Lecture 6: Autoregressive Integrated Moving Average Models

Introduction to Time Series, Fall 2023

Ryan Tibshirani

Related reading: Chapters 3.1, 3.3, and 3.6 in Shumway and Stoffer (SS); Chapters 9.1–9.5 and 9.8–9.9 of Hyndman and Athanasopoulos (HA).

1 AR models

- The *autoregressive* (AR) model is one of the foundational legs of ARIMA models, which we'll cover bit by bit in this lecture. (Recall, you've already learned about AR models, which were introduced all the way back in our first lecture)
- Precisely, an AR model of order $p \geq 0$, denoted AR(p), is of the form

$$x_t = \sum_{j=1}^{p} \phi_j x_{t-j} + w_t \tag{1}$$

  where w_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$ is a white noise sequence. Note that we allow the time index to be negative here (we extend time back to $-\infty$), which will useful in what follows
- The coefficients $\phi_1, \ldots, \phi_p$ in (1) are fixed (nonrandom), and we assume $\phi_p \neq 0$ (otherwise the order here would effectively be less than p). Note that in (1), we have $\mathbb{E}(x_t) = 0$ for all t
- If we wanted to allow for a nonzero but constant mean, then we could add an intercept to the model in (1). We'll omit this for simplicity in this lecture
- A useful tool for expressing and working with AR models is the *backshift operator*: this is an operator we denote by B that takes a given time series and shifts it back in time by one index,

$$Bx_t = x_{t-1}$$

- We can extend this to powers, as in $B^2 x_t = BBx_t = x_{t-2}$, and so on, thus

$$B^k x_t = x_{t-k}$$

- Returning to (1), note now that we can rewrite this as

$$x_t - \phi_1 x_{t-1} - \phi_2 - x_{t-2} - \cdots - \phi_p x_{t-p} = w_t$$

  or in other words, using backshift notation

$$\Big(1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p\Big) x_t = w_t \tag{2}$$

- Hence (2) is just a compact way to represent the AR(p) model (1) using the backshift operator B. Often, authors will write this model even more compactly as

$$\phi(B) x_t = w_t \tag{3}$$

  where $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$ is called the *autoregressive operator* of order p, associated with the coefficients $\phi_1, \ldots, \phi_p$
- Figure 1 shows two simple examples of AR processes

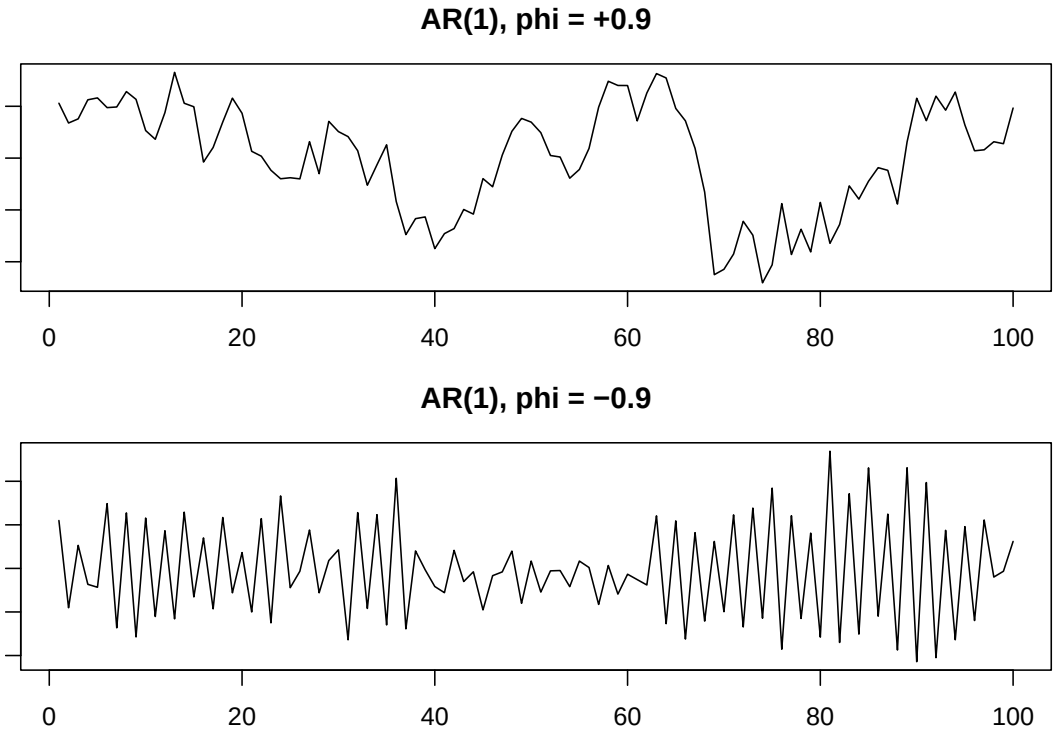

Figure 1: Two examples of AR(1) processes, with $\phi = \pm 0.9$.

1.1 AR(1): auto-covariance and stationarity

- A key question for us will be: *under what conditions does the AR model in* (1)*, or equivalently* (3)*, define a stationary process?*
- The answer will turn out to be fairly sophisticated, but we can glean some intuition by starting with the AR(1) case:

$$x_t = \phi x_{t-1} + w_t \tag{4}$$

- Note that a random walk is the special case with $\phi = 1$. We already know (from previous lectures) that this is nonstationary, so certainly (4) cannot be stationary for any ϕ
- Unraveling the iterations, we get

$$\begin{aligned} x_t &= \phi^2 x_{t-2} + \phi w_{t-1} + w_t \\ &= \phi^3 x_{t-3} + \phi^2 w_{t-2} + \phi w_{t-1} + w_t \\ &\vdots \\ &= \phi^k x_{t-k} + \sum_{j=0}^{k} \phi^j w_{t-j} \end{aligned}$$

- If $|\phi| < 1$, then we can send $k \to \infty$ in the last display to get

$$x_t = \sum_{j=0}^{\infty} \phi^j w_{t-j} \tag{5}$$

This is called the *stationary representation* of the AR(1) process (4)

- Why is it called this? We can compute the auto-covariance function, writing $\sigma^2 = \mathrm{Var}(w_t)$ for the noise variance, as

$$\begin{aligned}
\mathrm{Cov}(x_t, x_{t+h}) &= \mathrm{Cov}\bigg(\sum_{j=0}^{\infty} \phi^j w_{t-j}, \sum_{\ell=0}^{\infty} \phi^\ell w_{t+h-\ell}\bigg) \\
&= \sum_{j,\ell=0}^{\infty} \phi^j \phi^\ell \, \mathrm{Cov}(w_{t-j}, w_{t+h-\ell}) \\
&= \sum_{j=0}^{\infty} \phi^j \phi^{j+h} \sigma^2 \\
&= \sigma^2 \phi^h \sum_{j=0}^{\infty} \phi^{2j} \\
&= \sigma^2 \frac{\phi^h}{1-\phi^2} \qquad (6)
\end{aligned}$$

  where we used the fact that $\sum_{j=0}^{\infty} b^j = 1/(1-b)$ for $|b| < 1$. Since the auto-covariance in the last line only depends on h, we can see that the AR(1) process is indeed stationary

- To reiterate, the representation (5), and the auto-covariance calculation just given, would have not been possible unless $|\phi| < 1$. This condition is required in order for the AR(1) process to have a stationary representation. We will see later that we can generalize this to a condition that applies to an AR(p), yielding an analogous conclusion. The conclusion we will be looking for is explained next

1.2 Causality (no, not the usual kind)

- Now we will introduce a concept called *causality*, which generalizes what we just saw falls out of an AR(1) when $|\phi| < 1$. This is a slightly unfortunate bit of nomenclature that nonetheless seems to be common in the time series literature. It has really nothing to do with causality used in the broader sense in statistics. We will ... somewhat begrudgingly ... stick with the standard nomenclature in time series here

- We say that a series x_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$ is *causal* provided that it can be written in the form

$$x_t = \sum_{j=0}^{\infty} \psi_j w_{t-j} \qquad (7)$$

  for a white noise sequence w_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$, and coefficients such that $\sum_{j=0}^{\infty} |\psi_j| < \infty$

- You should think of this as a generalization of (5), where we allow for arbitrary coefficients $\psi_0, \psi_1, \psi_2, \ldots$, subject to an absolute summability condition

- It is straightforward to check that causality actually implies stationarity: we can just compute the auto-covariance function in (7), similar to the above calculation:

$$\begin{aligned}
\mathrm{Cov}(x_t, x_{t+h}) &= \mathrm{Cov}\bigg(\sum_{j=0}^{\infty} \psi_j w_{t-j}, \sum_{\ell=0}^{\infty} \psi_\ell w_{t+h-\ell}\bigg) \\
&= \sum_{j,\ell=0}^{\infty} \psi_j \psi_\ell \, \mathrm{Cov}(w_{t-j}, w_{t+h-\ell}) \\
&= \sigma^2 \sum_{j=0}^{\infty} \psi_j \psi_{j+h}
\end{aligned}$$

  The summability condition ensures that these calculations are well-defined and that the last display is finite. Since this only depends on h, we can see that the process is indeed stationary

- Thus, to emphasize, causality actually tells us *more* than stationary: it is stationary "plus" a representation a linear filter of past white noise variates, with summable coefficients
- Note that when $\psi_j = \phi^j$, the summability condition $\sum_{j=0}^{\infty} |\psi_j| < \infty$ is true if and only if $|\phi| < 1$. Hence what we actually proved above for AR(1) was that it is causal if and only if $|\phi| < 1$. And it is this condition—for causality—that we will actually generalize for AR(p) models, and beyond

2 MA models

- A *moving average* (MA) model is "dual", in a colloquial sense, to the AR model. Instead of having x_t evolve according to a linear combination of the recent past, the *errors* in the model evolve according to a linear combination of white noise
- Precisely, an MA model of order $q \geq 0$, denoted MA(q), is of the form

$$x_t = w_t + \sum_{j=1}^{q} \theta_j w_{t-j} \tag{8}$$

  where w_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$ is a white noise sequence
- The coefficients $\theta_1, \ldots, \theta_q$ in (8) are fixed (nonrandom), and we assume $\theta_q \neq 0$ (otherwise the order here would effectively be less than q). Note that in (8), we have $\mathbb{E}(x_t) = 0$ for all t
- Again, we can rewrite (8), using backshift notation, as

$$x_t = \Big(1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q\Big) w_t \tag{9}$$

- Often, authors will write (9) even more compactly as

$$x_t = \theta(B) w_t \tag{10}$$

  where $\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q$ is called the *moving average operator* of order q, associated with the coefficients $\theta_1, \ldots, \theta_q$
- Figure 2 shows two simple examples of MA processes

2.1 Stationarity

- Unlike AR processes, an MA process (8) is stationary *for any values of the parameters* $\theta_1, \ldots, \theta_q$
- To check this, we compute the auto-covariance function using a similar calculation to those we've done before, writing $\theta_0 = 1$ for convenience,

$$\begin{aligned} \mathrm{Cov}(x_t, x_{t+h}) &= \mathrm{Cov}\bigg(\sum_{j=0}^{q} \theta_j w_{t-j}, \sum_{\ell=0}^{q} \theta_\ell w_{t+h-\ell}\bigg) \\ &= \sum_{j,\ell=0}^{q} \theta_j \theta_\ell \, \mathrm{Cov}(w_{t-j}, w_{t+h-\ell}) \\ &= \sum_{j\,:\,0 \leq j, j+h \leq q} \theta_j \theta_{j+h} \sigma^2 \end{aligned} \tag{11}$$

  Since this only depends on h, we can see that the process is indeed stationary
- The similarity in these calculations brings us to pause to emphasize the following connection: *an AR(1) model with* $|\phi| < 1$ *is also a particular infinite-order MA model*, as we saw in the stationary representation (5). We will see later that there are more general connections to be made

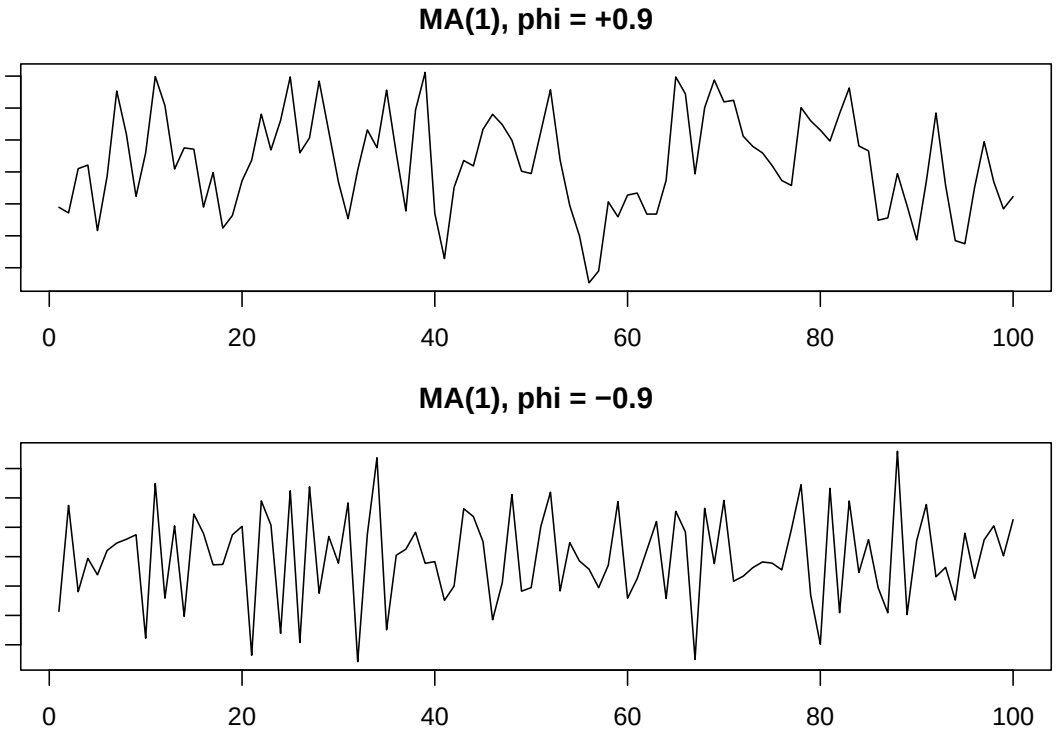

Figure 2: Two examples of MA(1) processes, with $\theta = \pm 0.9$.

2.2 MA(1): issues with non-uniqueness

- Consider the MA(1) model:

$$x_t = w_t + \theta w_{t-1} \tag{12}$$

- According to (11), we can compute its auto-covariance simply (recalling $\theta_0 = 1$) as

$$\gamma_x(h) = \begin{cases} (1+\theta^2)\sigma^2 & h = 0 \\ \theta\sigma^2 & |h| = 1 \\ 0 & |h| > 1 \end{cases} \tag{13}$$

- The corresponding auto-correlation function is thus

$$\rho(h) = \begin{cases} 1 & h = 0 \\ \frac{\theta}{1+\theta^2} & |h| = 1 \\ 0 & |h| > 1 \end{cases}$$

- If we look carefully, then we can see a problem lurking here: the auto-correlation function is unchanged if we replace θ by $1/\theta$
- And in fact, the auto-covariance function (13) is unchanged if we replace θ and σ^2 with $1/\theta$ and $\sigma^2\theta^2$; e.g., try $\theta = 5$ and $\sigma^2 = 1$, and $\theta = 1/5$ and $\sigma^2 = 25$, you'll find that the auto-covariance function is the same in both cases
- This is not good because it means we cannot detect the difference in an MA(1) model with parameter θ and normal noise with variance σ^2 from another MA(1) model with parameter $1/\theta$ and normal noise with variance $\sigma^2\theta^2$

- In other words, there is some *non-uniqueness* of *redundancy* in the parametrization—different choices of parameters will actually lead to the same behavior in the model at the end
- In the MA(1) case, the convention is to simply choose the parametrization with $|\theta| < 1$. Note that we can write

$$w_t = -\theta w_{t-1} + x_t$$

which is like an AR(1) process with the roles of x_t and w_t reversed. Thus by the same arguments that led to (5), when $|\theta| < 1$, we now have

$$w_t = \sum_{j=0}^{\infty} (-\theta)^j x_{t-j} \tag{14}$$

This is called the *invertible representation* of the MA(1) process (12)

- We will see soon that we can generalize this to a condition that applies to a general MA(q), yielding an analogous conclusion. The conclusion we will be looking for is explained next

2.3 Invertibility

- Before we turn to ARMA models, we define one last concept called *invertibility*, which generalizes what we just saw for MA(1) when $|\theta| < 1$
- We say that a series x_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$ is *invertible* provided that it can be written in the form

$$w_t = \sum_{j=0}^{\infty} \pi_j x_{t-j} \tag{15}$$

for a white noise sequence w_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$, and coefficients such that $\sum_{j=0}^{\infty} |\pi_j| < \infty$, where we set $\pi_0 = 1$

- You should think of this as a generalization of (14), where we allow for arbitrary coefficients $\pi_1, \pi_2, \ldots$, subject to an absolute summability condition
- And of course, note how invertibility (15) is kind of an opposite condition to causality (7)

3 ARMA models

- AR and MA models have complementary characteristics. The auto-covariance of an AR model generally decays away from $h = 0$, whereas that for an MA process has finite support—in other words, at a certain lag, variates along an MA sequence are completely uncorrelated. You can compare (6) and (13) for the AR(1) and MA(1) models
- (The spectral perspective, by the way, provides another nice way of viewing these complementary characteristics. In the spectral domain, the story is somewhat flipped: the spectral density of an MA process generally decays away from $\omega = 0$, whereas that for an AR process can be much more locally concentrated around particular frequencies; recall our examples from the last lecture)
- Sure, there is some duplicity in representation here, as we will see—we can write some AR models as infinite-order MA models, and some MA models as infinite-order AR models. But that's OK! We can take the most salient features that each model represents, and combine them to get a simple model formulation that exhibits both sets of features, simultaneously. This is exactly what an ARMA model does
- Precisely, an ARMA model of orders $p, q \geq 0$, denoted ARMA(p, q), is of the form

$$x_t = \sum_{j=1}^{p} \phi_j x_{t-j} + \sum_{j=0}^{q} \theta_j w_{t-j} \tag{16}$$

where w_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$ is a white noise sequence

- The coefficients $\phi_1, \ldots, \phi_p, \theta_0, \ldots, \theta_q$ in (16) are fixed (nonrandom), and we assume $\phi_p, \theta_q \neq 0$, and we set $\theta_0 = 1$. Note that in (16), we have $\mathbb{E}(x_t) = 0$ for all t
- As before, we can represent an ARMA model more compactly using backshift notation, rewriting (16) as

$$\Big(1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p\Big) x_t = \Big(1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q\Big) w_t \tag{17}$$

- Often, authors will write (17) even more compactly as

$$\phi(B) x_t = \theta(B) w_t \tag{18}$$

where $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$ and $\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q$ are the AR and MA operators, as before

3.1 Parameter redundancy

- For ARMA models, there is an issue of parameter redundancy, just like there are for MA models. If $\eta(B)$ is any (invertible) operator, then we can transform (18) by applying $\eta(B)$ on both sides,

$$\eta(B)\phi(B) x_t = \eta(B)\theta(B) w_t$$

which may look like a different model, but the dynamics are entirely the same

- As an example, consider white noise $x_t = w_t$, and multiply both sides by $\eta(B) = 1 - B/2$. This gives:

$$x_t = \frac{1}{2} x_{t-1} + w_t - \frac{1}{2} w_{t-1}$$

which looks like an ARMA(1,1) model, but it is nothing else that white noise!

- How do we resolve this issue? Doing so requires introducing another concept. The *AR and MA polynomials* associated with the ARMA process (16) are

$$\phi(z) = 1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p \tag{19}$$

$$\theta(z) = 1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q \tag{20}$$

respectively. To be clear, these are polynomials over *complex numbers* $z \in \mathbb{C}$. (Note that these are just what we get by taking the AR and MA operators, and replacing the backshift operator B by a complex argument z)

- As it turns out, several important properties of ARMA models can be derived by placing conditions on the AR and MA polynomials. Here we'll see the first one, to deal with parameter redundancy: *when we speak of an ARMA model, we implicitly assume that the AR and MA polynomials $\phi(z)$ and $\theta(z)$, in (19) and (20), have no common factors.* This rules out a case like

$$x_t = \frac{1}{2} x_{t-1} + w_t - \frac{1}{2} w_{t-1}$$

because the polynomials each have $1 - z/2$ as a common factor. Hence we do not even refer to the above as an ARMA(1,1) model

3.2 Causality and invertibility

- Now we learn about two more conditions on the AR and MA polynomials that imply important general properties of the underlying ARMA process, and generalize calculations we saw earlier for AR(1) and MA(1) models
- Before we describe these, we recall the following terminology: for a polynomial $P(z) = \sum_{j=0}^k a_j z^j$, we say that z is a *root* of P provided $P(z) = 0$

- And we say that a point $z \in \mathbb{C}$ lies *outside of the unit circle* (in the complex plane $\mathbb{C}$) provided that $|z| > 1$, where $|z|$ is the complex modulus of z (recall $|z|^2 = \mathrm{Re}\{z\}^2 + \mathrm{Im}\{z\}^2$)
- The first property: *the ARMA process* (16) *has a causal representation* (7) *if and only if all roots of the AR polynomial* (19) *lie outside of the unit circle*. The coefficients $\psi_0, \psi_1, \psi_2, \ldots$ in the causal representation can be determined by solving

$$\psi(z) = \phi(z)^{-1}\theta(z) \iff \sum_{j=0}^{\infty} \psi_j z^j = \frac{1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q}{1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p}, \quad \text{for } |z| < 1$$

- The second property: *the ARMA process* (16) *is invertible* (15) *if and only if all roots of the MA polynomial* (20) *lie outside of the unit circle*. The coefficients $\pi_1, \pi_2, \ldots$ in the invertible representation can be determined by solving

$$\pi(z) = \theta(z)^{-1}\phi(z) \iff \sum_{j=0}^{\infty} \pi_j z^j = \frac{1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p}{1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q}, \quad \text{for } |z| < 1$$

- (As an aside, you can see that parameter redundancy issues would not affect the causal and invertible representations, as they shouldn't, because any common factors in $\phi(z)$ and $\theta(z)$ would cancel in their ratios which determine the coefficients in the causal and invertible expansions)
- Interestingly, these results can also be interpreted as follows:
  - An AR(q) process, such that the roots of ϕ lie outside the unit circle, can also be written as an MA(∞) process (this is what causality (7) means)
  - An MA(q) process, such that the roots of θ lie outside the unit circle, can also be written as an AR(∞) process (this is what invertibility (15) means, as it says $x_t = -\sum_{j=1}^{\infty} \pi_j x_{t-j} + w_t$)
- We won't cover the proofs of these properties or go into further details about them. But you will see several hints of their significance (what the causal and invertible representations allow us to do) in what follows. At a high-level, they are worth knowing because they are considered foundational results for ARMA modeling (just like it is worth knowing foundations for regression modeling, and so on). You can refer to Appendix B.2 of SS for proofs, or read more in Chapter 3 of SS
- Also, recall that causality implies stationarity, so what we have just learned is the general result that renders an ARMA(p, q) process stationary
- Finally, as a summary, here are three equivalent ways to represent a causal, invertible ARMA(p, q) model:

$$\begin{aligned} &\phi(B)x_t = \theta(B)w_t \\ &x_t = \psi(B)w_t, \quad \text{for } \psi(B) = \phi(B)^{-1}\theta(B) \\ &\pi(B)x_t = w_t, \quad \text{for } \pi(B) = \theta(B)^{-1}\phi(B) \end{aligned}$$

3.3 Auto-covariance

- The auto-covariance for an MA(q) model was already given in (11). We can see that it is zero when $|h| > q$: this is a signature structure of the MA(q) model
- The auto-covariance for a causal AR(1) model was given in (6). We can see that it decays away from $h = 0$: this is a signature structure of AR models. But what does the precise auto-covariance look like for a general causal AR(p) model? How about a general causal ARMA(p, q) model?
- The answer is much more complicated, but still possible to characterize precisely. We'll do it first for an AR(p) model, and then look at an ARMA(p, q) model, which will have an analogous behavior

- For an AR(p) model (1), assumed causal (i.e., all roots of $\phi(z)$ are outside of the unit circle), we can focus on the auto-covariance at lags $h \geq 0$ (without a loss of generality, because γ_x is symmetric around zero),

$$\mathrm{Cov}(x_t, x_{t+h}) = \sum_{j=1}^{p} \phi_j \,\mathrm{Cov}(x_t, x_{t+h-j}) + \mathrm{Cov}(x_t, w_{t+h})$$

- The last term on the right-hand side zero for $h > 0$; recall, x_t is a causal process, and only depends on white noise in the past. On the other hand, when $h = 0$, writing $x_t = \sum_{j=0}^{\infty} \psi_j w_{t-j}$,

$$\mathrm{Cov}(x_t, w_t) = \sum_{j=0}^{\infty} \psi_j \,\mathrm{Cov}(w_{t-j}, w_t) = \sigma^2 \psi_0$$

- Combining the last two displays, we learn that the auto-covariance function satisfies

$$\gamma_x(h) - \sum_{j=1}^{p} \phi_j \gamma_x(h-j) = 0, \quad h > 0$$
$$\gamma_x(0) - \sum_{j=1}^{p} \phi_j \gamma_x(-j) = \sigma^2 \psi_0,$$

- This is called a *difference equation* of order p for the auto-covariance function γ_x. Some simple difference equations can be solved explicitly without complicated mathematics, but to solve a difference equation in general requires knowing something (again) about the roots of a certain complex polynomial associated with the difference equation, when it is represented in operator notation. You can read Chapter 3.2 of SS for an introduction to the theory of difference equations

- For an AR(p) model, it turns out we can solve the difference equation in the last display and get

$$\gamma_x(h) = P_1(h) z_1^{-h} + \cdots + P_r(h) z_r^{-h} \tag{21}$$

  where each P_j is a polynomial, and each z_j is a root of the AR polynomial ϕ. Because each $|z_j| > 1$ (all roots lie outside the unit circle), we see that the auto-covariance function will decay to zero as $h \to \infty$. In the case that some roots are complex, then what will actually happen is that the auto-covariance will dampen to zero but in doing so oscillate in a sinusoidal fashion

- Now what about an ARMA(p, q) model (16), assumed causal? Similarly, we can focus on the auto-covariance a lags $h \geq 0$,

$$\mathrm{Cov}(x_t, x_{t+h}) = \sum_{j=1}^{p} \phi_j \,\mathrm{Cov}(x_t, x_{t+h-j}) + \sum_{j=0}^{q} \theta_j \,\mathrm{Cov}(x_t, w_{t+h-j})$$

- The last term on the right-hand side zero for $h > q$, whereas when $h \leq q$, writing $x_t = \sum_{\ell=0}^{\infty} \psi_\ell w_{t-\ell}$, we see that for $j \geq h$,

$$\mathrm{Cov}(x_t, w_{t+h-j}) = \sum_{\ell=0}^{\infty} \psi_\ell \,\mathrm{Cov}(w_{t-\ell}, w_{t+h-j}) = \sigma^2 \psi_{j-h}$$

- Combining the last two displays, we learn that the auto-covariance function satisfies

$$\gamma_x(h) - \sum_{j=1}^{p} \phi_j \gamma_x(h-j) = 0, \qquad h > q$$
$$\gamma_x(h) - \sum_{j=1}^{p} \phi_j \gamma_x(h-j) = \sigma^2 \sum_{j=h}^{q} \psi_{j-h}, \quad h \leq q$$

- This is again a difference equation of order p that determines γ_x, but the boundary condition (what happens when $h \leq q$) is more complicated. Nonetheless, the solution is still the form (21), and the qualitative behavior is still the same as in the AR(p) case
- Figure 3, top row, shows sample auto-correlation functions for simple MA and AR models

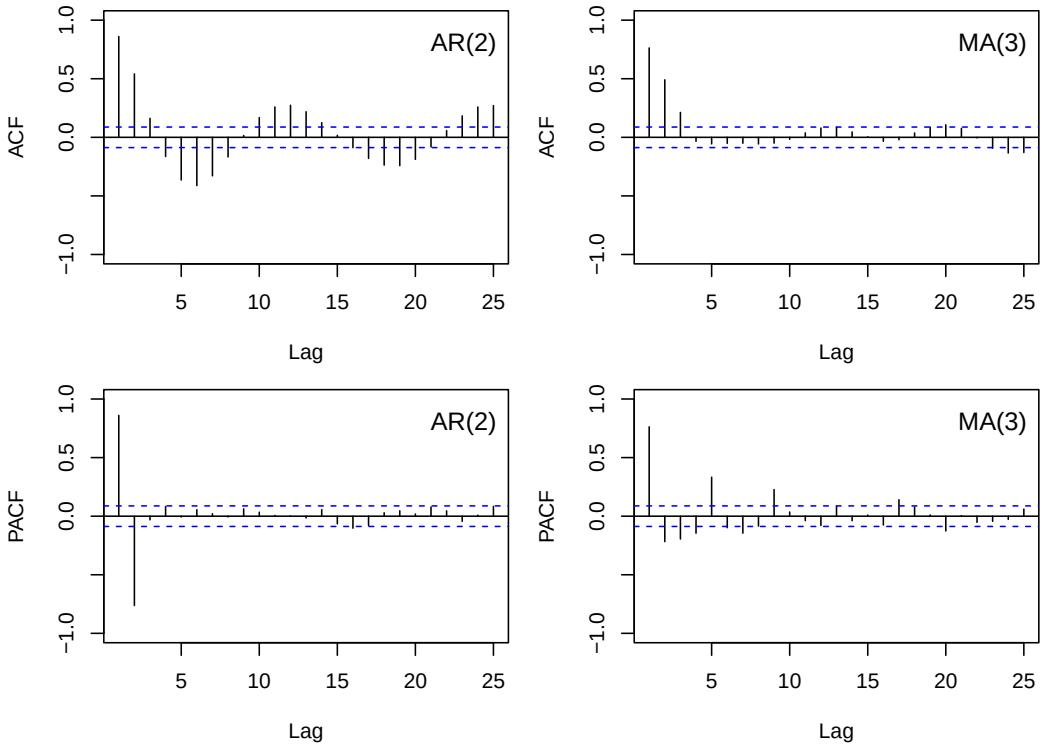

Figure 3: Top row: sample auto-correlation functions for data from AR(2) and MA(3) models. Bottom row: sample partial auto-correlation functions for these same data.

3.4 Partial auto-covariance

- The auto-covariance function for an MA model provides a considerable amount of information that will help identify its structure: since it is zero for lags $h > q$, if we were to compute a sample version based on data, then by seeing where the sample auto-correlation "cuts off", we could roughly identify the order q of the underlying MA process (see top right of Figure 3 again)
- For an AR (or ARMA) model, this is not the case. As we saw from (21), the auto-covariance decays to zero, but this tells us little about the AR order of dependence p (see also top left of Figure 3). Thus it is worth pursuing a type of *modified* correlation function for the AR model that behaves like the auto-correlation does for the MA model
- Such a modification will be given to us by the *partial auto-correlation function*. In general, the partial correlation between random variables X, Y given Z is denoted $\rho_{XY|Z}$ and defined as

$$\begin{aligned} &\rho_{XY|Y} = \mathrm{Cor}(X - \hat{X}, Y - \hat{Y}), \quad \text{where} \\ &\hat{X} \text{ is the linear regression of } X \text{ on } Z, \quad \text{and} \\ &\hat{Y} \text{ is the linear regression of } Y \text{ on } Z \end{aligned}$$

  Here, and in what follows, by "linear regression" we mean regression in the population sense, so that precisely $\hat{X} = Z^\mathsf{T} \mathrm{Cov}(Z)^{-1} \mathbb{E}(ZX)$ and $\hat{Y} = Z^\mathsf{T} \mathrm{Cov}(Z)^{-1} \mathbb{E}(ZY)$

- Said differently, the partial correlation of two random variables given Z is the correlation *after we remove ("partial out") the linear dependence of each random variable on Z*
- We note that when X, Y, Z are jointly normal, then this definition coincides with conditional correlation: $\rho_{XY|Z} = \text{Cor}(X, Y|Z)$, but not in general
- We are now ready to define the partial auto-correlation function for a stationary time series x_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$, denoted $\phi_x(h)$ at a lag h. Without a loss of generality we will only define it for $h \geq 0$, since it will be symmetric around zero (due to stationarity). First, at lag $h = 0$ or $h = 1$, we simply define:

$$\phi_x(0) = 1$$
$$\phi_x(1) = \text{Cor}(x_t, x_{t+1})$$

  Next, at all lags $h \geq 2$, we define:

$$\phi_x(h) = \text{Cor}(x_t - \hat{x}_t, x_{t+h} - \hat{x}_{t+h}), \quad \text{where}$$
$$\hat{x}_t \text{ is the linear regression of } x_t \text{ on } x_{t+1}, \ldots, x_{t+h-1}, \quad \text{and}$$
$$\hat{x}_{t+h} \text{ is the linear regression of } x_{t+h} \text{ on } x_{t+1}, \ldots, x_{t+h-1}$$

- To best see the effect of this definition we can go straight back to the causal AR(p) model. When $h > p$, it can be shown that the population linear regression $\hat{x}_{t+h}$, of x_{t+h} on $x_{t+1}, \ldots, x_{t+h-1}$, is

$$\hat{x}_{t+h} = \sum_{j=1}^{p} \phi_j x_{t+h-j}$$

  Thus $x_{t+h} - \hat{x}_{t+h} = x_{t+h} - \sum_{j=1}^{p} \phi_j x_{t+h-j} = w_{t+h}$, and the partial auto-correlation is

$$\phi_x(h) = \text{Cor}(x_t - \hat{x}_t, w_{t+h}) = 0$$

  because causality implies that x_t can only depend on white noise through time t, and $\hat{x}_t$ can only depend on white noise through time $t + h - 1$
- That is, the *partial auto-correlation function for an AR(p) model is exactly zero at all lags $h > p$*
- Figure 3, bottom row, shows sample partial auto-correlation functions for AR and MA models
- The table below summarizes the behavior of the auto-correlation function (ACF) and partial auto-correlation function (PACF) for causal AR(p) and invertible MA(q) models. By "tails off" we mean decays to zero as $h \to \infty$ without dropping to zero exactly; by "cuts off" we mean drops to zero at a given finite lag h

	AR(p)	MA(q)	ARMA(p, q)
ACF	tails off	cuts off at lag q	tails off
PACF	cuts off at lag p	tails off	tails off

- The fact that the partial auto-correlation function for an invertible MA(q) model "tails off" was not derived in these notes, but you can read more in Section 3.3 of SS if you are curious. Same with the behavior of a causal, invertible ARMA(p, q)

3.5 Estimation and selection

- Estimation in an ARMA(p, q) model—estimating the coefficients $\phi_1, \ldots, \phi_p, \theta_1, \ldots \theta_q$ in (16)—is in general fairly complicated. Much more so than in linear regression
- Estimation is usually performed by maximum likelihood (assuming Gaussian errors), but there are many other (nonequivalent) approaches, such as the method of moments. Maximum likelihood is no longer a simple least squares minimization (as it is for regression) over a linear parameterization. There are various approaches, typically iterative, for carrying out maximum likelihood, and different approaches will give different answers

- We won't cover estimation techniques in detail at all, but we'll just note that a simple approach is as follows (which dates back to Durbin in 1960). Start with some estimates $\hat{w}_t$ the noise variates w_t. Then use these as covariates, i.e., regress x_t on $x_{t-p}, \ldots, x_{t-1}, \hat{w}_{t-q}, \ldots, \hat{w}_{t-1}$, over $t = t_0 + 1, t_0 + 2, \ldots, n$, where $t_0 = \max\{p, q\}$. This is like a conditional maximum likelihood approach (where we condition on the initial values $x_1, \ldots, x_{t_0}, w_1, \ldots, w_{t_0}$ and our estimates $\hat{w}_t$)
- You can read about other approaches in Chapter 3.5 of SS, Chapter 9.6 of HA, or references therein
- R provides an `arima()` function in base R; in addition, both the `astsa` package (written by Stoffer of SS) and `fable` package (written by Hyndman of HA) provide functionality for ARIMA modeling
- If there's one thing even more complicated than fitting ARIMA models, it's choosing an ARIMA model—that is, *order selection*, or determining the choice of p, q from data
- At least, this topic seems to be more controversial ... some authors like Hyndman believe that this can be automated (via algorithms like the Hyndman-Khandakar algorithm), and this is what is implemented in `ARIMA()` in the `fable` package when the order p, q is left unspecified. This kind of automated model building is also central to some ML perspectives on forecasting (cf. auto ML). But other authors like Stoffer believe that this doesn't really work,[1] and recommend more human-expert-driven model building
- If the point is to identify the "right" structure, from a model specification point of view, then Stoffer may have a point (see R notebook examples). But to HA's credit, their Chapter 9.8 does also recommend more of a human-in-the-loop procedure than just calling `ARIMA()` once, involving diagnostics
- The gist of the non-automated part (which is fairly standard) is as follows:

  0. Plot the data to identify (and possibly remove) any outliers. Apply data transformation (e.g., Box-Cox), if needed, to stabilize the variance
  1. If the data appears nonstationary, take differences until it is stationary
  2. Plot the ACF and PACF to determine possible MA and AR orders
  3. Fit an ARMA model, inspect the residuals: they should look like white noise

- Step 1 is at the heart of ARIMA, which we'll cover soon. Steps 2-3 are the part that can be (but arguably, should not be) automated by Hyndman-Khandakar: instead of a single ARMA model, it fits a number of different ARMA models, using a stepwise search procedure, and then uses an information criterion (like AICc) to select a final one
- Chapter 3.7 in SS also goes into details about a similar sequence of steps for building ARIMA models, with worked examples. We'll also go through an example shortly, in the ARIMA section
- The ACF and PACF are great tools, and looking at them to get a sense of MA and AR dependence is generally helpful, but we will not concern ourselves too much with the formality of ARMA order selection (just like we did not with model selection in regression)
- Since our focus is on prediction, we will adopt the following simple perspective (just like in regression): *an ARMA model is useful if it predicts well.* And as before, we can assess this with time series cross-validation. We'll go through an example later, in the forecasting section

3.6 Regression with correlated errors

- Very briefly, we describe regression with auto-correlated errors. Suppose we assume a model

$$y_t = \sum_{j=1}^{k} x_{tj}\beta_j + z_t, \quad t = 1, \ldots, n$$

  where instead of white noise, the error sequence z_t, $t = 1, \ldots, n$ has some ARMA structure

[1] See https://github.com/nickpoison/astsa/blob/master/fun_with_astsa/fun_with_astsa.md#arima-estimation.

- In the case that the noise was AR(p), with associated operator $\phi(B)$, we could then simply apply this operator to both sides, to yield

$$\phi(B)y_t = \sum_{j=1}^k \phi(B)x_{tj}\beta_j + \phi(B)z_t, \quad t = 1, \ldots, n$$

or defining $y'_t = \phi(B)y_t$, $x'_{tj} = \phi(B)x_{tj}$, $w_t = \phi(B)z_t$,

$$y'_t = \sum_{j=1}^k x'_{tj}\beta_j + w_t, \quad t = 1, \ldots, n$$

where now w_t, $t = 1, \ldots, n$ is white noise

- If we knew the coefficients $\phi_1, \ldots, \phi_p$ that comprise the AR operator $\phi(B)$, then we could just obtain estimates of the regression coefficients $\beta_1, \ldots, \beta_k$ by regressing y'_t on x'_t. But since we don't know the coefficients $\phi_1, \ldots, \phi_p$ in general, these would need to be estimated as well

- We could solve for $\beta_1, \ldots, \beta_k$ and $\phi_1, \ldots, \phi_p$ jointly using maximum likelihood, or least squares minimization (which are not equivalent). For example, the latter would solve

$$\min_{\beta \in \mathbb{R}^k, \phi \in \mathbb{R}^p} \sum_{t=1}^n \bigg(\phi(B)y_t - \sum_{j=1}^k \phi(B)x_{tj}\beta_j \bigg)^2$$

which is called a nonlinear least squares problem (since each square is applied to a nonlinear function of the parameters β, ϕ)

- For (invertible) ARMA noise, the same approach carries over but with $\pi(B) = \phi(B)^{-1}\theta(B)$ in place of $\phi(B)$, which only makes the nonlinear least squares optimization much more complicated

- Identifying the ARMA structure in regression errors can be done by fitting a regression with regular (white noise) errors, and then applying the same ideas as those described above for order selection in ARMA to the residuals (inspect the ACF and PACF of residuals, and so on). For more, you can read Chapter 3.8 of SS and Chapters 10.1-10.2 of HA. In R, the `ARIMA()` function in the `fable` package allows us to fit regression models with ARIMA errors

4 ARIMA models

- Finally, we arrive at ARIMA models. We've hinted at what these are about a few times already, but it is nonetheless worth making the motivation explicit: *the main point behind the new "I" component here is to account for nonstationary.* Well-behaved ARMA models (causal ones) are stationary, and ARIMA allows us to handle nonstationary data

- The "I" stands for "integration", so an ARIMA model is an autoregressive *integrated* moving average model. Integration is to be understood here as the inverse of differencing, because we are effectively just differencing the data to render it stationary, then assuming the differenced data follows ARMA

- First, we define the differencing operator ∇ that takes a given sequence and returns pairwise differences,

$$\nabla x_t = x_t - x_{t-1}$$

- We can extend this to powers by iterating, as in

$$\nabla^2 x_t = \nabla\nabla x_t = x_t - 2x_{t-1} - x_{t-2}$$

- Note that we can also write ∇ in terms of the backshift operator B as $\nabla = 1 - B$, so that a general d^{th} order difference is

$$\nabla^d = (1 - B)^d$$

- Now we can formally define, an ARIMA(p, d, q) model, for orders $p, d, q \geq 0$: this is a model for x_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$ such that $\nabla^d x_t$ follows an ARMA(p, q) model, i.e.,

$$\phi(B)\nabla^d x_t = \theta(B)w_t \tag{22}$$

  where w_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$ is a white noise sequence, and $\phi(B), \theta(B)$ are the AR and MA operators, respectively, as before. In other words, x_t is given by d^{th} order integration of an ARMA(p, q) sequence

- Note that ARIMA(0,1,0) says that the differences in the sequence are white noise,

$$x_t = x_{t-1} + w_t$$

  which is nothing more than a random walk, which we already know is nonstationary (the variance grows over time)

- Below is a summary of some of the basic models that the ARIMA framework encompasses. We write c for the intercept in the model, as in

$$\phi(B)\nabla^d x_t = c + \theta(B)w_t \tag{23}$$

  Recall, this was assumed zero in (22), and throughout, only for simplicity—in general, we allow it

White noise	ARIMA(0,0,0) with $c = 0$
Random walk	ARIMA(0,1,0) with $c = 0$
Random walk with drift	ARIMA(0,1,0) with $c \neq 0$
Autoregressive	ARIMA$(p, 0, 0)$
Moving average	ARIMA$(0, 0, q)$

- A general warning should be given about choosing large d; HA say that in practice, $d > 2$ is never really needed, and also give a cautionary note about taking $d = 2$ with $c \neq 0$ (more later)

- Now we walk through an example from HA on using ARIMA to model (and forecast) Central African Republic exports. The data is shown in Figure 4, top row. It does not look stationary

- Taking first differences of the data, as shown in the second row, renders the data reasonably stationary-looking

- ACF and PACF plots are shown in the third row of Figure 4. To be clear, these are applied to first differences of the data. They are suggestive of MA(3) and AR(2) dynamics, respectively. Thus we can go and fit each model: ARIMA(2,1,0) and ARIMA(0,1,3) (note that these are ARIMA models with $d = 1$, since we are going to specify the differencing as part of the model itself). We can also use the Hyndman-Khandakar algorithm. This ends up choosing ARIMA(2,1,2)

- For prediction-focused tasks (and probably/necessarily, longer sequences), we would look at an estimate of (say) MAE using time series cross-validation to help us choose a model. Here, HA recommend looking at AICc, which is what is also built into the `fable` package's reporting of the `ARIMA()` output. This leads us to choose ARIMA(2,1,0). The residuals from this model also look close to white noise (see the R notebook for diagnostics) which is good

- Forecasts from our fitted ARIMA(2,1,2) model are shown in the rop row of Figure 5, for a 5-year horizon. Also plotted are prediction intervals around the forecast (more on this in the last section). Qualitatively, the forecasts aren't very impressive at first glance—they look more or less like what we'd get if we used a random walk (zero mean function, growing variance function), which is just ARIMA(0,1,0)

- However, HA comment that the prediction intervals here are narrower than they would be for a random walk, which is due to the contribution of the nontrivial AR and MA components that we've been able to pick up and model. To check this, we plot forecasts from ARIMA(0,1,0) in the bottom row of Figure 5, and indeed you can see they have wider uncertainty bands

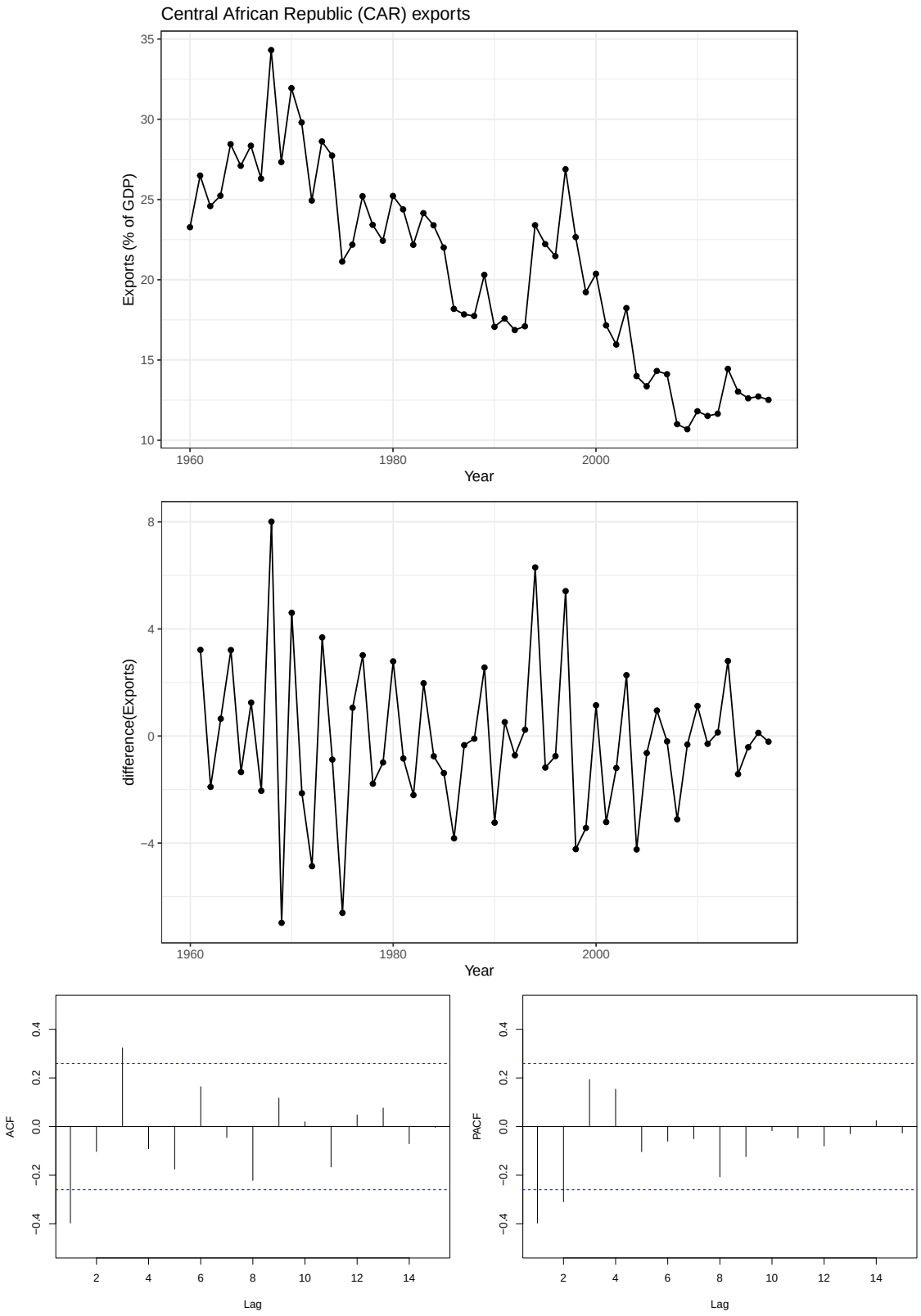

Figure 4: Top row: exports (as a percentage of GDP) for Central African Republic (from HA). Middle row: first differences of exports. Bottom row: ACF and PACF functions of first differences.

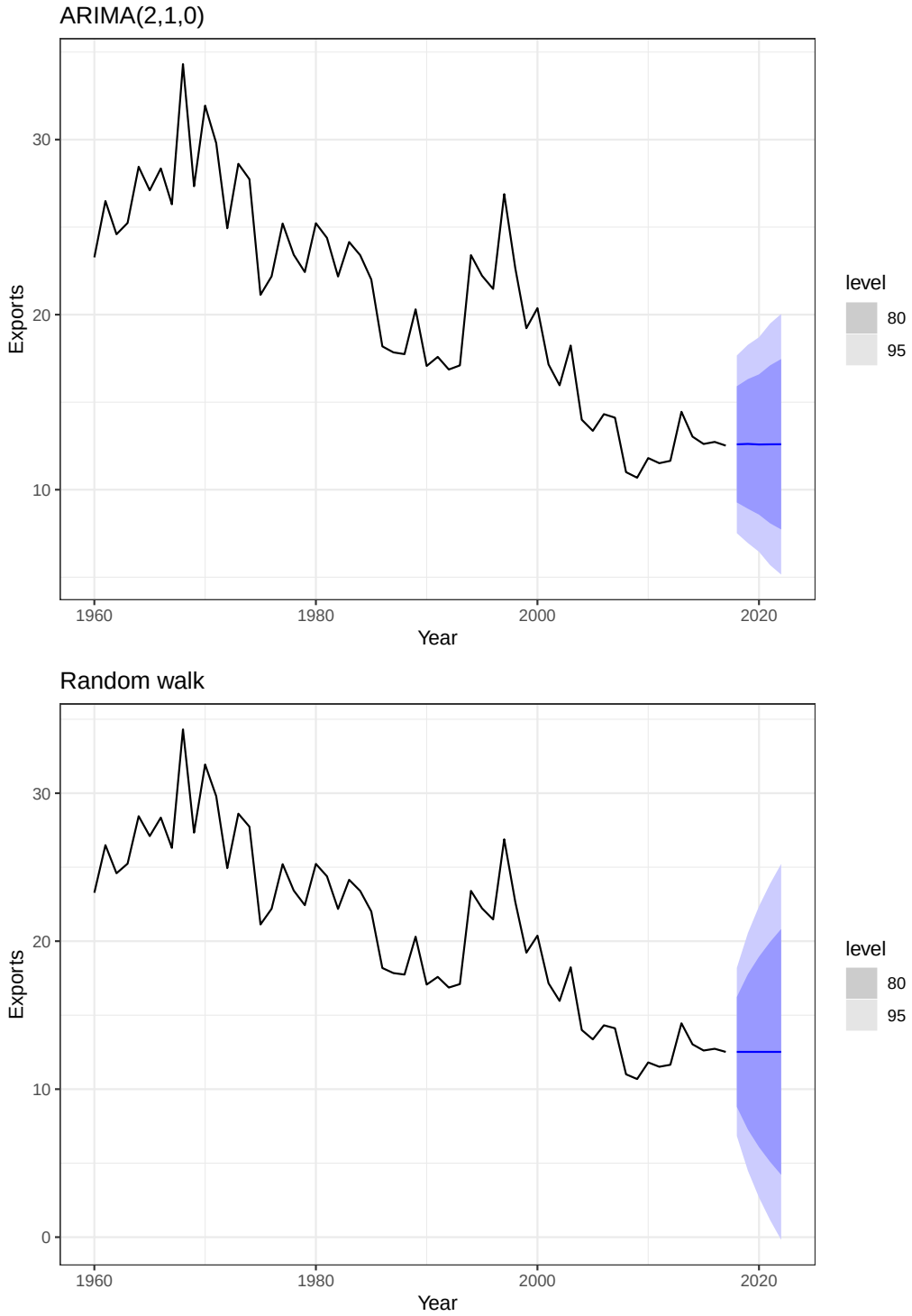

Figure 5: Top row: forecasts from ARIMA(2,1,2). Bottom row: forecasts from ARIMA(0,1,0) with $c = 0$, i.e., a random walk.

- A general note to close out our example: the `ARIMA()` function in the `fable` package can be handy *but also a bit dangerous* because it may automate away some part of the model building procedure even when you think you've specified a model manually. Check out the R notebook for an example and guidance on how to explicitly get what you want ...

4.1 Seasonality extensions

- Seasonality is typically handled in ARIMA models by seasonal differencing. This assumes that the seasonal periods are known
- For example, let's suppose that the seasonal period is known to be s. Then, to start off as simple as possible, a *purely seasonal ARMA* model with orders $P, Q \geq 0$ and period s, denoted ARMA$(P,Q)_s$, is of the form
$$\Phi(B^s)x_t = \Theta(B^s)w_t \tag{24}$$
where $\Phi(B^s) = 1 - \Phi_1 B^s - \Phi_2 B^{2s} - \cdots - \Phi_P B^{Ps}$ and $\Theta(B) = 1 + \Theta_1 B + \Theta_2 B^{2s} + \cdots + \Theta_Q B^{Qs}$. Note that these model AR and MA dynamics, but on the scale of seasons: since all backshifts are in multiples of s
- To select orders for seasonal dynamics we would follow ideas just like those for regular (nonseasonal) ARMA models: look at ACF and PACF plots, *but for lags on the seasonal scale*, in multiples of s
- To make this more sophisticated, we can add in AR and MA dynamics on the original scale, which leads to a *seasonal ARMA* model of orders $p, q, P, Q \geq 0$ and period s, denoted ARMA$(p,q)(P,Q)_s$,
$$\phi(B)\Phi(B^s)x_t = \theta(B)\Theta(B^s)w_t \tag{25}$$
where $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$ and $\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q$ are the AR and MA operators, as before, and $\Phi(B^s), \Theta(B^s)$ are the seasonal operators, as above
- The full scale sophistication is achieved by adding in integration to account for nonstationarity. However, now we can also account for nonstationarity on the seasonal scale; first define a D^{th} order seasonal difference eperator
$$\nabla_s^D = (1 - B^s)^D$$
Then define the *seasonal ARIMA* (SARIMA) model of orders $p, d, q, P, D, Q \geq 0$ and period s, denoted ARIMA$(p,d,q)(P,D,Q)_s$,
$$\phi(B)\Phi(B^s)\nabla^d\nabla_s^D x_t = \theta(B)\Theta(B^s)w_t \tag{26}$$
As usual, we are omitting an intercept c for simplicity, but in general we could include this on the right-hand side in (26). (Phew! That's the full beast. We've maxed out in complexity, at least for this lecture ...)
- Estimation is even more complicated in SARIMA models (more parameters, more nonlinearity) but thankfully software does it for us
- We won't go into any further details than what we have discussed already, but will walk through an example from HA to model (and forecast) US employment numbers in leisure/hospitality. The data is shown in Figure 6, top row. It has a clear seasonal pattern with (safe guess) a 1 year period
- Taking seasonal differences (i.e., applying ∇_s with $s = 12$, since this is monthly data) has removed the seasonality, but the result, in the middle row of Figure 6, is still very clearly nonstationary
- Additionally taking monthly differences (i.e., applying the composite operator $\nabla\nabla_s$), in the bottom row of Figure 6, is reasonably stationary-looking
- ACF and PACF plots are displayed in the top row of Figure 7. To be clear, these are applied to the differences of yearly differences ($\nabla\nabla_s x_t$, $t = 1, 2, 3, \ldots$, as plotted in the bottom row of Figure 7). The ACF shows two moderate correlations at lags 1 and 2 and then a big spike at lag 12. This is suggestive of an nonseasonal MA(2) and a seasonal MA(1), and thus we can go and fit an

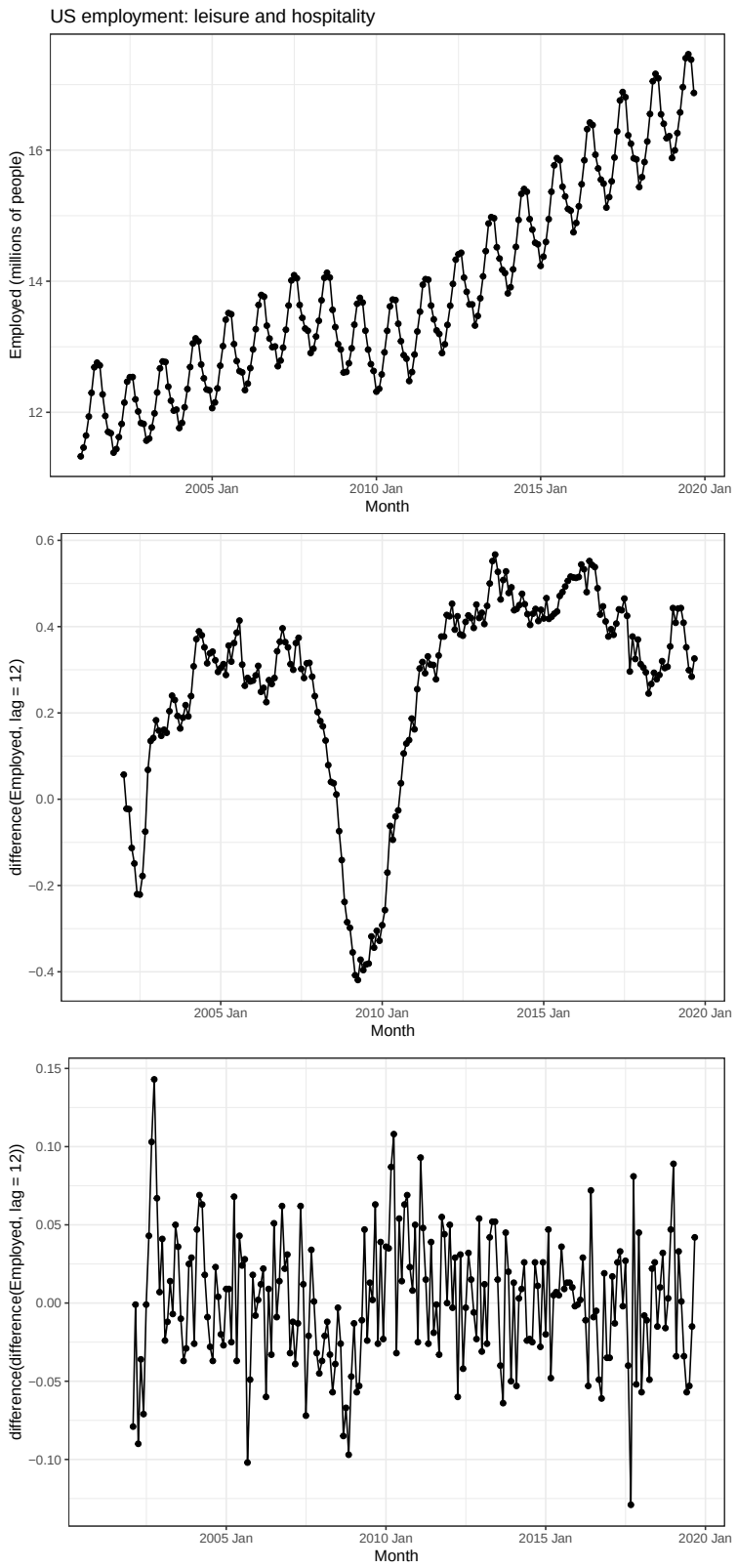

Figure 6: Top row: number of US jobs (in millions) in leisure/hospitality (from HA). Middle row: seasonal (yearly) differences. Bottom row: differences of seasonal differences.

ARIMA$(0,1,2)(0,1,1)_{12}$ model. Meanwhile, the PACF shows again to moderate correlations at lags 1 and 2 and a big spike at lag 12. This suggests a nonseasonal AR(2) and a seasonal AR(1), so we can go and fit an ARIMA$(2,1,0)(1,1,0)_{12}$ model. Lastly, we can also try the Hyndman-Khandakar algorithm. This ends up choosing ARIMA$(1,1,1)(2,1,1)_{12}$

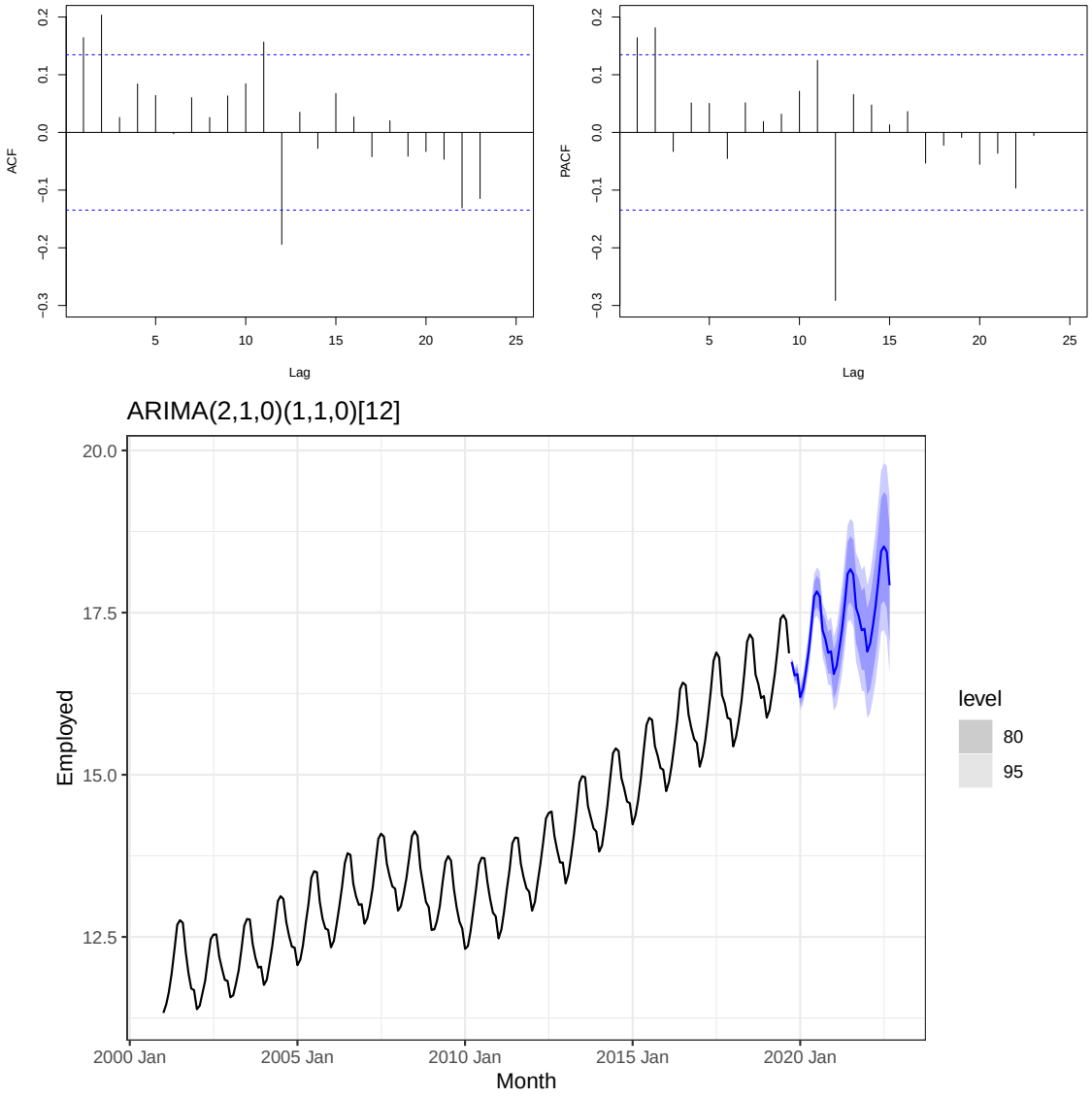

Figure 7: Top row: ACF and PACF functions of differences of seasonal differences. Bottom row: forecasts from ARIMA(2,1,2).

- AICc (see the R notebook for details and diagnostics) tells us to choose the ARIMA$(1,1,1)(2,1,1)_{12}$ model. But that seems a bit doubtful because it suggests there is a second-order autoregressive dynamic on the yearly scale, which makes the model much more complicated. As usual, if we cared about prediction, then we should probably just use time series cross-validation to evaluate the different models. More on this in the forecasting section. For now, we'll just produce 3-year forecasts from the ARIMA$(0,1,2)(0,1,1)_{12}$ model
- These are shown in the bottom row of 7. They actually seem fairly plausible and interesting. Note that the seasonality was nicely captured by the seasonal differencing, and the increasing trend seems to have been nicely captured by the nonseasonal differencing

5 Forecasting

- ARIMA models, especially with seasonality extensions, can be powerful forecasters, as we've seen in a couple of examples. Some authors treat torecasting with ARIMA in a complex and notation-heavy way, but in fact it can be explained intuitively (which we borrow from HA)
- As before, we will use the notation $\hat{x}_{t+h|t}$ to denote a forecast of x_{t+h} made using data up through t
- Some potentially helpful nomenclature: $t+h$ here is called the *target time* (time of the target we are trying to forecasting), and t is called the *forecast time* (time at which we make the forecast)
- Obtaining the forecast $\hat{x}_{t+h|t}$ from an ARIMA model can be done by iterating the following steps:
  1. Start with the ARIMA equation
     $$\phi(B)\Phi(B^s)\nabla^d\nabla_s^D x_t = c + \theta(B)\Theta(B^s)w_t$$
     Plug in for parameter estimates, and rearrange so that x_t is on the left-hand side an all other terms are on the right-hand side
  2. Rewrite the equation by replacing t with $t+h$
  3. On the right-hand side of the equation, replace future observations (x_{t+k}, $k \geq 1$) with their forecasts, future errors (w_{t+k}, $k \geq 1$) with zero, and past errors (w_{t-k}, $k \geq 0$) with their ARIMA residuals
- Beginning with $h = 1$, we iterate this procedure for $h = 2, 3, \ldots$ as needed to get the forecasts at the ultimate horizon we desire
- It helps to walk through an example. Consider forecasting with a fitted ARIMA(3,1,1) model, which we start by writing as
  $$(1 - \hat{\phi}_1 B - \hat{\phi}_2 B^2 - \hat{\phi}_3 B^2)(x_t - x_{t-1}) = \hat{c} + (1 + \hat{\theta}_1 B)w_t$$
- We expand this as
  $$x_t - \hat{\phi}_1 x_{t-1} - \hat{\phi}_2 x_{t-2} - \hat{\phi}_3 x_{t-3} - (x_{t-1} - \hat{\phi}_1 x_{t-2} - \hat{\phi}_2 x_{t-3} - \hat{\phi}_3 x_{t-4}) = \hat{c} + w_t + \hat{\theta}_1 w_{t-1}$$
- And rearrange as
  $$x_t = \hat{c} + (1 + \hat{\phi}_1)x_{t-1} + (\hat{\phi}_2 - \hat{\phi}_1)x_{t-2} + (\hat{\phi}_3 - \hat{\phi}_2)x_{t-3} - \hat{\phi}_3 x_{t-4} + w_t + \hat{\theta}_1 w_{t-1}$$
- To make a 1-step ahead forecast, we first replace t by $t+1$ above:
  $$x_{t+1} = \hat{c} + (1 + \hat{\phi}_1)x_t + (\hat{\phi}_2 - \hat{\phi}_1)x_{t-1} + (\hat{\phi}_3 - \hat{\phi}_2)x_{t-2} - \hat{\phi}_3 x_{t-3} + w_{t+1} + \hat{\theta}_1 w_t$$
  and then we set $w_{t+1} = 0$, and replace w_t by $\hat{w}_t$, which is the residual from our fitted ARIMA model, yielding
  $$\hat{x}_{t+1|t} = \hat{c} + (1 + \hat{\phi}_1)x_t + (\hat{\phi}_2 - \hat{\phi}_1)x_{t-1} + (\hat{\phi}_3 - \hat{\phi}_2)x_{t-2} - \hat{\phi}_3 x_{t-3} + \hat{\theta}_1 \hat{w}_t$$
- To make a 2-step ahead forecast, we similarly start with
  $$x_{t+2} = \hat{c} + (1 + \hat{\phi}_1)x_{t+1} + (\hat{\phi}_2 - \hat{\phi}_1)x_t + (\hat{\phi}_3 - \hat{\phi}_2)x_{t-1} - \hat{\phi}_3 x_{t-2} + w_{t+2} + \hat{\theta}_1 w_{t+1}$$
  and then we set $w_{t+2} = w_{t+1} = 0$, and replace x_{t+1} by $\hat{x}_{t+1|t}$, yielding
  $$\hat{x}_{t+2|t} = \hat{c} + (1 + \hat{\phi}_1)\hat{x}_{t+1|t} + (\hat{\phi}_2 - \hat{\phi}_1)x_t + (\hat{\phi}_3 - \hat{\phi}_2)x_{t-1} - \hat{\phi}_3 x_{t-2}$$
- This process can be iterated any number of times to obtain forecasts arbitrarily far into the future

5.1 Behavior of long-term forecasts

- *Should* we keep iterating the process above to generate forecasts arbitrarily far into the future?
- The answer is almost certainly "no", in most applications! You should be more careful. Short of detecting and projecting seasonality, ARIMA models are limited for long-term forecasts (in general, long-term forecasts are very hard without precise domain-specific knowledge of what the long-range dynamics look like ... and even then, many domains to not offer much to say there ...)
- It is important to have a qualitative sense of what long-term forecasts look like in ARIMA. Again, we borrow this nice explanation from HA. Starting with (23), a nonseasonal ARIMA model, we can break down the explanation into cases (recall c is the intercept, and d is the differencing order):
  1. If $c = 0$ and $d = 0$, then the long-term forecasts will converge to zero
  2. If $c = 0$ and $d = 1$, then the long-term forecasts will converge to a nonzero constant
  3. If $c = 0$ and $d = 2$, then the long-term forecasts will follow a linear trend
  4. If $c \neq 0$ and $d = 0$, then the long-term forecasts will converge to the historical mean
  5. If $c \neq 0$ and $d = 1$, then the long-term forecasts will follow a linear trend
  6. If $c \neq 0$ and $d = 2$, then the long-term forecasts will follow a quadratic trend (can be dangerous! some software packages do not allow $d \geq 2$ when $c \neq 0$ ...)
- You will verify this on your homework for some simple ARIMA models, where you'll also work out the qualitative behavior for long-term forecasts from seasonal ARIMA models

5.2 Split-sample and cross-validation

- At last, we arrive to our bread-and-butter tools for evaluating ARIMA models for their predictive accuracy: split-sample and time series cross-validation (CV)
- They apply exactly as described in the regression lecture: to recap, in split-sample validation, we pick a time t_0, train our model (regression, ARIMA, whatever) on data up through t_0, and then we evaluate forecasts $\hat{x}_{t|t_0}$ over times $t > t_0$:

$$\text{SplitMSE} = \frac{1}{n - t_0} \sum_{t=t_0+1}^{n} \left(\hat{x}_{t|t_0} - x_t\right)^2$$

  In time series CV, we use data before t_0 as a burn-in period (usually we would choose a smaller t_0 here than for split-sample) to ensure the initial model is decent, and then iteratively retrain at each $t > t_0$ in order to make forecasts. For h-step ahead forecasts:

$$\text{CVMSE} = \frac{1}{n - t_0} \sum_{t=t_0+1}^{n} \left(\hat{x}_{t|t-h} - x_t\right)^2$$

  (Instead of MSE, we could also use MAE, or MAPE, or MASE as our metric ...)
- There is not much more to say other than to reiterate that these are our most robust (assumption-lean) tools for assessing utility in a predictive context. You'll get continued practice with time series CV on your homework. The `fable` package that we'll use for a lot of forecasting tools has a handy (though memory-inefficient) way of implementing this, which saves you from writing a loop by hand
- There is one important warning to give however! This applies to cross-validation, generally, in any setting. Cross-validation evaluates the prediction error of *an entire procedure*. Thus the entire procedure *must be rerun* each time predictions are to be validated. So, e.g., if we are using auto-ARIMA (as implemented by `ARIMA()` in the `fable` package) to select an ARIMA order for us, then we must rerun auto-ARIMA at each time $t > t_0$ in order to evaluate forecasts in the last display. (And, note, it will generally pick different orders at each $t > t_0$ ... the way to think about it is that we are evaluating the prediction error of auto-ARIMA *procedure* itself, not any particular selected order)

- In other words, we *cannot* run auto-ARIMA once over all time, and then take the selected ARIMA order and apply this retrospectively. That would not yield an honest estimate of prediction error (since we would have used the future at $t > t_0$ to select the ARIMA order). Summary:

Wrong way:	Run auto-ARIMA at time n (all time) to select p, d, q, and use this retrospectively in time series CV to fit ARIMA(p, d, q) at each $t > t_0$ and make forecasts
Right way:	Run auto-ARIMA at each $t > t_0$ to select (time-dependent) p, d, q, and use this to fit ARIMA(p, d, q) and make forecasts in time series CV

- (Note: we are not advertising or advocating for the use of auto-ARIMA, but are simply explaining that this would be the right way to validate it)
- Finally, as a rule-of-thumb, we recall Occam's razor: if a number of models all have reasonably similar split-sample or cross-validated prediction error, then we would generally prefer to use the simplest among them

5.3 Prediction intervals

- The predictions $\hat{x}_{t+h|t}$ from ARIMA described above are known as *point forecasts*. But if you look back at Figures 5 and 7, you can see bands around the point forecasts (which are the dark lines)
- These bands represent *prediction intervals*, which are typically quite important in practice, since they reflect uncertainty in the predictions that are made (and can be useful for downstream decision-making)
- There are various ways to compute prediction intervals. One method is to obtain an estimate $\hat{\sigma}_h^2$ of the variance of the "h-step ahead forecast distribution". This is in quotes because we have not defined this precisely, and we will avoid going into details because this approach is not always tractable, and also requires more assumptions about the ARIMA model, including (typically) normality of the noise distribution. But in any case, having computed such an estimate, we could model the h-step ahead forecast distribution at time t as
$$N\big(\hat{x}_{t+h|t}, \hat{\sigma}_h^2\big)$$
and compute prediction intervals accordingly. For example, to compute a central 90% prediction interval, we would use
$$\big[\hat{x}_{t+h|t} - \hat{\sigma}_h q_{0.95},\, \hat{x}_{t+h|t} + \hat{\sigma}_h q_{0.95}\big]$$
where $q_{0.95}$ is the 0.95 quantile of the standard normal distritbuion
- Another method is adapt the method described previously to iteratively produce point forecasts to instead iteratively produces samples paths from the forecast distribution. This is what is done in the `fable` package's `forecast()` function when `bootstrap = TRUE`. The gist of the idea is as follows: in step 3, instead of replacing future errors (w_{t+k}, $k \geq 0$) by zero, we replace them by a bootstrap draw (i.e., a uniform sample with replacement) from the empirical distribution of past residuals. Given a bunch of sample paths, we can then read off empirical quantiles at $t + h$ for a prediction interval
- Unfortunately, neither of these methods (nor any other traditional methods) for producing prediction intervals are guaranteed to actually give *coverage* in practice. That is, you would hope that our 95% prediction intervals actually cover 95% of the time, but this need not be the case
- Fortunately, we will learn that there are some relatively simple correction methods that can act as post-processors to endow any given sequence of prediction intervals with long-run coverage. We will talk about this near the end of the course when we discuss calibration

5.4 ARIMAX models

- In practice, including auxiliary features in an ARIMA model can make it much more powerful. This is true in general: finding good predictors can make a huge difference in forecasting!

- Auxiliary features are often referred to as *exogenous* in the context of time series models, and an ARIMA model with exogenous predictors is often abbreviated ARIMAX
- Formally, an ARIMAX(p, d, q) model is an extension of the ARIMA model (22) of the form

$$\phi(B)\nabla^d x_t = \beta^\mathsf{T} u_t + \theta(B) w_t$$

  where u_t, $t = 0, \pm 1, \pm 2, \pm 3, \ldots$ is an exogenous predictor sequence
- Forecasting with ARIMAX models is a little more complicated because, for "true" ex-ante forecasts, we will typically not have the values of the exogenous predictor available that we need (think about the steps we outlined for forecasting with ARIMA models, and how we substituted future observations with their forecasted values—unless we build a separate forecast model for u_t, we will not know how to impute them)
- As usual (as we described in the regression lecture), we can use lagged exogenous features as predictors in order to circumvent this issue
- (Note: be careful not to confuse an ARIMAX model with a regression with correlated errors! They are not the same thing. In both models, there are exogenous predictors and and errors with MA dynamics, but the distinguishing factor is the AR dynamic: in an ARIMAX model, the "response", which is x_t in our notation here, exhibits AR dynamics; in a regression with correlated errors, the errors exhibit AR dynamics)

5.5 IMA models

- Forecasting with the ARIMA(0,1,1) model, also written as IMA(1,1), bears an interesting connection to what is called *simple exponential smooting* (SES). This is also sometimes called an *exponentially weighted moving averages* (EWMA) forecaster. It tends to be popular in economics (SS call it "frequently used and abused")
- To develop the connection, consider the IMA model with an MA coefficient of $\theta = -(1-\alpha)$, where $0 < \alpha < 1$. We can write this as

$$x_t - x_{t-1} = w_t - (1-\alpha) w_{t-1}$$

- Because $\alpha < 1$, this has an invertible representation: writing $y_t = x_t - x_{t-1}$, recall what we developed in (14), which yields

$$w_t = \sum_{j=0}^{\infty} (1-\alpha)^j y_{t-j}$$

  or

$$y_t = -\sum_{j=1}^{\infty} (1-\alpha)^j y_{t-j} + w_t$$

- Substituting $y_t = x_t - x_{t-1}$, and rearranging, this gives

$$\begin{aligned}
x_t &= x_{t-1} - \sum_{j=1}^{\infty} (1-\alpha)^j (x_{t-j} - x_{t-j-1}) + w_t \\
&= x_{t-1} + \sum_{j=1}^{\infty} (1-\alpha)^j x_{t-j-1} - \sum_{j=1}^{\infty} (1-\alpha)^j x_{t-j} + w_t \\
&= \sum_{j=1}^{\infty} (1-\alpha)^{j-1} x_{t-j} - \sum_{j=1}^{\infty} (1-\alpha)^j x_{t-j} + w_t \\
&= \sum_{j=1}^{\infty} \alpha (1-\alpha)^{j-1} x_{t-j} + w_t
\end{aligned}$$

- The 1-step ahead prediction from this model is therefore

$$\begin{aligned}
\hat{x}_{t+1|t} &= \sum_{j=1}^{\infty} \alpha(1-\alpha)^{j-1} x_{t+1-j} \\
&= \alpha x_t + \sum_{j=2}^{\infty} \alpha(1-\alpha)^{j-1} x_{t+1-j} \\
&= \alpha x_t + \sum_{j=1}^{\infty} \alpha(1-\alpha)^{j} x_{t-j} \\
&= \alpha x_t + (1-\alpha) \sum_{j=1}^{\infty} \alpha(1-\alpha)^{j-1} x_{t-j} \\
&= \alpha x_t + (1-\alpha) \hat{x}_{t|t-1}
\end{aligned}$$

- In other words, the 1-step ahead prediction is a weighted combination of the current observation x_t and the previous prediction $\hat{x}_{t|t-1}$
- Smaller values of α (closer to 0) lead to smoother forecasts, since we are "borrowing" more from the previous forecast, in the last display
- This is the simplest in a class of methods based on exponential smoothing, which we will cover in the next lecture