <- t(chol(Gam.0))
x0 <- Gam.half %*% rnorm(2)
z <- rnorm(n)
xvec <- matrix(0,nrow=2,ncol=n)
xhat <- rep(0,n)
xvec[,1] <- x0
for(t in 2:n)
{
  xvec[,t] <- Phi %*% xvec[,t-1] + c(z[t],0)
  xhat[t] <- sum(phi*xvec[,t-1])
}
plot(ts(xvec[1,]),xlab="Time",ylab="")
lines(ts(xhat),col=2)
```



Linear Prediction of AR(p) Processes

- The AR(p) process has the equation

$$X_t = \sum_{j=1}^{p} \phi_j X_{t-j} + Z_t,$$

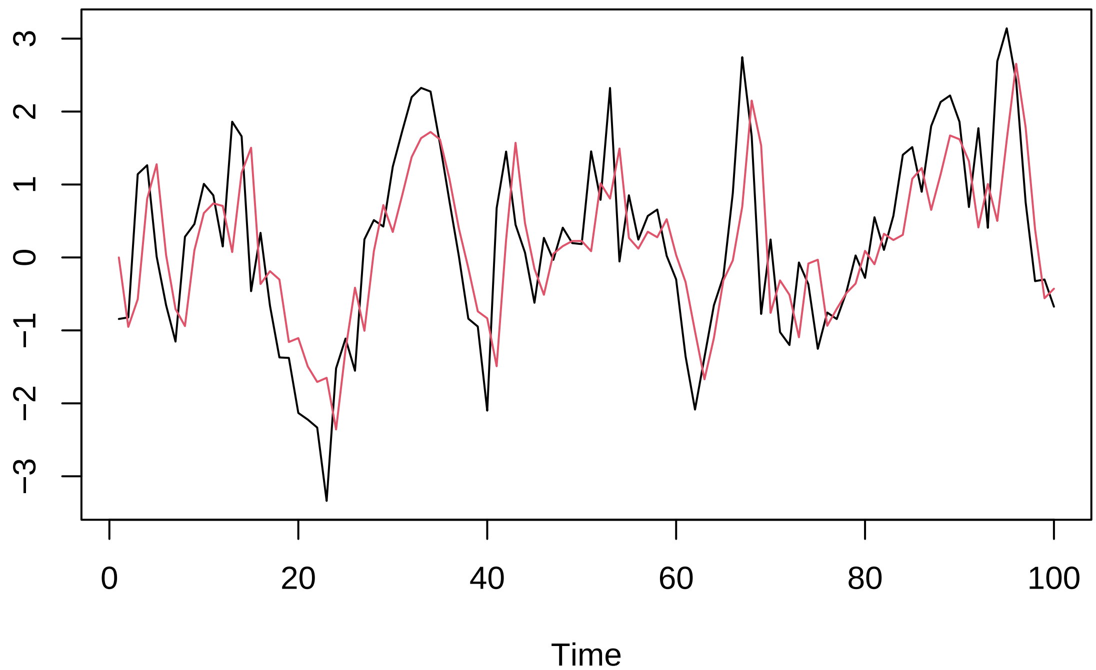
  with $Z_t \sim i.i.d.(0, \sigma^2)$. Suppose the recursion is initialized such that the process is stationary.
- The one-step ahead forecast based on the infinite past is denoted $\widehat{X}_{n+1} = P_{\overline{\text{sp}}\{X_t, t \leq n\}}[X_{n+1}]$.

- Its formula is
$$\widehat{X}_{n+1} = \sum_{j=1}^{p} \phi_j X_{t-j},$$
which is established by verifying the normal equations:
$$\langle \widehat{X}_{n+1} - X_{n+1}, X_t \rangle = \langle Z_{n+1}, X_t \rangle = 0$$
for $t \leq n$.
- We look at a $p = 3$ example with $\phi_1 = .7$, $\phi_2 = .2$, and $\phi_3 = -.2$, and $\sigma^2 = 1$.

```
set.seed(123)
n <- 100
phi <- c(.7,.2,-.2)
sigma <- 1
Phi <- rbind(phi,c(1,0,0),c(0,1,0))
Mod(eigen(Phi)$values)
```

```
## [1] 0.6324555 0.6324555 0.5000000
```

```
Sigma <- rbind(c(sigma^2,0,0),c(0,0,0),c(0,0,0))
Gam.0 <- solve(diag(9) - Phi %x% Phi,matrix(Sigma,ncol=1))
Gam.0 <- matrix(Gam.0,nrow=3)
Gam.half <- t(chol(Gam.0))
x0 <- Gam.half %*% rnorm(3)
z <- rnorm(n)
xvec <- matrix(0,nrow=3,ncol=n)
xhat <- rep(0,n)
xvec[,1] <- x0
for(t in 2:n)
{
  xvec[,t] <- Phi %*% xvec[,t-1] + c(z[t],0,0)
  xhat[t] <- sum(phi*xvec[,t-1])
}
plot(ts(xvec[1,]),xlab="Time",ylab="")
lines(ts(xhat),col=2)
```



Lesson 4-8: Projection of Signals

- We investigate signal extraction through the device of *latent processes*.

Latent Processes

- Suppose $\{W_t\}$ and $\{Z_t\}$ are independent of each other.
- Suppose $X_t = W_t + Z_t$. They are both called latent processes of $\{X_t\}$.

Signal and Noise

- The dynamics of $\{X_t\}$ are a combination of those of the latent processes.
- The autocovariance functions sum up, due to independence:

$$\gamma_X = \gamma_W + \gamma_Z.$$

- Perhaps we are interested in $\{Z_t\}$, and $\{W_t\}$ is viewed as irrelevant. Then Z_t is *signal* and W_t is *noise*.

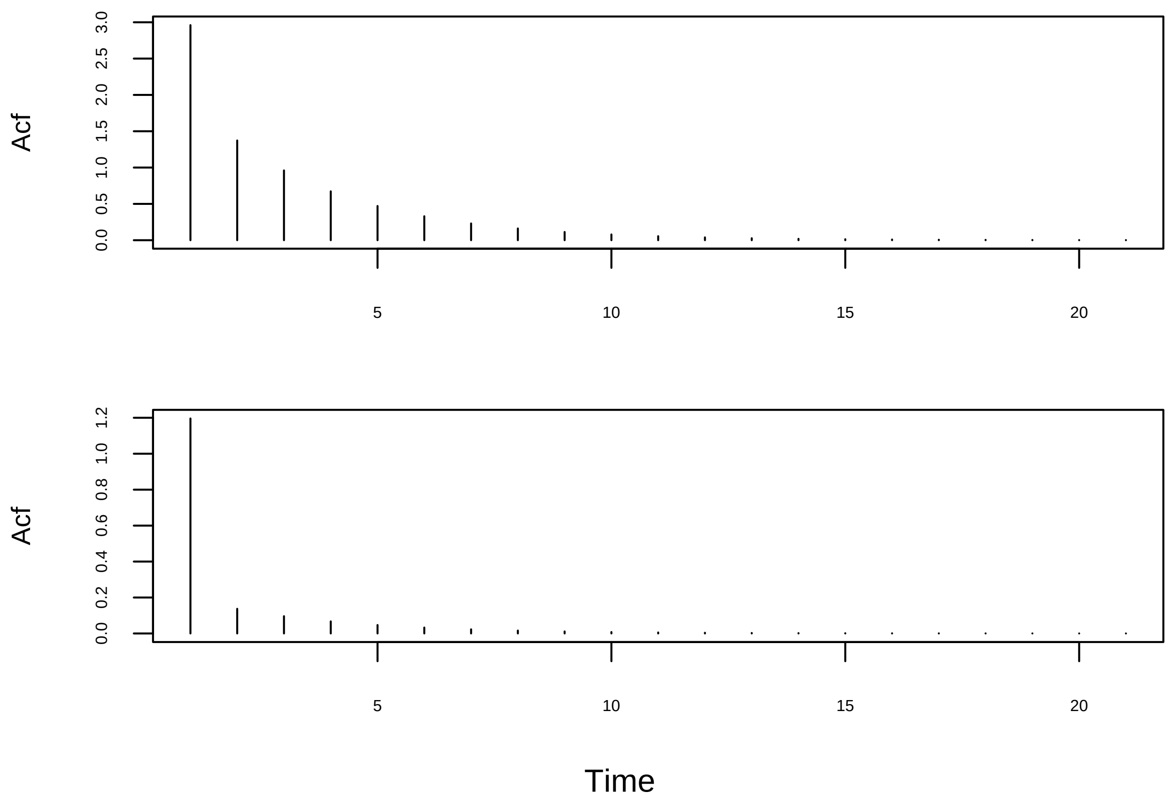
Example 4.8.3. Latent AR(1) with White Noise

- Suppose $\{Z_t\}$ is an AR(1) and $\{W_t\}$ is white noise of variance σ^2.
- We suppose the autoregressive parameter is ϕ and the error variance is $q\sigma^2$, for some $q > 0$.
- Recall that $\gamma_Z(h) = \phi^{|h|}(1-\phi^2)^{-1} q\sigma^2$.
- Then

$$\gamma_X(0) = (1-\phi^2)^{-1} q\sigma^2 + \sigma^2$$
$$\gamma_X(h) = \phi^{|h|}(1-\phi^2)^{-1} q\sigma^2 \quad h \neq 0.$$

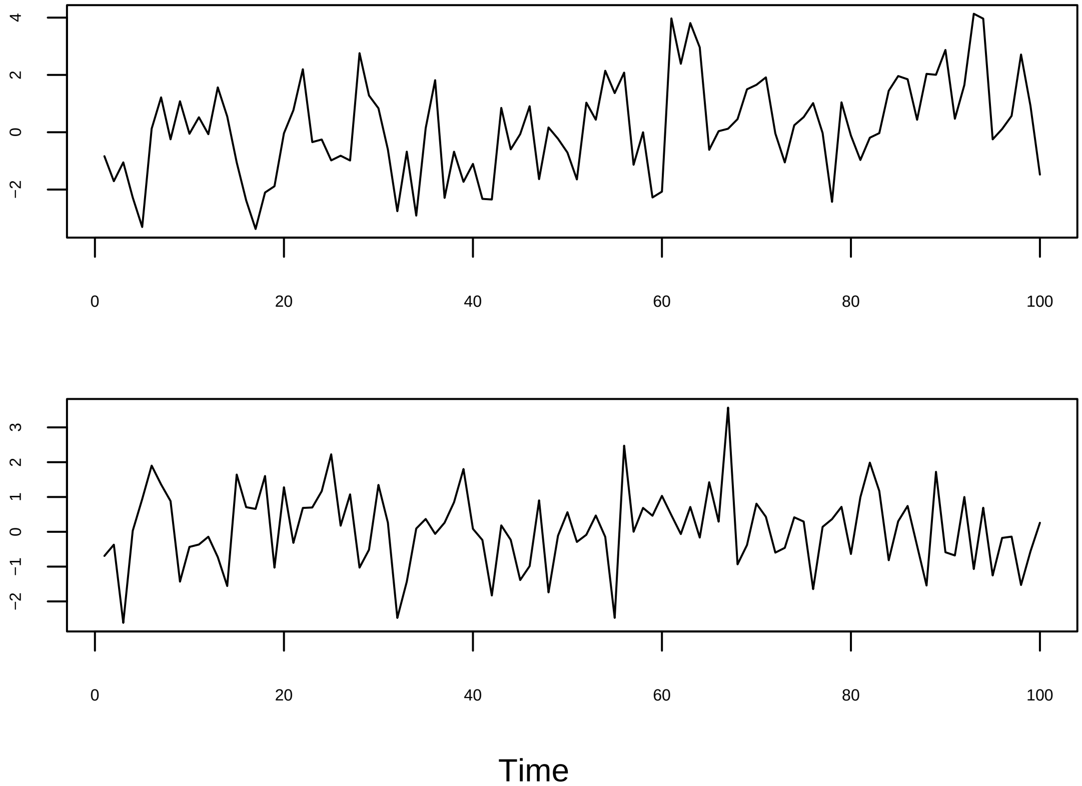
- We can view the impact of q on the autocovariance, with $\phi = .7$ and $\sigma = 1$
- First we examine the case with $q = 1$. Second, we decrease to $q = .1$, which makes the noise relatively stronger, thus dampening the serial correlation.
- q= snr = signal to noise ratio, $q\sigma^2$: residual in AR, hence q smaller noise stronger

```
snr <- 1
phi <- .7
gamma <- snr*phi^{seq(0,20)}/(1-phi^2)
gamma[1] <- gamma[1] + 1
par(oma=c(2,0,0,0),mar=c(2,4,2,2)+0.1,mfrow=c(2,1),cex.lab=.8)
plot(ts(gamma),xlab="",ylab="Acf",yaxt="n",xaxt="n",type="h")
axis(1,cex.axis=.5)
axis(2,cex.axis=.5)

snr <- .1
phi <- .7
gamma <- snr*phi^{seq(0,20)}/(1-phi^2)
gamma[1] <- gamma[1] + 1
plot(ts(gamma),xlab="",ylab="Acf",yaxt="n",xaxt="n",type="h")
axis(1,cex.axis=.5)
axis(2,cex.axis=.5)
mtext(text="Time",side=1,line=1,outer=TRUE)
```



- We also examine a sample path, first with $q = 1$ and second with $q = .1$.
- We see that the second simulation has less structure, and more resembles white noise.

```r
snr <- 1
w <- rnorm(100)
e <- rnorm(100,sd=sqrt(snr))
z <- rep(0,100)
phi <- .7
z0 <- rnorm(1,sd=sqrt(snr))/sqrt(1-phi^2)
z[1] <- phi*z0 + e[1]
for(t in 2:100) { z[t] <- phi*z[t-1] + e[t] }
x <- z + w
par(oma=c(2,0,0,0),mar=c(2,4,2,2)+0.1,mfrow=c(2,1),cex.lab=.8)
plot(ts(x),xlab="",ylab="",yaxt="n",xaxt="n")
axis(1,cex.axis=.5)
axis(2,cex.axis=.5)

snr <- .1
w <- rnorm(100)
e <- rnorm(100,sd=sqrt(snr))
z <- rep(0,100)
phi <- .7
z0 <- rnorm(1,sd=sqrt(snr))/sqrt(1-phi^2)
z[1] <- phi*z0 + e[1]
for(t in 2:100) { z[t] <- phi*z[t-1] + e[t] }
x <- z + w
plot(ts(x),xlab="",ylab="",yaxt="n",xaxt="n")
axis(1,cex.axis=.5)
axis(2,cex.axis=.5)
mtext(text="Time",side=1,line=1,outer=TRUE)
```



Paradigm 4.8.7. Signal Extraction

- Suppose we wish to know the signal, and get rid of the noise. This topic is called *signal extraction*.
- We can approach this as a projection problem: we project Z_t (for any time t) onto $\{X_t\}$. So we seek $\widehat{Z}_t = P_{\overline{\text{sp}}\{X_t\}}[Z_t]$.
- This $\widehat{Z}_t$ is a linear combination of the $\{X_t\}$ variables, and can be written as a linear filter of $\{X_t\}$.
- The finite-sample signal extraction problem is to find $\widehat{Z}_t = P_{\overline{\text{sp}}\{X_1,\ldots,X_n\}}[Z_t]$, for any $1 \leq t \leq n$.

Case of White Noise

signal retrieved as difference with projection of noise

- Suppose that $\{W_t\}$ is white noise (with variance σ^2), and that the signal $\{Z_t\}$ is stationary.
- Then the normal equations yield

$$\widehat{W}_t = P_{\overline{\text{sp}}\{X_1,\ldots,X_n\}}[W_t] = \sigma^2 \underline{e}_t' \Gamma_n^{-1} \underline{X},$$

  where $\underline{e}_t$ is the tth unit vector and Γ_n is the Toeplitz covariance matrix of $\underline{X} = [X_1, \ldots, X_n]'$.
- Then we find

$$\widehat{Z}_t = X_t - \widehat{W}_t,$$

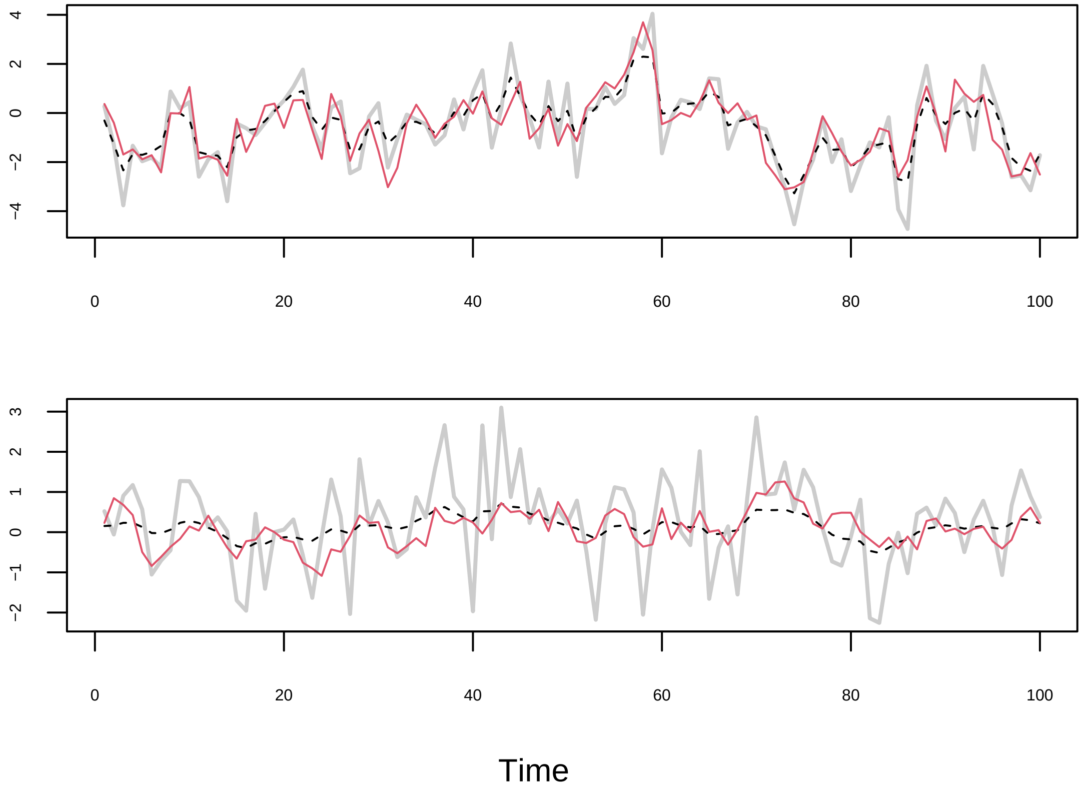
  which follows from $\widehat{Z}_t + \widehat{W}_t = X_t$ (by the linearity of the projection).

Example 4.8.8. Extracting AR(1) Signal from White Noise

- We apply the signal extraction formulas to Example 4.8.3.
- The signal extraction is the dotted line, and the simulation is the solid grey line. The true latent signal is red.

- The first plot has $q = 1$, the second has $q = .1$. The former has a more accurate signal extraction.

```
snr <- 1
w <- rnorm(100)
e <- rnorm(100,sd=sqrt(snr))
z <- rep(0,100)
phi <- .7
z0 <- rnorm(1,sd=sqrt(snr))/sqrt(1-phi^2)
z[1] <- phi*z0 + e[1]
for(t in 2:100) { z[t] <- phi*z[t-1] + e[t] }
x <- z + w
gamma <- snr*phi^{seq(0,99)}/(1-phi^2)
gamma[1] <- gamma[1] + 1
zhat <- x - solve(toeplitz(gamma),x)
par(oma=c(2,0,0,0),mar=c(2,4,2,2)+0.1,mfrow=c(2,1),cex.lab=.8)
plot(ts(x),xlab="",ylab="",col=gray(.8),lwd=2,yaxt="n",xaxt="n")
axis(1,cex.axis=.5)
axis(2,cex.axis=.5)
lines(ts(zhat),lty=2)
lines(ts(z),col=2)

snr <- .1
w <- rnorm(100)
e <- rnorm(100,sd=sqrt(snr))
z <- rep(0,100)
phi <- .7
z0 <- rnorm(1,sd=sqrt(snr))/sqrt(1-phi^2)
z[1] <- phi*z0 + e[1]
for(t in 2:100) { z[t] <- phi*z[t-1] + e[t] }
x <- z + w
gamma <- snr*phi^{seq(0,99)}/(1-phi^2)
gamma[1] <- gamma[1] + 1
zhat <- x - solve(toeplitz(gamma),x)
plot(ts(x),xlab="",ylab="",col=gray(.8),lwd=2,yaxt="n",xaxt="n")
axis(1,cex.axis=.5)
axis(2,cex.axis=.5)
lines(ts(zhat),lty=2)
lines(ts(z),col=2)
mtext(text="Time",side=1,line=1,outer=TRUE)
```



Paradigm 4.8.9. Time Series Interpolation.

- Suppose that we have a single time series with an NA at time t.
- We can approach as a projection problem: we project X_t onto $\{X_s, s \neq t\}$. So we seek $\widehat{X}_t = P_{\overline{\mathrm{sp}}\{X_s, s \neq t\}}[X_t]$.
- This $\widehat{X}_t$ is a linear combination of the $\{X_s, s \neq t\}$ variables, and can be written as a linear filter of them.
- The finite-sample interpolation problem is to find $\widehat{X}_t = P_{\overline{\mathrm{sp}}\{X_1, \ldots, X_{t-1}, X_{t+1}, \ldots, X_n\}}[X_t]$.
- Then the normal equations yield

$$\widehat{X}_t = \underline{v}' \Gamma_{n-1}^{-1} [X_1, \ldots, X_{t-1}, X_{t+1}, \ldots, X_n]',$$

  where $\underline{v} = [\gamma(t-1), \ldots, \gamma(1), \gamma(-1), \ldots, \gamma(t-n)]'$ and Γ_{n-1} is the Toeplitz covariance matrix of $[X_1, \ldots, X_{t-1}, X_{t+1}, \ldots, X_n]'$.
- This is verified by checking the normal equations.

Example: Interpolation for an AR(1) Process.

- Consider an AR(1) process. We claim that

$$\widehat{X}_t = \frac{\phi}{1 + \phi^2} (X_{t+1} + X_{t-1}),$$

  which is verified through checking the normal equations.
- We apply the missing value interpolation to an AR(1) simulation.
- The red dot is the imputation, and the green square is the true value (which we treat as missing).

```r
phi <- .9
e <- rnorm(100,sd=1)
x <- rep(0,100)
x0 <- rnorm(1,sd=1)/sqrt(1-phi^2)
x[1] <- phi*x0 + e[1]
for(t in 2:100) { x[t] <- phi*x[t-1] + e[t] }
x.val <- x[50]
x[50] <- NA
xhat <- (phi/(1+phi^2))*(x[49]+x[51])
plot(ts(x),ylab="")
points(ts(c(rep(NA,49),xhat,rep(NA,50))),col=2,pch=19)
points(ts(c(rep(NA,49),x.val,rep(NA,50))),col=3,pch=22)
```