Time Series Analytics Notes

Some Remarks on ARIMA

ARIMA is a strategy for time series modeling of univariate time series. Also called "Box-Jenkins" time series analysis, after two of its main champions, statisticians George Box and Gwilym Jenkins,[1] the term ARIMA is an acronym for **A**uto**R**egressive **I**ntegrated **M**oving Average. There are three parts in the name, which should be thought of separately: **AR** for autoregression, **MA** for moving average, and **I** for integration. These are three types of statistical models that can be used – singly, or in conjunction with each other. The "integration" has nothing to do with calculus. Instead, it refers to differencing (changes), which have a big role in the methodology.

You should be aware that ARIMA is complex. A quick glance at the documentation for SAS's PROC ARIMA should convince you of that. The tool is powerful, but requires skill that comes from practice. A good way to acquire that skill is to start trying simple ARIMA models cautiously and conservatively. In these notes, I will limit myself to explaining some of the ideas in ARIMA. You can apply these ideas "by hand" in time series analysis so that you can see how they work in situations in which you should feel comfortable and able to diagnose when things may go wrong. At the end of this document are a few comments on using ARIMA in SAS. Even if you do not have access to powerful forecasting software like SAS, you can profit from awareness of the ARIMA strategy in making your own forecasting models. This is because much of what ARIMA software does automatically, you can do "by hand" with small to moderate sized datasets in Excel or other general purpose software – although more laboriously.

The ARIMA strategy has three phases:

1. Identification
2. Estimation and diagnostics
3. Forecasting

These three phases correspond roughly to the three general steps of statistical modeling:

1. Propose the model
2. Validate the model
3. Use the model

Let us suppose that Y_t is a time series. The first step in ARIMA modeling is to reduce Y_t to a *stationary* time series by removing any trend and seasonal components and stabilizing the variance. A time series is **stationary** if it has a constant mean level and constant variance.[2] You will recognize this as the L and H specifications. Removing the trend is called **detrending**. Removing the seasonality is called **deseasonalizing**. If you have both trend and seasonality in your data, you can deal with these two features sequentially or simultaneously. If sequentially,

[1] Their original work is *Time Series Analysis: Forecasting and Control*, 1976, by George Box and Gwilym Jenkins, Holden-Day.

[2] Actually, stationarity requires a little more: the lag k autocorrelation must not evolve over time for any k. E.g., the lag 2 autocorrelation is the same later on as early on.

you should detrend first and then deseasonalize the detrended data. If you deseasonalize first, your seasonal estimates will be contaminated by the trends.[3]

There are various ways to detrend. One way starts with fitting a regression model as a function of time to match the overall direction of the trend of the plotted data. Call the regression fits the **trend effect** and the regression residuals the **detrended data**. This means that you get the detrended data by subtracting the regression equation from Y_t. This "takes out" the trend from the data.

There are various ways to deseasonalize after detrending. One way starts by sorting the detrended data by season. For example, put all of the detrended Januaries together, all of the detrended Februaries together, etc. Then average each season of detrended data. Call the average of each season of detrended data the **seasonal effect**. For example, the mean of the detrended Januaries is the January effect, etc. Finally, subtract the appropriate seasonal effect from each detrended data value to get the **deseasonalized** detrended data value.

At the end of this process, you will have

Y_t = **trend effect + seasonal effect + deseasonalized detrended data value**

This equation emphasizes the useful intuition of thinking about modeling the time series by successively "taking out" structural components: First, take out the trend, then take out the seasonality.

The process of taking out the trend and seasonal components need not be done in separate operations. It can be done all at once in a single regression. For example, you could represent the trend with an appropriate function of time (e.g., linear, quadratic, etc.); you could include 0-1 indicator variables for each season but one. The residuals from such a model represent the detrended and deseasonalized data – if those residuals are stationary, they could be appropriate inputs to ARIMA.

Although the detrending and deseasonalizing methods discussed in the preceding paragraphs are perfectly OK, the primary tool for detrending and deseasonalizing in ARIMA is differencing (the "I" part of ARIMA). To the student who is new to ARIMA it often seems that mere differencing could hardly be a good way to detrend and/or deseasonalize. So here are a few examples that illustrate the utility of differencing:

- Suppose Y_t has a linearly increasing mean: 1, 3, 5, 7, 9, …. Then differencing yields the sequence 2, 2, 2, 2, …Thus, differencing adjacent values (called **first-order differencing** in the ARIMA approach) removes a linear trend.
- Suppose Y_t has a quadratic increasing mean: 1, 4, 9, 16, 25, …. Then differencing yields the sequence 3, 5, 7, 9, … . Differencing again (called **second-order differencing**) yields 2, 2, 2, 2, … Second-order differencing removes a quadratic trend.
- The pattern of the preceding two bullet points continues: Third-order differencing removes a cubic trend. Fourth-order differencing removes a quartic trend. Etc.

[3] Trend estimates are not similarly contaminated by seasonality when trend is estimated first – at least not majorly. The intuitive reason is that seasonality is a regular up-and-down pattern that is superimposed on top of the trend. As long as there are sufficiently many ups and downs, they will tend to cancel each other when estimating trend.

- Suppose Y_t is a Random Walk. Random Walks fail stationarity. But first differencing yields a Random Sample, which is stationary.
- Suppose Y_t is quarterly data for which the mean function is periodic (strongly seasonal) with linear trend, for example: 1, 4, 5, 2 | 3, 6, 7, 4 | 5, 8, 9, 6 | 7, 10, 11, 8, … . Then first differencing at lag 4 yields 2, 2, 2, 2 | 2, 2, 2, 2 | 2, 2, 2, 2 … Differencing at an appropriate lag can remove seasonality and possibly a trend.

To stabilize the variance of Y_t, a number of methods are available. Here are two methods that are frequently effective and both accessible at the level of this course:

- Replace Y_t by $\log Y_t$ in the modeling effort. This is effective if the variability (standard deviation) of Y_t is growing proportionately to Y_t.
- Replace Y_t by percentage differences in Y_t: $\frac{Y_{t+1}-Y_t}{Y_t}$. This is also effective if the variability (standard deviation) of Y_t is growing proportionately to Y_t.

It is worth saying a bit about the relationship between $\log Y_t$ and percentage differences in Y_t that are featured in the preceding two bullets. Suppose, for example, that the first difference of $\log Y_t$ passes all tests for being a Random Sample. That is, $\log Y_{t+1} - \log Y_t$ is a Random Sample (which says that $\log Y_t$ is a Random Walk). Now $\log Y_{t+1} - \log Y_t = \log \frac{Y_{t+1}}{Y_t} = \log\left(1+\frac{Y_{t+1}-Y_t}{Y_t}\right) \approx \frac{Y_{t+1}-Y_t}{Y_t}$ if $\frac{Y_{t+1}-Y_t}{Y_t}$ is small, since $\log(1+x) \approx x$ if x is small. So $\frac{Y_{t+1}-Y_t}{Y_t}$ is a Random Sample. The argument is reversible. Thus, $\log Y_t$ is a Random Walk if and only if (approximately) the percentage differences $\frac{Y_{t+1}-Y_t}{Y_t}$ are a Random Sample.

A third method to stabilize the variance of Y_t is very popular (and also effective), but requires some important subtlety that is often not exercised:

- Suppose that X_t is a time series (or deterministic function) *that is growing at the same rate as the variability of* Y_t. Then you can try dividing the regression equation by X_t. This method is best understood in an example:

  Example: Suppose that the time series regression model $y_t = \alpha + \beta x_t + \varepsilon_t$ satisfies L,I,N but not H: the magnitudes of the errors are growing proportional to x_t. That is, the errors ε_t are independent normally distributed with mean 0 and standard deviation $x_t\sigma$. Divide the model by x_t: $y_t / x_t = \alpha \cdot 1/x_t + \beta + \varepsilon_t / x_t$. This modified model has independent errors ε_t / x_t that are normal with mean 0 and standard deviation σ, which satisfies LHIN. So instead of regressing y_t on x_t in order to estimate α and β, regress y_t / x_t on $1/x_t$. Use the estimates of α and β that you get from $y_t / x_t = \alpha \cdot 1/x_t + \beta + \varepsilon_t / x_t$ to estimate y_t in $y_t = \alpha + \beta x_t + \varepsilon_t$. Just note that the estimate that you want for β in $y_t = \alpha + \beta x_t + \varepsilon_t$ is the estimate of the *intercept* in $y_t / x_t = \alpha \cdot 1/x_t + \beta + \varepsilon_t / x_t$ and that the estimate that you want for α in $y_t = \alpha + \beta x_t + \varepsilon_t$ is the estimate of the *slope* in

$y_t / x_t = \alpha \cdot 1 / x_t + \beta + \varepsilon_t / x_t$. This approach is OK if you do not care about the y_t / x_t model and you are just running it to get valid estimates of the parameters of the original model $y_t = \alpha + \beta x_t + \varepsilon_t$. It may even be that the ratio y_t / x_t does not make sense. On the other hand, you may reconsider and decide that y_t / x_t is really what you are interested in and the ratio makes sense. A non-time series example is given by the Austin apartment data set, in which the residuals grow larger as the size of the apartment grows larger. In that example, Y is rent and X could be area. Then Y / X makes sense as dollars per square foot.

Anyway, the objective of the above is to detrend and deseasonalize Y_t and stabilize its variance. Suppose that has been done, giving a new time series Z_t that is stationary. Now the AR and/or MA parts of ARIMA come into play. I now turn to them. It should be noted that for a lot of purposes, it is precisely the trend and seasonal parts that have been taken out of Y_t that are of major interest. There may not be a lot of interest in the Z_t that remains.

After detrending and deseasonalizing, Z_t is really the residuals (the part that is left) after fitting a model. What remains is to remove the autocorrelation structure – the "I" specification (not the "I" part of ARIMA). This is tricky. There are many possible autocorrelation structures. A major part of the identification phase of ARIMA is to identify the type of autocorrelation structure of Z_t. ARIMA tries to classify the structures into two classes: AR (autoregression) and MA (moving average), or mixtures of those two types. Each type leaves a different signature in the autocorrelation function. We can look at the autocorrelation function to identify the signature in order to complete the identification phase of ARIMA (corresponding to the "Propose" step of "Propose, Validate, Use").

But before getting into that, let us be sure that we understand what an AR model is and what an MA model is.

- Z_t is **first-order autoregressive [AR(1)]** if $Z_t = \mu + u_t$, where $u_t = \rho u_{t-1} + \varepsilon_t$ and ε_t is a zero-mean Random Sample.
- Z_t is **first-order moving average [MA(1)]** if $Z_t = \mu + u_t$, where $u_t = \beta \varepsilon_{t-1} + \varepsilon_t$ and ε_t is a zero-mean Random Sample.

You may need to look twice in order to see the difference between the two models. Examination of the definitions shows the origin of the names: The AR(1) error looks like an (auto)regression: $u_t = \rho u_{t-1} + \varepsilon_t$ - the error is regressed upon its own lag with no intercept. The MA(1) error is a weighted average $u_t = \beta \varepsilon_{t-1} + \varepsilon_t$ of two terms of a different time series. The terms being averaged "move" as time *t* moves forward into the future. At first glance, it does not appear as if there is much difference between AR(1) and MA(1). The AR(1) error depends upon its own past, whereas the MA(1) error depends upon the past of an error series. However, that seemingly small difference makes a sizable difference in signatures of their autocorrelation functions. By *autocorrelation function signature* I mean the pattern of the autocorrelation at lags 1, 2, 3, … This difference can be exploited to identify the type and order of Z_t.

AR(1) autocorrelation signature. Suppose that Z_t is first-order autoregressive. Then using properties of covariance,[4] we have $Cov(u_t, u_{t-1})$ = (substituting the definition of AR(1), namely $u_t = \rho u_{t-1} + \varepsilon_t$) $Cov(\rho u_{t-1} + \varepsilon_t, u_{t-1}) = Cov(\rho u_{t-1}, u_{t-1}) + Cov(\varepsilon_t, u_{t-1})$ = (since ε_t comes at a later time period, it is independent of u_{t-1}) $\rho Cov(u_{t-1}, u_{t-1}) + 0 = \rho \sigma_u^{\,2}$. So $Corr(u_t, u_{t-1}) = \rho$. Now, at lag 2, we have $Cov(u_t, u_{t-2})$ = $Cov(\rho u_{t-1} + \varepsilon_t, u_{t-2}) = \rho Cov(u_{t-1}, u_{t-2}) + Cov(\varepsilon_t, u_{t-2})$ = (since ε_t comes at a later time period, it is independent of u_{t-2}) $\rho \cdot \rho \sigma_u^{\,2} + 0 = \rho^2 \sigma_u^{\,2}$, where $\sigma_u^{\,2}$ is the constant variance of u_t for all *t*. So $Corr(u_t, u_{t-2}) = \rho^2$. The pattern continues: $Corr(u_t, u_{t-k}) = \rho^k$. So an AR(1) manifests in exponentially declining residual autocorrelations as the lag increases.

MA(1) autocorrelation signature. Suppose that Z_t is first-order moving average. Then using properties of covariance,[5] we have $Cov(u_t, u_{t-1})$ = (substituting the definition of MA(1), namely $u_t = \beta \varepsilon_{t-1} + \varepsilon_t$ and $u_{t-1} = \beta \varepsilon_{t-2} + \varepsilon_{t-1}$) $Cov(\beta \varepsilon_{t-1} + \varepsilon_t, \beta \varepsilon_{t-2} + \varepsilon_{t-1})$ = (since the only coincident terms in the independent ε terms are *t*-1 and *t*-1) $0 + Cov(\beta \varepsilon_{t-1}, \varepsilon_{t-1}) + 0 + 0 =$ $\beta Cov(\varepsilon_{t-1}, \varepsilon_{t-1}) = \beta \sigma_\varepsilon^{\,2}$. So $Corr(u_t, u_{t-1}) = \beta \dfrac{\sigma_\varepsilon^{\,2}}{\sigma_u^{\,2}}$. From $u_t = \beta \varepsilon_{t-1} + \varepsilon_t$, we have $Var(u_t) = Var(\beta \varepsilon_{t-1} + \varepsilon_t)$ = (by independence of the ε_t) $Var(\beta \varepsilon_{t-1}) + Var(\varepsilon_t) = \beta^2 \sigma_\varepsilon^{\,2} + \sigma_\varepsilon^{\,2}$. So substitute into $Corr(u_t, u_{t-1}) = \beta \dfrac{\sigma_\varepsilon^{\,2}}{\sigma_u^{\,2}} = \beta \dfrac{\sigma_\varepsilon^{\,2}}{\beta^2 \sigma_\varepsilon^{\,2} + \sigma_\varepsilon^{\,2}} = \dfrac{\beta}{\beta^2 + 1}$. From this, it follows that $-\dfrac{1}{2} \le Corr(u_t, u_{t-1}) \le \dfrac{1}{2}$.

Now, at lag 2, we have $Cov(u_t, u_{t-2})$ = $Cov(\beta \varepsilon_{t-1} + \varepsilon_t, \beta \varepsilon_{t-3} + \varepsilon_{t-2}) = 0$ since there are no coincident times among the independent ε_t. The pattern continues: $Corr(u_t, u_{t-k}) = 0$ for $k \ge 2$. So an MA(1) manifests in zero autocorrelations after lag 1.

More generally, the AR and MA models may be extended beyond first order. For example, here are the definitions of AR(2) and MA(2):

- Z_t is **second-order autoregressive [AR(2)]** if $Z_t = \mu + u_t$, where $u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \varepsilon_t$ and ε_t is a zero-mean Random Sample.
- Z_t is **second-order moving average [MA(2)]** if $Z_t = \mu + u_t$, where $u_t = \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \varepsilon_t$ and ε_t is a zero-mean Random Sample.

The pattern continues – adding another term for each increase of one in the order.

What are the autocorrelation function signatures of higher order AR and MA processes? It turns out that in general, an AR(p) process has an autocorrelation function that is a mixture of exponentially decaying functions and damped sinusoidal functions (oscillating between positive

[4] See the Appendix to this document for properties of covariance.
[5] See the Appendix to this document for properties of covariance.

and negative autocorrelations, with the amplitude decaying). The autocorrelation function of an MA(q) process cuts out after lag q and equals zero thereafter.

So the autocorrelation function provides a means to identify whether a process is AR or MA: The autocorrelation function of an MA suddenly drops to zero; the autocorrelation of an AR process does not, but gradually tapers off. Moreover, the lag just before the MA autocorrelation drops to zero identifies the order of the MA.

So the autocorrelation function provides a reliable way to identify an MA and determine its order. The autocorrelation function is less helpful in identifying an AR and its order.

More useful than the autocorrelation function for identifying an AR process and its order is the partial autocorrelation function. The **partial autocorrelation of Z_t at lag *k*** is $corr(Z_t, Z_{t+k} \mid Z_{t+1}, Z_{t+2}, ..., Z_{t+k-1})$. That is, the correlation between two *Z*'s, separated by *k* time periods, controlling for, or holding constant, all *Z*'s in between.

How can the partial autocorrelation function help identify AR models? Consider the AR(2), for example. At lag 3, $corr(u_t, u_{t+3} \mid u_{t+1}, u_{t+2}) = corr(u_t, \rho_2 u_{t+2} + \rho_1 u_{t+1} + \varepsilon_t \mid u_{t+1}, u_{t+2}) =$ (since controlling for u_{t+1}, u_{t+2} treats them as constants and correlation is not affected by adding constants) $corr(u_t, \varepsilon_t \mid u_{t+1}, u_{t+2}) = 0$ since u_t, ε_t are independent as part of the model specification. So the AR(p) process has a partial autocorrelation function that cuts out and drops to zero after *p* lags. However, it can be shown that the partial autocorrelation function of an MA(q) process has a more complicated structure: It is a mixture of exponentially decaying functions and damped sinusoidal functions.

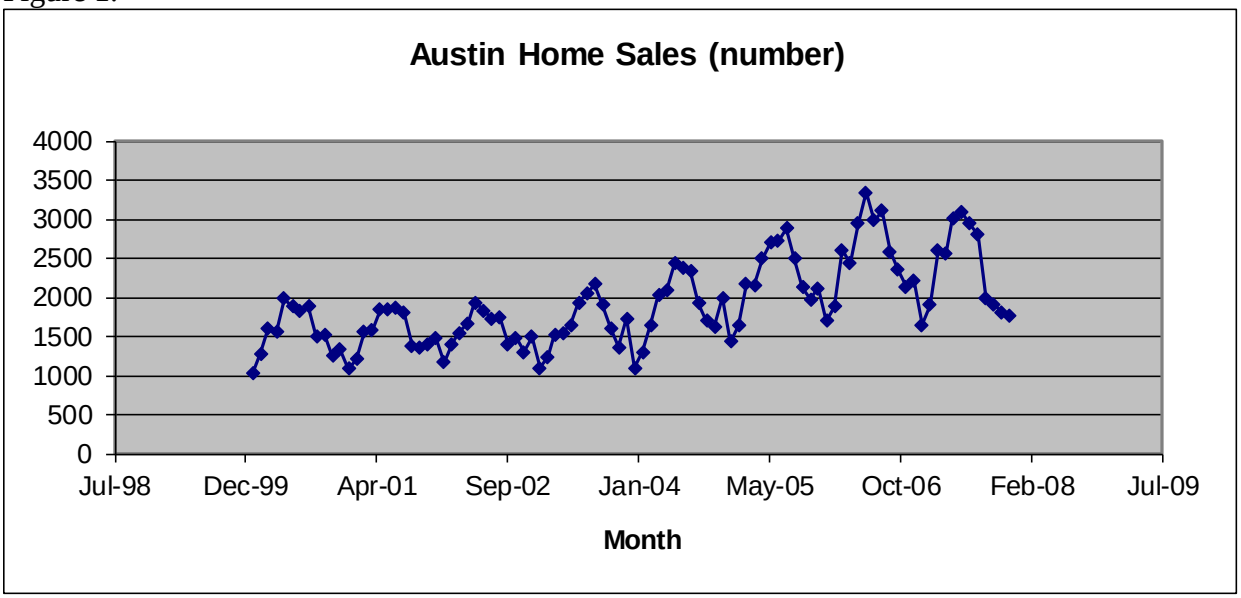
In summary, the autocorrelation function of an MA(q) process becomes zero after q lags.
The partial autocorrelation function of an AR(p) process becomes zero after p lags.
You can identify whether a time series is AR or MA by these rules – and if an AR or MA, you can identify what the order is. But these rules should be used as guidelines, not as black-and-white laws. For complex time series may be both AR and MA simultaneously, random variation may fuzz the clarity of the drop to zero, and the real world need not follow the ARIMA prescription.

More detail on ARIMA may be found in Chapter 5 of the book *Introduction to Time Series Analysis and Forecasting* by Montgomery, Jennings, and Kuhlaci.

Some Notes on PROC ARIMA in SAS

PROC ARIMA implements the ARIMA strategy in SAS. PROC ARIMA is complex and requires experience and study to use well. However, the basic framework is not hard to use. Here is an example.

Figure 1.



The plot shows the number of homes sold in Austin, month by month. Clearly evident is the uptrend with strong seasonal variation. AUSTIN_HOMES is a SAS dataset that has the monthly sales counts. Here is some sample SAS code for analyzing these data by ARIMA.

```
proc arima data=Austin_homes;
  identify var=sales(1,12);
  estimate q=(1)(12);
  forecast lead=12;
run;
```

There are three key statements, which correspond to the three phases of the ARIMA strategy.

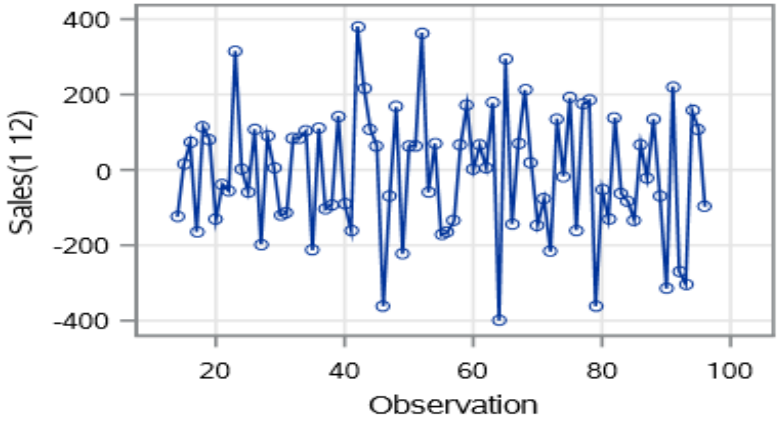
- First is the IDENTIFY statement, which specifies a time series variable to be modeled, as well as possible preliminary processing. For example, in the above code, VAR=SALES(1,12) says that the time series variable named SALES in the SAS dataset WORK.AUSTIN_HOMES will be modeled as the response variable (Y_t). In addition, SALES will be differenced at lag 1 and again at lag 12. This differencing is intended to remove linear trend in sales and periodic seasonality over 12 months. If you do not want to difference, just omit the parenthetical (1, 12). Note that in the code as written, the variable that will be fit is no longer sales, but sales that have been differenced at lag 1 and then again at lag 12 (see Figure 2). Differencing could be done in a DATA step prior to running PROC ARIMA, but it is convenient to be able to do it in PROC ARIMA.
- Second, the ESTIMATE statement specifies which AR and/or MA model is to be fit to the doubly differenced sales. The parameter Q = (1)(12) says to fit an MA(1) and MA(12) model simultaneously. To fit an MA(1) model, write ESTIMATE Q=1. To fit an AR(1)

model, write ESTIMATE P=1. To fit an AR(2) and MA(1) simultaneously, write ESTIMATE P=2 Q=1.

- Third, FORECAST LEAD=12 says to use the fitted model to forecast sales for each of the next 12 months. Note that it is *sales* forecasts that are reported in the output, not forecasts of differenced sales. To be sure, since the model was fit to differenced sales, differenced sales must be forecast. However, as a convenience to you, the forecasts of differenced sales are automatically collected behind the scenes and reconstituted into forecasts of sales. As a consequence, you should think about whether you really want to difference and otherwise preprocess sales in a DATA step prior to running PROC ARIMA. If you do, the output forecasts will be forecasts of your differenced sales, instead of sales. This could be inconvenient for you.

The output of PROC ARIMA also contains a number of diagnostic measures to help you decide if the model that you specified sufficiently meets the model specifications.
Here is selected output from running the above ARIMA statements:

Figure 2. Austin home sales differenced at lags 1 and 12 – input data for ARIMA machinery



The doubly differenced series appears suitably stationary (stable – L and H). What remains is to model the correlation structure.

Figure 3. Statistics for the model fit.

Conditional Least Squares Estimation					
Parameter	Estimate	Standard Error	t Value	Approx Pr > |t|	Lag
MU	-3.83538	8.63176	-0.44	0.6580	0
MA1,1	0.29066	0.10847	2.68	0.0089	1
MA2,1	0.34331	0.12107	2.84	0.0058	12

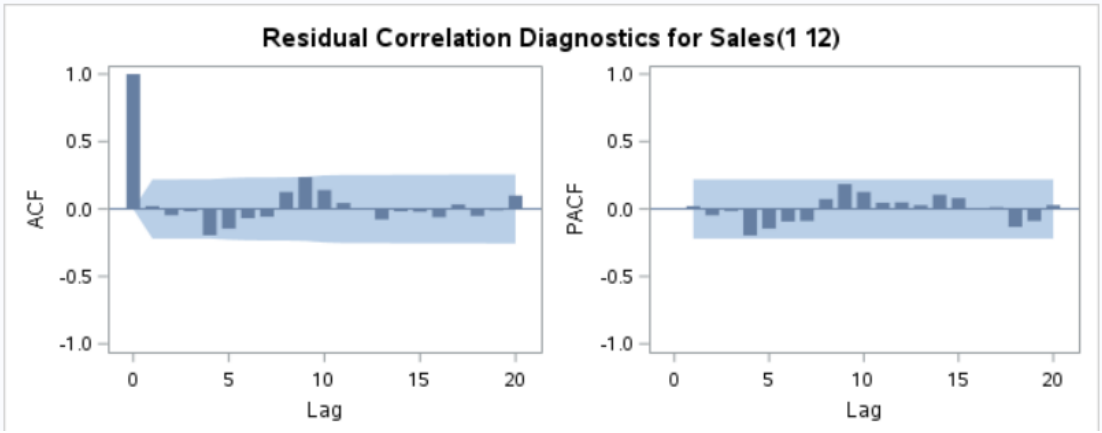
Constant Estimate	-3.83538
Variance Estimate	24305.66
Std Error Estimate	155.9027
AIC	1076.661
SBC	1083.917
Number of Residuals	83

.

Both the first (MA1,1) and 12th (MA2,1) differencing terms have significant coefficients (*p*-values = 0.0089 and 0.0058). The model can estimate monthly sales to within about ±156 (Std Error Estimate – i.e., RMSE).

Figure 4. Autocorrelation diagnostics

Autocorrelation Check of Residuals									
To Lag	Chi-Square	DF	Pr > ChiSq	Autocorrelations					
6	6.02	4	0.1974	0.021	-0.047	-0.017	-0.195	-0.146	-0.068
12	15.27	10	0.1227	-0.058	0.126	0.237	0.140	0.044	0.002
18	16.76	16	0.4013	-0.077	-0.017	-0.021	-0.061	0.034	-0.052
24	21.65	22	0.4811	-0.009	0.099	0.053	-0.134	0.107	0.017



The goal of ARIMA is to reduce the time series to white noise (essentially, a normal random sample). The "Pr > ChiSq" at top of Figure 4 shows that the remaining autocorrelation in the residuals is negligible across 24 lags. The ACF and PACF plots show that autocorrelation and partial autocorrelation among residuals of the model are consistent with white noise - note that both are contained within the blue 95% confidence bands around zero.

Figure 5. Model Forecasts of Next 12 Months of Sales, with 95% Confidence Bands

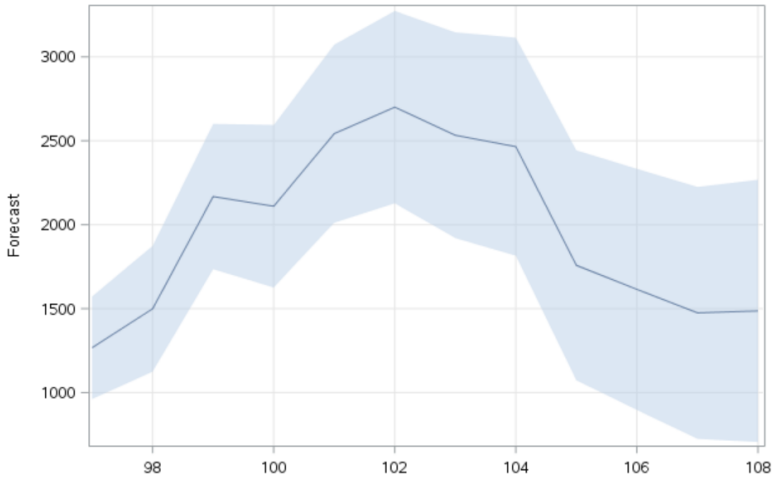


Figure 5 shows the pattern of forecasts of monthly sales for the next 12 months.

You may have noticed that the main strategy of ARIMA is oriented toward modeling a univariate time series in terms of its own past. The strategy, as presented up to this point, says nothing about trying to explain Austin home sales in terms of other times series, like mortgage interest rates, employment rates, and other potentially relevant time series, in addition to the past history of Austin home sales. Other time series can be used. That is more complicated, and there are some potential theoretical and practical pitfalls (such as whether the time series are co-integrated). In the terminology of ARIMA, use of other time series predictors involves **transfer functions**. Sometimes, ARIMA that uses other time series as predictors is called ARIMAX, with the postpended "X" standing for "X predictors."

In SAS, use of other predictor time series can be accommodated by the use of the CROSSCORR option in the ESTIMATE statement. For example, suppose that MORT_RATE and EMP_RATE are monthly time series variables in the data set WORK.AUSTIN_HOMES. Then the following code would use mortgage rate differenced at one lag and the employment rate not differenced at all as predictors of sales that are doubly differenced at lags 1 and 12, with an AR(1) model for the errors of that model, and a forecast for each of the next 12 months' sales with that model.

```
proc arima data=Austin_homes;
  identify var=sales(1,12) crosscorr=(mort_rate(1) emp_rate);
  estimate p=1;
  forecast lead=12;
run;
```

From this brief discussion, you may perhaps see that ARIMA modeling can indeed be complex. Compared with other types of time series modeling, ARIMA can often produce better fits of the data, but at a cost of some loss of interpretability.

APPENDIX

Some Properties of Correlation and Covariance

- $Corr(X,Y) = \dfrac{Cov(X,Y)}{\sigma_X \sigma_Y}$, where *Corr* mean correlation and *Cov* means covariance.
- In a simple linear regression of *Y* on *X*, the slope is $Corr(Y,X)\dfrac{\sigma_Y}{\sigma_X}$.
- $Cov(X,X) = Var(X)$
- Always: $-1 \le Corr(X,Y) \le +1$.
- $|Corr(X,Y)| = 1$ if and only if $Y = a + bX$ for some *a* and $b \neq 0$.
- If *X* and *Y* are uncorrelated, then $Cov(X,Y) = Corr(X,Y) = slope = 0$.
- For any random variables *X, Y, U, V*, and constants *a, b, c, d*, we have
  $Cov(aX + bY, cU + dV) = acCov(X,U) + adCov(X,V) + bcCov(Y,U) + bdCov(Y,V)$.

Special cases of the preceding bullet:

- $Cov(aX + bY, cX + dY) = acVar(X) + (ad + bc)Cov(X,Y) + bdVar(Y)$
- $Var(aX + bY) = a^2 Var(X) + 2abCov(X,Y) + b^2 Var(Y)$
- If *X* and *Y* are independent, then $Var(aX + bY) = a^2 Var(X) + b^2 Var(Y)$
- In particular, if *X* and *Y* are independent, then $Var(X + Y) = Var(X) + Var(Y)$ and $Var(X - Y) = Var(X) + Var(Y)$; and if *X* and *Y* are dependent, then $Var(X + Y) = Var(X) + 2Cov(X,Y) + Var(Y)$
- $Corr(a + bX, c + dY) = sign(bd)Corr(X,Y)$. That is, correlation is unaffected by a linear transformation of the variables, as long as the coefficients of *X* and *Y* have the same sign.