군 사 과 학 연 구 지
제14권 제1호(2021년 6월)

간소화된 그래픽 문법(Grammar of Graphics) 기반 데이터 시각화

Data Visualization Based on Lean Grammar of Graphics

윤봉규[1)]
Bong Kyoo Yoon

ABSTRACT

Grammar of Graphics(GoG) is a useful way for data visualization as is grammar of a language for communication. GoG is composed of several components (or procedures) such as data transformation and geometric shape of graphics. Since Wilkinson(1999) proposed the concept of GoG with seven orthogonal components, it has been adpated by several authors including Wickham(2010) who turned GoG into ggplot2 library of the statistical package R. As Wilkinson and Wickham pursued the completeness of explanation for GoG, their works are overwhelming for practitioners. We provide an alternative lean version of GoG focusing on core components to facilitate understanding of GoG among practioners. We also show how the concept of GoG is implemented in creating graphics with examples in ggplot2 of R.

Keyword: Grammar of Graphics(GoG), ggplot2, data visualization

논문접수일 : 2021년 4월 16일, 심사일 : 2021년 5월 10일, 게재확정일 : 2021년 5월 21일
1) 국방대학교 관리대학원 운영분석전공

1. 서 론

한 번 본 그림은 3일 뒤에 65% 정도가 기억에 남아 있지만, 문자로 된 정보는 10% 정도만 기억이 난다. 사람의 뇌에서 시각 정보를 처리하는 뉴런이 청각, 후각, 미각, 촉각을 처리하는 나머지 모든 감각 처리 뉴런보다 많다(Medina, 2009). 이 점이 시각 정보가 머리속에 더 오래 남는 이유에 대한 설명 중의 하나이다. 이런 이유로 데이터 (또는 자료) 시각화(Data Visualization)는 정보를 효율적으로 전달하는 방법으로 오랫동안 활용되어 왔다.

데이터 시각화의 의의를 극적으로 보여주는 사례는 '데이터사우르스 12(Datasaurus Dozen)'이다(Sarkar, 2018). 이 자료는 공룡, 별 등 12개의 서로 다른 그림을 만드는 점들의 x, y 좌표 값으로 구성되어 있다. 이 자료는 평균(Mean), 분산(Variance), 상관관계(Correlation)가 모두 동일하므로 테이블 형태의 정보로는 차이를 구분하기 어렵다. 그러나 공룡 그림과 별 그림을 구분하지 못할 사람은 거의 없을 것이다. 조금 덜 극적이지만 시각화의 중요성을 드러내면서 통계학의 역사에서도 중요한 의미를 가지는 사례는 Cleveland(1993)의 연구이다. 이에 대한 데이터 시각화 측면의 의의는 박동련(2011)에 상세하게 소개되어 있다.

데이터 시각화의 필요성은 누구나 공감할 수 있지만 여기에 일정한 규칙이 있고 이를 사용하면 효율적인 분석 그래프를 만들 수 있다는 주장에는 선뜻 공감하기 어려울 수 있다.

모든 언어는 문법이라는 규칙을 가지고 있다. 언어뿐 아니라 그림이나 음악도 표현을 위한 일정한 기법과 규칙이 있다. 이런 규칙을 아는 것이 언어를 통해 의사를 전달하고 그림을 그리거나 노래를 부르는데 꼭 필요한 것은 아니다. 그러나 이런 규칙을 아는 것은 효율적인 의사표현과 좋은 그림을 그리고 노래를 잘 부르는 데 도움이 된다.

언어, 그림, 음악은 모두 아이디어나 의미를 전달하는 수단이다. 데이터 시각화 또한 아이디어나 의미를 전달하기 위한 수단이다. 이런 점에서 언어와 다를 바가 없으며, 일반적인 규칙이 없을 리가 없다. 이 점에 착안해서 데이터 시각화의 공통 원칙을 모아 놓은 것이 그래픽 문법(Grammar of Graphics, GoG)이다.

GoG는 Wilkinson(1999)에 의해 처음 제시된 이후 다양한 사람들에 의해서 응용되면서 발전해 왔다. 특히 Wickham(2010)은 겹쳐 그리기를 응용한 계층적 접근법 (Layered Approach)의 개념을 제시했으며, 이를 자료처리 패키지인 R에 ggplot2 라이브러리로 만들었다. 한편, ggplot2에 2가 붙어 있는 것은 R의 기본 plot과 구분하기 위한 것이며 R에서 패키지명은 ggplot2를 사용하지만, 실제 명령어는 'ggplot'을 사용한다.

Wilkinson(1999, 2012)과 Wickham(2010)은 GoG의 개념에 대한 훌륭한 설명과 이를 컴퓨터 패키지에 구현하는 방법을 제시했음에도 불구하고, 데이터 시각화에 관심이 있지만 전산 관련 배경지식이 깊지 않은 일반적인 사람들이 이해하기에는 지나치게 상세한 설명이다. 그 결과 이들의 연구는 GoG의 전체적인 개념을 이해하고 활용하기에는 한계가 있다. 한편, GoG에 대해 소개하는 다른 자료들은 저자의 전문 분야와 시각에 따라서 GoG의 구성요소를 이리저리 다른 방식으로 분류하고 제시해서 GoG 개념 이해와 활용을 더욱 어렵게 하는 경우가 많다. 예를 들어, Sarkar(2018)는 지금까지의 GoG 개념을 종합하여 잘 정리했음에도 Wilkison과 Wickham의 설명과는 용어와 구성요소 구분에 차이가

있어 처음 GoG를 접하는 사람에게는 혼선을 줄 여지가 있다.

GoG가 데이터 시각화를 위한 일반적 규칙을 통해 시각화 작업에 도움을 주기 위한 것임에도 불구하고 개념 설명에서부터 혼선이 발생하는 것은 모든 구성요소를 똑같은 비중으로 설명하고, 연구자의 분야에 따라 구성요소를 나누는 분류 기준을 달리하면서 생긴 현상이다. 본 연구에서는 이를 개선하여 데이터 시각화에 반드시 필요하고, 여러 분야에서 공통적으로 제시되는 구성요소를 선별해서 '간소화된 GoG(Lean GoG)'를 소개한다.

본 연구에서는 그래프를 구성하기 위한 필수 요소를 중심으로 GoG를 설명하고 이를 R의 ggplot2 패키지로 구현하는 방법을 살펴본다. 이를 통해 현장의 실무자들이 GoG를 이해하고 데이터 시각화를 용이하게 활용할 수 있는 기반을 제공하고자 한다. 이를 위해 2장에서는 GoG의 개념과 핵심 구성요소를 도출하고, 이를 ggplot2로 구현해서 그래프로 그리는 간단한 사례를 살펴본다. 3, 4장에서는 GoG의 핵심 구성요소를 상세히 설명하고 이를 ggplot2에서 실행하는 방법을 살펴본다.

2. 그래픽 문법(GoG)과 ggplot2

Wilkinson(1999)은 데이터 시각화를 위한 구성요소와 절차를 그래픽 문법(Grammar of Graphics, GoG)이라는 개념으로 제시하며, 그래프를 그리는 것이 표현하고자 하는 정보와 이를 표현하는 사람에 따라서 형식과 표현 방식이 크게 달라지지만 공통의 규칙이 있음을 밝혔다. Wilkison은 데이터 시각화의 구성요소와 절차를 체계화하고, 더 나아가 각 절차를 수학적으로 표현 가능하다는 주장을 했다(Wilkinson, 2012). 그에 따르면 데이터 시각화는 데이터의 변수(Variable)를 연산(Algebra)을 통해 컴퓨터 화면에 출력 가능한 형태로 바꾸고(Scales), 필요한 정보를 요약(Statistics)해서 다양한 기하학적 형태(Geomtry)로 좌표축(Coordinates)에 인식 가능한 형태로 표현(Aesthetics)하는 과정이다. 각 구성요소를 만드는 과정에서 데이터(Data)가 변수집합을 거쳐 그래프(Graphics)로 바뀐다.

Wickham은 Wilkinson의 주장을 컴퓨터를 이용한 시각화의 관점을 강조해서 재정리 했다. 그는 데이터 시각화의 구성요소를 그래프를 그리기 위한 데이터(Data), 이를 좌표축에 대응시기는 매핑(Aesthetics), 기하학적 모양(Geometry), 통계적 변환(Statistical Transformation), 그래프 레전드와 같은 설명(Annotation), 변수 변화에 따라 여러 개의 그래프를 그리는 분할(Facet)의 구성요소/절차로 나누어 설명했다(2010, 2017). 한편, Wickham은 GoG를 데이터 처리(통계) 프로그램인 R에 ggplot2 패키지(Package)로 구현했다. 이 과정에서 Wickham은 계층적인 그리기 방법(Layered Approach)을 GoG와 결합하여 GoG의 개념을 절차적으로 정교화했다.

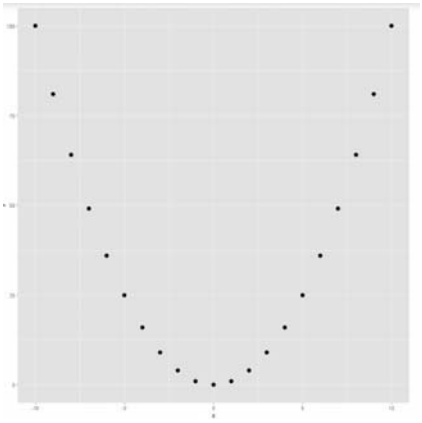
계층적 그리기 방법은 그림을 그릴 때 배경, 등장인물, 색깔 입히기를 개별적으로 그리고 이를 겹쳐서 완성된 그림을 만드는 방식이다. 투명 셀로판지에 개별 그림을 그리고 겹쳐서 훌륭한 하나의 그림을 만들 듯 데이터 시각화를 여러 개의 그림으로 나누어 겹쳐서 표현하는 방법이다. 계층적 그리기 방법은 다양한 그림그리기 앱에서 기본 방식으로 채용될 정도로 유용한 방법이다. 특히 그림의 재활용과 오류 수정 측면에서 강점이 있는 방법이다.

GoG는 데이터 시각화를 위한 규칙을 설명하면서 구성요소와 절차를 섞어서 사용하는 것처럼 보인다. 그 이유는 GoG가 '데이터를

컴퓨터 화면에 표현한 그래프'라는 결과물을 염두에 두고 만들어낸 규칙이기 때문이다. 뭔가를 만들기 위해 구성 요소를 구분하다 보면 아무래도 절차와 요소가 비슷해지는 것은 피하기 어렵다. 이런 특징은 Wickham이 ggplot2에 적용하면서 강화되어 GoG에서 구성요소와 절차는 동전의 양면처럼 구분이 어렵다.

Sakar(2018)는 Wilkinson과 Wickham의 구성요소/절차를 정리하며 좌표축에 변수를 대응시키는 과정을 'Aesthetics Mapping'이라는 용어로 더 구체적으로 표현하고, 선, 점, 막대, 레전드와 같은 그래프 구성요소의 위치를 정하는 절차를 위치설정(Position Adjustment)이라고 따로 분류해서 강조했다.

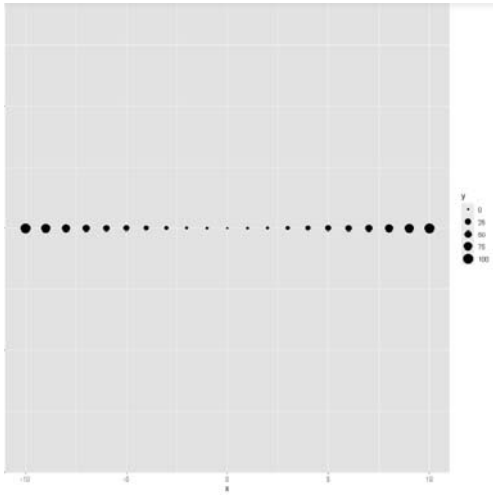
지금까지 살펴본 바와 같이 GoG는 다양한 요소와 절차로 구성되어 있고, 그 내용이 꼭 일치하지 않는다. 그럼에도 데이터 시각화의 결과물인 그래프에는 공통으로 들어가야 할 내용이 있다. 이를 살펴보기 위해서 간단한 함수인 $f(x) = x^2$을 시각화하는 경우를 생각해 보자.



<그림 1> $f(x) = x^2$ 그래프

<그림 1>과 같은 포물선 모양인 $f(x) = x^2$의 그래프는 너무 익숙하므로 바로 종이에 그릴 수 있겠지만 컴퓨터 화면에 이를 나타낸다고 생각하고 그 절차를 하나씩 살펴보자. 우선 이 그래프를 그리기 위해서 x축의 범위를 정해줘야 한다. 종이에 이 함수를 그릴 때는 x축, y축을 화살표로 표시하고 각 축의 범위를 고민할 필요는 없지만, 컴퓨터는 이런 방식의 추상적 연산을 할 수 없으므로 어디에 어떤 점을 찍어야 할지를 명확히 정의해 주어야 한다. x의 범위를 -10에서 10까지 1단위로 증가하는 수열로 정한다고 가정하자. 다음 절차는 이제 x값에 대응되는 y값을 구해야 한다. x값을 제곱해서 이에 대응되는 y값을 구한 후에는 이를 좌표축에 점으로 찍으면 <그림 1>이 완성된다. 이 과정은 그래프 그리기를 처음 배우는 학생들이 반복하는 과정이기도 하다.

<그림 1>을 그리는 과정을 다시 정리하면 다음과 같다. x값의 범위를 정하고 이에 따라 y값을 구해서 둘의 대응 관계를 나타내는 표를 만들고, 이 표에 따라서 xy 좌표평면에 대응되는 지점을 점으로 표시하는 두 가지 절차로 구성되어 있다. 모든 그래프는 이런 절차로 그린다. 데이터 시각화를 위한 그래



<그림 2> $f(x) = x^2$ 그래프
(y축: 점의 크기)

프도 이 절차를 따르며, 두 가지 절차 중 하나라도 빠지면 그래프를 그릴 수 없다. 이런 맥락에서 데이터를 만들고, 이를 좌표축에 기하학적 모양으로 대응시키는 절차는 모든 그래프에 필수적인 절차이다.

데이터를 만드는 작업은 x값이라는 최초의 데이터에서 출발하여 이를 제곱하여 y값이라는 새로운 데이터를 만들고 이들을 1:1로 대응시킨 표(Table)를 만드는 과정이다. 이 과정에서 데이터의 변환이 필요하고 그 결과로 새로운 데이터가 만들어 진다. 한편, 데이터를 점으로 좌표축에 그리는 과정은 우선 축을 설정하는 것에서 시작한다. 우리는 x를 가로축에 y를 세로축에 대응시키는 것이 익숙하지만 데이터 시각화 과정에서 항상 그렇지는 않다. 가로축과 세로축이 바뀌기도 하고 세로축 대신에 <그림 2>와 같이 점의 크기를 축으로 사용할 수도 있다. 약간의 상상을 더한다면 <그림 2>는 $f(x) = x^2$의 그래프를 3차원 공간에 그리고 위에서 내려다본 모습이다. 이렇듯 그래프를 그리는 과정에서 너무나 익숙해서 생략하는 경우가 많지만 축설정은 그래프의 모양을 통째로 바꾸는 매우 중요한 절차이다. 이후에 기하학적 모양을 정해서 이를 앞 단계에서 설정된 축에 그려 넣으면 그래프가 완성된다.

지금까지 설명한 바를 토대로 데이터 시각화를 위한 공통규칙은 다음과 같이 구성할 수 있다. 데이터의 변환(Data Transformation)을 통해 그래프를 그리기 위한 데이터(Data)를 만드는 '자료변환'과 축을 설정하고 설정된 축에 맞추어 기하학적 모양을 그려 넣는 '축설정과 기하학적 모양 선택/그리기'가 데이터 시각화의 핵심 구성요소/절차이다. 좀 더 세분화하면 (1) 자료변환(Data Transformation), (2) 데이터(Data), (3) 축설정(Aesthetics), (4) 기하학적 모양 선택/그리기(Geometry)의 네 가지 구성요소/절차가 데이터 시각화를 위한 필수 요소이다. 이 구성요소/절차들은 GoG를 설명한 대부분의 연구에서 공통적으로 나타난다. 이 구성요소는 GoG 기본적인 필수요소/절차이므로 여기에서는 GoG Basics라고 명명하겠다.

GoG는 데이터를 컴퓨터를 활용해서 시각화하는 규칙이므로 컴퓨터에서 구현하는 과정을 이해하는 것이 필요하다. 여기에서는 R의 ggplot2를 통해 GoG를 구현한다. 따라서 앞으로 논의를 위해 <그림 1, 2>를 R의 ggplot2에서 그리는 과정을 살펴보자. 여기에서는 R을 활용하지만 GoG는 Stata, Julia 등 다양한 컴퓨터 언어에서 구현되어 있는 보편성이 있는 개념이다.

GoG를 R에서 구현한 ggplot2를 활용하기 위해서는 우선 ggplot2 패키지를 R에 설치해야 한다. ggplot2 패키지를 직접 설치해도 되지만, 데이터 변환에 필요한 다른 패키지도 포함하고 있는 통합 패키지인 "tidyverse" 패키지를 다음과 같이 설치하는 것이 좋다.

```
>install.packages("tidyverse")
>library(tidyverse)
```

'>' 이후에 나타나는 것은 R의 명령어이므로 이를 그대로 R 콘솔에서 실행하면 결과를 얻을 수 있다. tidyverse 패키지를 설치한 후 x라는 변수에 데이터를 -10에서 10까지 저장하고, y에는 x를 제곱해서 다음과 같이 저장한다.

```
> x <- -10:10 # -10, 10까지 1씩 증가하는 수열을 만들어서 x에 저장 (1)
> y <- x^2 # x^2을 y에 저장
```

ggplot2에서는 데이터프레임을 그래프를 그리기 위한 기본 데이터로 사용하므로 x, y

변수를 데이터프레임 테이블(표)로 다음과 같이 변환한다.

```
> df <- data.frame(x,y) #x,y 테이블을 데이터 프레임으로 df에 저장 (2)
```

'ggplot' 명령어를 사용해서 데이터의 축을 다음과 같이 설정해서 p1에 저장한다. 축은 Aesthetics의 준말인 'aes' 옵션을 활용해서 설정한다.

```
> p1 <- ggplot(df,aes(x=x,y=y)) # df에 저장된 Data를 x축에 x, y축에 y로 축설정해서 p1에 점모양으로 매핑저장 (3)
```

'geom_point' 명령어를 사용해서 p1 위에 기하학적 모양을 선택해서 좌표의 위치에 다음과 같이 겹쳐 그린다. p1을 기반으로 기하학적 모양을 겹쳐 그린다는 것은 '+' 기호로 표시한다.

```
> p1 + geom_point() # <그림 1> (4)
```

<그림 2>는 축설정시 y값은 0으로 고정시키고, 'size'이라는 축에 y를 매핑해서 다음과 같이 그린다.

```
>ggplot(data.frame(x,y),aes(x=x,y=0,size=y)) + geom_point() # <그림 2>
```

3. 자료변환과 파이프 연산

GoG Basics는 데이터 변환(Data Transformation, 이하 DT)과 데이터(Data), 축설정(Aesthetics)과 기하학적 모양 선택/그리기(Geometry)로 구성되어 있다. R의 ggplot2 패키지는 GoG Basics와 1:1로 대응시킬 수 있다. 2장의 식 (1)-(4)에 제시된 R 명령어는 각각은 데이터 변환(DA), 데이터(Data), 축설정(Aesthetics), 기하학적 모양 선택/그리기(Geometry) 절차와 대응된다.

R의 명령어들이 GoG를 그대로 설명할 수 있고, 실무에서 데이터 시각화를 위해서는 패키지를 활용하는 것이 필요하므로, 설명의 편의와 그 과정에서 자연스럽게 GoG를 통한 데이터 시각화 구현 능력 습득이라는 두 마리 토끼를 한 번에 잡을 수 있도록 이제부터는 R의 명령어를 중심으로 GoG Basics의 상세 내용을 살펴본다.

2장에서 살펴보았듯이, GoG를 활용한 데이터 시각화는 데이터에서 출발한다. 그런데 이 데이터는 형태나 내용을 변환해야 원하는 형식이 되는 경우가 대부분이다. 따라서 3장에서는 데이터 시각화를 위해 필요한 데이터를 만드는 과정을 자세히 살펴본다.

우선 데이터 시각화를 위해서 ggplot2에서 필요한 데이터 형식은 데이터프레임(Dataframe)이다. 데이터프레임은 가장 일반적인 통계자료의 형태로 행(Row)은 관측값, 열(Column)은 속성(Attribute)으로 구성되어 있는 자료의 형태이다. 예를 들어 한 반의 키, 몸무게, 시력, 성별을 조사해서 테이블로 정리한다면 이름, 키, 몸무게, 시력, 성별이 열을 구성하고, 각 행은 구성원 하나하나의 정보가 들어간다. 이런 형태의 데이터를 데이터프레임이라고 한다. R에서는 데이터프레임이라는 변수형(Variable Type)이 있으며, 이 변수형을 가진 데이터만 ggplot을 통해 시각화가 가능하다. 따라서 다른 형태의 데이터를 데이터프레임으로 바꿔야 하는 경우가 그래프를 그리는 과정에서 많이 발생하며 이 명령어는 'data.frame'이다.

다음은 Name(이름), Height(키), Weight(몸무게), Vision(시력), Gender(성별)라는 변수명으로 데이터프레임 자료를 만드는 과정이다. <그림 3>은 실행 결과이다.

Name	Height	Weight	Vision	Gender
<chr>	<dbl>	<dbl>	<dbl>	<chr>
홍길동	180	70	1.5	남
신사임당	170	50	1.2	여
전우치	190	75	0.9	남
선덕여왕	160	45	1.8	여

<그림 3> 데이터프레임(df1)

```
> df1 <- data.frame(Name=c("홍길동", "신사임당","전우치","선덕여왕"), Height = c(180, 170, 190, 160), Weight=c(70, 50, 75,45), Vision=c(1.5,1.2,0.9,1.8), Gender=c("남","여","남","여")) # <그림 3>
```

<그림 3>과 같은 데이터프레임 데이터를 넓은 데이터 형태(Wide Data Format)라고 한다. 이와 달리 동일한 데이터를 <그림 4>와 같이 표현할 수 있다. 이 데이터 형식을 긴 데이터 형태(Long Data Format)라고 한다. <그림 3>을 <그림 4>와 같이 바꾸는 것은 다음을 실행하면 된다.

```
> df1_long <- gather(df1, Features, Value, c(Height, Weight,Vision, Gender)) # <그림4>
```

<그림 4>의 데이터프레임을 <그림 3>의 넓은 형식으로 바꾸기 위해서는 다음을 실행하면 된다.

```
> spread(df1_long,Features,Value)
```

데이터는 처음부터 우리가 원하는 형태로 제공되지 않는다. 그래프를 그리기에 적절한 데이터를 만들기 위해서는 변환이 필수적이다. 때로는 변환을 여러 번 수행해야 하는 경우도 있다. 이런 경우에 앞에서 연산한 결과를 계속 사용해야 한다. 이런 과정은 물을

Name	Features	Value
<chr>	<chr>	<chr>
홍길동	Height	180
신사임당	Height	170
전우치	Height	190
선덕여왕	Height	160
홍길동	Weight	70
신사임당	Weight	50
전우치	Weight	75
선덕여왕	Weight	45
홍길동	Vision	1.5
신사임당	Vision	1.2
전우치	Vision	0.9
선덕여왕	Vision	1.8
홍길동	Gender	남
신사임당	Gender	여
전우치	Gender	남
선덕여왕	Gender	여

<그림 4> 긴 데이터 형식

원하는 압력으로 가정에 공급하기 위해서 큰 수도관에서 점차 작고 가는 수도관으로 바꾸면서 압력을 낮추는 과정과 유사하다. 데이터 변환을 차례로 연속적으로 수행하는 방법을 수도(?) 파이프(Pipe) 연산이라고 한다.

R에서 파이프 연산은 '%>%'를 통해 수행한다. 파이프 연산은 데이터를 추출하고(filter), 변형하고(mutate), 그룹핑하고(group_by), 변수를 선택해서(select), 요약(summarize)하는 과정으로 이루어진다. 괄호안의 영문은 R 파이프 연산의 명령어이다. 예를 들어 <그림 3>의 데이터(df1)에서 키가 170 이상인 관측치를 뽑아내는 명령은 다음과 같다.

> df1 %>% filter(Height >= 170)

이제 키가 170이상인 사람들을 뽑아서 이들의 BMI를 계산해서 BMI라는 이름으로 추가해 보자. BMI는 몸무게 나누기 키의 제곱이므로 다음과 같이 데이터를 변환하여 df2에 저장할 수 있다.

> df2 <- df1 %>% filter(Height >= 170) %>% mutate(BMI = Weight/(Height)^2)

Name	Height	Weight	Vision	Gender	BMI
<chr>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>
홍길동	180	70	1.5	남	0.002160494
신사임당	170	50	1.2	여	0.001730104
전우치	190	75	0.9	남	0.002077562

<그림 5> df2 데이터

BMI는 10-40 사이의 값을 갖는데, <그림 5>의 BMI는 뭔가가 잘못되었다. 그 이유는 BMI 계산에는 키를 센티미터가 아니라 미터 단위로 계산해야 하기 때문이다. 이를 반영하여 다시 BMI를 계산하기 위해서 키 170 이상인 사람을 뽑고, 키를 100으로 나누어 미터 단위로 고치고, BMI를 계산해서 <그림 6>과 같은 결과를 얻는 방법은 다음과 같다.

> df2 <- df1 %>% filter(Height >= 170) %>% mutate(Height = Height/100) %>% mutate(BMI = Weight/(Height)^2)

Name	Height	Weight	Vision	Gender	BMI
<chr>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>
홍길동	1.8	70	1.5	남	21.60494
신사임당	1.7	50	1.2	여	17.30104
전우치	1.9	75	0.9	남	20.77562

<그림 6> df2 (BMI를 제대로 계산)

<그림 6>에서 키가 미터단위로 바뀐 것을 확인할 수 있다. 키에 따라 데이터를 뽑고, 단위를 바꾸고, BMI를 계산해서 새로운 열에 추가하는 일련의 과정이 물이 흐르듯이 순서대로 진행되므로 '%>%' 연산을 파이프 연산이라고 한다. 파이프 연산은 앞의 연산 결과를 새로운 입력으로 받아서 연산을 하므로 연산과정에서 매번 입력을 새롭게 정의해야 하는 번거로움이 없고, 차례대로 연산하는 과정을 볼 수 있어 연산 과정을 이해하기 쉬우므로 데이터 변환시 효율적인 방법이다.

<그림 3>의 df1을 남, 여 구분하여 평균 키를 구하는 방법은 다음과 같으며 그 결과는 <그림 7>에서 확인할 수 있다.

> df1 %>% group_by(Gender) %>% summarize(Mean_Height = mean(Height))

Gender	Mean_Height
<chr>	<dbl>
남	185
여	165

<그림 7> 그룹별 평균을 구한 결과

한편 <그림 4>의 데이터에서 Features와 Value 변수만 뽑아내는 방법은 다음과 같다.

> df1_long %>% select(Features, Value)

지금까지 살펴 본 과정이 데이터를 변환해서 그래프를 그리기에 적합한 데이터(Data)를 준비하는 절차이다. 데이터 변환은 다양한 방법과 아이디어가 존재한다. 여기에서 제시된 'filter', 'mutate', 'group_by', 'select', 'summarize'는 많이 쓰이는 기본 명령어이며, 이외에도 R에서는 다양한 데이터 변환 방법과 명령어가 존재한다. 이와 관련된 자세한

방법론은 매드코프(2014)와 Wickham(2017)을 참고하기 바란다.

4. 축설정과 모양 선택/그리기

데이터가 준비되었으면 이제 그래프를 그리기만 하면 된다. 그래프를 그리는 과정은 앞에서 설명한 바와 같이 축설정과 기하학적 모양 선택/그리기로 나눌 수 있다.

축설정의 영어표현인 Aesthetics는 그리스어에서 파생된 말로 원래 의미는 인식(Perception)이다(Wilkinson, 2012). 축을 설정하는 것은 데이터를 어떻게 나타내고 인식할 것인지를 결정하는 가장 중요한 절차이다. x축, y축만 바꾸어도 완전히 다른 그래프가 되기도 하고, 색깔이나 점의 크기, 선의 굵기 등에 따라서 그래프가 전달하는 정보가 완전히 다르게 인식될 수 있다는 면에서 축설정을 Aesthetics라고 표현한 것은 의미가 있다. 또한 축설정에 따라서 그래프의 미학적인 가치도 달라질 수 있으므로 Aesthetics라는 표현은 축설정 절차에 딱 맞는 표현이다.

Wilkinson(2012)은 축을 형태, 표면, 움직임, 음성, 텍스트의 5가지 범주, 16개 종류로 구분해서 제시했다. 색상이나 움직임 형태의 다양성을 고려할 때 xy축에 원하는 수준의 고차원 데이터를 얼마든지 표현할 수 있다. 이런 이유로 GoG로 구현된 그래픽 패키지는 원칙적으로 3차원 그래프가 없다.

ggplot2에서 축설정 하는 명령은 'ggplot'이며, 축은 'aes' 옵션을 사용해서 설정한다. 2장에서 그린 이차함수 그래프의 축설정 방식이 시각화를 위한 데이터에도 그대로 적용된다.

축을 설정한 후에는 기하학적 모양을 선택하고 이를 축에 그리는 절차(Geometry)가 필요하다. 이 절차는 GoG Basics의 마지막 단계이다. 기하학적 모양 선택/그리기는 가장 간단한 점찍기(geom_point)에서 선그리기(geom_line), 히스토그램(geom_histogram), 텍스트 표시(geom_text) 등 다양하게 존재한다. 데이터 시각화에 자주 활용되는 다양한 그래프는 Prabhakran(2015)과 RStudio(2020)를 참고하기 바라며, 여기에서는 그래프를 그리는 사례를 통해 ggplot에서 축설정과 기

manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
<chr>	<chr>	<dbl>	<int>	<int>	<chr>	<chr>	<int>	<int>	<chr>	<chr>
audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact
audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact
audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact
audi	a4	2.8	1999	6	auto(l5)	f	16	26	p	compact
audi	a4	2.8	1999	6	manual(m5)	f	18	26	p	compact
volkswagen	passat	1.8	1999	4	auto(l5)	f	18	29	p	midsize
volkswagen	passat	2.0	2008	4	auto(s6)	f	19	28	p	midsize
volkswagen	passat	2.0	2008	4	manual(m6)	f	21	29	p	midsize
volkswagen	passat	2.8	1999	6	auto(l5)	f	16	26	p	midsize
volkswagen	passat	2.8	1999	6	manual(m5)	f	18	26	p	midsize

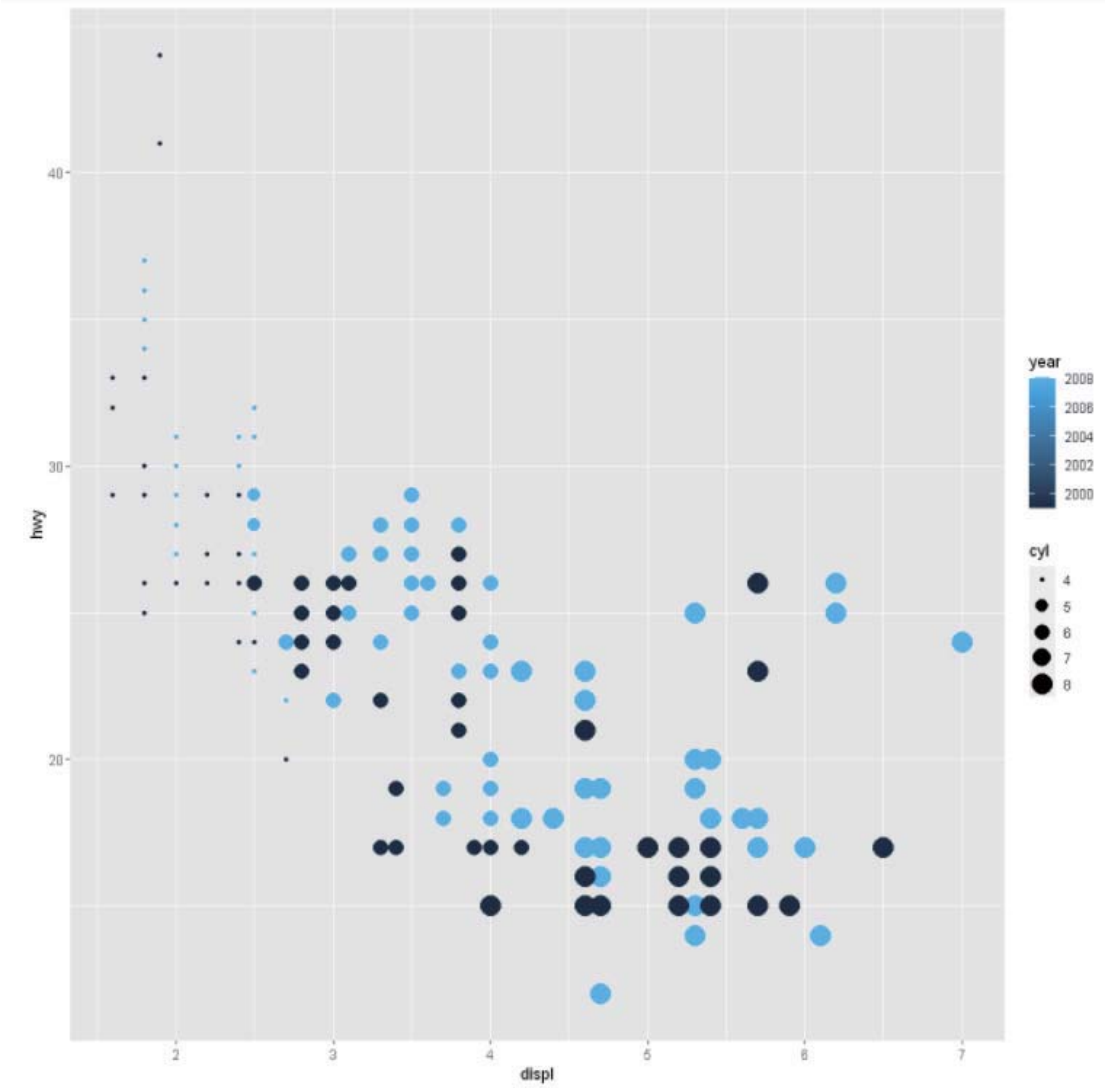
<그림 8> mpg dataset(R)

하학적 모양 선택/그리기 절차를 수행하는 방법을 살펴보자.

데이터 시각화는 변수의 관계를 파악하거나 변수의 분포를 파악하기 위한 것이 대부분이다. 이런 맥락에서 변수의 관계 파악을 위한 산점도(geom_point)와 분포를 개략적으로 파악하기 위한 밀도함수(geom_density)를 중심으로 데이터 시각화 사례를 살펴본다.

<그림 8>은 R에 있는 mpg 데이터이다. 이 데이터는 15개 자동차 제조사(manufacturer)의 38개 모델(model)에 대해서 제조연도(year), 배기량(displ), 고속도로연비(hwy), 실린더수(cyl), 구동방식(drv) 등을 조사한 234의 관측치(행이 234개)로 구성된 자료이다.

먼저 배기량과 고속도로 연비가 실린더수, 제조연도에 따라서 어떻게 변하는지를 나타내는 그래프를 그려보자. 데이터는 이미 준비되어 있으니, 축설정과 모양 선택/그리기만 하면된다. 축은 x축에 배기량(displ), y축



<그림 9> 배기량 vs 연비 (연도별, 실린더 개수별)

에 연비(hwy)로 표기하고 제조연도(year)는 색으로 구분한다. 마지막으로 실린더의 개수는 크기로 구분한다. 관계를 살펴볼 때 유용한 그래프가 xy축에 점을 찍는 것이므로 기하학적 모양은 점(geom_point)를 선택해서 그래프를 그린다. 이는 다음과 같은 명령으로 실행할 수 있으며, 그 결과는 <그림 9>에서 확인할 수 있다.

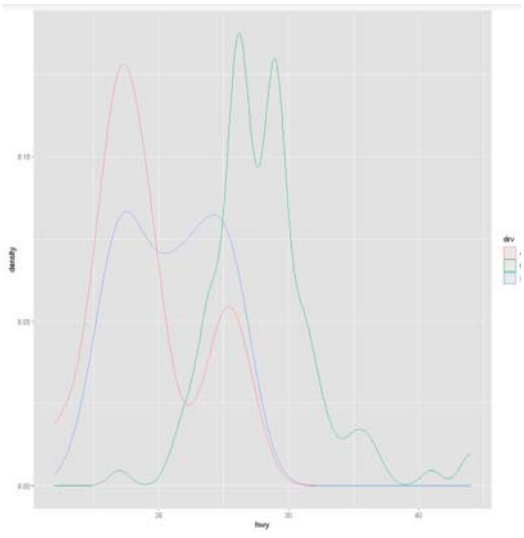
```
>ggplot(mpg, aes(x=displ,y=hwy,col=year,size=cyl))+
geom_point() # <그림 9>
```

<그림 9>는 간단한 명령으로 그렸지만 밝은 색으로 표시된 2008년 제조 자동차가 어두운 색으로 표시된 1998년 제조 자동차에 비해 전반적으로 연비가 향상되었음을 쉽게 확인할 수 있다. 또한 실린더 수와 배기량은 비례하고 배기량/실린더수와 연비는 반비례하는 경향이 있음을 쉽게 파악할 수 있다.

이제 변수의 분포를 파악할 수 있는 사례를 살펴보자. mpg 데이터에서 구동방식(f:전륜, r:후륜, 4:4륜)에 따른 연비의 분포를 파악하고자 한다고 가정하자. 연비의 분포 그래프는 x축에 연비 y축에 해당 연비가 출현하는 밀도(빈도)가 표현되어야 한다. y는 그리는 과정에서 자동으로 계산되므로 축설정은 x축과 구동방식 구분을 위한 색깔축만 설정 하면 된다. 이는 다음과 같은 명령으로 실행되며 출력 결과는 <그림 10>과 같다.

```
> ggplot(mpg, aes(x=hwy,col=drv)) +
  geom_density() #<그림 10>
```

<그림 10>에서 연비는 4륜<후륜<전륜 순이며 분포가 쌍봉을 이루고 있음을 쉽게 확인할 수 있다. 이 그래프만으로도 구동방식과 연비의 관계, 추가 분석 방향에 대한 시사점을 얻을 수 있다.



<그림 10> 구동방식별 연비 분포

<그림 10>을 그릴 때 y축은 자동으로 계산되었다. 정확하게는 빈도를 구하는 통계처리를 자동으로 한 것이다. 이 사례는 그래프 그리기 절차가 통계처리를 포함하고 있다는 것을 보여준다. 이런 이유로 기하학적 모양 선택/그리기 절차는 'geom_*' 대신에 'stat_*' 명령어를 통해서도 수행할 수 있다. 데이터 시각화의 기하학적 모양 선택과 통계처리는 불가분의 관계이므로 그리기에 초점을 맞춘 'geom_*' 명령어와 통계처리에 초점을 맞춘 'stat_*'이 따로 정의되어 있다.. 'stat_*' 계열의 함수와 명령어는 rstudio(2020)에 상세하게 제시되어 있으므로 참고하기 바란다.

지금까지 살펴본 사례에서 축설정과 기하학적 모양 선택/그리기 과정에서 간단한 명령으로도 다차원의 복잡한 시각화를 수행할 수 있다는 점을 확인할 수 있다. 이는 GoG와 ggplot의 장점이다. 이외에도 GoG와 ggplot은 그래프를 그려놓고 의미를 살피는 것이 아니라 축설정 단계에서 그래프의 형태를 미리 짐작하게 되므로 체계적인 방식으로 데이터를 시각화 할 수 있다는 장점이 있다. 또한 계층적 접근법을 활용함으로써 필요한

그래프를 조합해서 사용할 수 있어 자료 특성 파악을 위한 탐색적 연구에 유용한다.

GoG에는 GoG Basics 외에도 아름다운 그래프를 만들기 위해서 스케일, 타이틀 등의 다양한 옵션이 있다. 이에 대한 자세한 논의는 Wickham(2017)을 참고하기 바라며, 여기에서는 x축 이름, y축 이름, 그래프 전체의 이름을 붙이는 방식만 소개한다. <그림 9>에 x축 이름을 '배기량', y축 이름은 '연비', 그래프 이름은 '배기량/연비 비교'라고 설정하는 명령은 다음과 같다.

```
>ggplot(mpg, aes(x=displ,y=hwy,col=year,size=cyl))+
geom_point() + labs(x="배기량",y="연비",title="배기량/연비 비교")
```

5. 결론

인류가 가진 감각 중 가장 발달한 감각은 시각이다. 이런 차원에서 그래프는 데이터의 의미를 파악하고 전달하는 가장 효과적인 수단이다. GoG는 데이터 시각화를 위해 필요한 요소이자 절차를 표준화한 것이다. ggplot2는 R을 기반으로 GoG를 구현한 패키지로 계층적 접근법을 통해 좋은 그래프를 그릴 수 있는 기반을 제공한다.

GoG는 R뿐 아니라 Python, Julia, Stata 등의 다양한 컴퓨터 언어에 구현되어 있어 좋은 그래프를 다양한 환경에서 그릴 수 있다. 그러나 범용성으로 인해 GoG는 개념상 혼동의 여지가 있어 실무자들이 이를 이해하고 활용하기에는 어려움이 있었다. 여기에서는 이 점을 개선해 필수 구성요소를 중심으로 간소화된 형태의 GoG Basics라는 개념으로 GoG를 소개하고 이를 ggplot2를 통해 활용하는 사례를 제시함으로써 현장 활용 가능성을 높이기 위한 기반을 제공했다.

참 고 문 헌

[1] 매트로프, 노만 (권정민 역), 빅데이터 분석 도구 R 프로그래밍, 에이콘, 2014.

[2] 박동련, R에의한 통계 그래픽스, 자유아카데미, 2011.

[3] Medina, J (서영조 옮김), 브레인룰즈 (Brain Rules), 프런티어. 2009.

[4] Cleveland, William S., Visualizing Data, Hobart Press, 1993.

[5] Prabhakaran, Salva, "Top 50 ggplot2 Visualizations - The Master List (With Full R Code),", rstatistics.co, 2015. accessed on 2021-04-21. available: http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html#Dot%20Plot.

[6] RStudio, "Data Visualization Cheatsheet," RStudio, 2020. accessed on 2021-04-21. avaiable: https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf.

[7] Sarkar, DJ, "A Comprehensive Guide to the Grammar of Graphics for Effective Visualization of Multi-dimensional Data," Towards Data Science, Sep.18, 2018. accessed on 2020-03-21. avaiable: https://towardsdatascience.com/a-comprehensive-guide-to-the-grammar-of-graphics-for-effective-visualization-of-multi-dimensional-1f92b4ed4149.

[8] Wickham, Hadley and Garrett Grolemund, R for Data Science, O'REILLY, 2017.

[9] Wickham, Hadley. "A Layered Grammar of Graphics." Journal of Computational and Graphical Statistics 19(1): 3-28, 2010.

[10] Wilkinson, Leland, The Grammar of Graphics, Springer, 1999.

[11] Wilkinson, Leland, "The Grammer of Graphics," in Handbook of Computational Statistics, Springer, 2012, pp.375-414.

저 자 소 개

윤봉규(E-mail: bkyoon1@gmail.com)
1996 연세대학교 경영학사
1998 한국과학기술원 산업공학 석사
2002 한국과학기술원 산업공학 박사
현재 국방대학교 운영분석전공 교수
관심분야 : Agent Based Modeling, Stochastic Models in Military O.R, Biz. Performance Optimization & Innovation.