R Without Statistics

David Keyes

Contents

About the Book **5**

Introduction **9**

Why R Without Statistics? **9**

How New Zealand Used R to Fight COVID 9

How I Came to Use R 9

How This Book Works 15

A Favor to Ask 16

Illuminate **19**

Use General Principles of High-Quality Data Viz in R **19**

Close read of viz to show why it's effective 19

The grammar of graphics 20

ggplot2 21

Develop a Custom Theme to Keep Your Data Viz Consistent **29**

R is a Full-Fledged Map-Making Tool **31**

Make Tables That Look Good and Share Results Effectively **33**

Communicate 37

Use RMarkdown to Communicate Accurately and Efficiently 37

Use RMarkdown to Instantly Generate Hundreds of Reports 39

Create Beautiful Presentations with RMarkdown 41

Make Websites to Share Results Online 43

Automate 47

Access Up to Date Census Data with the `tidycensus` Package 47

Pull in Survey Results as Soon as They Come In 49

Stop Copying and Pasting Code by Creating Your Own Functions 51

Bundle Your Functions Together in Your Own R Package 53

Conclusion 57

Come for the Data, Stay for the Community 57

About the Book

This is the in-progress version of *R Without Statistics*, a forthcoming book from No Starch Press.

Since R was invented in 1993, it has become a widely used programming language for statistical analysis. From academia to the tech world and beyond, R is used for a wide range of statistical analysis.

R's ubiquity in the world of statistics leads many to assume that it is only useful to those who do complex statistical work. But as R has grown in popularity, the number of ways it can be used has grown as well. Today, R is used for:

- Data visualization
- Map making
- Sharing results through reports, slides, and websites
- Automating processes
- And much more!

The idea that R is only for statistical analysis is outdated and inaccurate. But, without a single book that demonstrates the power of R for non-statistical purposes, this perception persists.

Enter R Without Statistics.

R Without Statistics will show ways that R can be used beyond complex statistical analysis. Readers will learn about a range of uses for R, many of which they have likely never even considered.

Each chapter will, using a consistent format, cover one novel way of using R.

1. Readers will first be introduced to an R user who has done something novel and learn how using R in this way transformed their work.
2. Following this, there will be code samples that demonstrate exactly how the R user did the thing they are being profiled for.

3. Finally, there will be a summary, with lessons learned from this novel way of using R.

Written by David Keyes, Founder and CEO of R for the Rest of Us, R Without Statistics will be published by No Starch Press.

Introduction

Why R Without Statistics?

How New Zealand Used R to Fight COVID

TODO

How I Came to Use R

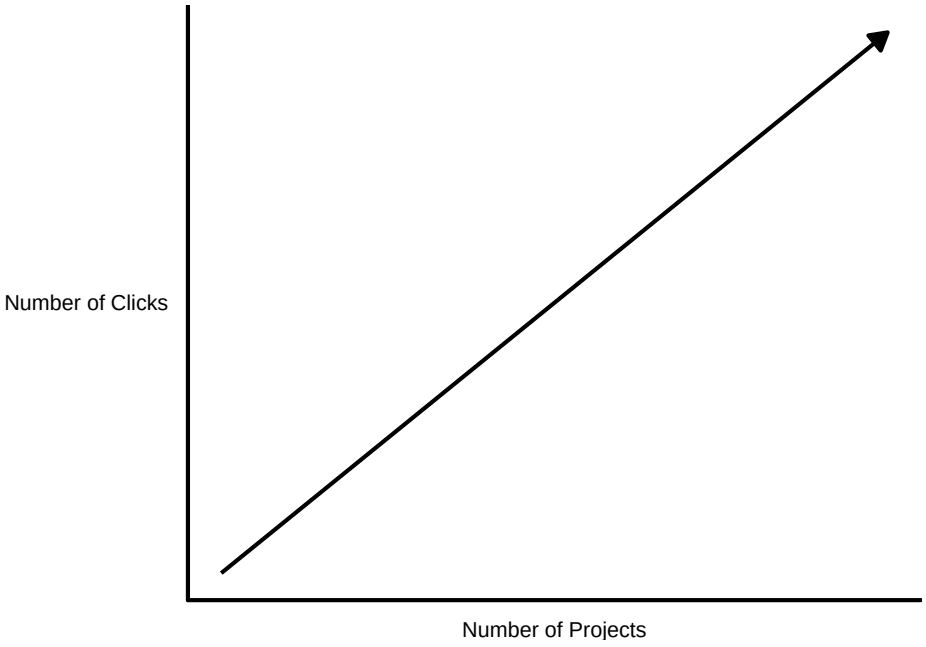
My own relationship with R goes back to 2016. At the time, I was a consultant, helping non-profits, government agencies, and educational institutions to measure the effectiveness of their work is (a field known as program evaluation). A lot of my work involved conducting surveys, analyzing the resulting the data, and sharing the results with clients.

The work itself was fine, but the tools I was using to do it were getting on my nerves. Well, one tool really: Excel.

Now look, this is not a place for an anti-Excel rant. Excel is a fine tool that has empowered millions to work with data in ways they would never have been able to otherwise.

But, for me, Excel was tedious. The amount of pointing and clicking I had to do when working with the amount of data I had got old fast. Each time I would conduct a survey, I'd know that it would yield an avalanche of data and that my wrists would end up exhausted from hours of pointing and clicking.

No matter what I did, analyzing data and creating charts in Excel just involved a lot of repetitive pointing and clicking. Kind of like this:



Endless pointing and clicking was just one problem I faced using Excel. Annoying though it was, it didn't affect the quality of my work. Or so I thought until I recalled a project I had worked on a few years earlier.

In this project, I was looking at which school districts in the state of Oregon have outdoor education programs known as Outdoor School. As part of this project, I had to download data on all school districts throughout Oregon, filter to only include relevant districts with fifth or sixth graders (the ages Outdoor School takes place), and then merge this with data that I collected as part of a survey I conducted.

I did the work in Excel, using a lot of (you guessed it!) pointing and clicking. The problem came when I was almost done with the project. I've blocked the details from my memory (as I've done with most things Excel-related), but what I do recall is that not being 100% certain I had done my filtering and joining correctly. And, to make it worse, I had no way to check my work. Why? Because all my pointing and clicking was ephemeral, gone in the ether as soon as I had completed it.

I finished the Outdoor School project and submitted my report. I think the work I did was *probably* accurate, but maybe it wasn't?

Now, you may be reading this thinking: why didn't you write down the steps you used in Excel so you could retrace them later? Sure, I could (and should) have done that. But let's be honest: most of us don't.

The reality is, we're human. We all make mistakes. And without a straightforward way to audit your work (and keeping a list of all of your Excel points

and clicks in a separate document is not, in my view, straightforward), mistakes will happen. If you've used Excel to work with data, I guarantee you've made a mistake, just like me.

The good news is that it's ok. There's a solution. And that solution is R.

If I were to redo that project on Outdoor School with R, here's what I'd do differently. Rather than watching points and clicks disappear into the ether, I'd write code that would serve as a record of everything I did. This code would:

Download data on all school districts:

Filter to only include districts with fifth or sixth graders:

Join the filtered data on school districts with my survey data:

Code can be scary. Having to write code is one of the reasons many people never learn R. But code is just a list of things you want to do to your data. It may be written in a hard-to-parse syntax (though it gets easier over time), but it's just a set of steps. The same steps that we should write out when we're working in Excel, but never do. Rather than having a separate document with my steps written down (the one that never gets written), I can see my steps in my code. See that line that says filter. Guess what it's doing? Yep, it's filtering!

If I had done things this way when working on the Outdoor School project, I could have looked back at any point to make sure what I thought was happening to my data was in fact happening. That nagging sensation I had near the end of the project that I may have made a mistake in one of my early points or clicks? It never would come up because I could just review my code to make sure it did what I thought it did. And if it didn't, I could rewrite and rerun my code to get updated results.

Using R won't mean you'll never make mistakes again (trust me, you will). But it will mean that you can easily spot your mistakes, make changes, and fix any issues.

I started learning R to avoid tedious pointing and clicking. But what I found was that R improved my work in ways I never expected. It's not just that my wrists are less tired. I now have more confidence that my work is accurate.

I used to feel ashamed about the way I use R.

I use R, a tool for statistical analysis, but I don't use it for complex statistical analysis. I don't do machine learning. I don't know what a random forest is. I've never even run a regression in R.

The only statistics I do in R are descriptive statistics. Counts, sums, averages: these are the statistics that I do in R.

For a long time, I felt like I wasn't a "real" R user. Real R users, in my mind, used R for hardcore stats. I "only" used R for descriptive stats.

I sometimes felt like I was using a souped up sports car to drive 20 miles an hour to the grocery store. What was the point in using a high-powered machine like R to do "simple" things?

Eventually, I realized that this framing misses the point. R started out as a tool created by statisticians for other statisticians. But, over a quarter century since its creation, R can do much more than statistical analysis.

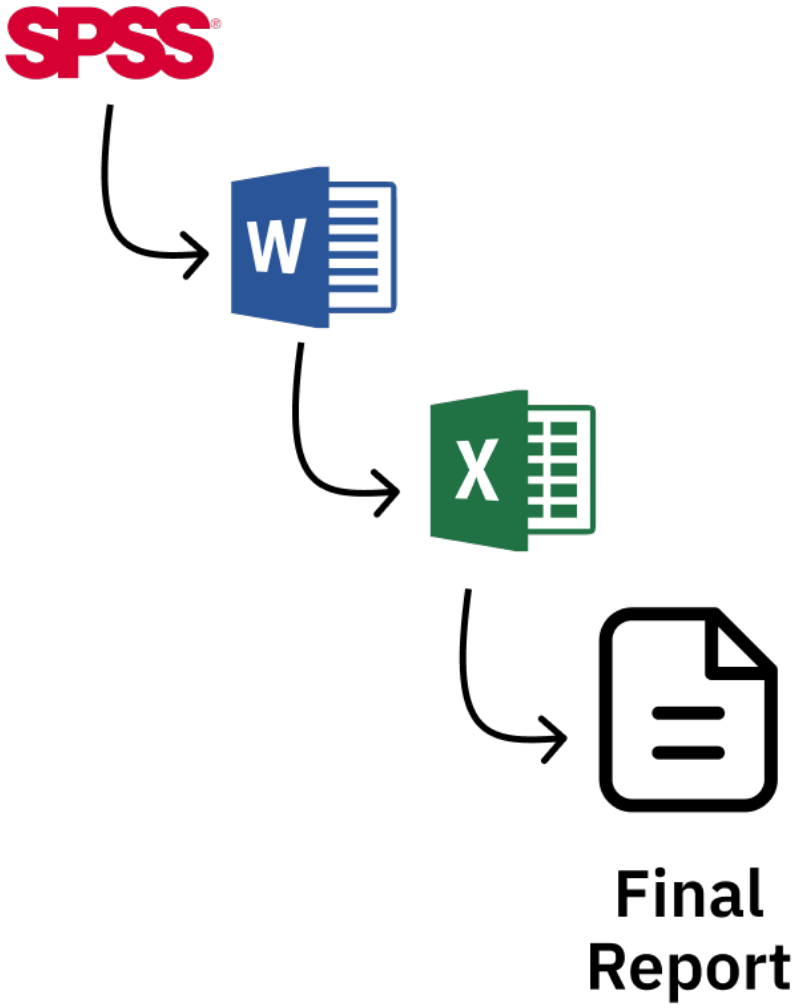
My own use of R is an example of this. I think of my work with R in three buckets:

Illuminate through data visualization: making graphs, maps, and tables that look good and share results effectively.

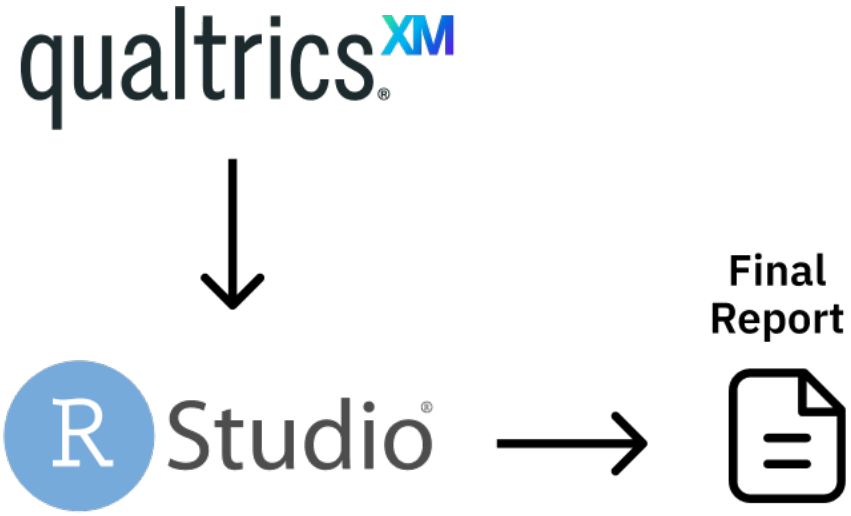
TODO: Add examples

Communicate by doing reporting with RMarkdown: moving away from the inefficiency and error-prone workflow of using multiple tools to create reports by instead doing it all in the one tool that I think of as R's killer feature.

TODO: Improve/explain graphics


SPSS
W
X
Final
Report

Automate tedious practices: Remember my Excel-burdened wrists? Since I moved to R I've found so many ways to automate tedious practices, from gathering data directly from the U.S. Census Bureau to pulling survey results in from Qualtrics and more.



The main reason I've come to accept that my way of using R is as valid as anyone else's has come through realizing that more "sophisticated" R users are doing many of the same things I am. Sure, they may also be doing statistical analyses that I am not, but everyone who uses R needs to illuminate, communicate, and automate.

Canadian statistician Sharla Gelfand has talked about how they used R to automate an annual report on nursing registration exams in Ontario. Sharla told me in 2019 that, despite being a statistician, the most statistical thing they did was calculating a median.

Take a look at the R community on Twitter (where users congregate under the #rstats hashtag). What gets people most excited is not the latest complex

statistical analysis. It's tips and tricks on the foundational work that everyone who uses R needs to do. Things like:

- Making illuminating data visualizations as part of the Tidy Tuesday project.
- Video tutorials on how to communicate through effective presentations using R.
- Love letters to the `clean_names()` function from the `janitor` package, which automates the process of making messy variable names easy to work with in R.

No matter what else you do in R, you have to **illuminate** your findings and **communicate** your results. And, the more you use R, the more you'll find yourself wanting to **automate** things you used to do manually (your wrists will thank you).

I realize now that the things that I use R for *are* the things that everyone uses R for. R was created for statistics. But today people are just as likely to use R without statistics.

Ten years ago, if you had told me I'd be writing a book on R, I'd have laughed. As someone with an extremely non-quantitative background (I did a PhD in anthropology) who never used R in graduate school, I never thought I'd be in a position to teach people about R. But here we find ourselves. And I'm excited to be your guide on this journey through the ways you can use R without statistics.

If I only used R for the things I thought "real" R users used it for, I wouldn't be writing this book. But, instead of slogging away in the world of complex statistical analysis, far outside of my area of expertise, I have found a place for myself in the world of R. Expanding my conception of what R can do has enabled me to get more out of this tool.

And here's the thing: if I, a qualitatively-trained anthropologist whose most complex statistical use for R is calculating averages, can find value in R, so can you. No matter what your background or what you think about R right now, using R without statistics can transform how you work in the future.

How This Book Works

This book shows the many ways that people use R without statistics. It's not comprehensive (trust me, there are many ways people use R not covered here). But I hope the ideas inspire you to think about learning to use R (if you're not yet an R user) or (if you are already on board the R train) learning to use R in ways you hadn't previously considered.

Each chapter focuses on one novel use of R. You'll begin by learning about a user or users who have transformed their work using R. You'll learn about a problem they had and how R helped them to solve it.We'll dive into their code, analyzing it line by line in order to help you understand how they used R. Each chapter will conclude with a short summary, offering lessons you can take from this novel way of using R.

I've tried to choose topics for each chapter that are relevant to a broad audience. Things like data visualization, report generation, and creating your own functions are things that anyone, no matter what you use R for, will find valuable.

There are some great topics that I thought to include but were just too narrow in their focus (for example, the world of generative art made with R. If, at any point while you're reading this book, you think, "why didn't David include X topic," please know that X might be a great topic, but I can only cover so much. The fact that you're able to come up with other ideas for things that R can do is a) fantastic and b) a further display of R's versatility. I eagerly await your follow-up book highlighting the myriad other things R can do that I am unable to cover in this book!

A Favor to Ask

Pedants of the world (as one of you, I come in peace), I have a favor to ask.

This book is called R Without Statistics. But it's not meant to be taken literally.

Of course it's true that if you're making a graph you're using statistics. So, before you start typing an angry email to me, please know that R Without Statistics is a mindset rather than a literal statement.

We're all using R with statistics already. Let's also learn to use R without statistics.

Illuminate

Use General Principles of High-Quality Data Viz in R

In the spring of 2021, nearly all of the American West was in a drought. In April of that year, officials in Southern California declared a water emergency, citing unprecedented conditions.

This wouldn't have come as news to those living in California and other Western states. In addition to the direct impact of drought (leading areas of California to implement water use restrictions), people could see the indirect impact of drought in increased wildfires. With forests dried out by years of drought conditions, wildfires became more frequent, filling skies in the West with smoke.

TODO: Add personal story

While more and more people are able to see the increase in drought conditions, communicating the extent of this change remains a challenge. How can we show the data in a way that accurately represents the data while is also compelling enough to have lay people take notice?

This was the challenge that freelance data visualization designers Cédric Scherer and Georgios Karamanis took on in November 2021. Commissioned by the magazine Scientific American to create a data visualization that would highlight the extent to which droughts in the United States have become common, they turned to the ggplot2 package to turn what could be (pardon the pun) dry data on droughts into a set of impactful data visualizations.

There was nothing unique about the data that Cédric and Georgios used. It was the same data from the National Drought Center that news organizations used in their stories. But what these two information designers did was visualize the data in a way that it both grabs attention and communicates clearly the scale of the phenomenon.

Close read of viz to show why it's effective

To understand why this visualization is effective, let's break it down into pieces.

At the broadest level, what we see as a single chart is actually a set of charts. Each rectangle represents one year in one region.

TODO: Add image

Looking at this single visualization of one year in one region, we can see that the x axis shows the week while the y axis shows the percentage of that region at different drought levels.

TODO: Add image

The stacked bars also use color to show the different drought levels. The lightest bar shows the percentage of the region that is abnormally dry while the darkest bar shows the percentage in exceptional drought conditions.

TODO: Add image

When I asked Cédric and Georgios to speak with me about this data visualization, they initially told me that the code for this piece might be too simple to highlight R's data viz power. No, I told them, I want to speak with you precisely *because* the code is not super complex. The fact that Cédric and Georgios were able to produce this complex graph with relatively simple code shows the power of R for data visualization. And this is made possible because of a theory called the grammar of graphics.

The grammar of graphics

If you've used Excel to make graphs, you're probably familiar with this menu:

TODO: Add image https://show.rfor.us/UewnER

Working in Excel, your graph-making journey begins with the step of selecting the type of graph you want to make. If you've only ever made data visualization in Excel, this first step may seem so obvious that you've never even considered conceptualizing the process of creating data visualization in any different way. This was certainly the case for me in my years as an Excel user.

But some people think of data visualization at a much deeper level. One of these was the late statistician Leland Wilkinson. Wilkinson thought deeply for years about what data visualization is and how we can describe it. In 1999, he published a book called *The Grammar of Graphics* that sought to develop a consistent way of describing *all* graphs.

Wilkinson argued that we should think of plots not as distinct types a la Excel, but as following a grammar that we can use to describe *any* plot. Throughout the book that Wilkinson is best remembered for, he presented general principles to describe graphs. Just as knowledge of English grammar tells us that a noun followed by a verb ("he goes") works while the opposite ("goes he") does not, knowledge of the grammar of graphics allows us to understand why certain graph types "work." Or, as Wilkinson put it,

> A language consisting of words and no grammar (statement = word) expresses only as many ideas as there are words. … The grammar of graphics takes us beyond a limited set of charts (words) to an almost unlimited world of graphical forms (statements).

As Paul Velleman and Howard Wainer wrote in an obituary for Wilkinson, *The Grammar of Graphics* is "not a book to curl up with in front of a fire on a cold winter's night." They continued: "As literature, the plot is weak, but as science, the plots are better described than you'll find anywhere else."

Thinking about data visualization through the lens of the grammar of graphics allow us to see that graphs typically have data that is plotted on the x axis and other data that is plotted on the y axis. And this is the case no matter whether the type of graph we end up with is, to take just two examples, a bar chart of a line chart. Consider these two graphs:

TODO: Add images of bar chart and line chart showing same data

While they look different (and would, to the Excel user, be different types of graphs), Wilkinson's grammar of graphics allows us to see similarities in them.

As an academic statistician, Wilkinson's goal in writing *The Grammar of Graphics* was to provide a novel way of thinking about data visualization. But his feelings on graph-making tools like Excel were clear when he wrote that "most charting packages channel user requests into a rigid array of chart types. To atone for this lack of flexibility, they offer a kit of post-creation editing tools to return the image to what the user originally envisioned."

The answer to this unspoken request for product would come in 2010, when Hadley Wickham announced the `ggplot2` package for R. Implementing Wilkinson's ideas not only to describe graphs, but also providing the tools to make them, `ggplot2` would come to revolutionize the world of data visualization.

ggplot2

Hadley Wickham's article announcing `ggplot2` (which I, like nearly everyone in the data viz world, will refer to simply as ggplot) was titled "A Layered Grammar of Graphics." This idea of layering graphical elements is key to understanding how ggplot works. Let's walk through some of the most important layers.

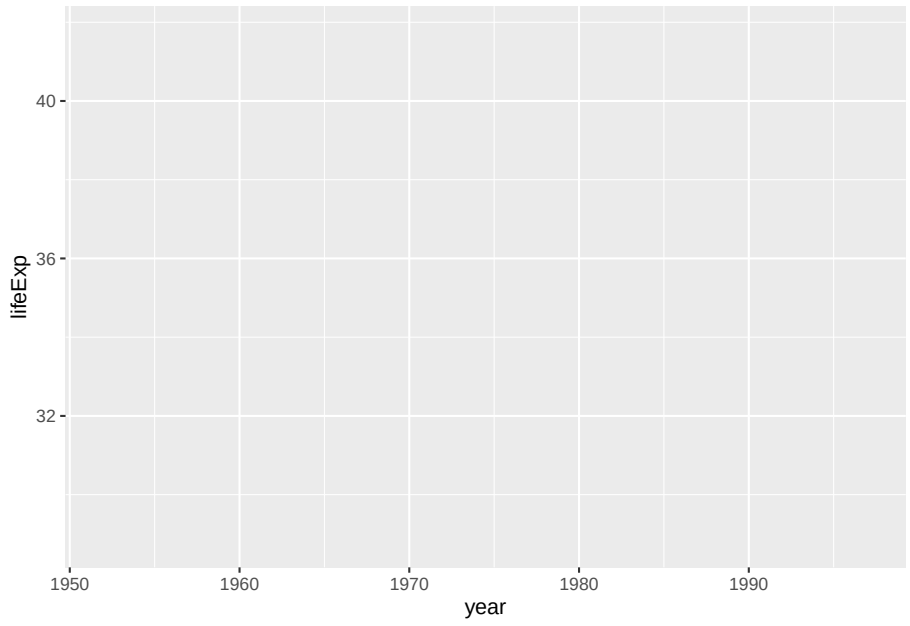
When creating a graph with ggplot, we begin by mapping data to aesthetic properties. To the uninitiated, this may sound like complete nonsense. But all it means is that we use things like the x or y axis, color, size (aka aesthetic properties) to represent variables.

Let's use some data to make this concrete. Below are 10 (out of over 1,700 total) rows of data from the gapminder dataset made famous by Swedish TODO: add background on him Hans Rosling (TODO: cite).

```
#> # A tibble: 10 x 6
#>    country     continent  year lifeExp      pop gdpPercap
#>    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
#>  1 Afghanistan Asia       1952    28.8  8425333      779.
#>  2 Afghanistan Asia       1957    30.3  9240934      821.
#>  3 Afghanistan Asia       1962    32.0 10267083      853.
#>  4 Afghanistan Asia       1967    34.0 11537966      836.
#>  5 Afghanistan Asia       1972    36.1 13079460      740.
#>  6 Afghanistan Asia       1977    38.4 14880372      786.
#>  7 Afghanistan Asia       1982    39.9 12881816      978.
#>  8 Afghanistan Asia       1987    40.8 13867957      852.
#>  9 Afghanistan Asia       1992    41.7 16317921      649.
#> 10 Afghanistan Asia       1997    41.8 22227415      635.
```

If we want to make a chart, we need to first decide which variable to use to put on the x axis and which to put on the y axis. Let's say we want to show life expectancy over time. That means using the variable `year` on the x axis and the variable `lifeExp` on the y axis.

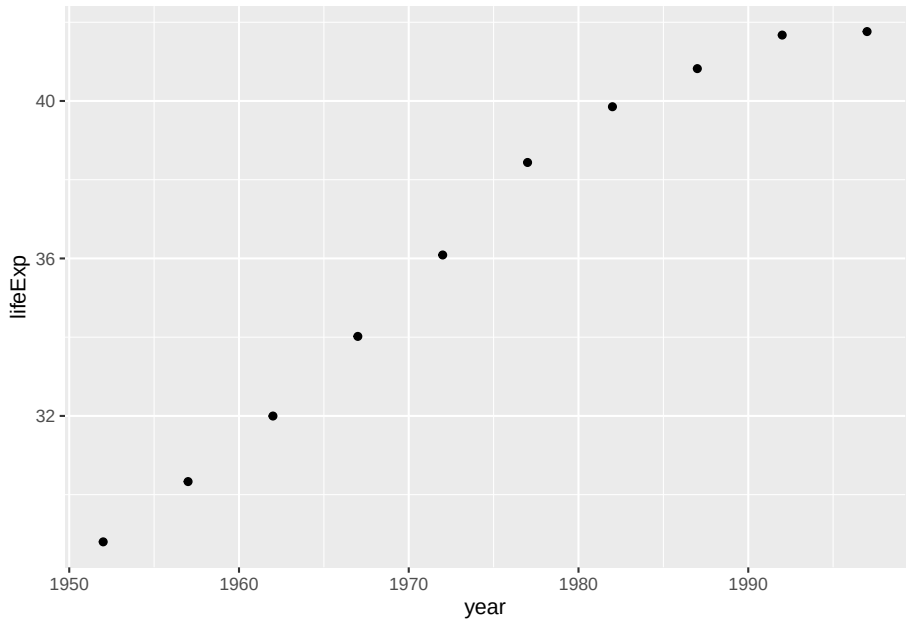
I begin by using the `ggplot()` function. Within this, I tell R that I'm using the data frame `gapminder_10_rows` (this is the shortened version I saved from the full `gapminder` data frame). The line following this tells R to use `year` on the x and `lifeExp` on the y axis. When I run my code, what I get doesn't look like much.



But if I look closely, I can see the beginnings of a plot. Remember that x axis using `year`? There it is! And `lifeExp` on the y axis? Yup, it's there too.

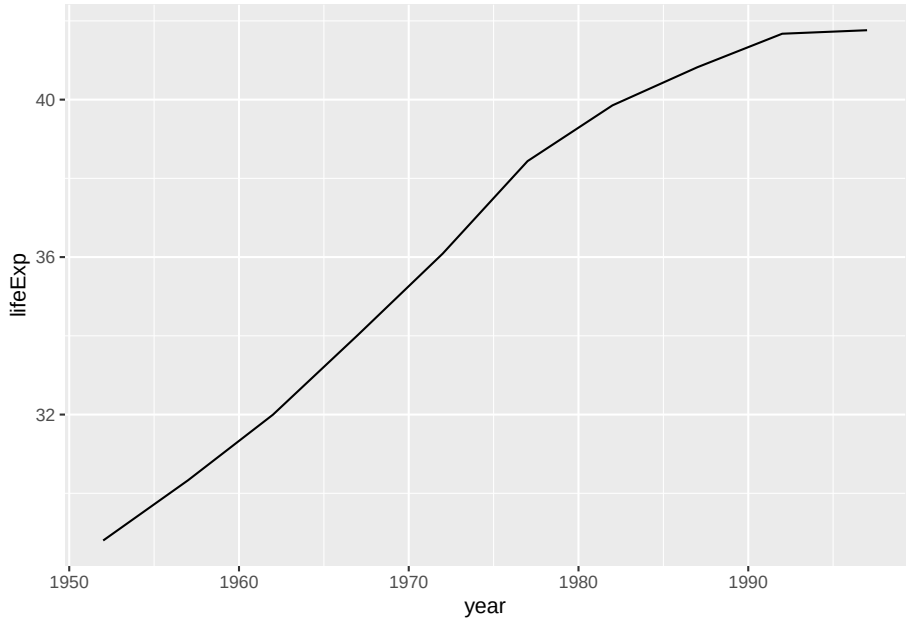
I can also see that the values on the x and y axes match up to our data. In the `gapminder_10_rows` data frame, the first year is 1952 and the last year is 1997. The range of the x axis seems to have been created with this data in mind (spoiler: it was). And `lifeExp`, which goes from about 28 to about 42 will fit nicely on our y axis.

Axes are nice, but we're missing any type of visual representation of the data. To get this, we need to add the next layer in ggplot: geoms. geoms, short for geometric objects, are different ways of representing data. For example, if we want to add points, we use `geom_point()`.

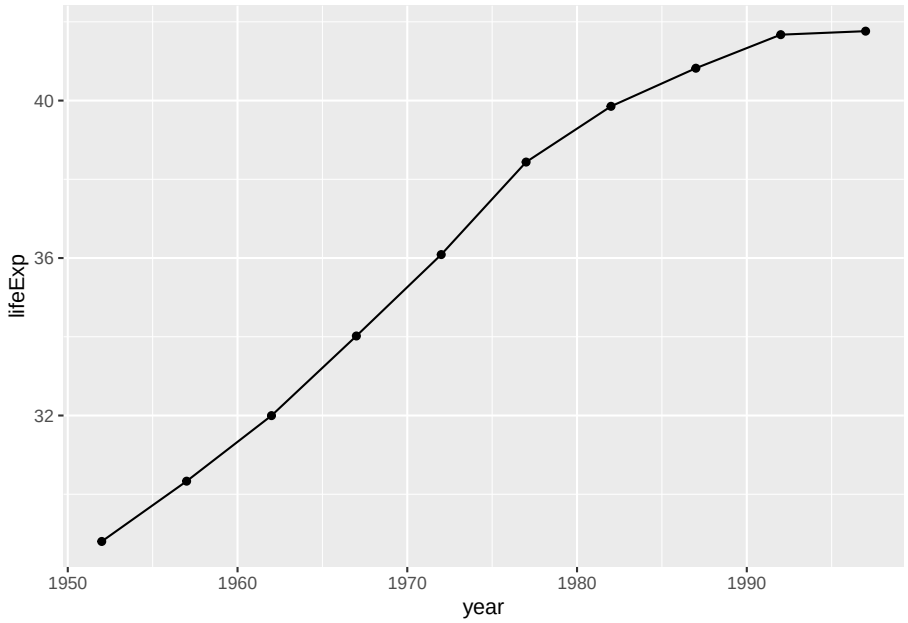


There we go! 1952 shows the life expectancy of about 28 and so on through every year in our data.

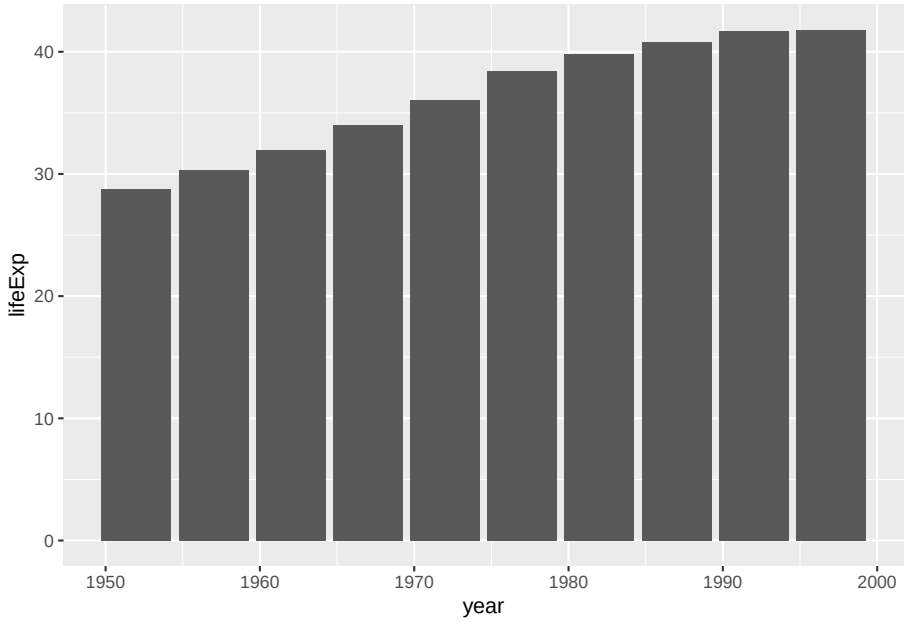
Let's say we change our mind and want to make a line chart instead. Well, all we have to do is replace `geom_point()` with `geom_line()`.



Or (and now we're really getting fancy), what if we add *both* `geom_point()` and `geom_line()`? A line chart with points!

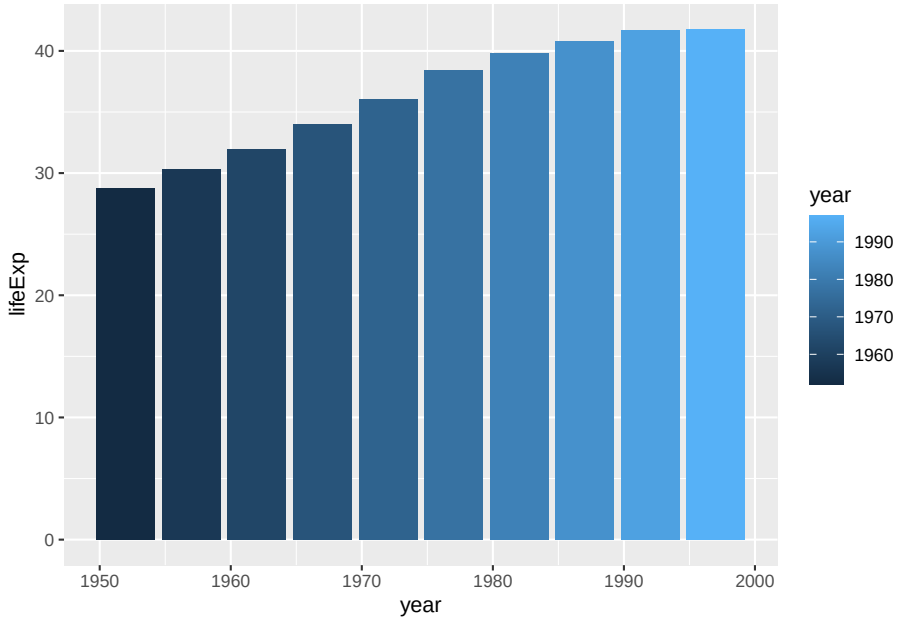


We can extend this idea further, swapping in `geom_col()` to create to a bar chart.

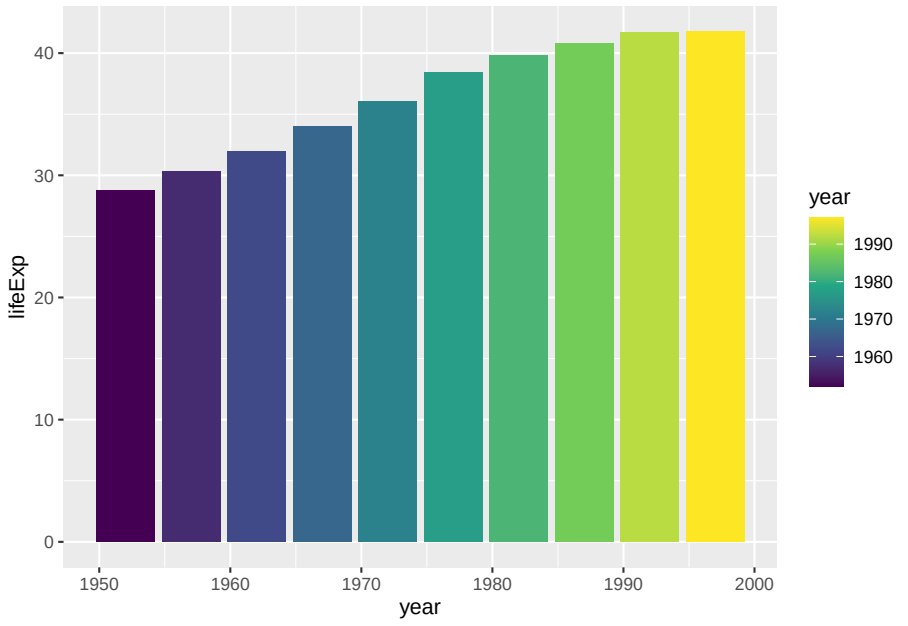


I hope you're seeing how ggplot is a direct implementation of Wilkinson's grammar of graphics. The difference between a line chart and a bar chart isn't that great. Both can have the same aesthetic properties (namely, putting year on the x axis and life expectancy on the y axis), but simply use different geometric objects to visually represent the data.

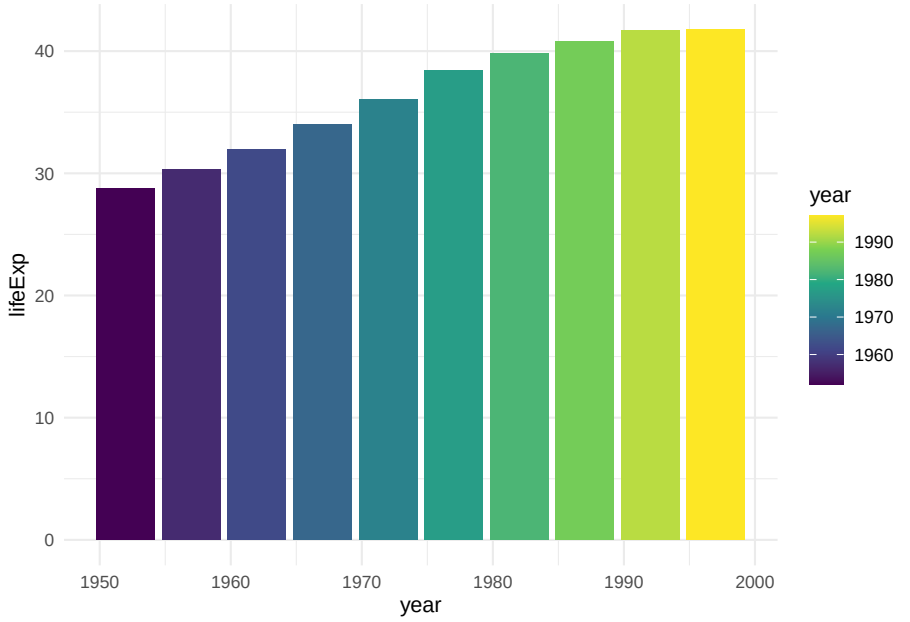
Before we return to the drought data viz, let's look at a few additional layers that can help us can alter our bar chart. Let's say we want to change the color of our bars. In the grammar of graphics approach to chart-making, this means mapping some variable to the aesthetic property of fill (slightly confusingly, the aesthetic property color would, for a bar chart, change the outline of each bar). In the same way that we mapped `year` to the x axis and y to `lifeExp`, we can also map fill to a variable. Let's try mapping fill to the year variable.



What we see now is that, for earlier years, the fill is darker while for later years, it is lighter (the legend, added to the right of our plot, shows this). What if we want to change the fill color? For that, we use a scale function. In this case, I'll use the `scale_fill_viridis_c()` function (the c at the end of the function name refers to the fact that the data is not continuous). This function, just one of many functions that start with `scale_` and can alter the fill scale, changes the default palette to one that is colorblind-friendly and prints well in grayscale.



Another layer we can use is the theme layer. This layer allows us to change the overall look-and-feel of plots (think: plot backgrounds, gridlines, etc). Just as there are a number of `scale_` functions, there are also a number of functions that start with `theme_`. Below, I've added `theme_minimal()`, my go-to theme, which gives us a much more streamlined look.



While adding `theme_minimal()` massively improves any plot, our bar chart here is not anything I would put forward as high-quality data visualization. But, at the very least, we've seen building a chart with ggplot involves working with multiple layers:

- First, we select variables to map to aesthetic properties such as x or y axis, color/fill, etc
- Second, we choose the geometric object (aka geom) we want to use to represent our data
- Third, if we want to change aesthetic properties (for example, using a different palette), we do this with a `scale_` function
- Fourth, we use a `theme_` function to set the overall look-and-feel of our plot.

This is, of course, just scratching the surface of what is possible with ggplot. There are many ways you could improve this plot. But rather than improving an ugly plot, let's instead return to the drought data viz that Cédric Scherer and Georgios Karamanis made. Going through their code will show us some familiar aspects of ggplot – and present some tips on how to make high-quality data visualization with R.

Let's look at a few other layers that we can

Develop a Custom Theme to Keep Your Data Viz Consistent

R is a Full-Fledged Map-Making Tool

Make Tables That Look Good and Share Results Effectively

https://clauswilke.com/dataviz/figure-titles-captions.html#tables

Communicate

Use RMarkdown to Communicate Accurately and Efficiently

Use RMarkdown to Instantly Generate Hundreds of Reports

Create Beautiful Presentations with RMarkdown

Make Websites to Share Results Online

- When to do static vs when you need Shiny

Automate

Access Up to Date Census Data with the `tidycensus` Package

Pull in Survey Results as Soon as They Come In

Stop Copying and Pasting Code by Creating Your Own Functions

Bundle Your Functions Together in Your Own R Package

Conclusion

Come for the Data, Stay for the Community