11: Time Series Analysis

Hydrologic Data Analysis | Kateri Salk

Fall 2019

Lesson Objectives

1. Discuss the purpose and application of time series analysis for hydrologic data
2. Decompose time series into individual components

Session Set Up

```
getwd()
```

```
## [1] "/Users/ks501/Box Sync/Courses/Hydrologic Data Analysis/Lessons"
```

```
library(tidyverse)
library(dataRetrieval)

theme_set(theme_classic())
```

Time Series Analysis

Time series are a special class of dataset, where a response variable is tracked over time. The frequency of measurement and the timespan of the dataset can vary widely. At its most simple, a time series model includes an explanatory time component and a response variable. Mixed models can include additional explanatory variables (check out the `nlme` and `lme4` R packages). We will be covering a few simple applications of time series analysis in these lessons.

Opportunities

Analysis of time series presents several opportunities. In aquatic sciences, some of the most common questions we can answer with time series modeling are:

- Has there been an increasing or decreasing **trend** in the response variable over time?
- Can we **forecast** conditions in the future?

Challenges

Time series datasets come with several caveats, which need to be addressed in order to effectively model the system. A few common challenges that arise (and can occur together within a single dataset) are:

- **Autocorrelation**: Data points are not independent from one another (i.e., the measurement at a given time point is dependent on previous time point(s))
- **Data gaps**: Data are not collected at regular intervals, necessitating *interpolation* between measurements.
- **Seasonality**: Cyclic patterns in variables occur at regular intervals, impeding clear interpretation of a monotonic (unidirectional) trend.
- **Heteroscedasticity**: The variance of the time series is not constant over time
- **Covariance**: the covariance of the time series is not constant over time

Visualizing a time series dataset

Today, we will analyze discharge data from Clear Creek in Colorado, USA. Let's first look at what types of data are available for this dataset.

```
ClearCreekSummary <- whatNWISdata(siteNumbers = "06719505")
```

Notice that mean daily discharge has been measured at this site since 1974, with over 16,000 measurements available. This will be a robust time series dataset for us to analyze changes that have occurred over the past 45 years.

Let's gather all the discharge data available from this site. Notice that `readNWISdv` gathers dates, which are already formatted as a date object. What do you notice about the data when plotted over time?

```
# Import data
ClearCreekDischarge <- readNWISdv(siteNumbers = "06719505",
                     parameterCd = "00060", # discharge (ft3/s)
                     startDate = "",
                     endDate = "")
names(ClearCreekDischarge)[4:5] <- c("Discharge", "Approval.Code")
class(ClearCreekDischarge$Date)
```

```
## [1] "Date"
```

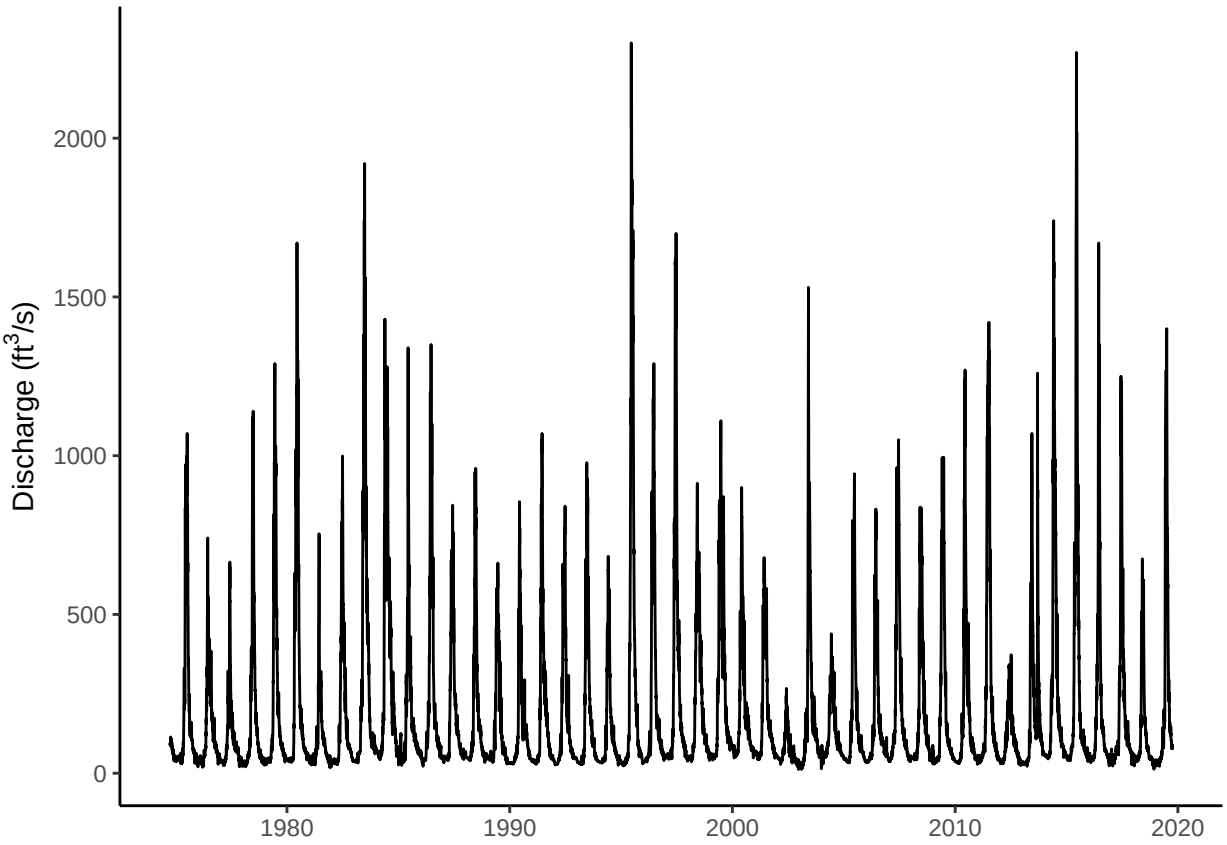
```
# Learn about the variables and site
attr(ClearCreekDischarge, "variableInfo")
```

```
##   variableCode           variableName              variableDescription
## 1        00060 Streamflow, ft&#179;/s Discharge, cubic feet per second
##       valueType  unit options noDataValue
## 1 Derived Value ft3/s    Mean          NA
```

```
attr(ClearCreekDischarge, "siteInfo")
```

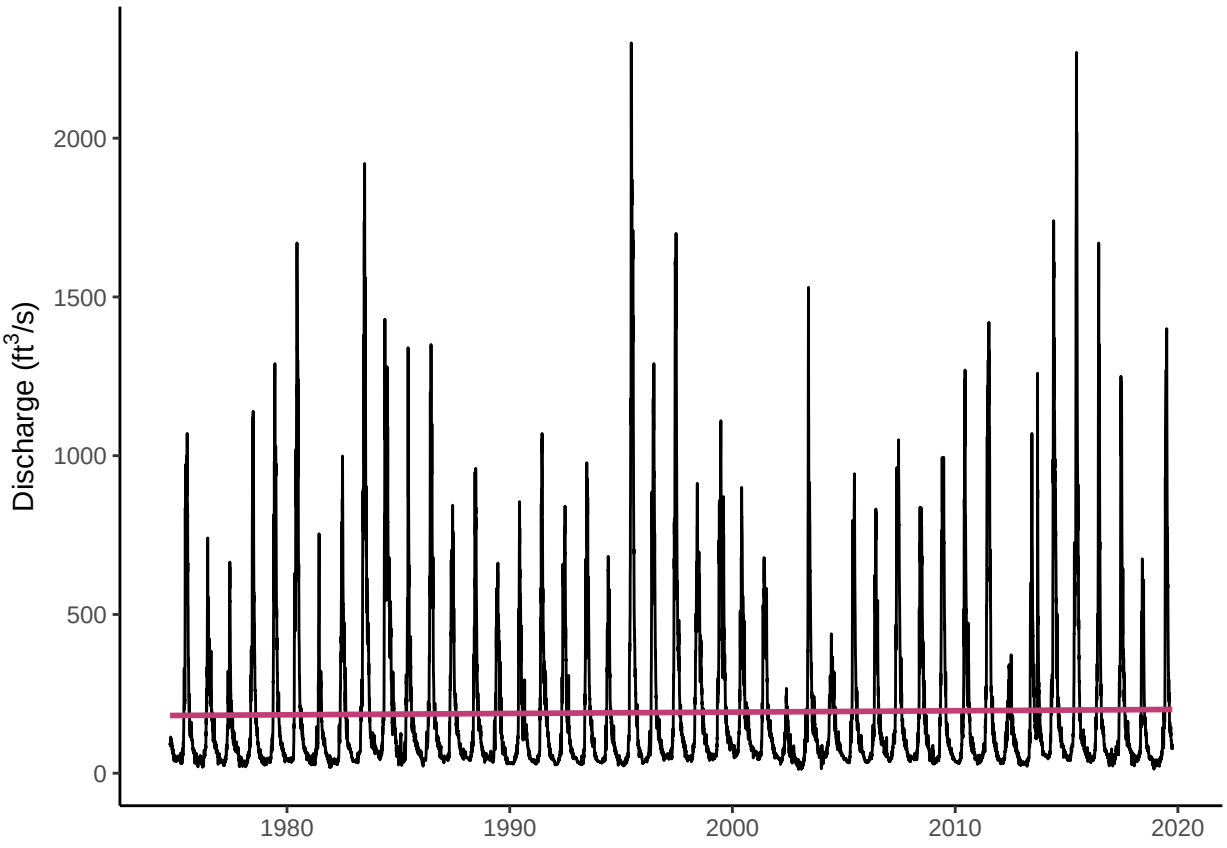
```
##                  station_nm  site_no agency_cd timeZoneOffset
## 1 CLEAR CREEK AT GOLDEN, CO 06719505      USGS         -07:00
##   timeZoneAbbreviation dec_lat_va dec_lon_va       srs siteTypeCd    hucCd
## 1                  MST   39.75304  -105.2353 EPSG:4326         ST 10190004
##   stateCd countyCd network
## 1      08    08059    NWIS
```

```
# Plot discharge over time
ClearCreekPlot <-
  ggplot(ClearCreekDischarge, aes(x = Date, y = Discharge)) +
  geom_line() +
  labs(x = "", y = expression("Discharge (ft"^3*"/s)")) +
  theme(plot.title = element_text(margin = margin(b = -10), size = 12),
        axis.title.x = element_blank())
print(ClearCreekPlot)
```



The simplest option to characterize a time series is a linear regression.

```
ClearCreekRegressionPlot <-
  ggplot(ClearCreekDischarge, aes(x = Date, y = Discharge)) +
  geom_line() +
  geom_smooth(method = "lm", se = FALSE, color = "#c13d75ff") +
  labs(x = "", y = expression("Discharge (ft"^3*"/s)")) +
  theme(plot.title = element_text(margin = margin(b = -10), size = 12),
        axis.title.x = element_blank())
print(ClearCreekRegressionPlot)
```



Is a linear regression an appropriate model for the data in this case? Why or why not?

Are there situations where it would be appropriate to use a linear regression to model a time series? If so, what is an example?

Decomposing a time series dataset

A given time series can be made up of several component series:

1. A **seasonal** component, which repeats over a fixed known period (e.g., seasons of the year, months, days of the week, hour of the day)
2. A **trend** component, which quantifies the upward or downward progression over time. The trend component of a time series does not have to be monotonic.
3. An **error** or **random** component, which makes up the remainder of the time series after other components have been accounted for. This component reflects the noise in the dataset.
4. (optional) A **cyclical** component, which repeats over periods greater than the seasonal component. A good example of this in hydrologic data is El Niño Southern Oscillation (ENSO) cycles, which occur over a period of 2-8 years.

We first need to turn the discharge data into a time series object in R. This is done using the `ts` function. Notice we can only specify one column of data and need to specify the period at which the data are sampled. The resulting time series object cannot be viewed like a regular data frame.

Note: time series objects must be equispaced, which requires interpolation if the data have not been collected

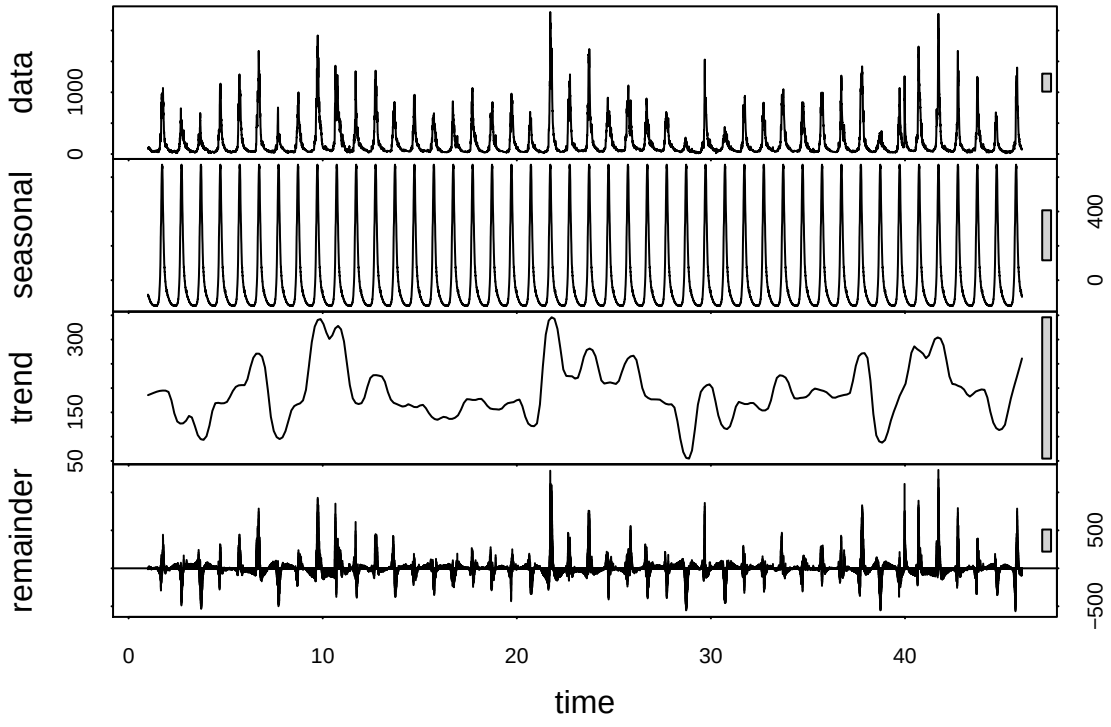
at regular intervals. In our case, we have daily data with no NAs in the data frame, so we don't need to worry about this.

```
ClearCreek_ts <- ts(ClearCreekDischarge[[4]], frequency = 365)
```

The `stl` function decomposes the time series object into its component parts. We must specify that the window for seasonal extraction is either "periodic" or a specific number of at least 7. The decomposition proceeds through a loess (locally estimated scatterplot smoothing) function.

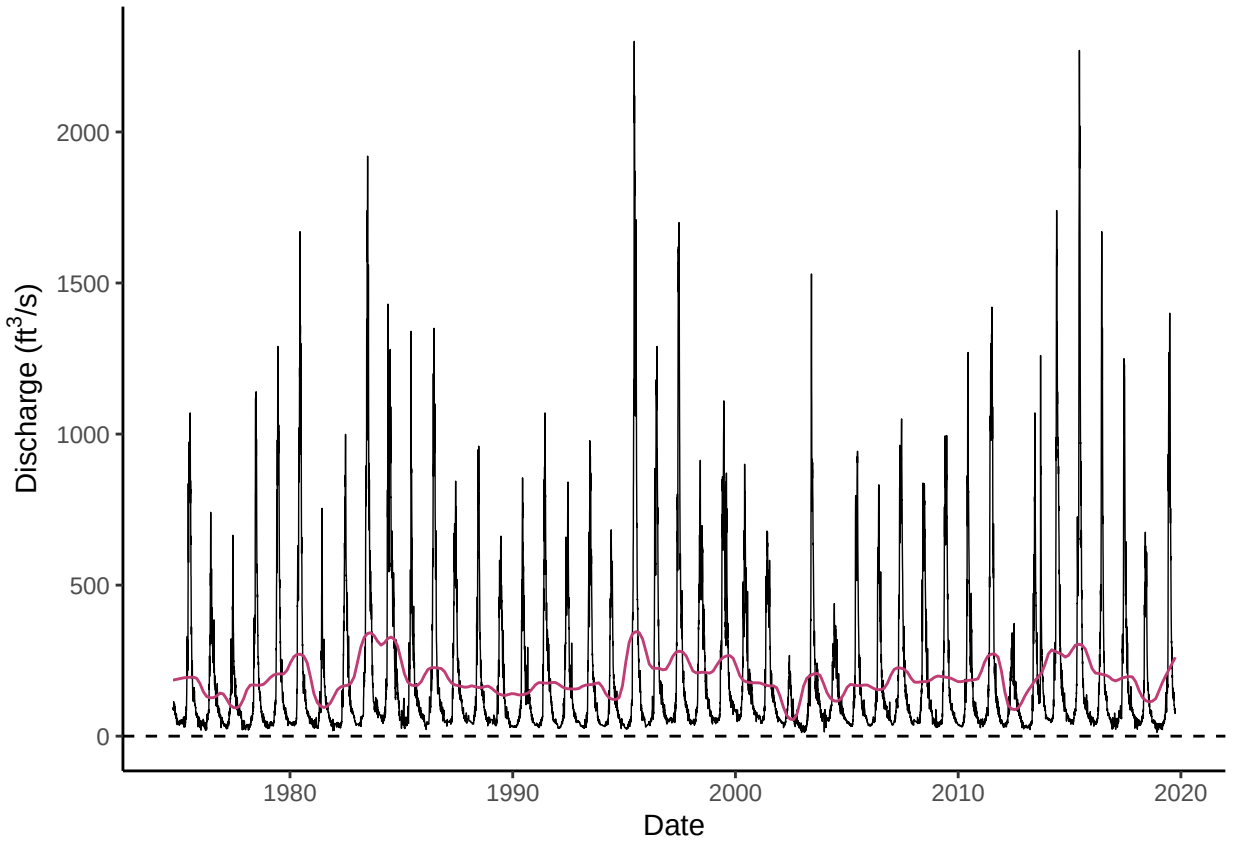
```
?stl
# Generate the decomposition
ClearCreek_Decomposed <- stl(ClearCreek_ts, s.window = "periodic")

# Visualize the decomposed series.
plot(ClearCreek_Decomposed)
```

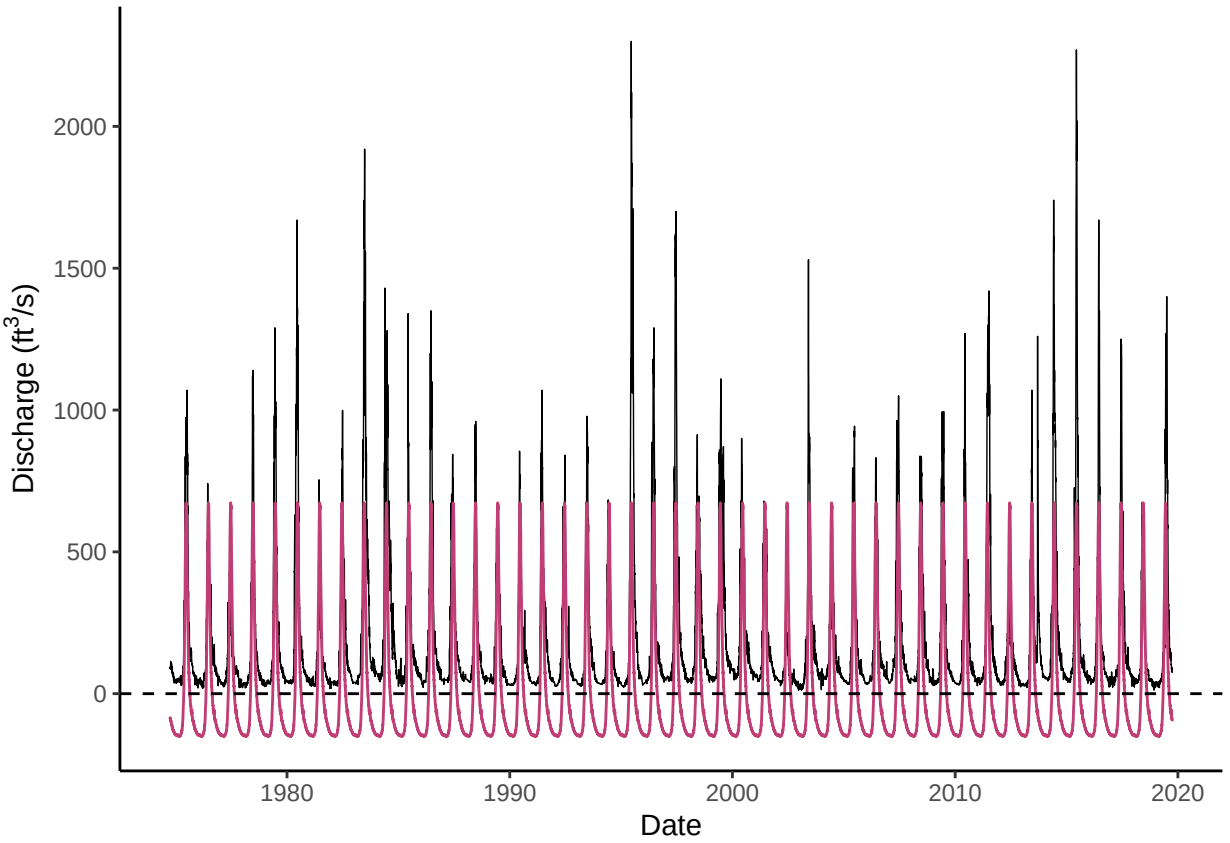



```
# We can extract the components and turn them into data frames
ClearCreek_Components <- as.data.frame(ClearCreek_Decomposed$time.series[,1:3])
ClearCreek_Components <- mutate(ClearCreek_Components,
                      Observed = ClearCreekDischarge$Discharge,
                      Date = ClearCreekDischarge$Date)

# Visualize how the trend maps onto the data
ggplot(ClearCreek_Components) +
  geom_line(aes(y = Observed, x = Date),  size = 0.25) +
  geom_line(aes(y = trend, x = Date), color = "#c13d75ff") +
  geom_hline(yintercept = 0, lty = 2) +
  ylab(expression("Discharge (ft"^3*"/s)"))
```




```
# Visualize how the seasonal cycle maps onto the data
ggplot(ClearCreek_Components) +
  geom_line(aes(y = Observed, x = Date),  size = 0.25) +
  geom_line(aes(y = seasonal, x = Date), color = "#c13d75ff") +
  geom_hline(yintercept = 0, lty = 2) +
  ylab(expression("Discharge (ft"^3*"/s)"))
```



Note that the decomposition can yield negative values when we apply a seasonal adjustment or a trend adjustment to the data. The decomposition is not constrained by a lower bound of zero as discharge is in real life. Make sure to interpret with caution!

Closing Discussion

Next lesson we will discuss how to backcast, forecast, and detect monotonic trends in time series data.