

ggplot2: Unleash Your Inner Artist

Tuhin rana

2024-03-17

Table of contents

Before We Start

In December 2017, after completing my first year of statistics, I delved into learning R. Having a background in Java, C, and C++ coding since high school, I enjoyed R but found its plots not so appealing and the code a bit tricky. On a quest for something beautiful and user-friendly, I stumbled upon a blog titled [Beautiful plotting in R: A ggplot2 cheatsheet](#) by Zev Ross, last updated in January 2016. Intrigued, I decided to follow the tutorial step by step, learning a great deal. As time passed, I tweaked and expanded the codes, adding new chart types and resources.

Realizing that Zev Ross's blog hadn't been updated for years, I took the initiative to create my own version, incorporating updates like the amazing `{patchwork}`, `{ggtext}`, and `{ggforce}` packages. I also shared insights on custom fonts, colors, and introduced a variety of R packages for interactive charts. The journey led to a unique tutorial, and now, I've decided to make it public, adding even more updates, such as Maps! because who doesn't love maps!!

I incorporated the following enhancements into my tutorial:

- Following the R style guide (e.g., by [Hadley Wickham](#), [Google](#), or the [Coding Club](#) style guides).
- Implementing changes to the style and aesthetics of plots, including axis titles, legends, and color schemes for all plots.
- Ensuring that the tutorial remains up-to-date with changes in `{ggplot2}` (current version: 3.4.0).
- Modifying data import methods to utilize GitHub as a data source.
- Offering additional tips on various topics such as chart selection, color palettes, title modifications, adding lines, adjusting legends, annotations with labels, arrows and boxes, multi-panel plots, Geospatial Visualizations and interactive visualizations. ...

What you'll discover

- Tie Your Seatbelt: Setting the stage for your journey into advanced plotting techniques.
- The Dataset: Understanding the importance of data in crafting compelling visualizations.
- The `{ggplot2}` Package: Unleashing the power of `{ggplot2}` for elegant and customizable plots.
- A Default ggplot: Exploring the basics with a default `{ggplot}` and understanding its components.

- Working with Axes: Mastering the art of manipulating axes to convey meaningful insights.
- Working with Titles: Crafting informative and visually appealing titles to captivate your audience.
- Working with Legends: Enhancing clarity and interpretation by effectively managing legends.
- Working with Backgrounds & Grid Lines: Elevating aesthetics with stylish backgrounds and grid lines.
- Working with Margins: Fine-tuning margins to optimize plot presentation.
- Working with Multi-Panel Plots: Diving into the world of multi-panel plots for comprehensive data representation.
- Working with Colors: Harnessing the power of color to convey information and evoke emotions.
- Working with Themes: Creating cohesive visual narratives with carefully curated themes.
- Working with Lines: Adding emphasis and clarity through strategic use of lines.
- Working with Text: Utilizing text annotations to provide context and highlight key findings.
- Working with Coordinates: Manipulating coordinates to achieve desired plot layouts and perspectives.
- Working with Chart Types: Expanding your repertoire with diverse and impactful chart types.
- Working with Ribbons (AUC, CI, etc.): Enhancing visualizations with ribbons for confidence intervals and more.
- Working with Smoothings: Incorporating smoothings to reveal underlying trends and patterns.
- Working with Interactive Plots: Engaging your audience with interactive visualizations for dynamic exploration.
- Remarks, Tipps & Resources: Leveraging insights, tips, and resources to further refine your plotting skills.

1 Tie Your Seatbelt

To fully execute the tutorial, you'll need to install the following packages:

- `{ggplot2}`, a part of the `{tidyverse}` package collection
- `{tidyverse}` package collection, including:
 - `{dplyr}` for data wrangling
 - `{tibble}` for modern data frames
 - `{tidyr}` for data cleaning
 - `{forcats}` for handling factors
- `{corrr}` for calculating correlation matrices
- `{cowplot}` for composing ggplots
- `{ggforce}` for creating sine plots and other advanced visualizations
- `{ggrepel}` for enhancing text labeling in plots
- `{ggridges}` for creating ridge plots
- `{ggsci}` for accessing nice color palettes
- `{ggtext}` for advanced text rendering in plots
- `{ggthemes}` for additional plot themes
- `{grid}` for creating graphical objects
- `{gridExtra}` for additional functions for “grid” graphics
- `{patchwork}` for generating multi-panel plots
- `{prismatic}` for manipulating colors
- `{rcartocolor}` for accessing great color palettes
- `{scico}` for perceptual uniform color palettes
- `{showtext}` for utilizing custom fonts
- `{shiny}` for developing interactive apps
- Several packages for interactive visualizations, including:
 - `{charter}`
 - `{echarts4r}`
 - `{ggiraph}`
 - `{highcharter}`
 - `{plotly}`

```
# install CRAN packages
install.packages(
  c("ggplot2", "tibble", "tidyr", "forcats", "purrr", "prismatic", "corrr",
    "cowplot", "ggforce", "ggrepel", "ggridges", "ggsci", "ggtext", "ggthemes",
    "grid", "gridExtra", "patchwork", "rcartocolor", "scico", "showtext",
    "shiny", "plotly", "highcharter", "echarts4r")
)

# install from GitHub since not on CRAN
install.packages(devtools)
devtools::install_github("JohnCoene/charter")
```

For instructional purposes, and to ensure smooth transitions for learners navigating directly to specific plots, I'll load the necessary package beside `{ggplot2}` in the corresponding section.

2 The Dataset

We are utilizing data from the *National Morbidity and Mortality Air Pollution Study* (NMMAPS), focusing specifically on data pertaining to Chicago and the years 1997 to 2000 to ensure manageability of the plots. For a more comprehensive understanding of this dataset, readers can refer to Roger Peng's book [Statistical Methods in Environmental Epidemiology with R](#).

To import the data into our R session, we can employ `read_csv()` from the `{readr}` package. Subsequently, we'll store the data in a variable named `chic` using the *assignment arrow* `<-`. Just Copy and Paste the following code.

```
chic <- readr::read_csv("https://raw.githubusercontent.com/rana2hin/ggplot_guide/master/chic.csv")
```

```
Rows: 1461 Columns: 10
-- Column specification -----
Delimiter: ","
chr (3): city, season, month
dbl (6): temp, o3, dewpoint, pm10, yday, year
date (1): date

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

The `::` symbolizes *namespace* and enables accessing a function without loading the entire package. Alternatively, you could load the `readr` package first using `library(readr)` and then execute `chic <- read_csv(...)` subsequently.

```
tibble::glimpse(chic)
```

```
Rows: 1,461
Columns: 10
$ city      <chr> "chic", "chic", "chic", "chic", "chic", "chic", "chic", "chic", "chic", "chic"
$ date      <date> 1997-01-01, 1997-01-02, 1997-01-03, 1997-01-04, 1997-01-05, ...
$ temp      <dbl> 36.0, 45.0, 40.0, 51.5, 27.0, 17.0, 16.0, 19.0, 26.0, 16.0, ...
$ o3        <dbl> 5.659256, 5.525417, 6.288548, 7.537758, 20.760798, 14.940874, ...
```

```
$ dewpoint <dbl> 37.500, 47.250, 38.000, 45.500, 11.250, 5.750, 7.000, 17.750, ~
$ pm10    <dbl> 13.052268, 41.948600, 27.041751, 25.072573, 15.343121, 9.3646~
$ season   <chr> "Winter", "Winter", "Winter", "Winter", "Winter", "Winter", "~
$ yday     <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18~
$ month    <chr> "Jan", "Jan", "Jan", "Jan", "Jan", "Jan", "Jan", "Jan", "Jan"~
$ year     <dbl> 1997, 1997, 1997, 1997, 1997, 1997, 1997, 1997, 1997, 1997, 1~
```

```
library(gt)
head(chic, 10) %>% gt()
```

city	date	temp	o3	dewpoint	pm10	season	yday	month	year
chic	9862	36.0	5.659256	37.500	13.052268	Winter	1	Jan	1997
chic	9863	45.0	5.525417	47.250	41.948600	Winter	2	Jan	1997
chic	9864	40.0	6.288548	38.000	27.041751	Winter	3	Jan	1997
chic	9865	51.5	7.537758	45.500	25.072573	Winter	4	Jan	1997
chic	9866	27.0	20.760798	11.250	15.343121	Winter	5	Jan	1997
chic	9867	17.0	14.940874	5.750	9.364655	Winter	6	Jan	1997
chic	9868	16.0	11.920985	7.000	20.228428	Winter	7	Jan	1997
chic	9869	19.0	8.678477	17.750	33.134819	Winter	8	Jan	1997
chic	9870	26.0	13.355892	24.000	12.118381	Winter	9	Jan	1997
chic	9871	16.0	10.448264	5.375	24.761534	Winter	10	Jan	1997

3 The ggplot2 Package

ggplot2 is a graphics system that facilitates the declarative creation of visualizations, founded on principles outlined in [The Grammar of Graphics](#). With ggplot2, you furnish the data, specify how variables should be mapped to aesthetics, define graphical parameters to employ, and the system handles the rest.

A ggplot is made up several key elements:

1. **Data**: Your raw dataset that you want to visualize.
2. **Geometries geom_**: These are the shapes that represent your data, like points, lines, or bars.
3. **Aesthetics aes()**: This controls how your data is visually represented, including aspects like color, size, and shape.
4. **Scales scale_**: These map the data onto the aesthetic dimensions, like converting data values to plot dimensions or factor values to colors.
5. **Statistical transformations stat_**: These are statistical summaries of your data, such as calculating quantiles or fitting curves.
6. **Coordinate system coord_**: This defines how your data coordinates are mapped onto the plot's coordinate system.
7. **Facets facet_**: This organizes your data into a grid of plots based on specified variables.
8. **Visual themes theme()**: These set the overall appearance of your plot, covering things like background, grids, axes, default fonts, sizes, and colors.

The number of elements may vary depending on the situation you're working on.

3.1 A Default ggplot

Before diving into the capabilities of {ggplot2}, we need to load the package. Alternatively, we can load it through the [tidyverse package collection](#):

```
library(ggplot2)
# Or,
library(tidyverse)
```

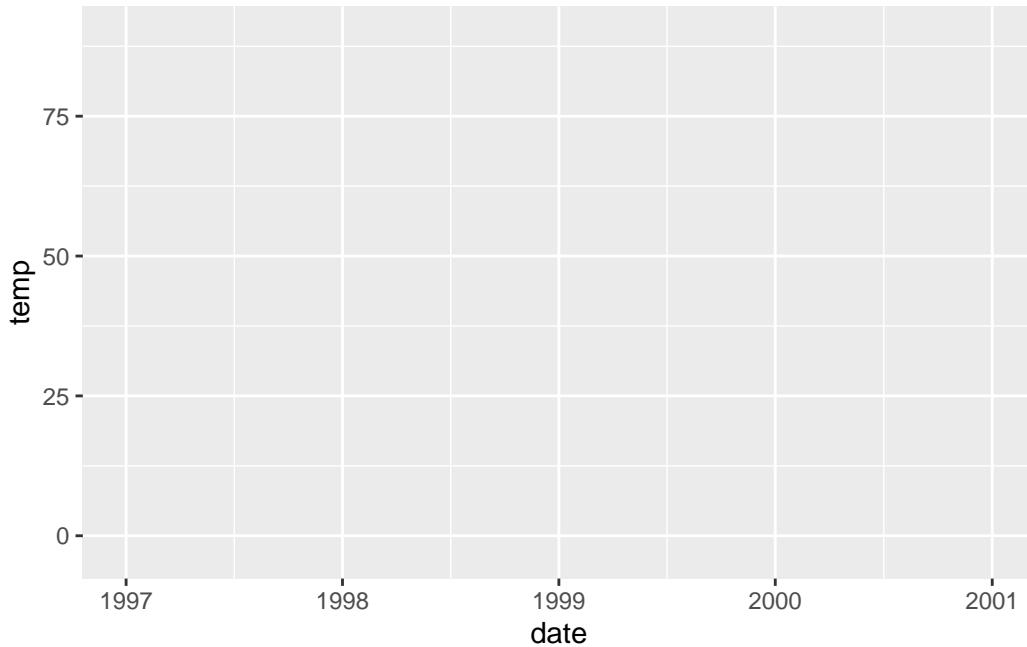
```
-- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v lubridate 1.9.3     v tibble    3.2.1
v purrr     1.0.2     v tidyr    1.3.1
-- Conflicts ----- tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become non-conflicting
```

The syntax of `{ggplot2}` differs from base R. Following the basic elements, a default ggplot requires specifying three things: the *data*, *aesthetics*, and a *geometry*. To begin defining a plotting object, we call `ggplot(data = df)`, indicating that we'll work with that dataset. Typically, we aim to plot two variables—one on the x-axis and one on the y-axis. These are *positional aesthetics*, so we add `aes(x = var1, y = var2)` to the `ggplot()` call (where `aes()` denotes aesthetics). However, there are cases where one may need to specify one, three, or more variables, which we'll address later.

We indicate the data *outside* of `aes()` and include the variables that ggplot maps to the aesthetics *inside* of `aes()`.

In this instance, we assign the variable `date` to the x-position and the variable `temp` to the y-position. Subsequently, we'll also map variables to various other aesthetics such as color, size, and shape.

```
(g <- ggplot(chic, aes(x = date, y = temp)))
```



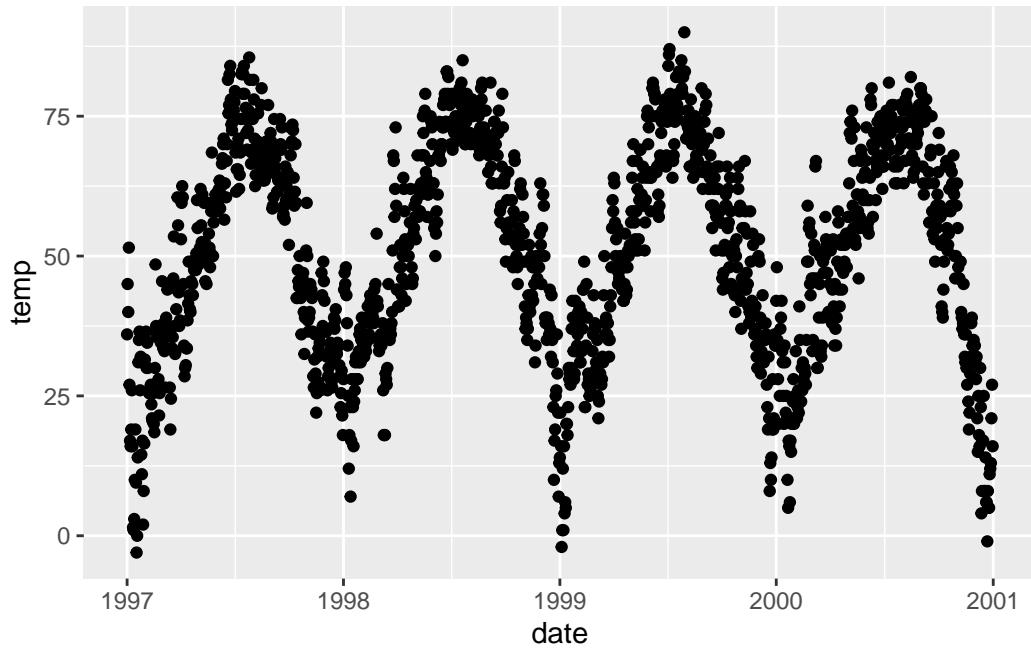
Ah, the reason only a panel is generated when executing this code is because `{ggplot2}` lacks information on how we want to visualize the data. We still need to specify a geometry!

In `{ggplot2}`, you can store the current `ggobject` in a variable of your choosing, such as `g`. This allows you to extend the `ggobject` later by adding additional layers, either all at once or by assigning it to the same or another variable.

By using parentheses when assigning an object, the object will be printed immediately. Instead of writing `g <- ggplot(...)` followed by `g`, we can simply write `(g <- ggplot(...))`.

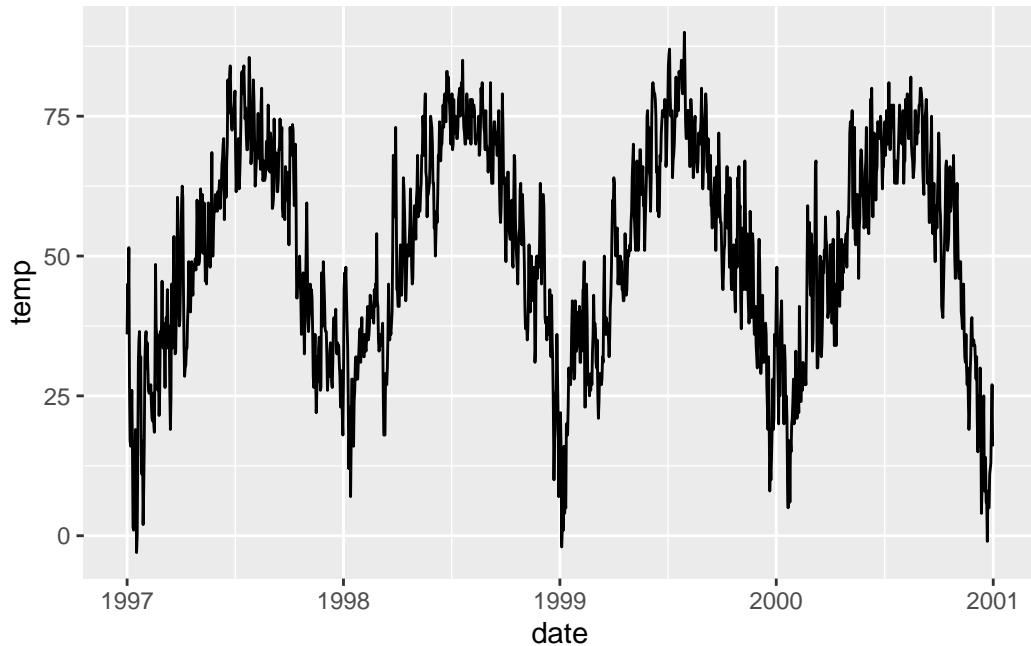
There's a wide array of geometries in `{ggplot2}`, often referred to as *geoms* because their function names typically start with `geom_`. You can find the full list of default geoms [here](#), and there are even more options available through extension packages, which you can explore [here](#). To instruct `{ggplot2}` on the style we want to use, we can, for example, add `geom_point()` to create a scatter plot:

```
g + geom_point()
```



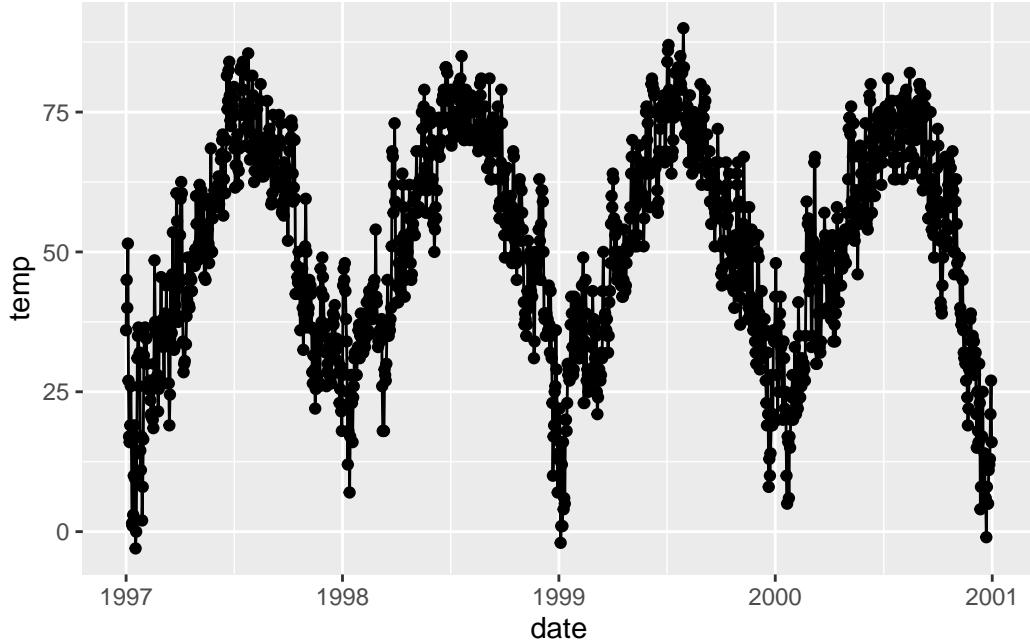
Great! However, this data could also be represented as a line plot (although it might not be the optimal choice, but it's a common practice). So, we can simply replace `geom_point()` with `geom_line()` and boom!

```
g + geom_line()
```



Indeed, one can combine multiple geometric layers, and this is where the magic and fun truly begin!

```
g + geom_line() + geom_point()
```

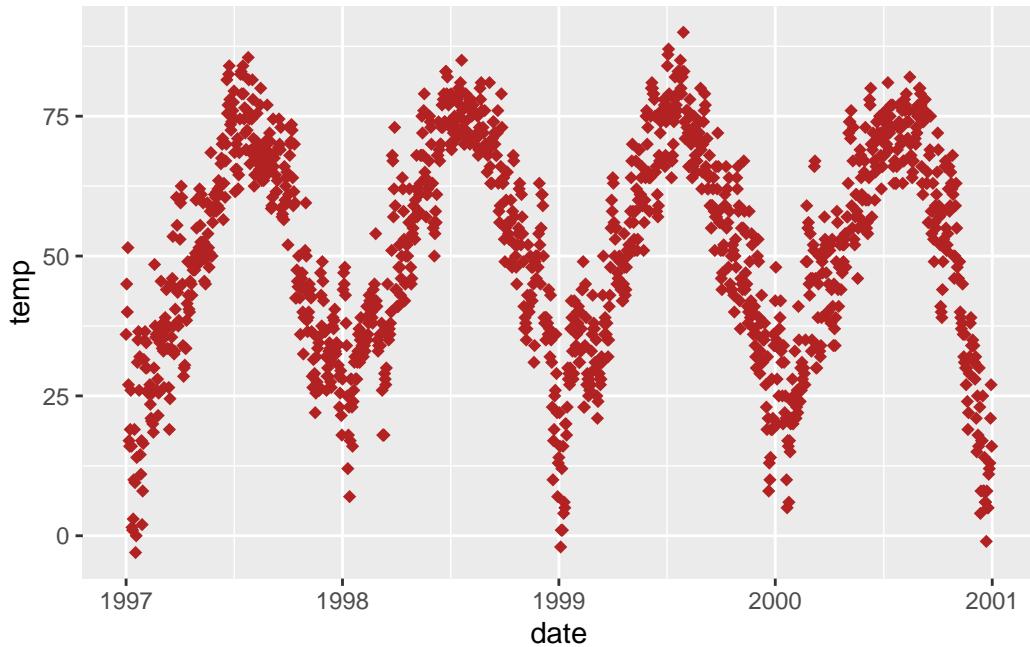


That's enough discussion on geometries for now. Don't worry, we'll dive into various plot types at a later point, as outlined here.

3.1.1 Change Properties of Geometries

Within the `geom_*` command, you can already manipulate visual aesthetics such as the color, shape, and size of your points. Let's transform all points into large fire-red diamonds!

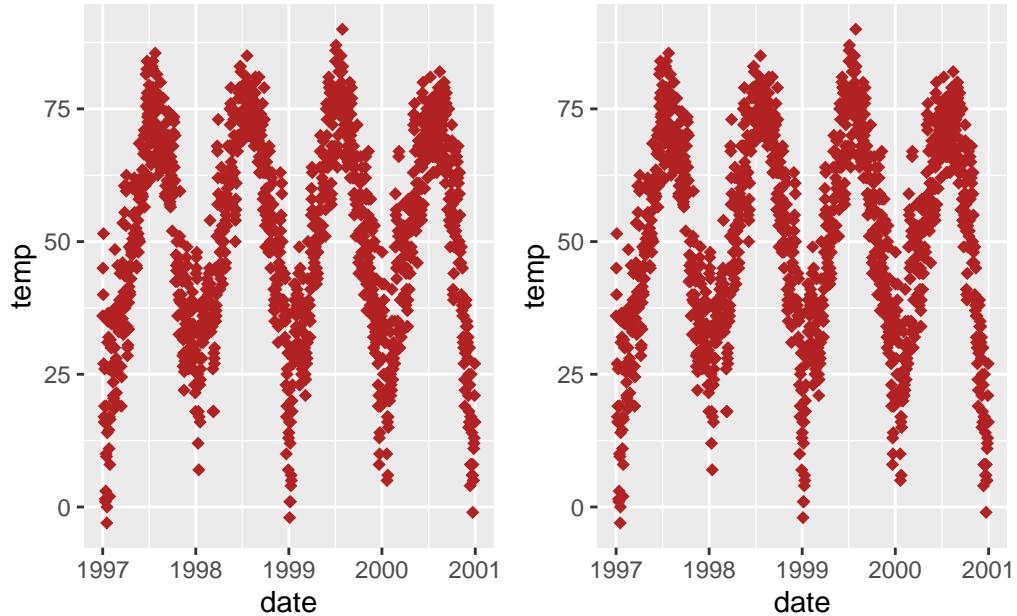
```
g + geom_point(color = "firebrick", shape = "diamond", size = 2)
```



`{ggplot2}` understands both `color` and `colour` as well as the short version `col`.

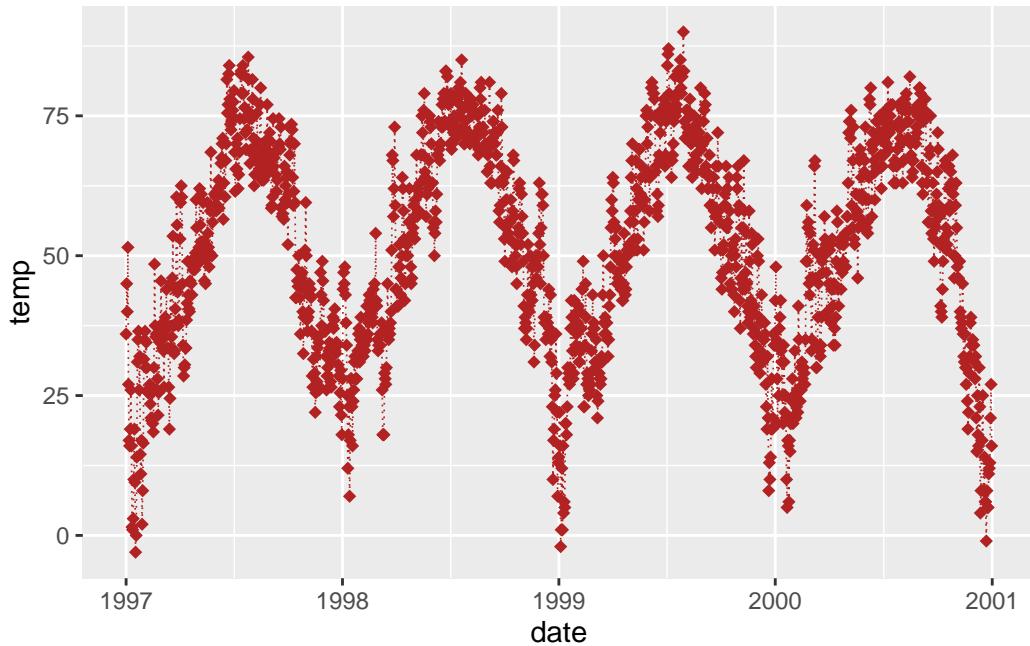
You can utilize preset colors (a full list can be found [here](#)) or [hex color codes](#), both enclosed in quotes. Additionally, you can specify RGB/RGBA colors using the `rgb()` function. Click to expand:

```
g + geom_point(color = "#b22222", shape = "diamond", size = 2)
g + geom_point(color = rgb(178, 34, 34, maxColorValue = 255), shape = "diamond", size = 2)
```



Each geom has its unique properties, referred to as *arguments*, and the same argument might produce different effects depending on the geom you're employing.

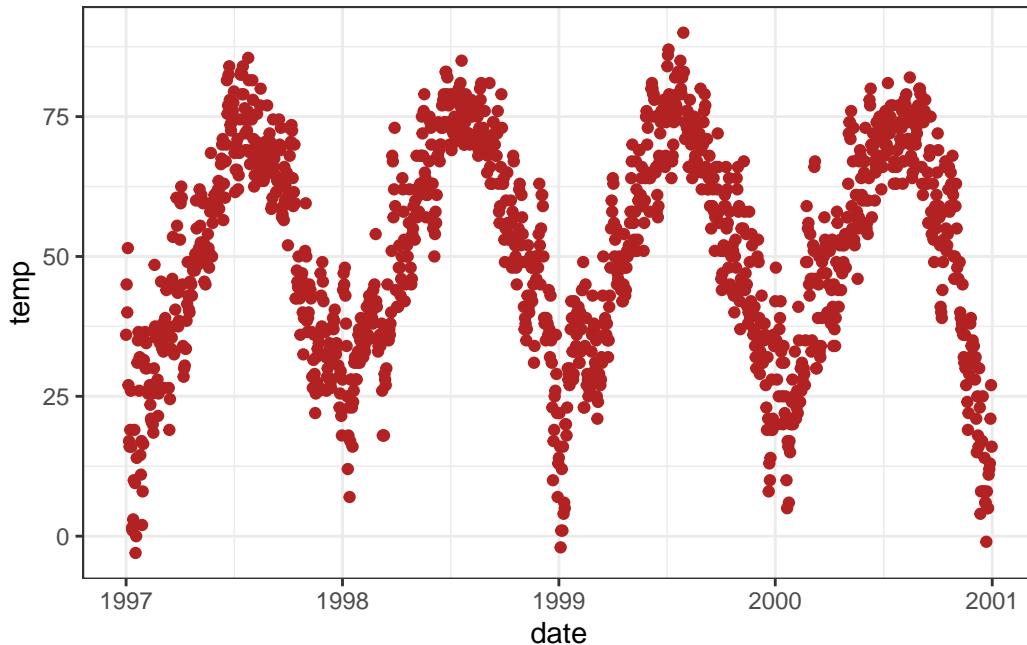
```
g + geom_point(color = "firebrick", shape = "diamond", size = 2) +
  geom_line(color = "firebrick", linetype = "dotted", lwd = .3)
```



3.1.2 Replace the default ggplot2 theme

To further demonstrate ggplot's versatility, let's enhance the appearance by removing the default grayish {ggplot2} style and setting a different built-in theme, such as `theme_bw()`. By using `theme_set()`, all subsequent plots will adopt the same black-and-white theme. This adjustment will notably enhance the appearance of the red points!

```
theme_set(theme_bw())  
  
g + geom_point(color = "firebrick")
```



For further details on using built-in themes and customizing themes, refer to the section “Working with Themes”. Starting from the next chapter, we'll also utilize the `theme()` function to customize specific elements of the theme.

theme() is a crucial command for manually adjusting various theme elements such as texts, rectangles, and lines.

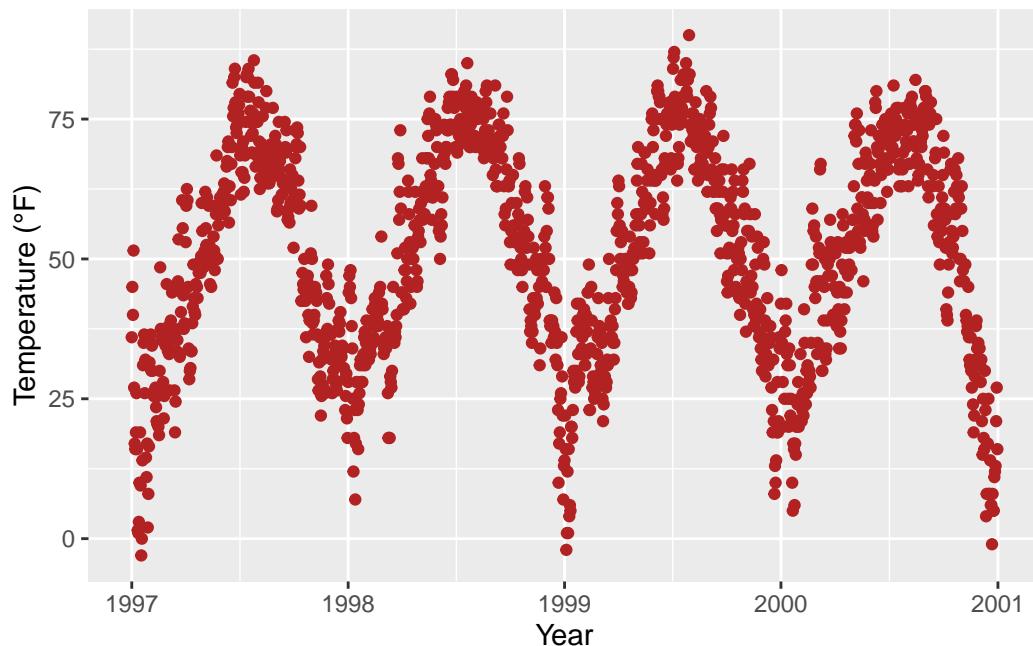
To explore the numerous details of a ggplot theme that can be modified, refer to the extensive list available [here](#). Take your time, as it's a comprehensive list!

4 Working with Axes

4.1 Change Axis Titles

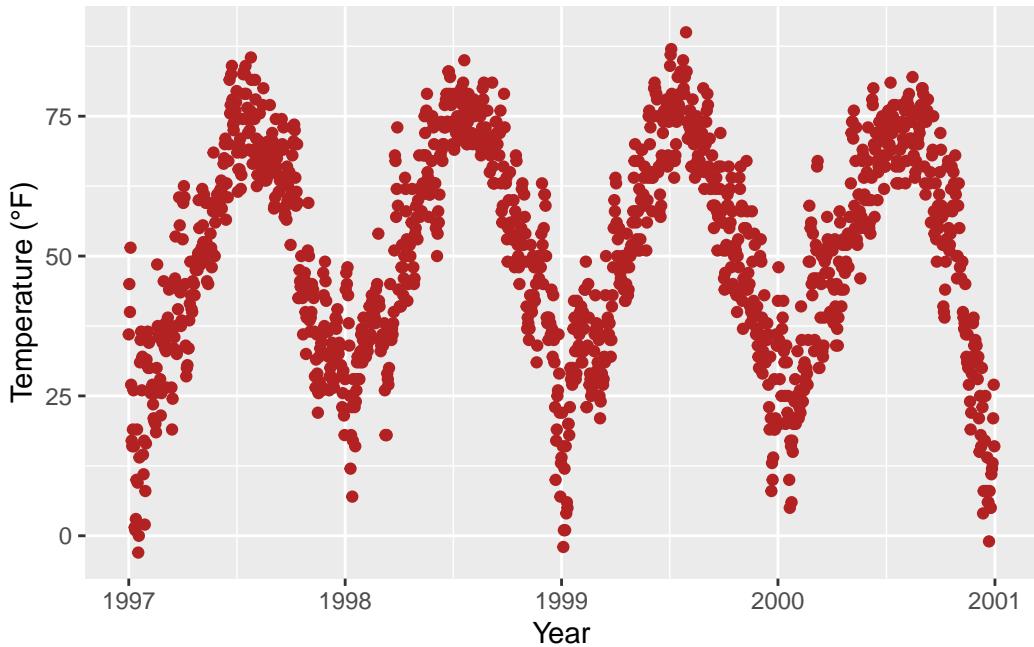
To add clear and descriptive labels to the axes, we can utilize the `labs()` function. This function allows us to provide a character string for each label we wish to modify, such as `x` and `y`:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)")
```



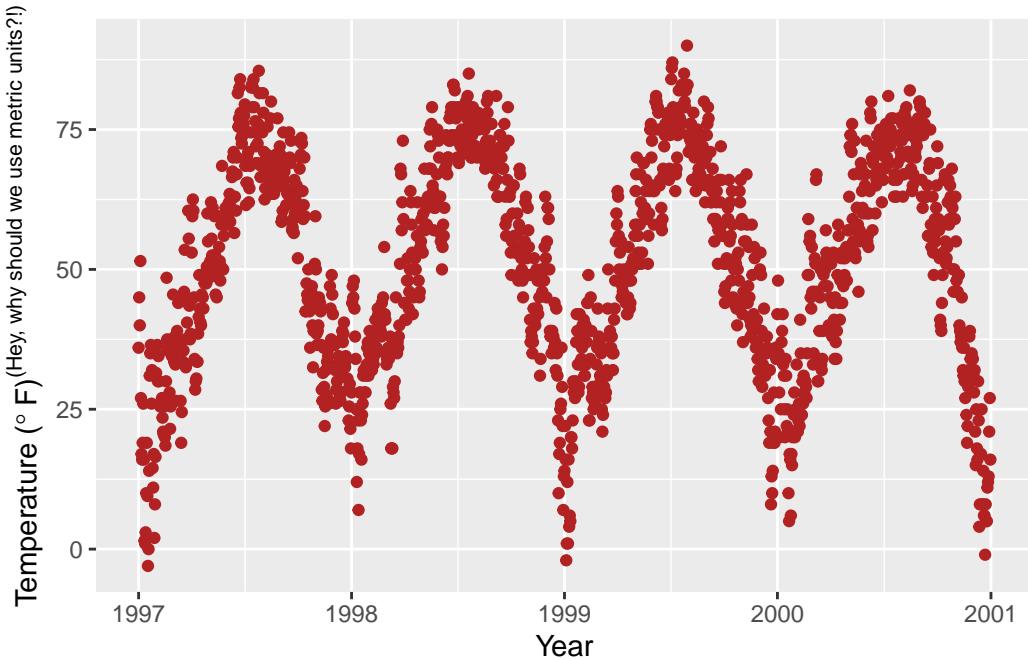
You also can add axis titles by using `xlab()` and `ylab()`. Click to see example.

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  xlab("Year") +  
  ylab("Temperature (°F)")
```



Typically, you can specify symbols by directly adding the symbol itself (e.g., “ \circ ”). However, the code below also enables the addition of not only symbols but also features like superscripts:

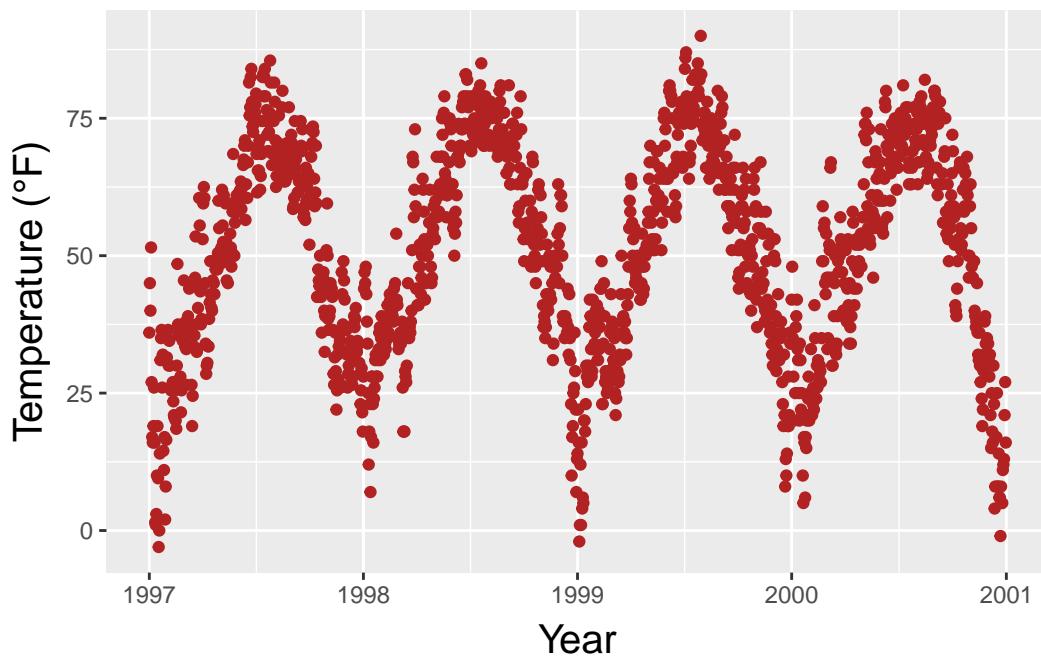
```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = expression(paste("Temperature (", degree ~ F, ")")^"(Hey, why should we")))
```



4.2 Increase Space between Axis and Axis Titles

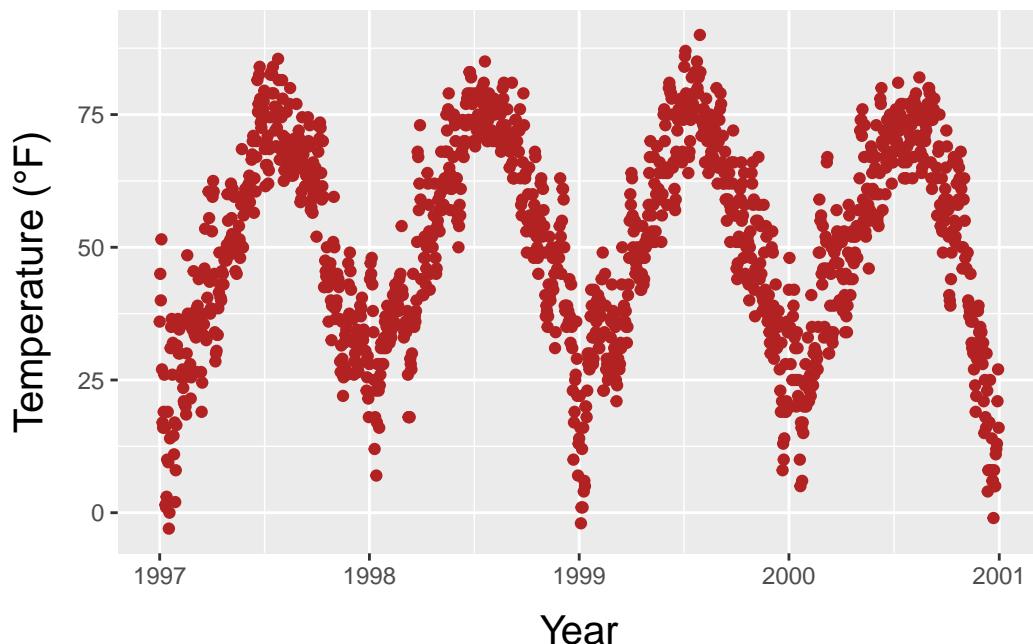
`theme()` is a crucial command for adjusting specific theme elements such as texts, titles, boxes, symbols, backgrounds, and more. We'll be utilizing this command extensively! Initially, we'll focus on modifying text elements. We can customize the properties of all text elements or specific ones, such as axis titles, by overriding the default `element_text()` within the `theme()` call:

```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(axis.title.x = element_text(vjust = 0, size = 15),
        axis.title.y = element_text(vjust = 2, size = 15))
```



The `vjust` parameter controls vertical alignment and typically ranges between 0 and 1, but you can also specify values outside that range. It's worth noting that even when adjusting the position of the axis title along the y-axis horizontally, we still need to specify `vjust` (which is correct from the perspective of the label's alignment). Additionally, you can modify the distance by specifying the margin for both text elements:

```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(axis.title.x = element_text(margin = margin(t = 10), size = 15),
        axis.title.y = element_text(margin = margin(r = 10), size = 15))
```



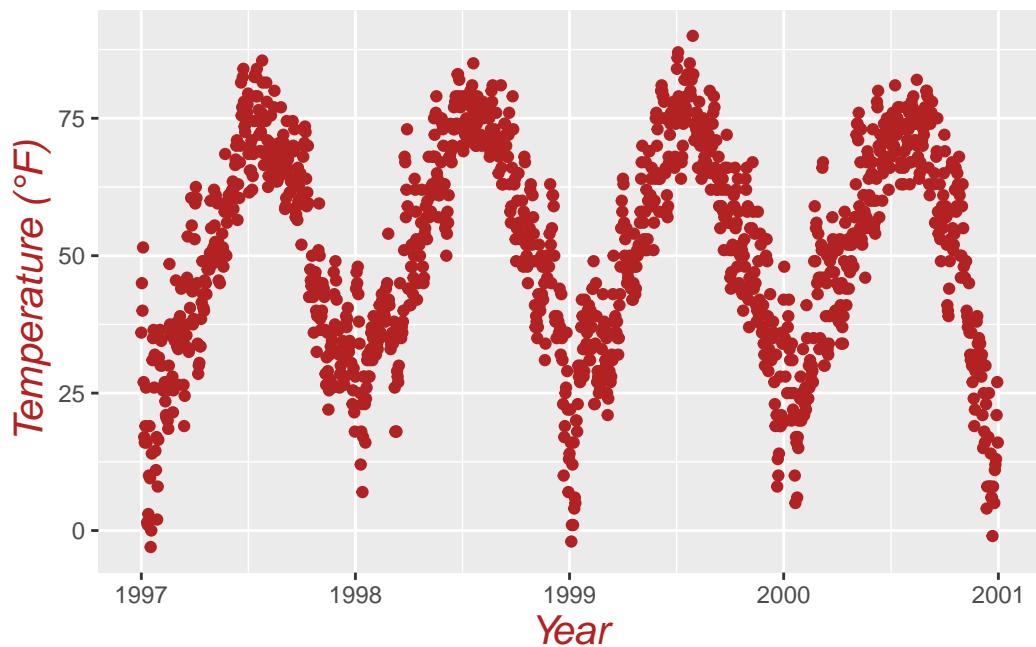
The labels `t` and `r` within the `margin()` object correspond to *top* and *right*, respectively. Alternatively, you can specify all four margins using `margin(t, r, b, l)`. It's important to note that we need to adjust the right margin to modify the space on the y-axis, not the bottom margin.

A helpful mnemonic for remembering the order of the margin sides is “*t-r-ou-bl-e*”.

4.3 Change Aesthetics of Axis Titles

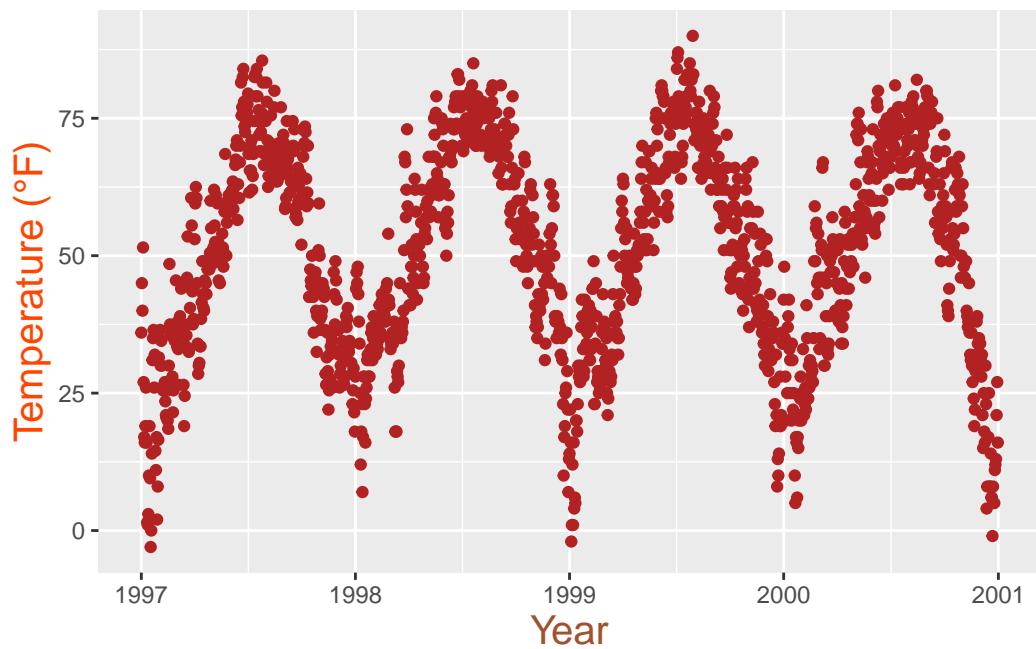
Once more, we utilize the `theme()` function to modify the `axis.title` element and/or its subordinated elements, `axis.title.x` and `axis.title.y`. Within the `element_text()` function, we can override defaults for properties such as `size`, `color`, and `face`:

```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(axis.title = element_text(size = 15, color = "firebrick",
                                    face = "italic"))
```



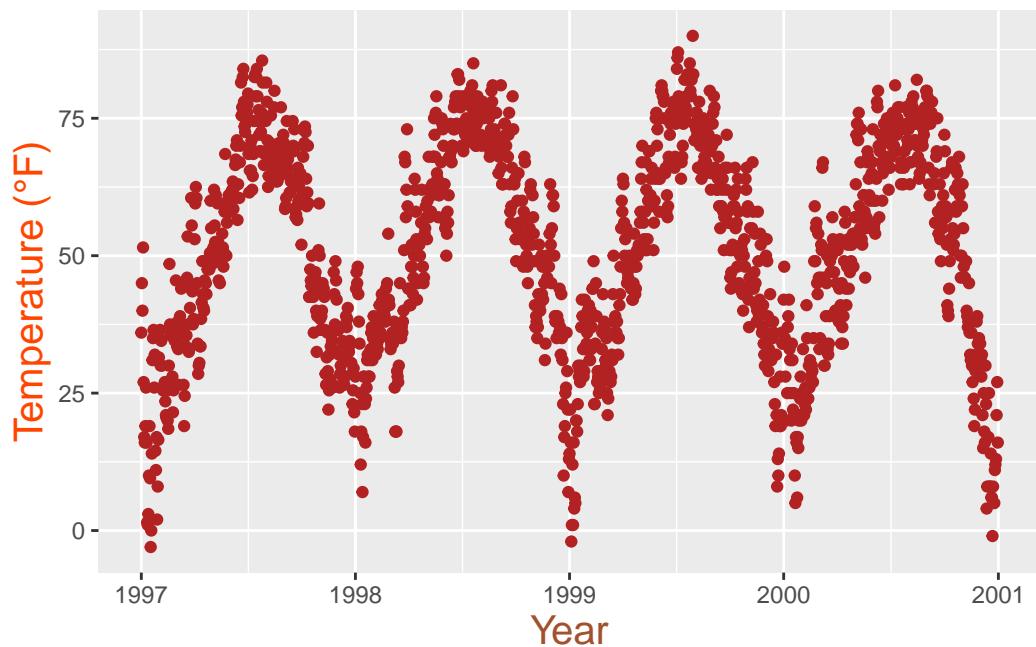
The `face` argument can be used to make the font `bold` or `italic` or even `bold.italic`.

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature ( $^{\circ}$ F)") +  
  theme(axis.title.x = element_text(color = "sienna", size = 15),  
        axis.title.y = element_text(color = "orangered", size = 15))
```



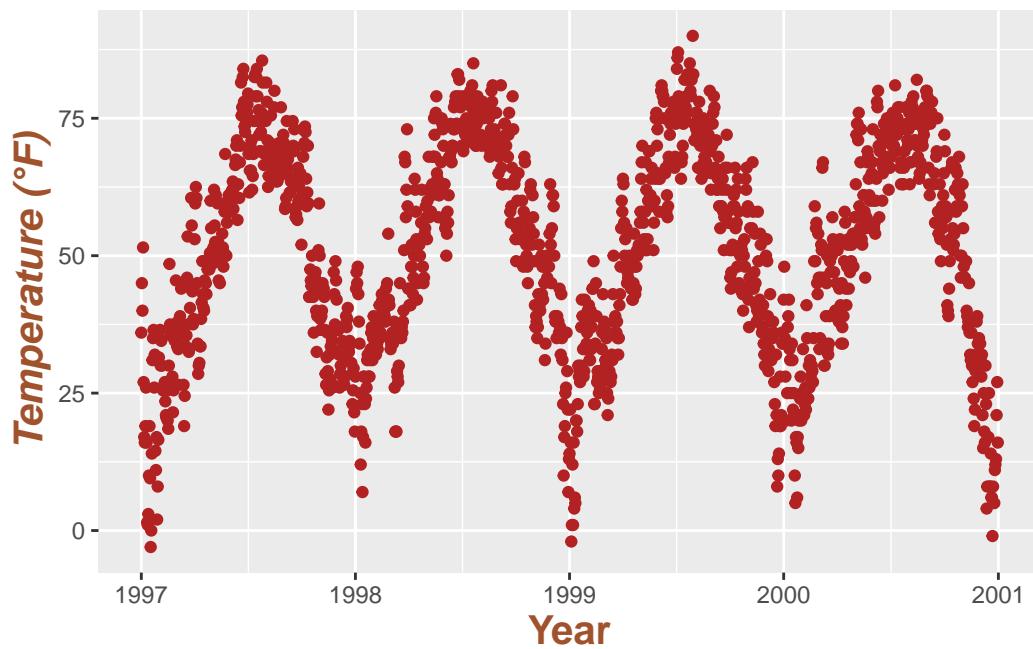
You could also employ a combination of `axis.title` and `axis.title.y`, as `axis.title.x` inherits values from `axis.title`. Expand to See the example below:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(axis.title = element_text(color = "sienna", size = 15),  
        axis.title.y = element_text(color = "orangered", size = 15))
```



You can adjust some properties for both axis titles simultaneously, while modifying others exclusively for one axis or individual properties for each axis title:

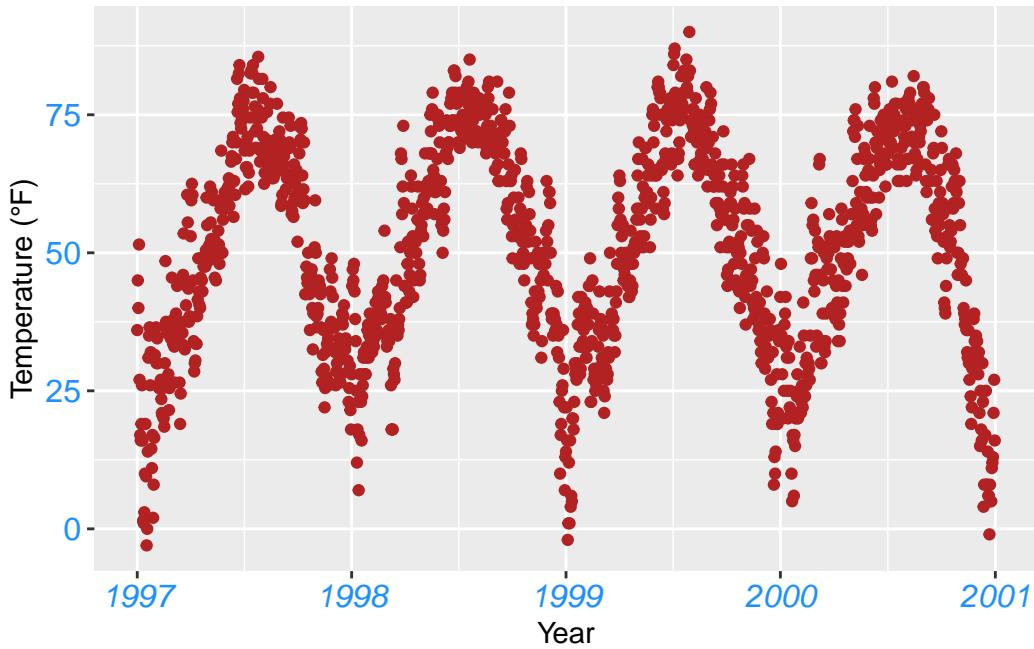
```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(axis.title = element_text(color = "sienna", size = 15, face = "bold"),  
        axis.title.y = element_text(face = "bold.italic"))
```



4.4 Change Aesthetics of Axis Text

Likewise, you can alter the appearance of the axis text (i.e., the numbers) by utilizing `axis.text` and/or its subordinated elements, `axis.text.x` and `axis.text.y`:

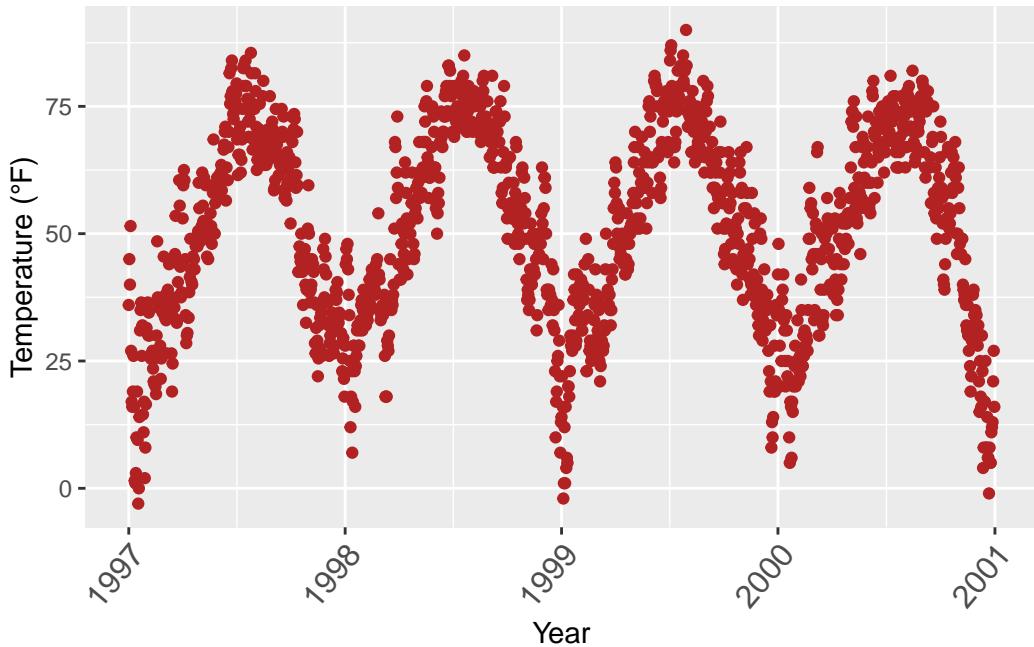
```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature ( $^{\circ}$ F)") +  
  theme(axis.text = element_text(color = "dodgerblue", size = 12),  
        axis.text.x = element_text(face = "italic"))
```



4.5 Rotate Axis Text

You can rotate any text elements by specifying an `angle`. Subsequently, you can adjust the position of the text horizontally (0 = left, 1 = right) and vertically (0 = top, 1 = bottom) using `hjust` and `vjust`:

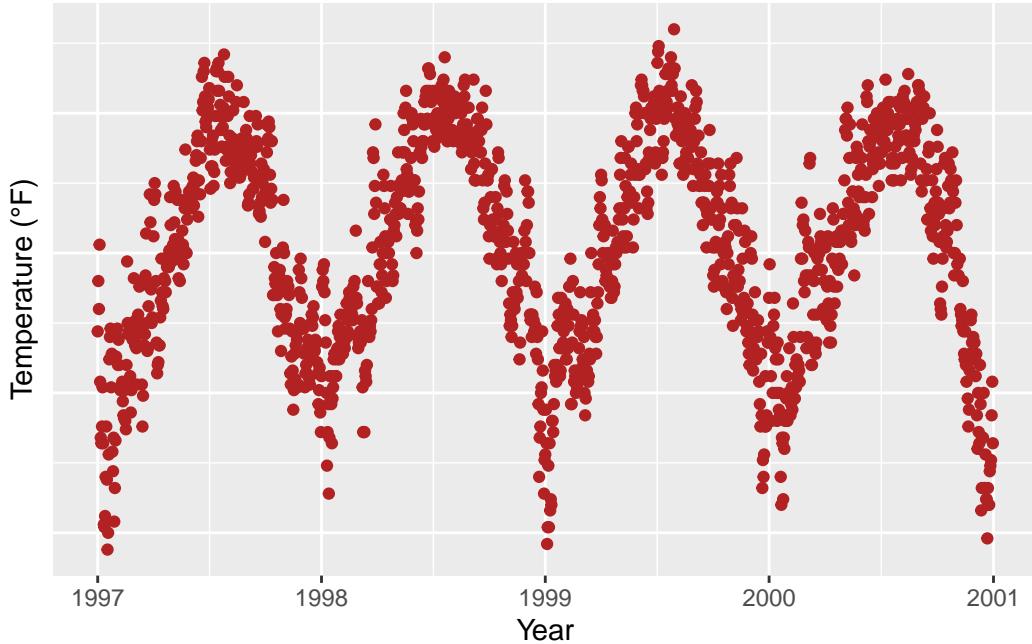
```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(axis.text.x = element_text(angle = 50, vjust = 1, hjust = 1, size = 12))
```



4.6 Removing Axis Text & Ticks

There might be rare occasions where you need to remove axis text and ticks. Here's how you can achieve it:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(axis.ticks.y = element_blank(),  
        axis.text.y = element_blank())
```



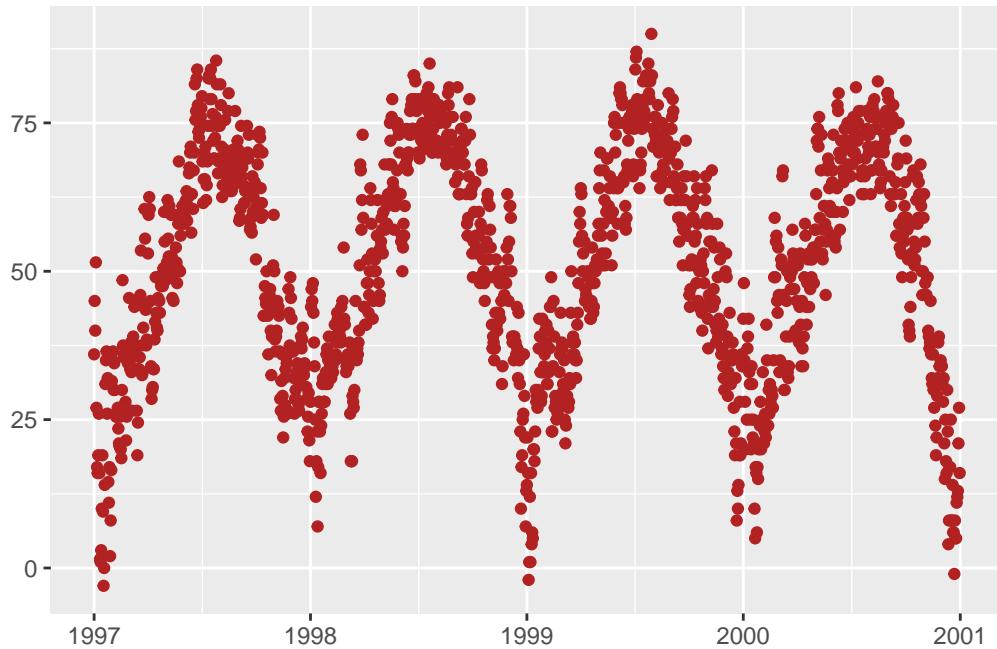
I've introduced three theme elements—text, lines, and rectangles—but there's actually one more: `element_blank()`, which removes the element entirely. However, it's not considered an official element like the others.

If you wish to remove a theme element entirely, you can always use `element_blank()`.

4.7 Removing Axis Titles

We can use `theme_blank()`, but it's much simpler to just omit the label in the `labs()` (or `xlab()`) call:

```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = NULL, y = "")
```



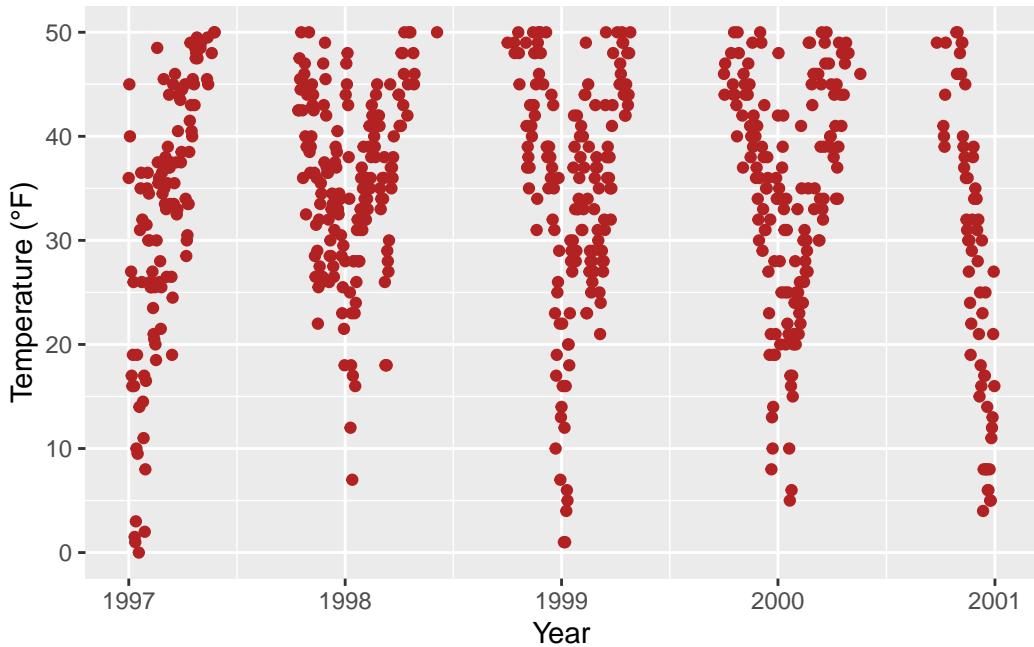
Note that `NULL` removes the element (similarly to `element_blank()`), while empty quotes `" "` will keep the spacing for the axis title but print nothing.

4.8 Limiting Axis Range

Occasionally, you may want to focus on a specific range of your data without altering the dataset itself. You can accomplish this with ease:

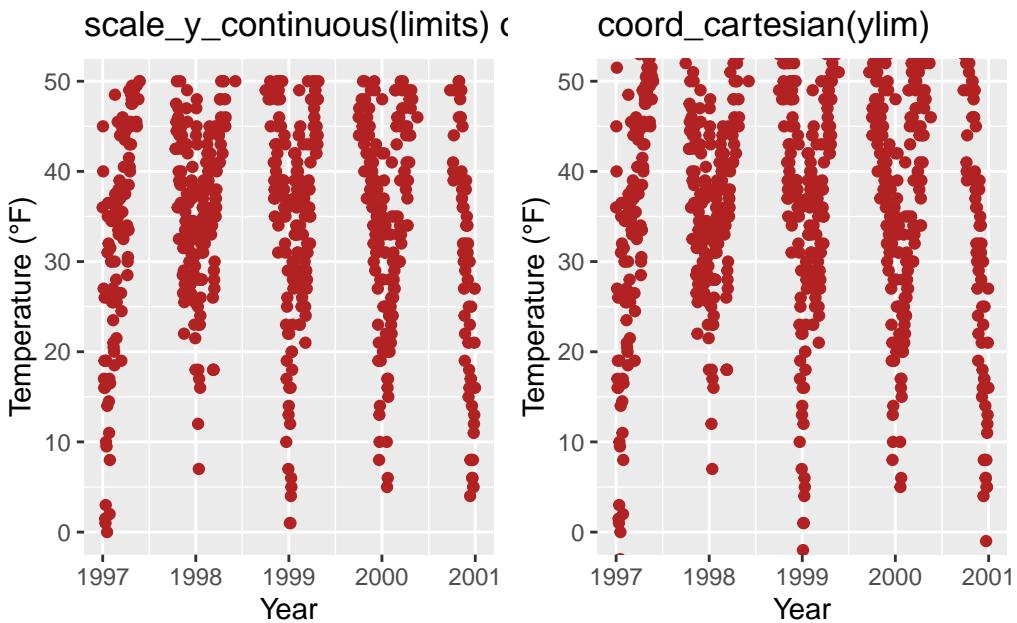
```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = "Temperature (°F)") +
  ylim(c(0, 50))
```

Warning: Removed 777 rows containing missing values or values outside the scale range (`geom_point()`).



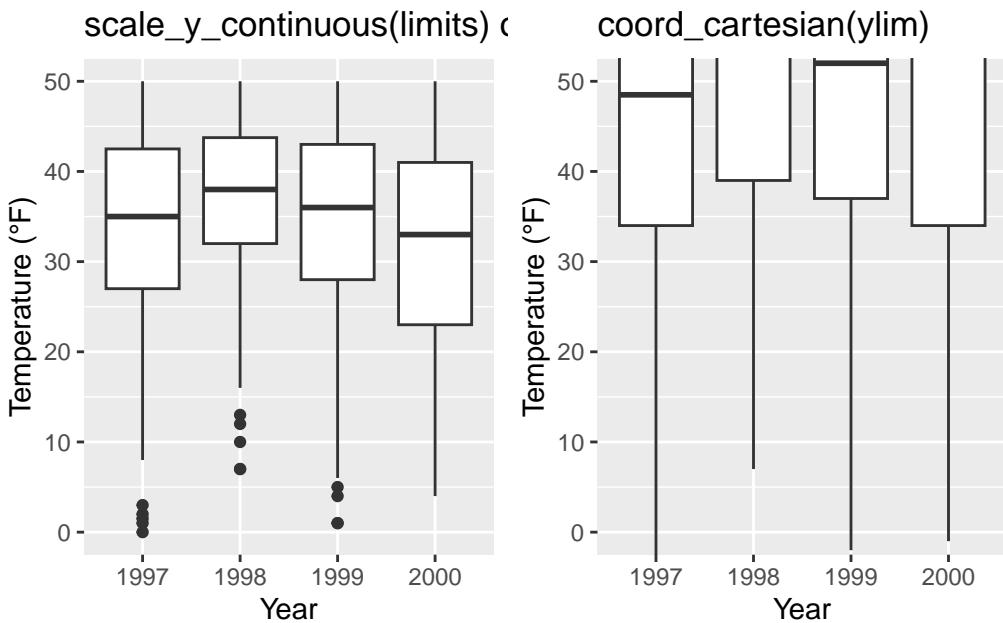
Alternatively, you can utilize `scale_y_continuous(limits = c(0, 50))` or `coord_cartesian(ylim = c(0, 50))`. The former removes all data points outside the specified range, while the latter adjusts the visible area, similar to `ylim(c(0, 50))`. At first glance, it may seem that both approaches yield the same result. However, there is an important distinction—compare the following two plots:

```
Warning: Removed 777 rows containing missing values or values outside the scale range
(`geom_point()`).
```



You may have noticed that on the left, there is some empty buffer around your y limits, while on the right, points are plotted right up to the border and even beyond. This effectively illustrates the concept of subsetting (left) versus zooming (right). To demonstrate why this distinction is significant, let's examine a different chart type: a box plot.

```
Warning: Removed 777 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```



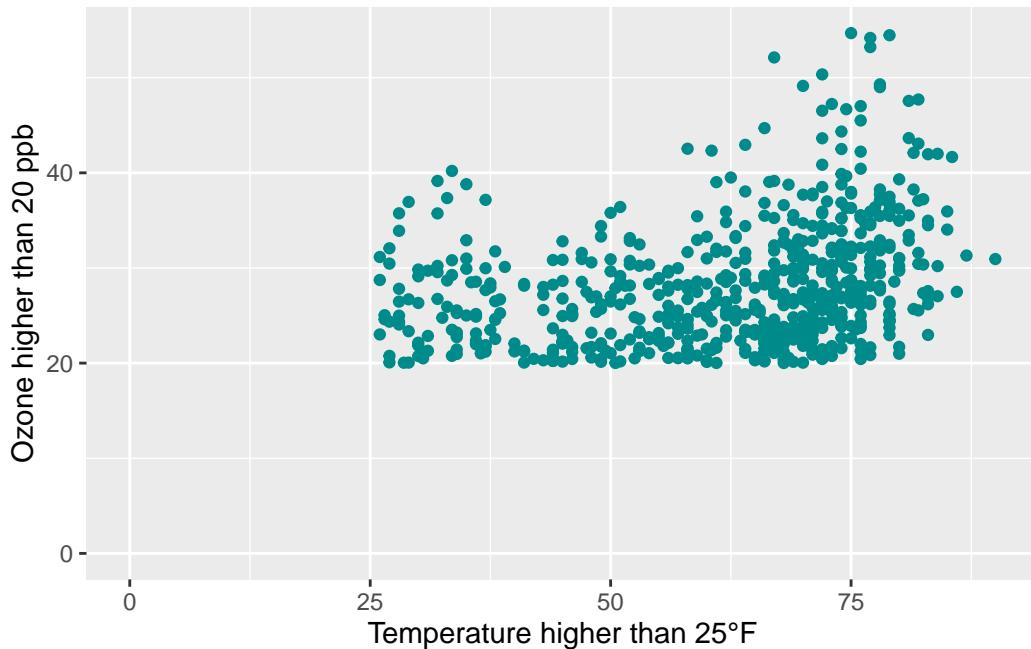
Indeed, because `scale_x|y_continuous()` subsets the data first, we obtain completely different (and potentially incorrect, especially if this was not our intention) estimates for the box plots! This realization highlights the importance of ensuring data integrity throughout the plotting process. It's crucial to avoid inadvertently manipulating the data while plotting, as it could lead to inaccurate summary statistics reported in your report, paper, or thesis.

4.9 Forcing Plot to Start at Origin

Related to that, you can instruct R to plot the graph starting at the origin:

```
chic_high <- dplyr::filter(chic, temp > 25, o3 > 20)

ggplot(chic_high, aes(x = temp, y = o3)) +
  geom_point(color = "darkcyan") +
  labs(x = "Temperature higher than 25°F",
       y = "Ozone higher than 20 ppb") +
  expand_limits(x = 0, y = 0)
```



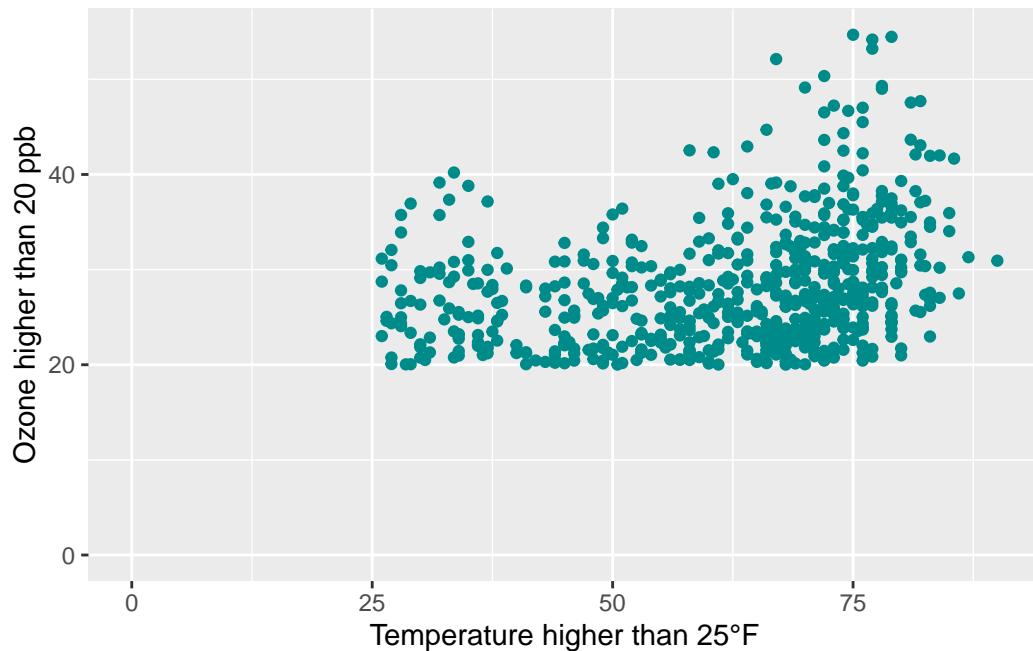
Using `coord_cartesian(xlim = c(0, NA), ylim = c(0, NA))` will produce the same result. CLICK to See the example below:

```

chic_high <- dplyr::filter(chic, temp > 25, o3 > 20)

ggplot(chic_high, aes(x = temp, y = o3)) +
  geom_point(color = "darkcyan") +
  labs(x = "Temperature higher than 25°F",
       y = "Ozone higher than 20 ppb") +
  coord_cartesian(xlim = c(0, NA), ylim = c(0, NA))

```

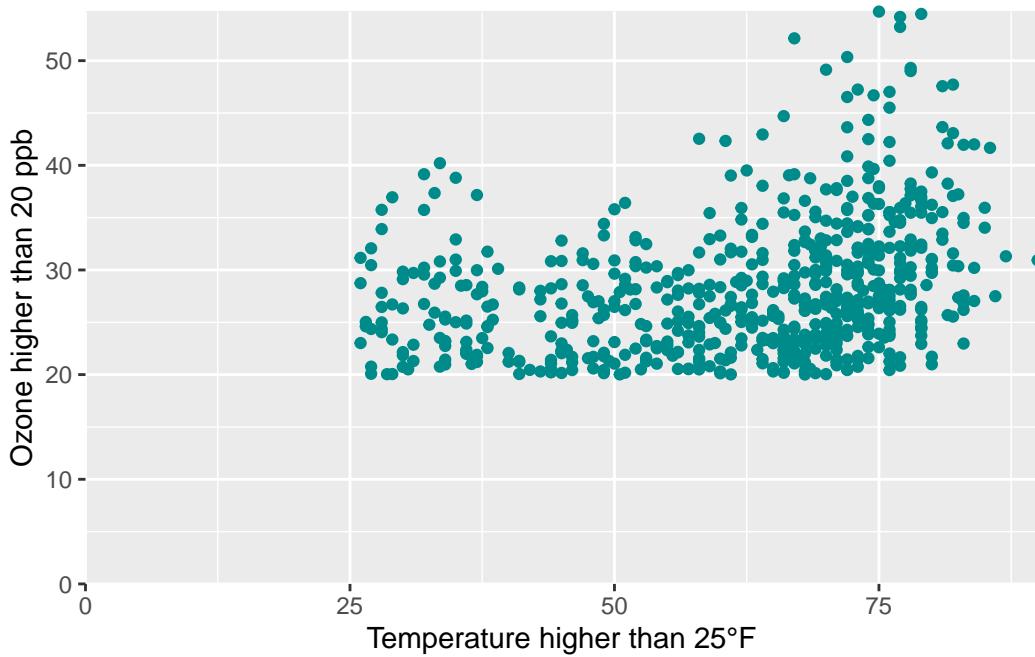


But we can also ensure that it *truly* starts at the origin!

```

ggplot(chic_high, aes(x = temp, y = o3)) +
  geom_point(color = "darkcyan") +
  labs(x = "Temperature higher than 25°F",
       y = "Ozone higher than 20 ppb") +
  expand_limits(x = 0, y = 0) +
  coord_cartesian(expand = FALSE, clip = "off")

```



The `clip = "off"` argument in any coordinate system, always starting with `coord_*`, enables drawing outside of the panel area.

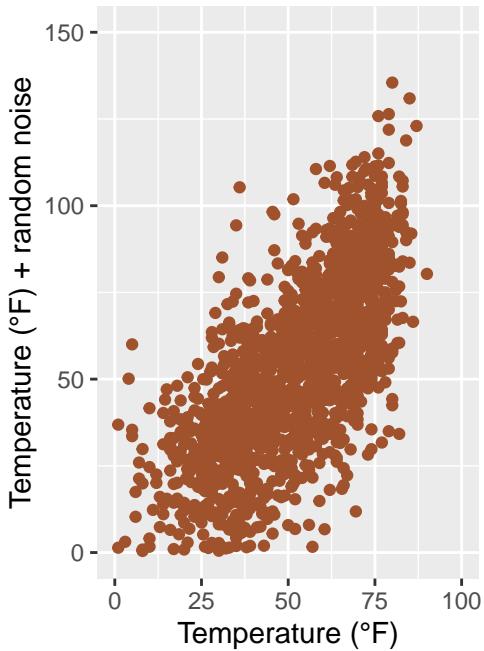
Here, I invoke it to ensure that the tick marks at `c(0, 0)` remain intact and are not truncated. For further insights, refer to the [Twitter thread by Claus Wilke](#).

4.10 Axes with Same Scaling

For demonstration purposes, let's plot temperature against temperature with some random noise. The `coord_equal()` function provides a coordinate system with a specified ratio, representing the number of units on the y-axis equivalent to one unit on the x-axis. By default, `ratio = 1` ensures that one unit on the x-axis is the same length as one unit on the y-axis:

```
ggplot(chic, aes(x = temp, y = temp + rnorm(nrow(chic), sd = 20))) +
  geom_point(color = "sienna") +
  labs(x = "Temperature (°F)", y = "Temperature (°F) + random noise") +
  xlim(c(0, 100)) + ylim(c(0, 150)) +
  coord_fixed()
```

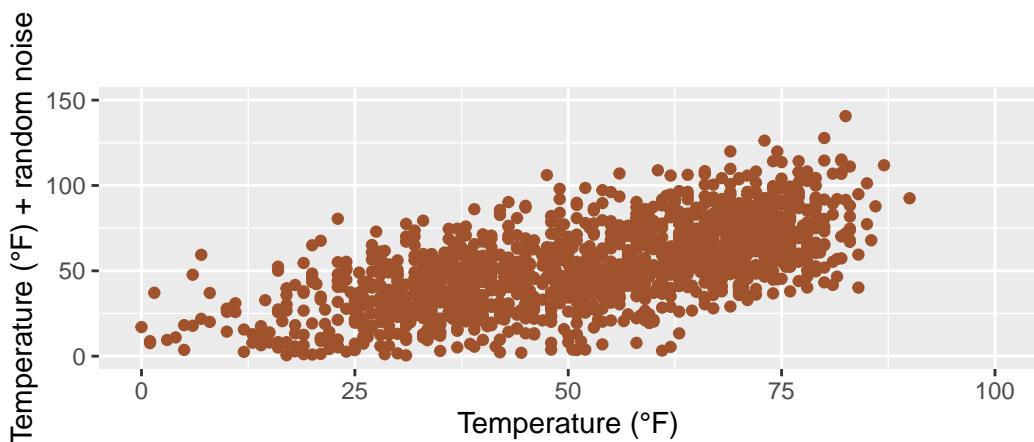
Warning: Removed 53 rows containing missing values or values outside the scale range (`geom_point()`).



Ratios higher than one result in units on the y-axis being longer than units on the x-axis, while ratios lower than one have the opposite effect:

```
ggplot(chic, aes(x = temp, y = temp + rnorm(nrow(chic), sd = 20))) +
  geom_point(color = "sienna") +
  labs(x = "Temperature (°F)", y = "Temperature (°F) + random noise") +
  xlim(c(0, 100)) + ylim(c(0, 150)) +
  coord_fixed(ratio = 1/5)
```

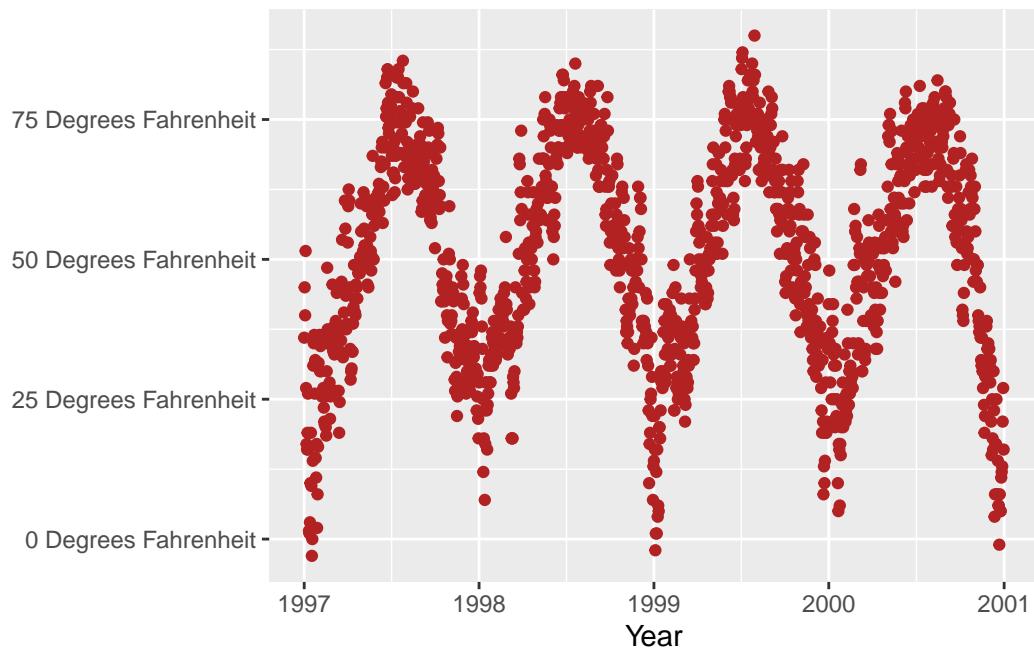
Warning: Removed 49 rows containing missing values or values outside the scale range (`geom_point()`).



4.11 Using a Function to Alter Labels

Occasionally, it's useful to slightly modify your labels, such as adding units or percent signs, without altering your underlying data. You can achieve this using a function:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = NULL) +  
  scale_y_continuous(label = function(x) {return(paste(x, "Degrees Fahrenheit"))})
```

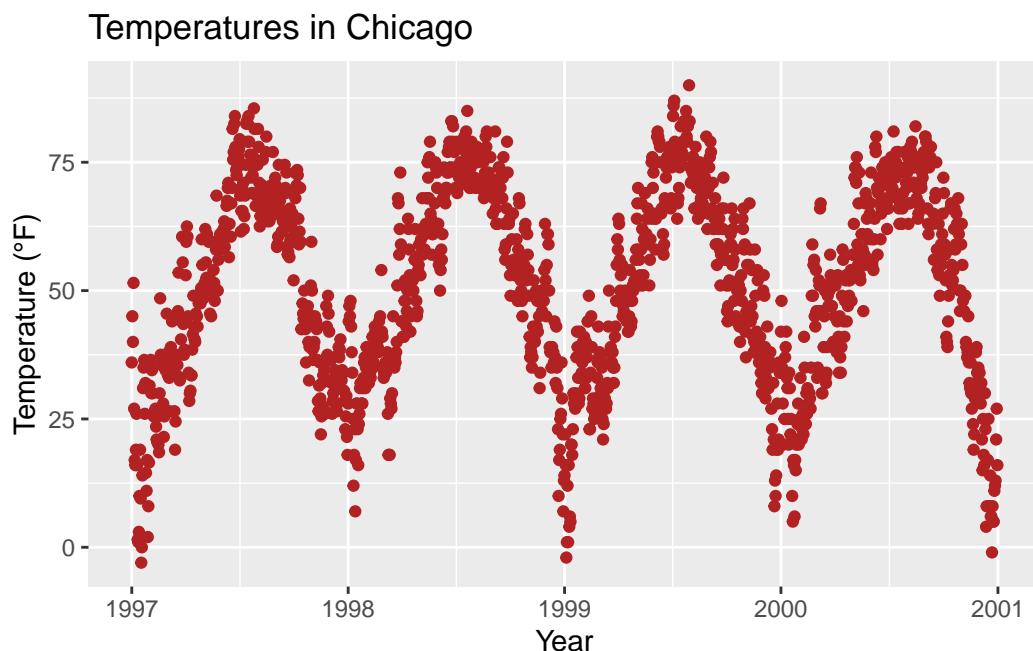


5 Working with Titles

5.1 Add a Title

We can add a title by using the `ggttitle()` function:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  ggttitle("Temperatures in Chicago")
```



Alternatively, you can utilize `labs()`. Here, you can include multiple arguments, such as a subtitle, a caption, and a tag, in addition to axis titles as demonstrated earlier:

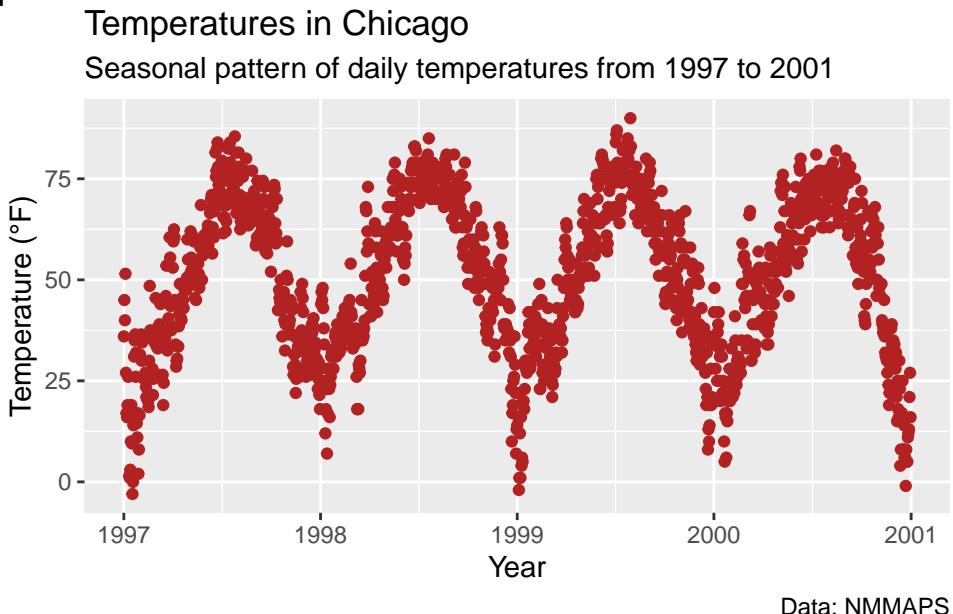
```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)",
```

```

title = "Temperatures in Chicago",
subtitle = "Seasonal pattern of daily temperatures from 1997 to 2001",
caption = "Data: NMMAPS",
tag = "Fig. 1")

```

Fig. 1



5.2 Making Title Bold & Adding a Space at the Baseline

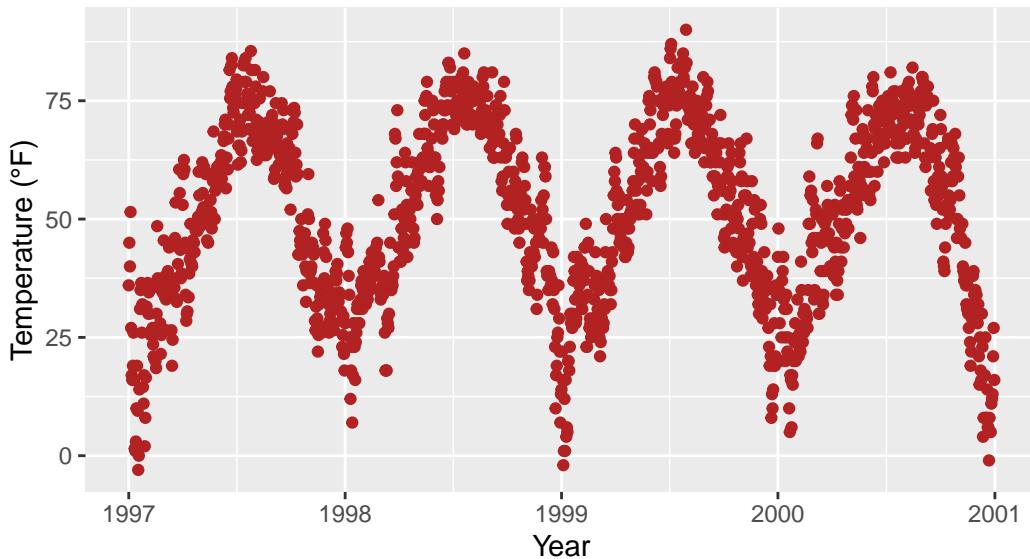
Once again, to adjust the properties of a theme element, we employ the `theme()` function. Similar to modifying text elements like `axis.title` and `axis.text`, we can alter the font face and margin for the title. These modifications apply not only to the title but also to other labels such as `plot.subtitle`, `plot.caption`, `plot.tag`, `legend.title`, `legend.text`, `axis.title`, and `axis.text`.

```

ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = "Temperature (°F)",
       title = "Temperatures in Chicago") +
  theme(plot.title = element_text(face = "bold",
                                   margin = margin(10, 0, 10, 0),
                                   size = 14))

```

Temperatures in Chicago



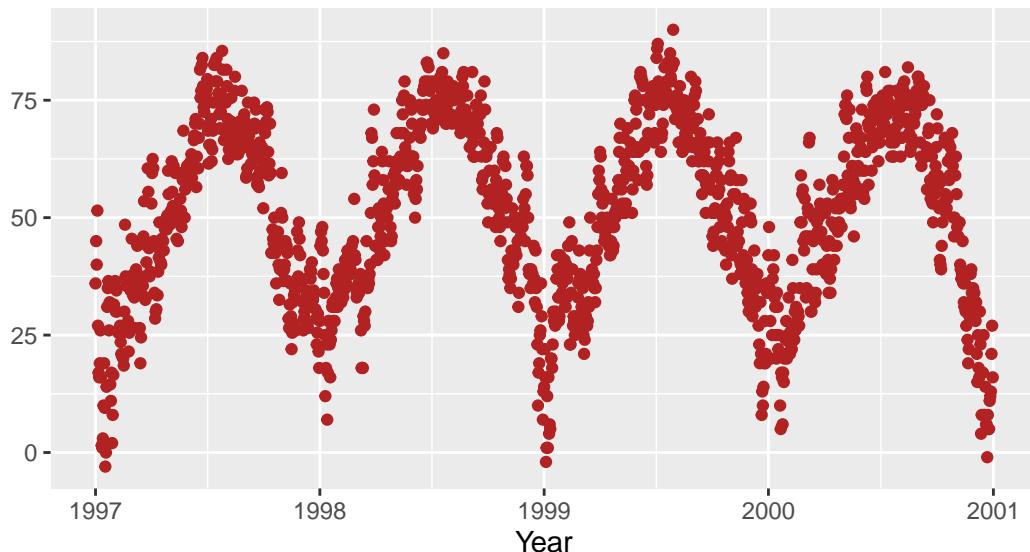
A helpful mnemonic for remembering the order of the margin arguments is “*t-r-oub-l-e*”, which corresponds to the first letter of each side.

5.3 Adjusting Position of Titles

The general alignment (left, center, right) is controlled by `hjust` (horizontal adjustment):

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = NULL,  
       title = "Temperatures in Chicago",  
       caption = "Data: NMMAPS") +  
  theme(plot.title = element_text(hjust = 1, size = 16, face = "bold.italic"))
```

Temperatures in Chicago

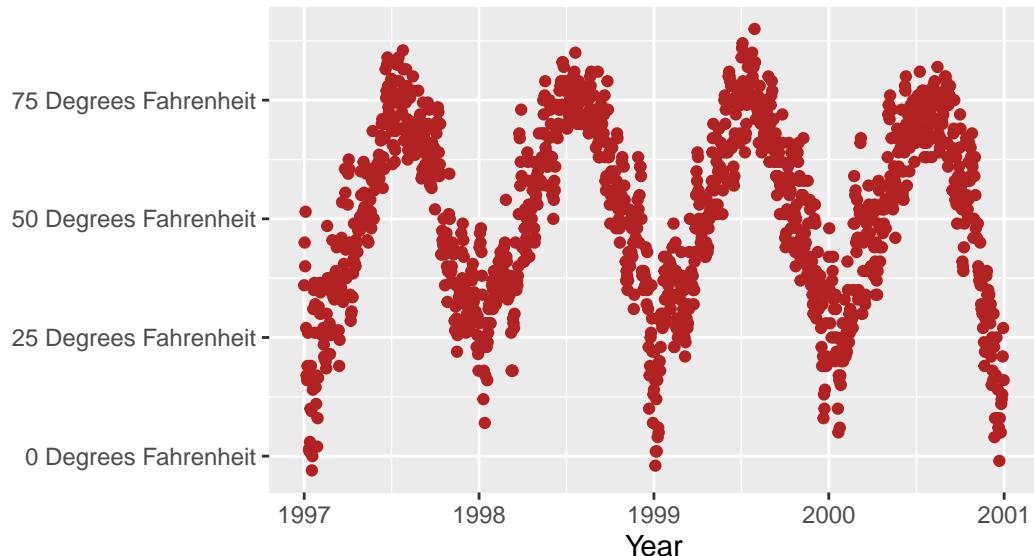


Data: NMMAPS

Certainly, it's also possible to adjust the vertical alignment, which is controlled by `vjust`. Since 2019, users have been able to specify the alignment of the title, subtitle, and caption either based on the panel area (the default) or the plot margin via `plot.title.position` and `plot.caption.position`. The latter is often the preferred choice from a design perspective, as it yields better results in most cases. Many users have expressed satisfaction with this new feature, particularly as it addresses issues with alignment, especially when dealing with very long y-axis labels:

```
(g <- ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  scale_y_continuous(label = function(x) {return(paste(x, "Degrees Fahrenheit"))}) +  
  labs(x = "Year", y = NULL,  
       title = "Temperatures in Chicago between 1997 and 2001 in Degrees Fahrenheit",  
       caption = "Data: NMMAPS") +  
  theme(plot.title = element_text(size = 14, face = "bold.italic"),  
        plot.caption = element_text(hjust = 0)))
```

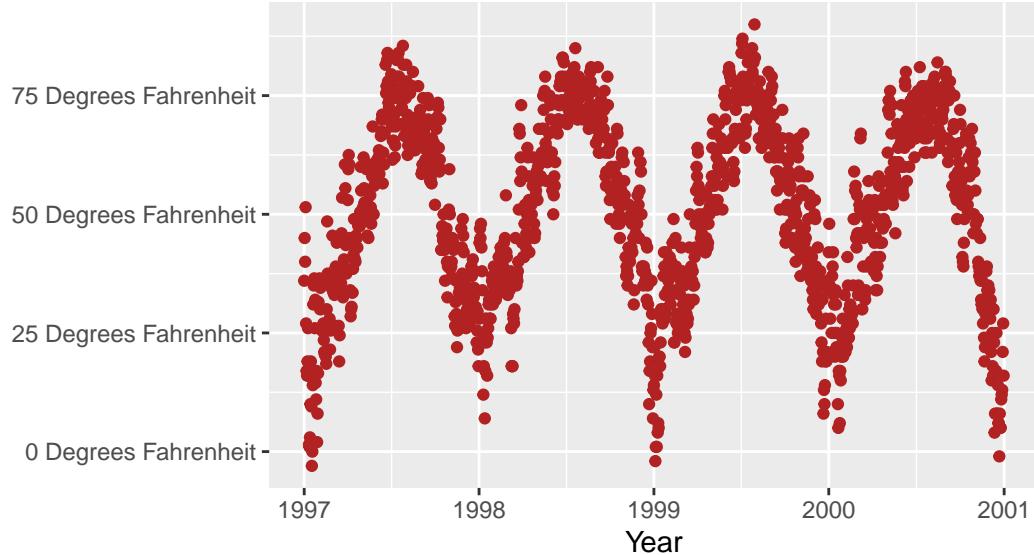
Temperatures in Chicago between 1997 and



Data: NMMAPS

```
g + theme(plot.title.position = "plot",
          plot.caption.position = "plot")
```

Temperatures in Chicago between 1997 and 2001 in Degrees



Data: NMMAPS

5.4 Using a Non-Traditional Font in Your Title

You can incorporate different fonts, not just the default one provided by ggplot (which can vary between operating systems). Several packages facilitate the usage of fonts installed on your machine, such as the [showtext package](#), which simplifies the utilization of various font types (TrueType, OpenType, Type 1, web fonts, etc.) in R plots.

Once the package is loaded, you'll need to import the desired font, which must also be installed on your device. I often utilize [Google fonts](#), which can be imported using the `font_add_google()` function. However, you can add other fonts using `font_add()` as well. It's important to note that even when using Google fonts, you must install the font and restart RStudio to apply it. If you found any warnings after doing all the steps, or the fonts aren't working. Just install `extrafont` package and run `font_import()` function to import all the fonts in your system. and then `loadfonts(device = "win", quiet = TRUE)` to load the fonts. It'll work like a charm. You can also check the available fonts in your system by running `fonts()`.

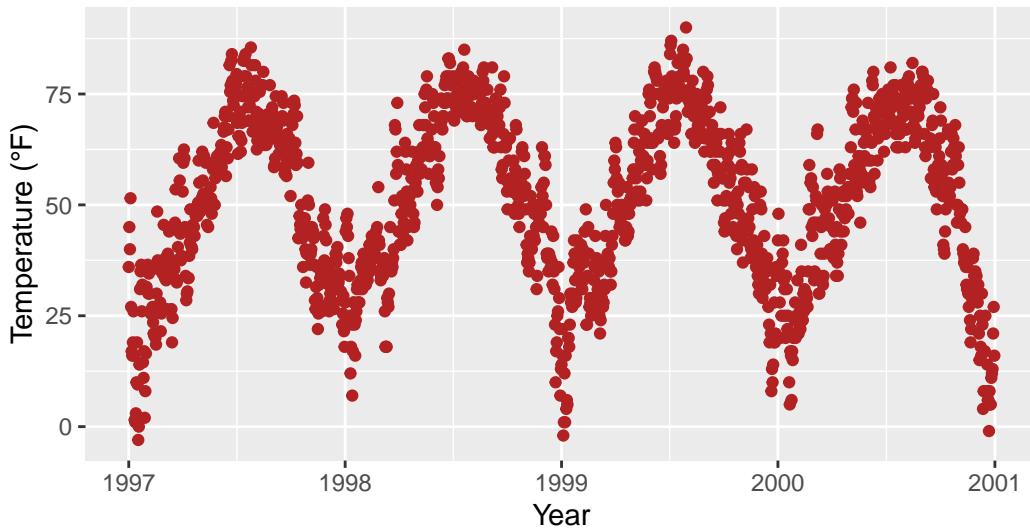
```
library(showtext)
library(extrafont)
font_add_google("Playfair Display", ## name of Google font
                "Playfair Display") ## name that will be used in R
font_add_google("Bangers", "Bangers")
loadfonts(device = "win", quiet = TRUE)
```

Now, we can use those font families by `theme()` function:

```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = "Temperature (°F)",
       title = "Temperatures in Chicago",
       subtitle = "Daily temperatures in °F from 1997 to 2001") +
  theme(plot.title = element_text(family = "Bangers", hjust = .5, size = 25),
        plot.subtitle = element_text(family = "Playfair Display", hjust = .5, size = 15))
```

Temperatures in Chicago

Daily temperatures in °F from 1997 to 2001



You can also establish a non-default font for all text elements of your plots. For more details, refer to the section “Working with Themes”. In this case, I’ll use *Roboto Condensed* as the new font for all subsequent plots.

```
font_add_google("Roboto Condensed", "Roboto Condensed")
theme_set(theme_bw(base_size = 12, base_family = "Roboto Condensed"))
```

(Previously, this tutorial utilized the [{extrafont} package](#), which performed admirably until last year. However, suddenly I encountered issues where I couldn’t add any new fonts, and even after acquiring a new laptop, the package failed to detect any fonts altogether. As an alternative, I typically recommend the [{fragg} package](#) now. However, I encountered difficulties in making it work for my homepage. Therefore, I’m utilizing the [{showtext}](#) package, which is also excellent, albeit with a key distinction: you need to explicitly import the font you wish to use with [{showtext}](#). Nonetheless, it appears that there are some technical challenges that are not optimally resolved by [{showtext}](#) (as mentioned in [this Twitter thread](#)), so you may want to consider using the package only as a last resort.)

5.5 Adjusting Spacing in Multi-Line Text

To modify the spacing between lines, you can utilize the `lineheight` argument. In the following example, I’ve compressed the lines together (`lineheight < 1`).

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  ggtitle("Temperatures in Chicago\nfrom 1997 to 2001") +  
  theme(plot.title = element_text(lineheight = .8, size = 16))
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics  
unknown for character 0x4d
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width  
unknown for character 0x67
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width  
unknown for character 0x6a
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width  
unknown for character 0x70
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width  
unknown for character 0x71
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width  
unknown for character 0x79
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width  
unknown for character 0x51
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font  
width unknown for character 0x31
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font  
width unknown for character 0x39
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font  
width unknown for character 0x39
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font  
width unknown for character 0x37
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x38

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics
unknown for character 0x4d

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x6a
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x70

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x51

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics
unknown for character 0x4d

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x6a

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x70

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x51

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x54

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0xab
```

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xcd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xf4

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xfd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x7f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x54

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x70

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x74

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x75

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x73

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x69

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6e

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x43

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x68

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x69

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x63

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x67

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x66

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x74

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x59

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x38

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x59
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x54

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xab

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xcd

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xf4

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xfd

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x7f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x54

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6d
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x70

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x74

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x75

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x73

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x69

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6e

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x43

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x68

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x69

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x63

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x67

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x66

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x74

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

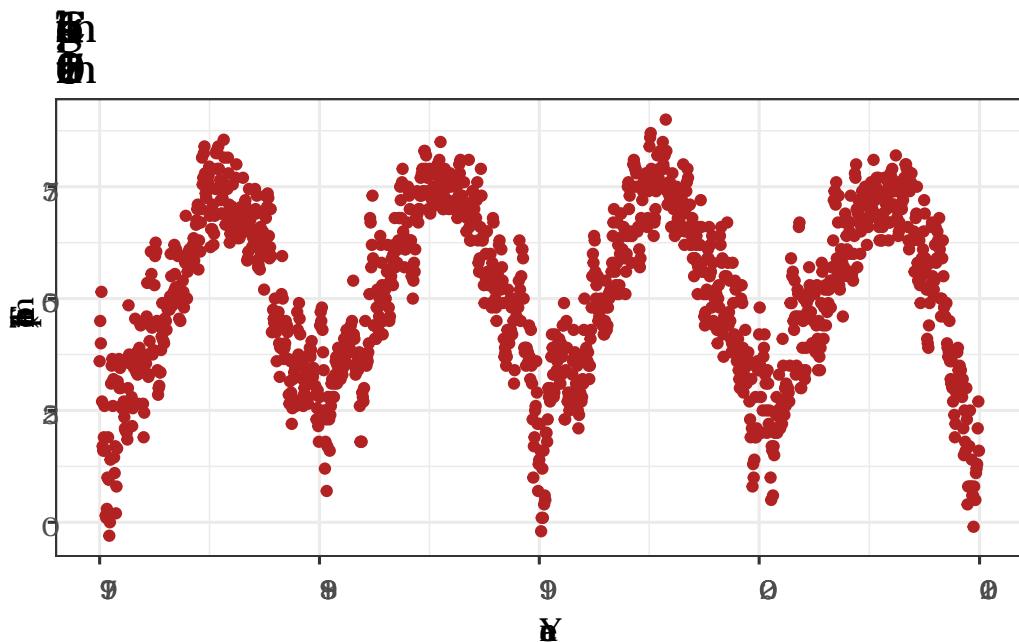
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

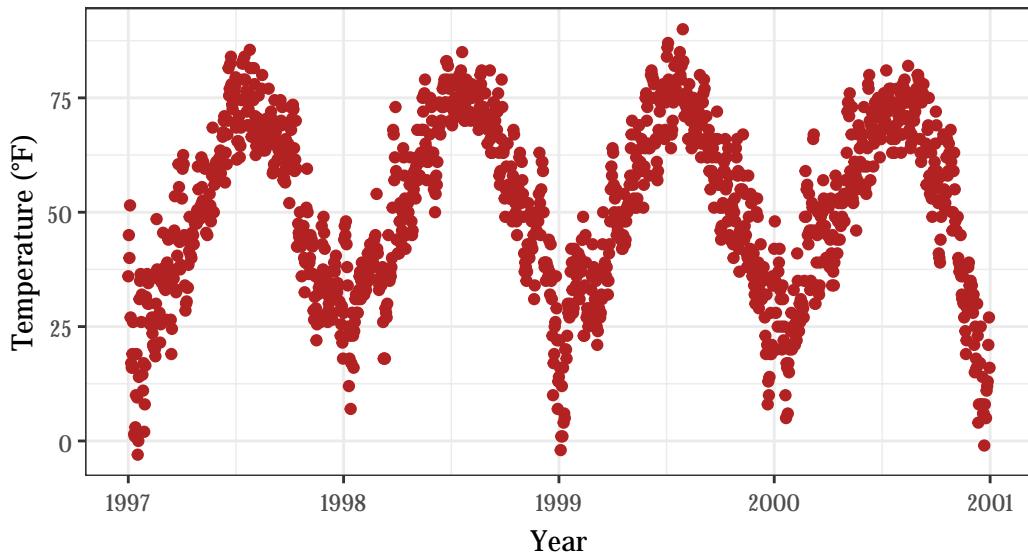
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31
```



Now You can Change fonts on the fly!

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  ggtitle("Temperatures in Chicago\nfrom 1997 to 2001") +  
  theme_bw(base_family = "Berkshire Swash")
```

Temperatures in Chicago from 1997 to 2001



Or, Change it to Traditional Times New Roman:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  ggttitle("Temperatures in Chicago\\nfrom 1997 to 2001") +  
  theme_bw(base_family = "Times New Roman")
```

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font metrics unknown for character 0x4d

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x6a

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x70

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x71

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x51

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x38

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39
```

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics
unknown for character 0x4d

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x6a

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x70

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x51

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics
unknown for character 0x4d

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x6a

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x70
```

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x51

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x54

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xab

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xcd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xf4

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xfd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x7f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x54

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x70

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x74

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x75

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x73

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x69

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6e

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x43

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x68

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x69

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x63

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x67

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x66

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x74

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x6f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x59
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x38

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
```

```
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x59

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x54

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xab

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xcd
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xf4

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xfd

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x7f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x54

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x70

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x74

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x75

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x73

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x69

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6e

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x43

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x68

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x69

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x63

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x67

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x66

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x74

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f
```

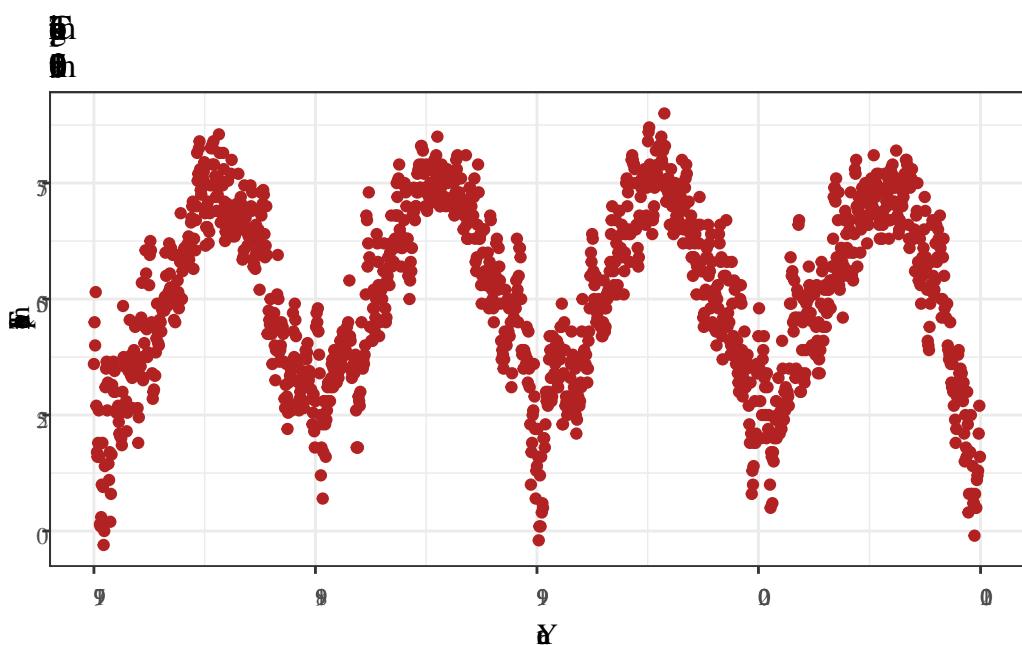
```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :  
font width unknown for character 0x20
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :  
font width unknown for character 0x32
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :  
font width unknown for character 0x30
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :  
font width unknown for character 0x30
```

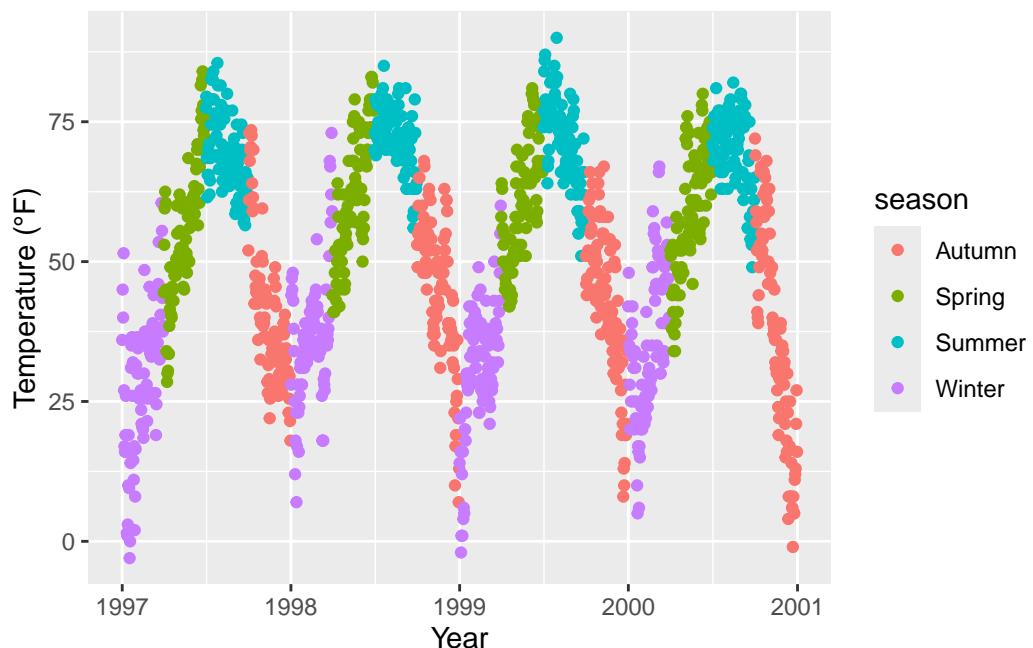
```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :  
font width unknown for character 0x31
```



6 Working with Legends

In this section, we will color code the plot based on the season. Or, to phrase it more in the style of ggplot: we'll map the variable `season` to the aesthetic `color`. One of the advantages of `{ggplot2}` is that it automatically adds a legend when mapping a variable to an aesthetic. As a result, the legend title defaults to what we specified in the `color` argument:

```
ggplot(chic,
       aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)")
```

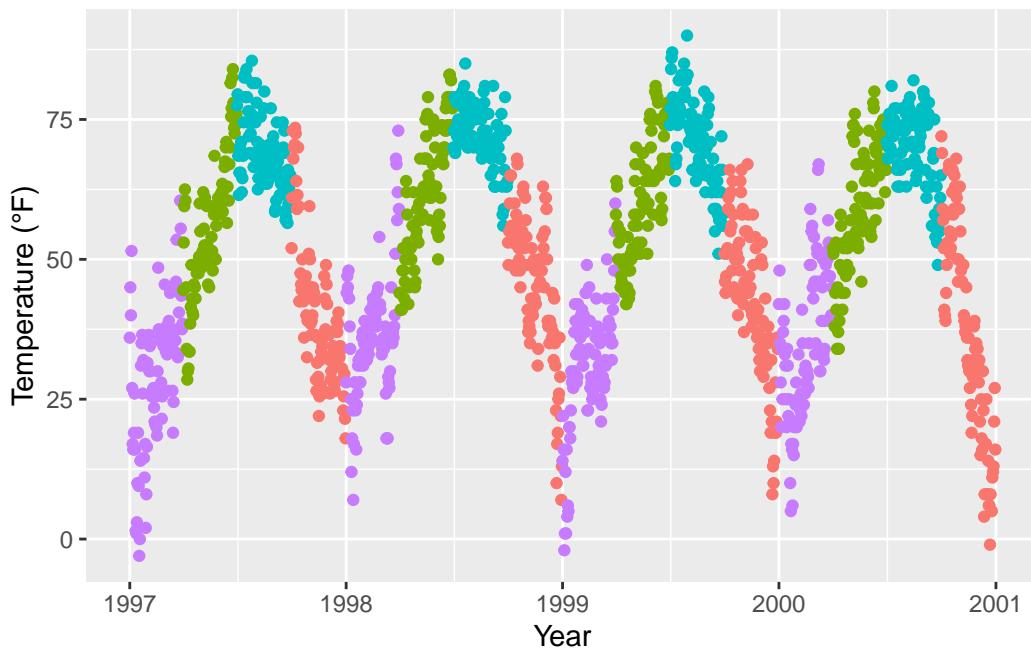


6.1 Disabling the Legend

One of the most common questions is: “How do I remove the legend?”

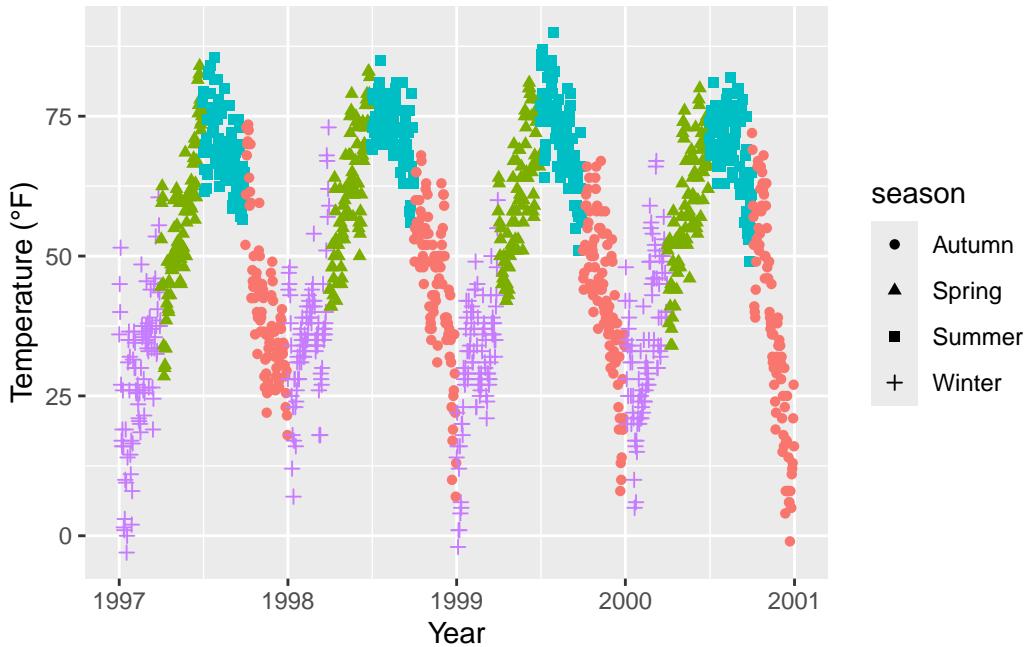
It's quite straightforward and always effective with `theme(legend.position = "none")`:

```
ggplot(chic,
       aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.position = "none")
```



You can also utilize `guides(color = "none")` or `scale_color_discrete(guide = "none")`, depending on the specific case. While altering the theme element removes all legends at once, you can selectively remove specific legends using the latter options while keeping others:

```
ggplot(chic,
       aes(x = date, y = temp,
           color = season, shape = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  guides(color = "none")
```

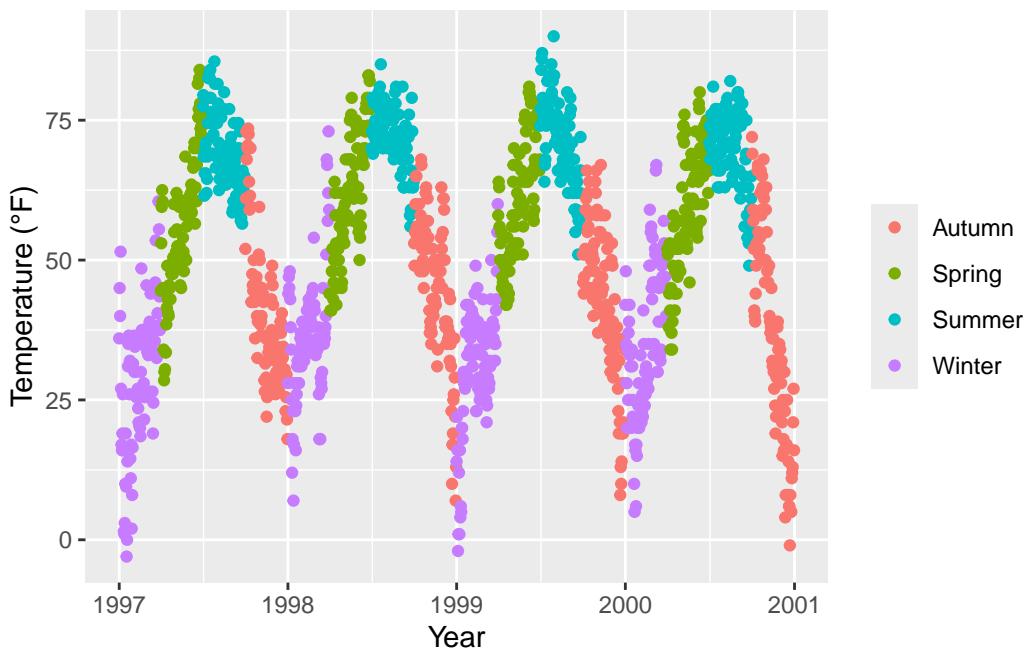


Here, for example, we retain the legend for the shapes while discarding the one for the colors.

6.2 Eliminating Legend Titles

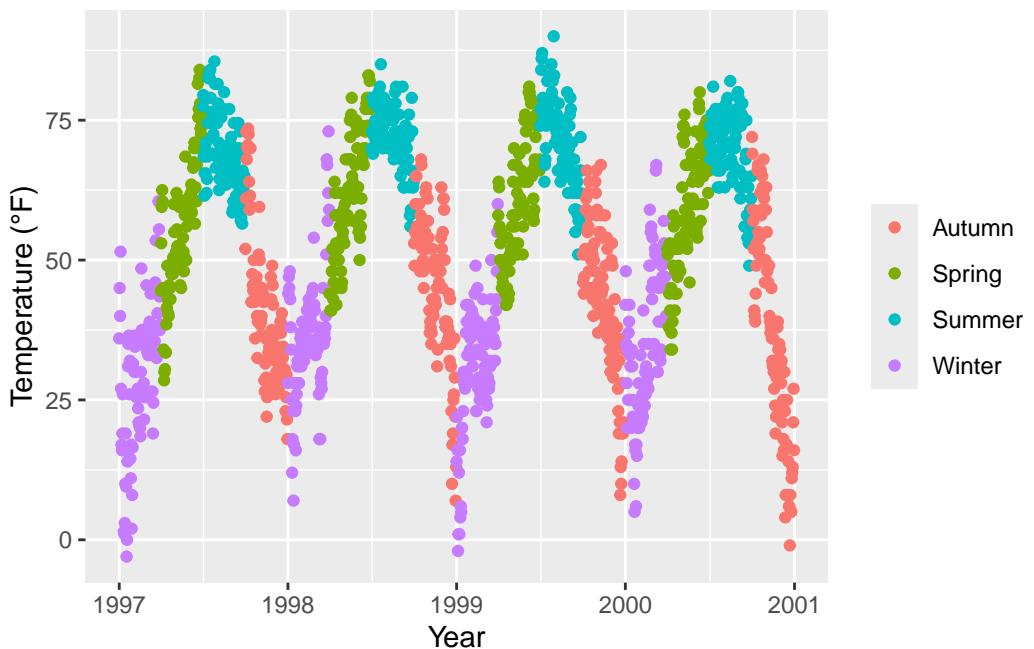
As we've previously learned, utilize `element_blank()` to render *nothing*:

```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.title = element_blank())
```

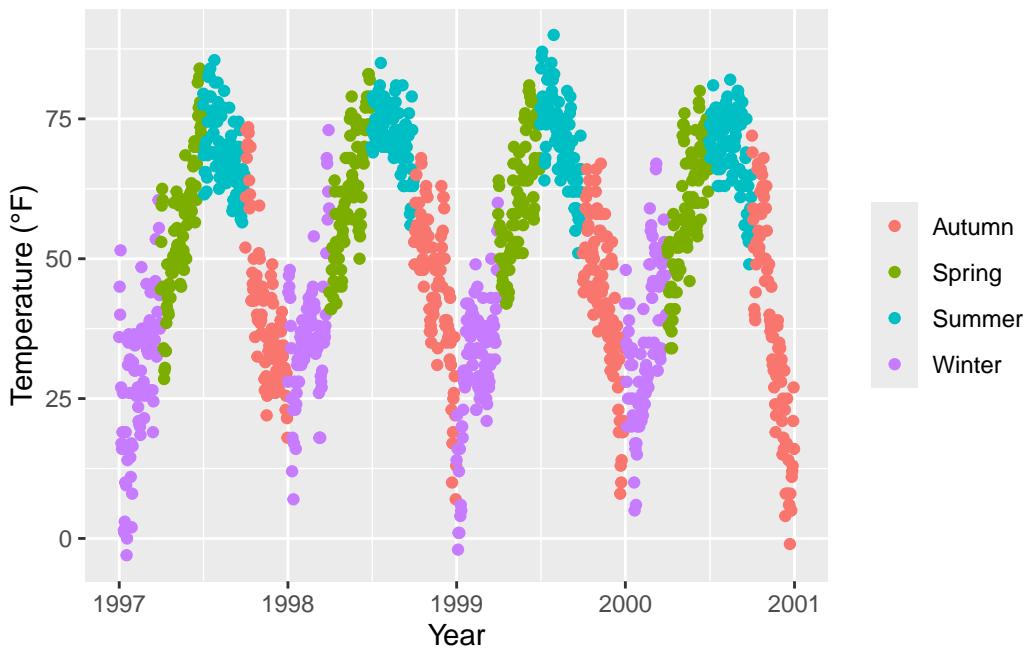


You can achieve the same outcome by setting the legend name to `NULL`, either through `scale_color_discrete(name = NULL)` or `labs(color = NULL)`. Expand to see examples.

```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  scale_color_discrete(name = NULL)
```



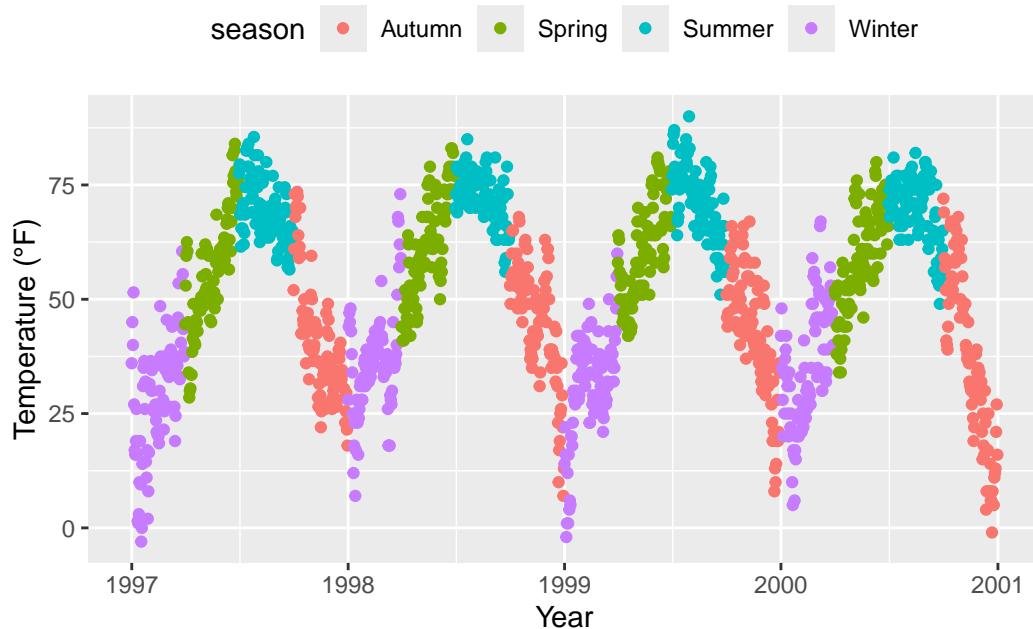
```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  labs(color = NULL)
```



6.3 Adjusting Legend Position

To relocate the legend from its default position on the right side, you can use the `legend.position` argument within `theme`. Available positions include “top”, “right” (the default), “bottom”, and “left”.

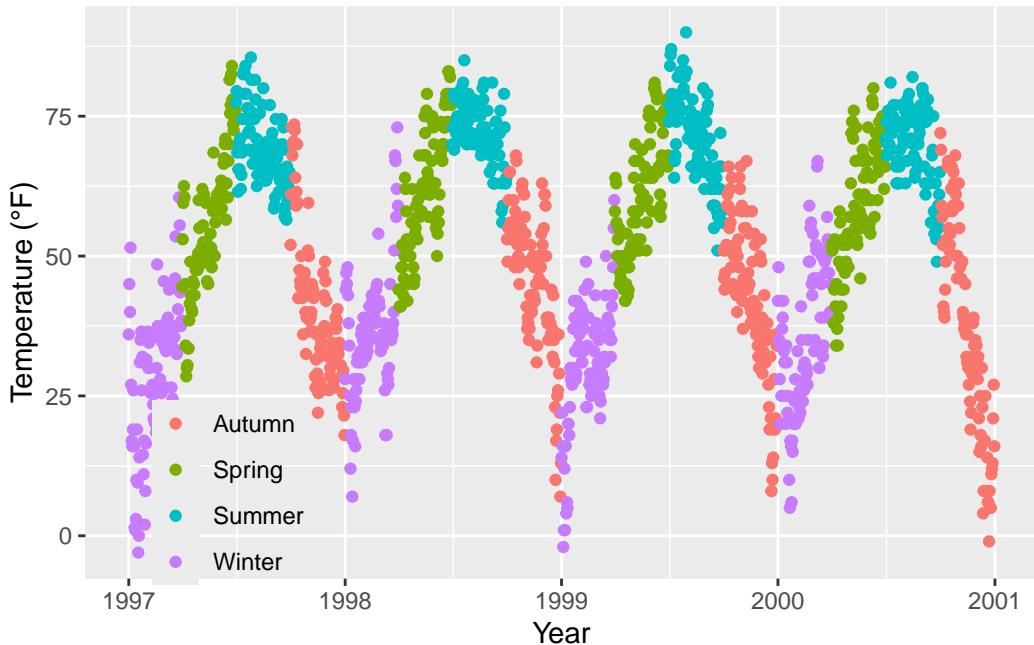
```
ggplot(chic, aes(x = date, y = temp, color = season)) +  
  geom_point() +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(legend.position = "top")
```



You can also position the legend inside the panel by specifying a vector with relative `x` and `y` coordinates ranging from 0 (left or bottom) to 1 (right or top):

```
ggplot(chic, aes(x = date, y = temp, color = season)) +  
  geom_point() +  
  labs(x = "Year", y = "Temperature (°F)",  
       color = NULL) +  
  theme(legend.position = c(.15, .15),  
        legend.background = element_rect(fill = "transparent"))
```

Warning: A numeric `legend.position` argument in `theme()` was deprecated in ggplot2 3.5.0.
i Please use the `legend.position.inside` argument of `theme()` instead.

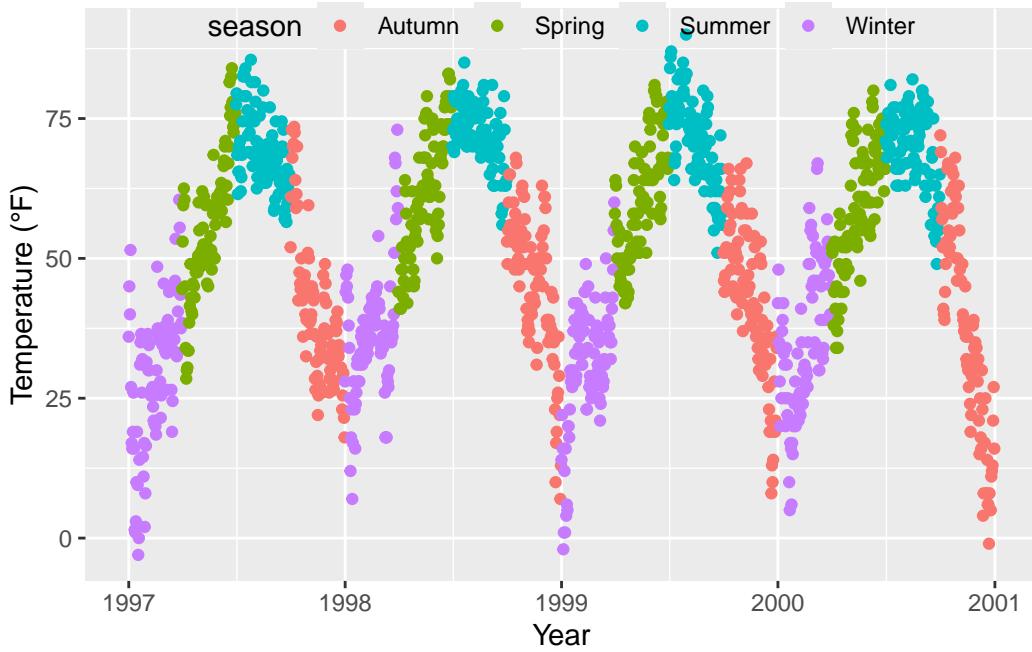


Here, I also override the default white legend background with a transparent fill to ensure the legend doesn't obscure any data points.

6.4 Modifying Legend Direction

By default, the legend direction is vertical. However, when you select either the “top” or “bottom” position, it becomes horizontal. Nevertheless, you can freely switch the direction as desired:

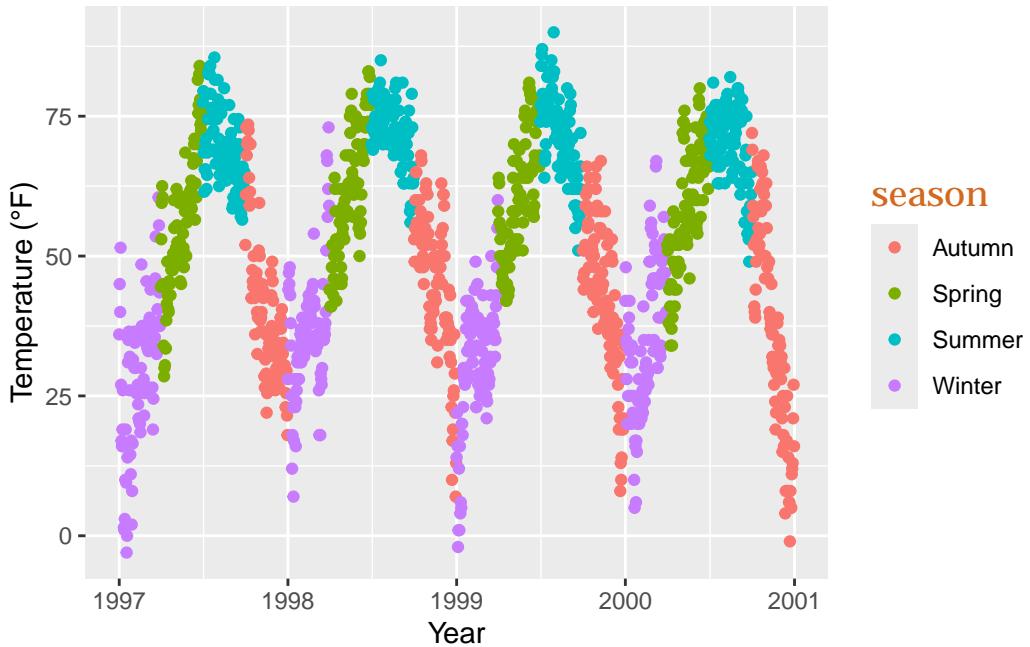
```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.position = c(.5, .97),
        legend.background = element_rect(fill = "transparent")) +
  guides(color = guide_legend(direction = "horizontal"))
```



6.5 Change Style of the Legend Title

You can customize the appearance of the legend title by adjusting the theme element `legend.title`:

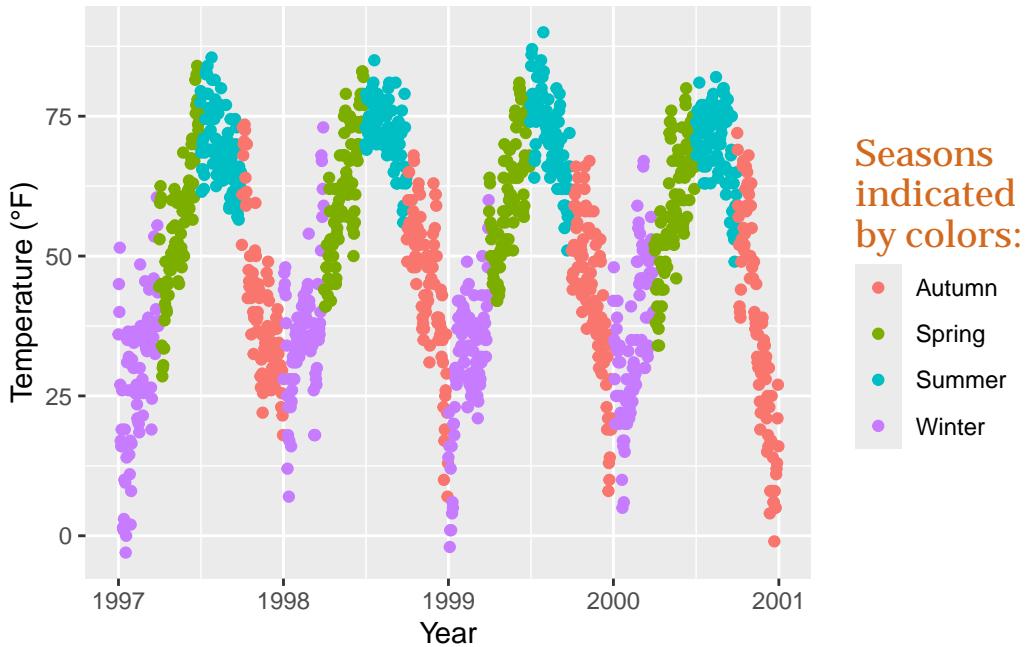
```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.title = element_text(family = "Playfair Display",
                                    color = "chocolate",
                                    size = 14, face = "bold"))
```



6.6 Modifying Legend Title

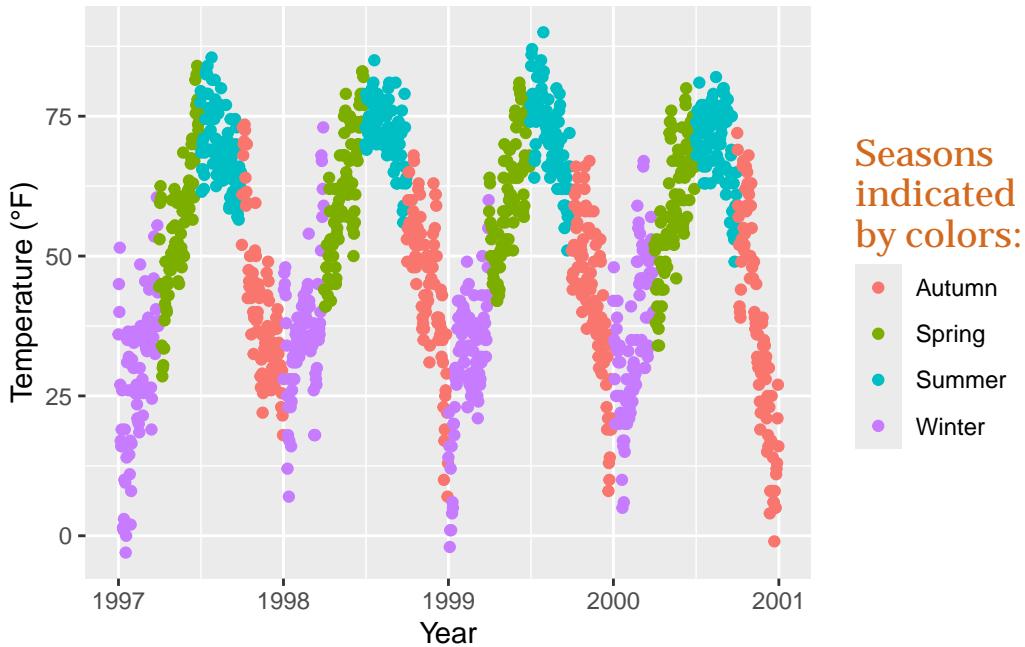
The simplest method to change the title of the legend is through the `labs()` layer:

```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)",
       color = "Seasons\nindicated\nby colors:") +
  theme(legend.title = element_text(family = "Playfair Display",
                                    color = "chocolate",
                                    size = 14, face = "bold"))
```



You can adjust the legend details using `scale_color_discrete(name = "title")` or `guides(color = guide_legend("title"))`:

```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.title = element_text(family = "Playfair Display",
                                    color = "chocolate",
                                    size = 14, face = "bold")) +
  scale_color_discrete(name = "Seasons\nindicated\nby colors:")
```

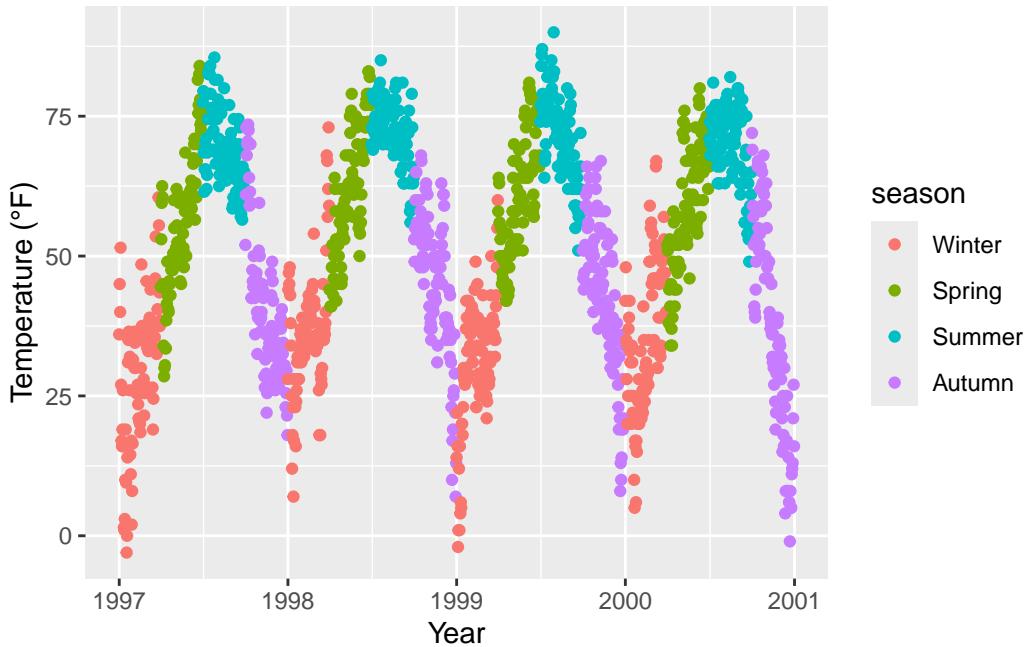


6.7 Rearrange Order of Legend Keys

This can be accomplished by changing the levels of `season`:

```
chic$season <-
  factor(chic$season,
        levels = c("Winter", "Spring", "Summer", "Autumn"))

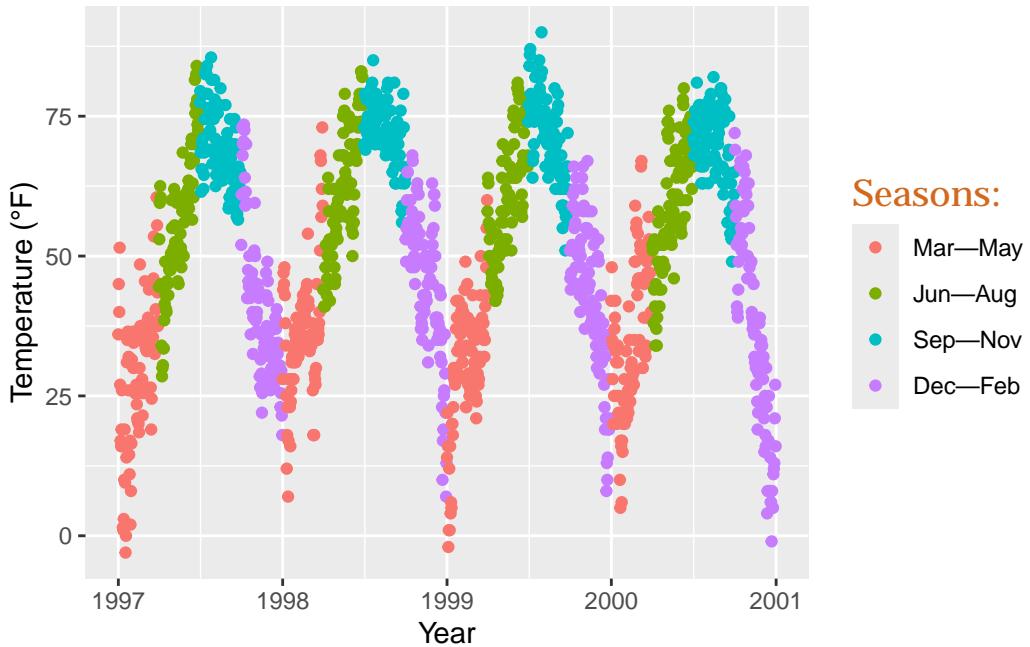
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)")
```



6.8 Modify Legend Labels

To replace the seasons with the months they represent, provide a vector of names in the `scale_color_discrete()` call:

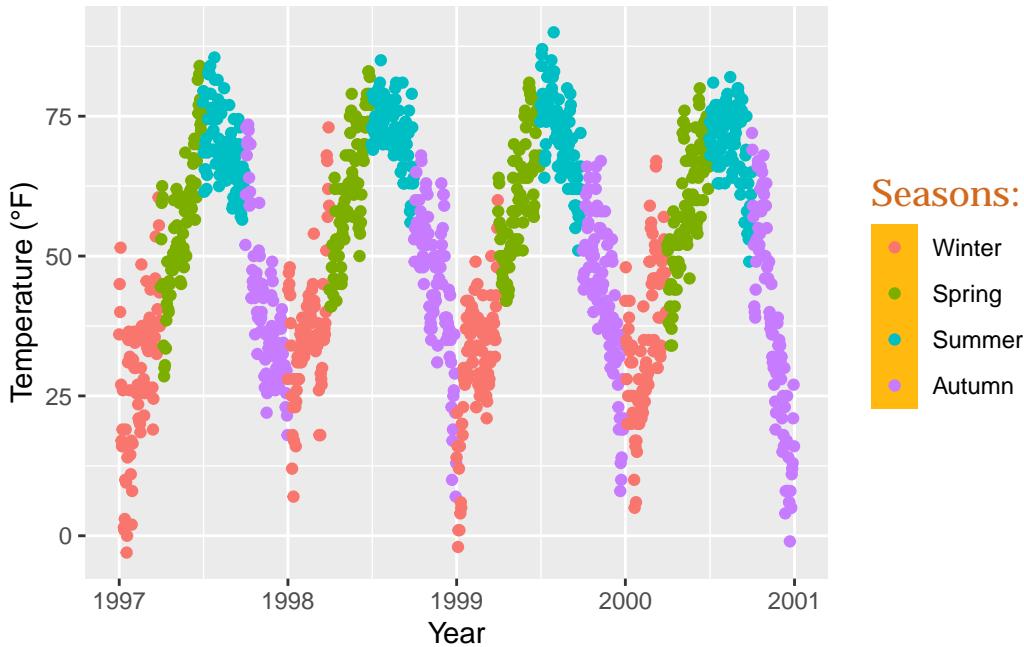
```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  scale_color_discrete(
    name = "Seasons:",
    labels = c("Mar-May", "Jun-Aug", "Sep-Nov", "Dec-Feb")
  ) +
  theme(legend.title = element_text(
    family = "Playfair Display", color = "chocolate", size = 14, face = 2
))
```



6.9 Adjust Background Boxes in the Legend

To alter the background color (fill) of the legend keys, we modify the setting for the theme element `legend.key`:

```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.key = element_rect(fill = "darkgoldenrod1"),
        legend.title = element_text(family = "Playfair Display",
                                    color = "chocolate",
                                    size = 14, face = 2)) +
  scale_color_discrete("Seasons:")
```

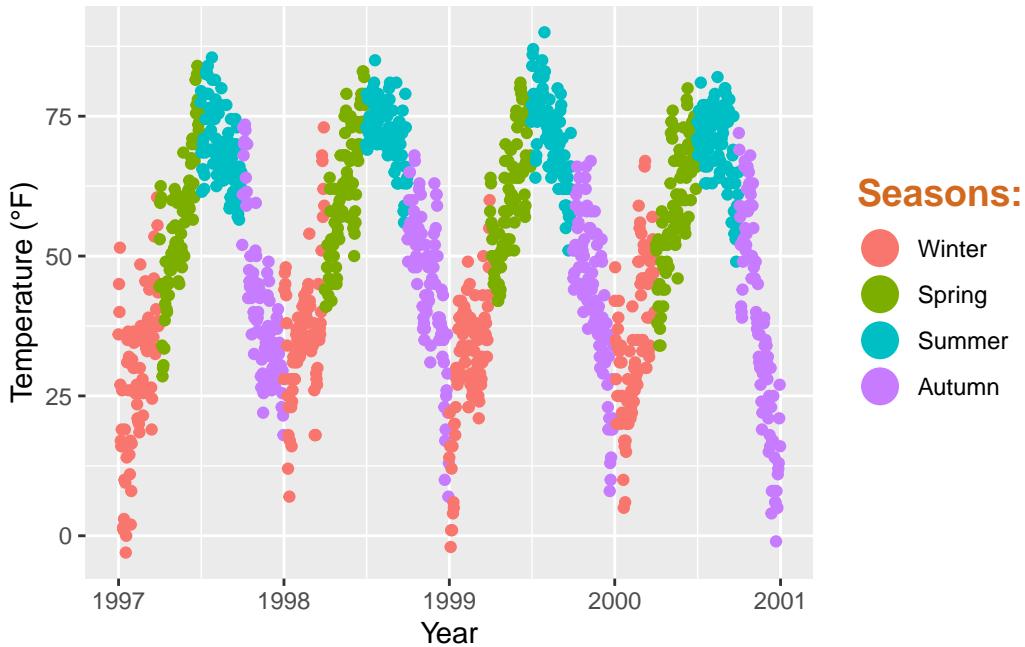


If you wish to remove them entirely, use `fill = NA` or `fill = "transparent"`.

6.10 Adjust Size of Legend Symbols

The default size of points in the legend may cause them to appear too small, especially without boxes. To modify this, you can again use the `guides` layer as follows:

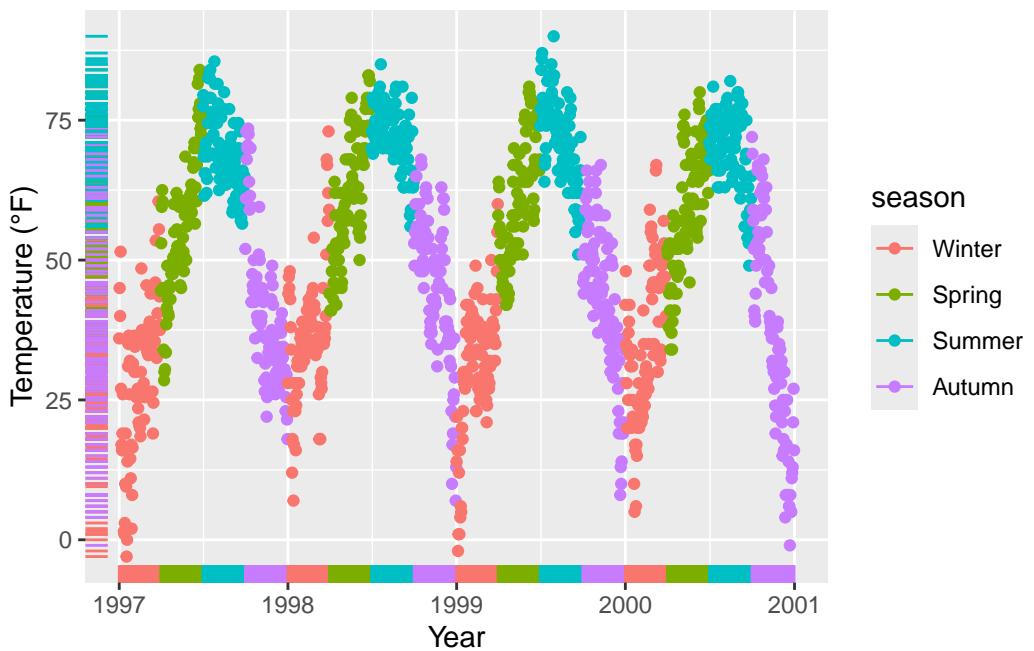
```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.key = element_rect(fill = NA),
        legend.title = element_text(color = "chocolate",
                                     size = 14, face = 2)) +
  scale_color_discrete("Seasons:") +
  guides(color = guide_legend(override.aes = list(size = 6)))
```



6.11 Exclude a Layer from the Legend

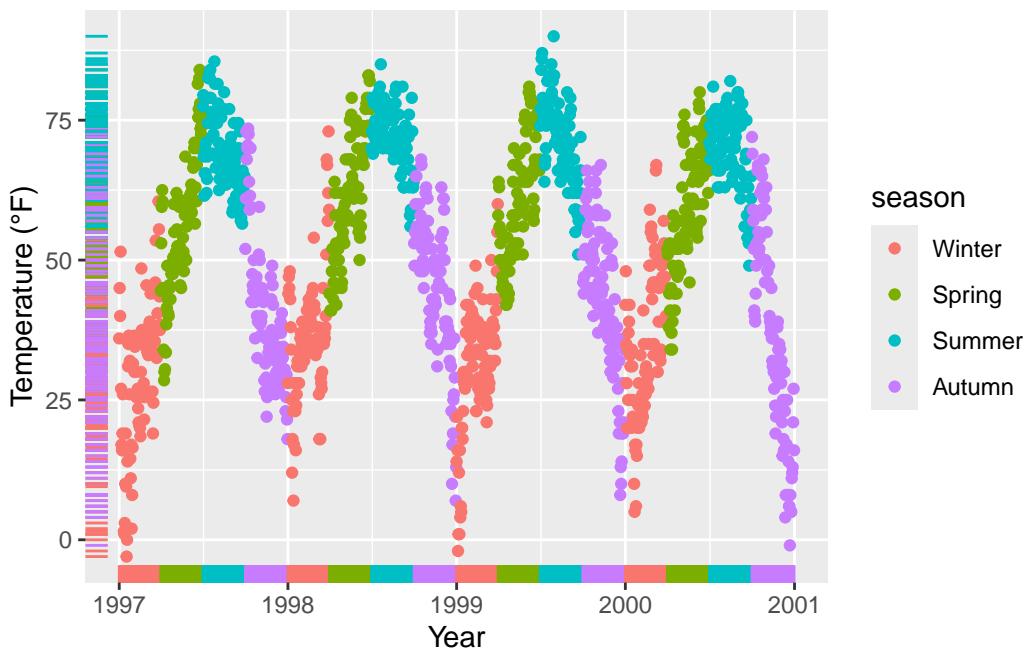
Suppose you have two different geometric layers mapped to the same variable, such as color being used as an aesthetic for both a point layer and a rug layer of the same data. By default, both the points and the “line” end up in the legend like this:

```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  geom_rug()
```



You can utilize `show.legend = FALSE` to exclude a layer from the legend:

```
ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  geom_rug(show.legend = FALSE)
```

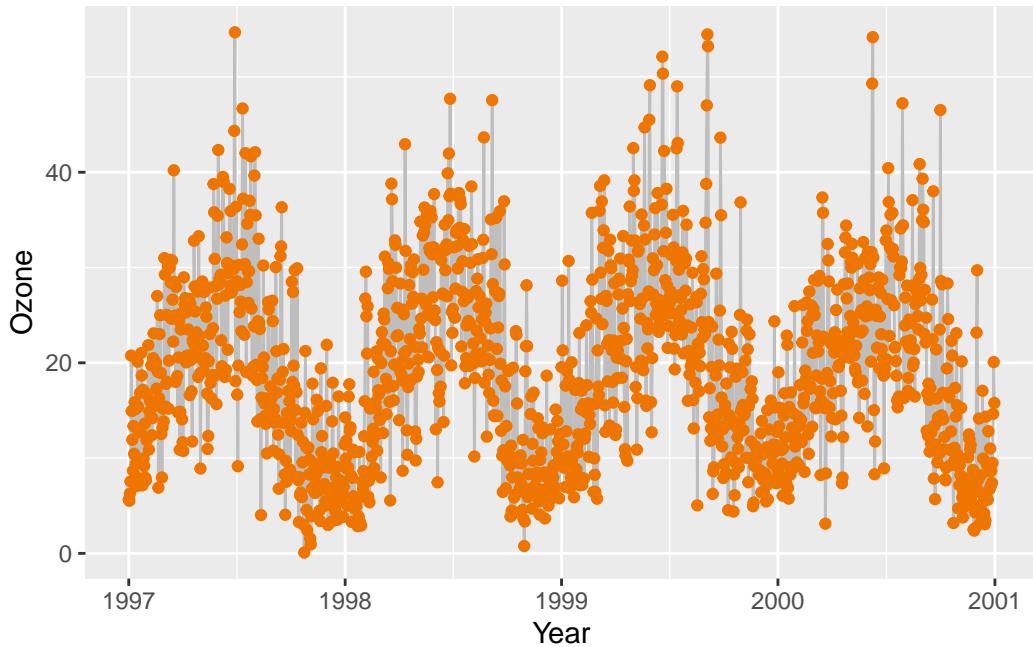


6.12 Manually Adding Legend Items

By default, `{ggplot2}` won't add a legend unless you map aesthetics (color, size, etc.) to a variable. However, there are occasions where you may want to include a legend for clarity.

Here's the default behavior:

```
ggplot(chic, aes(x = date, y = o3)) +
  geom_line(color = "gray") +
  geom_point(color = "darkorange2") +
  labs(x = "Year", y = "Ozone")
```



To force a legend, we can map a guide to a *variable*. Here, we're mapping the lines and the points using `aes()`, but we're not mapping to a variable in our dataset. Instead, we're using a single string for each, ensuring we get just one color for each.

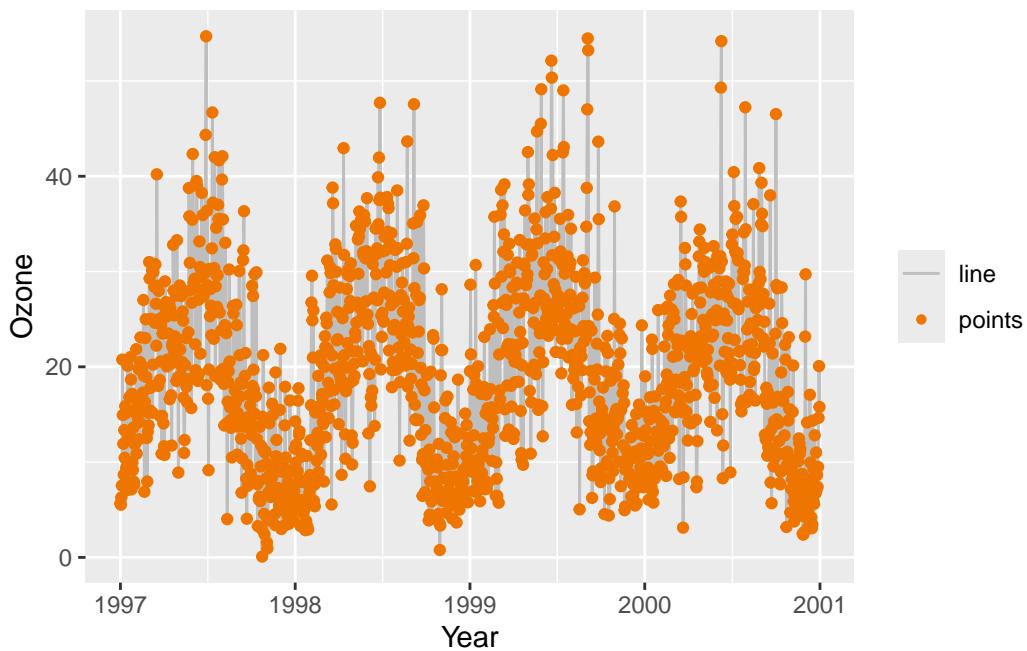
```
ggplot(chic, aes(x = date, y = o3)) +
  geom_line(aes(color = "line")) +
  geom_point(aes(color = "points")) +
  labs(x = "Year", y = "Ozone") +
  scale_color_discrete("Type:")
```



We're getting close, but this is not what we want. We desire gray lines and red points. To change the colors, we use `scale_color_manual()`. Additionally, we override the legend aesthetics using the `guide()` function.

Now, we have a plot with gray lines and red points, as well as a single gray line and a single red point as legend symbols.

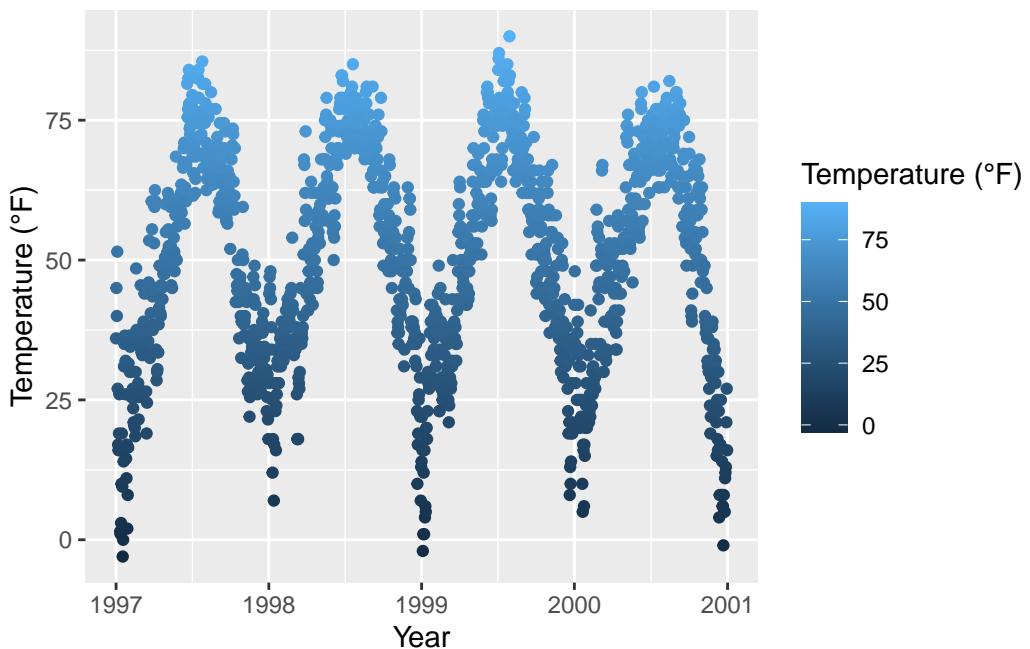
```
ggplot(chic, aes(x = date, y = o3)) +
  geom_line(aes(color = "line")) +
  geom_point(aes(color = "points")) +
  labs(x = "Year", y = "Ozone") +
  scale_color_manual(name = NULL,
                     guide = "legend",
                     values = c("points" = "darkorange2",
                               "line" = "gray")) +
  guides(color = guide_legend(override.aes = list(linetype = c(1, 0),
                                                 shape = c(NA, 16))))
```



6.13 Use Other Legend Styles

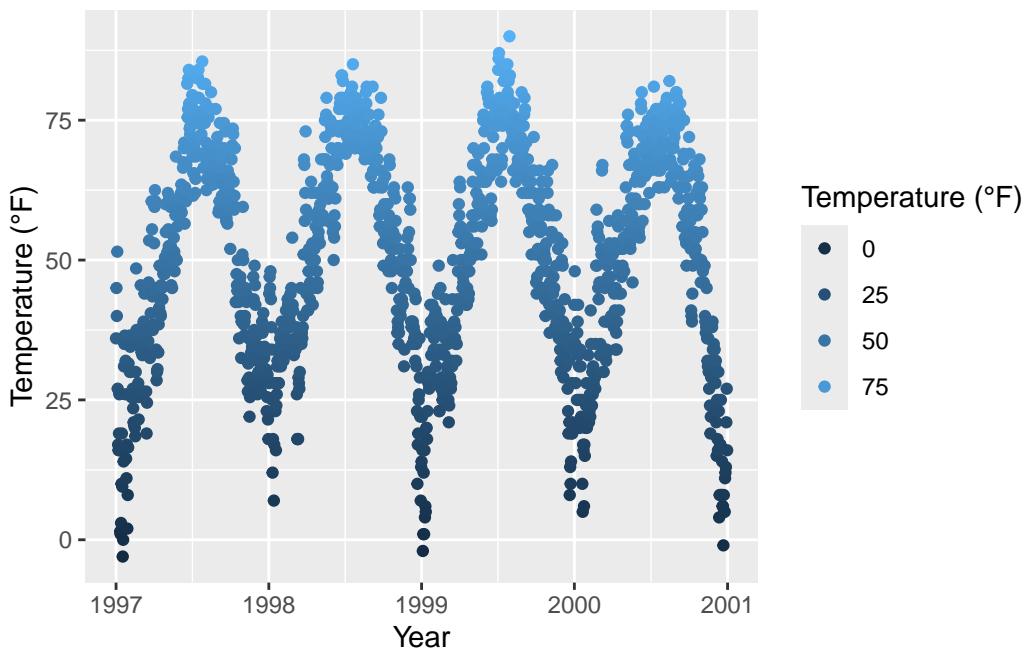
The default legend for categorical variables such as `season` is a `guide_legend()`, as you have seen in several previous examples. However, if you map a continuous variable to an aesthetic, `{ggplot2}` will by default not use `guide_legend()` but `guide_colorbar()` (or `guide_colourbar()`).

```
ggplot(chic,
       aes(x = date, y = temp, color = temp)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)", color = "Temperature (°F)")
```



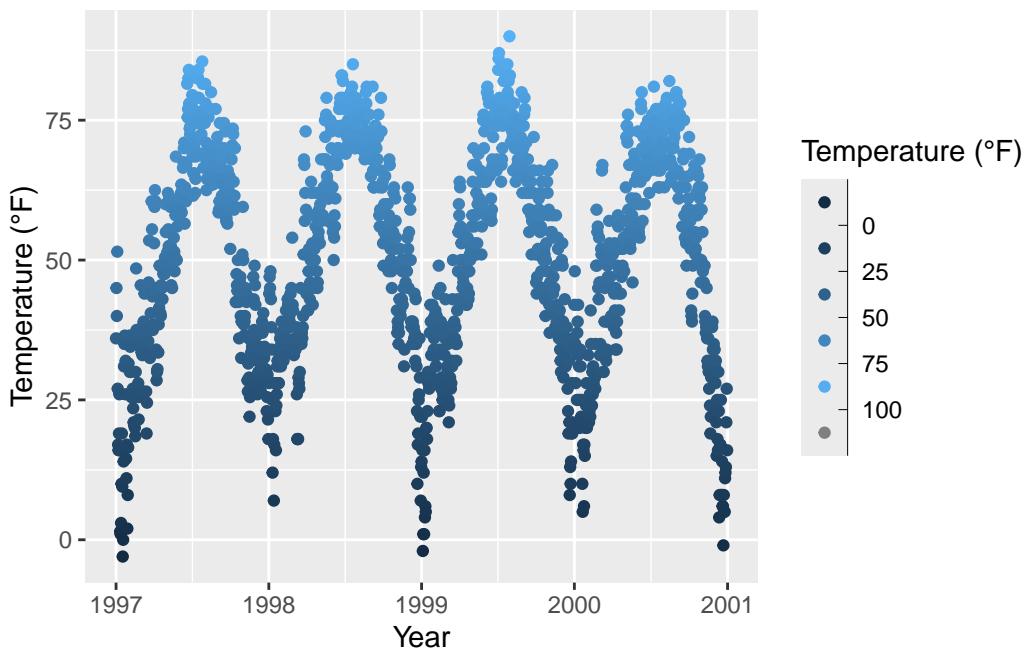
However, by using `guide_legend()`, you can force the legend to display discrete colors for a given number of breaks as in the case of a categorical variable:

```
ggplot(chic,
       aes(x = date, y = temp, color = temp)) +
  geom_point() +
  labs(x = "Year", y = "Temperature ( $^{\circ}$ F)", color = "Temperature ( $^{\circ}$ F)") +
  guides(color = guide_legend())
```



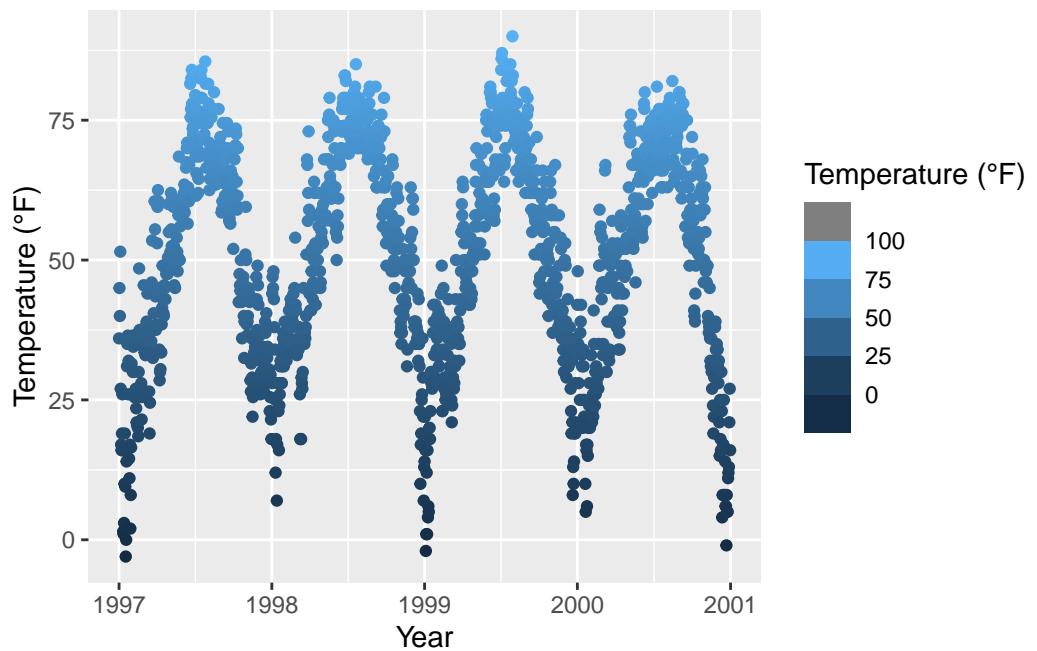
You can also utilize binned scales:

```
ggplot(chic,
       aes(x = date, y = temp, color = temp)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)", color = "Temperature (°F)") +
  guides(color = guide_bins())
```



... or binned scales as discrete colorbars:

```
ggplot(chic,
       aes(x = date, y = temp, color = temp)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)", color = "Temperature (°F)") +
  guides(color = guide_colorsteps())
```



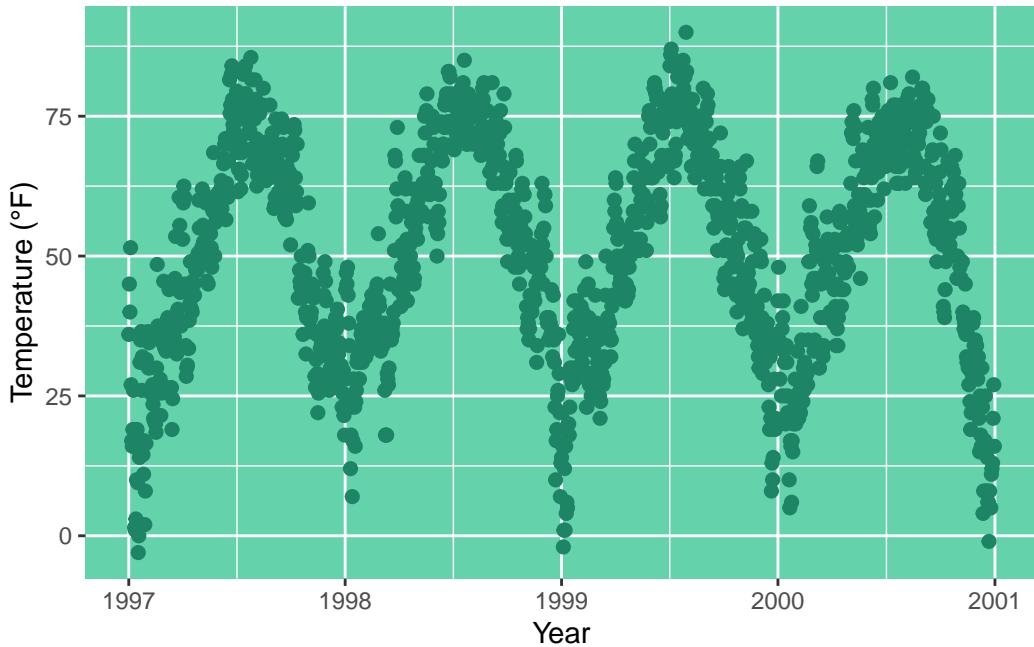
7 Working with Backgrounds & Grid Lines

To modify the overall appearance of your plot, you can use various functions. While altering the entire theme of your plot is one option (covered in detail in the “Working with Themes” section below), you can also make specific changes to individual elements such as backgrounds and grid lines.

7.1 Change the Panel Background Color

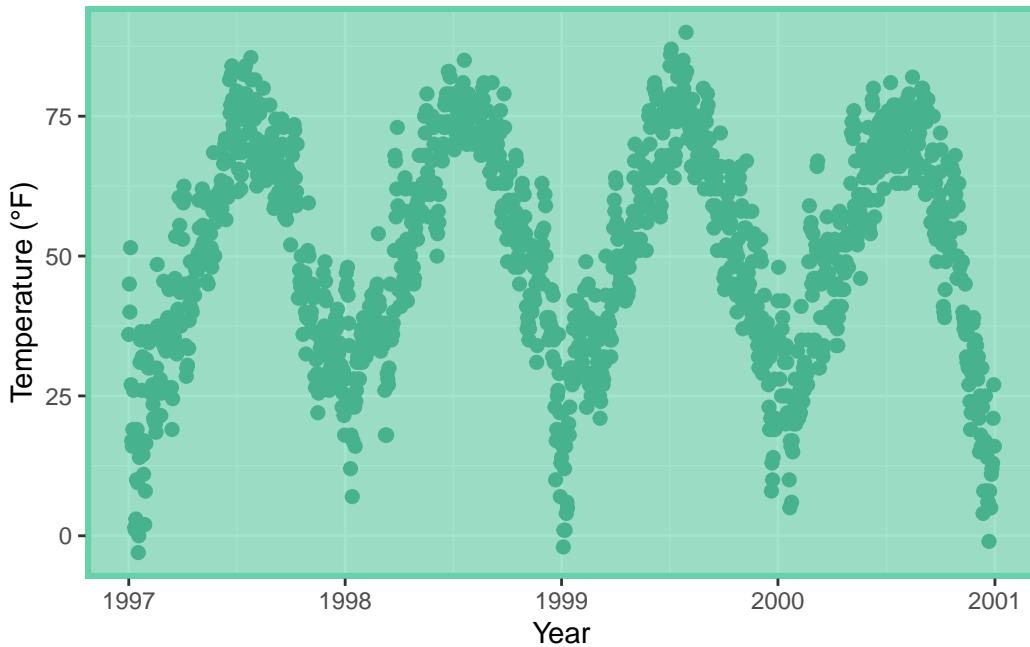
You can adjust the background color (fill) of the panel area (where the data is plotted) by modifying the theme element `panel.background`:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "#1D8565", size = 2) +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(panel.background = element_rect(  
    fill = "#64D2AA", color = "#64D2AA", linewidth = 2)  
)
```



Keep in mind that the true color — the outline of the panel background — didn't change despite our specification. This is because there's a layer on top of `panel.background`, namely `panel.border`. However, it's important to use a transparent fill here; otherwise, your data will be hidden behind this layer. In the following example, I illustrate this by using a semi-transparent hex color for the `fill` argument in `element_rect`:

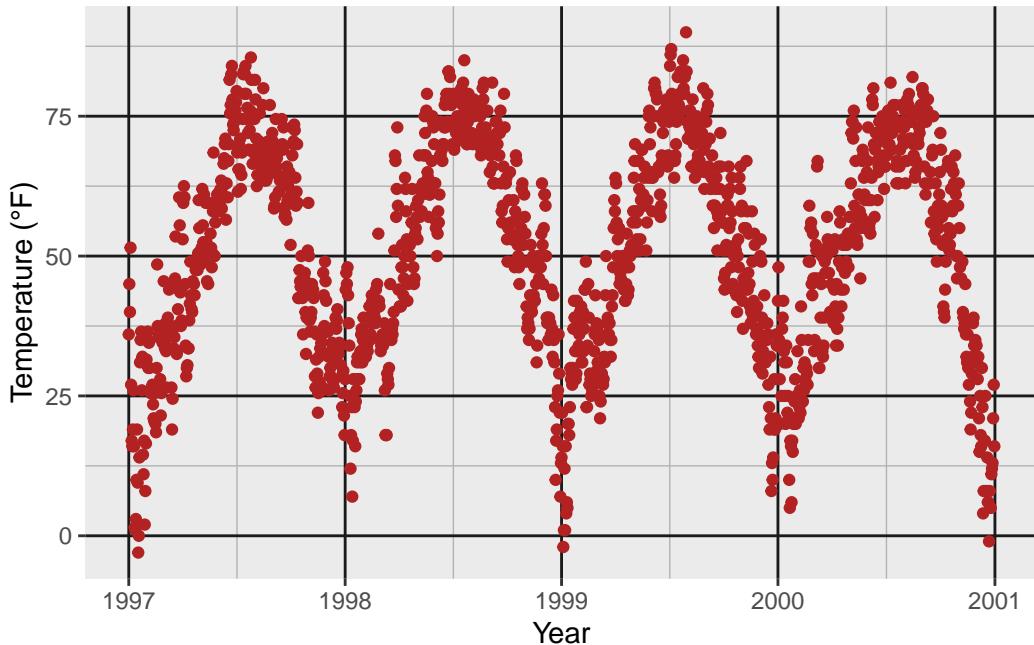
```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "#1D8565", size = 2) +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(panel.border = element_rect(
    fill = "#64D2AA99", color = "#64D2AA", linewidth = 2)
)
```



7.2 Change Grid Lines

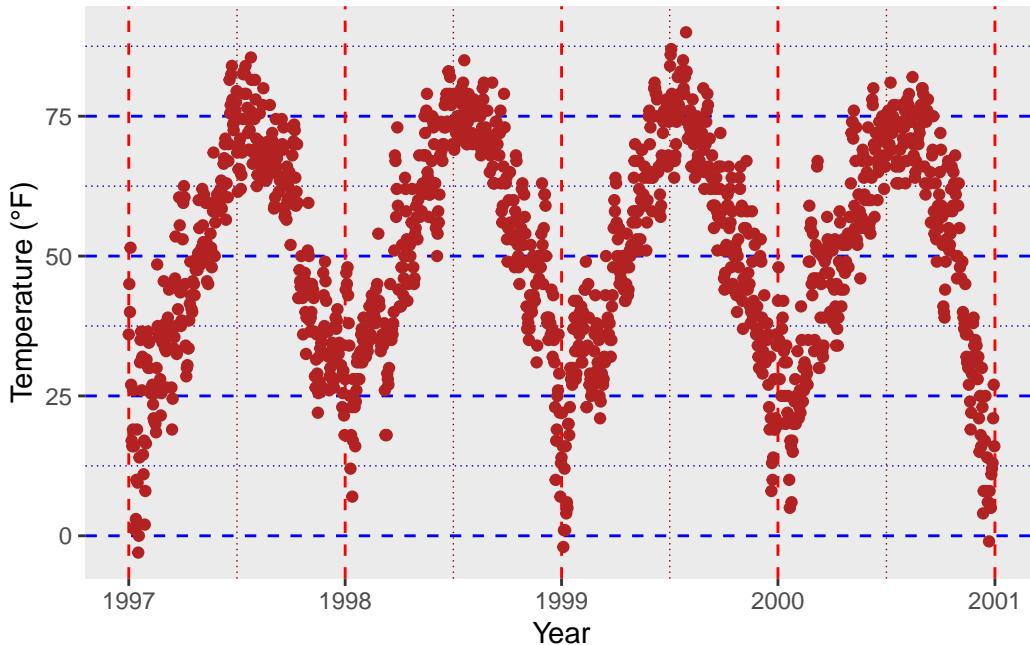
There are two types of grid lines: major grid lines indicating the ticks and minor grid lines between the major ones. You can customize both by overwriting the defaults for `panel.grid` or for each set of gridlines separately, `panel.grid.major` and `panel.grid.minor`.

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(panel.grid.major = element_line(color = "gray10", linewidth = .5),  
        panel.grid.minor = element_line(color = "gray70", linewidth = .25))
```



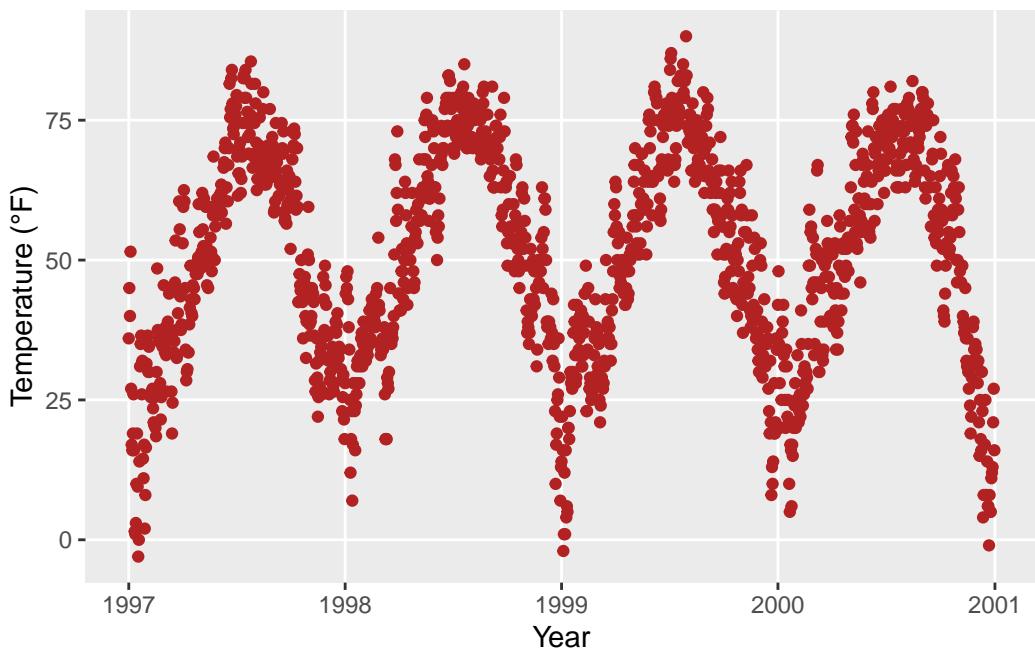
You can even specify settings for all four different levels of grid lines: major horizontal, major vertical, minor horizontal, and minor vertical.

```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(panel.grid.major = element_line(linewidth = .5, linetype = "dashed"),
        panel.grid.minor = element_line(linewidth = .25, linetype = "dotted"),
        panel.grid.major.x = element_line(color = "red1"),
        panel.grid.major.y = element_line(color = "blue1"),
        panel.grid.minor.x = element_line(color = "red4"),
        panel.grid.minor.y = element_line(color = "blue4"))
```

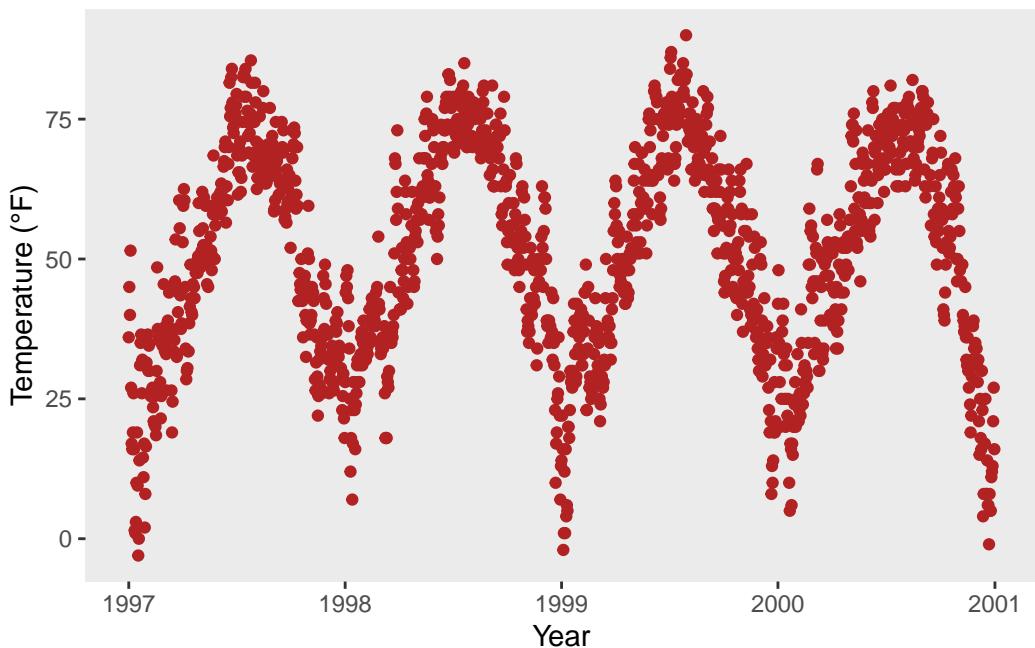


And, of course, you can remove some or all grid lines if you like. For instance, to remove all grid lines, you can set `panel.grid = element_blank()`. Alternatively, you can remove only major or minor grid lines by specifying `panel.grid.major` or `panel.grid.minor` accordingly and setting them to `element_blank()`.

```
ggplot(chic, aes(x = date, y = temp)) +
  geom_point(color = "firebrick") +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(panel.grid.minor = element_blank())
```



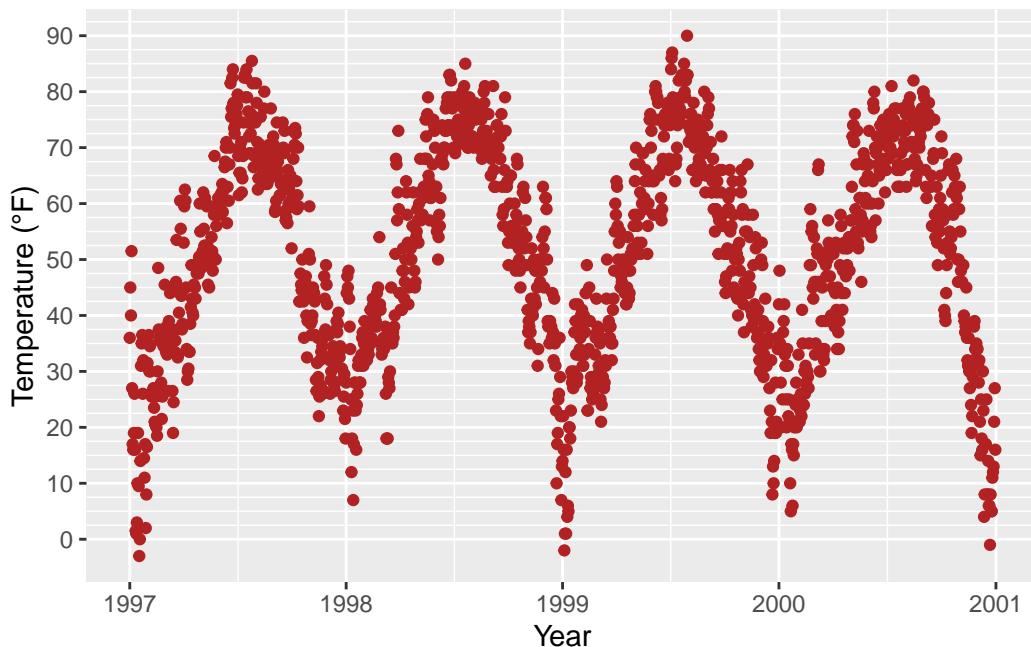
```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(panel.grid = element_blank())
```



7.3 Change Spacing of Gridlines

Furthermore, you can also define the breaks between both major and minor grid lines by specifying the `breaks` argument.

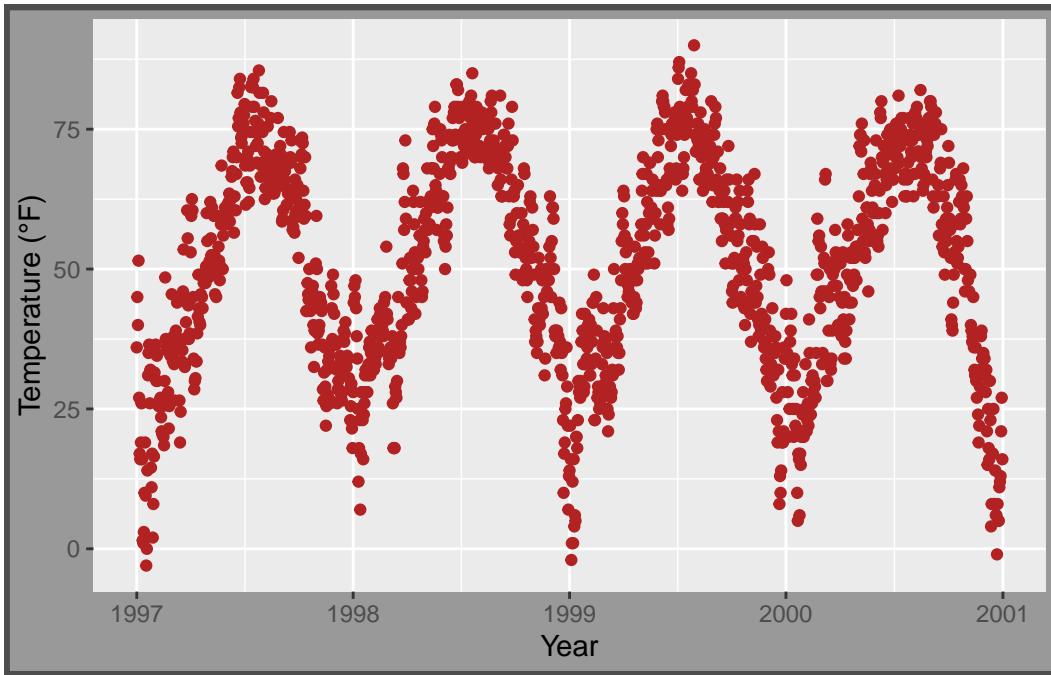
```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  scale_y_continuous(breaks = seq(0, 100, 10),  
                     minor_breaks = seq(0, 100, 2.5))
```



7.4 Change the Plot Background Color

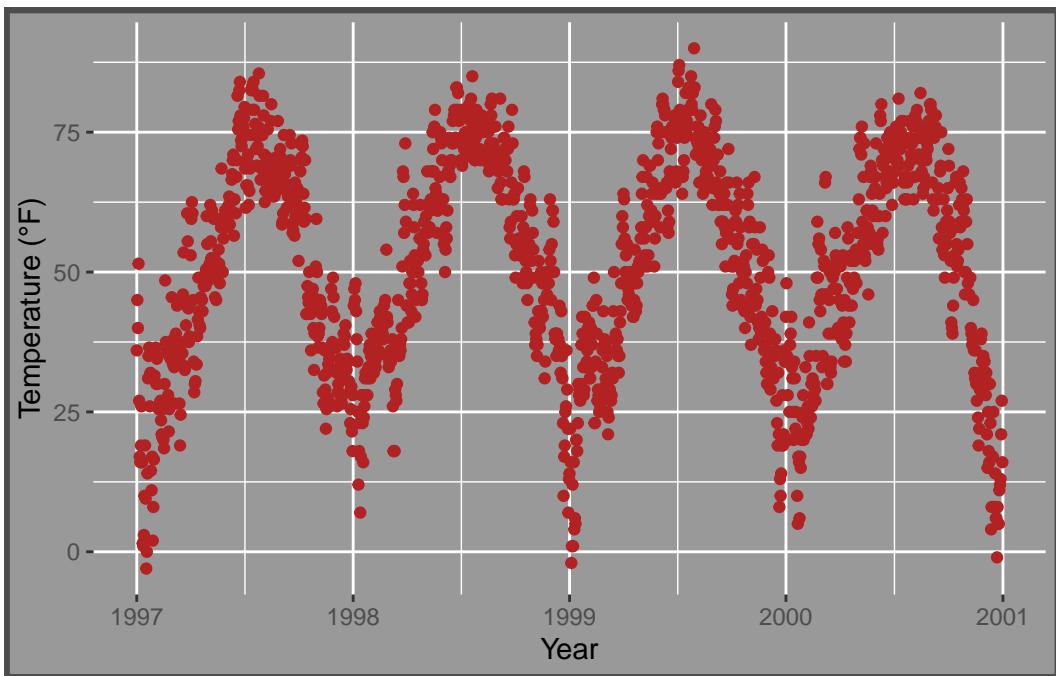
Similarly, to change the background color (fill) of the plot area, you can modify the theme element `plot.background` using the `theme()` function. This allows you to customize the appearance of the entire plot area according to your preferences.

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(plot.background = element_rect(fill = "gray60",  
                                       color = "gray30", linewidth = 2))
```



You can achieve a unique background color by either setting the same colors in both `panel.background` and `plot.background` or by setting the background filling of the panel to "transparent" or NA. This customization can help you create visually appealing plots that match your design preferences.

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(panel.background = element_rect(fill = NA),  
        plot.background = element_rect(fill = "gray60",  
                                      color = "gray30", linewidth = 2))
```

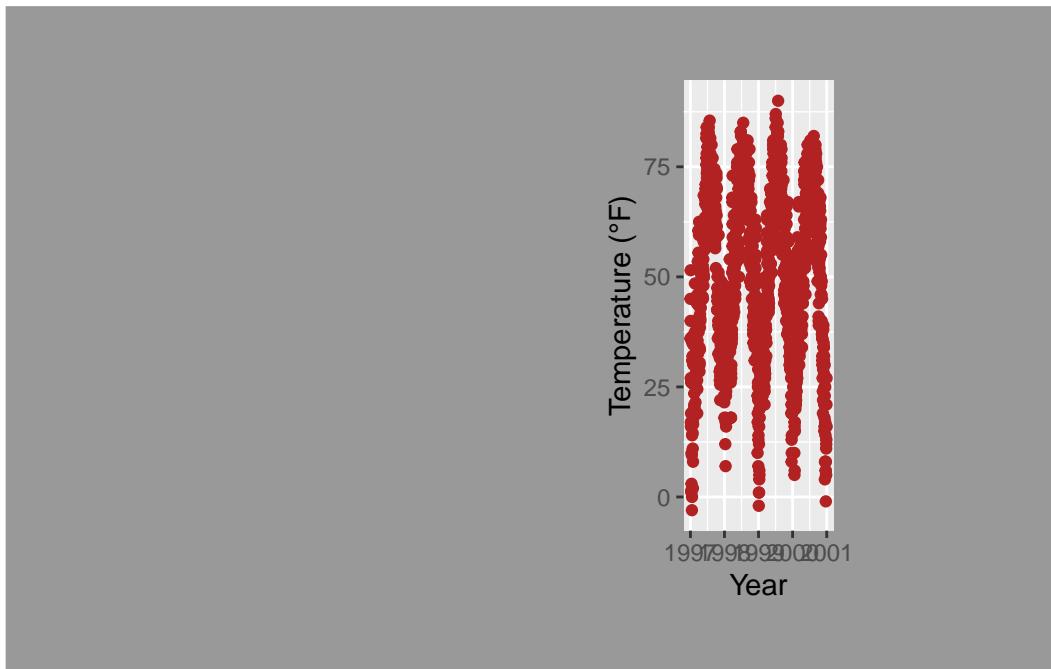


8 Working with Margins

Sometimes it is useful to add a little space to the plot margin. Similar to the previous examples, we can use an argument to the `theme()` function. In this case, the argument is `plot.margin`. As illustrated in the previous example where we changed the background color using `plot.background`, we can now add extra space to both the left and right.

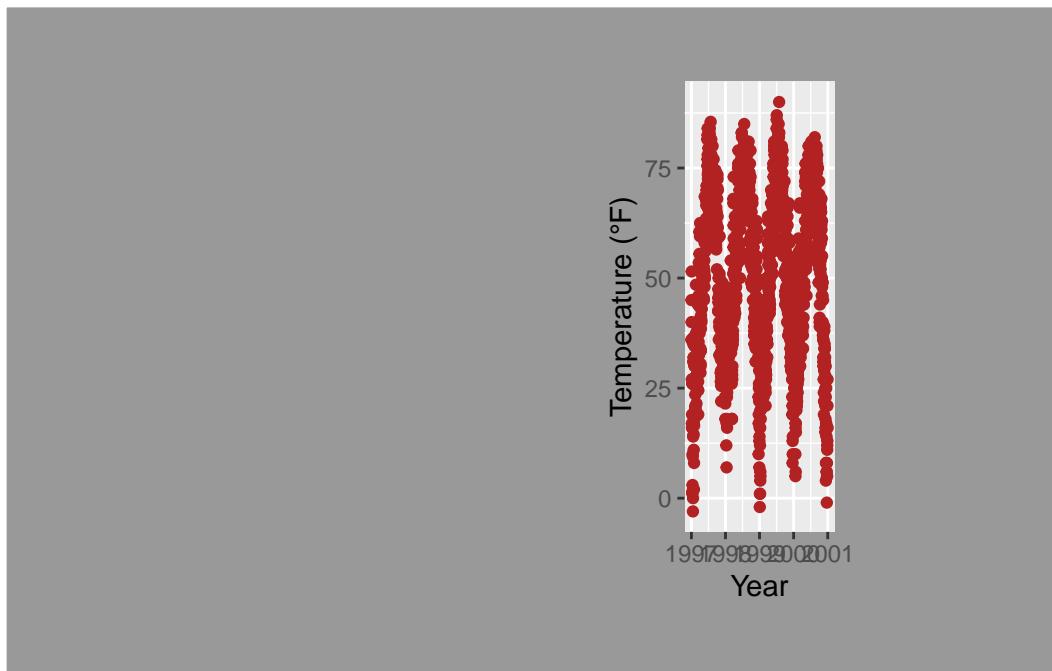
The `plot.margin` argument can handle a variety of different units (cm, inches, etc.), but it requires the use of the `unit` function from the package `grid` to specify the units. You can either provide the same value for all sides (easiest via `rep(x, 4)`) or particular distances for each. Here, I am using a 1cm margin on the top and bottom, 3 cm margin on the right, and an 8 cm margin on the left.

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(plot.background = element_rect(fill = "gray60"),  
        plot.margin = margin(t = 1, r = 3, b = 1, l = 8, unit = "cm"))
```



You can also use `unit()` instead of `margin()`. Expand to see example.

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)") +  
  theme(plot.background = element_rect(fill = "gray60"),  
        plot.margin = unit(c(1, 3, 1, 8), "cm"))
```



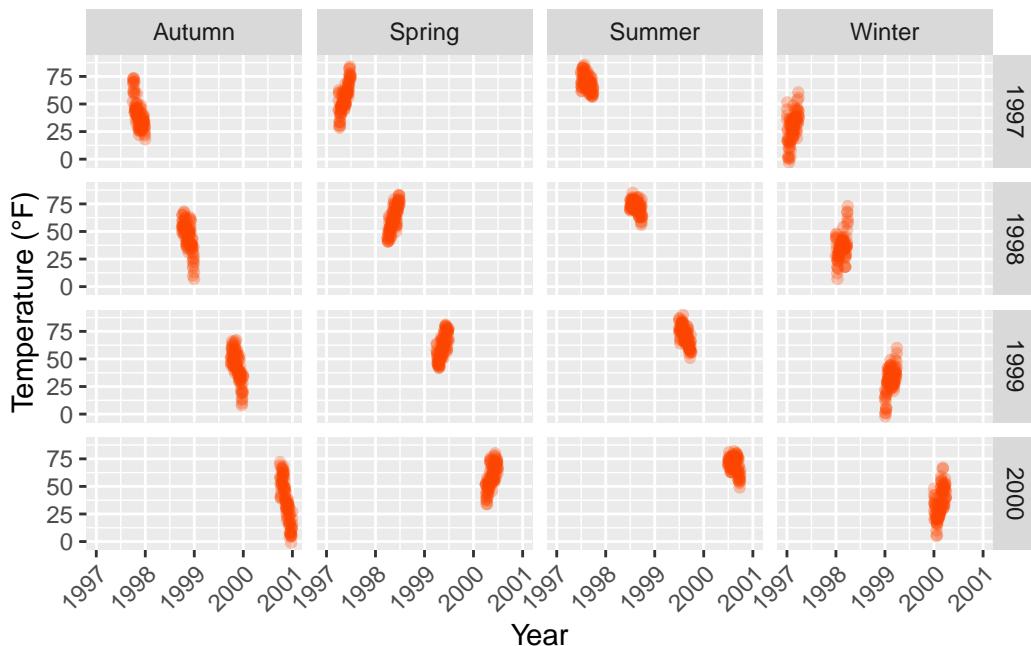
9 Working with Multi-Panel Plots

The `{ggplot2}` package offers two handy functions for creating multi-panel plots, called *facets*. They are related but have slight differences: `facet_wrap` creates a ribbon of plots based on a single variable, while `facet_grid` spans a grid of plots based on two variables.

9.1 Create a Grid of Small Multiples Based on Two Variables

When dealing with two variables, `facet_grid` is the appropriate choice. In this function, the order of the variables determines the number of rows and columns in the grid:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "orangered", alpha = .3) +  
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1)) +  
  labs(x = "Year", y = "Temperature (°F)") +  
  facet_grid(year ~ season)
```

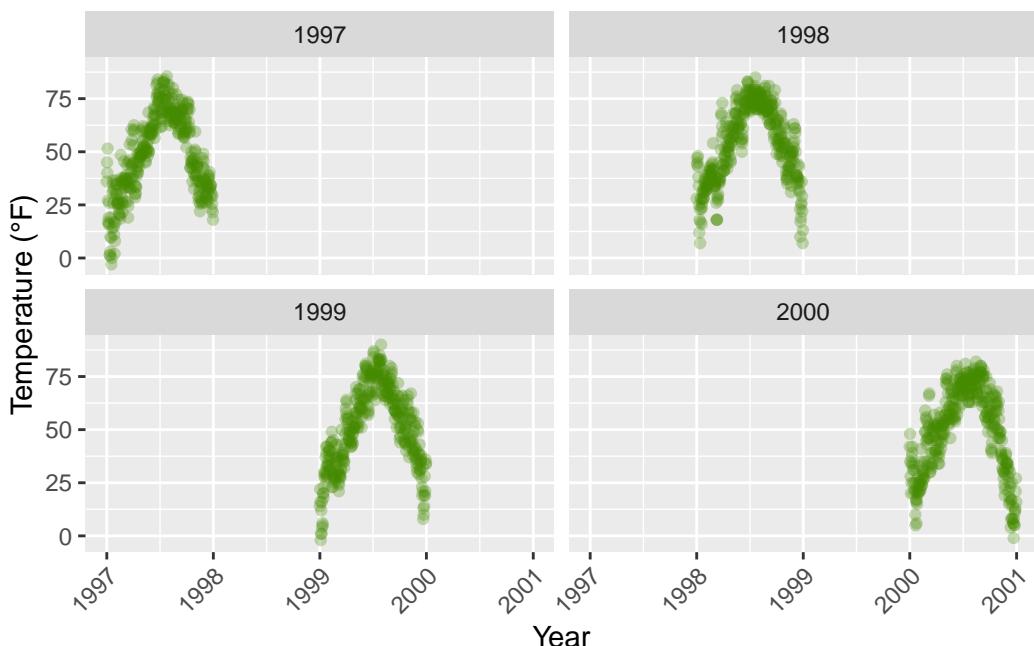


To switch from a row-based arrangement to a column-based one, you can modify `facet_grid(year ~ season)` to `facet_grid(season ~ year)`.

9.2 Create Small Multiples Based on One Variable

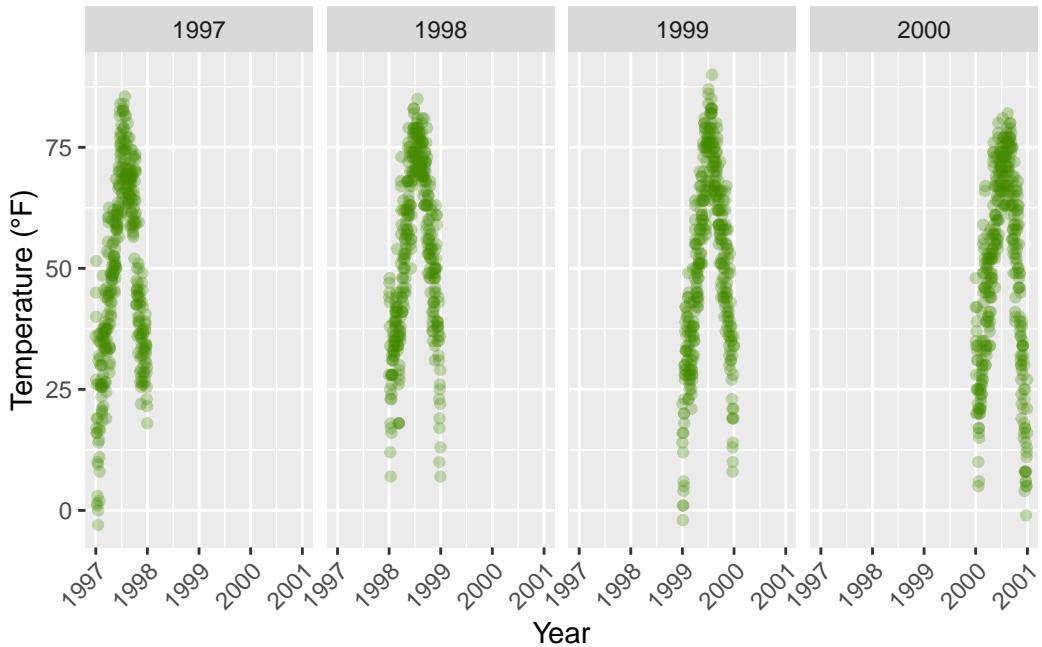
`facet_wrap` creates a facet of a single variable, specified with a tilde in front: `facet_wrap(~ variable)`. The appearance of these subplots is determined by the arguments `ncol` and `nrow`:

```
g <-  
  ggplot(chic, aes(x = date, y = temp)) +  
    geom_point(color = "chartreuse4", alpha = .3) +  
    labs(x = "Year", y = "Temperature (°F)") +  
    theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1))  
  
g + facet_wrap(~ year)
```



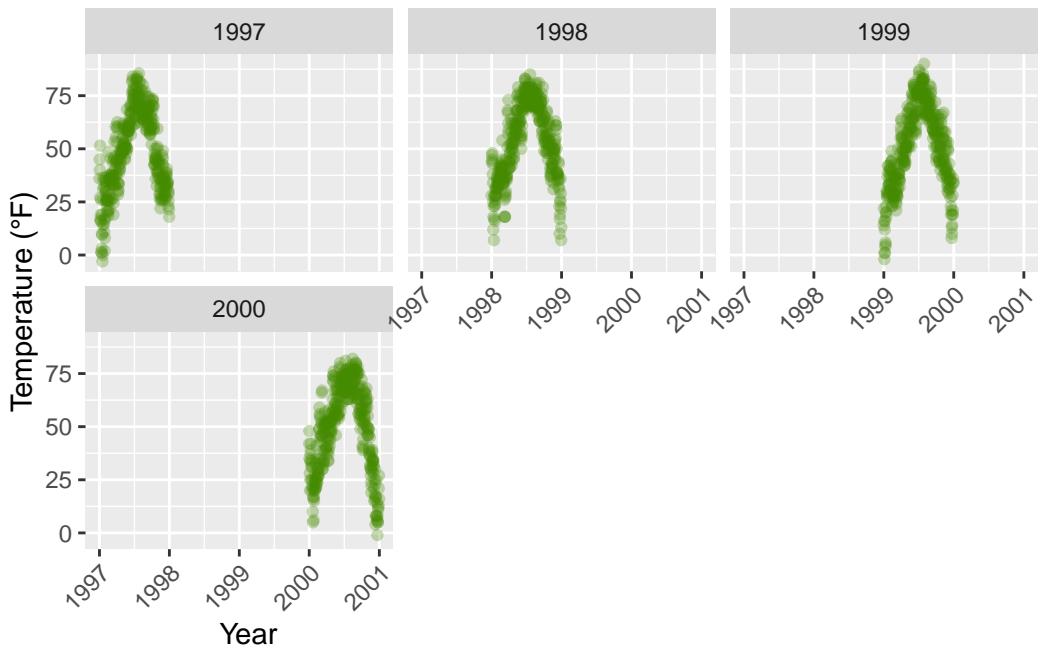
Accordingly, you can arrange the plots as you like, instead as a matrix in one row...

```
g + facet_wrap(~ year, nrow = 1)
```



... or even as a asymmetric grid of plots:

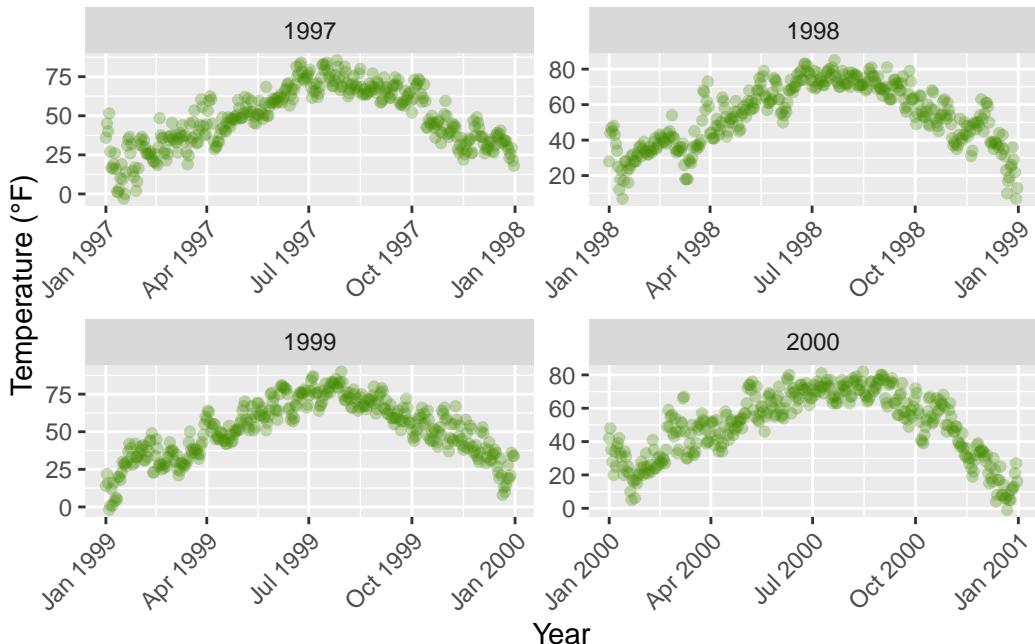
```
g + facet_wrap(~ year, ncol = 3) + theme(axis.title.x = element_text(hjust = .15))
```



9.3 Allow Axes to Roam Free

The default for multi-panel plots in `{ggplot2}` is to use equivalent scales in each panel. But sometimes you want to allow a panels own data to determine the scale. This is often not a good idea since it may give your user the wrong impression about the data. But sometimes it is indeed useful and to do this you can set `scales = "free"`:

```
g + facet_wrap(~ year, nrow = 2, scales = "free")
```

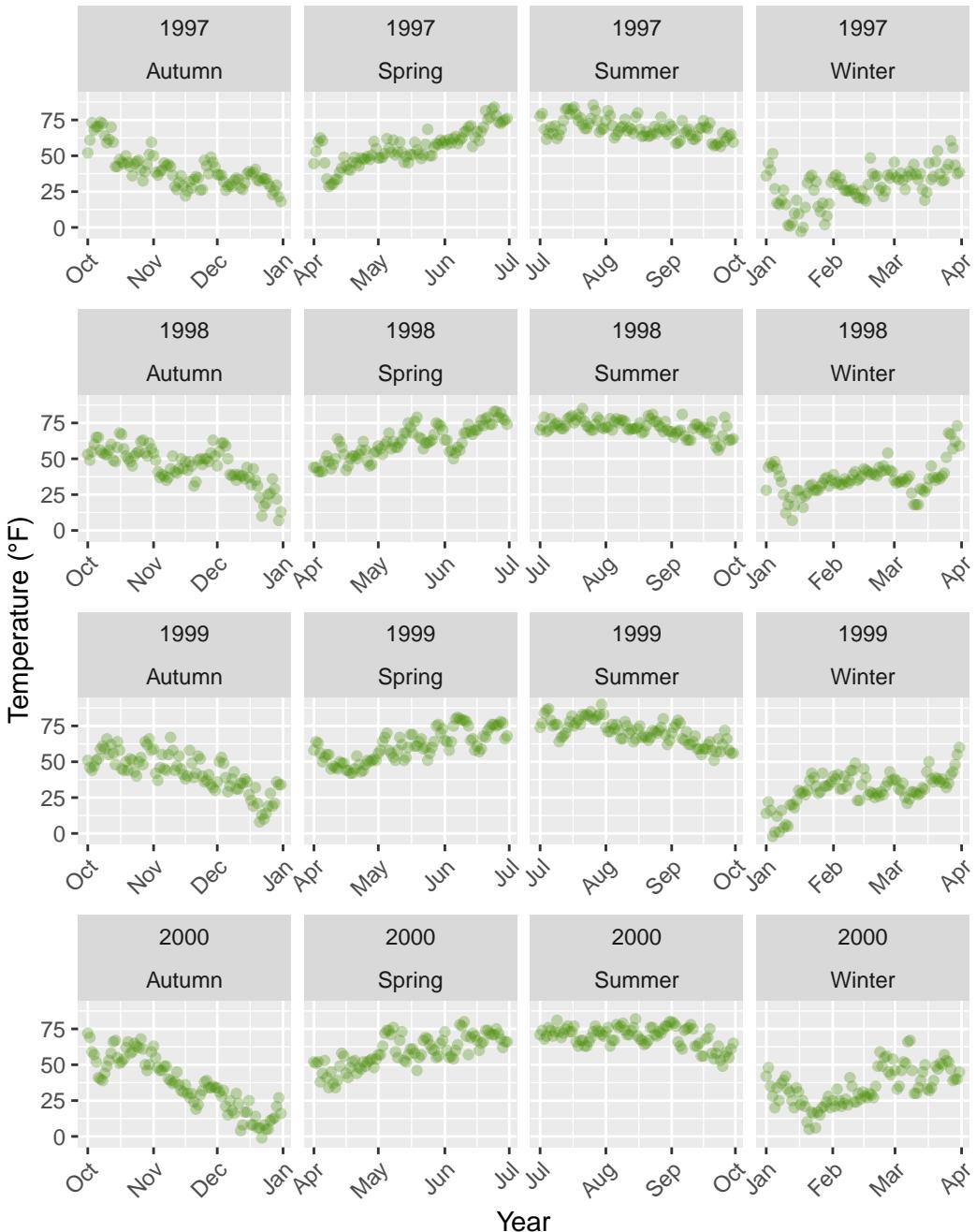


Note that both, x and y axes differ in their range!

9.3.0.1 Use `facet_wrap` with Two Variables

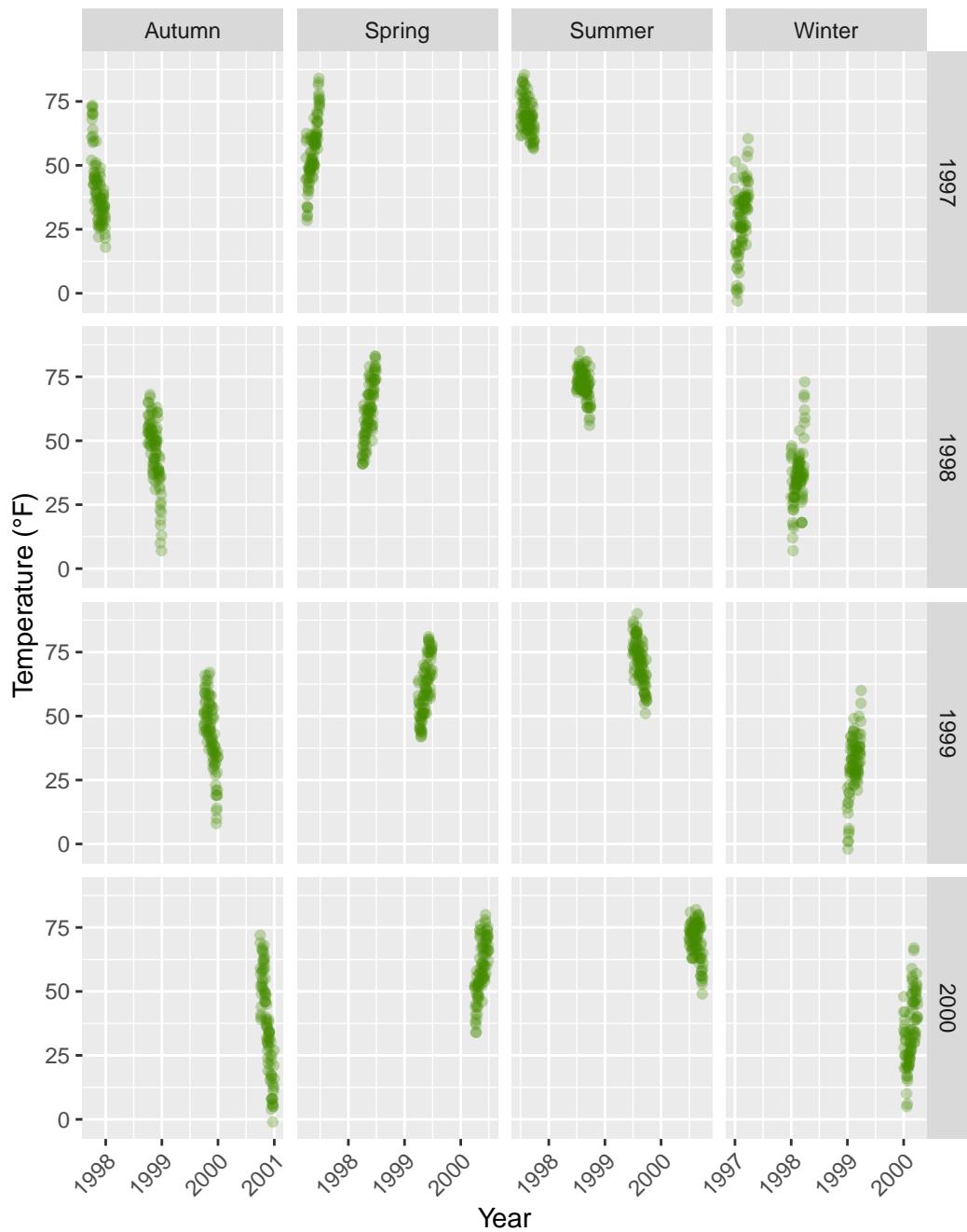
The function `facet_wrap` can also take two variables:

```
g + facet_wrap(year ~ season, nrow = 4, scales = "free_x")
```



When using `facet_wrap` you are still able to control the grid design: you can rearrange the number of plots per row and column and you can also let all axes roam free. In contrast, `facet_grid` will also take a `free` argument but will only let it roam free per column or row:

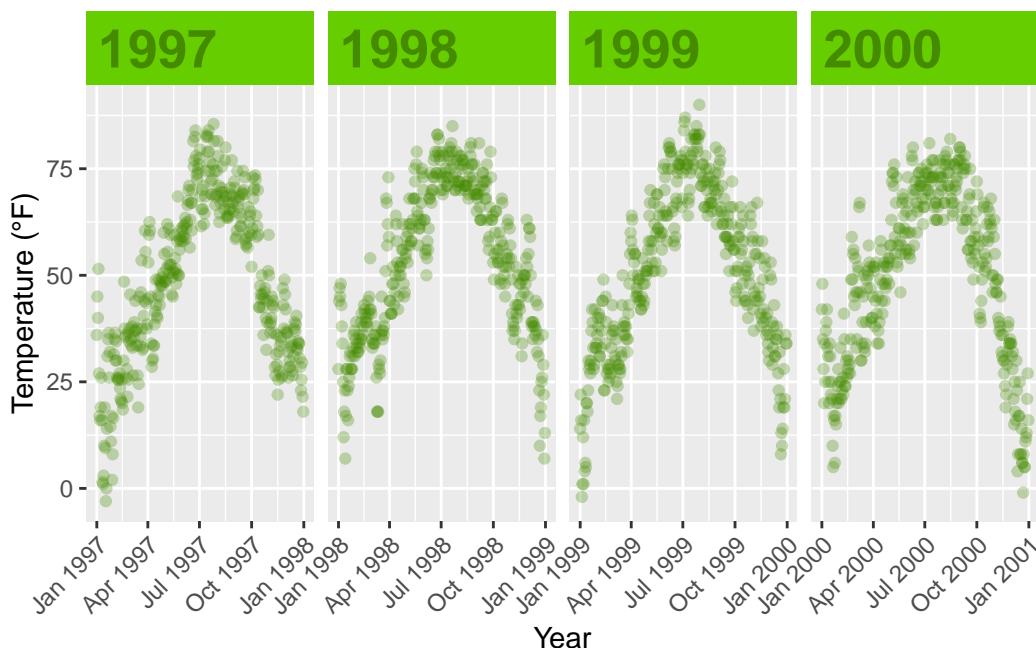
```
g + facet_grid(year ~ season, scales = "free_x")
```



9.4 Modify Style of Strip Texts

By using `theme`, you can modify the appearance of the strip text (i.e. the title for each facet) and the strip text boxes:

```
g + facet_wrap(~ year, nrow = 1, scales = "free_x") +
  theme(strip.text = element_text(face = "bold", color = "chartreuse4",
                                  hjust = 0, size = 20),
        strip.background = element_rect(fill = "chartreuse3", linetype = "dotted"))
```



The following two functions adapted from this answer by Claus Wilke, the author of the `{ggtext}` package, allow to highlight specific labels in combination with `element_textbox()` that is provided by `{ggtext}`.

```
library(ggtext)
library(purrr) ## for %/!

element_textbox_highlight <- function(..., hi.labels = NULL, hi.fill = NULL,
                                    hi.col = NULL, hi.box.col = NULL, hi.family = NULL) {
  structure(
    c(element_textbox(...),
      list(hi.labels = hi.labels, hi.fill = hi.fill, hi.col = hi.col, hi.box.col = hi.box.col
    )),
    class = "element_textbox"
  )
}
```

```

    class = c("element_textbox_highlight", "element_textbox", "element_text", "element")
  )
}

element_grob.element_textbox_highlight <- function(element, label = "", ...) {
  if (label %in% element$hi.labels) {
    element$fill <- element$hi.fill %||% element$fill
    element$colour <- element$hi.col %||% element$colour
    element$box.colour <- element$hi.box.col %||% element$box.colour
    element$family <- element$hi.family %||% element$family
  }
  NextMethod()
}

```

Now you can use it and specify for example all striptexts:

```

g + facet_wrap(year ~ season, nrow = 4, scales = "free_x") +
  theme(
    strip.background = element_blank(),
    strip.text = element_textbox_highlight(
      family = "Playfair Display", size = 12, face = "bold",
      fill = "white", box.color = "chartreuse4", color = "chartreuse4",
      halign = .5, linetype = 1, r = unit(5, "pt"), width = unit(1, "npc"),
      padding = margin(5, 0, 3, 0), margin = margin(0, 1, 3, 1),
      hi.labels = c("1997", "1998", "1999", "2000"),
      hi.fill = "chartreuse4", hi.box.col = "black", hi.col = "white"
    )
  )

```

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x41

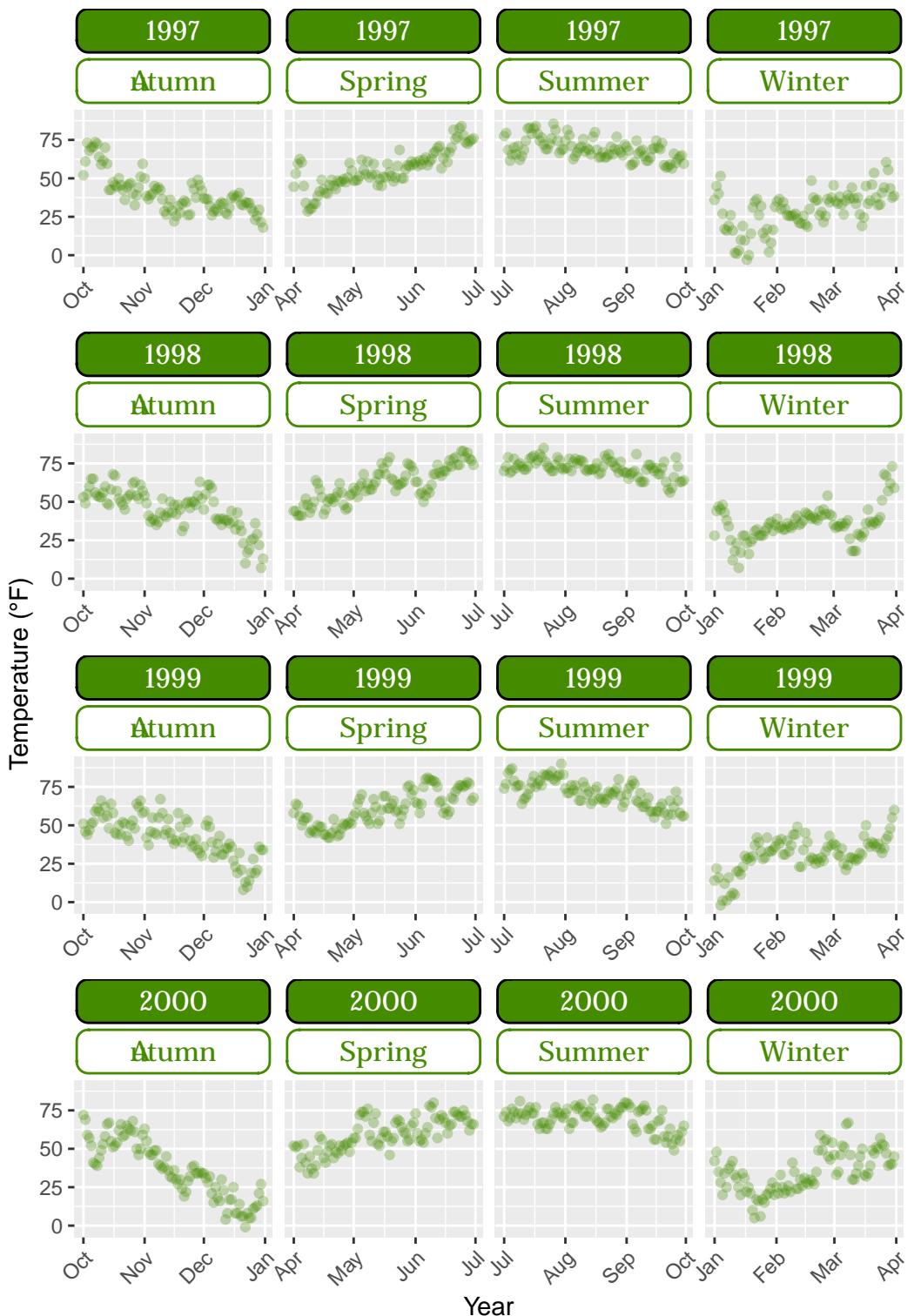
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x41

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x41

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x41

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, :
font width unknown for character 0x41

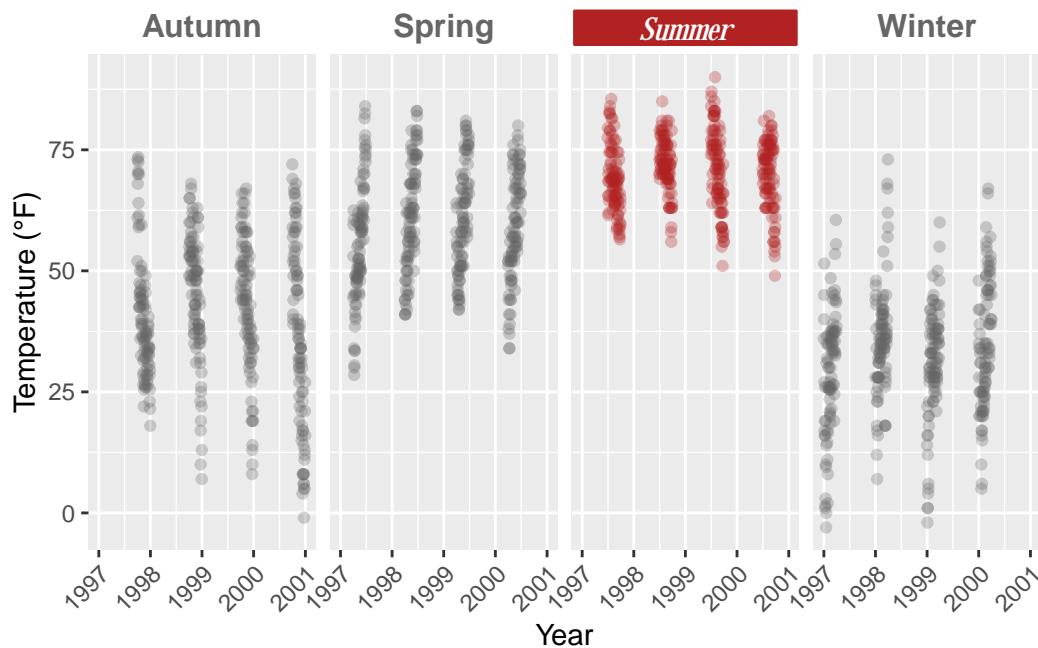
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, :
font width unknown for character 0x41



```

ggplot(chic, aes(x = date, y = temp)) +
  geom_point(aes(color = season == "Summer"), alpha = .3) +
  labs(x = "Year", y = "Temperature (°F)") +
  facet_wrap(~ season, nrow = 1) +
  scale_color_manual(values = c("gray40", "firebrick"), guide = "none") +
  theme(
    axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1),
    strip.background = element_blank(),
    strip.text = element_textbox_highlight(
      size = 12, face = "bold",
      fill = "white", box.color = "white", color = "gray40",
      halign = .5, linetype = 1, r = unit(0, "pt"), width = unit(1, "npc"),
      padding = margin(2, 0, 1, 0), margin = margin(0, 1, 3, 1),
      hi.labels = "Summer", hi.family = "Bangers",
      hi.fill = "firebrick", hi.box.col = "firebrick", hi.col = "white"
    )
  )
)

```



9.5 Create a Panel of Different Plots

There are several ways how plots can be combined. The easiest approach in my opinion is the [{patchwork}](#) package by Thomas Lin Pedersen:

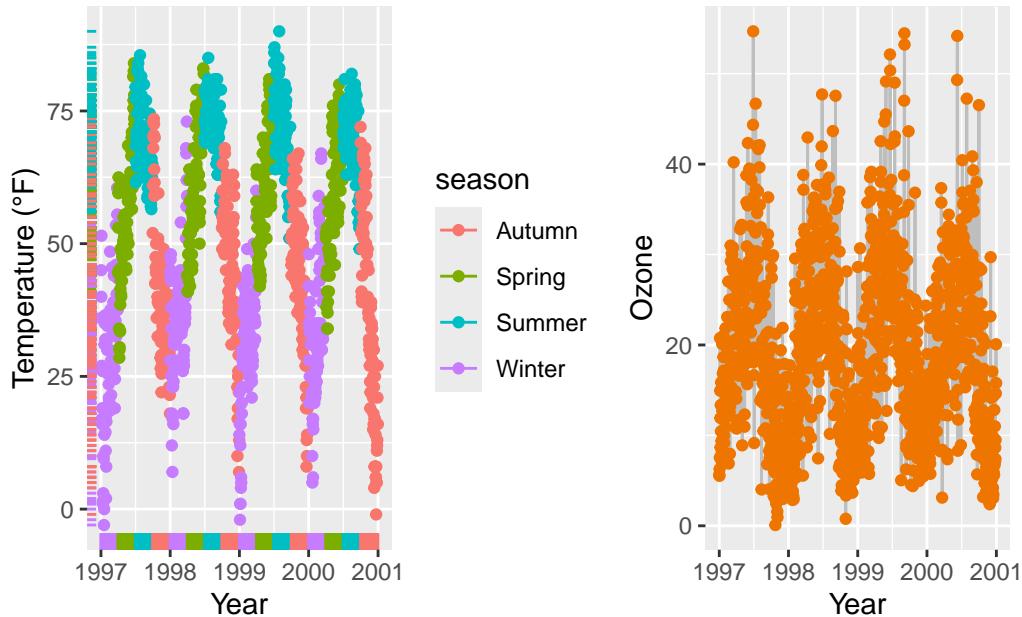
```

p1 <- ggplot(chic, aes(x = date, y = temp,
                        color = season)) +
  geom_point() +
  geom_rug() +
  labs(x = "Year", y = "Temperature (°F)")

p2 <- ggplot(chic, aes(x = date, y = o3)) +
  geom_line(color = "gray") +
  geom_point(color = "darkorange2") +
  labs(x = "Year", y = "Ozone")

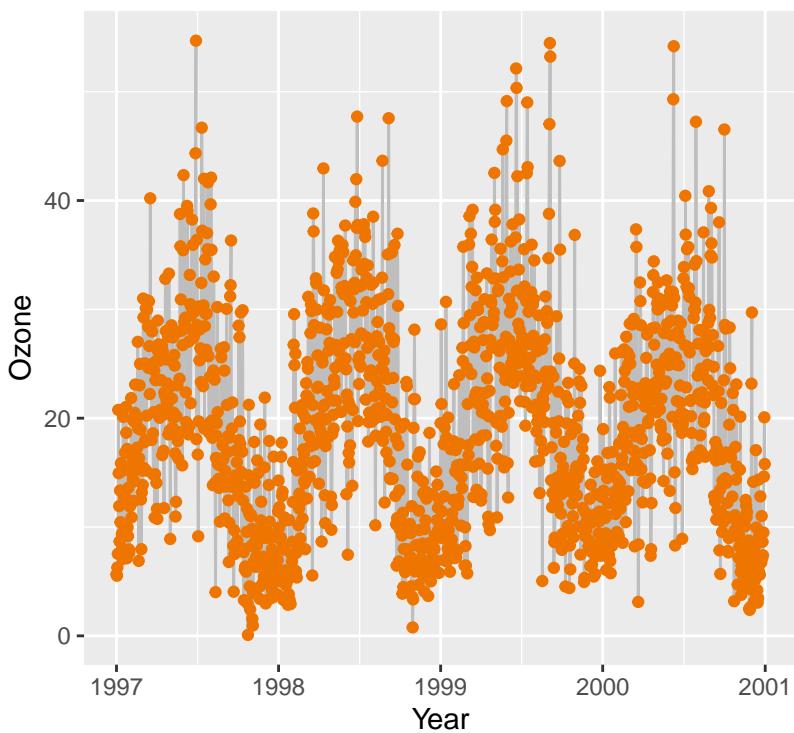
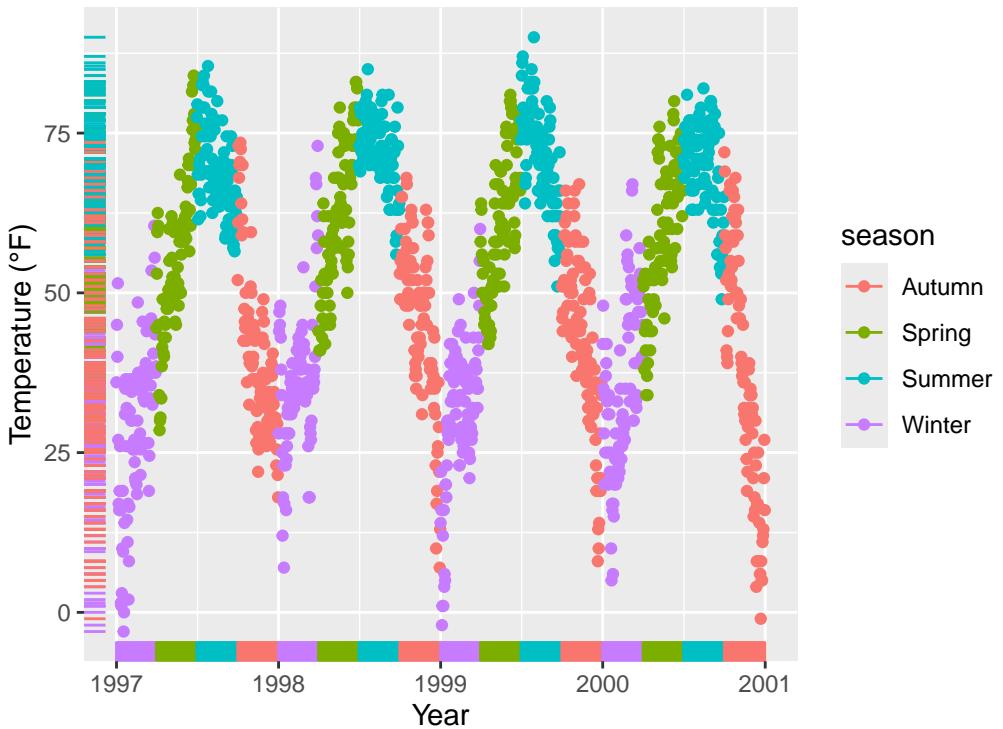
library(patchwork)
p1 + p2

```



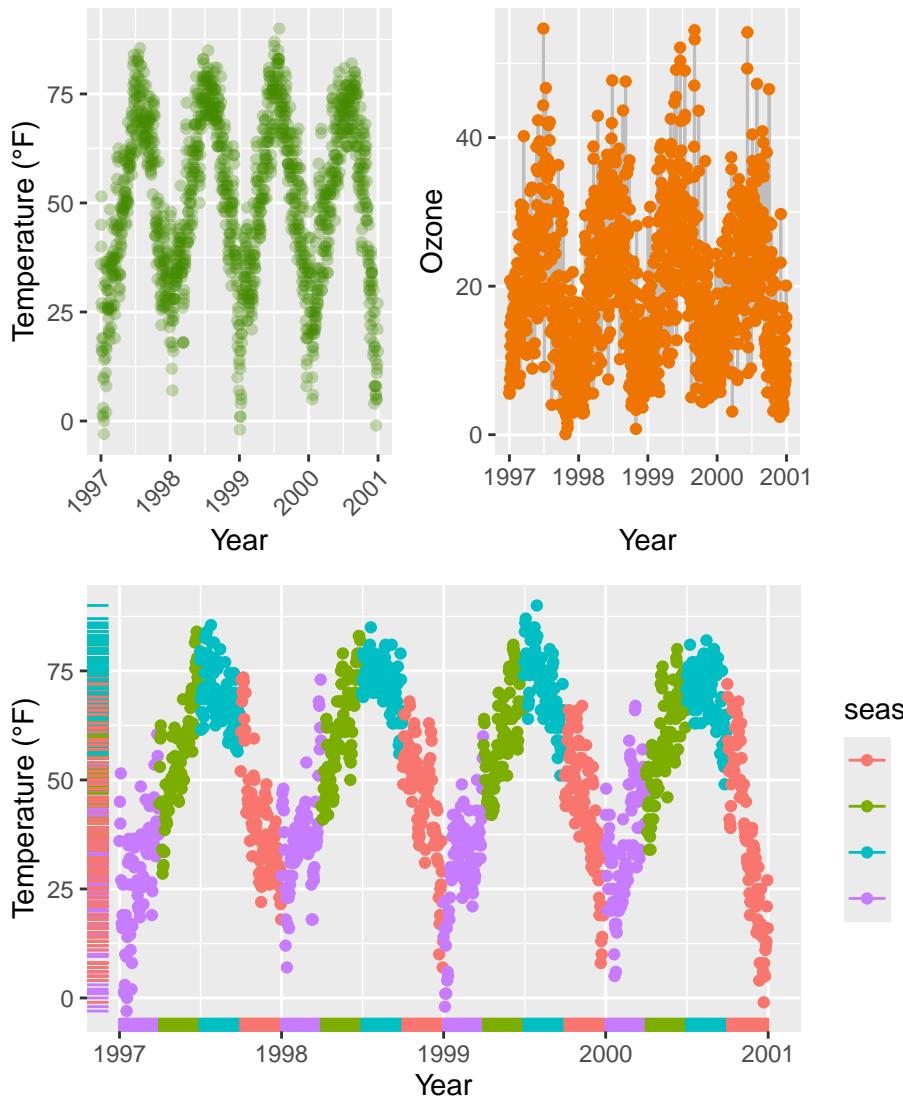
We can change the order by “dividing” both plots (and note the alignment even though one has a legend and one doesn’t!):

```
p1 / p2
```



And also nested plots are possible!

```
(g + p2) / p1
```



(Note the alignment of the plots even though only one plot includes a legend.)

Alternatively, the [cowplot](#) package by Claus Wilke provides the functionality to combine multiple plots (and lots of other good utilities):

```
library(cowplot)
```

```
Attaching package: 'cowplot'
```

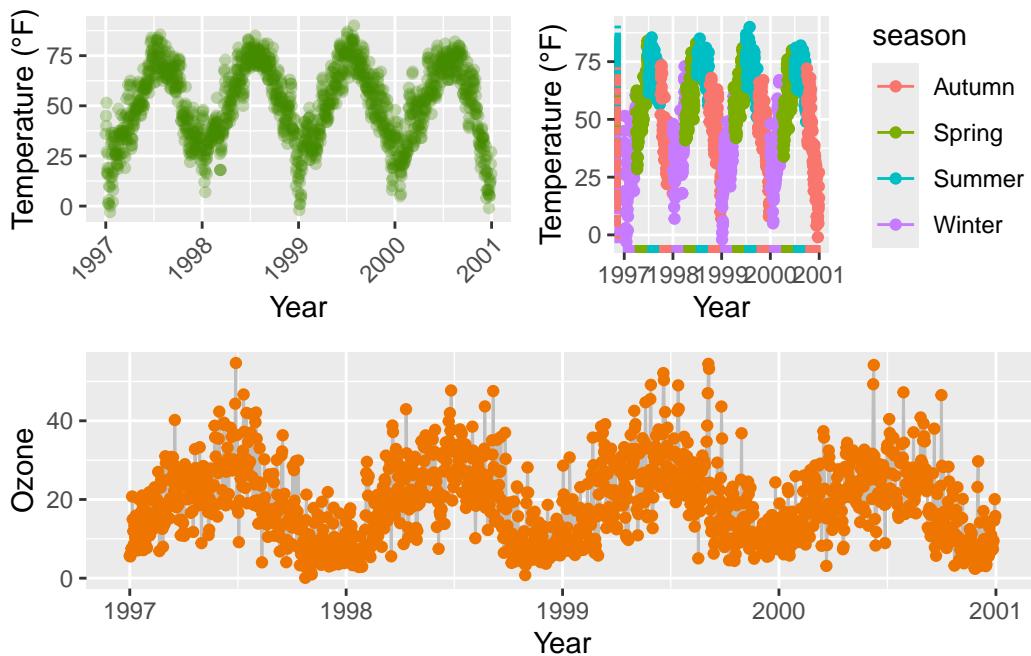
```
The following object is masked from 'package:patchwork':
```

```
align_plots
```

```
The following object is masked from 'package:lubridate':
```

```
stamp
```

```
plot_grid(plot_grid(g, p1), p2, ncol = 1)
```



```
... and so does the {gridExtra} package as well:
```

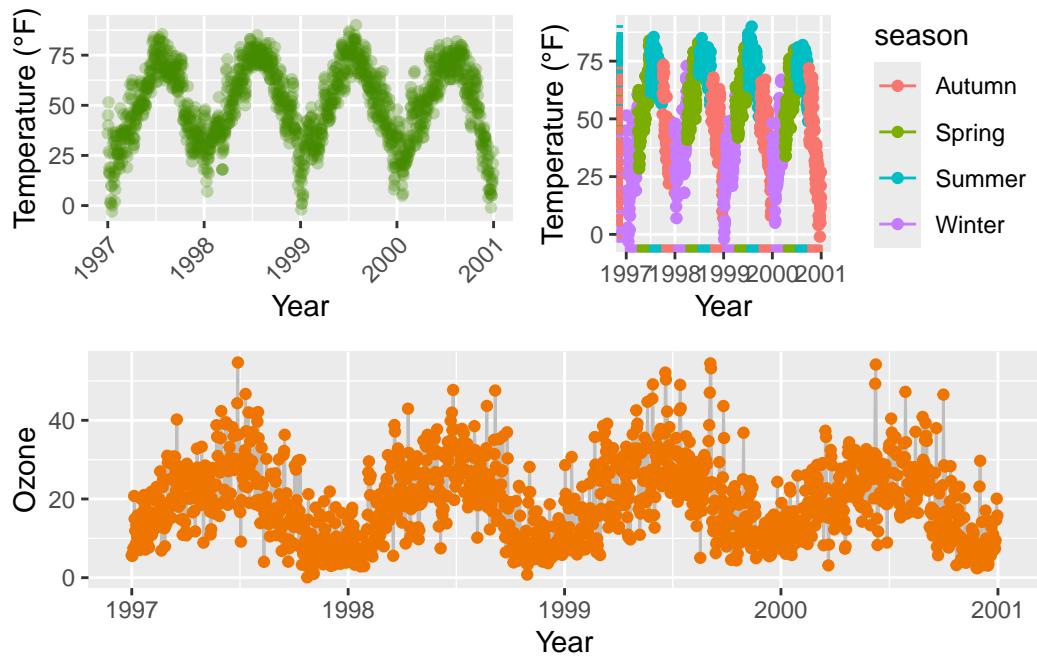
```
library(gridExtra)
```

```
Attaching package: 'gridExtra'
```

```
The following object is masked from 'package:dplyr':
```

```
combine
```

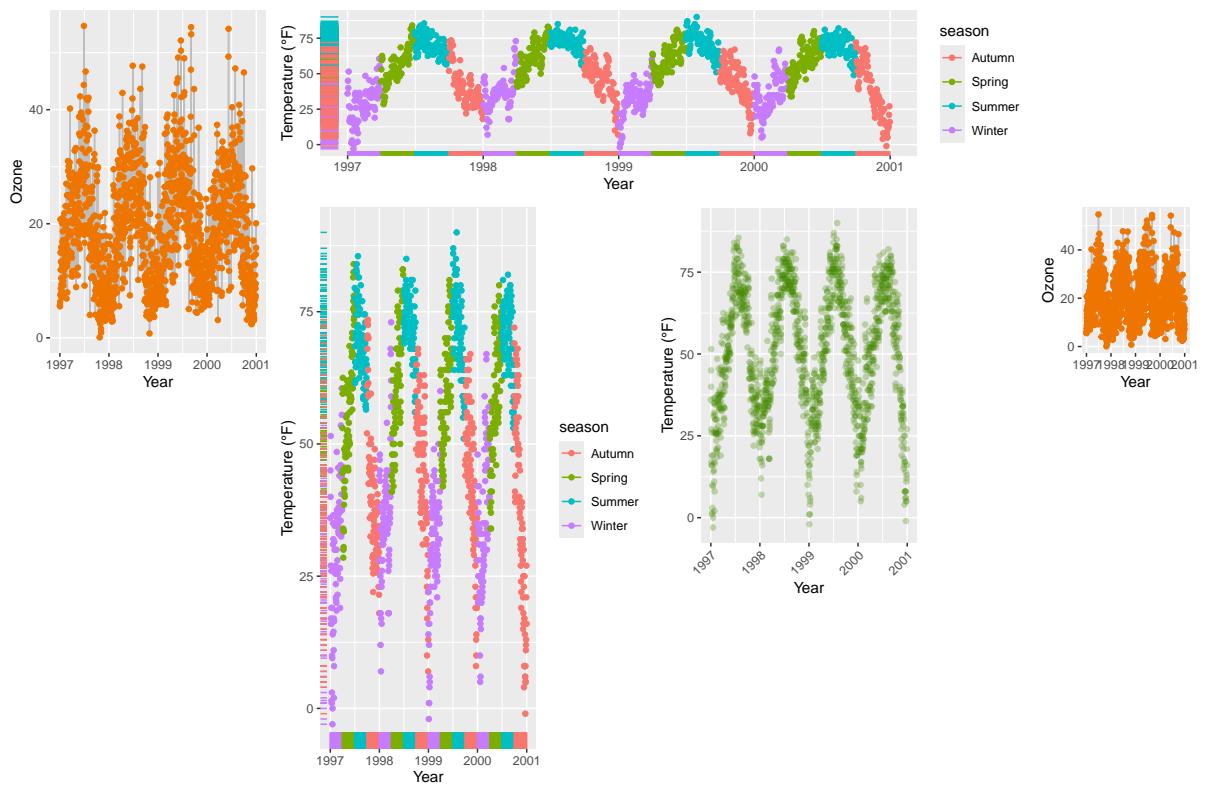
```
grid.arrange(g, p1, p2,
             layout_matrix = rbind(c(1, 2), c(3, 3)))
```



The same idea of defining a layout can be used with `{patchwork}` which allows creating complex compositions:

```
layout <- "
AABB#B#
AACCDDE
##CCDD#
##CC###"
"

p2 + p1 + p1 + g + p2 +
  plot_layout(design = layout)
```



10 Working with Colors

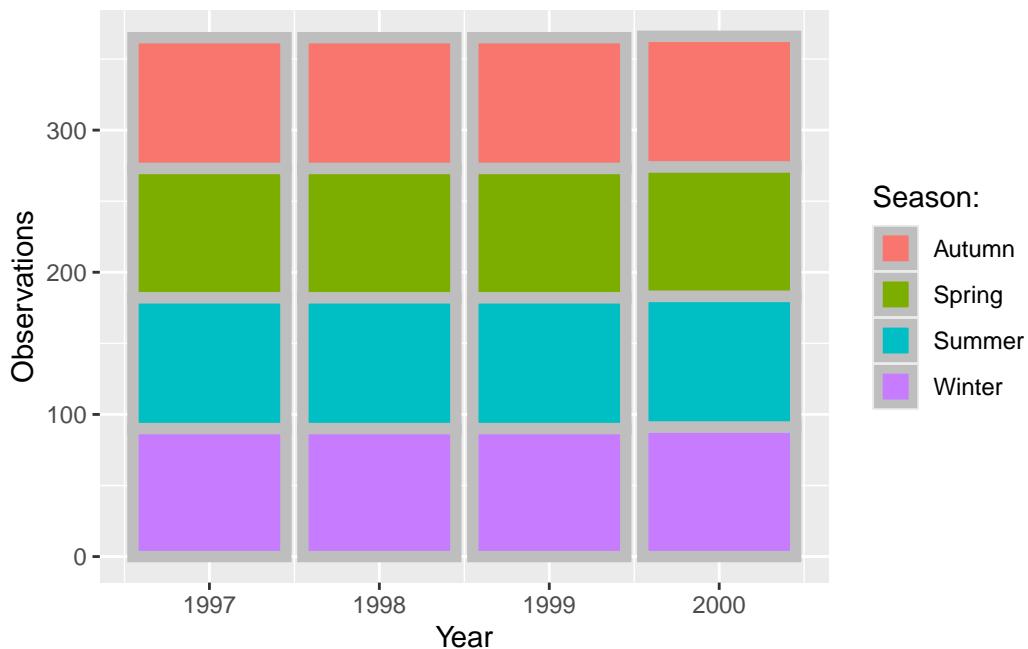
For simple applications working with colors is straightforward in `{ggplot2}`. For a more advanced treatment of the topic you should probably get your hands on [Hadley's book](#) which has nice coverage. Other good sources are the [R Cookbook](#) and the ['color section in the R Graph Gallery](#) by Yan Holtz.

There are two main differences when it comes to colors in `{ggplot2}`. Both arguments, `color` and `fill`, can be

1. specified as single color or
2. assigned to variables.

As you have already seen in the beginning of this tutorial, variables that are *inside* the `aesthetics` are encoded by variables and those that are *outside* are properties that are unrelated to the variables. This complete nonsense plot showing the number of records per year and season illustrates that fact:

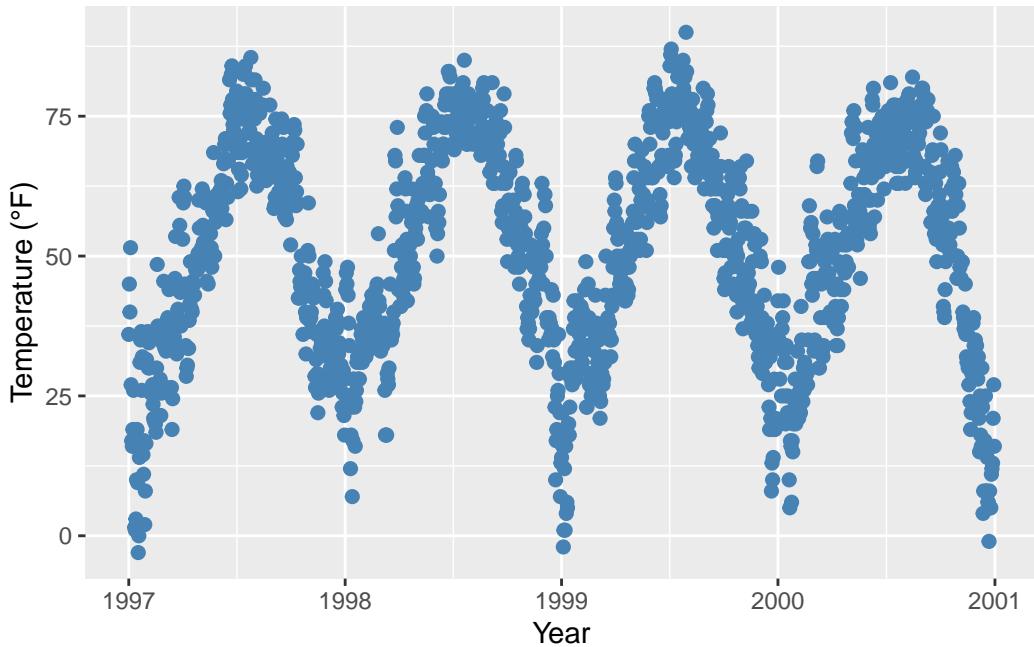
```
ggplot(chic, aes(year)) +  
  geom_bar(aes(fill = season), color = "grey", linewidth = 2) +  
  labs(x = "Year", y = "Observations", fill = "Season:")
```



10.1 Specify Single Colors

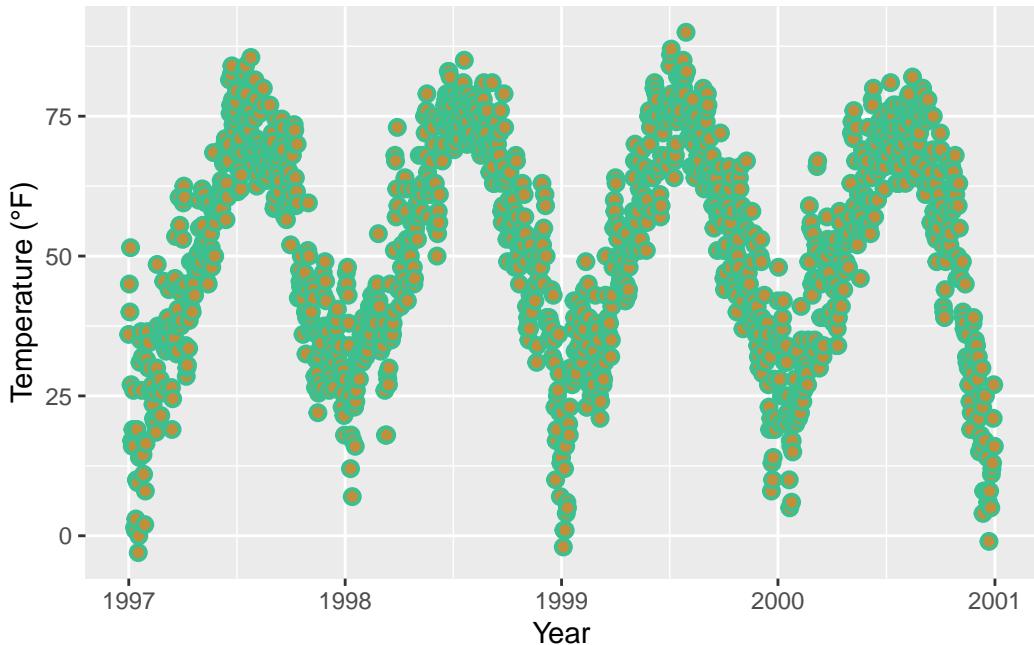
Static, single colors are simple to use. We can specify a single color for a geom:

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "steelblue", size = 2) +  
  labs(x = "Year", y = "Temperature (°F)")
```



... and in case it provides both, a `color` (outline color) and a `fill` (filling color):

```
ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(shape = 21, size = 2, stroke = 1,  
             color = "#3cc08f", fill = "#c08f3c") +  
  labs(x = "Year", y = "Temperature (°F)")
```



Tian Zheng at Columbia has created a useful [PDF of R colors](#). Of course, you can also specify hex color codes (simply as strings as in the example above) as well as RGB or RGBA values (via the `rgb()` function: `rgb(red, green, blue, alpha)`).

10.2 Assign Colors to Variables

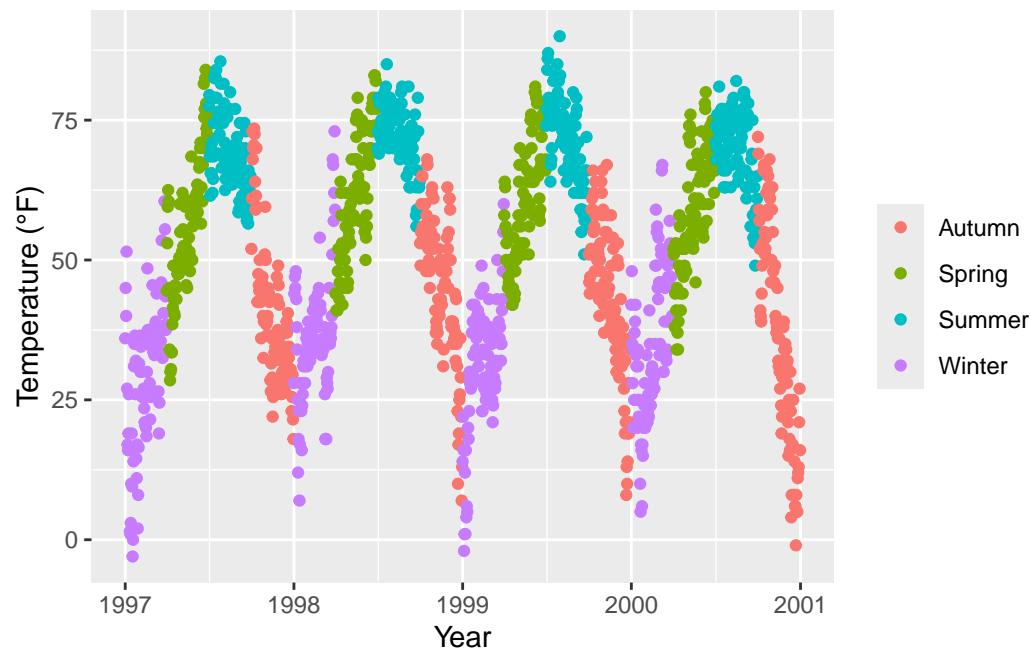
In `{ggplot2}`, colors that are assigned to variables are modified via the `scale_color_*` and the `scale_fill_*` functions. In order to use color with your data, most importantly you need to know if you are dealing with a categorical or continuous variable. The color palette should be chosen depending on type of the variable, with sequential or diverging color palettes being used for continuous variables and qualitative color palettes for categorical variables:

10.3 Qualitative Variables

Qualitative or categorical variables represent types of data which can be divided into groups (*categories*). The variable can be further specified as nominal, ordinal, and binary (dichotomous). Examples of qualitative/categorical variables are:

The default categorical color palette looks like this:

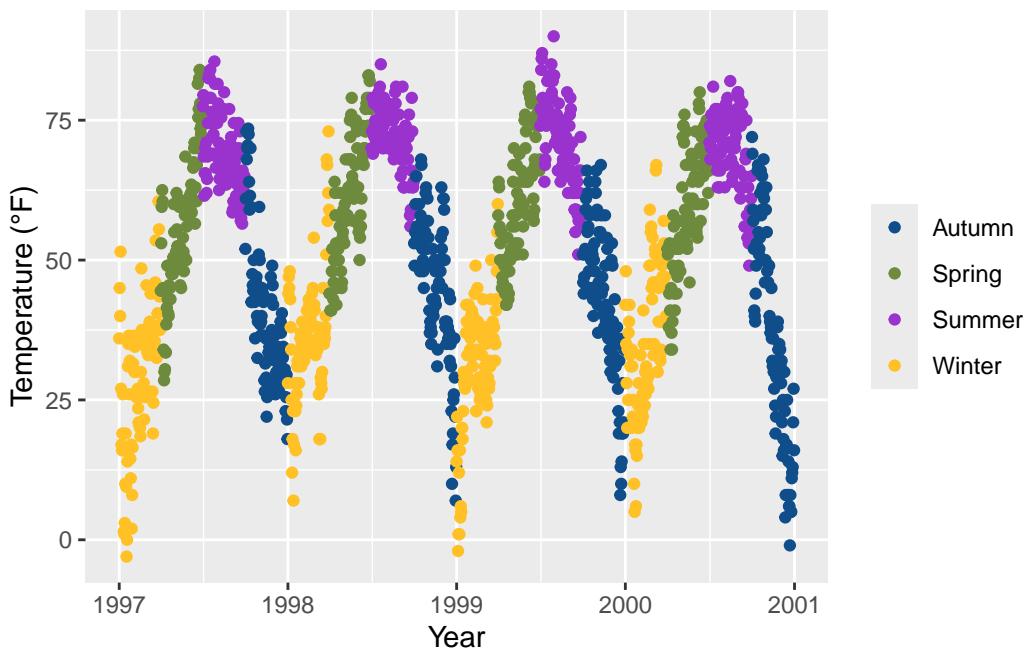
```
(ga <- ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)", color = NULL))
```



10.3.1 Manually Select Qualitative Colors

You can pick your own set of colors and assign them to a categorical variables via the function `scale_*_manual()` (the `*` can be either `color`, `colour`, or `fill`). The number of specified colors has to match the number of categories:

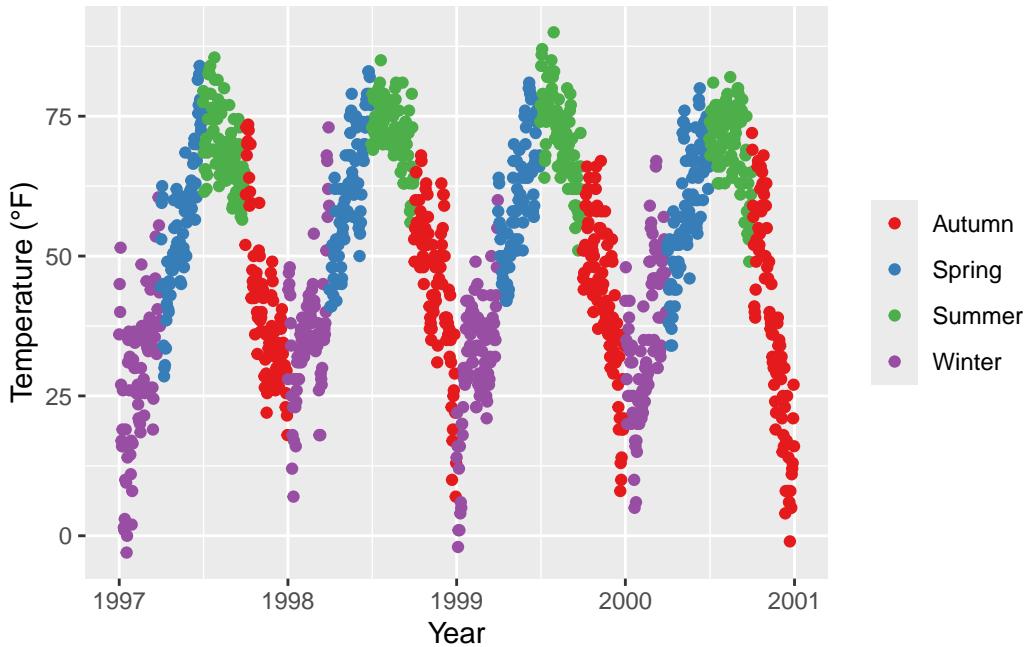
```
ga + scale_color_manual(values = c("dodgerblue4",
                                    "darkolivegreen4",
                                    "darkorchid3",
                                    "goldenrod1"))
```



10.3.2 Use Built-In Qualitative Color Palettes

The [ColorBrewer palettes](#) is a popular online tool for selecting color schemes for maps. The different sets of colors have been designed to produce attractive color schemes of similar appearance ranging from three to twelve. Those palettes are available as built-in functions in the `{ggplot2}` package and can be applied by calling `scale_*_brewer()`:

```
ga + scale_color_brewer(palette = "Set1")
```



You can explore all schemes available via `RColorBrewer::display.brewer.all()`.

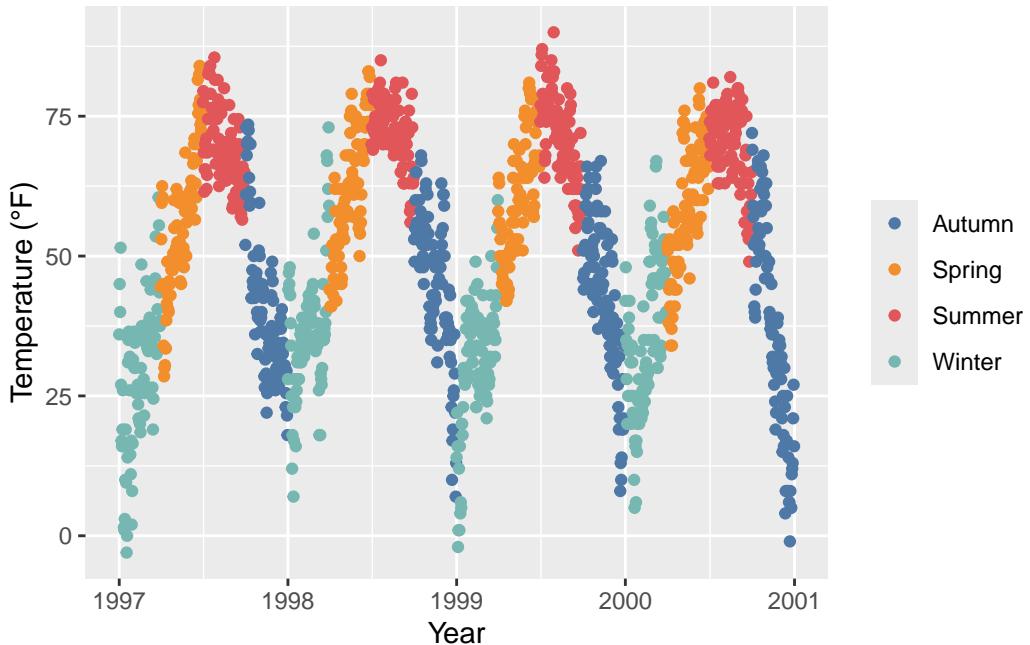
10.3.3 Use Qualitative Color Palettes from Extension Packages

There are many extension packages that provide additional color palettes. Their use differs depending on the way the package is designed. For an extensive overview of color palettes available in R, check the [collection provided by Emil Hvitfeldt](#). One can also use his [{palatteer} package](#), a comprehensive collection of color palettes in R that uses a consistent syntax.

Examples:

The [{ggthemes}](#) package for example lets R users access the Tableau colors. Tableau is a famous visualization software with a [well-known color palette](#).

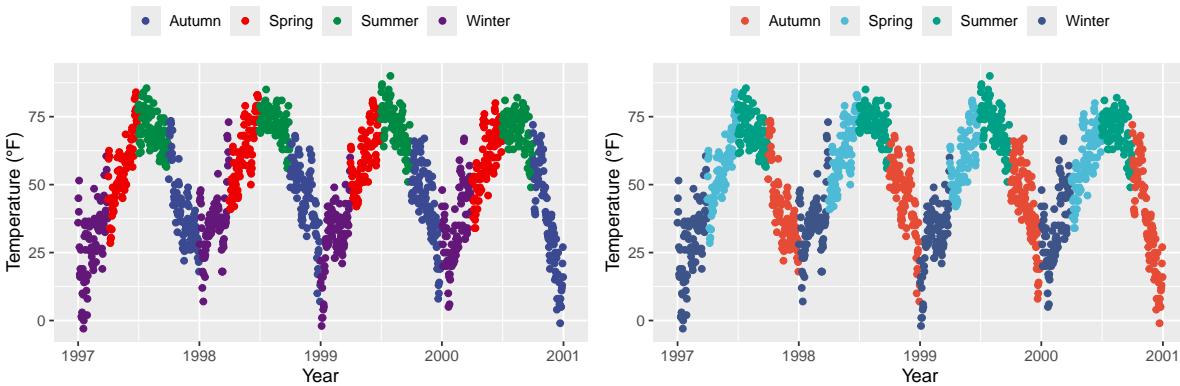
```
library(ggthemes)
ga + scale_color_tableau()
```



The [{ggsci} package](#) provides scientific journal and sci-fi themed color palettes. Want to have a plot with colors that look like being published in *Science* or *Nature*? Here you go!

```
library(ggsci)
g1 <- ga + scale_color_aaas()
g2 <- ga + scale_color_npg()

library(patchwork)
(g1 + g2) * theme(legend.position = "top")
```



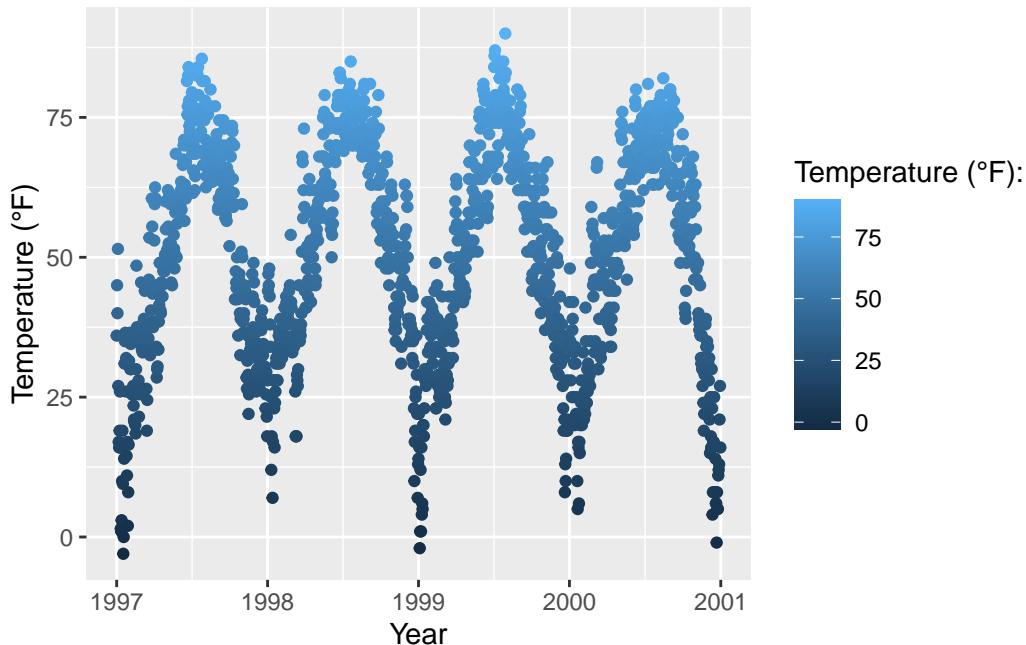
10.4 Quantitative Variables

Quantitative variables represent a measurable quantity and are thus numerical. Quantitative data can be further classified as being either continuous (floating numbers possible) or discrete (integers only):

In our example we will change the variable we want to color to ozone, a continuous variable that is strongly related to temperature (higher temperature = higher ozone). The function `scale_*_gradient()` is a sequential gradient while `scale_*_gradient2()` is diverging.

Here is the default `{ggplot2}` sequential color scheme for continuous variables:

```
gb <- ggplot(chic, aes(x = date, y = temp, color = temp)) +  
  geom_point() +  
  labs(x = "Year", y = "Temperature (°F)", color = "Temperature (°F):")  
  
gb + scale_color_continuous()
```

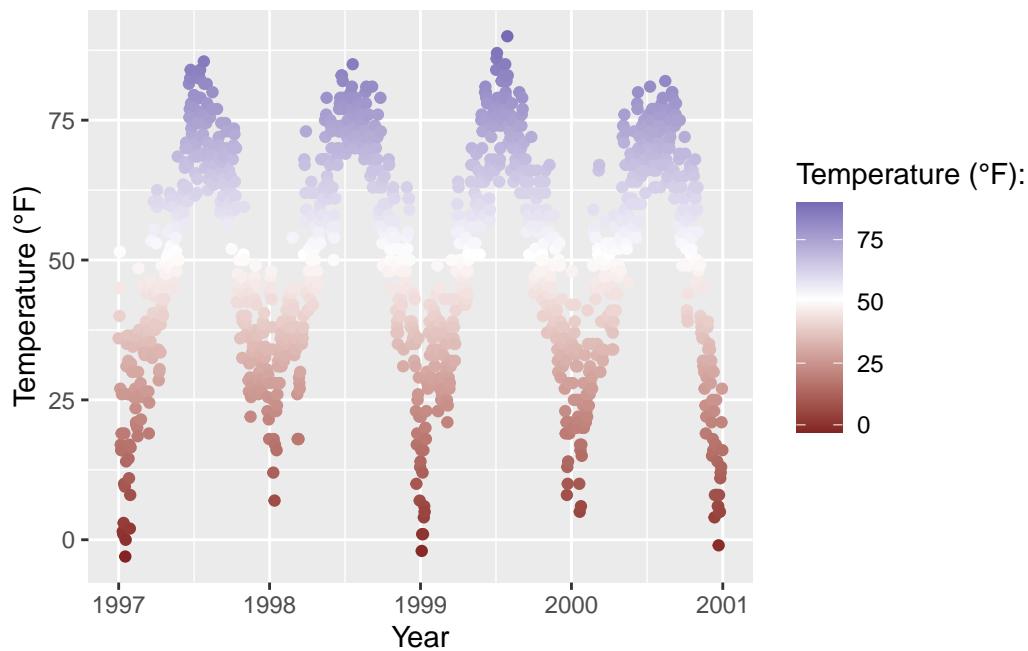


This code produces the same plot:

```
gb + scale_color_gradient()
```

And here is the diverging default color scheme:

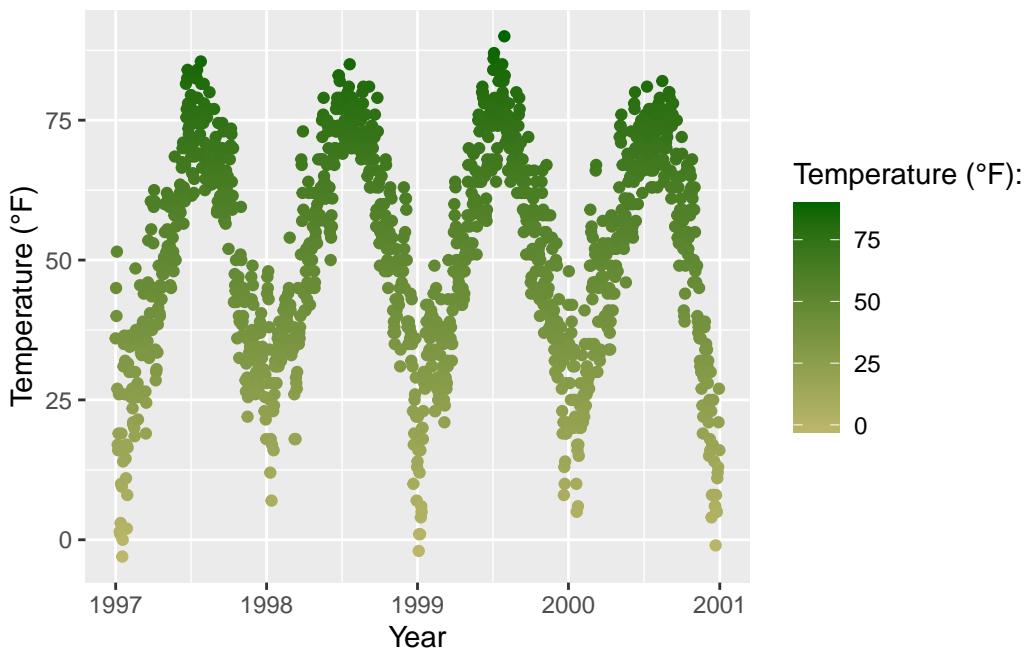
```
mid <- mean(chic$temp) ## midpoint  
gb + scale_color_gradient2(midpoint = mid)
```



10.4.1 Manually Set a Sequential Color Scheme

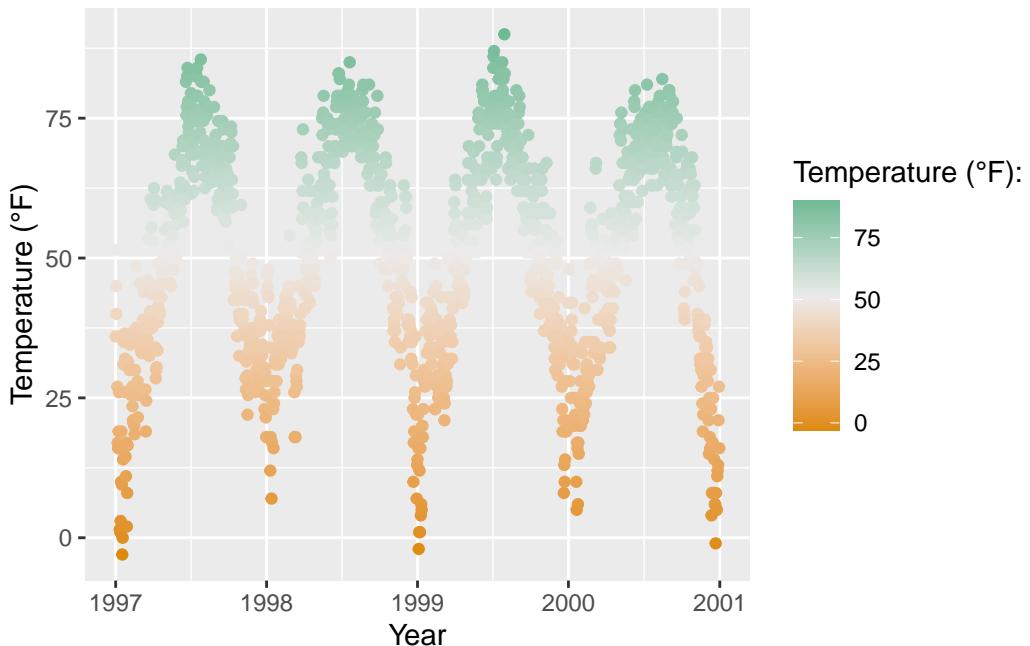
You can manually set gradually changing color palettes for continuous variables via `scale_*_gradient()`:

```
gb + scale_color_gradient(low = "darkkhaki",  
                           high = "darkgreen")
```



Temperature data is normally distributed so how about a diverging color scheme (rather than sequential)... For diverging color you can use the `scale_*_gradient2()` function:

```
gb + scale_color_gradient2(midpoint = mid, low = "#dd8a0b",
                           mid = "grey92", high = "#32a676")
```



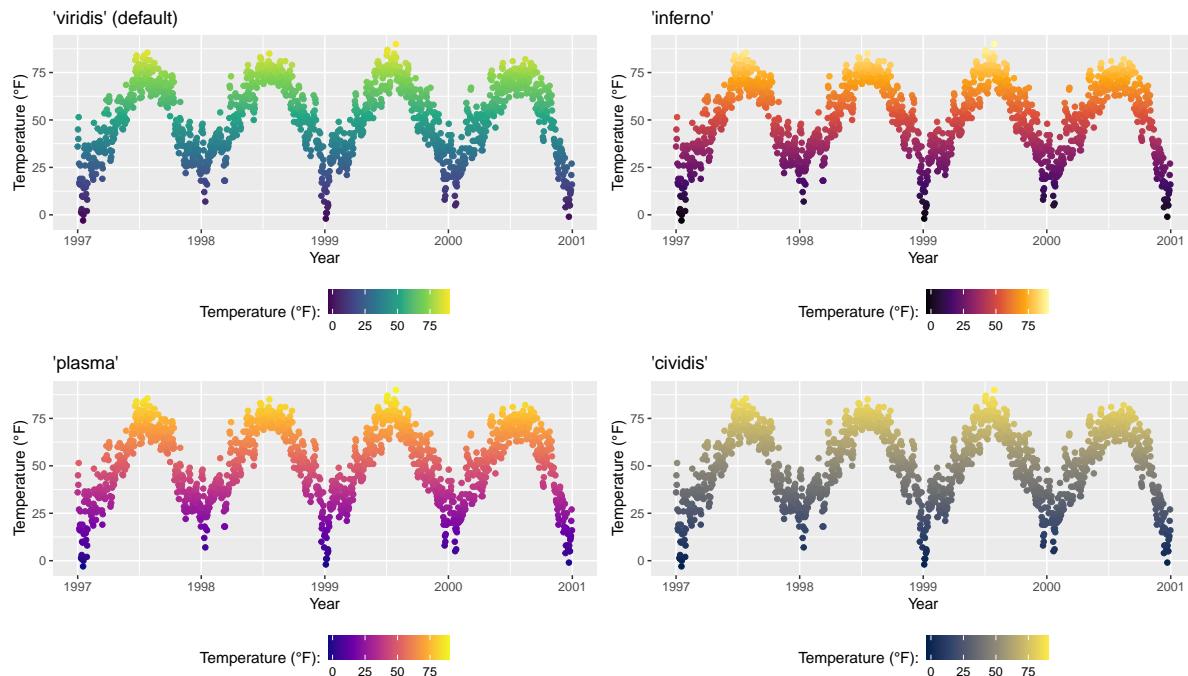
10.4.2 The Beautiful Viridis Color Palette

The [viridis color palettes](#) do not only make your plots look pretty and good to perceive but also easier to read by those with colorblindness and print well in gray scale. You can test how your plots might appear under various form of colorblindness using [{dichromat}](#) package.

And they also come now shipped with [{ggplot2}](#)! The following multi-panel plot illustrates three out of the four viridis palettes:

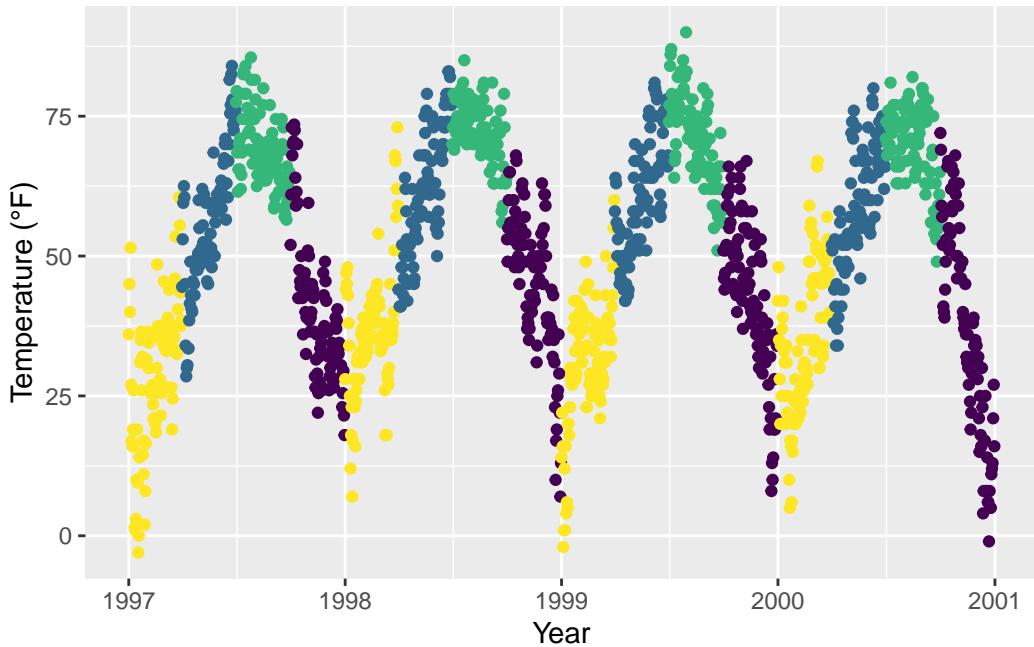
```
p1 <- gb + scale_color_viridis_c() + ggtitle("'viridis' (default)")
p2 <- gb + scale_color_viridis_c(option = "inferno") + ggtitle("'inferno'")
p3 <- gb + scale_color_viridis_c(option = "plasma") + ggtitle("'plasma'")
p4 <- gb + scale_color_viridis_c(option = "cividis") + ggtitle("'cividis'")

library(patchwork)
(p1 + p2 + p3 + p4) * theme(legend.position = "bottom")
```



It is also possible to use the viridis color palettes for discrete variables:

```
ga + scale_color_viridis_d(guide = "none")
```



10.4.3 Use Quantitative Color Palettes from Extension Packages

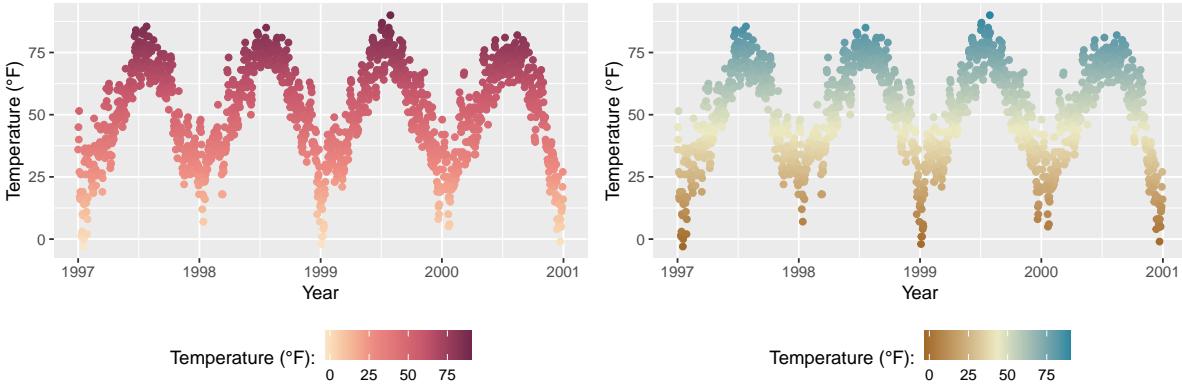
The many extension packages provide not only additional categorical color palettes but also sequential, diverging and even cyclical palettes. Again, I point you to the great [collection provided by Emil Hvitfeldt](#) for an overview.

Examples:

The [{rcartocolor}](#) package ports the beautiful [CARTOcolors](#) to [{ggplot2}](#) and contains several of my most-used palettes:

```
library(rcartocolor)
g1 <- gb + scale_color_carto_c(palette = "BurgYl")
g2 <- gb + scale_color_carto_c(palette = "Earth")

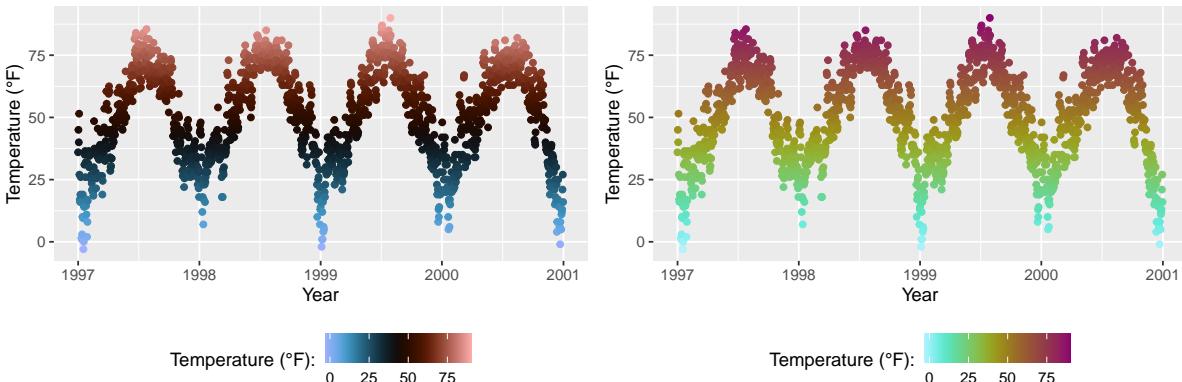
(g1 + g2) * theme(legend.position = "bottom")
```



The [{scico}](#) package provides access to the [color palettes developed by Fabio Crameri](#). These color palettes are not only beautiful and often unusual but also a good choice since they have been developed to be perceptually uniform and ordered. In addition, they work for people with color vision deficiency and in grayscale:

```
library(scico)
g1 <- gg + scale_color_scico(palette = "berlin")
g2 <- gg + scale_color_scico(palette = "hawaii", direction = -1)

(g1 + g2) * theme(legend.position = "bottom")
```



10.4.3.1 Modify Color Palettes Afterwards

Since the release of `ggplot2` 3.0.0, one can modify layer aesthetics after they have been mapped to the data. Or as the `{ggplot2}` phrases it: “Use `after_scale()` to flag evaluation of mapping for after data has been scaled.”

So why not use the modified colors in the first place? Since `{ggplot2}` can only handle one `color` and one `fill` scale, this is an interesting functionality. Look closer at the following example where we use `clr_negate()` from the [{prismatic}](#) package:

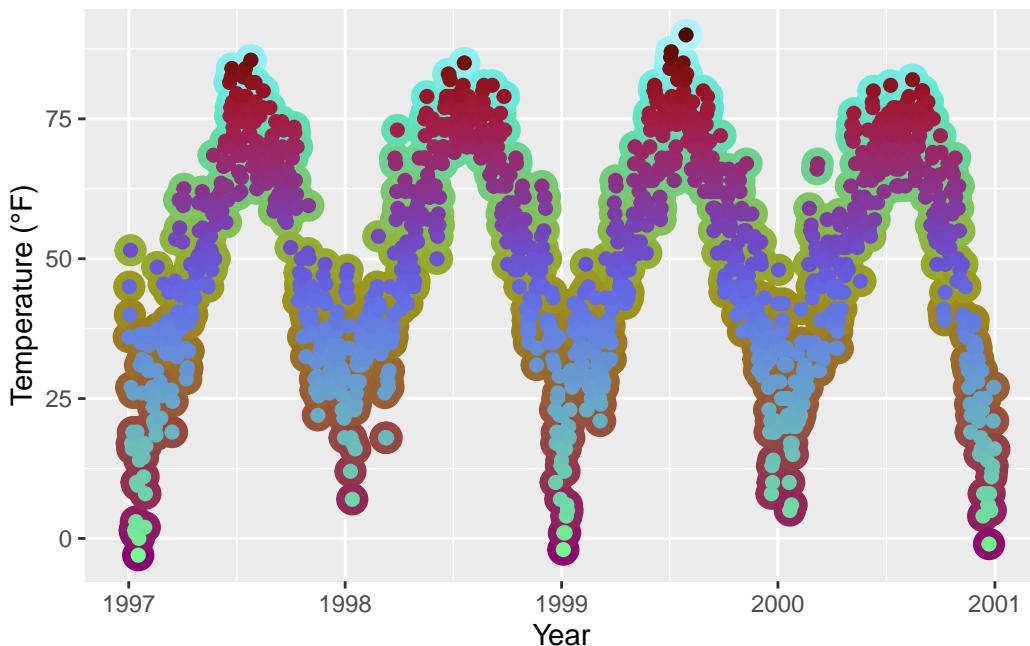
```

library(prismatic)

ggplot(chic, aes(date, temp, color = temp)) +
  geom_point(size = 5) +
  geom_point(aes(color = temp,
                 color = after_scale(clr_negate(color))),
             size = 2) +
  scale_color_scico(palette = "hawaii", guide = "none") +
  labs(x = "Year", y = "Temperature (°F)")

```

Warning: Duplicated aesthetics after name standardisation: colour



Changing the color scheme afterwards is especially fun with functions from the `{prismatic}` package, namely `clr_negate()`, `clr_lighten()`, `clr_darken()` and `clr_desaturate()`. You can even combine those functions. Here, we plot a box plot that has both arguments, `color` and `fill`:

```

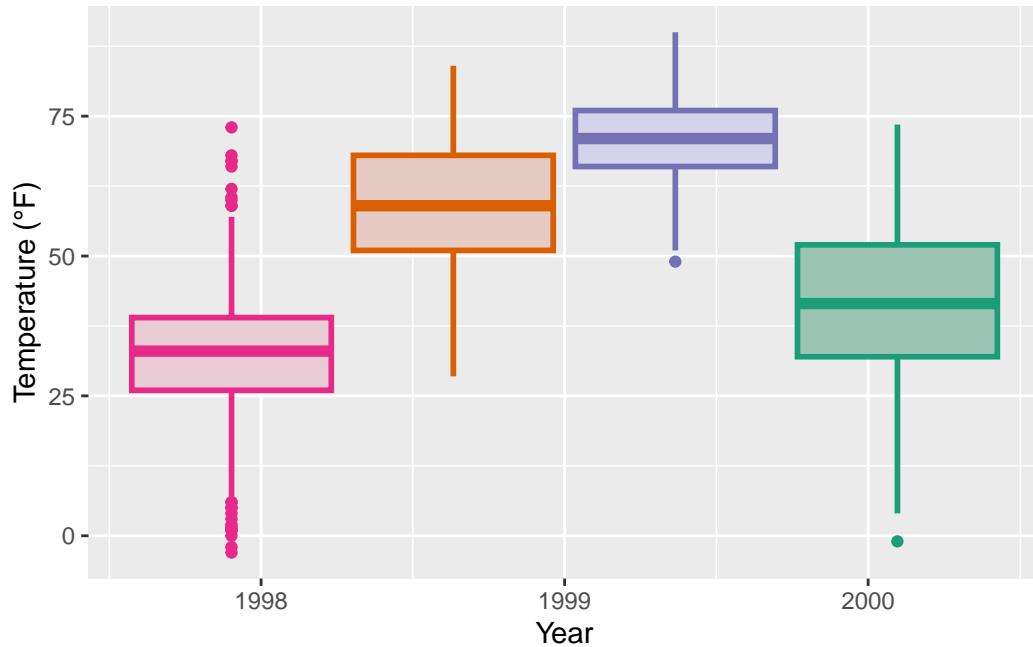
library(prismatic)

ggplot(chic, aes(date, temp)) +
  geom_boxplot(
    aes(color = season,
        fill = after_scale(clr_desaturate(clr_lighten(color, .6), .6))),
```

```

    linewidth = 1
) +
scale_color_brewer(palette = "Dark2", guide = "none") +
labs(x = "Year", y = "Temperature (°F)")

```



Note that you need to specify the `color` and/or `fill` in the `aes()` of the respective `geom_*`() or `stat_*`() to make `after_scale()` work.

This seems a bit complicated for now—one could simply use the `color` and `fill` scales for both. Yes, that is true but think about use cases where you need several `color` and/or `fill` scales. In such a case, it would be senseless to occupy the `fill` scale with a slightly darker version of the palette used for `color`.

11 Working with Themes

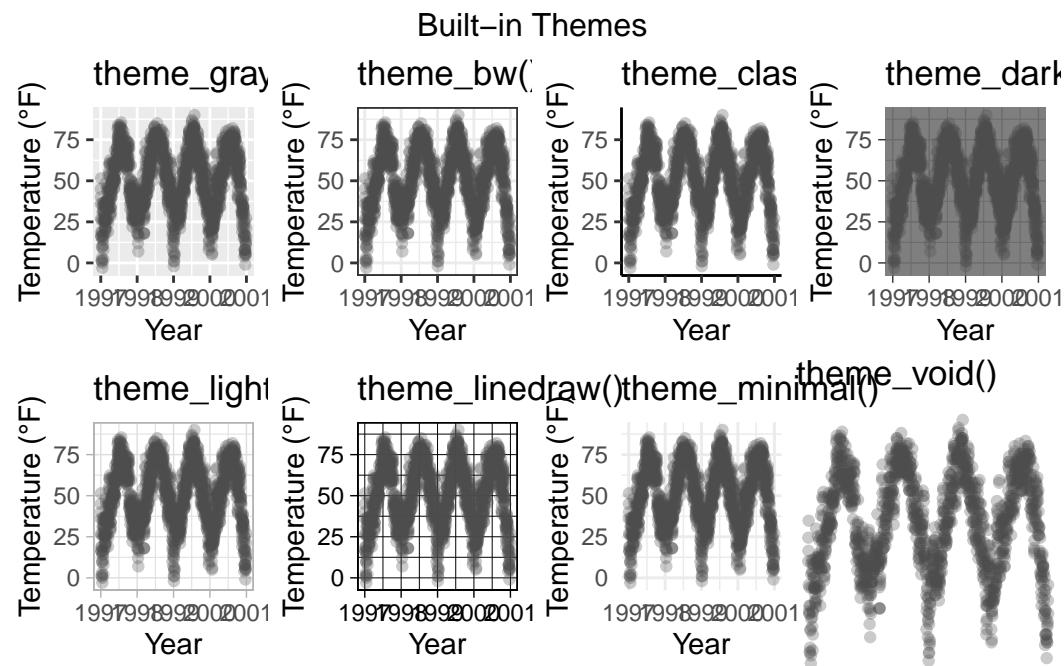
11.1 Change the Overall Plotting Style

You can change the entire look of the plots by using themes. `{ggplot2}` comes with eight built-in themes:

```
Attaching package: 'gridExtra'
```

```
The following object is masked from 'package:dplyr':
```

```
combine
```



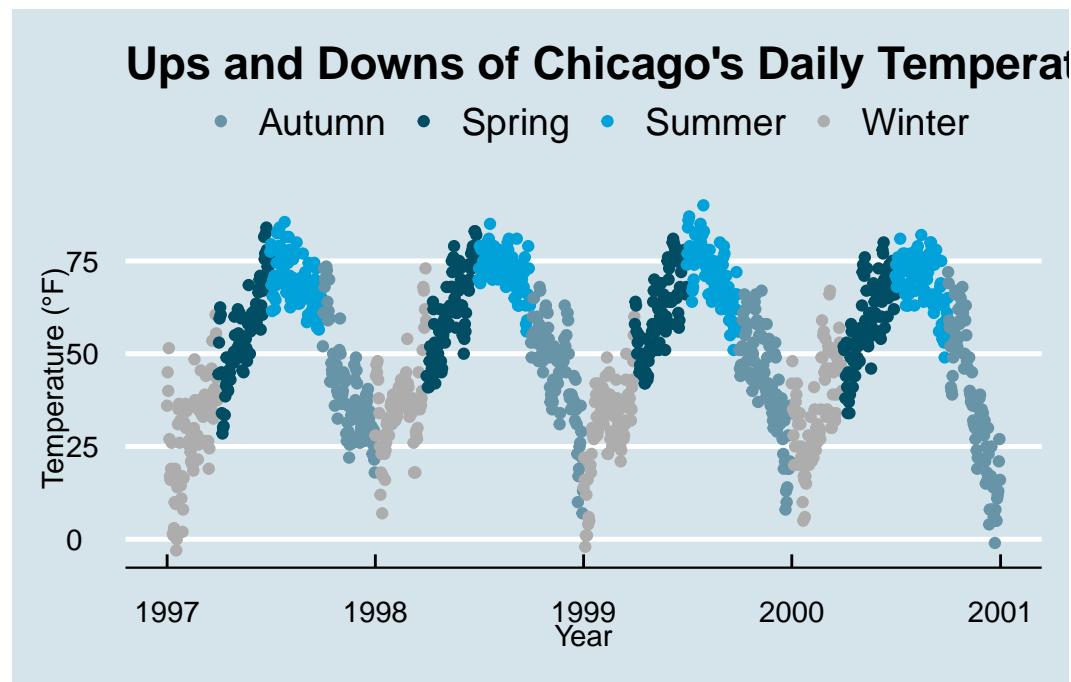
There are several packages that provide additional themes, some even with different default color palettes. As an example, Jeffrey Arnold has put together the library `{ggthemes}` with

several custom themes imitating popular designs. For a list you can visit the [ggthemes package site](#). Without any coding you can just adapt several styles, some of them well known for their style and aesthetics.

Here is an example copying the [plotting style](#) in the [The Economist](#) magazine by using `theme_economist()` and `scale_color_economist()`:

```
library(ggthemes)

ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() +
  labs(x = "Year", y = "Temperature (°F)") +
  ggtitle("Ups and Downs of Chicago's Daily Temperatures") +
  theme_economist() +
  scale_color_economist(name = NULL)
```



Another example is the plotting style of Tufte, a minimal ink theme based on [Edward Tufte's](#) book [The Visual Display of Quantitative Information](#). This is the book that popularized [Minard's chart depicting Napoleon's march on Russia](#) as one of the **best statistical drawings ever created**. Tufte's plots became famous due to the purism in their style. But see yourself:

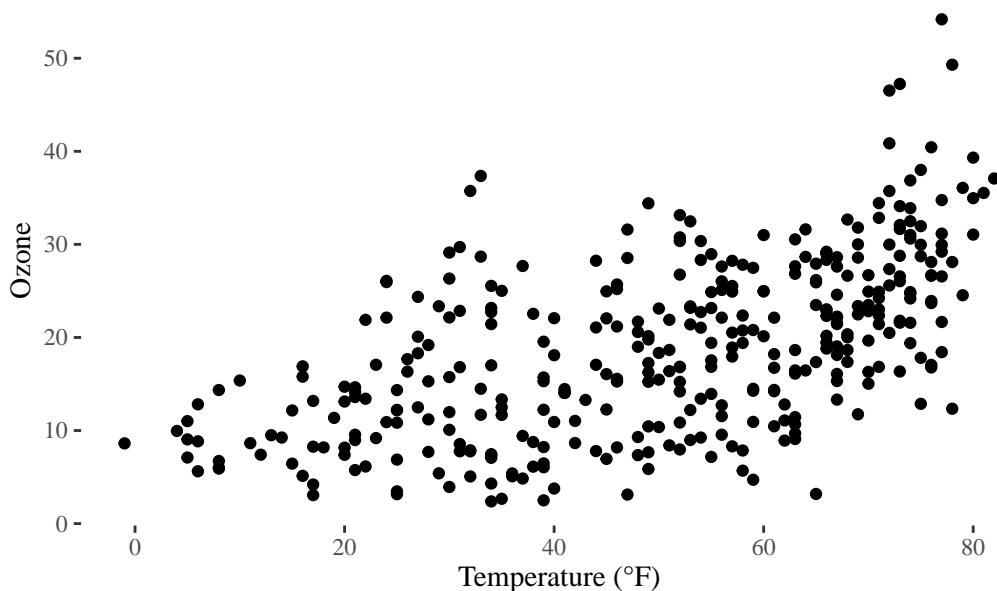
```

library(dplyr)
chic_2000 <- filter(chic, year == 2000)

ggplot(chic_2000, aes(x = temp, y = o3)) +
  geom_point() +
  labs(x = "Temperature (°F)", y = "Ozone") +
  ggtitle("Temperature and Ozone Levels During the Year 2000 in Chicago") +
  theme_tufte()

```

Temperature and Ozone Levels During the Year 2000 in Chicago



I reduced the number of data points here simply to fit it Tufte's minimalism style. If you like the way of plotting have a look on [this blog entry](#) creating several Tufte plots in R.

Another neat packages with modern themes and a preset of non-default fonts is the `{hrbrthemes}` package by Bob Rudis with several light but also dark themes:

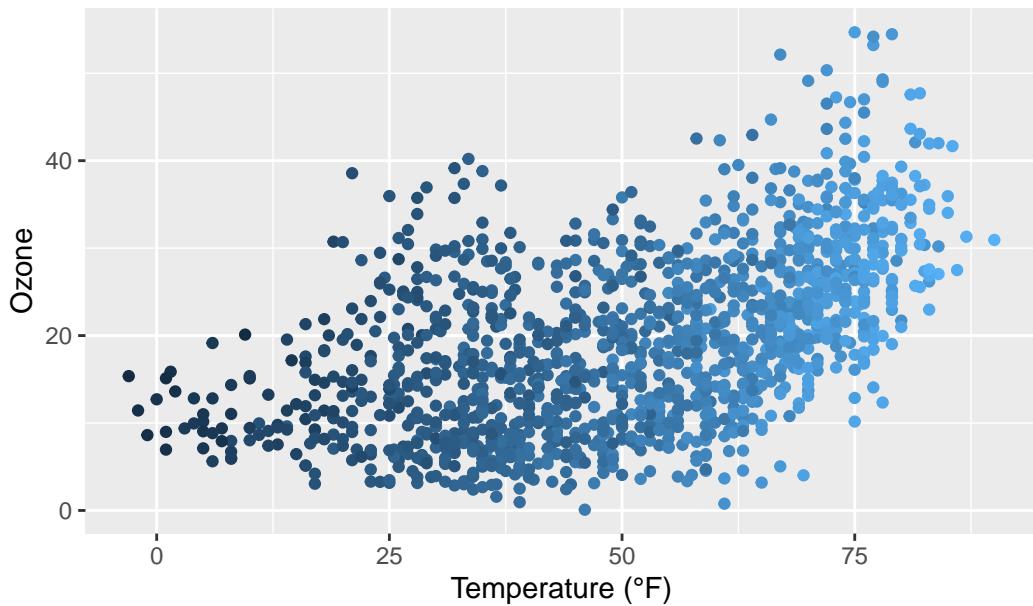
```

library(hrbrthemes)

ggplot(chic, aes(x = temp, y = o3)) +
  geom_point(aes(color = dewpoint), show.legend = FALSE) +
  labs(x = "Temperature (°F)", y = "Ozone") +
  ggtitle("Temperature and Ozone Levels in Chicago")

```

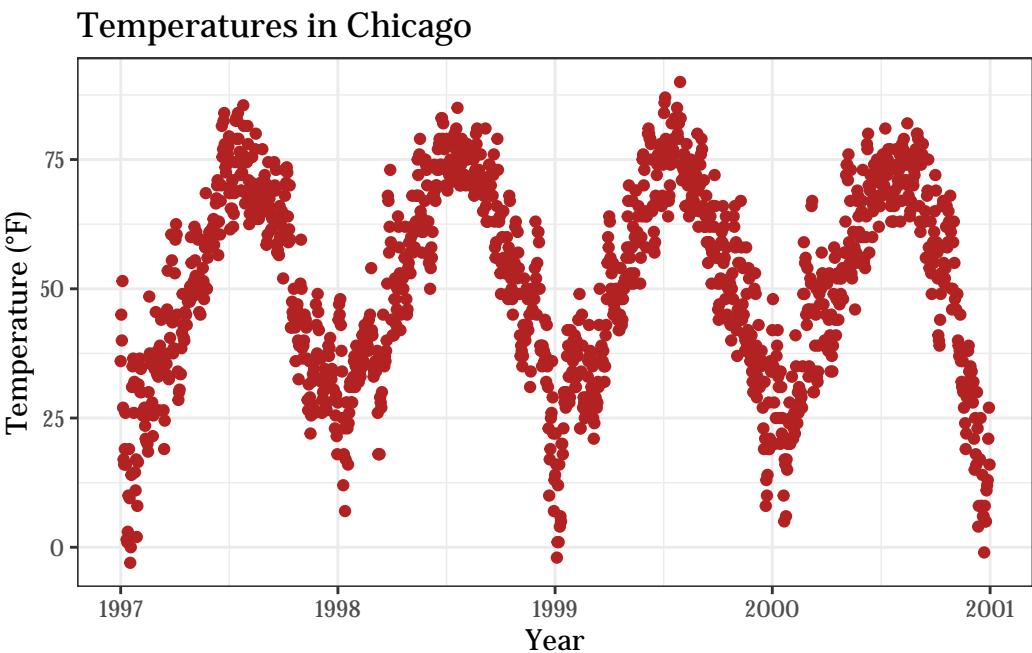
Temperature and Ozone Levels in Chicago



11.2 Change the Font of All Text Elements

It is incredibly easy to change the settings of all the text elements at once. All themes come with an argument called `base_family`:

```
g <- ggplot(chic, aes(x = date, y = temp)) +  
  geom_point(color = "firebrick") +  
  labs(x = "Year", y = "Temperature (°F)",  
       title = "Temperatures in Chicago")  
  
g + theme_bw(base_family = "Playfair Display")
```



11.3 Change the Size of All Text Elements

The `theme_*`() functions also come with several other `base_*` arguments. If you have a closer look at the default theme (see chapter “Create and Use Your Custom Theme” below) you will notice that the sizes of all the elements are relative (`rel()`) to the `base_size`. As a result, you can simply change the `base_size` if you want to increase readability of your plots:

```
g + theme_bw(base_size = 30, base_family = "Roboto Condensed")
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics
unknown for character 0x4d
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x67
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x6a
```

```
Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x70
```

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x51

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x38

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics
unknown for character 0x4d

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x6a

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x70

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x51

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics
unknown for character 0x4d
```

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x6a

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x70

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x51

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x54

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xab

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xcd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xf4

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xfd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x7f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x54

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x70

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x74

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x75

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x73

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x69

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6e

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x20

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x43

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x68

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x69

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x63

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x67

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x6f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x59

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x38

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x59

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x54

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xab

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xcd

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xf4

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xfd
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x7f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x54

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x70

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x74

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x75

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x73
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x69

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6e

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x43

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x68

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x69

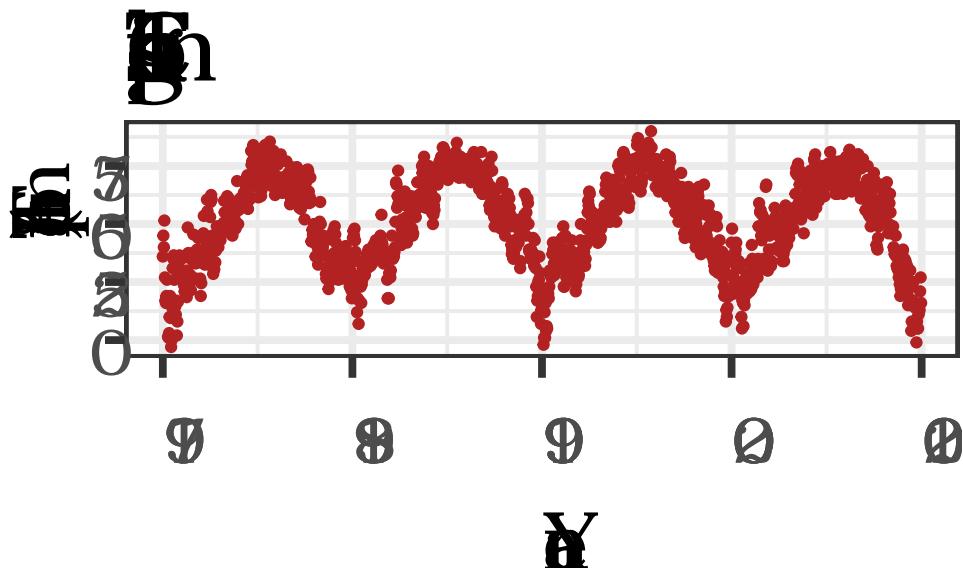
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x63

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x67

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

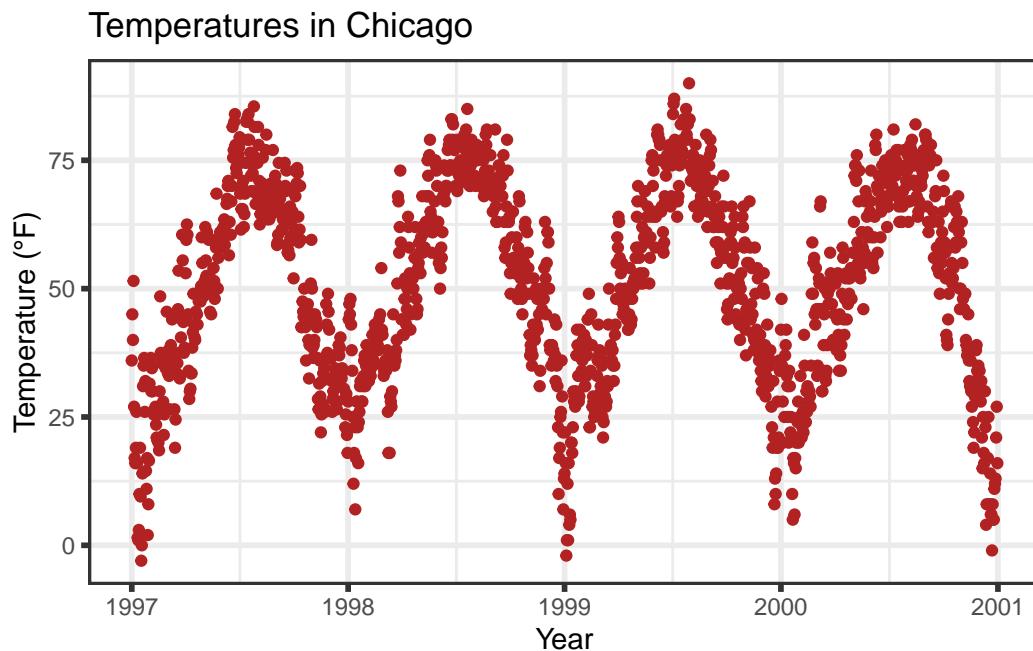
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d
```



11.4 Change the Size of All Line and Rect Elements

Similarly, you can change the size of all elements of type `line` and `rect`:

```
g + theme_bw(base_line_size = 1, base_rect_size = 1)
```



11.5 Create Your Own Theme

If you want to change the theme for an entire session you can use `theme_set` as in `theme_set(theme_bw())`. The default is called `theme_gray` (or `theme_gray`). If you wanted to create your own custom theme, you could extract the code directly from the gray theme and modify. Note that the `rel()` function change the sizes relative to the `base_size`.

```
theme_gray
```

```
function (base_size = 11, base_family = "", base_line_size = base_size/22,
         base_rect_size = base_size/22)
{
  half_line <- base_size/2
  t <- theme(line = element_line(colour = "black", linewidth = base_line_size,
                                 linetype = 1, lineend = "butt"), rect = element_rect(fill = "white",
                                 colour = "black", linewidth = base_rect_size, linetype = 1),
             text = element_text(family = base_family, face = "plain",
                                 colour = "black", size = base_size, lineheight = 0.9,
                                 hjust = 0.5, vjust = 0.5, angle = 0, margin = margin(),
                                 debug = FALSE), axis.line = element_blank(), axis.line.x = NULL,
             axis.line.y = NULL, axis.text = element_text(size = rel(0.8),
                                 colour = "grey30"), axis.text.x = element_text(margin = margin(t = 0.8 *
                                 half_line/2), vjust = 1), axis.text.x.top = element_text(margin = margin(b = 0.8 *
                                 half_line/2), vjust = 0), axis.text.y = element_text(margin = margin(r = 0.8 *
                                 half_line/2), hjust = 1), axis.text.y.right = element_text(margin = margin(l = 0.8 *
                                 half_line/2), hjust = 0), axis.text.r = element_text(margin = margin(l = 0.8 *
                                 half_line/2, r = 0.8 * half_line/2), hjust = 0.5),
             axis.ticks = element_line(colour = "grey20"), axis.ticks.length = unit(half_line/2,
                                 "pt"), axis.ticks.length.x = NULL, axis.ticks.length.x.top = NULL,
             axis.ticks.length.x.bottom = NULL, axis.ticks.length.y = NULL,
             axis.ticks.length.y.left = NULL, axis.ticks.length.y.right = NULL,
             axis.minor.ticks.length = rel(0.75), axis.title.x = element_text(margin = margin(t =
                                 vjust = 1), axis.title.x.top = element_text(margin = margin(b = half_line/2),
                                 vjust = 0), axis.title.y = element_text(angle = 90,
                                 margin = margin(r = half_line/2), vjust = 1), axis.title.y.right = element_text(
                                 margin = margin(l = half_line/2), vjust = 1), legend.background = element_rect(
                                 legend.spacing = unit(2 * half_line, "pt"), legend.spacing.x = NULL,
                                 legend.spacing.y = NULL, legend.margin = margin(half_line,
                                 half_line, half_line, half_line), legend.key = NULL,
                                 legend.key.size = unit(1.2, "lines"), legend.key.height = NULL,
                                 legend.key.width = NULL, legend.key.spacing = unit(half_line,
                                 "pt"), legend.text = element_text(size = rel(0.8))),
```

```

legend.title = element_text(hjust = 0), legend.ticks.length = rel(0.2),
legend.position = "right", legend.direction = NULL, legend.justification = "center",
legend.box = NULL, legend.box.margin = margin(0, 0, 0,
      0, "cm"), legend.box.background = element_blank(),
legend.box.spacing = unit(2 * half_line, "pt"), panel.background = element_rect(fill =
      colour = NA), panel.border = element_blank(), panel.grid = element_line(colour =
panel.grid.minor = element_line(linewidth = rel(0.5)),
panel.spacing = unit(half_line, "pt"), panel.spacing.x = NULL,
panel.spacing.y = NULL, panel.on top = FALSE, strip.background = element_rect(fill =
      colour = NA), strip.clip = "inherit", strip.text = element_text(colour = "grey10",
      size = rel(0.8), margin = margin(0.8 * half_line,
      0.8 * half_line, 0.8 * half_line, 0.8 * half_line)),
strip.text.x = NULL, strip.text.y = element_text(angle = -90),
strip.text.y.left = element_text(angle = 90), strip.placement = "inside",
strip.placement.x = NULL, strip.placement.y = NULL, strip.switch.pad.grid = unit(half_
      "pt"), strip.switch.pad.wrap = unit(half_line/2,
      "pt"), plot.background = element_rect(colour = "white"),
plot.title = element_text(size = rel(1.2), hjust = 0,
      vjust = 1, margin = margin(b = half_line)), plot.title.position = "panel",
plot.subtitle = element_text(hjust = 0, vjust = 1, margin = margin(b = half_line)),
plot.caption = element_text(size = rel(0.8), hjust = 1,
      vjust = 1, margin = margin(t = half_line)), plot.caption.position = "panel",
plot.tag = element_text(size = rel(1.2), hjust = 0.5,
      vjust = 0.5), plot.tag.position = "topleft", plot.margin = margin(half_line,
      half_line, half_line, half_line), complete = TRUE)
ggplot_global$theme_all_null %+replace% t
}
<bytecode: 0x000001650e3112b0>
<environment: namespace:ggplot2>
```

Now, let us modify the default theme function and have a look at the result:

```

theme_2hin <- function (base_size = 12, base_family = "Roboto Condensed") {
  half_line <- base_size/2
  theme(
    line = element_line(color = "black", linewidth = .5,
      linetype = 1, lineend = "butt"),
    rect = element_rect(fill = "white", color = "black",
      linewidth = .5, linetype = 1),
    text = element_text(family = base_family, face = "plain",
      color = "black", size = base_size,
      lineheight = .9, hjust = .5, vjust = .5,
```

```

            angle = 0, margin = margin(), debug = FALSE),
axis.line = element_blank(),
axis.line.x = NULL,
axis.line.y = NULL,
axis.text = element_text(size = base_size * 1.1, color = "gray30"),
axis.text.x = element_text(margin = margin(t = .8 * half_line/2),
                           vjust = 1),
axis.text.x.top = element_text(margin = margin(b = .8 * half_line/2),
                               vjust = 0),
axis.text.y = element_text(margin = margin(r = .8 * half_line/2),
                           hjust = 1),
axis.text.y.right = element_text(margin = margin(l = .8 * half_line/2),
                                 hjust = 0),
axis.ticks = element_line(color = "gray30", linewidth = .7),
axis.ticks.length = unit(half_line / 1.5, "pt"),
axis.ticks.length.x = NULL,
axis.ticks.length.x.top = NULL,
axis.ticks.length.x.bottom = NULL,
axis.ticks.length.y = NULL,
axis.ticks.length.y.left = NULL,
axis.ticks.length.y.right = NULL,
axis.title.x = element_text(margin = margin(t = half_line),
                           vjust = 1, size = base_size * 1.3,
                           face = "bold"),
axis.title.x.top = element_text(margin = margin(b = half_line),
                               vjust = 0),
axis.title.y = element_text(angle = 90, vjust = 1,
                           margin = margin(r = half_line),
                           size = base_size * 1.3, face = "bold"),
axis.title.y.right = element_text(angle = -90, vjust = 0,
                                 margin = margin(l = half_line)),
legend.background = element_rect(color = NA),
legend.spacing = unit(.4, "cm"),
legend.spacing.x = NULL,
legend.spacing.y = NULL,
legend.margin = margin(.2, .2, .2, .2, "cm"),
legend.key = element_rect(fill = "gray95", color = "white"),
legend.key.size = unit(1.2, "lines"),
legend.key.height = NULL,
legend.key.width = NULL,
legend.text = element_text(size = rel(.8)),
legend.text.align = NULL,

```

```

legend.title = element_text(hjust = 0),
legend.title.align = NULL,
legend.position = "right",
legend.direction = NULL,
legend.justification = "center",
legend.box = NULL,
legend.box.margin = margin(0, 0, 0, 0, "cm"),
legend.box.background = element_blank(),
legend.box.spacing = unit(.4, "cm"),
panel.background = element_rect(fill = "white", color = NA),
panel.border = element_rect(color = "gray30",
                             fill = NA, linewidth = .7),
panel.grid.major = element_line(color = "gray90", linewidth = 1),
panel.grid.minor = element_line(color = "gray90", linewidth = .5,
                                 linetype = "dashed"),
panel.spacing = unit(base_size, "pt"),
panel.spacing.x = NULL,
panel.spacing.y = NULL,
panel.on top = FALSE,
strip.background = element_rect(fill = "white", color = "gray30"),
strip.text = element_text(color = "black", size = base_size),
strip.text.x = element_text(margin = margin(t = half_line,
                                             b = half_line)),
strip.text.y = element_text(angle = -90,
                            margin = margin(l = half_line,
                                            r = half_line)),
strip.text.y.left = element_text(angle = 90),
strip.placement = "inside",
strip.placement.x = NULL,
strip.placement.y = NULL,
strip.switch.pad.grid = unit(0.1, "cm"),
strip.switch.pad.wrap = unit(0.1, "cm"),
plot.background = element_rect(color = NA),
plot.title = element_text(size = base_size * 1.8, hjust = .5,
                           vjust = 1, face = "bold",
                           margin = margin(b = half_line * 1.2)),
plot.title.position = "panel",
plot.subtitle = element_text(size = base_size * 1.3,
                             hjust = .5, vjust = 1,
                             margin = margin(b = half_line * .9)),
plot.caption = element_text(size = rel(0.9), hjust = 1, vjust = 1,
                           margin = margin(t = half_line * .9)),

```

```

    plot.caption.position = "panel",
    plot.tag = element_text(size = rel(1.2), hjust = .5, vjust = .5),
    plot.tag.position = "topleft",
    plot.margin = margin(rep(base_size, 4)),
    complete = TRUE
)
}

```

You can only overwrite the defaults for all elements you want to change. Here I listed all so you can see that you can change *literally* change everything!

Have a look on the modified aesthetics with its new look of panel and gridlines as well as axes ticks, texts and titles:

```

theme_set(theme_2hin())

ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() + labs(x = "Year", y = "Temperature (°F)") + guides(color = "none")

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font metrics
unknown for character 0x4d

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x6a

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x70

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x$label)): font width
unknown for character 0x51

```

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x38

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font metrics unknown for character 0x4d

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x67

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x6a

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x70

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x71

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x79

Warning in grid.Call(C_stringMetric, as.graphicsAnnot(x\$label)): font width unknown for character 0x51

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x54

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xab

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xcd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xf4

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0xfd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x7f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x59

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x37

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x38

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x59

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0x54
```

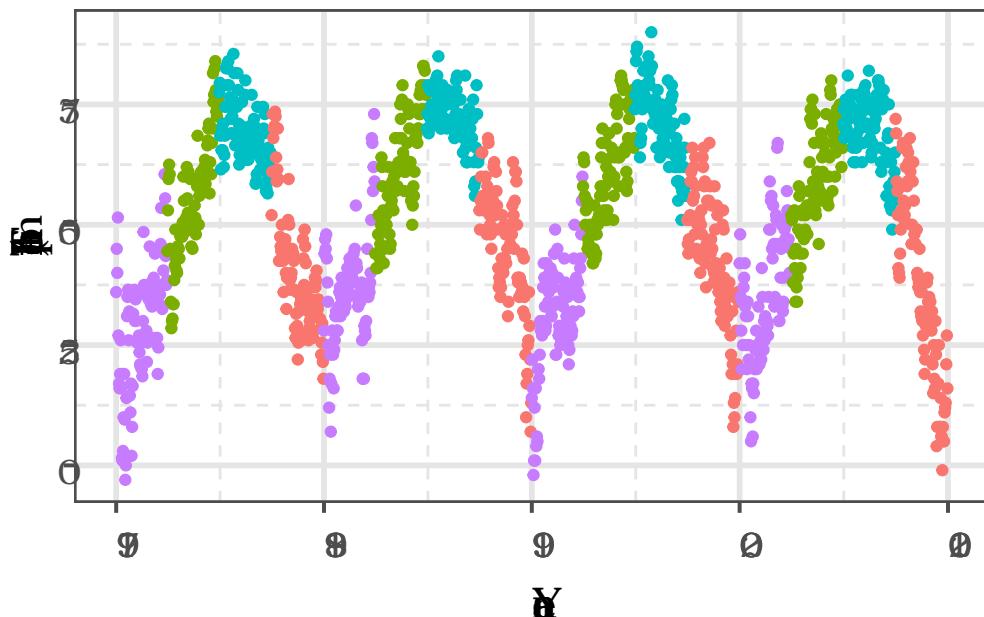
```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0xab
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0xcd
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0xf4
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0xfd
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0x7f
```



This way of changing the plot design is highly recommended! It allows you to quickly change any element of your plots by changing it once. You can within a few seconds plot all your results in a congruent style and adapt it to other needs (e.g. a presentation with bigger font size or journal requirements).

11.6 Update the Current Theme

You can also set quick changes using `theme_update()`:

```
theme_2hin <- theme_update(panel.background = element_rect(fill = "gray60"))

ggplot(chic, aes(x = date, y = temp, color = season)) +
  geom_point() + labs(x = "Year", y = "Temperature (°F)") + guides(color = "none")
```

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x38

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x39

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x31

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x32
```

```
Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x30

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x37

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x35

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x54

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0xab

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0xcd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0xf4

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0xfd

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x7f

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
metrics unknown for character 0x4d

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : font
width unknown for character 0x59
```

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x65

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x61

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x35

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, : font width unknown for character 0x39

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x37

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x38

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x39

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x32
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x30

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x31

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x59

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x54

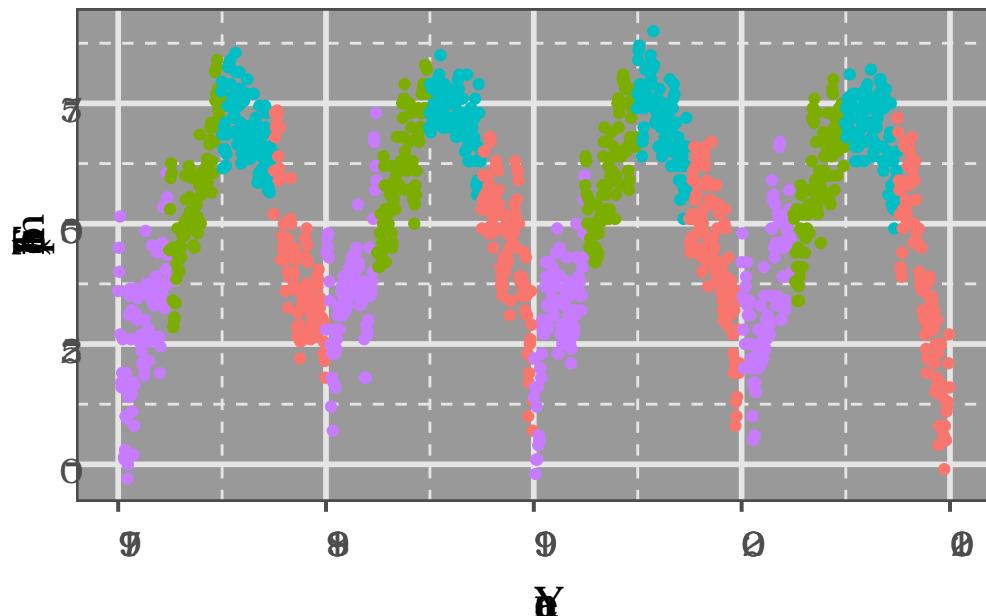
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xab

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xcd

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xf4

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0xfd
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0x7f
```



For further exercises, we are going to use our own theme with a white filling and without the minor grid lines:

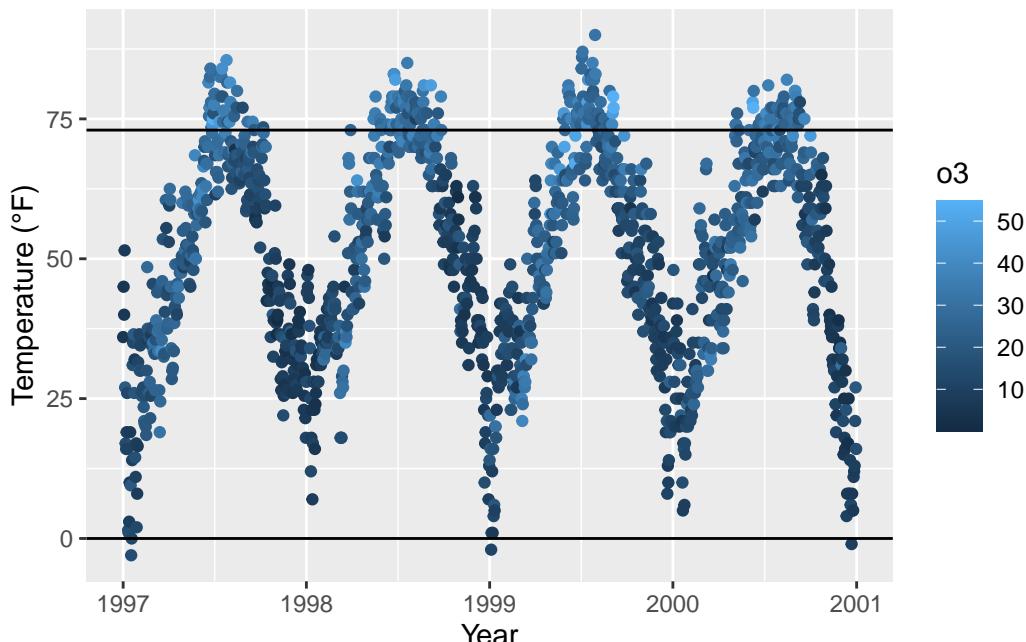
```
theme_2hin <- theme_update(
  panel.background = element_rect(fill = "white"),
  panel.grid.major = element_line(linewidth = .5),
  panel.grid.minor = element_blank()
)
```

12 Working with Lines

12.1 Add Horizontal or Vertical Lines to a Plot

You might want to highlight a given range or threshold, which can be done plotting a line at defined coordinates using `geom_hline()` (for “horizontal lines”) or `geom_vline()` (for “vertical lines”):

```
ggplot(chic, aes(x = date, y = temp, color = o3)) +  
  geom_point() +  
  geom_hline(yintercept = c(0, 73)) +  
  labs(x = "Year", y = "Temperature (°F)")
```

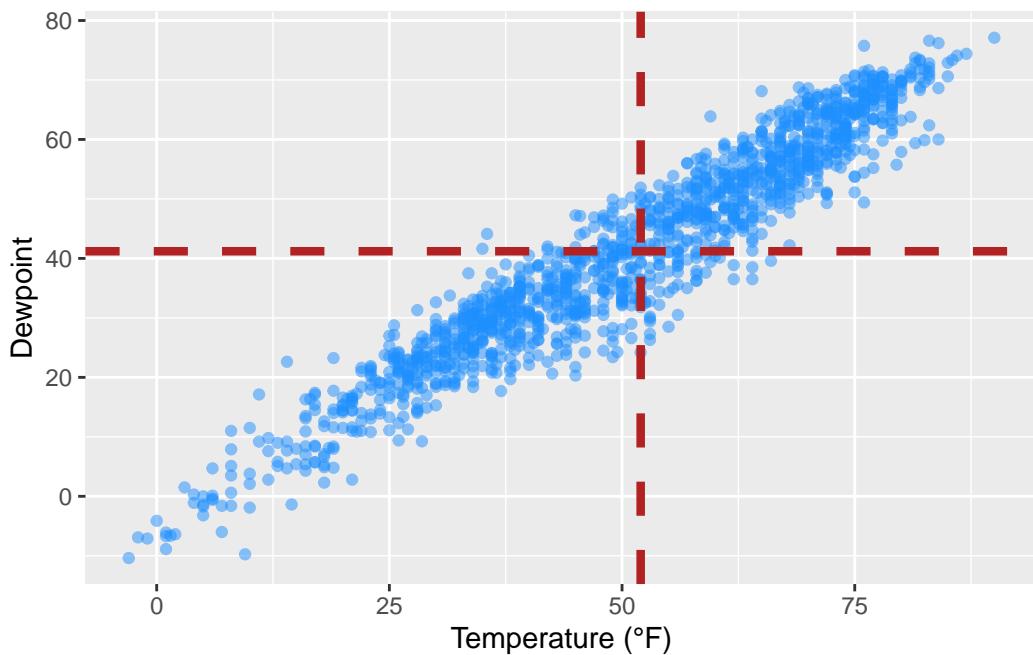


```
g <- ggplot(chic, aes(x = temp, y = dewpoint)) +  
  geom_point(color = "dodgerblue", alpha = .5) +  
  labs(x = "Temperature (°F)", y = "Dewpoint")
```

```

g +
  geom_vline(aes(xintercept = median(temp)), linewidth = 1.5,
             color = "firebrick", linetype = "dashed") +
  geom_hline(aes(yintercept = median(dewpoint)), linewidth = 1.5,
             color = "firebrick", linetype = "dashed")

```



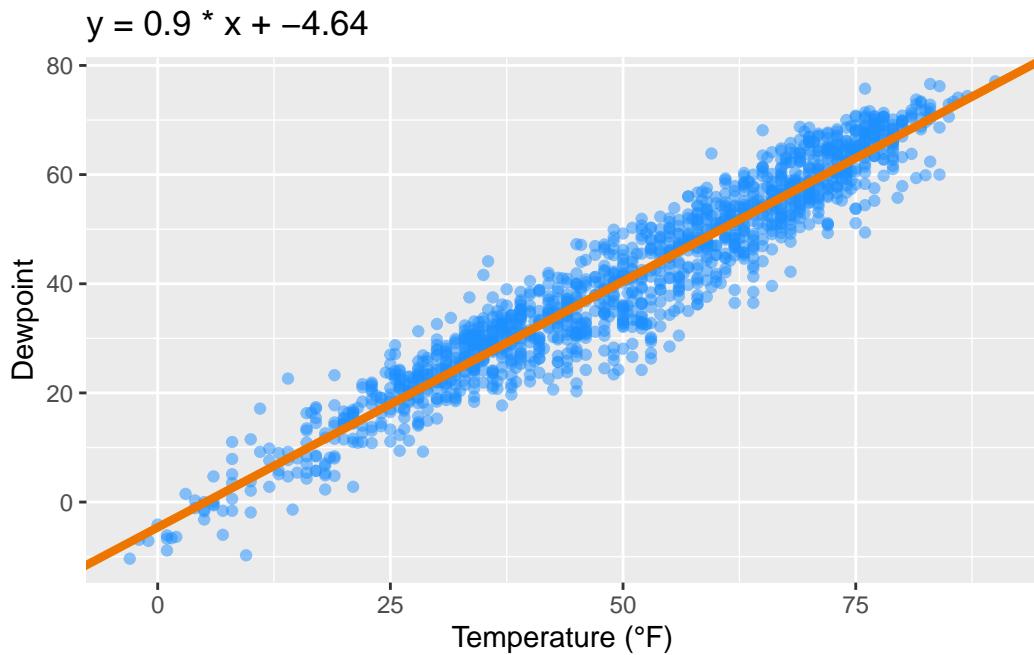
If you want to add a line with a slope not being 0 or 1, respectively, you need to use `geom_abline()`. This is for example the case if you want to add a regression line using the arguments `intercept` and `slope`:

```

reg <- lm(dewpoint ~ temp, data = chic)

g +
  geom_abline(intercept = coefficients(reg)[1],
              slope = coefficients(reg)[2],
              color = "darkorange2",
              linewidth = 1.5) +
  labs(title = paste0("y = ", round(coefficients(reg)[2], 2),
                     " * x + ", round(coefficients(reg)[1], 2)))

```

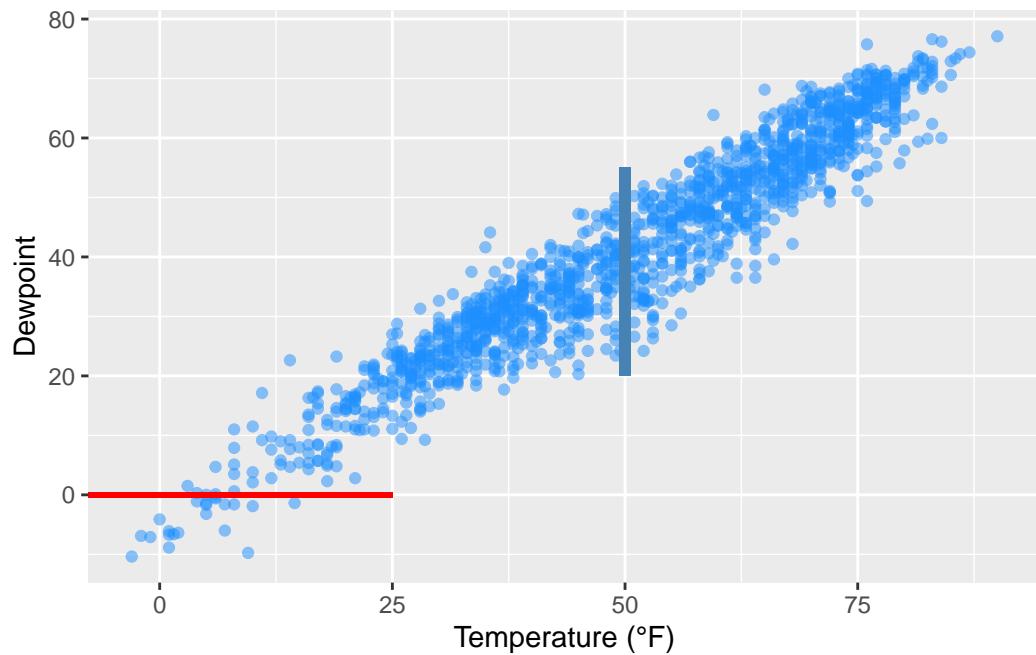


Later, we will learn how to add a linear fit with one command using `stat_smooth(method = "lm")`. However, there might be other reasons to add a line with a given slope and this is how one does it

12.2 Add a Line within a Plot

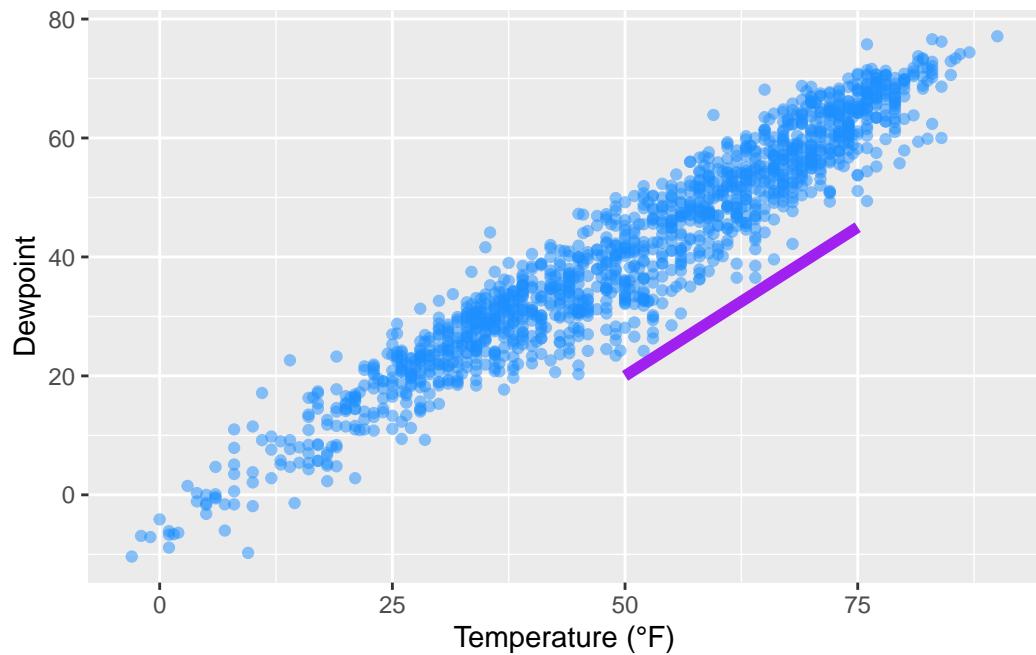
The previous approaches always covered the whole range of the plot panel, but sometimes one wants to highlight only a given area or use lines for annotations. In this case, `geom_linerange()` is here to help:

```
g +
  ## vertical line
  geom_linerange(aes(x = 50, ymin = 20, ymax = 55),
                 color = "steelblue", linewidth = 2) +
  ## horizontal line
  geom_linerange(aes(xmin = -Inf, xmax = 25, y = 0),
                 color = "red", linewidth = 1)
```



Or you can use `annotate(geom = "segment")` to draw lines with a slope differing from 0 and 1:

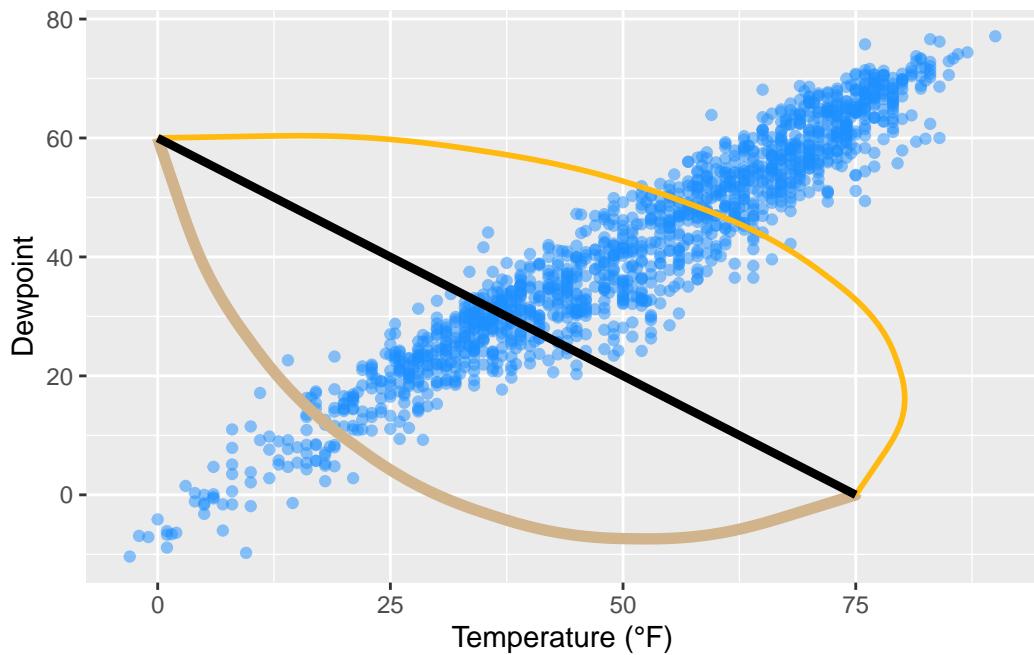
```
g +
  annotate(geom = "segment",
            x = 50, xend = 75,
            y = 20, yend = 45,
            color = "purple", linewidth = 2)
```



12.3 Add Curved Lines and Arrows to a Plot

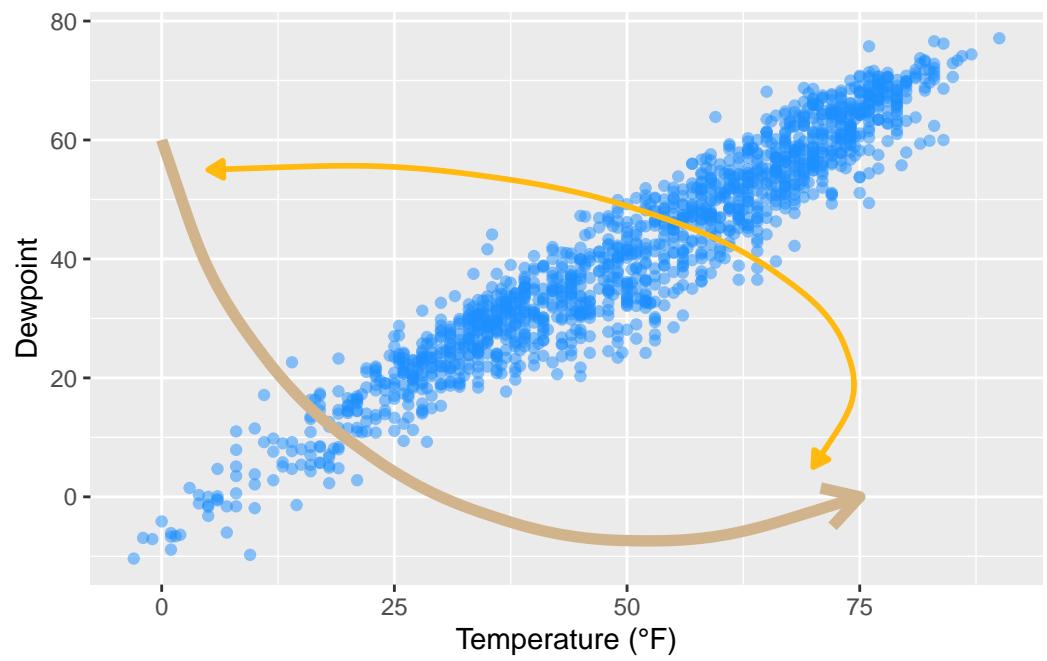
`annotate(geom = "curve")` adds curves. Well, and straight lines if you like:

```
g +
  annotate(geom = "curve", x = 0, y = 60, xend = 75, yend = 0,
           color = "tan", linewidth = 2) +
  annotate(geom = "curve",
           x = 0, y = 60, xend = 75, yend = 0,
           curvature = -0.7, angle = 45,
           color = "darkgoldenrod1", linewidth = 1) +
  annotate(geom = "curve", x = 0, y = 60, xend = 75, yend = 0,
           curvature = 0, linewidth = 1.5)
```



The same geom can be used to draw arrows:

```
g +
  annotate(geom = "curve", x = 0, y = 60, xend = 75, yend = 0,
           color = "tan", linewidth = 2,
           arrow = arrow(length = unit(0.07, "npc")))) +
  annotate(geom = "curve", x = 5, y = 55, xend = 70, yend = 5,
           curvature = -0.7, angle = 45,
           color = "darkgoldenrod1", linewidth = 1,
           arrow = arrow(length = unit(0.03, "npc"),
                         type = "closed",
                         ends = "both"))
```



13 Working with Text

13.1 Add Labels to Your Data

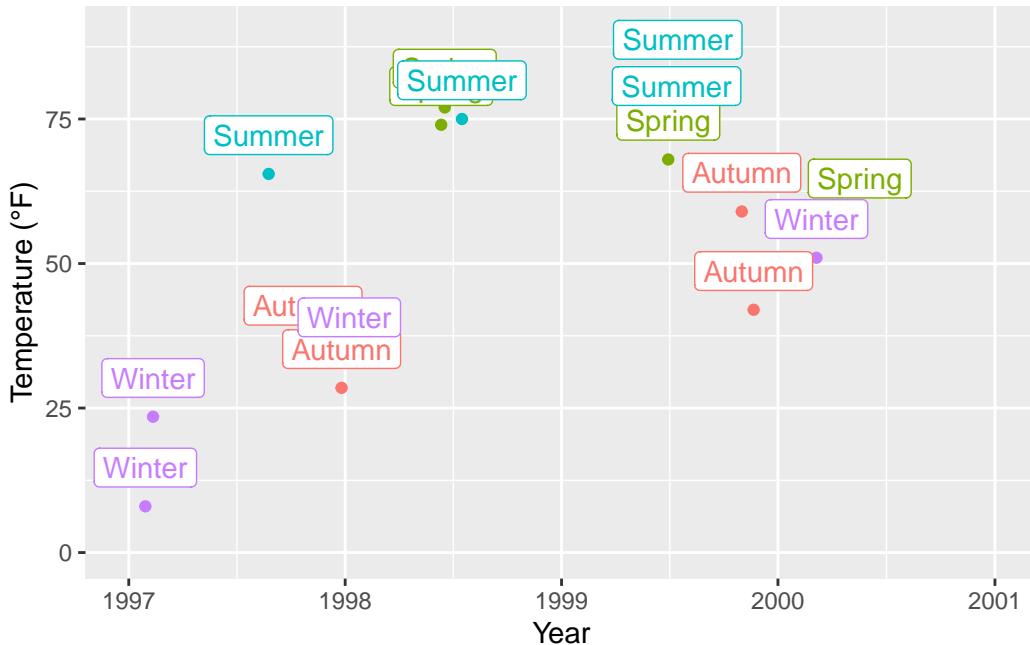
Sometimes, we want to label our data points. To avoid overlaying and crowding by text labels, we use a 1% sample of the original data, equally representing the four seasons. We are using `geom_label()` which comes with a new aesthetic called `label`:

```
set.seed(2020)

sample <- chic |>
  dplyr::group_by(season) |>
  dplyr::sample_frac(0.01)

## code without pipes:
## sample <- sample_frac(group_by(chic, season), .01)

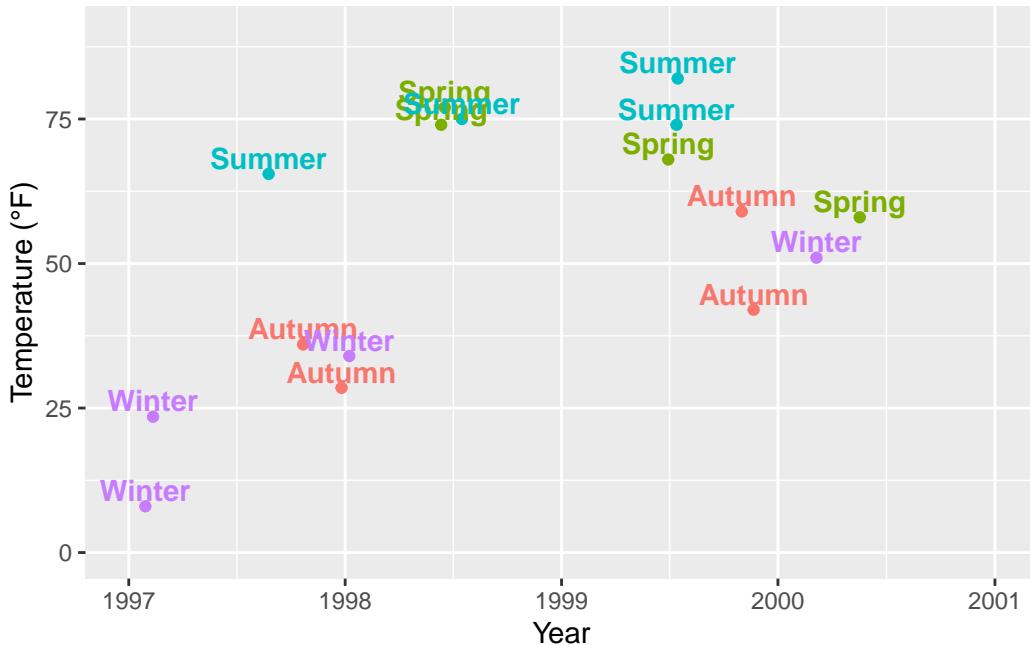
ggplot(sample, aes(x = date, y = temp, color = season)) +
  geom_point() +
  geom_label(aes(label = season), hjust = .5, vjust = -.5) +
  labs(x = "Year", y = "Temperature (°F)") +
  xlim(as.Date(c('1997-01-01', '2000-12-31'))) +
  ylim(c(0, 90)) +
  theme(legend.position = "none")
```



Okay, avoiding overlap of labels did not work out. But don't worry, we are going to fix it in a minute!

You can also use `geom_text()` if you don't like boxes around your labels. Expand to see an example.

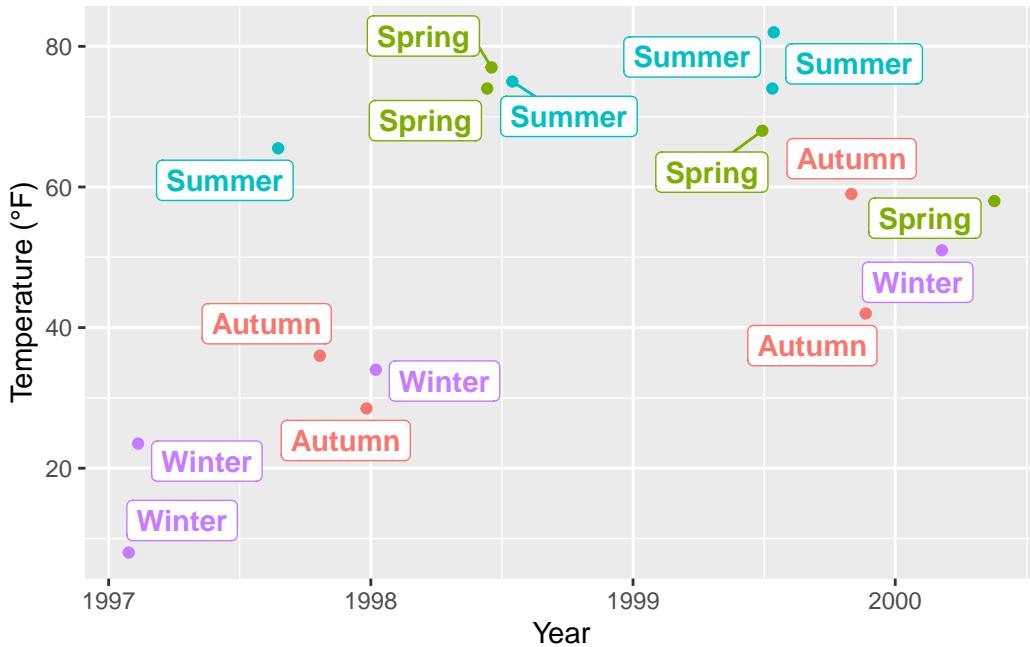
```
ggplot(sample, aes(x = date, y = temp, color = season)) +
  geom_point() +
  geom_text(aes(label = season), fontface = "bold",
            hjust = .5, vjust = -.25) +
  labs(x = "Year", y = "Temperature (°F)") +
  xlim(as.Date(c('1997-01-01', '2000-12-31'))) +
  ylim(c(0, 90)) +
  theme(legend.position = "none")
```



The `{ggrepel}` package offers some great utilities by providing geoms for `{ggplot2}` to repel overlapping text as in our examples above. We simply replace `geom_text()` by `geom_text_repel()` and `geom_label()` by `geom_label_repel()`:

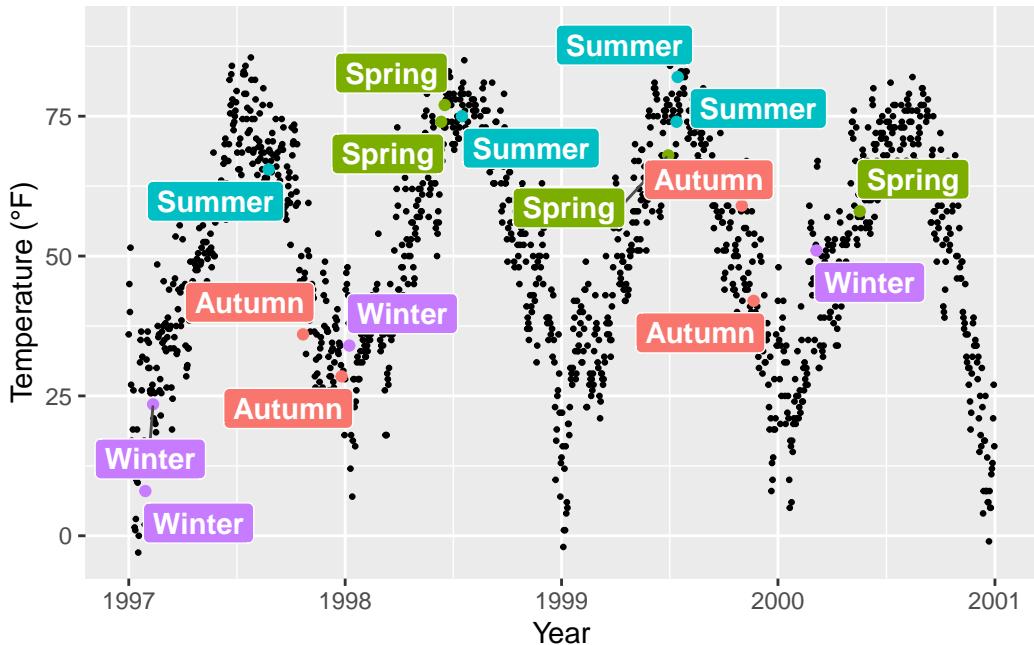
```
library(ggrepel)

ggplot(sample, aes(x = date, y = temp, color = season)) +
  geom_point() +
  geom_label_repel(aes(label = season), fontface = "bold") +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.position = "none")
```



It may look nicer with filled boxes so we map `season` to `fill` instead to color and set a white color for the text:

```
ggplot(sample, aes(x = date, y = temp)) +
  geom_point(data = chic, size = .5) +
  geom_point(aes(color = season), size = 1.5) +
  geom_label_repel(aes(label = season, fill = season),
                  color = "white", fontface = "bold",
                  segment.color = "grey30") +
  labs(x = "Year", y = "Temperature (°F)") +
  theme(legend.position = "none")
```

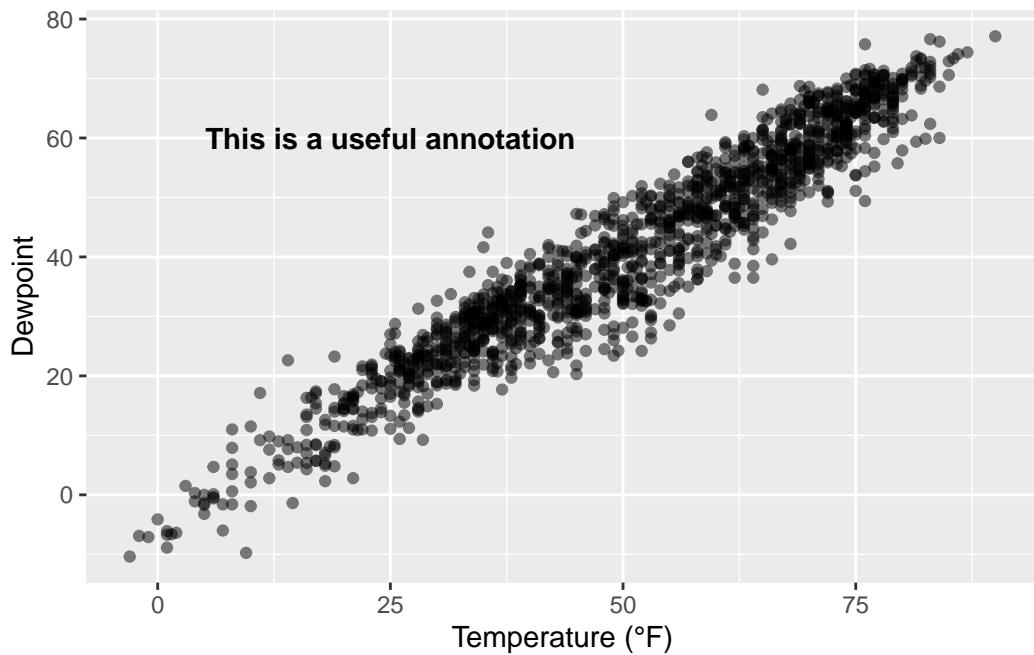


This also works for the pure text labels by using `geom_text_repel()`. Have a look at all the [usage examples](#).

13.2 Add Text Annotations

There are several ways how one can add annotations to a ggplot. We can again use `annotate(geom = "text")`, `annotate(geom = "label")`, `geom_text()` or `geom_label()`:

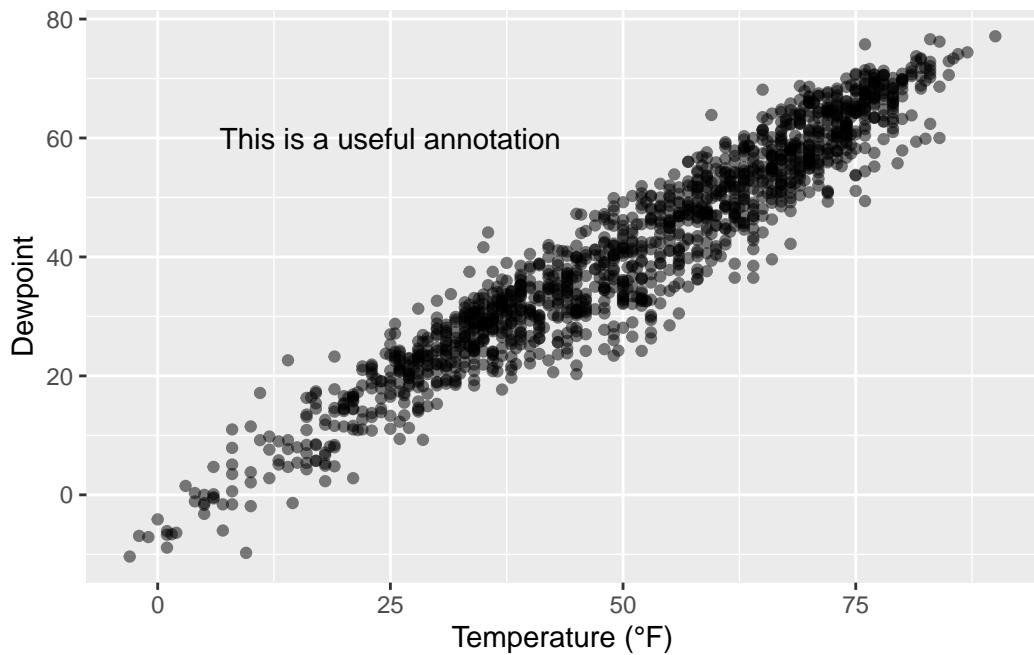
```
g <-  
  ggplot(chic, aes(x = temp, y = dewpoint)) +  
  geom_point(alpha = .5) +  
  labs(x = "Temperature (°F)", y = "Dewpoint")  
  
g +  
  annotate(geom = "text", x = 25, y = 60, fontface = "bold",  
          label = "This is a useful annotation")
```



However, now ggplot has drawn one text label per data point—that's 1,461 labels and you only see one! You can solve that by setting the `stat` argument to "unique":

```
g +
  geom_text(aes(x = 25, y = 60,
                label = "This is a useful annotation"),
            stat = "unique")
```

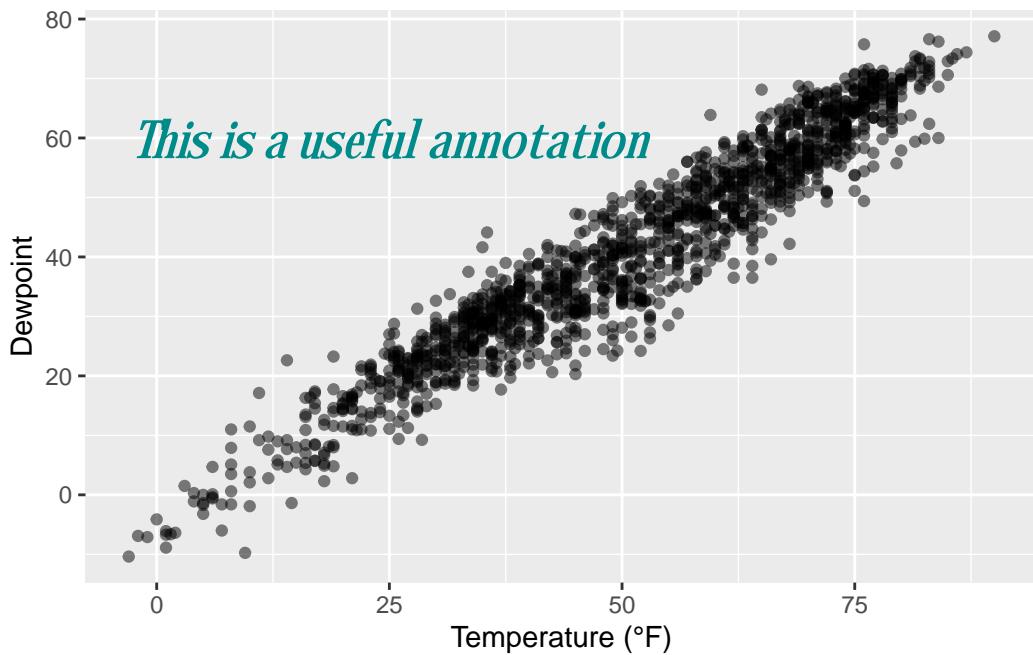
```
Warning in geom_text(aes(x = 25, y = 60, label = "This is a useful annotation"), : All aesthetics
i Did you mean to use `annotate()``?
```



By the way, of course one can change the properties of the displayed text:

```
g +
  geom_text(aes(x = 25, y = 60,
                 label = "This is a useful annotation"),
            stat = "unique", family = "Bangers",
            size = 7, color = "darkcyan")
```

Warning in geom_text(aes(x = 25, y = 60, label = "This is a useful annotation"), : All aesthetics are being recycled. Did you mean to use `annotate()``?



In case you use one of the facet functions to visualize your data you might run into trouble. One thing is that you may want to include the annotation only once:

```
ann <- data.frame(
  o3 = 30,
  temp = 20,
  season = factor("Summer", levels = levels(chic$season)),
  label = "Here is enough space\nfor some annotations."
)

g <-
  ggplot(chic, aes(x = o3, y = temp)) +
  geom_point() +
  labs(x = "Ozone", y = "Temperature (°F)")

g +
  geom_text(data = ann, aes(label = label),
            size = 7, fontface = "bold",
            family = "Roboto Condensed") +
  facet_wrap(~season)
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0x48
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x69

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x73

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6e

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x75

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x67

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x68
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x73

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x70

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x63

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font metrics unknown for character 0x4d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x66

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x72

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x73

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6d

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x65

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x20

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6e

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x74

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x61

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x74

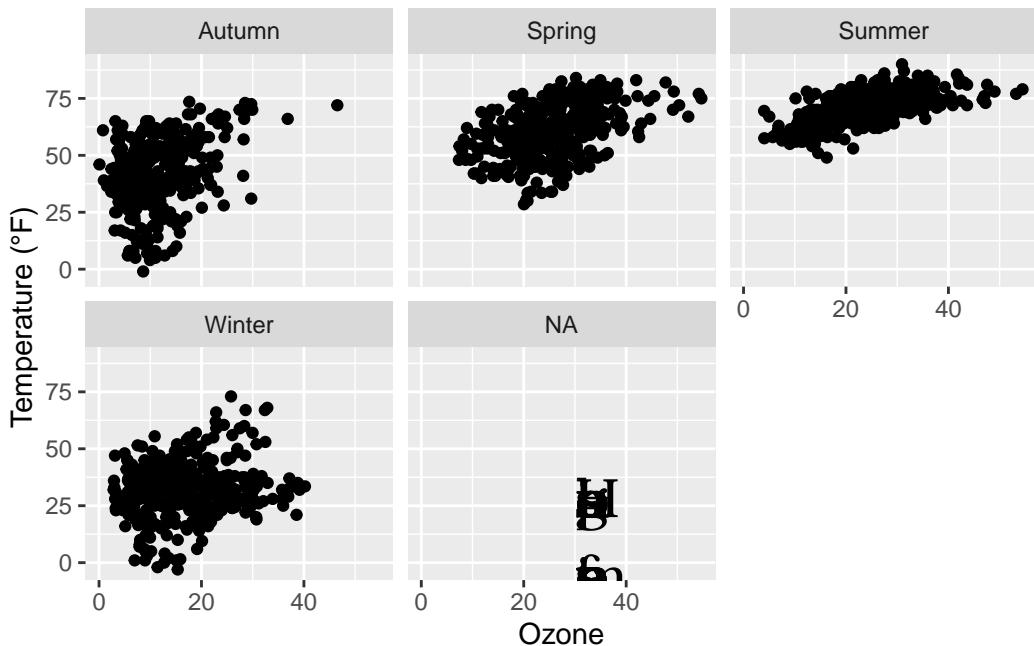
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x69

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6f

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
font width unknown for character 0x6e
```

```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0x73
```

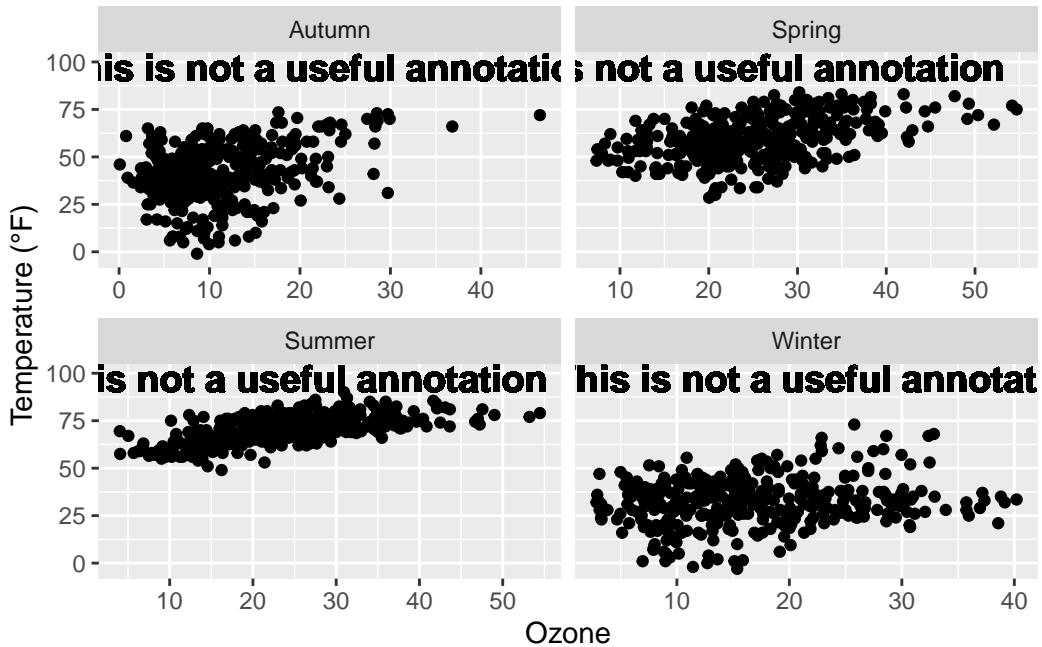
```
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
  font width unknown for character 0x2e
```



Another challenge are facets in combination with free scales that might cut your text:

```
g +
  geom_text(aes(x = 23, y = 97,
                 label = "This is not a useful annotation"),
            size = 5, fontface = "bold") +
  scale_y_continuous(limits = c(NA, 100)) +
  facet_wrap(~season, scales = "free_x")
```

```
Warning in geom_text(aes(x = 23, y = 97, label = "This is not a useful annotation"), : All a
i Did you mean to use `annotate()``?
```



One solution is to calculate the midpoint of the axis, here x, beforehand:

```
ann <-
  chic |>
  dplyr::group_by(season) |>
  dplyr::summarize(
    o3 = min(o3, na.rm = TRUE) +
      (max(o3, na.rm = TRUE) - min(o3, na.rm = TRUE)) / 2
  )

ann

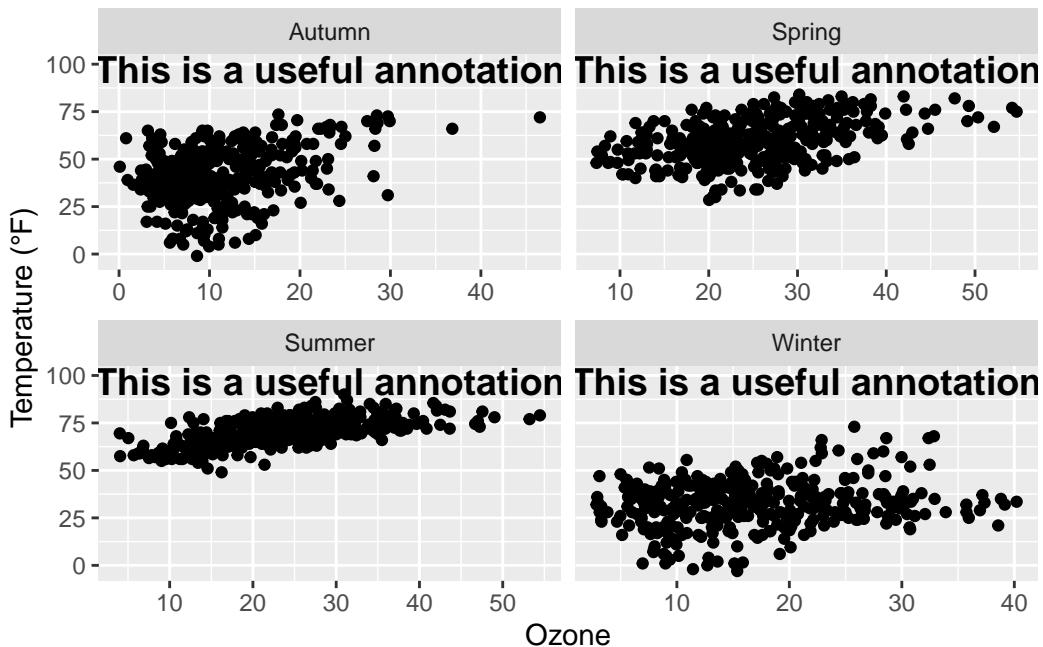
# A tibble: 4 x 2
  season     o3
  <chr>   <dbl>
1 Autumn  23.3
2 Spring  31.0
3 Summer  29.2
4 Winter  21.5
```

... and use the aggregated data to specify the placement of the annotation:

```

g +
  geom_text(data = ann,
             aes(x = o3, y = 97,
                 label = "This is a useful annotation"),
             size = 5, fontface = "bold") +
  scale_y_continuous(limits = c(NA, 100)) +
  facet_wrap(~season, scales = "free_x")

```



However, there is a simpler approach (in terms of fixing the coordinates)—but it also takes a while to know the code by heart. The `{grid}` package in combination with `{ggplot2}`'s `annotation_custom()` allows you to specify the location based on scaled coordinates where 0 is low and 1 is high. `grobTree()` creates a grid graphical object and `textGrob` creates the text graphical object. The value of this is particularly evident when you have multiple plots with different scales.

```

library(grid)
my_grob <- grobTree(textGrob("This text stays in place!",
                             x = .1, y = .9, hjust = 0,
                             gp = gpar(col = "black",
                                       fontsize = 15,
                                       fontface = "bold")))

```

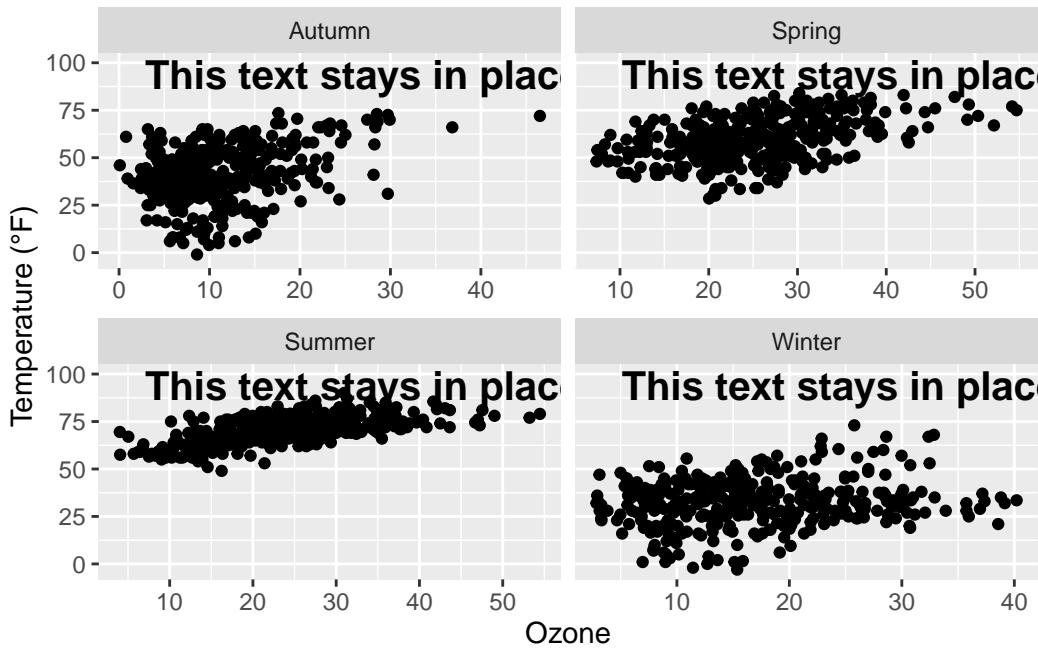


```

g +
  annotation_custom(my_grob) +

```

```
facet_wrap(~season, scales = "free_x") +
  scale_y_continuous(limits = c(NA, 100))
```



13.3 Use Markdown and HTML Rendering for Annotations

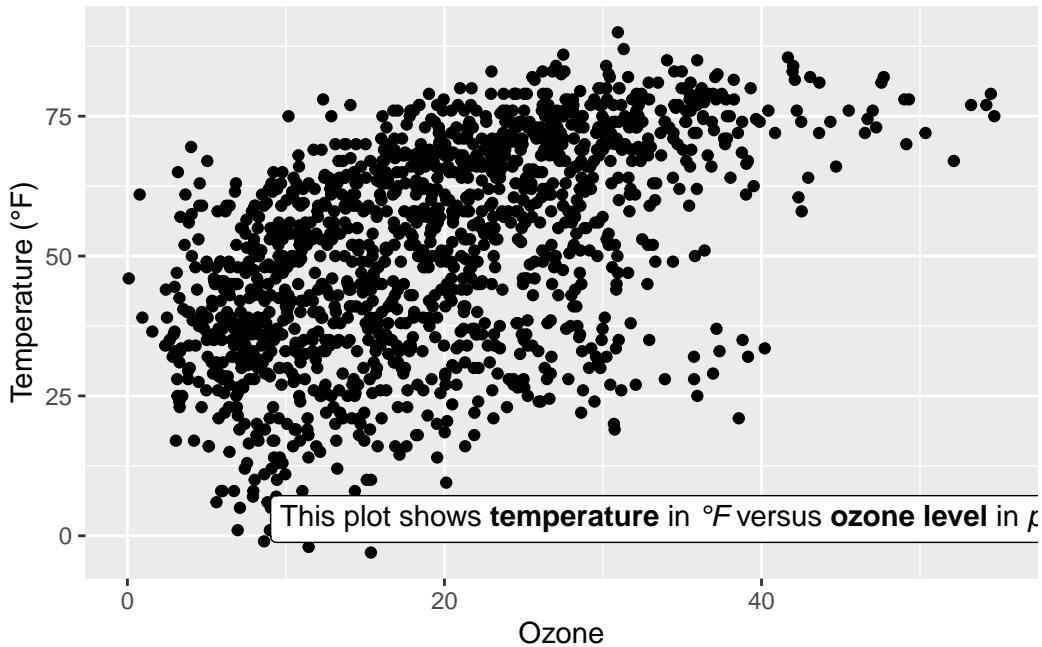
Again, we are using Claus Wilke's `ggtext` package that is designed for improved text rendering support for `ggplot2`. The `ggtext` package defines two new theme elements, `element_markdown()` and `element_textbox()`. The package also provides additional geoms. `geom_richtext()` is a replacement for `geom_text()` and `geom_label()` and renders text as markdown...

```
library(ggtext)

lab_md <- "This plot shows **temperature** in *°F* versus **ozone level** in *ppm*"

g +
  geom_richtext(aes(x = 35, y = 3, label = lab_md),
                stat = "unique")
```

```
Warning in geom_richtext(aes(x = 35, y = 3, label = lab_md), stat = "unique"): All aesthetic
  i Did you mean to use `annotate()``?
```



... or html:

```
lab_html <- "This plot shows "temperature in °F versus ozone level in ppb"
```

```
g +
  geom_richtext(aes(x = 33, y = 3, label = lab_html),
                stat = "unique")
```

Warning in geom_richtext(aes(x = 33, y = 3, label = lab_html), stat = "unique"): All aesthetics must be used with at least one layer

i Did you mean to use `annotate()``?

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <e2>

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <98>

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <85>

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <e2>

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <98>

Warning in grid.Call(C_textBounds, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <85>

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <e2>

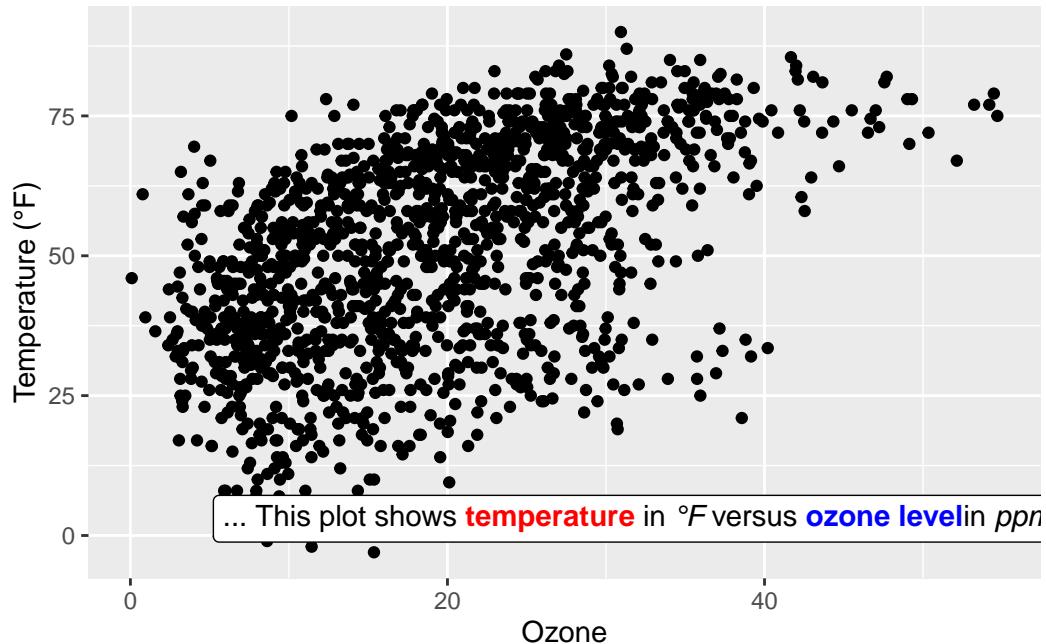
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <98>

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <85>

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <e2>

Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <98>

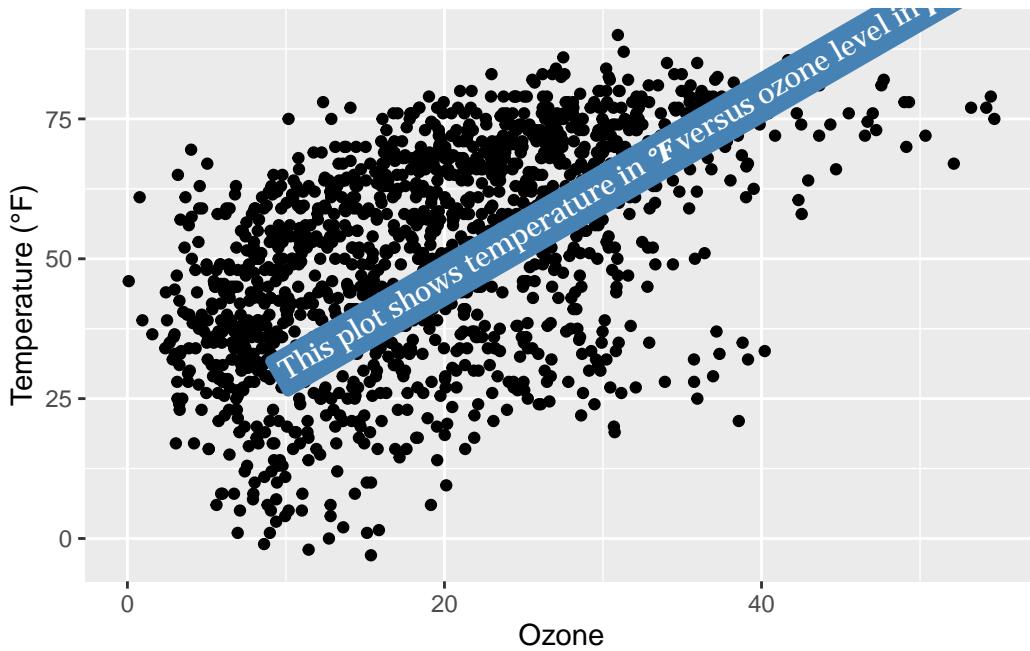
Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x\$label), x\$x, x\$y, :
conversion failure on ' ' in 'mbcsToSbcs': dot substituted for <85>



The geom comes with a lot of details one can modify, such as angle (which is not possible in the default `geom_text()` and `geom_label()`), properties of the box and properties of the text.

```
g +
  geom_richtext(aes(x = 10, y = 25, label = lab_md),
                stat = "unique", angle = 30,
                color = "white", fill = "steelblue",
                label.color = NA, hjust = 0, vjust = 0,
                family = "Playfair Display")
```

Warning in `geom_richtext(aes(x = 10, y = 25, label = lab_md), stat = "unique", : All aesthetics i Did you mean to use `annotate()``?`

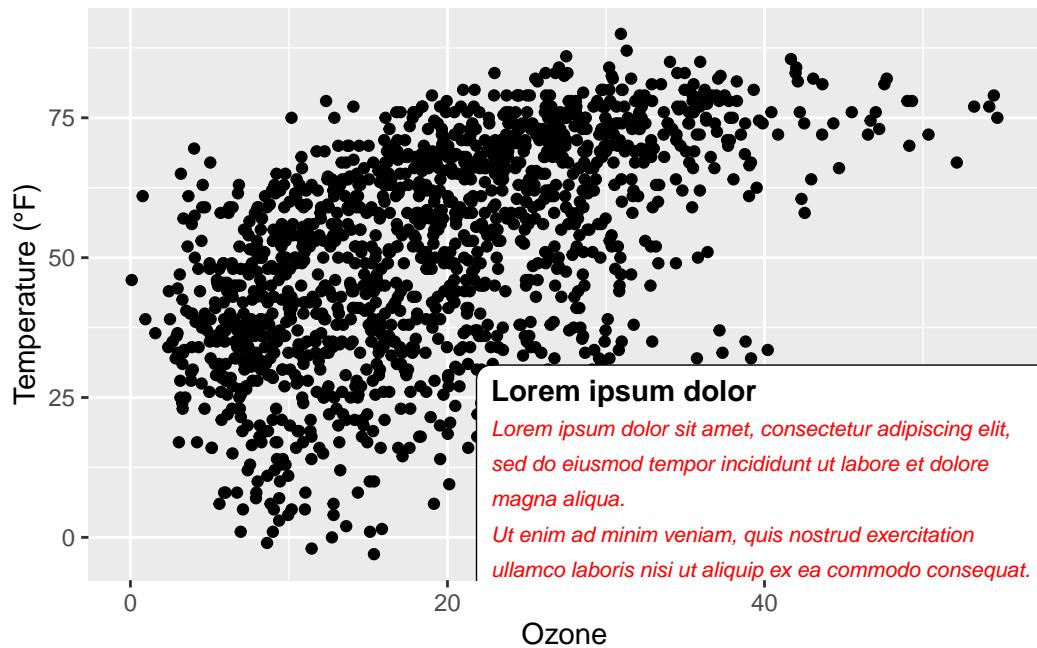


The other geom from the `{ggtext}` package is `geom_textbox()`. This geom allows for dynamic wrapping of strings which is very useful for longer annotations such as info boxes and subtitles.

```
lab_long <- "***Lorem ipsum dolor***<br><i style='font-size:8pt;color:red;'>Lorem ipsum dolor s

g +
  geom_textbox(aes(x = 40, y = 10, label = lab_long),
               width = unit(15, "lines"), stat = "unique")
```

Warning in geom_textbox(aes(x = 40, y = 10, label = lab_long), width = unit(15, : All aesthetics
i Did you mean to use `annotate()``?



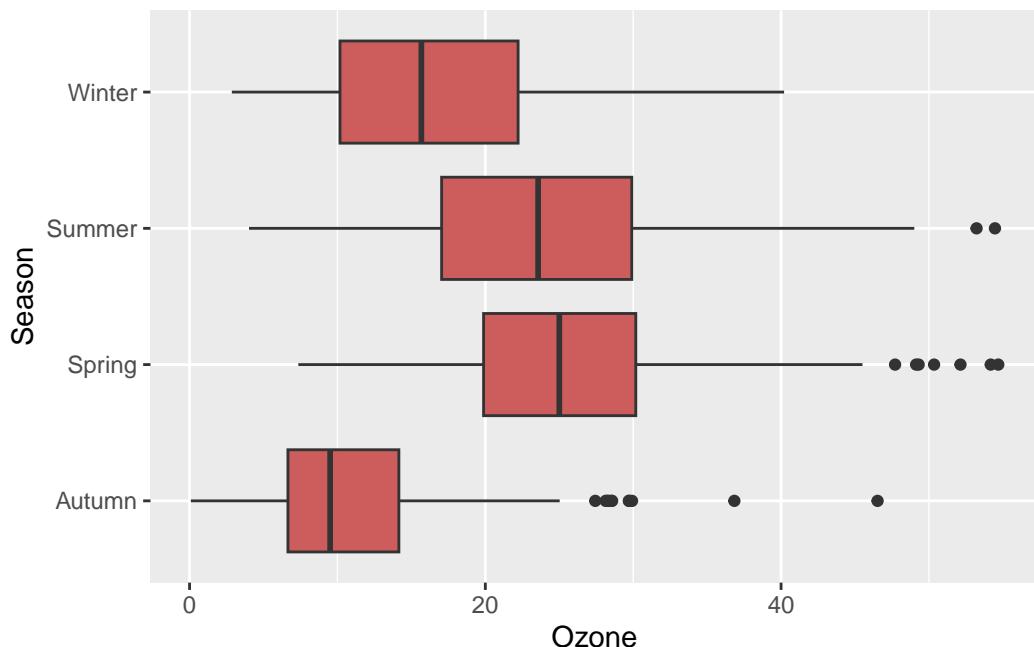
Note that it is not possible to either rotate the textbox (always horizontal) nor to change the justification of the text (always left-aligned).

14 Working with Coordinates

14.1 Flip a Plot

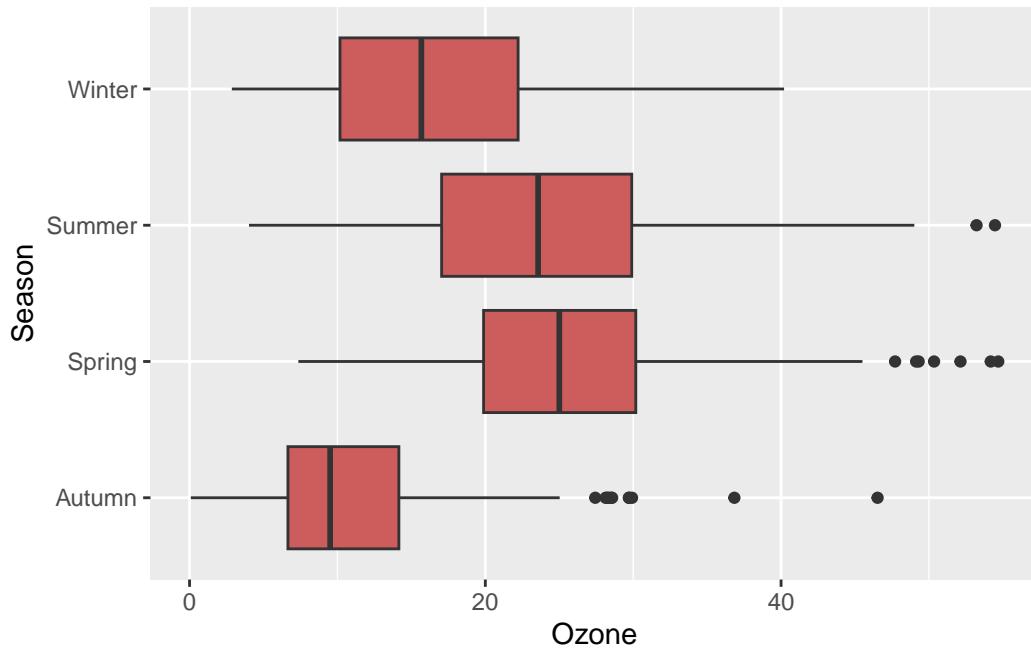
It is incredibly easy to flip a plot on its side. Here I have added the `coord_flip()` which is all you need to flip the plot. This makes most sense when using geom's to represent categorical data, for example bar charts or, as in the following example, box and whiskers plots:

```
ggplot(chic, aes(x = season, y = o3)) +  
  geom_boxplot(fill = "indianred") +  
  labs(x = "Season", y = "Ozone") +  
  coord_flip()
```



Since `{ggplot2}` version 3.0.0 it is also possible to draw geom's horizontally via the argument `orientation = "y"`. Expand to see example.

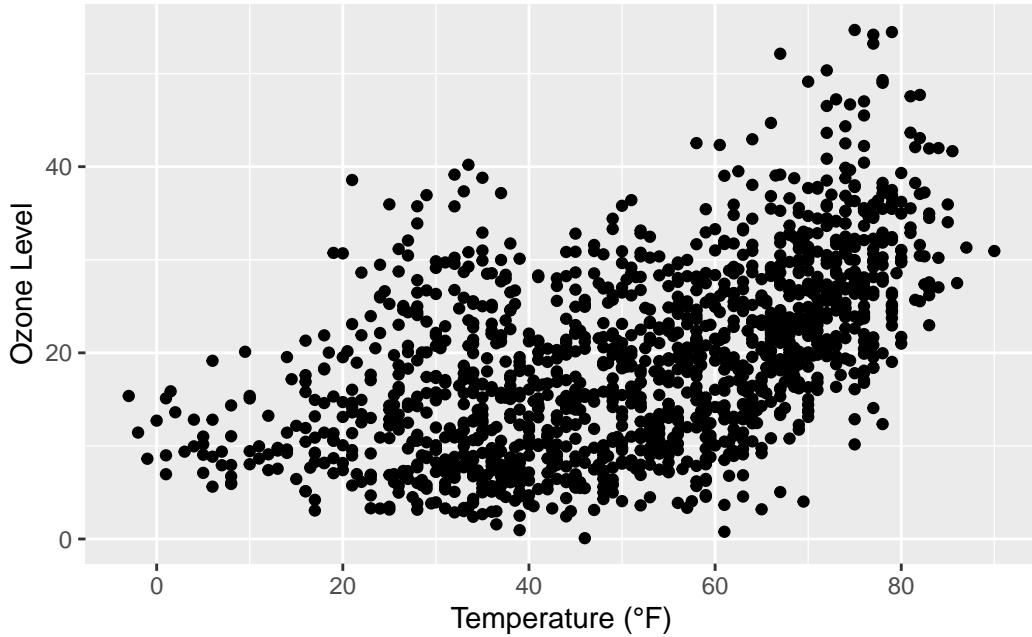
```
ggplot(chic, aes(x = o3, y = season)) +
  geom_boxplot(fill = "indianred", orientation = "y") +
  labs(x = "Ozone", y = "Season")
```



14.2 Fix an Axis

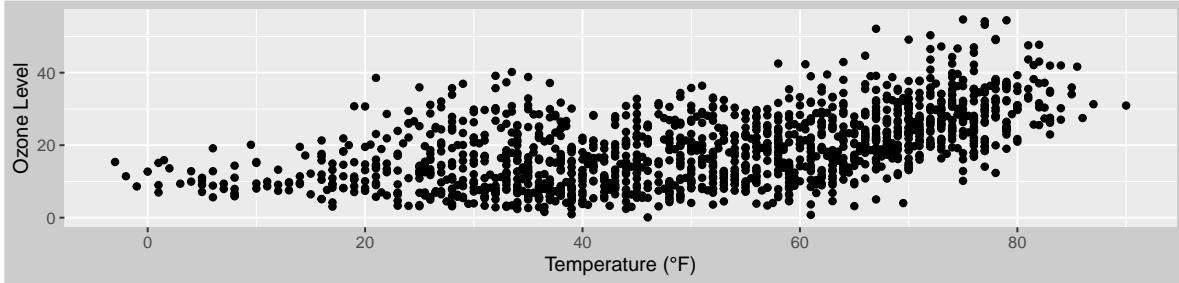
One can fix the aspect ratio of the Cartesian coordinate system and literally force a physical representation of the units along the x and y axes:

```
ggplot(chic, aes(x = temp, y = o3)) +
  geom_point() +
  labs(x = "Temperature (°F)", y = "Ozone Level") +
  scale_x_continuous(breaks = seq(0, 80, by = 20)) +
  coord_fixed(ratio = 1)
```



This way one can ensure not only a fixed step length on the axes but also that the exported plot looks as expected. However, your saved plot likely contains a lot of white space in case you do not use a suitable aspect ratio:

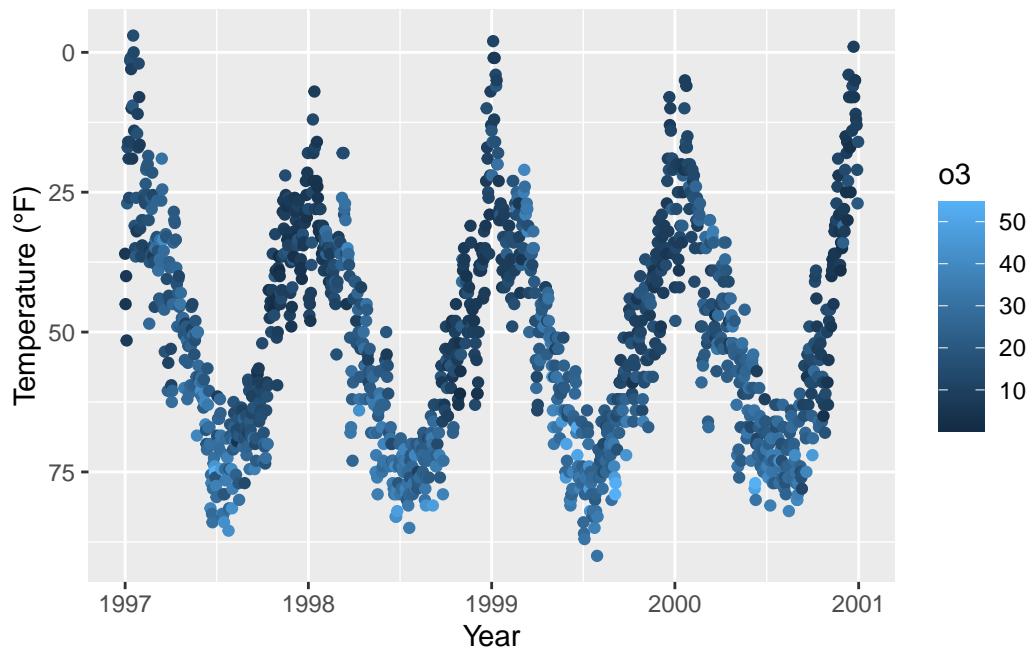
```
ggplot(chic, aes(x = temp, y = o3)) +
  geom_point() +
  labs(x = "Temperature (°F)", y = "Ozone Level") +
  scale_x_continuous(breaks = seq(0, 80, by = 20)) +
  coord_fixed(ratio = 1/3) +
  theme(plot.background = element_rect(fill = "grey80"))
```



14.3 Reverse an Axis

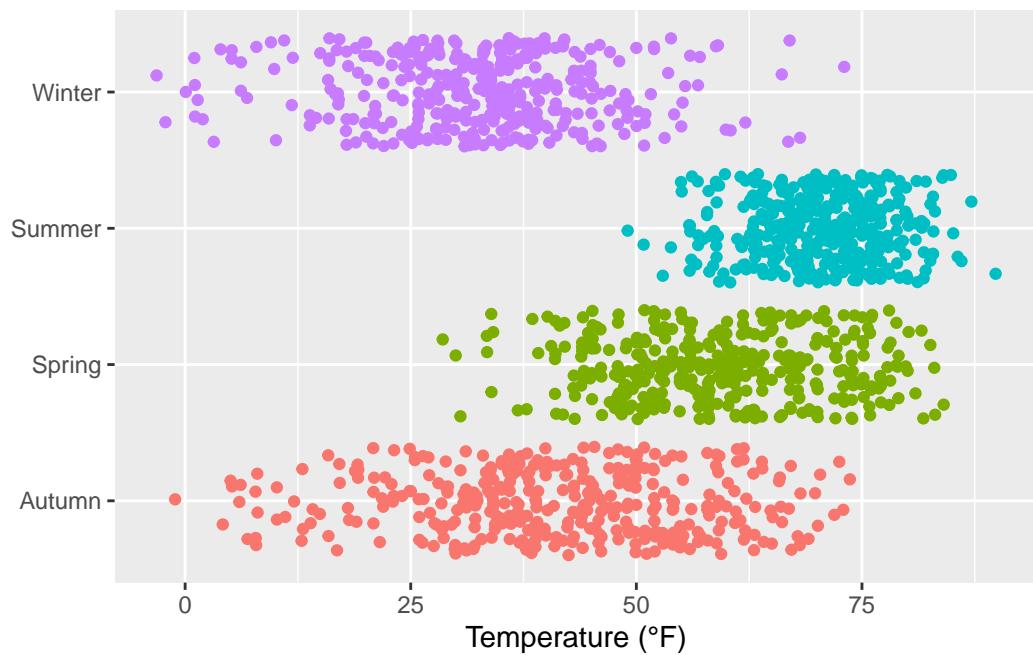
You can also easily reverse an axis using `scale_x_reverse()` or `scale_y_reverse()`, respectively:

```
ggplot(chic, aes(x = date, y = temp, color = o3)) +  
  geom_point() +  
  labs(x = "Year", y = "Temperature (°F)") +  
  scale_y_reverse()
```

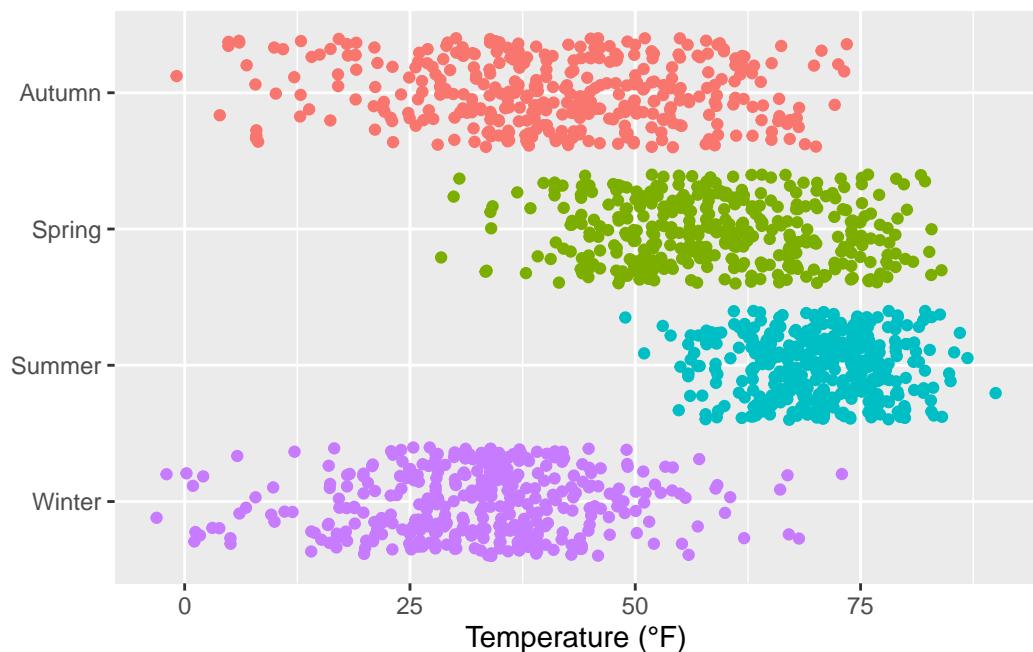


Note that this will only work for continuous data. If you want to reverse discrete data, use the `fct_rev()` function from the `{forcats}` package. Expand to see example.

```
## the default  
ggplot(chic, aes(x = temp, y = season)) +  
  geom_jitter(aes(color = season), show.legend = FALSE) +  
  labs(x = "Temperature (°F)", y = NULL)
```



```
library(forcats)
set.seed(10)
ggplot(chic, aes(x = temp, y = fct_rev(season))) +
  geom_jitter(aes(color = season), show.legend = FALSE) +
  labs(x = "Temperature (°F)", y = NULL)
```



14.4 Transform an Axis

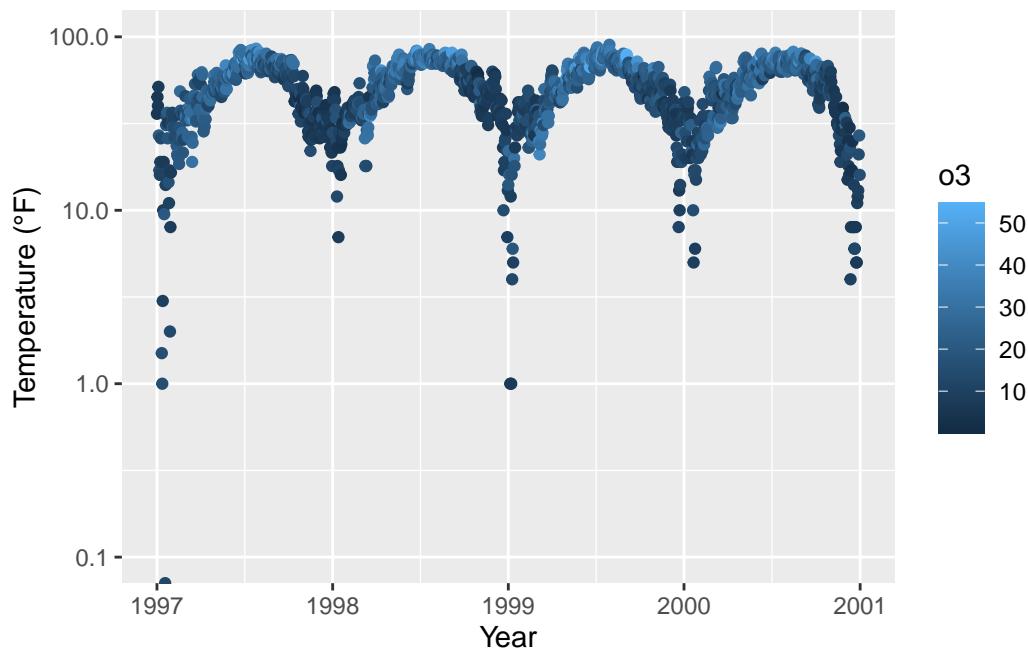
... or transform the default linear mapping by using `scale_y_log10()` or `scale_y_sqrt()`. As an example, here is a log10-transformed axis (which introduces NA's in this case so be careful):

```
ggplot(chic, aes(x = date, y = temp, color = o3)) +  
  geom_point() +  
  labs(x = "Year", y = "Temperature (°F)") +  
  scale_y_log10(lim = c(0.1, 100))
```

Warning in transformation\$transform(x): NaNs produced

Warning in scale_y_log10(lim = c(0.1, 100)): log-10 transformation introduced infinite values.

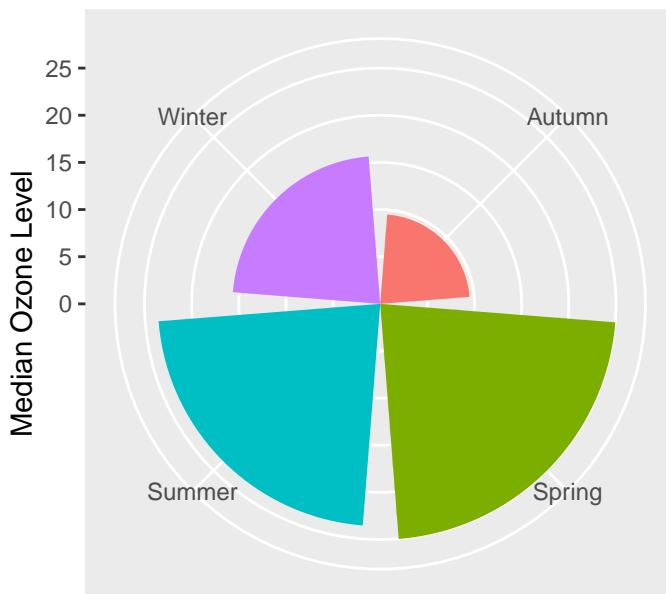
Warning: Removed 3 rows containing missing values or values outside the scale range (`geom_point()`).



14.5 Circularize a Plot

It is also possible to circularize (polarize?) the coordinate system by calling `coord_polar()`.

```
chic |>
  dplyr::group_by(season) |>
  dplyr::summarize(o3 = median(o3)) |>
  ggplot(aes(x = season, y = o3)) +
  geom_col(aes(fill = season), color = NA) +
  labs(x = "", y = "Median Ozone Level") +
  coord_polar() +
  guides(fill = "none")
```



This coordinate system allows to draw pie charts as well:

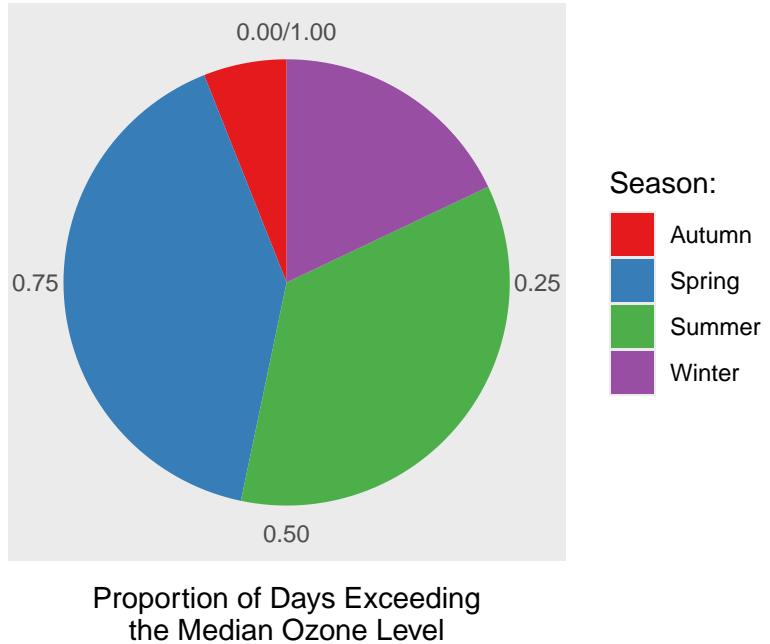
```
chic_sum <-
  chic |>
  dplyr::mutate(o3_avg = median(o3)) |>
  dplyr::filter(o3 > o3_avg) |>
  dplyr::mutate(n_all = n()) |>
  dplyr::group_by(season) |>
  dplyr::summarize(rel = n() / unique(n_all))

ggplot(chic_sum, aes(x = "", y = rel)) +
  geom_col(aes(fill = season), width = 1, color = NA) +
```

```

labs(x = "", y = "Proportion of Days Exceeding\nthe Median Ozone Level") +
coord_polar(theta = "y") +
scale_fill_brewer(palette = "Set1", name = "Season:") +
theme(axis.ticks = element_blank(),
panel.grid = element_blank())

```



I suggest to always look also at the outcome of the same code in a Cartesian coordinate system, which is the default, to understand the logic behind `coord_polar()` and `theta`:

```

ggplot(chic_sum, aes(x = "", y = rel)) +
  geom_col(aes(fill = season), width = 1, color = NA) +
  labs(x = "", y = "Proportion of Days Exceeding\nthe Median Ozone Level") +
  #coord_polar(theta = "y") +
  scale_fill_brewer(palette = "Set1", name = "Season:") +
  theme(axis.ticks = element_blank(),
  panel.grid = element_blank())

```

