Modeling and Analysis of Time Series Data
Chapter 6: Extending the ARMA model: Seasonality, integration and trend

STATS 531, Winter 2026

Edward Ionides

Seasonal autoregressive moving average (SARMA) models

- A general SARMA$(p,q)\times(P,Q)_{12}$ model for monthly data is

[S1] $$\phi(B)\Phi(B^{12})(Y_n-\mu)=\theta(B)\Theta(B^{12})\epsilon_n,$$

where $\{\epsilon_n\}$ is a white noise process and

$$\begin{aligned}\mu &= \mathrm{E}[Y_n]\\ \phi(x) &= 1-\phi_1 x-\cdots-\phi_p x^p,\\ \theta(x) &= 1+\theta_1 x+\cdots+\theta_q x^q,\\ \Phi(x) &= 1-\Phi_1 x-\cdots-\Phi_P x^P,\\ \Theta(x) &= 1+\Theta_1 x+\cdots+\Theta_Q x^Q.\end{aligned}$$

- SARMA is a special case of ARMA, where the AR and MA polynomials are factored into a **monthly** polynomial in B and an **annual (or seasonal) polynomial** in B^{12}.
- Everything we learned about ARMA models (including assessing causality, invertibility and reducibility) also applies to SARMA.

Choosing the period for a SARMA model

- For the SARMA$(p,q)\times(P,Q)_{12}$ model, 12 is called the **period**.
- One could write a SARMA model for a period other than 12.
- A SARMA$(p,q)\times(P,Q)_4$ model could be appropriate for quarterly data.
- In principle, a SARMA$(p,q)\times(P,Q)_{52}$ model could be appropriate for weekly data, though in practice ARMA and SARMA may not work so well for higher frequency data.

- The seasonal period should be appropriate for the system being modeled. It is usually inappropriate to fit a SARMA$(p,q)\times(P,Q)_9$ model just because you notice a high sample autocorrelation at lag 9.

Consider the following two models:

[S2] $Y_n = 0.5Y_{n-1} + 0.25Y_{n-12} + \epsilon_n,$

[S3] $Y_n = 0.5Y_{n-1} + 0.25Y_{n-12} - 0.125Y_{n-13} + \epsilon_n,$

Question. Which of [S2] and/or [S3] is a SARMA model?

Question. Why do we assume a multiplicative structure in the SARMA model, [S1]? What theoretical and practical advantages (or disadvantages) arise from requiring that an ARMA model for seasonal behavior has polynomials that can be factored as a product of a monthly polynomial and an annual polynomial?

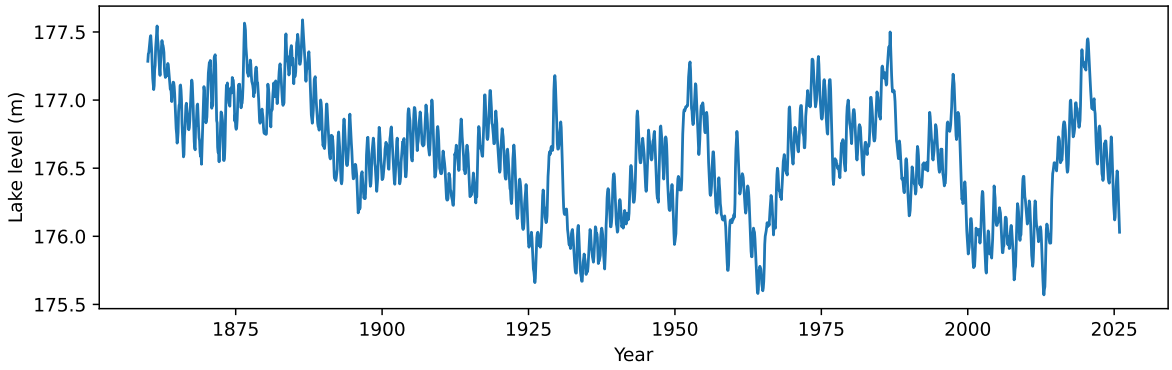
Fitting a SARMA model

We fit a monthly version of the Lake Huron depth data.

```
dat = pd.read_csv("huron_level.csv", comment='#')
dat.columns = dat.columns.str.strip()
print(dat.iloc[:3, :6])
```

```
   Year      Jan      Feb      Mar      Apr      May
0  1860  177.285  177.339  177.349  177.388  177.425
1  1861  177.077  177.105  177.224  177.254  177.382
2  1862  177.227  177.181  177.272  177.321  177.397
```

```
# Reshape data to monthly time series
huron_level = dat.iloc[:, 1:13].values.flatten()
years = np.repeat(dat['Year'].values, 12)
months = np.tile(np.arange(12), len(dat))
time = years + months/12
```



Based on our previous analysis, we try fitting AR(1) for the annual polynomial. We try ARMA(1,1) for the monthly part, giving

$$(1-\Phi_1 B^{12})(1-\phi_1 B)(Y_n-\mu) = (1+\theta_1 B)\epsilon_n. \tag{1}$$

- As discussed earlier, we analyze data only up to 2014, shown by a red line on the plot.

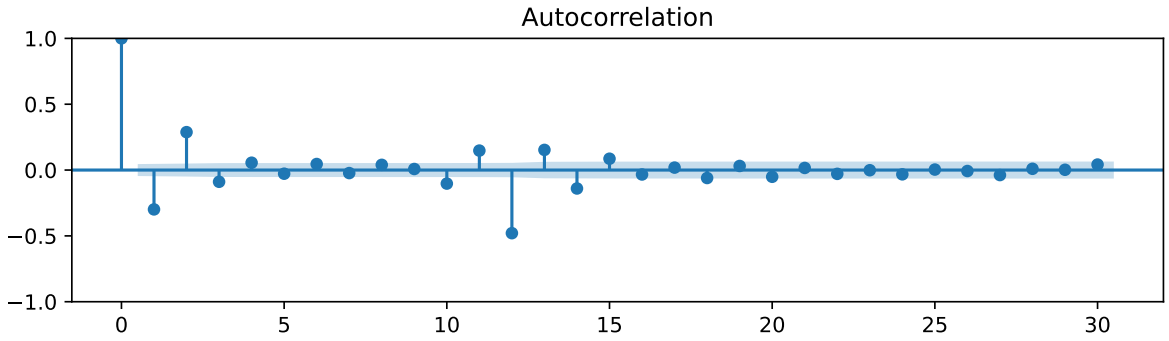
```
huron_level_sub = huron_level[time < 2014.99]
time_sub = time[time < 2014.99]

# Fit SARMA(1,1)x(1,0)_12 model
huron_sarma11x10 = SARIMAX(huron_level_sub,
    order=(1, 0, 1), trend='c',
    seasonal_order=(1, 0, 0, 12)).fit()
```

```
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept   9.988e-06      0.001      0.018      0.986      -0.001       0.001
ar.L1          0.9926      0.009    116.192      0.000       0.976       1.009
ma.L1          0.5997      0.012     48.318      0.000       0.575       0.624
ar.S.L12       1.0000   2.32e-05   4.32e+04      0.000       1.000       1.000
sigma2         0.0039      0.000     33.443      0.000       0.004       0.004
==============================================================================
```

Residual analysis is similar to non-seasonal ARMA models.

- We look for residual correlations at lags corresponding to multiples of the period (here, 12, 24, 36, …) for misspecified annual dependence.
- Analyze `huron_sarma11x10.resid[1:]` to ignore the first residual, which is large due to initialization.



Question. What do you conclude from this residual analysis? What would you do next?

ARMA models for differenced data

Applying a difference operation to the data can make it look more stationary and therefore more appropriate for ARMA modeling.

- This can be viewed as a **transformation to stationarity**.
- We can transform the data $y_{1:N}$ to $z_{2:N}$

$$z_n = \Delta y_n = y_n - y_{n-1}. \tag{2}$$

Definition: an ARMA(p,q) model $Z_{2:N}$ for the differenced data $z_{2:N}$ is called an **integrated autoregressive moving average** model for $y_{1:N}$ and is written as ARIMA(p,1,q).

Formally, the ARIMA(p,d,q) model with intercept μ for $Y_{1:N}$ is

[S4] $\quad \phi(B)[(1-B)^d Y_n - \mu] = \theta(B)\,\epsilon_n,$

where $\{\epsilon_n\}$ is a white noise process; $\phi(x)$ and $\theta(x)$ are ARMA polynomials.

- It is unusual to fit an ARIMA model with $d > 1$.

An ARIMA(p,1,q) model is almost a special case of an ARMA(p+1,q) model with a **unit root** to the AR(p+1) polynomial.

Question. Why "almost" not "exactly" in the previous statement?

Two reasons to fit an ARIMA(p,d,q) model with $d > 0$

1. You may really think that modeling the differences is a natural approach for your data. The S&P 500 stock market index analysis in Chapter 3 is an example of this, as long as you remember to first apply a logarithmic transform to the data.
2. Differencing often makes data look "more stationary" and perhaps it will then look stationary enough to justify applying the ARMA machinery.

- We should be cautious about this second reason. It can lead to poor model specifications and hence poor forecasts or other conclusions.
- The second reason was more compelling in the 1970s and 1980s. Limited computing power resulted in limited alternatives, so it was practical to force as many data analyses as possible into the ARMA framework and use method of moments estimators.

Practical advice on using ARIMA models

- ARIMA analysis is relatively simple to do. It has been a foundational component of time series analysis since the publication of the influential book "Time Series Analysis" [1] which developed and popularized ARIMA modeling.
- A practical approach is:

1. Do a competent ARIMA analysis.
2. Identify potential limitations in this analysis and remedy them using more advanced methods.
3. Assess whether you have in fact learned anything from (2) that goes beyond (1).

The SARIMA$(p, d, q) \times (P, D, Q)$ model

Combining integrated ARMA models with seasonality, we can write a general SARIMA$(p, d, q) \times (P, D, Q)_{12}$ model for nonstationary monthly data, given by

[S5] $\phi(B)\Phi(B^{12})[(1-B)^d(1-B^{12})^D Y_n - \mu] = \theta(B)\Theta(B^{12})\epsilon_n,$

where $\{\epsilon_n\}$ is a white noise process, the intercept μ is the mean of the differenced process $\{(1-B)^d(1-B^{12})^D Y_n\}$, and we have ARMA polynomials $\phi(x)$, $\Phi(x)$, $\theta(x)$, $\Theta(x)$ as in model [S1].

- The SARIMA$(0, 1, 1) \times (0, 1, 1)_{12}$ model has often been used for forecasting monthly time series in economics and business. It is sometimes called the **airline model** after a data analysis by Box and Jenkins (1970).

Trend estimation: regression with ARMA errors

Definition: A general **signal plus noise** model is

[S6] $Y_n = \mu_n + \eta_n,$

where $\{\eta_n\}$ is a stationary, mean zero stochastic process, and μ_n is the mean function.

- If, in addition, $\{\eta_n\}$ is uncorrelated, then we have a **signal plus white noise** model. The usual linear trend regression model fitted by least squares in Chapter 2 corresponds to a signal plus white noise model.
- We can say **signal plus colored noise** if we wish to emphasize that we're not assuming white noise.
- Here, **signal** and **trend** are used interchangeably. In other words, we are assuming a deterministic signal.
- A **signal plus ARMA(p,q) noise model** occurs when $\{\eta_n\}$ is a stationary, causal, invertible, mean zero ARMA(p,q) process.

- As well as the $p + q + 1$ parameters in the ARMA(p,q) noise model, there will usually be unknown parameters in the mean function.
- When the mean (also called trend) has a linear specification,

$$\mu_n = \sum_{k=1}^{K} Z_{n,k}\beta_k, \tag{3}$$

the model is **linear regression with ARMA errors**.

- Writing Y for a column vector of $Y_{1:N}$, μ for a column vector of $\mu_{1:N}$, η for a column vector of $\eta_{1:N}$, and Z for the $N \times K$ matrix with (n,k) entry $Z_{n,k}$, we have a general linear regression model with correlated ARMA errors,

$$Y = Z\beta + \eta. \tag{4}$$

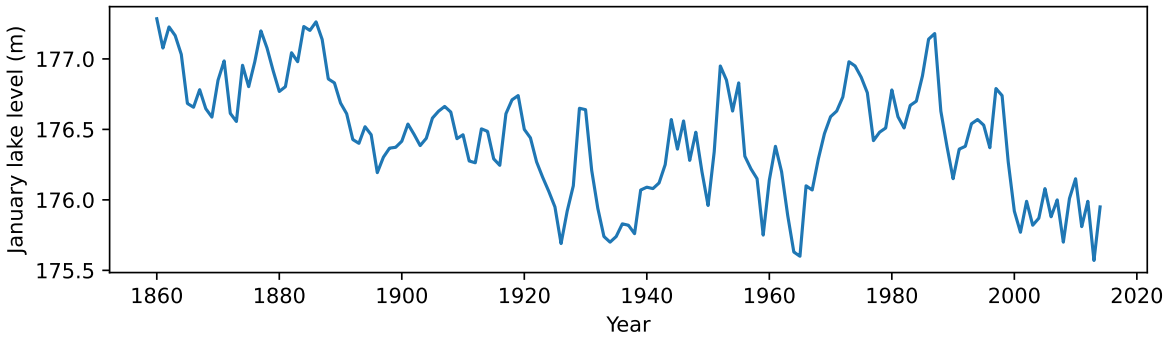
- From this equation, $Y - Z\beta$ is ARMA so likelihood evaluation and numerical maximization can build on ARMA methods.

Inference for the linear regression model with ARMA errors

- Maximum likelihood estimation of $\theta = (\phi_{1:p}, \theta_{1:q}, \sigma^2, \beta)$ is a nonlinear optimization problem.
- Fortunately, Python's SARIMAX can do it for us using the `exog` argument.
- As usual, we should look out for signs of numerical problems.
- Data analysis for a linear regression with ARMA errors model, using the framework of likelihood-based inference, is procedurally similar to fitting an ARMA model.
- This is a powerful technique, since the covariate matrix Z can include other time series. We can evaluate associations between different time series.
- With appropriate care (since **association is not causation**) we can draw inferences about mechanistic relationships between dynamic processes.

Evidence for systematic trend in Lake Huron level?

We return to annual data, say the January level, to avoid seasonality.



- Visually, there seems some evidence for a decreasing trend, but there are also considerable fluctuations.

- Let's test for a trend, using a regression model with Gaussian AR(1) errors. We have previously found that this is a reasonable model for these data.
- First, for comparison, we fit a null model with no trend.

```
from io import StringIO
fit0 = SARIMAX(huron_jan, order=(1, 0, 0),
    trend='c').fit()
params_csv = fit0.summary().tables[1].as_csv()
pd.read_csv(StringIO(params_csv))
```

		coef	std err	z	P>|z|	[0.025	0.975]
0	intercept	26.2506	6.887	3.812	0.0	12.753	39.748
1	ar.L1	0.8512	0.039	21.814	0.0	0.775	0.928
2	sigma2	0.0440	0.005	8.439	0.0	0.034	0.054

AIC: -37.06

- We compare `fit0` with a linear trend model, coded as `fit1`.
- The covariate is included via the `exog` argument.

```
fit1 = SARIMAX(huron_jan, order=(1, 0, 0), trend='c',
    exog=year_jan-np.mean(year_jan)).fit()
```

		coef	std err	z	P>|z|	[0.025	0.975]
0	intercept	32.5374	7.856	4.142	0.000	17.140	47.935
1	x1	-0.0050	0.002	-2.814	0.005	-0.008	-0.002
2	ar.L1	0.8156	0.045	18.319	0.000	0.728	0.903
3	sigma2	0.0422	0.005	7.981	0.000	0.032	0.053

AIC: -40.72

- The optimization fails to find the maximum unless the covariate is centered by subtracting

the mean. Optimization for ARMA models can be fragile!

Setting up a formal hypothesis test

- To talk formally about these results, we must set down a model and some hypotheses.
- Writing the data as $y_{1:N}$, collected at years $t_{1:N}$, the model we have fitted is

$$(1 - \phi_1 B)(Y_n - \mu - \beta t_n) = \epsilon_n, \tag{5}$$

where $\{\epsilon_n\}$ is Gaussian white noise with variance σ^2. Our null model is

$$H^{\langle 0 \rangle} : \beta = 0, \tag{6}$$

and our alternative hypothesis is

$$H^{\langle 1 \rangle} : \beta \neq 0. \tag{7}$$

Question. How do we test $H^{\langle 0 \rangle}$ against $H^{\langle 1 \rangle}$? onstruct two different tests using the Python output above. Which test do you think is more accurate, and why?

Question. How would you check whether your preferred test is indeed better? What other supplementary analysis could you do to strengthen your conclusions?

Testing for trend vs unit root: The ADF test

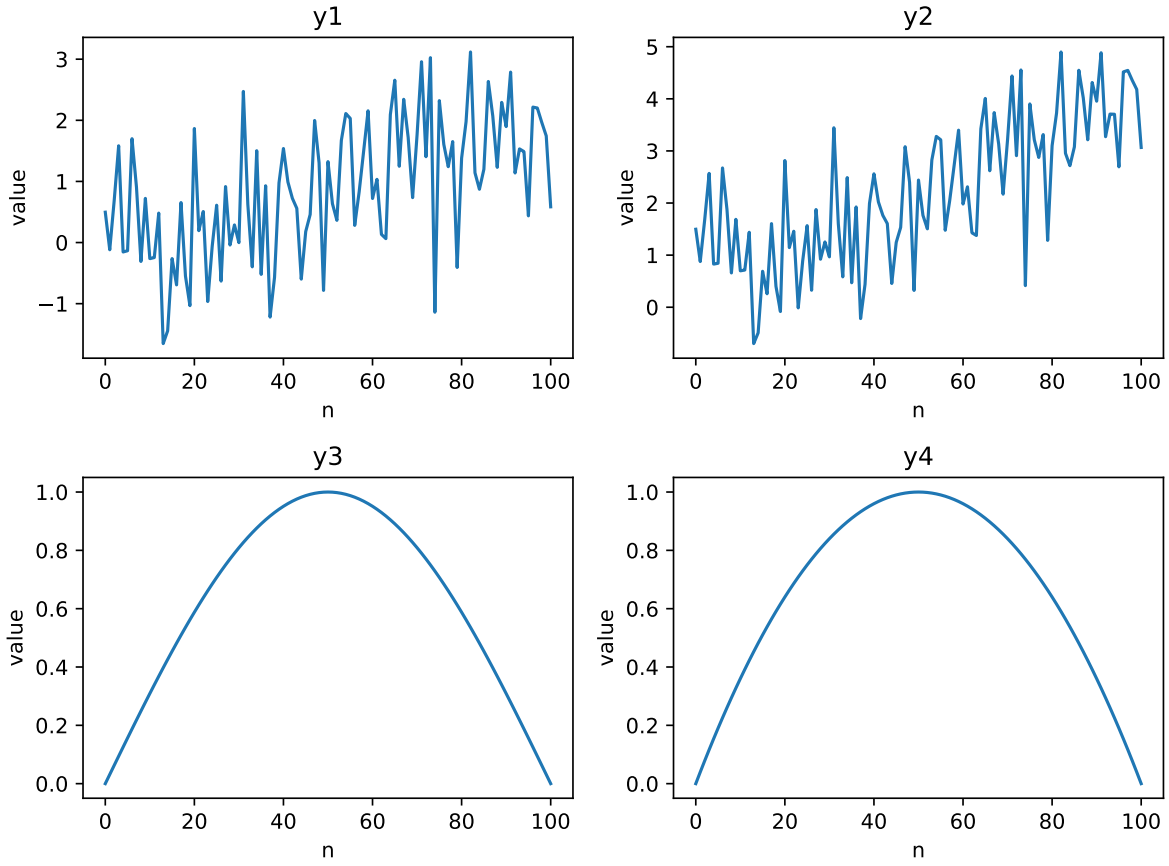
Definition: A time series model has a **unit root** if its first difference is stationary.

- For a linear time series model, this is a $(1 - B)$ factor in the AR specification, and hence an AR polynomial root of 1.
- A unit root corresponds to a "stochastic trend" (formally an oxymoron) described by a random walk. This is the null hypothesis of the **augmented Dickey-Fuller (ADF)** test.
- The ADF test is sometimes called a test for "stationarity". However, non-stationary models may remain non-stationary after any amount of differencing. e.g., any model with non-polynomial trend.

- Assessing evidence for a trend is better than applying an ADF test for a unit root if a model with trend is statistically (or scientifically) preferable to a model with a random walk.

Question. Which of `y1`, `y2`, `y3`, `y4` do you think will be considered non-stationary by a unit root test?

```
np.random.seed(42)
n = np.arange(101)
epsilon = np.random.normal(0, 1, len(n))
y1 = 2*n/100 + epsilon
y2 = np.exp(1.5*n/100) + epsilon
y3 = np.sin(2*np.pi*n/200)
y4 = n*(100-n)/2500
```



ADF test using `adfuller`

```
def print_adf(series, name):
    result = adfuller(series, maxlag=4,
        regression='ct', autolag=None)
    print(f"{name}:",
        f"ADF statistic = {result[0]:.3E},",
        f"Lag order = {result[2]},",
        f"p-value = {result[1]:.4f}.")
```

```
y1: ADF statistic = -3.983E+00, Lag order = 4, p-value = 0.0093.
y2: ADF statistic = -3.925E+00, Lag order = 4, p-value = 0.0112.
y3: ADF statistic = -3.541E+12, Lag order = 4, p-value = 0.0000.
y4: ADF statistic = 1.037E+00, Lag order = 4, p-value = 1.0000.
```

- Alternative hypothesis: stationary
- Results can be sensitive to choice of test arguments. Here, arguments are set to match defaults of R `tseries::adf.test`

When to use the ADF test

- If you are interested in a null hypothesis of a random walk against an alternative of a stationary ARMA model, then this test is applicable.
- Generalizations of ADF claim to identify random walks on top of a nonlinear trend, but distinguishing these two options is hard and beyond our scope.
- Most midterm projects benefit from avoiding the ADF test and focusing on a direct investigation of the evidence for trend or other non-stationary behavior.

Further reading

- Section 3.9 of Shumway and Stoffer [3] discusses SARIMA modeling.
- Section 3.8 of Shumway and Stoffer [3] introduces regression with ARMA errors, calling it ARMAX.
- Section 5.3 of Huang and Petukhina [2] presents regression with ARMA errors, calling it REGARMA.
- Section 9.3 of Huang and Petukhina [2] discusses unit root tests.

Acknowledgments

- Compiled on February 5, 2026 using Python.

- We acknowledge previous versions of this course.

References

[1] G. E. P. Box and G. M. Jenkins. *Time Series Analysis: Forecasting and Control*. First. San Francisco: Holden–Day, 1970.

[2] Changquan Huang and Alla Petukhina. *Applied Time Series Analysis and Forecasting with Python*. Springer, 2022.

[3] Robert H Shumway and David S Stoffer. *Time Series Analysis and its Applications: With R Examples*. 4th. Springer, 2017.