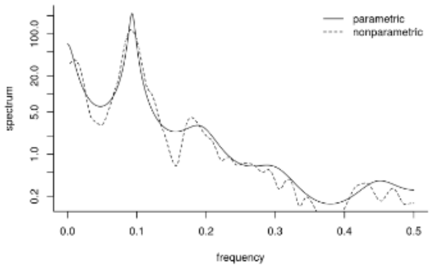
Lecture 8, MTH 400, Fall 2024

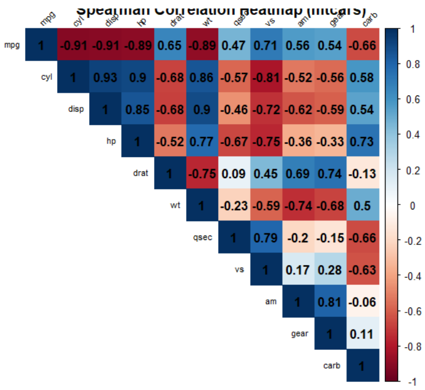
Animations and Time Series

When performing exploratory data analysis (EDA) on time series data, several specific techniques and visualizations are commonly used to gain insights into the underlying patterns, trends, and characteristics of the data. Here are some typical exploratory analysis approaches for time series data:

1. Time Series Plot: A basic starting point is to plot the time series data itself. A simple line plot with time on the x-axis and the variable of interest on the y-axis can reveal trends, seasonality, and overall patterns in the data.
2. Seasonal Decomposition: Time series data often exhibit seasonal patterns, which can be decomposed into trend, seasonal, and residual components. Decomposition techniques such as additive or multiplicative decomposition help separate these components and provide a clearer understanding of the underlying structure.
3. Rolling Statistics: Computing rolling statistics, such as rolling mean, rolling standard deviation, or rolling correlation, can help identify trends, variations, and cyclical patterns in the data. Rolling statistics are calculated over a specific window or interval of time.
4. Autocorrelation and Partial Autocorrelation: Autocorrelation measures the relationship between observations at different lags, while partial autocorrelation measures the relationship between observations after accounting for the effects of intermediate lags. These plots can help identify the presence of dependencies or time-dependent relationships in the data.
5. Periodogram and Spectral Analysis: Periodogram analysis helps identify dominant frequencies or periodic components in the time series data. Spectral analysis techniques, such as Fourier transforms or wavelet analysis, provide insights into the frequency content and power spectrum of the time series.



6. Boxplots and Histograms: Visualizations like boxplots and histograms can help examine the distributional properties of the time series data, such as skewness, kurtosis, and outliers. These plots can provide insights into the data's central tendency and variability.
7. Time Series Clustering: Clustering techniques, such as k-means or hierarchical clustering, can be applied to identify similar patterns or groups within the time series data. Clustering can help uncover distinct temporal behavior or identify anomalous patterns.
8. Heatmaps and Correlation Plots: Heatmaps and correlation plots can be used to visualize the relationships between multiple time series variables. These plots help identify associations, dependencies, and lead-lag relationships among different variables over time.



9. Event Analysis: If there are specific events or interventions of interest in the time series data, analyzing the data around those events can provide insights into their impact and effectiveness. Event analysis involves comparing the behavior of the time series before, during, and after the events.

These are just some examples of the exploratory analysis techniques commonly used for time series data. The choice of techniques depends on the specific characteristics and goals of the analysis. It's important to consider the temporal nature of the data and leverage appropriate visualizations and statistical methods to gain insights into the patterns and dynamics present in the time series.

Using animations to visualize time series data can provide several advantages, such as revealing trends, patterns, and changes over time in a dynamic and engaging way. Animations can make it easier to understand the temporal evolution of data, highlight key events, and facilitate the comparison of multiple time series. Here's how you can create and use animations in R to enhance the visualization of time series data:

Advantages of Using Animations for Time Series Data

- Temporal Dynamics: Animations can show how data evolves over time, making it easier to understand trends and patterns.
- Engagement: Animated visualizations can be more engaging and can capture the viewer's attention better than static charts.
- Comparison: Animations can help compare different time periods or datasets more effectively by animating transitions between states.
- Highlighting Events: Key events or changes can be highlighted more clearly through animation, making it easier to identify and analyze significant moments.

Creating Animations in R

One popular package for creating animations in R is gganimate, which extends the ggplot2 package to include animation capabilities.

Installation

First, you need to install the gganimate package (and its dependencies if not already installed):

```
install.packages("gganimate")
install.packages("ggplot2") # if not already installed
install.packages("gapminder") # for example dataset
```

Example: Animated Time Series with gganimate

Here is an example of how to create an animated time series plot using the gganimate package with the gapminder dataset:

- Load Required Libraries:

```
library(ggplot2)
library(gganimate)
library(gapminder)
```

- Prepare the Data:

```
data("gapminder")
head(gapminder)
```

- Create a Static Plot:

```
static_plot <- ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, size = pop, color = continent, frame =
year)) +
  geom_point(alpha = 0.7) +
  scale_x_log10() +
  labs(title = "Gapminder: GDP per Capita vs Life Expectancy",
       x = "GDP per Capita",
       y = "Life Expectancy")
```

- Convert to an Animated Plot:

```
animated_plot <- static_plot +
  transition_time(year) +
  labs(title = "Year: {frame_time}")
```

- Render the Animation:

```
animate(animated_plot, renderer = gifski_renderer())
```

- Save the Animation:

```
anim_save("gapminder_animation.gif", animation = last_animation())
```

Advanced Animation Techniques

- Faceted Animations: Create animations with facets to compare different subsets of data.

```
animated_plot <- static_plot +
  transition_time(year) +
  labs(title = "Year: {frame_time}") +
  facet_wrap(~continent)
animate(animated_plot, renderer = gifski_renderer())
```

- Animating Lines and Paths: Animate lines to show the trajectory of changes over time.

```
line_plot <- ggplot(gapminder, aes(x = year, y = lifeExp, group = country, color = continent)) +
  geom_line() +
  transition_reveal(year)
animate(line_plot, renderer = gifski_renderer())
```

- Using plotly for Interactive Animations: The plotly package can also be used for creating interactive and animated plots.

r

```
install.packages("plotly")
library(plotly)

p <- gapminder %>%
  plot_ly(x = ~gdpPercap, y = ~lifeExp, size = ~pop, color = ~continent, frame = ~year,
         text = ~country, hoverinfo = "text", type = 'scatter', mode = 'markers') %>%
  layout(xaxis = list(type = "log"))
p
```

Using animations to visualize time series data can significantly enhance the understanding and communication of temporal patterns and trends. The gganimate package in R offers a powerful and flexible way to create such animations, allowing you to convey complex time-dependent information in an intuitive and visually appealing manner. Whether you are working with economic data, climate data, or any other time series, animations can help bring your data to life and provide deeper insights.

Resources:

1. https://towardsdatascience.com/how-to-create-animated-plots-in-r-adf53a775961
2. https://r-graph-gallery.com/time-series.html
3. https://www.datascienceinstitute.net/blog/time-series-decomposition-in-r
4. https://rpubs.com/davoodastaraky/TSA1
5. https://ramikrispin.github.io/halloween-time-series-workshop/ts-decomposition.html
6. https://www.geeksforgeeks.org/how-to-calculate-a-rolling-average-in-r/
7. https://www.statology.org/rolling-average-in-r/
8. https://www.geeksforgeeks.org/how-to-calculate-autocorrelation-in-r/
9. https://www.r-bloggers.com/2022/11/autocorrelation-and-partial-autocorrelation-in-time-series/
10. https://www.rdocumentation.org/packages/TSA/versions/1.3/topics/periodogram
11. https://online.stat.psu.edu/stat510/lesson/6/6.1
12. https://ms.mcmaster.ca/bolker/eeid/2010/Ecology/Spectral.pdf
13. https://libguides.princeton.edu/R-Timeseries
14. https://www.datanovia.com/en/blog/gganimate-how-to-create-plots-with-beautiful-animation-in-r/