Automate Professional PowerPoint Presentations Using R

The PowerPoint presentation is the vehicle of choice for information sharing among many businesses. R brings unparalleled power to data analysis and visualization. But getting those visualizations from a window in R Studio to a formatted slide via copy/paste is tedious and error prone.

There’s a better way.

**R Markdown can automate the creation of presentation-worthy slides from R code.** This feature saves hours of time, eliminates errors, and allows a user to update a two-hundred-page slide deck with a key stroke. This is key for decks that include state-by-state data, profit margins across dozens of product lines, or complex visualizations reliant on constantly updated data.

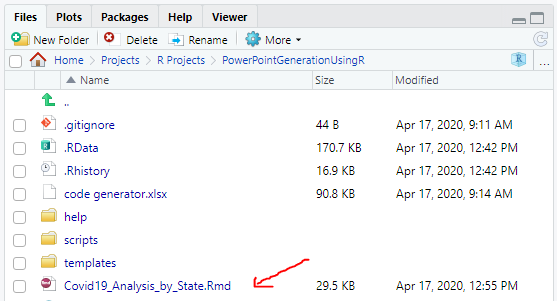
In this example we use data on COVID-19 data provided by the *New York Times* which is updated daily. The R code allows users to get the latest statistics by state in a formatted PowerPoint deck instantly. Check out the code and the outputs on Github: (<https://github.com/jeffrenz/Power-Point-Generation-Using-R>).

The following sections provide step by step instructions on how to:

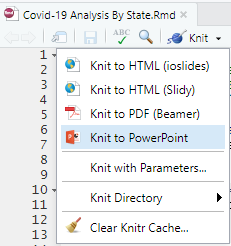
1. Run R Markdown and knit to create power point output
2. Create power point elements:
   1. Title slide
   2. A full-page element, in this case a nicely formatted table
   3. Side-by-side elements
3. Fix PowerPoint formatting problems

# Creating PowerPoint Output

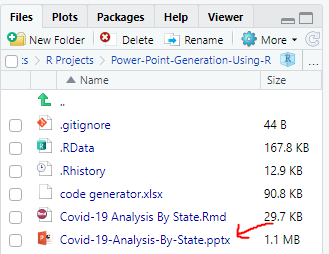
Open up Covid19\_Analysis\_by\_State.Rmd



Click Knit -> Knit to PowerPoint (Takes about 1 minute to generate ppt file)



Click to view PowerPoint



# Generate title slide

This section references the Coid-19-Analysis-By-State.Rmd file on Github (<https://github.com/jeffrenz/Power-Point-Generation-Using-R>).

This tutorial assumes some familiarity with R, R Studio, and R Markdown. If you’ve ever used R Markdown to output an HTML file or a PDF, the format will look very familiar. Simply change the output in the header to powerpoint\_presentation.

Here we’ve also added `r Sys.Date()` will show the current date and added a PASS Template.

---

title: "Contagion Overview by US State"

author: "Jeff Renz, Brian Liberatore"

date: "`r Sys.Date()`"

output:

powerpoint\_presentation:

reference\_doc: PASS\_Template.potx

---

We’ll get into more detail on making advanced changes to the PowerPoint template. Overall, it’s important to retain the default names of the layout slides in the PowerPoint Master View. R Markdown will look for those names when in renders the presentation.

**Power Point Title Slide:**



# Create full page table slide

For this document, we used the following R libraries.

#Load libraries

library(usmap) #used to generate map graphic

library(tidyverse) #includes dplyr for cleaning data and ggplot2 for visualizations

library(scales) #handy way to format numbers

library(lubridate) #used to format dates

library(knitr) #using the kable() function to output professionally formatted tables

We are pulling the data from the *New York Times* Github repository. We use the lubridate function as\_datetime() to format the dates as dates and R’s built-in state.abb and state.name libraries to turn the state names into their abbreviations.

#load data

data <- read.csv("https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties.csv")

data$date <- as\_datetime(data$date)

data$state <- state.abb[match(data$state, state.name)]

Slides with a single element follow the standard R Markdown pattern. A hash tag precedes the header text. The following encapsulates the code block: ```{ r …} [code] ```. We are setting warnings, messages, and echo to FALSE to limit the output of this block to the graph. (No one wants to see R warning messages in a presentation.)

In this block we use the dplyr library (part of the tidyverse) to organize the data. The dplyr library allows for the use of pipes (%>%), which takes the output from the previous function and feeds it into the following function. It keeps the code cleaner and easier to read. We’re using the kable() function (part of the knitr package) to output the chart.

# States most impacted by COVID-19

```{r States\_overview, warning = FALSE, message = FALSE, echo=FALSE, fig.width=12}

State <- as.factor(state.name[match(data$state, state.abb)])

state\_summary <- data %>%

cbind(State) %>%

group\_by(State, date) %>%

summarise("Total Cases" = sum(cases), "Total Deaths" = sum(deaths)) %>%

group\_by(State) %>%

summarize("Total Cases" = max(`Total Cases`),"Total Deaths" = max(`Total Deaths`)) %>%

arrange(desc(`Total Cases`)) %>%

head(15)

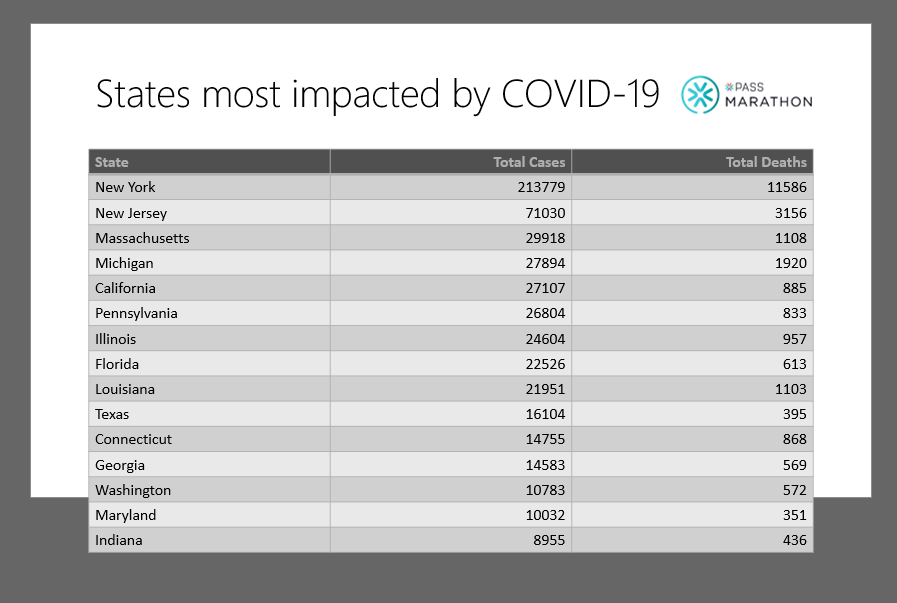
state\_summary$`Total Cases` <- comma(state\_summary$`Total Cases`)

state\_summary$`Total Deaths` <- comma(state\_summary$`Total Deaths`)

kable(state\_summary)

```

But wait, the output runs off the page. Don’t worry, we’ll cover this in advanced formatting.



# Create slide with side-by-side elements

To create a slide with two components, we need to invoke the columns pattern. Note the series of colons ( ::::::::::::: ) encapsulating the code for the slide content. We use inline R code (`r [code]`) to set the title of the slide as the state name.

Each new column is denoted with three colons followed by {.column}. Colorado is the 6th state (alphabetically), so state.name[6] outputs “Colorado”.

# `r state.name[6]`

:::::::::::::: {.columns}

::: {.column}

```{r echo=FALSE, message=FALSE, warning=FALSE}

state\_chart\_function(state.abb[6])

```

:::

::: {.column}

```{r echo=FALSE,message=FALSE,warning=FALSE}

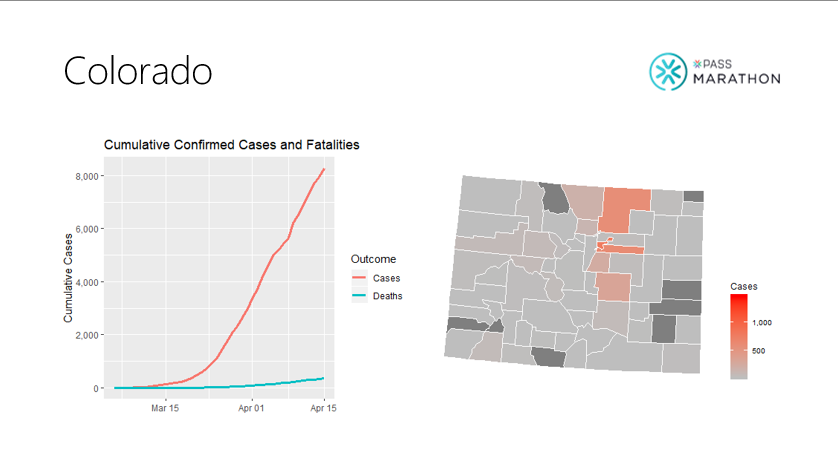
state\_map\_function(state.abb[6])

```

:::

::::::::::::::

We used ggplot2 to create the graph with the cumulative cases and deaths per state and the usmap library to create the county-level heat map.



We used other functions to output additional analysis on the rate of infections. The functions are in the Git repository at scrpits>02\_load\_functions.R.

# `r state.name[6]` infection statistics

:::::::::::::: {.columns}

::: {.column}

```{r echo=FALSE, message=FALSE, warning=FALSE}

key\_figures(state.abb[6])

```

:::

::: {.column}

```{r echo=FALSE,message=FALSE,warning=FALSE}

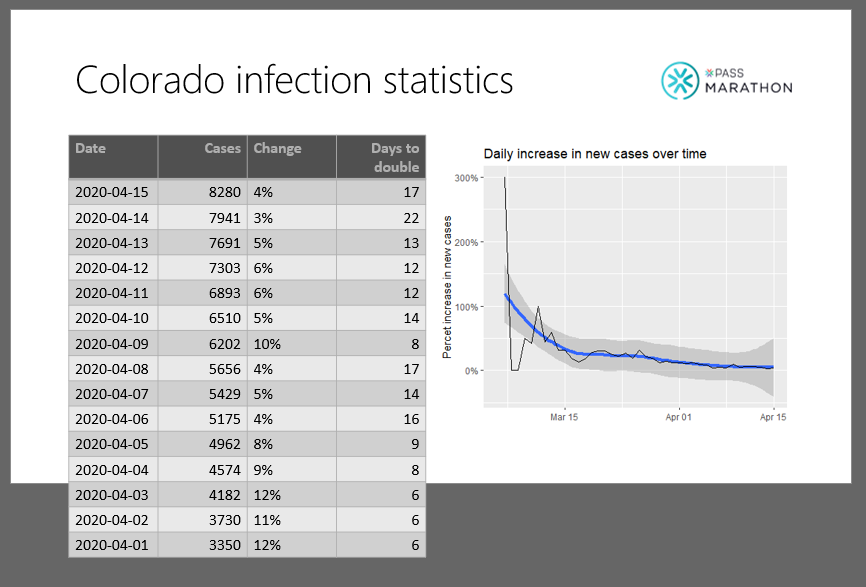
percent\_change\_chart(state.abb[6])

```

:::

::::::::::::::

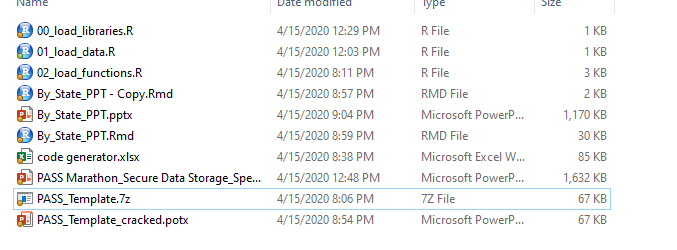
The outputs a slide with the same formatting issue as the state summary slide. We’ll fix this later.



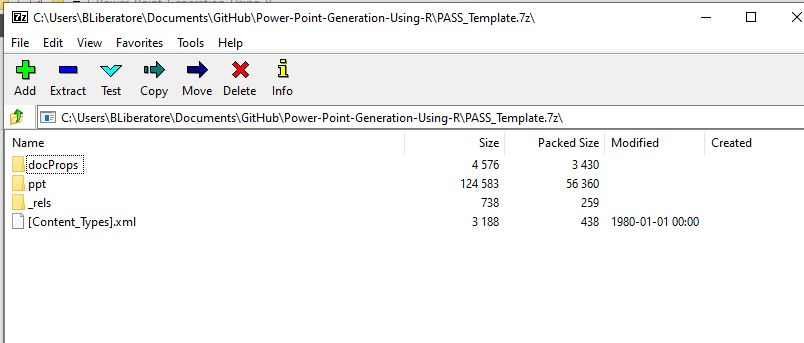
# Advanced formatting of PowerPoint template

Remember that chart that spilled out the slide. This is easily fixed by changing the font size in the chart from 18 pt to 12 pt. However, this is a little trickier than changing a setting in the Master Slide View.

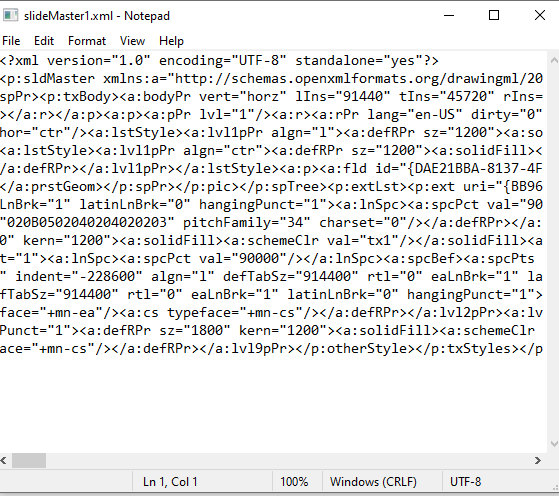
To accomplish this, we must crack the template file. To do so, rename the extension on the template file from .potx (PowerPoint template) to .7z (a 7 zip file).



Now use a zip program such as 7-zip File Manager to open the file.

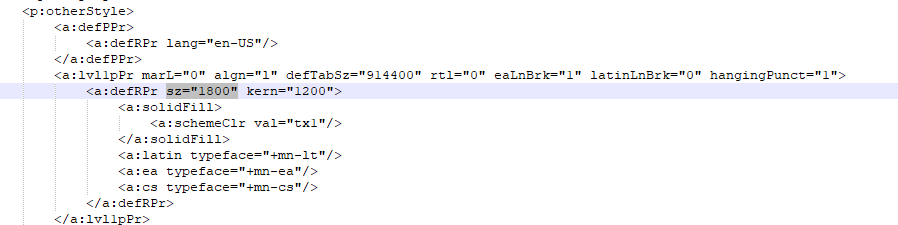


Go to ppt>slideMaster>slideMaster.xml and edit the file.



We pasted the code into Notepad++ to take advantage of the program’s XML formatting.

Under <p:otherStlye>, change that sz=”1800” to sz=”1200” (18 pt font to 12 pt font).

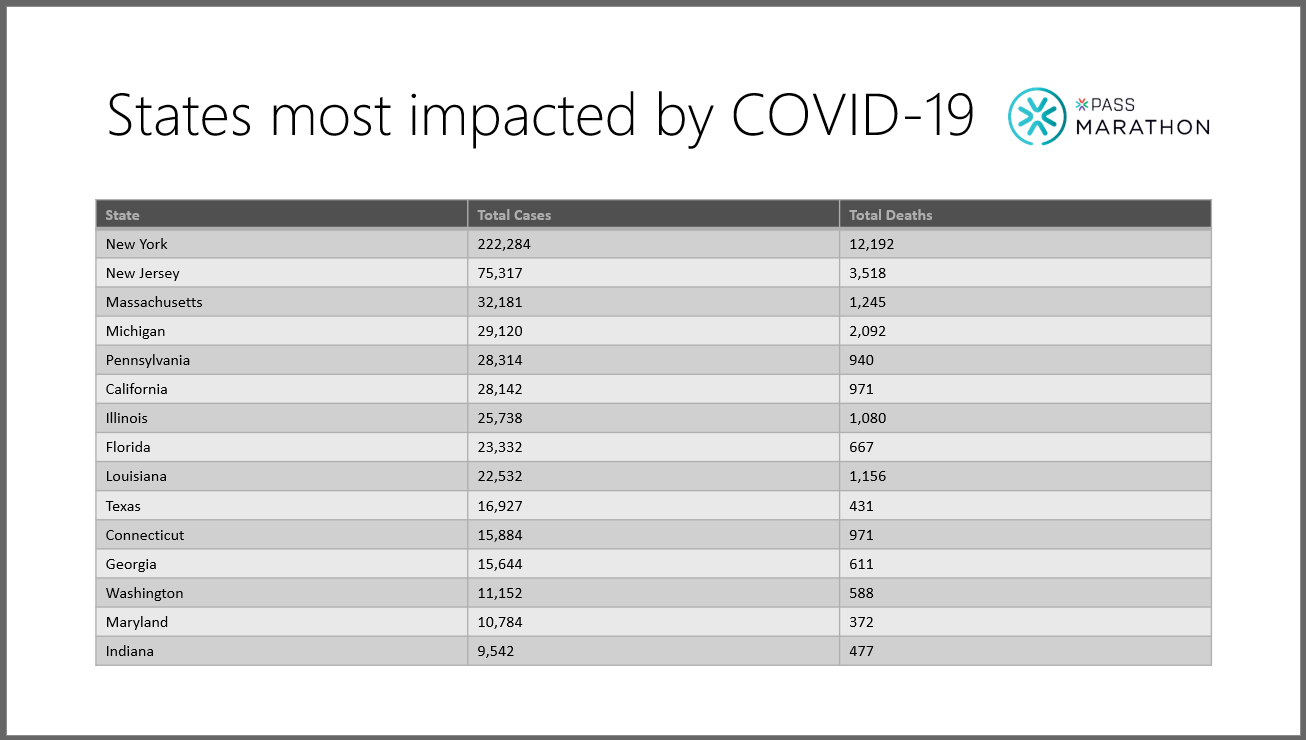


(We changed all 18 pt font to 12 pt font for the template)

Copy/paste the modified code into the edit window and save. Close the window and 7-zip file manager and change the file extension back to the .potx. We renamed the file **PASS\_Template\_cracked.potx.**

Now the default font size for the table is 12 point. It fits nicely on the page, making for a presentable presentation. Now producing a 102-page PowerPoint with the latest statewide analysis of COVID-19 infections requires pressing a button (Ctrl+Shift+K).

Here’s the newly formatted state summary slide.



Here’s the fixed slide showing the state-specific analysis of infection rates:

