

Chapter 11

분석실습: 지하철 이용분석

오 세 종

목표

- 서울도시철도공사에서 제공받은 2010~2013년 지하철 역별 승하차 정보 데이터를 바탕으로 탑승객 수를 역, 노선, 연도, 월별로 자료를 정리하는 기법을 습득한다
- 탑승객 기준 상위 10개 역을 추출하여 이를 시각화 하는 방법을 학습한다.
- 구글지도를 활용하여 추출된 자료를 시각화하는 기법을 학습한다

본 단원은 K-ICT 빅데이터 센터(<https://kbig.kr/portal/>)의 교육실습 콘텐츠를 기본으로 일부 내용을 추가하여 작성하였음

1. 데이터셋 소개

- subway.csv

- 2010년 1월부터 2014년 7월까지 서울지하철역 및 시간대별 승하차 인원수 정보를 제공

변 수 명	설 명
station	역 코 드
stat_name	역 명
income_date	일 자
on_tot	당 일 해 당 역 의 총 탑 승 인 원 수
on_xx	당 일 해 당 역 의 xx시 간 대 의 탑 승 인 원 수 (xx는 05부 터 24까 지)
off_tot	당 일 해 당 역 의 총 하 차 인 원 수
off_xx	당 일 해 당 역 의 xx시 간 대 의 하 차 인 원 수 (xx는 05부 터 24까 지)

1. 데이터셋 소개

- subway.csv

station	stat_name	income_da	on_tot	on_05	on_06	on_07	on_08	on_09	off_tot	off_05	off_06	off_07	off_08	off_09	off_10	off_11
2511	방화	20100101	3084	74	126	121	251	188	2933	1	154	89	68	98	79	118
2511	방화	20100102	4676	80	128	210	366	345	4334	2	62	58	148	133	130	147
2511	방화	20100103	3942	60	125	131	283	264	3785	0	80	55	76	133	153	118
2511	방화	20100104	10641	168	654	2006	1566	774	10299	4	117	169	502	489	459	316
2511	방화	20100105	10129	192	718	1954	1542	742	9224	1	128	180	566	479	275	256
2511	방화	20100106	8914	155	552	1762	1379	645	8453	1	115	161	535	380	225	210
2511	방화	20100107	8448	139	527	1643	1306	603	7721	0	106	134	484	363	201	175
2511	방화	20100108	8544	139	484	1553	1259	591	7741	0	108	136	489	365	196	202
2511	방화	20100109	6177	94	198	382	551	504	5800	2	101	100	208	144	171	161
2511	방화	20100110	4570	76	142	203	340	342	4373	0	84	61	100	125	189	143
2511	방화	20100111	8442	173	555	1646	1304	612	7786	2	118	125	517	347	203	228
2511	방화	20100112	8137	134	485	1603	1273	511	7537	2	88	131	475	307	209	220
2511	방화	20100113	7804	129	475	1523	1292	572	7180	1	113	111	453	291	179	199
2511	방화	20100114	7819	128	462	1490	1249	538	7305	1	110	127	475	282	180	207
2511	방화	20100115	8658	134	448	1542	1302	612	7685	0	108	126	501	351	208	216
2511	방화	20100116	6022	92	217	371	560	457	5544	1	96	82	205	156	187	244
2511	방화	20100117	4275	67	164	163	317	339	4301	3	99	58	114	155	183	126
2511	방화	20100118	8209	166	512	1629	1186	592	7491	1	102	116	489	285	204	229
2511	방화	20100119	8308	144	502	1616	1223	567	7725	0	109	119	443	315	216	224
2511	방화	20100120	7945	127	476	1517	1232	580	7328	1	97	115	427	294	205	208
2511	방화	20100121	7976	152	462	1522	1228	583	7413	2	113	117	443	309	180	209
2511	방화	20100122	8065	143	437	1480	1226	552	7207	1	105	99	481	318	199	206
2511	방화	20100123	6053	129	196	364	522	461	5657	0	106	94	201	165	163	219

2010-01-23

1. 데이터셋 소개

- subway_latlong.csv

- [서울 열린데이터 광장]에서 제공하는 지하철 노선별 역이름 및 위치 정보(위도, 경도) 자료 및 각 역의 노선명을 제공

변 수 명	설 명
STATION_CD	역 코드
STATION_NM	역 명
LINE_NUM	호 선
FR_CODE	외 부 코 드 (외 국 인 의 경 우 , 역 명 보 다 역 번 호 로 문 의 하 는 경 우 가 많 음)
CYBER_ST_CODE	사 이 버 스 테 이 션 (환 승 역 의 경 우 마 스 터 가 되 는 노 선 의 전 철 역 코 드)
XPOINT	X좌 표
YPOINT	Y좌 표
XPOINT_WGS	X좌 표 (WGS)
YPOINT_WGS	Y좌 표 (WGS)

1. 데이터셋 소개

- subway_latlong.csv

STATION_CD	STATION_NM	LINE_ID	IFR_CODE	CYBER_ST_CODE	XPOINT	YPOINT	XPOINT_WGS	YPOINT_WGS
330	교대	3	340	330	502900	1108655	37.493415	127.0141
331	남부터미널	3	341	331	503497	1106375	37.485013	127.0162
332	양재	3	342	332	507387	1106032	37.484147	127.0346
333	매봉	3	343	333	510257	1106985	37.486947	127.0468
334	도곡	3	344	334	512105	1107980	37.490858	127.0554
335	대치	3	345	335	513907	1108850	37.494612	127.0636
336	학여울	3	346	336	515537	1109537	37.496663	127.0706
337	대청	3	347	337	517440	1108825	37.493514	127.0795
338	일원	3	348	338	518585	1106192	37.483681	127.0844
339	수서	3	349	339	522445	1106977	37.487371	127.1019
341	경찰병원	3	351	341	527245	1109277	37.495918	127.1245
409	당고개	4	409	409	517267	1157807	37.670272	127.0791
410	상계	4	410	410	515995	1155000	37.660878	127.0736

2. 데이터셋 읽어오기 및 전처리

- 실습에 필요한 패키지
 - ggplot2 : 시각화 기법 제공
 - ggmap : 구글지도 시각화

```
library(ggplot2)
library(ggmap)
```

2. 데이터셋 읽어오기 및 전처리

```
setwd("c:/Rworks")           # 데이터셋 경로지정
subway <- read.csv("subway.csv", header=TRUE,
  stringsAsFactors=FALSE)
head(subway)
str(subway)

# incomedate 변수를 표준 날짜형식으로 전환
class(subway[, "income_date"]) <- "character"
subway[, "income_date"] <- as.Date(subway[,
  "income_date"], format="%Y%m%d")
unique(format(subway[, "income_date"], "%Y"))
```



2. 데이터셋 읽어오기 및 전처리

- 2014년 자료는 7월까지의 정보만 포함하고 있으므로 2014년 자료를 분석에서 제외하고, 나머지 연도의 자료를 `subway2` 이름의 데이터프레임으로 저장한다

```
idx <- format(subway[, "income_date"], "%Y") ==  
"2014"  
unique(format(subway[idx, "income_date"], "%m"))  
subway2 <- subset(subway, subset =  
  format(income_date, "%Y") != "2014")
```

```
> unique(format(subway[idx, "income_date"], "%m"))  
[1] "01" "02" "03" "04" "05" "06" "07"
```

`format(, "%Y")` 는 날짜표준서식에서 연도를 추출

`format(, "%m")` 는 날짜표준서식에서 월(01, 02, 등)을 추출



2. 데이터셋 읽어오기 및 전처리

- 역명이 어떻게 되어있는지 살펴보자

```
sort(unique(subway2[ , 'stat_name']))
```

```
> sort(unique(subway[ , 'stat_name']))
[1] "가락시장(8)"      "가산디지털단지(7)"  "강남구청"
[4] "강동"            "강동구청"          "개롱"
[7] "개화산"          "거여"              "건대입구(7)"
[10] "고덕"            "고려대"            "고속터미널(7)"
[13] "공덕(5)"         "공덕(6)"           "공릉"
[16] "광나루"          "광명사거리"        "광화문"
[19] "광흥창"          "구산"              "군자(5)"
[22] "군자(7)"          "굴포천"            "굽은다리"
[25] "길동"            "김포공항(5)"       "까치산(5)"
... ..
```

환승역의 경우 노선번호가 붙어 있음

공덕(5), 공덕(6) 은 분석에서는 동일한 역이므로 통일 필요

2. 데이터셋 읽어오기 및 전처리

- 역명에서 (노선) 을 삭제하자

```
# 역명에 ( 가 포함된 행을 검색한다
idx <- grep( "\\(" , subway2$stat_name )
unique(subway2$stat_name[idx])

# ()를 제거한다. 역명에서 뒤의 3글자 제거
subway2$stat_name[idx] =
  substr(subway2$stat_name[idx],
    1, nchar(subway2$stat_name[idx])-3)
```

`substr(x,1,5)` : 문자열 `x` 에서 1~5자리의 문자를 자른다



2. 데이터셋 읽어오기 및 전처리

- 연도별, 월별 집계를 용이하도록 하기 위해, 연도, 월 컬럼을 추가한다

```
year <- format(subway2$income_date, "%Y")
month <- format(subway2$income_date, "%m")
subway2 <- cbind(subway2, year, month)
head(subway2)
```

	on_23	on_24	off_tot	off_05	off_06	off_07	off_08	off_09	off_10	off_11
1	19	0	2933	1	154	89	68	98	79	118
2	27	0	4334	2	62	58	148	133	130	147
3	18	0	3785	0	80	55	76	133	153	118
4	87	64	10299	4	117	169	502	489	459	316
5	76	22	9224	1	128	180	566	479	275	256
6	50	19	8453	1	115	161	535	380	225	210
	off_12	off_13	off_14	off_15	off_16	off_17	off_18	off_19	off_20	off_21
1	119	121	211	166	170	213	252	253	190	212
2	198	213	207	282	352	344	326	340	306	316
3	106	198	161	263	308	312	369	341	295	313
4	345	340	433	346	590	648	924	1359	949	714
5	290	284	352	346	475	528	819	1119	820	738
6	252	303	276	318	417	512	705	1064	843	619
	off_22	off_23	off_24	year	month					
1	196	188	35	2010	01					
2	399	295	76	2010	01					
3	283	189	32	2010	01					
4	729	510	356	2010	01					
5	736	531	301	2010	01					
6	753	495	269	2010	01					



2. 데이터셋 읽어오기 및 전처리

- 메인 파일에 대한 전처리 작업 내용 정리
 - 2014년도는 7월까지 밖에 없어서 제외
 - 숫자형태로 되어 있는 날짜 컬럼을 date 포맷으로 변경
 - 역명에서 노선을 나타내는 () 제거
 - 연도, 월 컬럼을 추가

전처리시 어떤 작업을 할지는 데이터의 특징과 형태,
분석작업의 종류에 따라 달라진다

2. 데이터셋 읽어오기 및 전처리

- subway_latlong.csv 파일을 읽는다.

```
subname <- read.csv('subway_latlong.csv',  
                    header=TRUE, stringsAsFactors=FALSE)  
subname
```

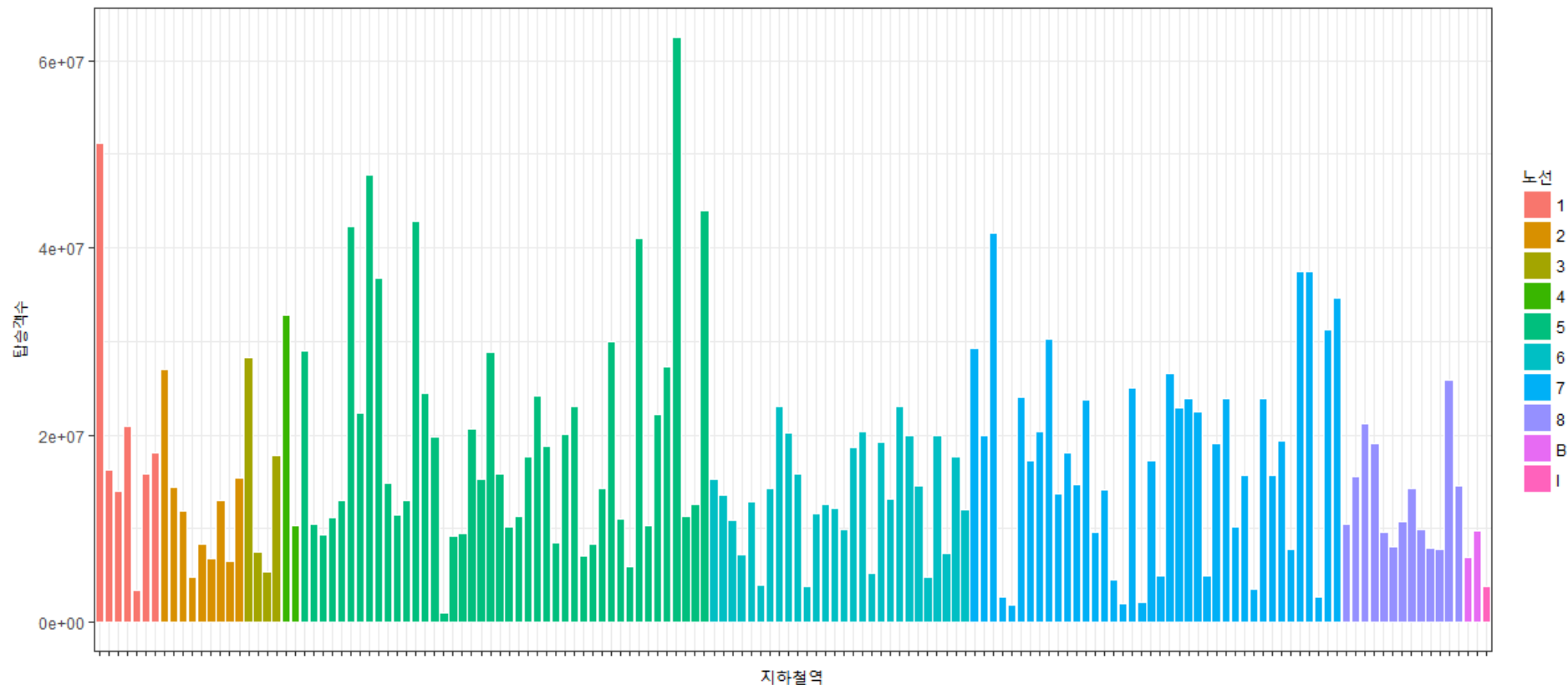
```
> head(subname)  
  STATION_CD STATION_NM LINE_NUM FR_CODE CYBER_ST_CODE XPOINT YPOINT  
1         330      교대         3      340         330 502900 1108655  
2         331 남부터미널         3      341         331 503497 1106375  
3         332      양재         3      342         332 507387 1106032  
4         333      매봉         3      343         333 510257 1106985  
5         334      도곡         3      344         334 512105 1107980  
6         335      대치         3      345         335 513907 1108850  
  XPOINT_WGS YPOINT_WGS  
1  37.49341  127.0141  
2  37.48501  127.0162  
3  37.48415  127.0346  
4  37.48695  127.0468  
5  37.49086  127.0554  
6  37.49461  127.0636
```

전처리 할 내용이 없음



3. 탑승객 상위역 분석 및 시각화

- (1) 역별 지하철 탑승객 수의 계산
 - 최종 목표



3. 탑승객 상위역 분석 및 시각화

- (1) 역별 지하철 탑승객 수의 계산
 - 절차 : **subway2** 에서 역이름을 기준으로 탑승객수(on_tot) 를 집계 (aggregate) 한 다음 **subname** 를 merge 하여 노선번호(LINE_NUM) 를 알아낸다. 그 결과를 가지고 그래프를 작성

```
> head(subway2)[,2:4]
  stat_name income_date on_tot
1   방화    2010-01-01   3084
2   방화    2010-01-02   4676
3   방화    2010-01-03   3942
4   방화    2010-01-04  10641
5   방화    2010-01-05  10129
6   방화    2010-01-06   8914
```

```
> head(subname)[,2:3]
  STATION_NM LINE_NUM
1   교대         3
2 남부터미널         3
3   양재         3
4   매봉         3
5   도곡         3
6   대치         3
```



3. 탑승객 상위역 분석 및 시각화

- (1) 역별 지하철 탑승객 수의 계산

```
tot <- aggregate(subway2[, "on_tot"],  
  by = list(stat_name = subway2$stat_name),  
  FUN = sum)
```

tot

```
> tot
```

	stat_name	x
1	가락시장	10418449
2	가산디지털단지	51204299
3	강남구청	29259632
4	강동	28951286
5	강동구청	15529354
6	개롱	10454942
7	개화산	9395169
8	거여	11136961
9	건대입구	27046964
10	고덕	13081309
11	고려대	15337466
12	고속터미널	28277563
13	공덕	42218105
14	공릉	19951424
15	광나루	22384826
16	광명사거리	41624495
17	광화문	47791232
18	광흥창	13540062
19	구산	10901284
20	군자	36716691

3. 탑승객 상위역 분석 및 시각화

```
# tot, subname 합병
cc = merge(x=tot, y=subname, by.x="stat_name",
           by.y="STATION_NM")
df2 <- data.frame(stat_name=cc$stat_name,
                  line_num=cc$LINE_NUM, on_tot=cc$x)

# 노선별로 정렬
# 그래프 그리면 노선번호별로 역이 모이도록 하기 위함
#df2 <- df2[with(df2, order(line_num)),]
df2 <- df2[order(df2$line_num),]

df2$stat_name <- factor(df2$stat_name,
                       levels=df2$stat_name)
```

3. 탑승객 상위역 분석 및 시각화

- 그래프 작성

```
plt <- ggplot(df2, aes(x=stat_name, y=on_tot,  
  fill=line_num, order=line_num))  
plt + theme_bw() + geom_bar(stat="identity",  
  colour="white") +  
  scale_x_discrete("지하철역", labels=NULL) +  
  ylab("탑승객수") +  
  scale_fill_discrete(name=c("노선"))
```

3. 탑승객 상위역 분석 및 시각화

```
plt <- ggplot(df2, aes(x=stat_name, y=on_tot,  
  fill=line_num, order=line_num))
```

* `ggplot()`를 이용하여 결과를 막대그래프로 시각화함

* `aes()` 함수를 이용하여 x축변수(`x=stat_name`), y축변수
(`y=on_tot`), 노선별 색상(`fill=line_num`)을 지정

```
plt + theme_bw() + geom_bar(stat="identity",  
  colour="white") +
```

* `theme_bw()`는 그림의 배경을 흰색으로 설정

* `geom_bar()`를 이용하여 막대그래프를 생성

+ `stat="identity"`는 y축변수의 값을 막대그래프의 높이로 지정. 디폴트는 도수(count)를 막대의 높이로 이용

+ `colour="white"`는 막대의 경계색을 흰색으로 지정. 이는 막대간의 여백을 추가하는 효과를 주어 그래프의 가독성을 높임

3. 탑승객 상위역 분석 및 시각화

```
scale_x_discrete("지하철역", labels=NULL) +  
  ylab("탑승객수") +  
  scale_fill_discrete(name=c("노선"))
```

* 그림 축에 대한 레이블 및 범례의 이름을 재지정

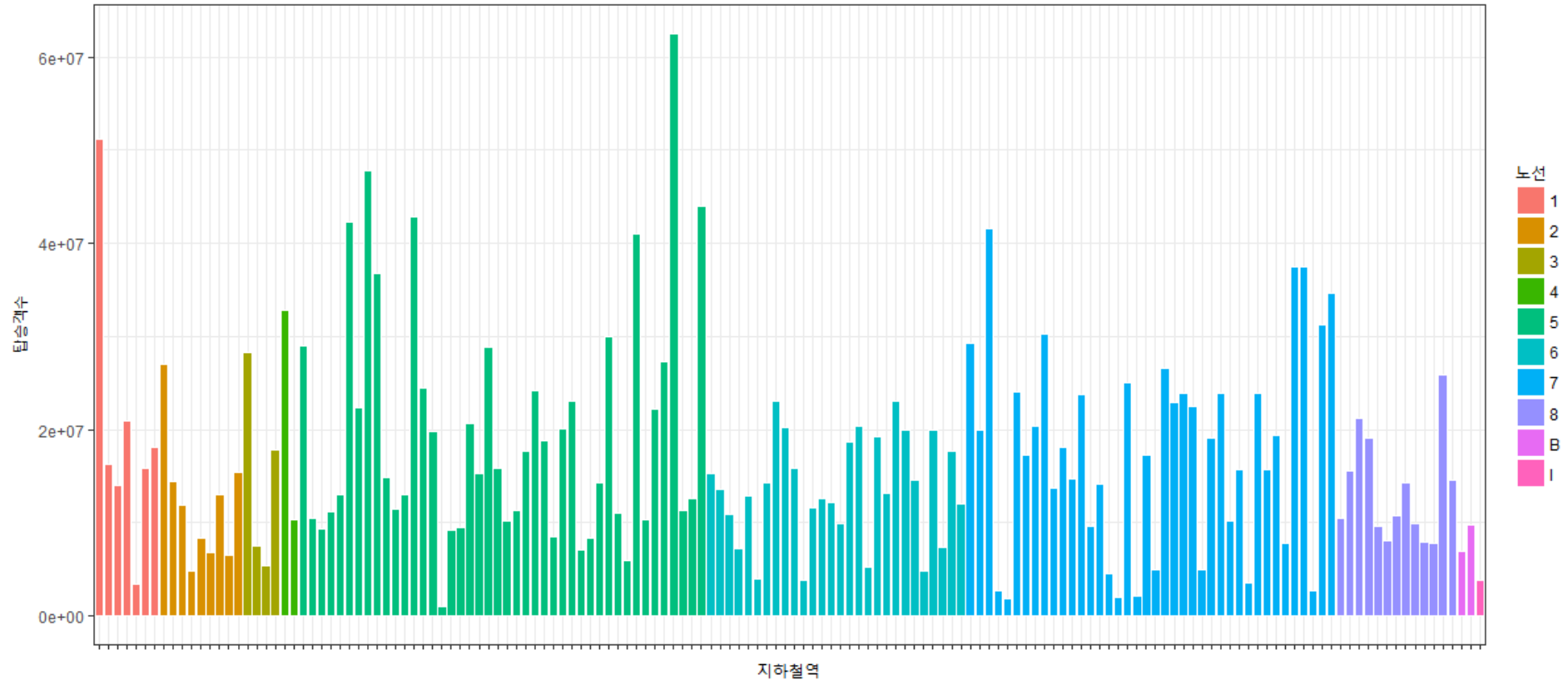
+ `scale_x_discrete()`은 이산형변수로 주어진 x축의 이름을 지정하고
`labels=NULL`을 설정하여 불필요한 역이름이 x축에 표기하지 않게 하여
번잡함을 없앴

+ `ylab()`함수를 이용하여 y축의 이름을 표기함

+ `scale_fill_discrete()`은 이산형변수로 주어진 색상에 대한 범례의 이
름을 재지정하기 위해 사용

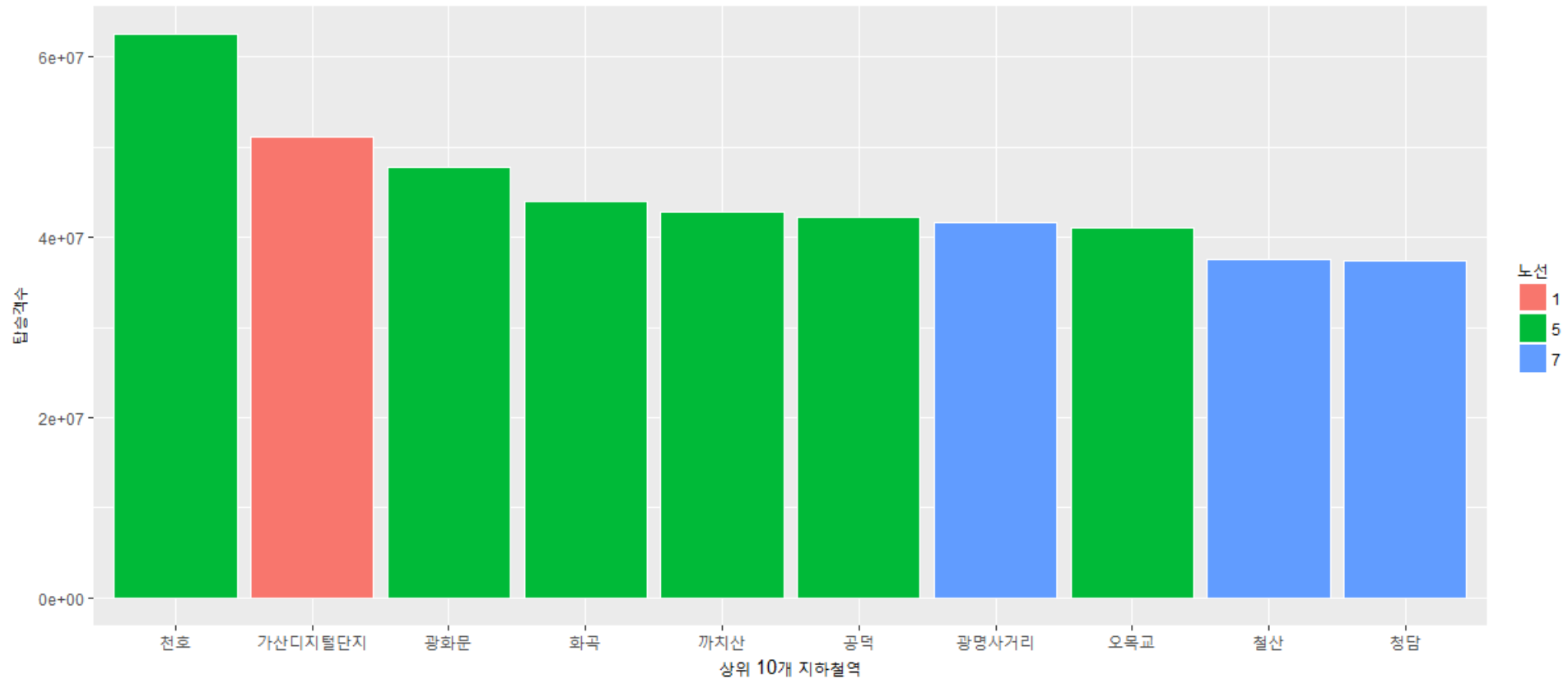
3. 탑승객 상위역 분석 및 시각화

- 결과



3. 탑승객 상위역 분석 및 시각화

- (2) 탑승객수 상위 10개 역
 - 목표 그래프



3. 탑승객 상위역 분석 및 시각화

- (2) 탑승객수 상위 10개 역
 - 처리 방법 : 앞에서 구한 역별 탑승객수 데이터(df2)에 대해 탑승객수 기준 내림차순 정렬후 상위 10개역을 뽑는다. 이 데이터를 가지고 그래프 작성

```
> head(df2)
```

	stat_name	line_num	on_tot
2	가산디지털단지	1	51204299
26	도봉산	1	16335645
27	동묘앞	1	14078349
62	석계	1	20994258
68	신길	1	3373506
85	온수	1	15847510

3. 탑승객 상위역 분석 및 시각화

- (2) 탑승객수 상위 10개 역

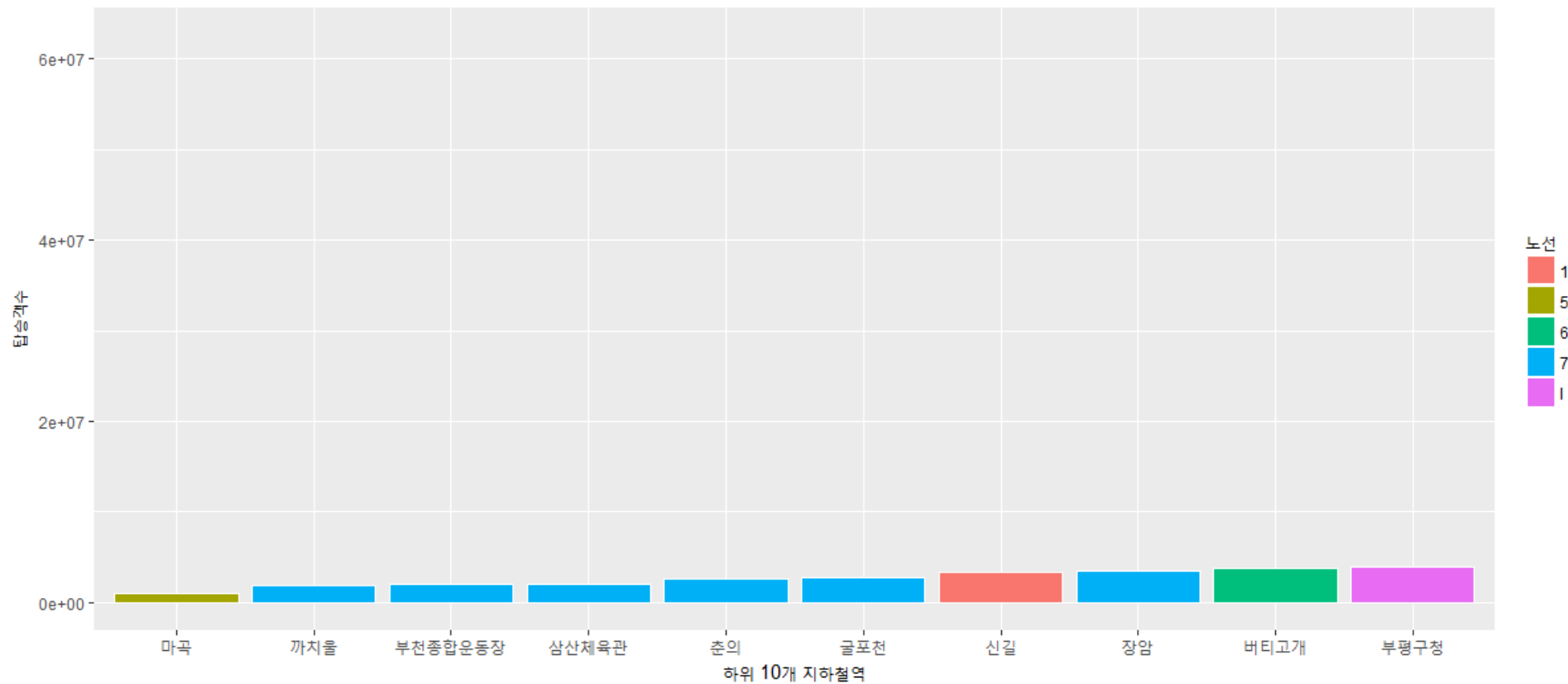
```
df3 <- df2[order(-df2$on_tot),] #내림차순 정렬
df3 <- df3[1:10,]
df3$stat_name<-factor(df3$stat_name,
                      levels=df3$stat_name)

lim <- c(0, max(df3$on_tot))
plt <- ggplot(df3, aes(stat_name, y=on_tot,
                      fill=line_num))
plt + geom_bar(stat="identity", colour="white") +
  xlab("상위 10개 지하철역") +
  scale_y_continuous("탑승객수", lim=lim) +
  scale_fill_discrete(name=c("노선"))
```



3. 탑승객 상위역 분석 및 시각화

- (3) 탑승객수 하위 10개 역
 - 목표 그래프



B :분당선
I : 인천1호선

3. 탑승객 상위역 분석 및 시각화

- (3) 탑승객수 하위 10개 역

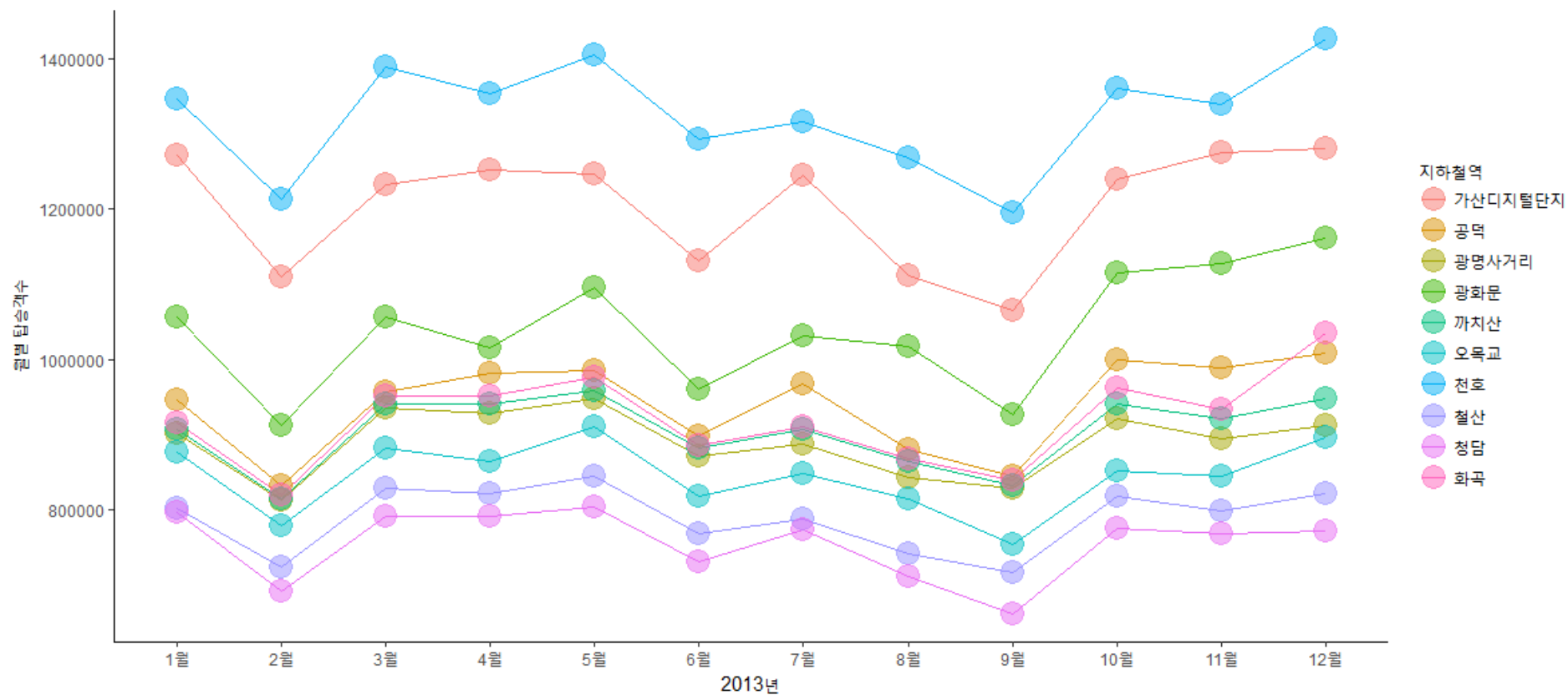
```
df4 <- df2[order(df2$on_tot),] #오름차순 정렬
df4 <- df4[1:10,]
df4$stat_name<-factor(df4$stat_name,
                      levels=df4$stat_name)

lim <- c(0, max(df2$on_tot))
plt <- ggplot(df4, aes(stat_name, y=on_tot,
                      fill=line_num))
plt + geom_bar(stat="identity", colour="white") +
  xlab("하위 10개 지하철역") +
  scale_y_continuous("탑승객수", lim=lim) +
  scale_fill_discrete(name=c("노선"))
```



3. 탑승객 상위역 분석 및 시각화

- (4) 탑승객 상위 10개역의 2013년도 월별 승객 추이
 - 목표 그래프



3. 탑승객 상위역 분석 및 시각화

- (4) 탑승객 상위 10개역의 2013년도 월별 승객 추이
 - 처리 절차
 - 상위 10개역의 이름을 수집한다 (from df3)
 - 메인 데이터셋(subway2)에서 상위 10개역에 대해 2013년도 데이터를 뽑는다
 - 월별, 역별로 데이터를 집계한다
 - 그래프를 그린다

```
> head(subway2[,c(2,3,4,46,47)])
```

	stat_name	income_date	on_tot	year	month
1	방화	2010-01-01	3084	2010	01
2	방화	2010-01-02	4676	2010	01
3	방화	2010-01-03	3942	2010	01
4	방화	2010-01-04	10641	2010	01
5	방화	2010-01-05	10129	2010	01
6	방화	2010-01-06	8914	2010	01

3. 탑승객 상위역 분석 및 시각화

- (4) 탑승객 상위 10개역의 2013년도 월별 승객 추이

```
# 상위 10개역 이름
ten.station <- df3$stat_name

# 상위 10개역의 2013년 자료
tmp <- subset(subway2, subset = stat_name %in%
              ten.station & year=="2013",
              select=c("stat_name", "on_tot", "month"))

# 월별, 역별 집계
stat_top10_2013 <- aggregate(tmp$on_tot,
                              by = list(month=tmp$month,
                                          stat_name=tmp$stat_name),
                              FUN=sum)
names(stat_top10_2013)[3] = "on_tot"
```

3. 탑승객 상위역 분석 및 시각화

월별, 역별 그래프

```
plt <- ggplot(stat_top10_2013, aes(x=month,
y=on_tot, colour=stat_name, group=stat_name))
plt <- plt + theme_classic() + geom_line() +
      geom_point(size=6, shape=19, alpha=0.5)
plt + scale_x_discrete("2013년",
labels=paste0(unique(as.numeric(month)), "월")) +
      ylab("월별 탑승객수") +
      scale_colour_discrete(name=c("지하철역"))
```



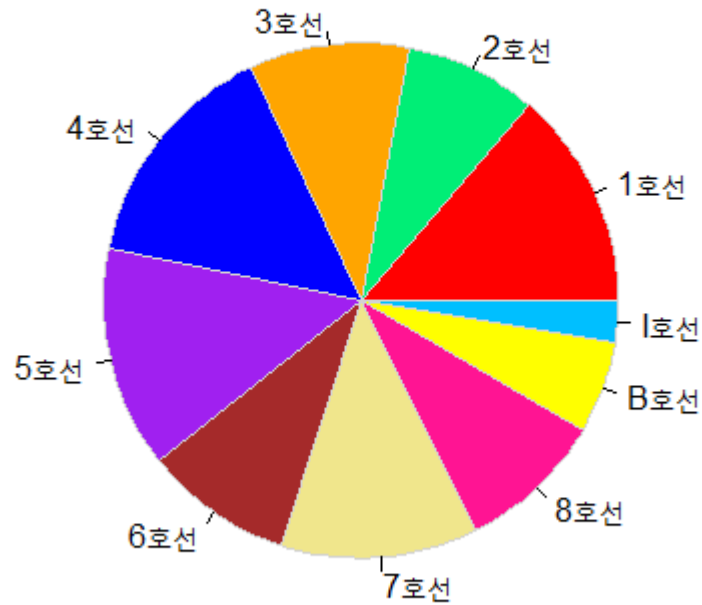
[연습문제 1]

1. 탑승객 하위 10개역의 2013년도 월별 승객 추이도를 작성하시오
2. 상위10개역의 추이와 비교하여 다른 점이 있다면 서술하시오

4. 노선별 분석 및 시각화

- (5) 노선별 역당 평균 탑승객수의 계산 및 비교
 - 목표 그래프

노선별 평균 지하철 탑승객 수



B : 분당선
I : 인천1호선

4. 노선별 분석 및 시각화

- (5) 노선별 역당 평균 탑승객수의 계산 및 비교
 - 처리 절차
 - 메인데이터셋(subway2) 와 subname 을 merge 하여 노선 번호를 가져온다
 - 노선별, 역별 탑승객수를 집계(sum) 한다
 - 노선별 탑승객수 평균을 계산한다
 - 파이그래프를 작성한다

4. 노선별 분석 및 시각화

- (5) 노선별 역당 평균 탑승객수의 계산 및 비교

```
subway3 <- merge(subway2, subname,
                  by.x="stat_name", by.y="STATION_NM")
# 노선별, 역별 탑승객 합계
tmp1 <- aggregate(subway3[, "on_tot"],
                  by = list(LINE_NUM=subway3$LINE_NUM,
                           stat_name=subway3$stat_name),
                  FUN= sum,
                  na.rm=TRUE)
names(tmp1)[3] = "on_tot"
# 노선별 평균
tmp2 <- aggregate(tmp1[, "on_tot"],
                  by = list(LINE_NUM=tmp1$LINE_NUM),
                  FUN = mean,
                  na.rm=TRUE)
names(tmp2)[2] = "on_tot"
head(tmp2)
```

4. 노선별 분석 및 시각화

그래프 작성

```
col <- c("red", "springgreen2", "orange", "blue",  
"purple", "brown", "khaki", "deeppink", "yellow",  
"deepskyblue")
```

```
pie(tmp2$on_tot,  
    labels=paste0(unique(tmp2$LINE_NUM), "호선"),  
    col=col,  
    border="lightgray",  
    main="노선별 평균 지하철 탑승객 수")
```

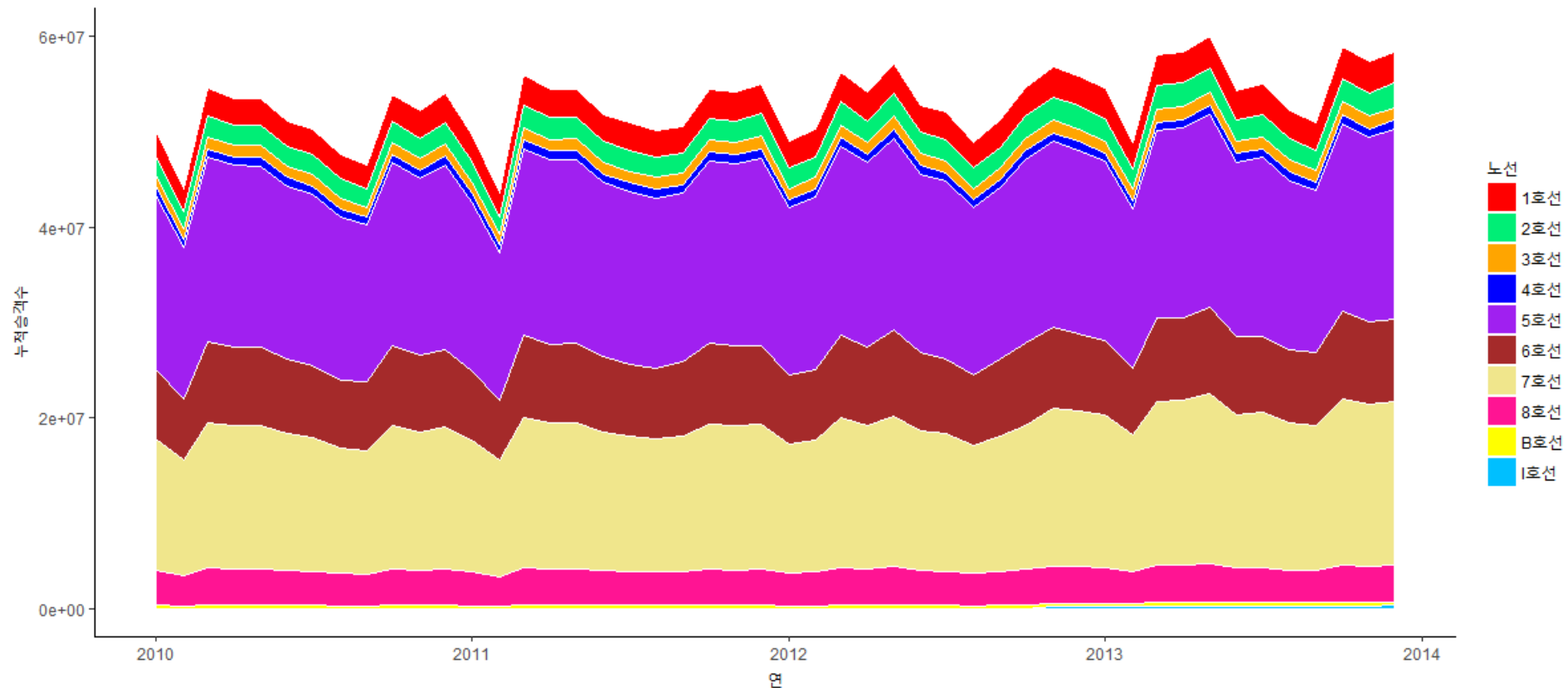


4. 노선별 분석 및 시각화

- (6) 노선별 누적 승객수의 상대비교
 - 각 노선에 대하여 월별 총탑승객수를 계산하여 이를 영역차트(area chart)로 시각화하여 전체 탑승객수에서 각 노선이 차지하는 비율의 추이를 시계열적으로 파악.
 - 단, 노선에 속하는 모든 역에 대한 정보가 없고 일부의 역만을 이용하고 있으므로 결과해석에 주의를 요한다.

4. 노선별 분석 및 시각화

- (6) 노선별 누적 승객수의 상대비교
 - 목표 그래프



B :분당선
I : 인천1호선

4. 노선별 분석 및 시각화

- (6) 노선별 누적 승객수의 상대비교

- 처리 절차

- 메인데이터셋과 subname 을 merge 한 subway3 으로 작업
- 년월컬럼을 추가
- 호선별, 년월별로 탑승객수 집계(sum)
- 영역 chart 작성

```
> names(subway3)
[1] "stat_name"      "X.U.FEFF.station" "income_date"      "on_tot"
[5] "on_05"          "on_06"            "on_07"            "on_08"
[9] "on_09"          "on_10"            "on_11"            "on_12"
[13] "on_13"          "on_14"            "on_15"            "on_16"
[17] "on_17"          "on_18"            "on_19"            "on_20"
[21] "on_21"          "on_22"            "on_23"            "on_24"
[25] "off_tot"        "off_05"           "off_06"           "off_07"
[29] "off_08"        "off_09"           "off_10"           "off_11"
[33] "off_12"        "off_13"           "off_14"           "off_15"
[37] "off_16"        "off_17"           "off_18"           "off_19"
[41] "off_20"        "off_21"           "off_22"           "off_23"
[45] "off_24"        "year"             "month"            "STATION_CD"
[49] "LINE_NUM"      "FR_CODE"          "CYBER_ST_CODE"    "XPOINT"
[53] "YPOINT"       "XPOINT_WGS"      "YPOINT_WGS"
```

4. 노선별 분석 및 시각화

```
# year-month 컬럼 추가
```

```
yearmonth <- paste(subway3$year, subway3$month,  
  "01", sep="-")
```

```
yearmonth <- as.Date(yearmonth)
```

```
tmp3 <- cbind(subway3, yearmonth)
```

```
# LINE_NUM 컬럼에 '호선' 붙이기
```

```
tmp3$LINE_NUM <- paste0(tmp3$LINE_NUM, "호선")
```

```
# 호선별, 년월별로 집계 (sum)
```

```
tmp4 <- aggregate(tmp3[, "on_tot"],  
  by = list(LINE_NUM=tmp3$LINE_NUM,  
    yearmonth=tmp3$yearmonth),  
  FUN = sum,  
  na.rm=TRUE)
```

```
names(tmp4)[3] = "on_tot"
```


4. 노선별 분석 및 시각화

그래프 그리기

```
col <- c("red", "springgreen2", "orange", "blue",  
"purple", "brown", "khaki", "deeppink", "yellow",  
"deepskyblue")
```

```
plt <- ggplot(tmp4, aes(x=yearmonth, y=on_tot,  
fill=LINE_NUM))
```

```
plt <- plt + geom_area(colour="white", size=0.2)
```

```
plt <- plt + scale_fill_manual(name="노선",  
values=col)
```

```
plt + theme_classic() + 티뮤("연도") +  
ylab("누적승객수")
```



5. 시간대별 탑승객수 분석

- (1) 전처리

	A	B	C	D	E	F	G
172301	2537	동대문역사	20130105	1289	1	5	11
172302	2537	동대문역사	20130106	544	-	13	7
172303	2537	동대문역사	20130107	3575	-	9	20
172304	2537	동대문역사	20130108	3541	4	5	24
172305	2537	동대문역사	20130109	3537	2	8	24
172306	2537	동대문역사	20130110	3580	1	6	27
172307	2537	동대문역사	20130111	3460	-	9	24
172308	2537	동대문역사	20130112	1303	2	5	19

위와 같이 데이터가 누락된 시간대들은 컬럼의 자료형이 숫자가 아닌 문자로 저장되어 있음

- (1) 문자로 된 컬럼을 숫자로 전환
- (2) 누락된 곳은 NA로 전환되는데 이들을 제거

5. 시간대별 탑승객수 분석

- (1) 전처리

```
# convert char to integer columns
toint <- apply(subway2[,5:24], 2, FUN=as.integer)
subway2[,5:24] <- toint # 문자타입을 숫자로
subway.tmp = subway2
```

```
# remove NA rows
nrow(subway.tmp)
subway.tmp <-
  subway.tmp[complete.cases(subway.tmp), ]
nrow(subway.tmp)
```

```
> nrow(subway.tmp)
[1] 171633
> subway.tmp <-
+   subway.tmp[complete.cases(subway.tmp), ]
> nrow(subway.tmp)
[1] 163799
```



5. 시간대별 탑승객수 분석

- (2) 한 역당 시간대별 평균 탑승객 추이

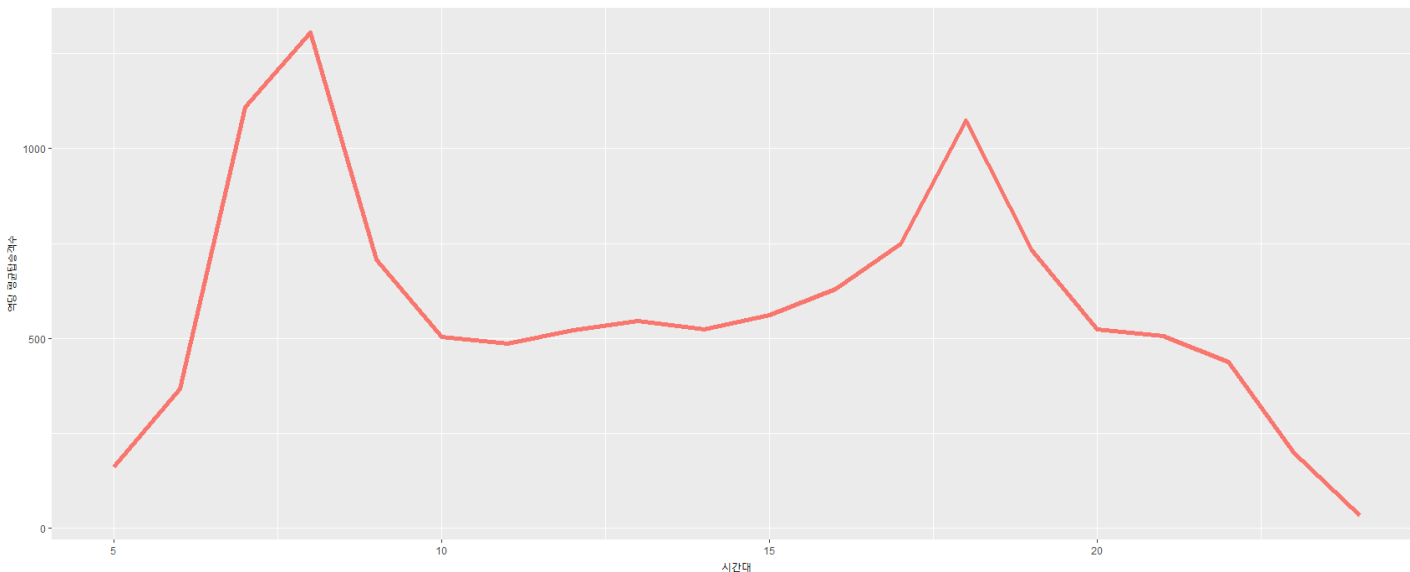
```
avg.on <- colMeans(subway.tmp[,5:24])
avg.on
# 소수점 이하 반올림
avg.on <- round(avg.on, 0)
avg.on
```

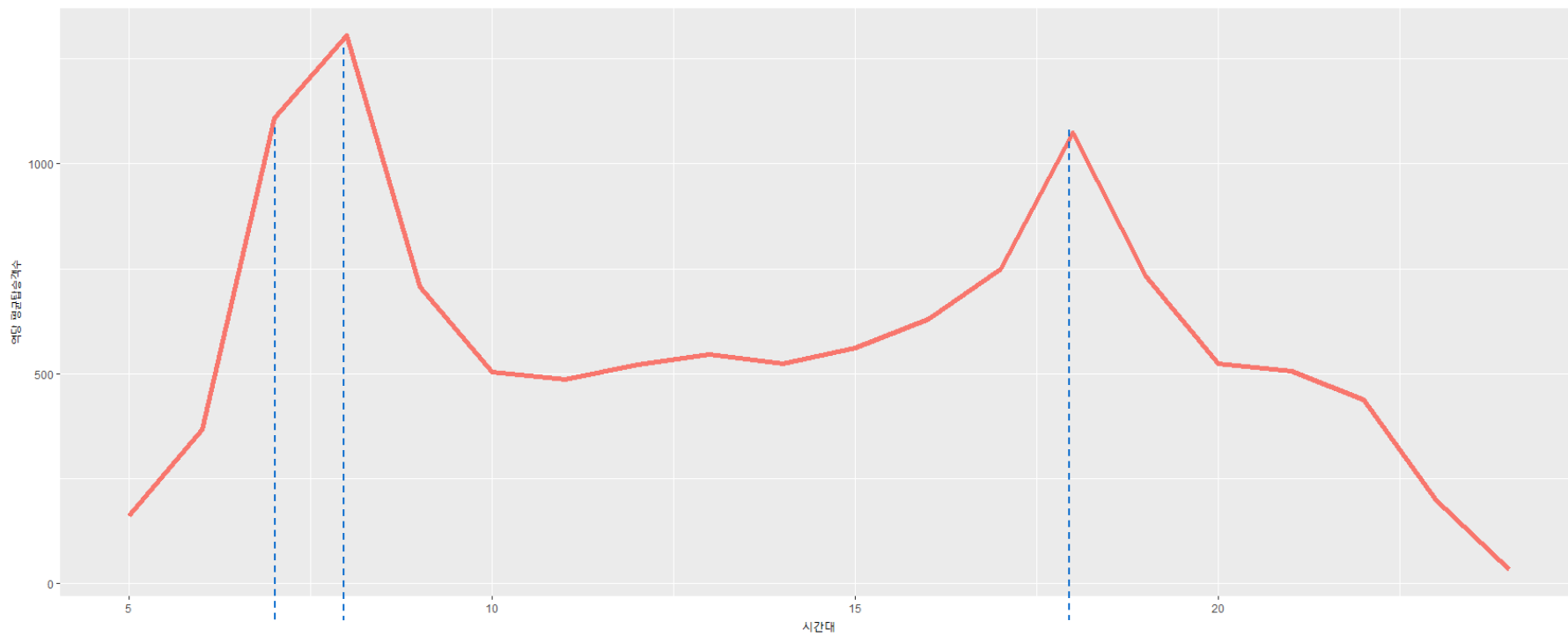
```
> avg.on <- colMeans(subway.tmp[,5:24])
> avg.on
      on_05      on_06      on_07      on_08      on_09      on_10      on_11
161.05960 367.24274 1108.17318 1305.59227 707.49318 504.82142 486.76903
      on_12      on_13      on_14      on_15      on_16      on_17      on_18
521.47226 546.11255 524.09465 560.95261 628.62228 747.56715 1073.29316
      on_19      on_20      on_21      on_22      on_23      on_24
730.54960 522.87396 506.68084 438.64870 198.54541 32.90825
> # 소수점 이하 반올림
> avg.on <- round(avg.on, 0)
> avg.on
on_05 on_06 on_07 on_08 on_09 on_10 on_11 on_12 on_13 on_14 on_15 on_16 on_17
  161   367  1108  1306   707   505   487   521   546   524   561   629   748
on_18 on_19 on_20 on_21 on_22 on_23 on_24
 1073   731   523   507   439   199    33
```

5. 시간대별 탑승객수 분석

- (2) 한 역당 시간대별 평균 탑승객 추이

```
ggplot(data.frame(on_time = 5:24, no_on = avg.on),  
       aes(x = on_time, y = no_on, color="red")) +  
  geom_line(size=2) +  
  xlab("시간대") +  
  ylab("역당 평균탑승객수") +  
  theme(legend.position="none")
```





7 8

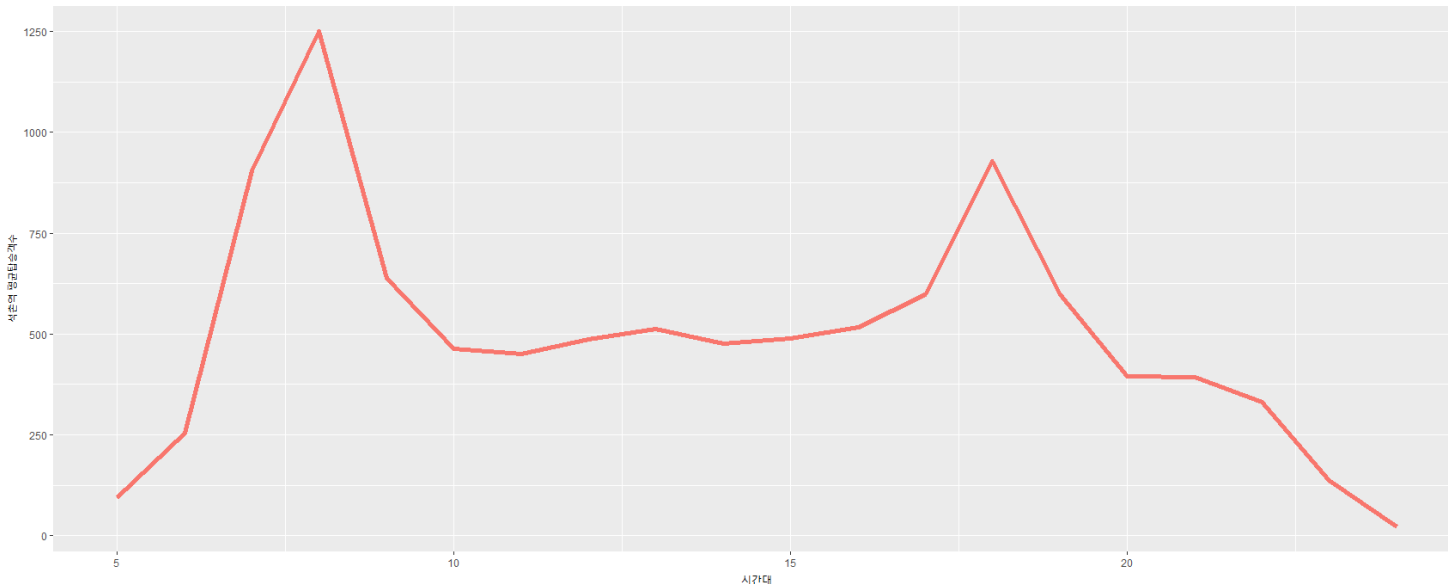
18



5. 시간대별 탑승객수 분석

- (2) 석촌역 시간대별 평균 탑승객 추이

```
seok <- subset(subway.tmp, stat_name=="석촌")
avg.on <- colMeans(seok[,5:24])
avg.on <- round(avg.on, 0)
ggplot(data.frame(on_time = 5:24, no_on = avg.on),
       aes(x = on_time, y = no_on, color="red")) +
  geom_line(size=2) +
  xlab("시간대") +
  ylab("석촌역 평균탑승객수") +
  theme(legend.position="none")
```



5. 시간대별 탑승객수 분석

- (2) 노선별 시간대별 평균 탑승객 추이

```
library(reshape2) # for melt
sub.merge <- merge(x=subway.tmp, y=subname,
                  by.x="stat_name",
                  by.y="STATION_NM")

#호선별 평균 탑승객수
subway.agg <- aggregate(sub.merge[, 5:24],
                        by=list(line_num=sub.merge$LINE_NUM), FUN=mean,)

#그래프 작성에 적합하게 데이터형태 변형
melted <- melt(subway.agg, value.name="cnt")
names(melted)[2] <- "time.zone"
head(subway.agg)
head(melted)
```



```
> head(subway.agg)
  line_num   on_05   on_06   on_07   on_08   on_09   on_10   on_11
1         1 159.74018 356.3451 1107.3547 1098.3124 597.7861 451.5849 449.5372
2         2 105.98164 154.0737 345.0691 506.8557 363.7279 290.2891 313.0607
3         3 62.70764 137.8843 465.7385 589.0193 427.4589 402.9439 455.9802
4         4 141.58799 337.7130 903.4169 1019.1089 698.6564 650.9249 741.8314
5         5 174.73521 419.0624 1292.5853 1470.9135 772.0586 556.0040 533.8717
6         6 153.71134 346.5279 1066.0693 1263.7614 655.2139 453.1850 418.8632
      on_12   on_13   on_14   on_15   on_16   on_17   on_18   on_19
1 519.5544 585.5610 595.5981 658.2518 757.9886 1011.1415 1857.0371 1232.1376
2 355.6414 400.1852 420.6366 471.5243 531.9298 643.3392 925.4733 694.4522
3 517.4585 565.4167 574.9322 640.2687 718.5198 839.2367 946.5921 731.4503
4 782.8516 833.8163 845.9979 945.8488 1019.2643 1101.9721 1283.0014 897.8691
5 560.6038 576.7342 543.4670 566.1231 616.2702 731.8495 1077.4100 700.4928
6 453.8931 465.5325 428.8049 458.9253 530.8495 625.5304 777.7006 506.7178
      on_20   on_21   on_22   on_23   on_24
1 808.6186 791.7312 558.4679 250.2506 35.60196
2 552.4044 531.2386 464.6709 231.7052 41.86400
3 639.2920 603.6541 514.2877 201.0638 28.70963
4 843.0503 723.7902 764.3635 323.1865 55.00698
5 502.0129 472.4833 400.0940 173.9759 27.82437
6 367.1161 371.0787 340.9042 172.8721 28.95030
```

```
> melt(subway.agg, value.name="cnt")
```

Using line_num as id variables

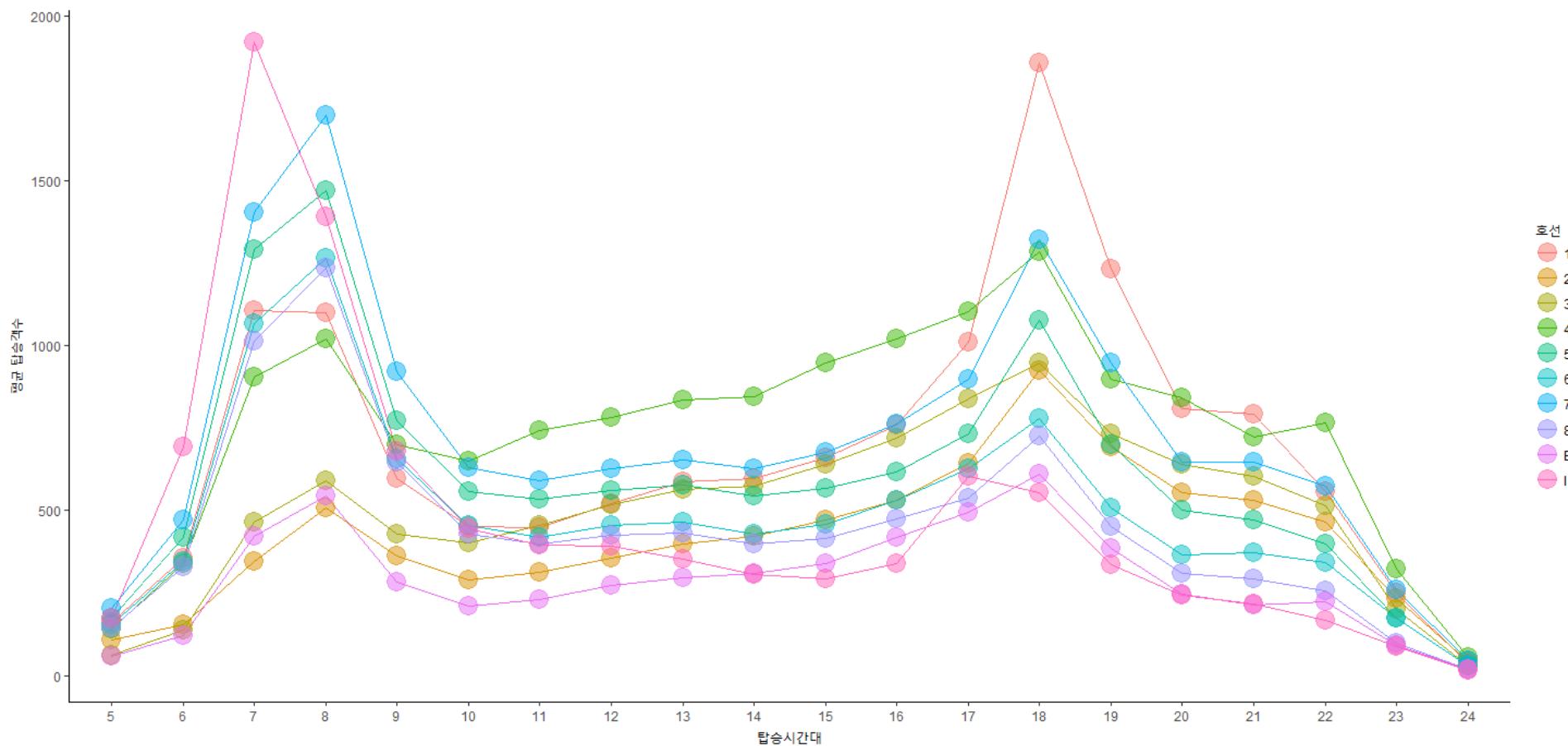
```
  line_num variable      cnt
1         1   on_05 159.74018
2         2   on_05 105.98164
3         3   on_05 62.70764
4         4   on_05 141.58799
5         5   on_05 174.73521
6         6   on_05 153.71134
7         7   on_05 204.24921
8         8   on_05 144.00710
```

#그래프 작성

```
plt <- ggplot(melted, aes(x=time.zone, y=cnt,  
  colour=line_num, group=line_num))
```

```
plt <- plt + theme_classic() + geom_line() +  
  geom_point(size=6, shape=19, alpha=0.5)
```

```
plt + scale_x_discrete("탑승시간대",  
  labels=as.character(5:24)) +  
  ylab("평균 탑승객수") +  
  scale_colour_discrete(name=c("호선"))
```



[연습문제 2]

- 천호역의 시간대별 탑승인원과 하차인원 추이 그래프를 작성하시오

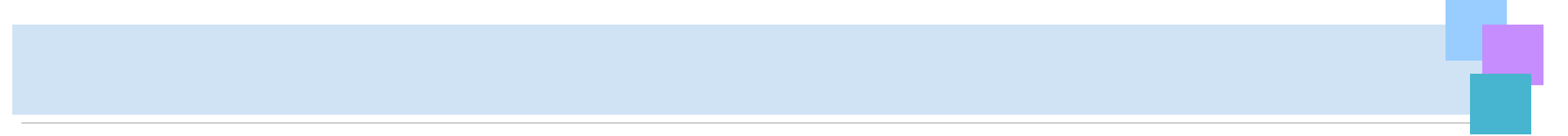
6. 구글맵 위에 탑승객수 매핑

- (1) 구글맵을 이용한 지도 매핑
 - 2012년 5월 8일 하루동안 탑승한 인원을 각 역별로 수치화하고 이를 지하철역 위치에 크기에 비례하는 원으로 표현



6. 구글맵 위에 탑승객수 매핑

```
library(ggmap)
dat1 <- subset(subway3, income_date=="2012-05-08",
               select=c("XPOINT_WGS", "YPOINT_WGS",
                        "on_tot", "stat_name", "LINE_NUM"))
Map_Seoul <- get_map(location=c(lon=126.97,
                                lat=37.55), zoom=11,
                    maptype="roadmap")
MM <- ggmap(Map_Seoul)
MM2 <- MM+geom_point(aes(x=YPOINT_WGS,
                        y=XPOINT_WGS, size=on_tot,
                        colour=as.factor(LINE_NUM)), data=dat1)
MM2 + scale_size_area(name=c("탑승객수")) +
      scale_colour_discrete(name=c("노선")) +
      labs(x="경도", y="위도")
```



```
scale_size_area(name=c("탑승객수"))
```

: 원의 크기를 설명하는 범례의 제목

```
scale_colour_discrete(name=c("노선"))
```

: 원의 색깔을 설명하는 범례의 제목

6. 구글맵 위에 탑승객수 매핑



6. 구글맵 위에 탑승객수 매핑

- (2) 2013년도의 탑승객 수 상위 10개역을 지도에 시각화

```
# 상위 10개역의 데이터 추출
```

```
stat_top10_2013 <- subset(subway2, subset =  
  year=="2013" & stat_name %in% ten.station)
```

```
# 역이름 기준으로 탑승객수 집계 (sum)
```

```
dat2 <- aggregate(stat_top10_2013[, "on_tot"],  
  by=  
    list(stat_name=stat_top10_2013$stat_name),  
    FUN=sum)
```

```
names(dat2)[2] = "on_tot"
```

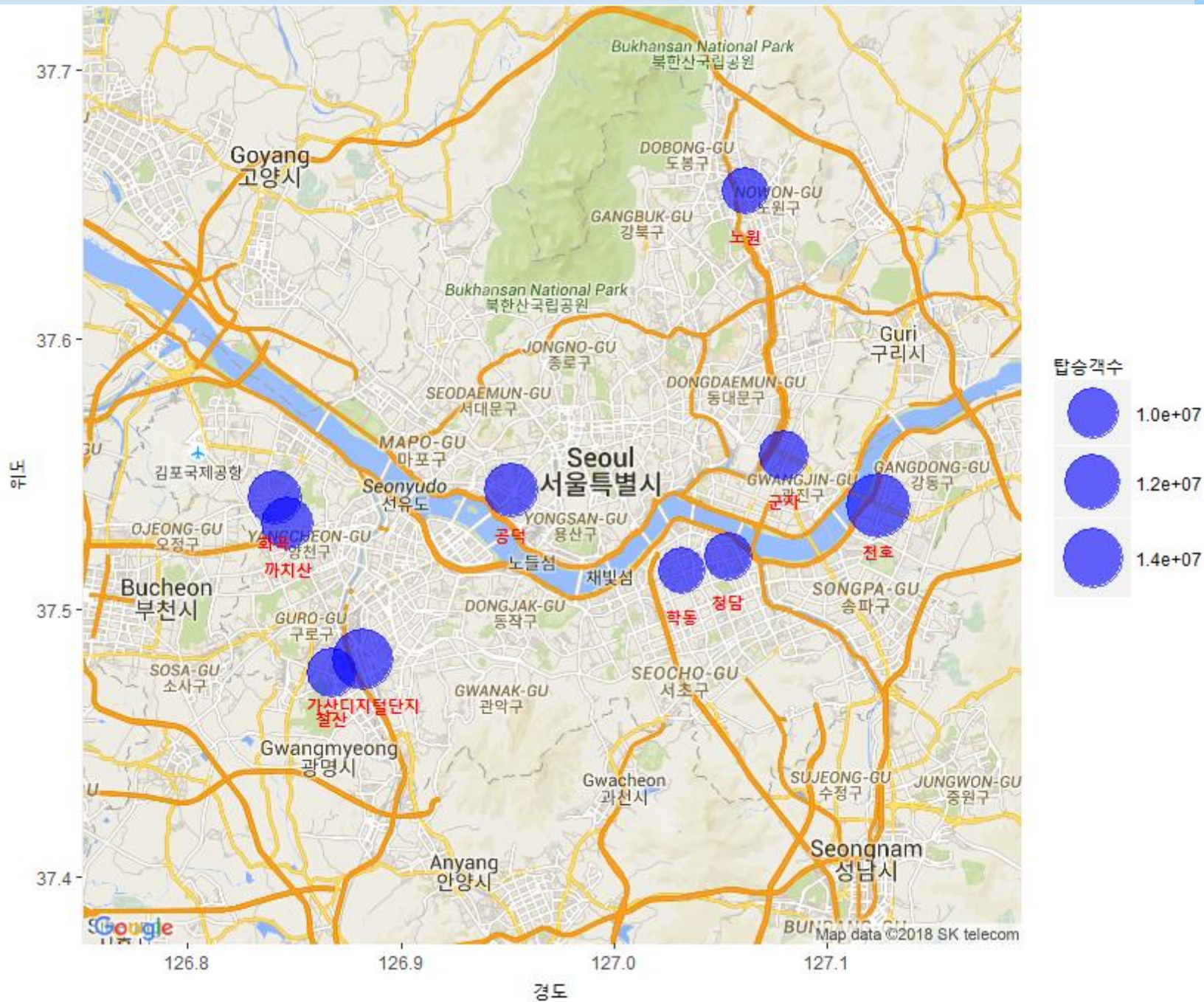
```
# 노선번호 추가하기
```

```
dat2 <- merge(dat2, subname, by.x="stat_name",  
  by.y="STATION_NM")
```

6. 구글맵 위에 탑승객수 매핑

지도그리기

```
Map_Seoul <- get_map(location=c(lon=126.97,  
                                lat=37.55), zoom=11, maptype="roadmap")  
MM <- ggmap(Map_Seoul)  
MM3 <- MM + geom_point(aes(x=YPOINT_WGS,  
                            y=XPOINT_WGS, size=on_tot), alpha=0.6,  
                        colour="blue", data=dat2)  
MM3 + scale_size_area(name=c("탑승객수"),  
                      max_size=15) +  
  geom_text(aes(x=YPOINT_WGS,  
                y=XPOINT_WGS, label=stat_name),  
            colour="red", vjust=3, size=3.5,  
            fontface="bold", data=dat2) +  
  labs(x="경도", y="위도")
```



[연습문제 3]

- 누적 탑승 인원을 각 역별로 수치화하고 이를 지하철역 위치에 크기에 비례하는 원으로 표현하시오