

데이터 시각화

한국 R 사용자회

2022-05-03

Contents

데이터 시각화	5
1 헬로우 월드	7
1.1 나폴레옹 러시아 침공	7
1.2 시각화	9
1.3 참고서적	9
시각화 구성요소	11
2 글꼴	13
2.1 R 코딩 글꼴	13
2.2 ggplot 시각화 글꼴	14
2.3 showtext 패키지	16
2.4 로컬 글꼴 적용	17
3 구성요소	19
3.1 시각적 매핑(Aesthetics)	19
3.2 척도 매핑	19
3.3 좌표계	21
3.4 축(axis)	21
시각화	23

4	그래프 문법	25
4.1	그래프 문법의 존재이유	25
4.2	그래프 문법	25
4.3	ggplot 확장	30
5	시각화 패턴	31
5.1	라벨 붙은 시계열	31
5.2	막대그래프 그룹별 색상	32
5.3	추세선 강조 + 라벨	32
5.4	롤리팝(lolly-pop) 그래프	33
5.5	아령(dumbbell) 그래프	35
5.6	경사(Slope) 그래프	35
	사례	39
6	데이터 저널리즘	41
7	러시아 월드컵	43
8	코로나19	45

데이터 시각화

후원계좌

디지털 불평등 해소를 위해 제작중인 오픈 통계패키지 개발과 고품질 콘텐츠 제작에 큰 힘이 됩니다.

- 하나은행 448-910057-06204
- 사단법인 한국알사용자회

Chapter 1

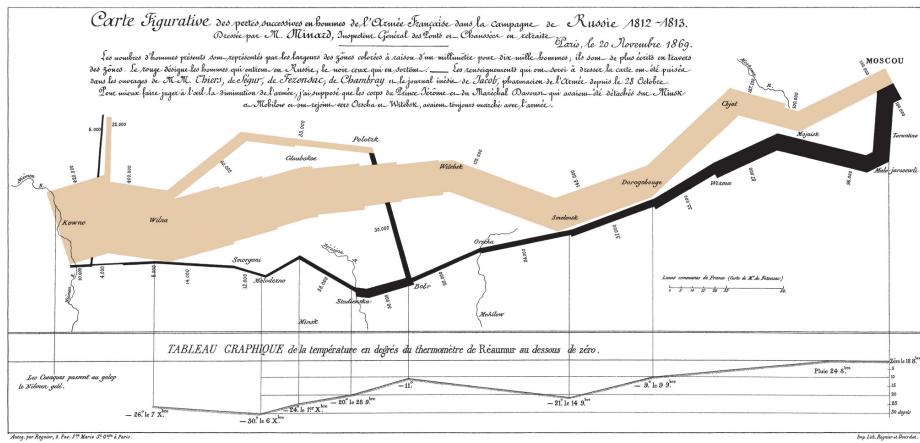
헬로우 월드

1.1 나폴레옹 러시아 침공

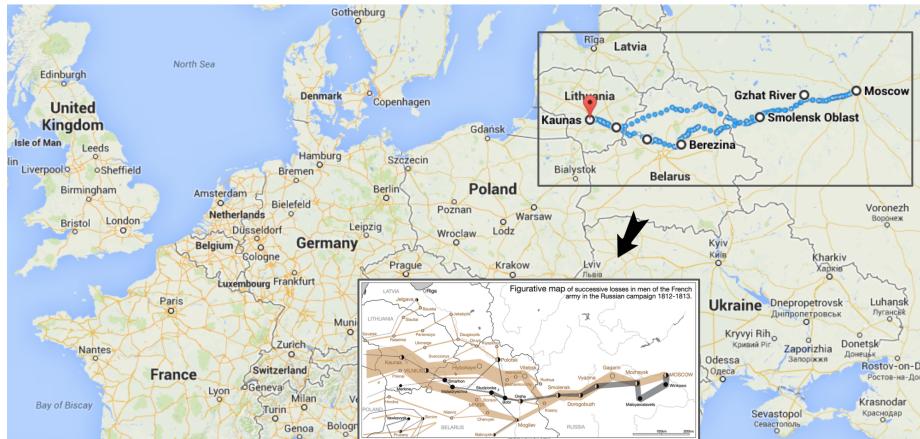
나폴레옹 황제가 프랑스를 통치할 때 1812년 최대 전성기를 구가했다.



최대 전성기를 구가하던 1812년 나폴레옹은 러시아 침공을 감행했다. 나폴레옹의 러시아침공은 데이터 시각화의 역작을 남기는 계기가 되기도 했다. 미나르는 프랑스의 러시아 침공을 군더더기 없이 시작부터 폐퇴하여 돌아온 과정까지 간략하게 표현했다.

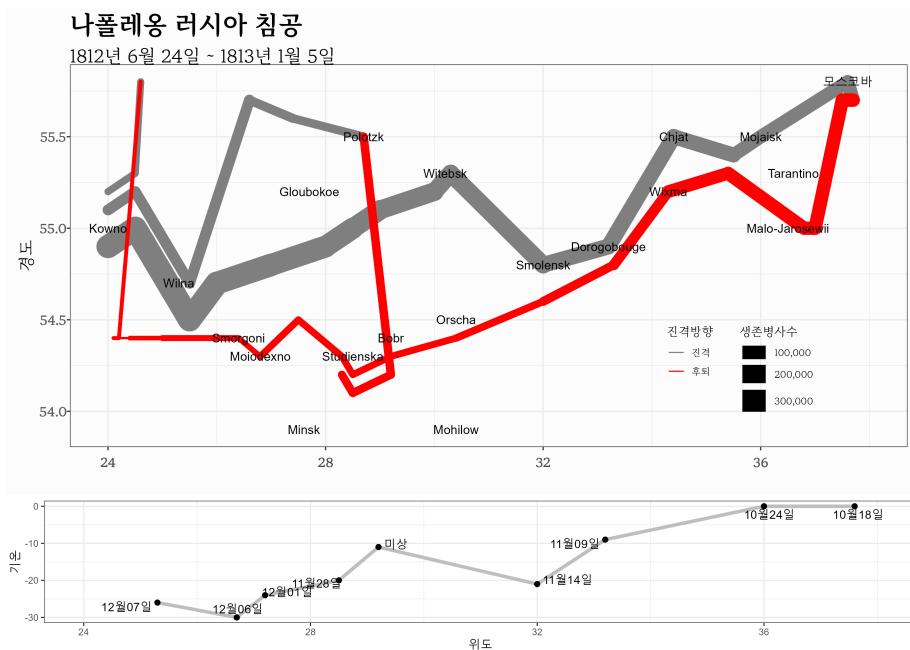


미나르 데이터 시각화의 지명을 자세히 보면 프랑스의 러시아 침공에 대해 대략적으로 인식하고 있는 것과 다소 차이가 난다. 통상 프랑스 수도 파리에서 나폴레옹 대군이 출발하여 러시아 모스크바에서 한동안 머물다가 다시 수도 파리로 돌아와서 황제에서 끌어내려져 엘바섬으로 위배를 떠난 것으로 알려져 있지만 실제 미나르가 데이터에 기반하여 제작한 시각화 그래프에는 진실이 담겨있다.¹



¹ Alexey Tereshchenko, "Why did Napoleon's Russian campaign end in failure?", Quora

1.2 시각화



1.3 참고서적

영어로 참고할 오픈 데이터 시각화 교재

- (Healy, 2018)
- (Dougherty and Ilyankou, 2021)
- (Wilke, 2019)

시각화 구성요소

데이터 시각화를 할 때 다양한 구성요소를 이해하고 이를 체계적으로 조합하여 시각화 객체를 제작해야만 이후 재사용가능한 시각화 산출물을 제작하여 지속적인 발전을 이뤄낼 수 있다.

Chapter 2

글꼴

R을 단순히 통계 언어로 생각하지 말고 적용범위를 확대해서 활용하면 데이터 과학 산출물을 다양한 전자문서로 제작하여 커뮤니케이션 할 수 있다. PDF, HTML, 워드 등 문서 뿐만 아니라, 파워포인트 같은 발표자료를 슬라이드로 제작하여 배포할 수 있다. 그래프 문법(Grammar of Graphics)에 따라 `ggplot` 시각화를 산출물에도 다양한 글꼴(font)을 반영하여 좀더 관심을 끌 수 있는 그래프 제작도 가능하다. 데이터 과학자나 개발자 관점에서도 통합개발환경(IDE)이 필요한데 개발과 저작에 집중할 수 있는 글꼴을 지정하여 활용할 경우 생산성도 높일 수 있고 좀더 쾌적한 환경에서 개발을 진행할 수 있다.

R 스크립트 작성을 위한 글꼴과 그래프에 한글 글꼴(font)을 적용한다. `ggplot`을 비롯한 시각화를 위해 `extrafont`와 `showtext` 패키지를 활용하여 적절한 한글 글꼴을 사용할 뿐만 아니라 코딩 개발할 때 R 스크립트(.R) 및 R마크다운(.Rmd)에서도 적절한 한글글꼴 사용을 위해서 코딩관련 글꼴도 설치한다.

기본적인 작업흐름은 운영체제에 먼저 외부에서 가져온 폰트를 설치한다. 그리고 나서 `extrafont` 팩키지 `font_import()` 함수를 사용해서 폰트를 R에서 불러 사용할 수 있도록 설치한다. 그리고 나서 `loadfonts()` 함수를 사용해서 글꼴을 `ggplot`등에서 불러 사용한다. 구글 글꼴을 사용하고자 할 경우 `showtext` 팩키지를 사용해서 로컬 컴퓨터에 설치하여 적용한다.

2.1 R 코딩 글꼴

문서를 위해 작성하는데 사용되는 글꼴과 R 코딩을 위해 사용되는 글꼴은 차이가 난다. 왜냐하면 R 코딩에 사용되는 글꼴은 가독성이 좋아야하고 디버깅에 용이해야 된다. 영어는 `consolas` 글꼴을 많이 사용하는데 무료가 아니다. 그래서 `consolas`에서 영감을 받은 SIL 오픈 폰트 라이선스를 따르는 Inconsolata가 R 코딩에 많이 사용되고 있다. 하지만, R코드를 작성할 때 주석을 한글로 달거나 R마크다운 작업을 할 경우 유사한 기능을 하는 한글 글꼴이 필요하다.

- 네이버 나눔고딕 코딩글꼴
- D2 Coding 글꼴

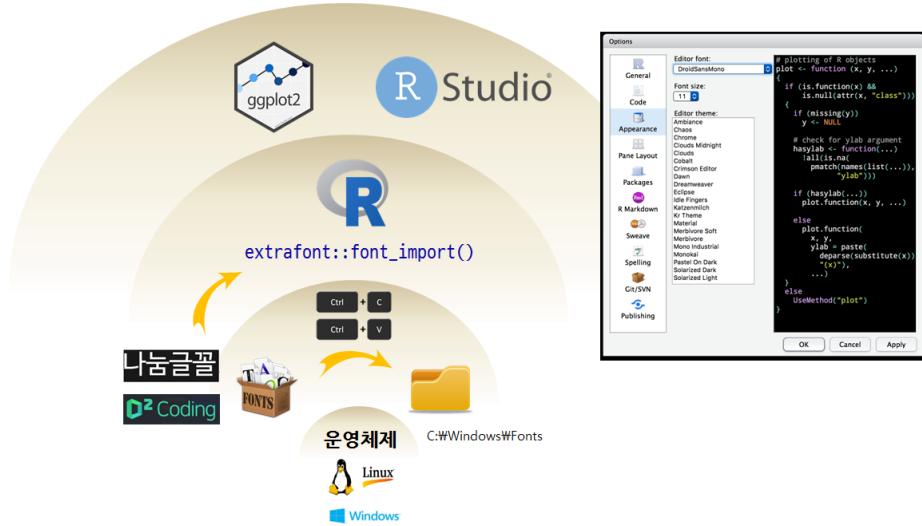


Figure 2.1: R 폰트/글꼴 설치

“네이버 나눔고딕 코딩글꼴”과 “D2 Coding 글꼴”을 설치하고 나서 RStudio IDE에서 “Tools” → “Global Options…”를 클릭하면 “Options”창에서 Appearance에서 **Editor font:**에서 설치한 코딩전용 글꼴을 선택하고 **Editor theme:**도 지정한다.

2.2 ggplot 시각화 글꼴

`extrafont` 패키지에서 `font_import()` 함수로 운영체제(윈도우/리눅스)에 설치된 글꼴을 R로 가져온다. 그리고 나서 `loadfonts()` 함수를 사용해서 설치된 글꼴을 사용하는 작업흐름을 따르게 된다.

2.2.1 ggplot 한글 글꼴 사례

`extrafont` 패키지 `loadfonts()` 함수를 사용해서 `ggplot`에서 적용시킬 수 있는 글꼴을 불러냈다. R 내장 데이터셋 `iris`를 사용하여 나눔글꼴 “Nanum Pen Script”을 기본 글꼴로 적용시켰다.

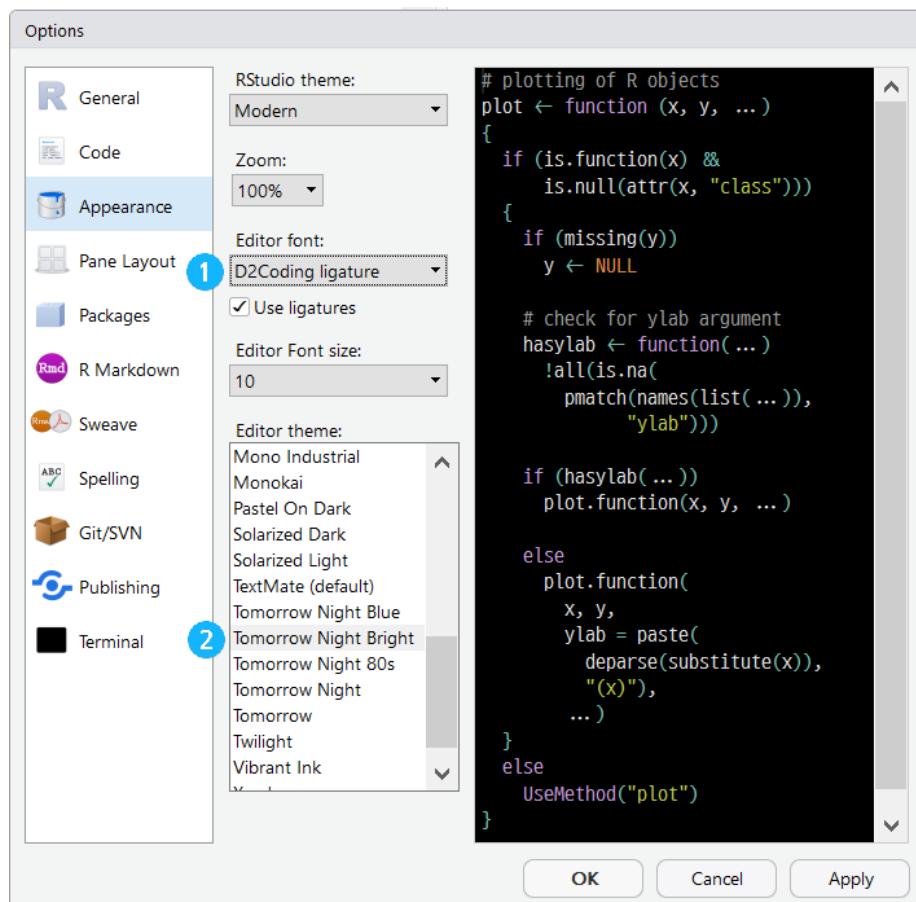
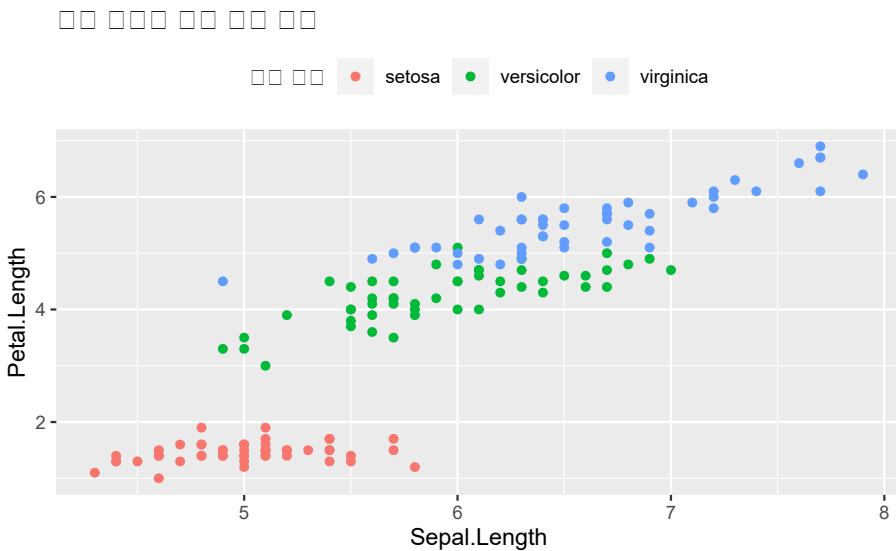


Figure 2.2: D2 코딩폰트 설치



2.3 showtext 패키지¹

`extrafont` 패키지를 통해 한자를 포함한 한글을 처리할 수 있었으나, `extrafont`는 트루타입폰트 (.ttf)를 PDF 그래픽 장치에 초점을 맞춰 개발이 되었다. 따라서, 데이터과학 최종산출물이 PDF 형태 책이 아닌 경우 여러가지 면에서 다양한 한글 글꼴을 표현하는데 있어 한계가 있다.

새로 개발된 `showtext` 팩키지는 `Ghostscript` 같은 외부 소프트웨어를 활용하지 않고도 다양한 (그래픽) 글꼴을 지원한다. `showtext`로 R 그래프를 생성할 때, 다양한 글꼴(TrueType, OpenType, Type 1, web fonts 등)을 지원한다.

과거 PDF와 같은 책형태로 정보를 공유하고 전달하는 방식이 주류를 이뤘다면 인터넷 등장 이후 웹으로 정보 생성과 소비가 주류로 떠오르게 되면서 글꼴에도 변화가 생겼다. 가까운 미래에는 웹을 우선시하는 글꼴이 대세를 이룰 것으로 보인다.

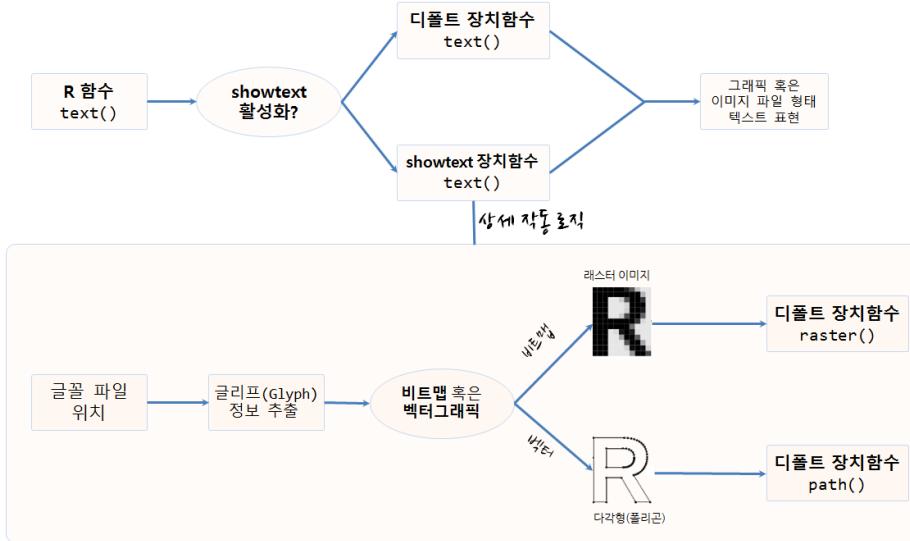
사용자가 그래프에 텍스트를 넣기 위해 R 함수에서 `text()`를 호출할 때 `showtext`가 활성화되어 있으면 `showtext` 팩키지 `text()` 함수를 호출해서 그래픽 혹은 이미지 파일에 텍스트를 표현하고 그렇지 않는 경우는 디폴트 장치함수 `text()` 함수를 호출하게 되어 있다.

내부적으로 상세 작동 로직은 글꼴 위치를 파악해서 글리프(glyph) 정보를 추출하고 비트맵 형식, 벡터그래픽 형식에 따라서 비트맵일 경우 `raster()` 장치함수를 호출하고, 벡터그래픽인 경우 `path()` 장치함수를 호출해서 기능을 수행한다.

2.3.1 R 설치 글꼴 확인

`extrafont` 팩키지 `loadfonts()` 함수를 통해 .ttf 파일 정보를 확인한다. 현재 구글 글꼴 페이지에서 많은 한글 글꼴을 지원하지 않고 있다. 구글에서 전세계 글꼴을 지원하다보 동아시아

¹ `showtext: Using Fonts More Easily in R Graphs`

Figure 2.3: `showtext` 글꼴

3국 대상으로 지원되는 글꼴은 적은 것으로 보인다.

2.3.2 ggplot 글꼴 적용

한글 글꼴을 바로 적용하기에 앞서 `showtext` 패키지 포함된 영문글꼴 적용 사례를 먼저 들려보자. `ggplot` 그래프에 적용되는 `showtext` 활용 기본 작업흐름은 다음과 같다.

1. 글꼴을 적재한다.
2. 그래픽 장치를 연다
3. `showtext`를 통해 텍스트를 표시한다고 지정한다.
4. 그라프를 그린다.
5. 장치를 닫는다.

2.4 로컬 글꼴 적용

로컬 컴퓨터에 저장된 `.ttf` 파일을 사용자 지정해서 가져온 후 이를 `ggplot`에 반영하여 한글을 R 그래프에 적용하는 것도 가능하다. `showtext`는 `extrafont` 보다 나중에 개발되어 `extrafont`가 로컬 컴퓨터에 설치된 글꼴을 `ggplot`에 구현되는데 전력을 다했다면 `showtext`는 이를 발판으로 나중에 개발되어 구글 폰트와 같은 인터넷 글꼴과 최근 웹출판에 대한 개념도 넣어 개발된 것이 차이점이다.

Chapter 3

구성요소

깔끔한 데이터(tidy data)가 준비되면 이를 시각화하기 위해서는 데이터를 시각화 객체에 매핑하게 되는 메커니즘이 필요하다. 주요 구성요소로 시각적 매핑(Aesthetics), 척도(Scale), 좌표계(Coordinate System), 축(Axis) 등이 필요하다.

3.1 시각적 매핑(Aesthetics)

ggplot에서 시각적 매핑(Aesthetics)은 `aes()` 내에 시각적으로 표현되는 모든 것을 담고 있다. 시각화 대상은 크게 두 가지로 나눠지는데 연속형과 범주형이다. 온도나 몸무게 같은 경우는 특정 두 데이터 값 사이 연속된 값이 있지만, 수학에서 정수와 같이 표현되는 쪼갤 수 없는 경우는 국가를 생각하면 한국과 미국 사이 존재하는 값이 없다. 시각화를 할 경우 위치(position), 모양(shape), 크기(size), 색상(color), 선굵기(line width), 선유형(line type)을 사용하여 좌표계(coordinate system), 척도(scale)와 결합하여 최종 시각화 결과물을 제작된다. Claus Wilke 시각화 책에 언급된 다양한 시각적 매핑을 통해 효과적으로 데이터의 정보를 표현할 수 있다.(Wilke, 2019) 당연히 위치, 모양, 크기, 색상, 선굵기, 선유형 중 어떤 것을 사용하느냐에 따라 시각화로 전달되는 정보의 양은 달라진다. 예를 들어, 위치는 시각적 변별력이 선 유형보다 크다.

3.2 척도 매핑

데이터가 주어지면 시각적 객체(모양, 색상, 크기 등)와 매핑을 해야 되는데 그 둘 사이를 연결하는 것이 척도(Scale)이다. x축과, y축 척도에 시각적 객체를 매핑해야 비로서 데이터 값을 그래프로 표현된다. 척도에 위치가 지정되면 시각적 매핑 객체를 다양하게 표현할 수 있다. 만약 위치가 척도에 모호하게 표현되게 되면 시각적 객체가 제대로 그래프에 표현되는데 문제가 된다.

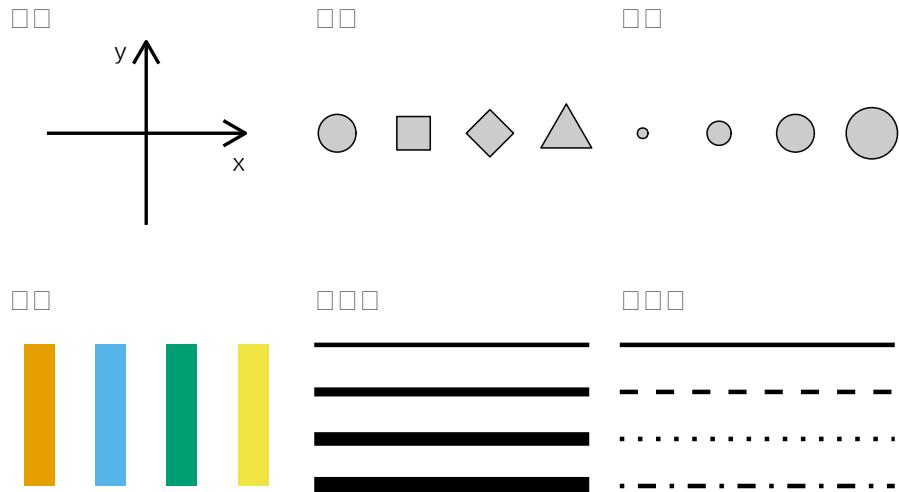


Figure 3.1: 위치, 모양, 크기, 색상, 선굵기, 선유형 등 시각적 매핑

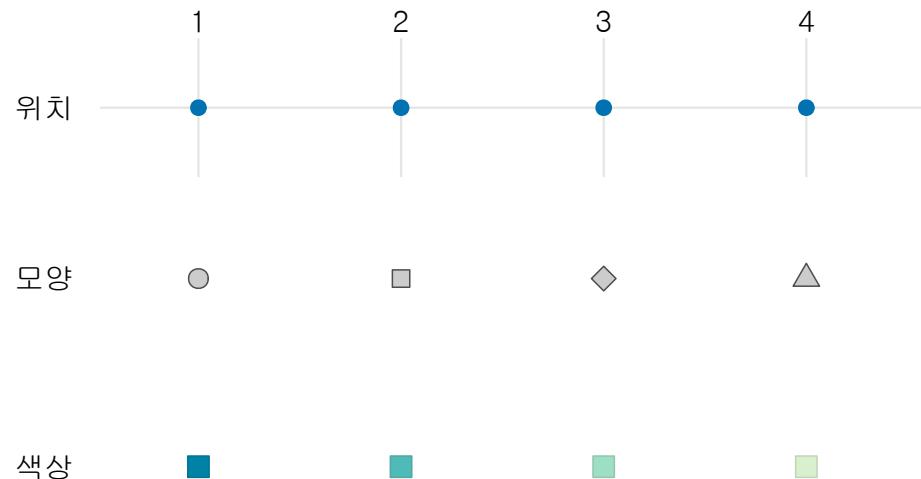


Figure 3.2: 시각적 객체 척도 매핑

3.3 좌표계

데이터 시각화에서 앞서 데이터를 척도와 시각적 객체로 준비를 했다면 이를 최종 그래프로 제작하기 위해서는 좌표계가 필요하다. 네이버 사전에서 좌표계는 공간상의 한 점의 위치를 표시하는 숫자들의 순서쌍인 좌표를 정하기 위한 체계로서, 원점과 기준 길이, 기준 축이나 기준선들의 집합을 통틀어 이르는 말이다. 가장 널리 사용되는 좌표계는 데카르트 직교좌표계와 극좌표계가 널리 사용된다. 다음은 `mtcars` 데이터 차량 기통수를 막대그래프와 원그래프를 사용하여 동일한 내용을 좌표계만 달리하여 시각적으로 표현해따.

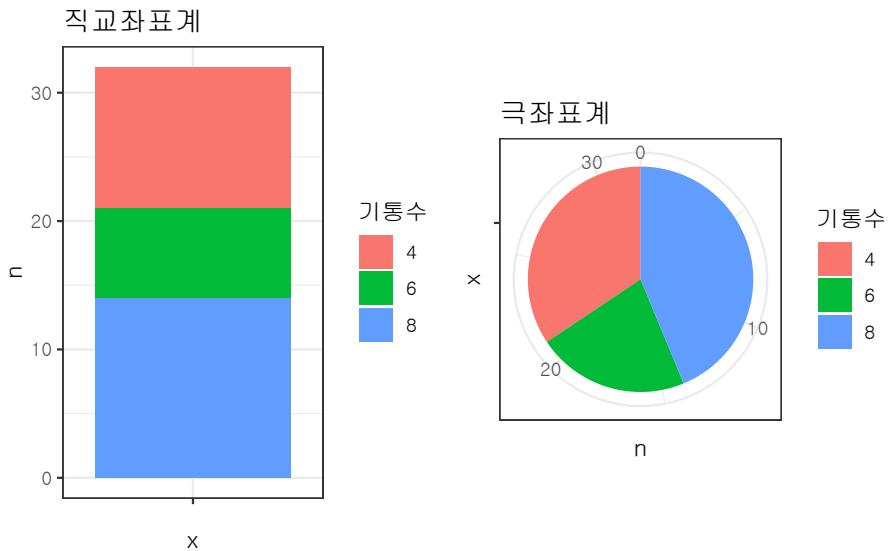


Figure 3.3: 직교좌표계와 극좌표계

3.4 축(axis)

데이터 값을 선형 척도에 매핑하여 시각화하는 것이 일반적이지만 x , y 축이 대표적으로 날짜와 같은 시계열 데이터인 경우 혹은 시분초를 나타내는 경우도 있어 이에 맞춰 적절히 축을 맞춰야 한다. 덧셈이 아니라 곱셈에 대해 선형인 경우 로그변환을 취하여 데이터 본연의 척도를 반영하도록 축을 조정한다. 많이 사용되는 로그 변환의 경우 밑을 10으로 하는 상용로그와 자연로그가 있어 명확히 축라벨에 적시하여 혼동을 피하는 것이 좋다. 로그 변환시 0이 있는 경우 문제가 되기 때문에 `sqrt()` 변환도 로그변환이 갖는 표현법의 장점을 갖추면서도 로그변환 시 생기는 번거러움을 해소하기 자주 사용된다.

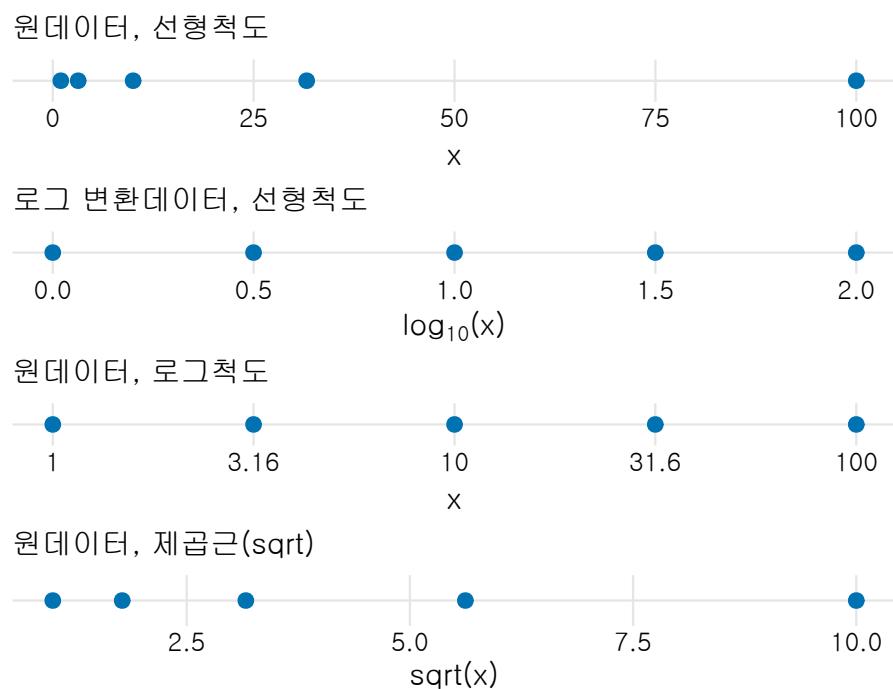


Figure 3.4: 선형, 로그, 제곱근 축

시각화

Chapter 4

그래프 문법

4.1 그래프 문법의 존재이유

Leland Wilkinson의 “The Grammar of Graphics”는 1999년 첫 출간된 이후 데이터 그래픽(data graphics)에 많은 영향을 주어 ggplot2, Polaris → Tableau, Vega-Lite 등의 형태로 우리곁에 다가섰다. 기존 데이터를 시각화한 다양한 그래픽 객체를 만들려면 각 그래픽 객체별로 따로 사용법을 익혀야만 되었다. 이것이 갖는 한계는 그래프 종류가 적은 경우 유용하지만 그래프 종류가 많아지면 매번 따로 사용법을 배워야되서 확장가능성이 무척 떨어지게 된다. 그래서, 이를 일반화한 무언가 필요한데 데이터 그래픽 객체를 분해해서 8개 계층으로 분해하여 조립하여 그래프를 제작하게 되면 앞선 문제를 일거에 해소할 수 있다. **그래프 문법(grammar of graphics)**을 통해 데이터를 가장 잘 표현할 수 있는 그래프를 생성할 수 있게 되었다.



Figure 4.1: ggplot이 필요한 이유

4.2 그래프 문법

그래프 문법은 총 8가지 층으로 구성되어 있는데 이를 각 층별로 나눠보면 다음과 같다.

4.2.1 데이터(Data)

그래프 문법 `ggplot`에 데이터는 깔끔한 데이터(tidy data)를 가정한다. 이를 위해서 기존 `wide` 형태 데이터는 key-value `long` 형태로 바꿔어 준비한다.

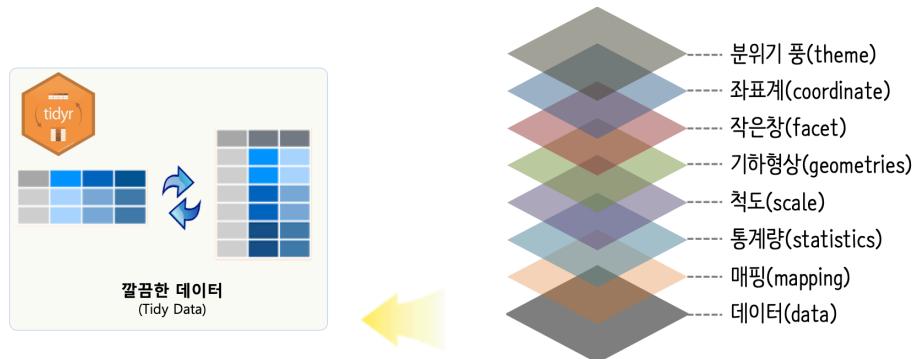


Figure 4.2: 깔끔한 데이터 wide, long 데이터

4.2.2 매핑(Mapping)

깔끔한 데이터가 준비되면 다음 단계로 칼럼에 해당되는 각 변수를 `aes()` 함수를 사용해서 `aes(x=x, y=y, color=z, ...)`와 같은 방식으로 데이터와 그래프를 매핑한다.

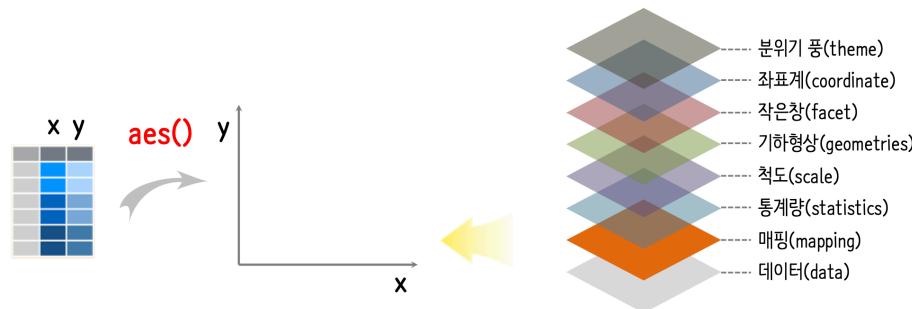


Figure 4.3: 데이터를 그래프에 매핑

4.2.3 통계량(statistics)

범주형 그래프를 시각화할 경우 빈도수를 통계량으로 계산해놔야 하고, 연속형 변수를 히스토그램으로 표현할 때도 마찬가지 방식으로 구간별 빈도수를 계산해놔야 하고, 특히 상자그림(boxplot)을 시각화할 경우 각 분위수는 물론이고 중위수도 및 interquartile도 계산해서 수염의 끝도 계산해놔야 제대로된 상자그림을 시각화할 수 있다.

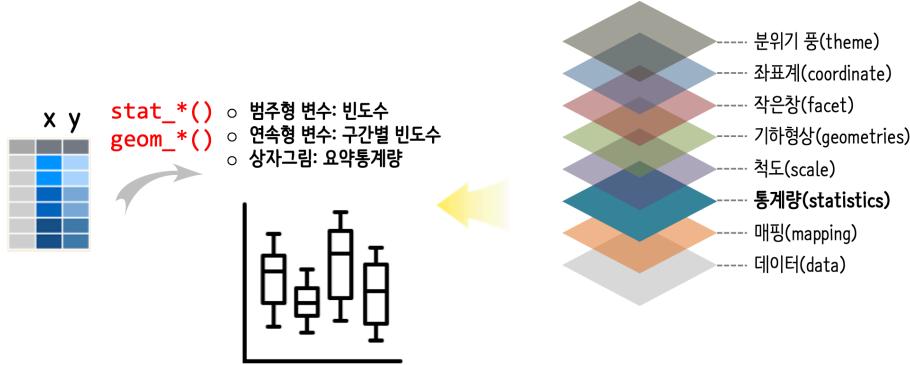


Figure 4.4: 그래프별 기본설정 통계량

4.2.4 척도(scales)

X축, Y축의 척도를 그래프에서 자동으로 인식하는데는 한계가 있어 이를 필요한 경우 적절한 형태로 설정한다. `scale_<x, y, color, fill, ...>_<유형>()` 구문을 갖는다. 예를 들어 Y축을 담당하는 변수가 로그척도(log)가 적합한 경우 이를 `scale_y_log10()`와 같이 변수를 특성을 반영한 척도를 설정한다. 변수가 날짜나 시간인 경우 `scale_x_date()`, `scale_x_datetime()`을 활용하여 적절한 형태로 설정한다.

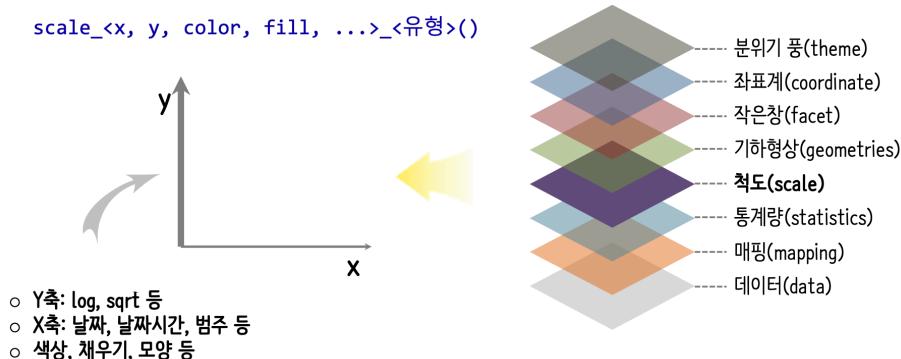


Figure 4.5: aes 매핑된 축 척도지정

4.2.5 기하형상(geometries)

예를 들어, 범주형 변수를 `aes()`로 지정하고 이를 적절한 그래프로 표현하기 위한 결정과정으로 `geom_*`() 방식으로 원그래프, 막대그래프, 겹그래프 등으로 변수를 시각화 객체로 지정한다.

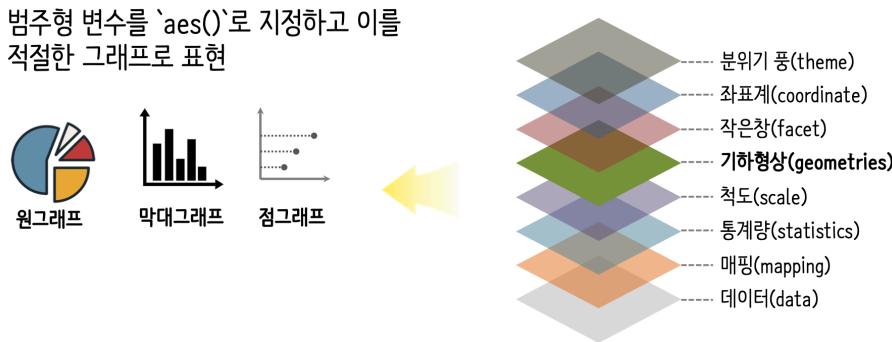


Figure 4.6: 기하형상 지정

4.2.6 작은 창(facet)

원본 데이터를 그룹으로 쪼개 작은 창에 동일한 시각화 객체를 표현하는 방법으로 다차원 데이터를 차원별로 나눠 볼 수 있다. 중요한 점은 각 작은 창이 동일한 유형의 그라프라는 점이 중요하다.

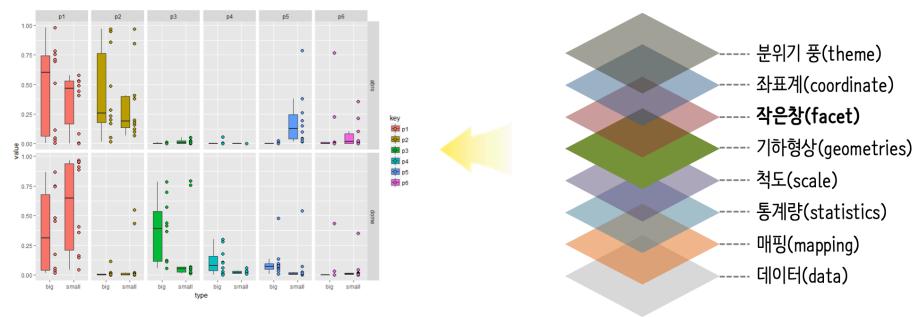


Figure 4.7: 그룹으로 쪼갠 작은 창(facet)

4.2.7 좌표계(coordinate)

깔끔한 데이터를 그래프에 매칭하여 시각화한 후에 경우에 따라서 좌표계를 변경할 경우가 있다. `coord_fixed()`, `coord_polar()`, `coord_flip()` 등을 사용해서 특정 영역 확대, 데카르트 좌표계에서 극좌표계, X-Y 축 변경 등의 작업을 수행할 수 있다.

4.2.8 분위기 풍(theme)

분위기 풍(theme)은 앞선 깔끔한 데이터를 시각화 객체로 변환시키는 과정과 아무런 연관이 없다. 대신 외양을 보기좋게 하는 역할을 수행한다. 즉, 시각화에 알맞는 색상 팔레트를 적용시키고, 글꼴을 바꾸는 작업이 여기에 해당된다.

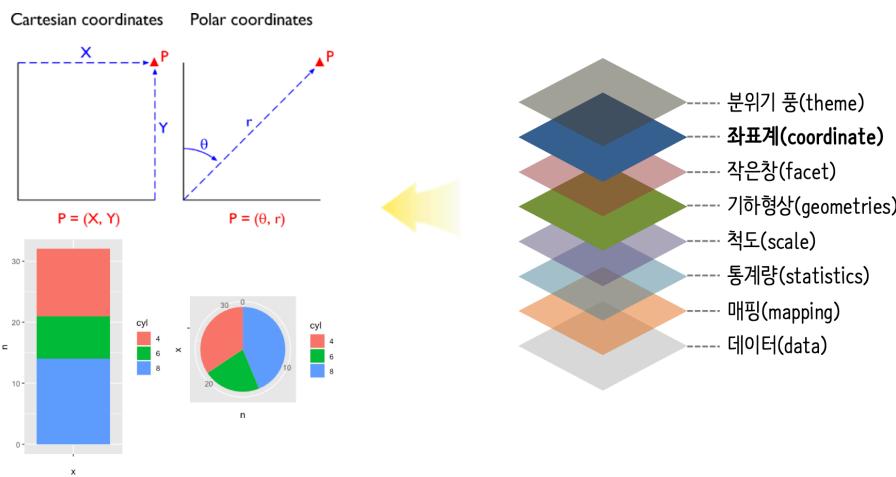


Figure 4.8: 좌표계 변환

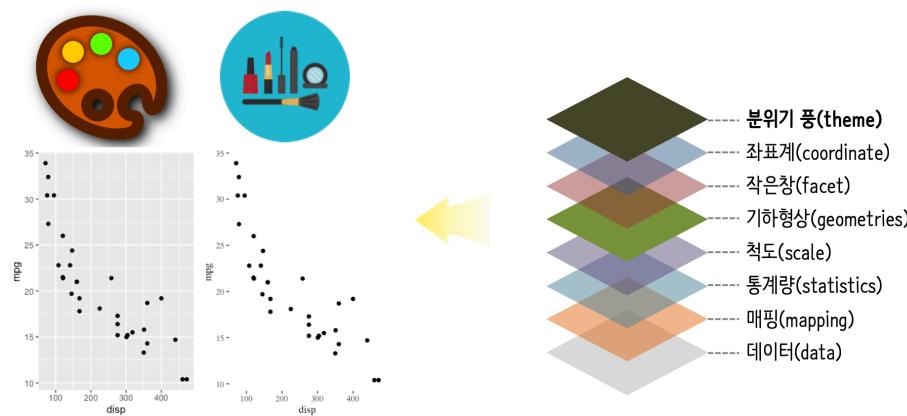
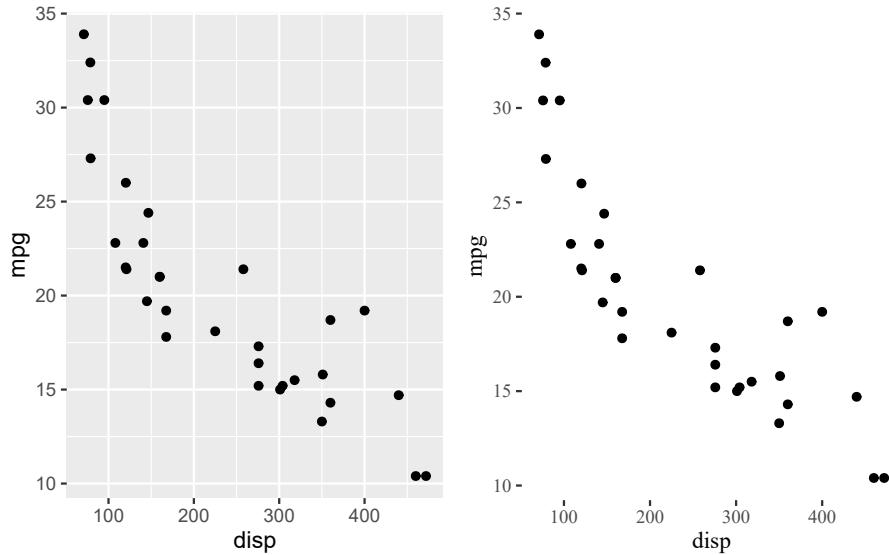
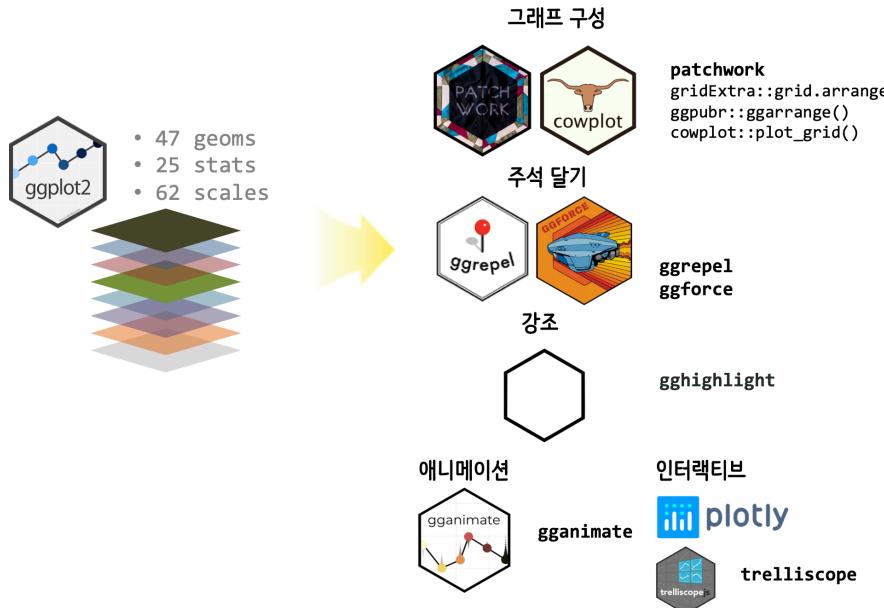


Figure 4.9: tufte 분위기 품(theme)



4.3 ggplot 확장

기본 ggplot 그래프 객체를 갖게 되면 2개 이상의 그래프를 합치거나 배열을 달리하고, 강조를 하고 주석(annotation)을 달고 애니메이션과 인터랙티브 기능을 추가하여 확장시킬 수 있다.



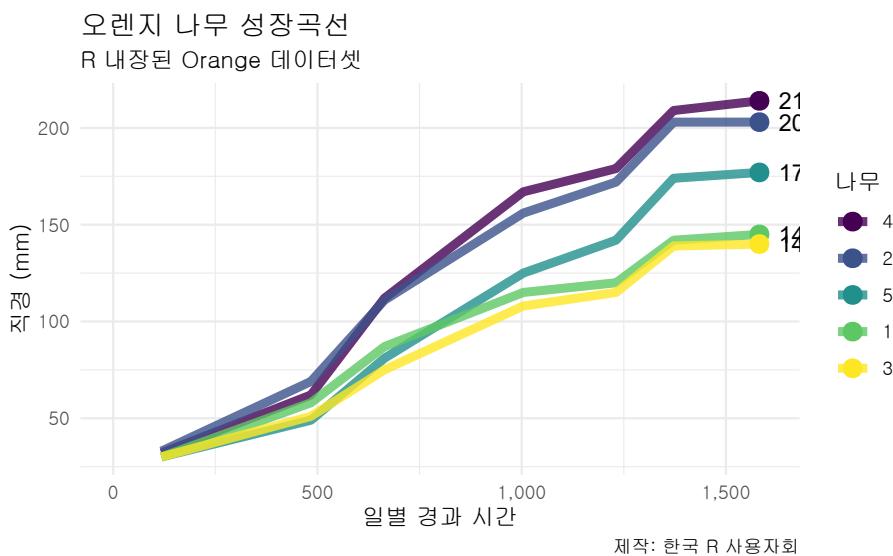
Chapter 5

시각화 패턴

5.1 라벨 붙은 시계열

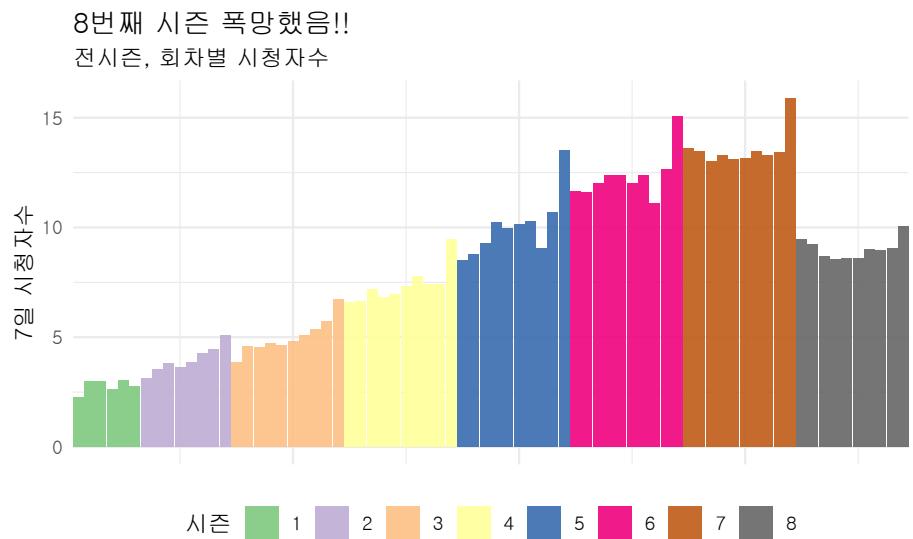
시계열 데이터를 제작하게 되면 추세를 파악할 수 있지만 결국 그래서 가장 최근 값이 어떻게 되는지 관심이 많다. 이런 사용자 요구를 맞추는데 시계열 데이터 마지막 시점에 라벨값을 붙이게 되면 가독성도 좋아진다. 기본적인 작업흐름은 데이터셋에서 가장 최근 관측점을 뽑아서 별도 데이터프레임으로 저장하고 이를 `geom_text()` 혹은 `geom_text_repel()` 함수를 사용해서 해결한다.

BLOGR 님이 작성한 Label line ends in time series with ggplot2 코드를 참조하여 `ggplot`으로 코드를 작성한다.



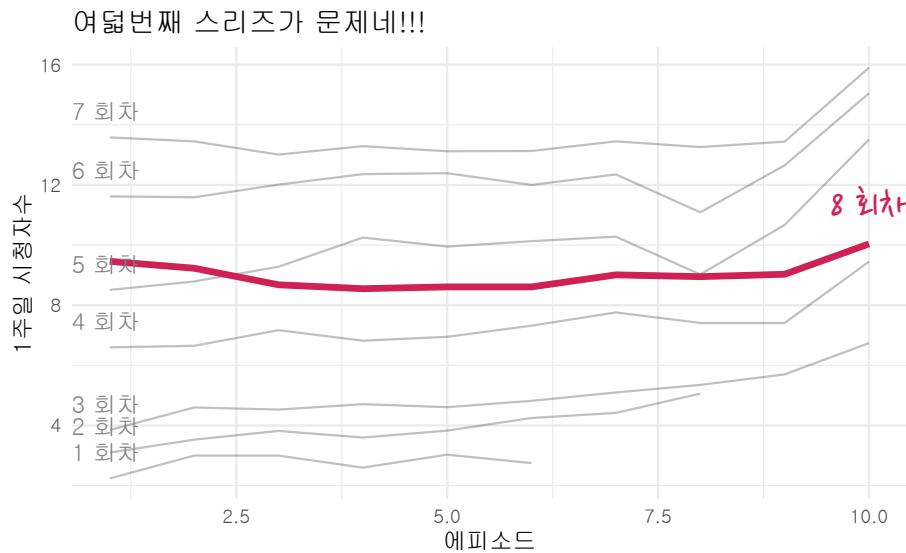
5.2 막대그래프 그룹별 색상

RStudio를 거쳐 IBM에서 근무하고 있는 Alison Presmanes Hill 의 GitHub 저장소에 공개된 TV 시리즈 데이터를 사용해서 막대그래프를 작성할 때 그룹별 색상을 적용하여 가시성을 높인다. TV 시리즈별 색상을 달리할 경우 RColorBrewer 패키지 생상 팔레트를 범주형에 맞춰 각 시리즈별로 가장 잘 구분될 수 있도록 색상을 칠해 시각화를 한다.



5.3 추세선 강조 + 라벨

시각화의 백미는 아무래도 대조와 비교를 통해 강한 인상을 주는 것이다. 앞선 `ratings` TV 시리즈 시청자 평가 데이터를 대상으로 추세선에 강조를 넣고 라벨 텍스트도 넣어 하이라이트 강조 그래프를 작성해보자. `geom_line()`을 두개 포함시켜 강조하고하는 색상을 별도로 지정하고 선굵기도 달리한다. 라벨도 동일한 방법으로 `geom_text()`을 두개 포함시켜 강조하고자하는 색상과 글꼴크기도 달리 지정한다.

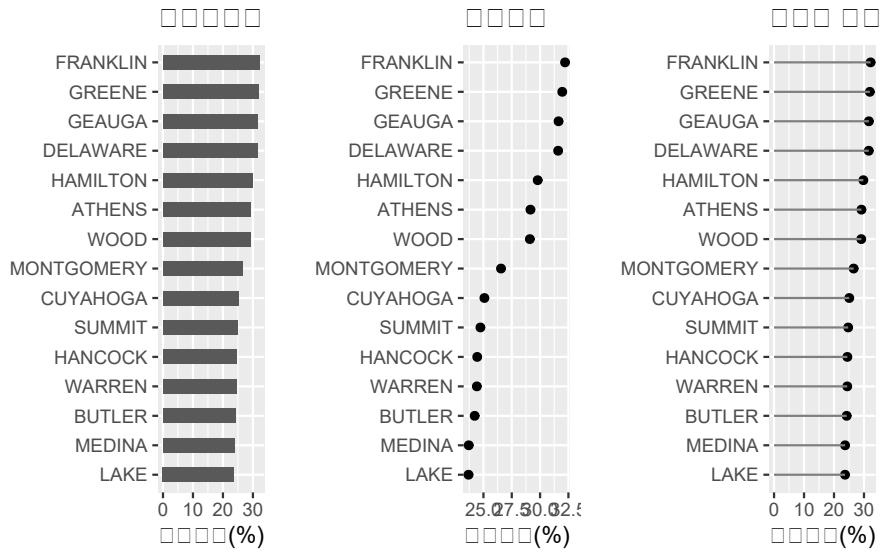


5.4 롤리팝(lolly-pop) 그래프

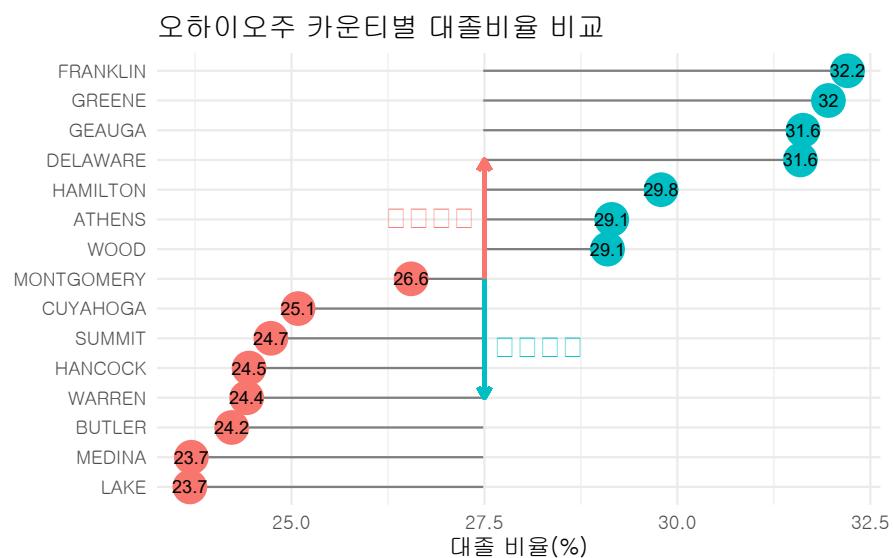
롤리팝(Lollipop) 사탕 그래프는 막대그래프와 클리블랜드 점그래프를 합성한 것으로 한축에는 연속형, 다른 한축에는 범주형을 두고 사용자의 관심을 점그래프로 집중시키는데 효과적이다. 단순히 막대그래프를 제작하는 것과 비교하여 임팩트있는 시각화를 가능하게 한다.

제작순서는 막대그래프 → 점그래프 → 롤리팝 그래프로 뼈대 골격을 만들어 나간다. 대략 골격이 제작되고 나면 외양과 필요한 경우 값도 텍스트로 넣어 시각화 제품을 완성한다. 롤리팝 사탕 그래프를 작성할 때 `geom_point()`을 사용해서 롤리팝 사탕을 제작하고, `geom_segment()` 함수를 사용해서 사탕 막대를 그린다. 이때 막대 사탕의 시작과 끝을 시작은 `x`, `y`에 넣어주고 끝은 `xend`와 `yend`에 넣어 마무리한다.

데이터는 ggplot2에 내장된 `midwest` 데이터를 사용하자. `midwest` 데이터셋은 2000년 미국 중서부 센서스 데이터로 인구통계 조사가 담겨있다. `percollege` 변수는 카운티(우리나라 군에 해당) 별 대학졸업비율을 나타낸다.

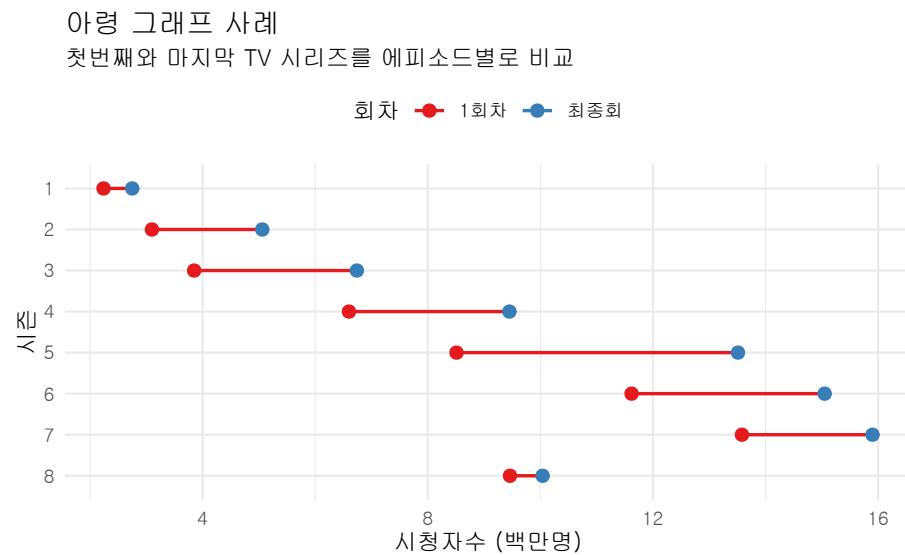


한발더 나아가, 평균값에서 얼마나 차이가 있느냐를 롤리팝 그래프로 시각화하는 패턴이 많이 사용된다. 이를 위해서, 앞서와 마찬가지로 15개 카운티를 뽑아내고 평균을 구하고 평균이하에 대한 요인(factor)도 함께 만들어낸다. 반영한다.



5.5 아령(dumbbell) 그래프

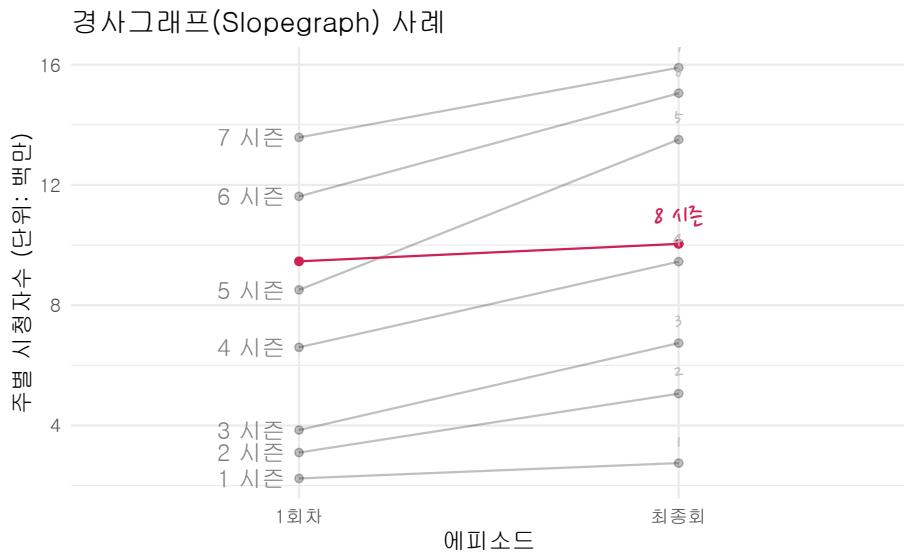
두 시점을 비교하여 전후를 비교한다던가 두 지역을 비교할 때 아령 그래프는 매우 효과적이다. TV 시리즈별로 회차를 달리하여 첫번째와 가장 마지막 시청자수를 비교하여 시각화하는데 아령(dumbbell) 그래프가 적절한 예시가 될 것으로 보인다. 이를 위해서 `ggplot()`에 들어가는 자료형을 미리 준비하고 이에 맞춰 `geom_line()`과 `geom_point()`를 결합시켜 시각화한다.



5.6 경사(Slope) 그래프

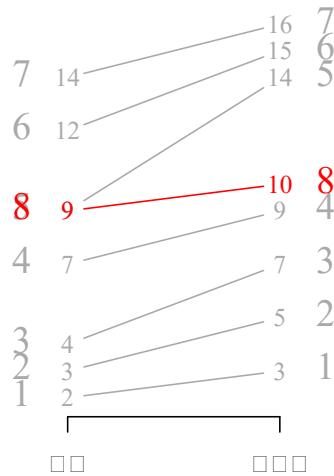
아령 그래프를 제작한 동일한 데이터를 터프티(tufte) 스타일 경사그래프로 구현하면 시즌별 첫회와 최종회 시청자수 비교를 좀 더 직관적으로 만들 수 있다.

`ggplot`의 기본기능을 활용하여 경사그래프를 시각화하고 강조하고자 하는 시즌을 색상을 달리하여 표현한다. 이를 통해 1~7번째 시즌은 1회차 시청률은 낮으나 최종회는 높게 마무리된 것을 알 수 있고, 더불어 시즌이 진행될 수록 1회차 시청률도 높아지고 있었다. 하지만 8번째 시즌은 다른 시즌과 달리 낮게 시작했고 최종회 시청률도 크게 나아지지 않은 것을 한눈에 파악할 수 있다.

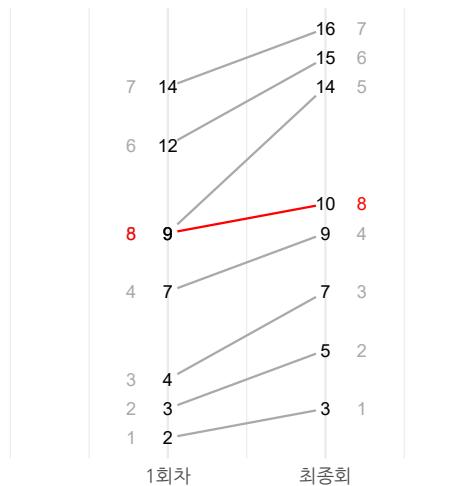


경사그래프를 제작하고자 하는 심으나 전반적으로 시간이 더 필요하신 분을 위해 `slopegraph` 패키지가 있다.

`slopegraph`는 Base 그래픽을 기본으로 삼고 있어 자료구조도 `rownames`를 갖는 전통적인 데이터프레임이다. 기본 Base 그래픽을 염두에 두고 상기 TV 연속물 경사그래프를 다음과 같이 작성할 수 있다.



`slopegraph()` 함수 대신 `ggslopegraph()` 함수를 사용하게 되면 `ggplot()`으로도 시각화를 할 수 있다. `slopegraph()` 함수는 자료구조가 직관적이라 처음 시각화를 하는 분에게 적절한 듯 보인다. 따라서, 앞서 `ggplot` 기반 경사그래프를 제작하고자 하는 경우 `ggslopegraph()`을 통해서도 `ggplot` 나머지 기능을 그대로 적용 가능하다.

경사그래프 사례 - `ggplot`

사례

Chapter 6

데이터 저널리즘

Chapter 7

러시아 월드컵

Chapter 8

코로나19

- GitHub 저장소 : clauswilke/dataviz

Bibliography

- Dougherty, J. and Ilyankou, I. (2021). *Hands-On Data Visualization*. O'Reilly Media.
- Healy, K. (2018). *Data visualization: a practical introduction*. Princeton University Press.
- Wilke, C. O. (2019). *Fundamentals of data visualization: a primer on making informative and compelling figures*. O'Reilly Media.