제 1 장

SAS 기초

이 장에서는 SAS 그래픽에서 사용할 기초적인 몇 가지 문법에 대해서 알아보자. SAS에서는 한 문장의 명령이 끝나면 세미콜론(;)으로 문장이 끝났음을 표시하며 하나의 명령문을 여러 줄에 걸쳐 쓸 수도 있고 한 줄에 여러 명령문을 사용할 수도 있다.

SAS의 시작은 SAS 아이콘의 더블 클릭이나 데이터 파일이 ASCII 파일로 준비되어 있는 경우 시작, 프로그램, SAS System 순서로 메뉴를 선택하여 할 수 있다. SAS를 처음 시작하면 그림 1.1과 같은 창이 뜬다. 이 그림의 왼쪽 창은 탐색기 창이며, 오른쪽 위는 SAS 로그창, 오른쪽 아래는 SAS 프로그램 창이다. 프로그램 창에 프로그램을 입력하거나 파일 메뉴에서 미리 만든 프로그램을 불러와서 실행할 수 있다. SAS가 정상적으로 실행되면 통계계산 결과는 출력창이 새로 만들어지면서 계산결과가 화면에 인쇄되며 그래픽이 있는 경우 그래픽은 그래픽 창에 출력이 만들어진다. SAS에서 사용하는 기본적인 파일의 확장자는 SAS 프로그램에서 사용하는 `.sas`, 로그파일에서 사용하는 `.log` 및 출력을 저장할 때 사용하는 `.lst`가 있다.

1.1. SAS의 시작과 자료의 입력

SAS를 이용하여 분석할 자료는 자료가 다른 파일로 만들어져 있을 때는 1.3.2절에서 볼 `INFILE` 명령을 사용하여 자료를 읽어들이고 SAS 프로그램 안에서 자료를 읽을 때에는 `CARDS` 문을 사용한다. 자료의 값은 줄은 맞추지 않아도 되나(경우에 따라 맞춰서 읽을 수는 있다.) 빈칸 등으로 데이터의 각 값이 분리되어 있어야 한다. 프로그램의 작성이 끝나면 프로그램을 실행시키는데 메뉴에서 사람이 달리는 모양의 버튼을 클릭하거나 `Run` 메뉴의 `Submit` 버튼을 클릭한다.

한 번 만든 SAS 프로그램 파일은 하드 디스크 등에 저장해 두면 다음에 이를 불러들여 수정하는 방법으로 좀더 편리하게 SAS를 사용할 수 있다.

SAS의 실행결과로는 로그창, 결과창 및 그래픽 창이 뜨는데 SAS 실행 중의 각종 에러 메세지나 실행에 소요된 시간 등 SAS 관련정보는 로그창에 뜨고 이 창의 내용은 확장자가 `.log`인 파일로 저장한다. 에러 메세지 없이 SAS 프로그램이 실행되었으면 계산결과가 인쇄되는데 이는 출력창에 그 내용이 저장되며 이 내용을 파일로 저장할 경우 기본 확장자가 `.lst`이다. 고해상도 그래픽이 포함된 SAS Graphics를 사용한

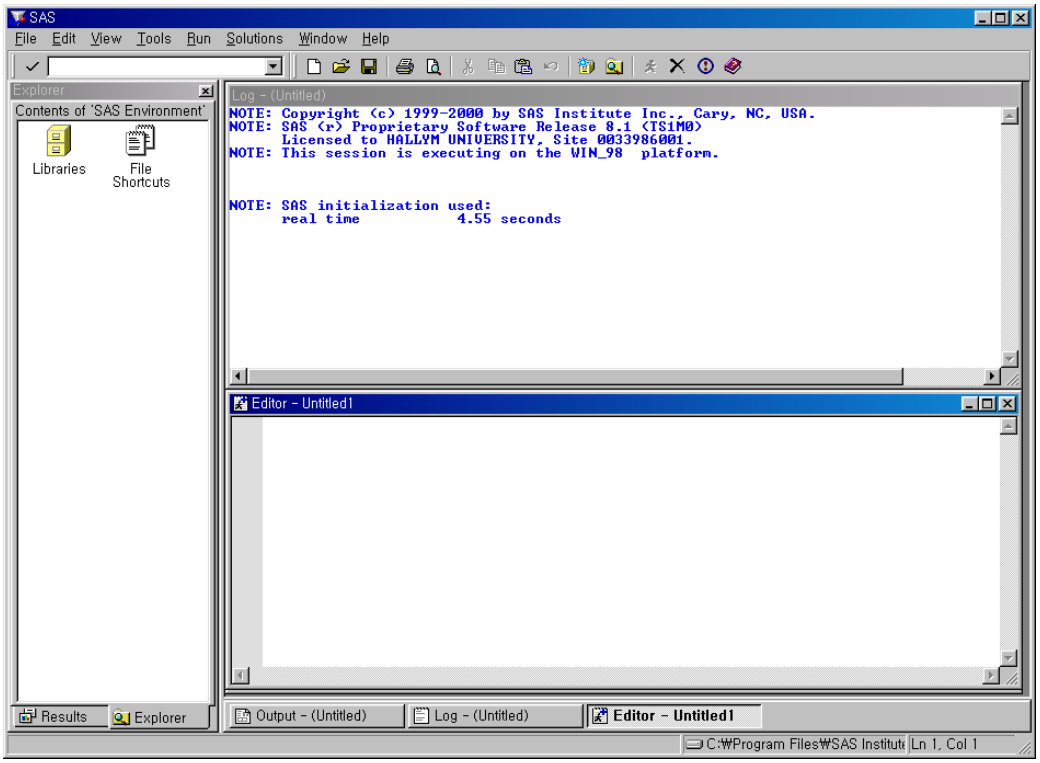

그림 1.1: SAS 창

경우에는 생성된 그래픽이 그래픽 창에 그려지며 이 그래픽은 `jpeg`, `bmp`, `ps` 등의 파일 형식으로 저장할 수 있다.

1.1.1 변수의 변환

변수의 변환은 기존의 변수를 이용하여 새로운 변수를 만들어 내는 것을 말하는 것으로, 예를 들어 키와 몸무게를 이용하여 체질량지수(Body mass index; BMI)를(이때 몸무게는 kg 단위로, 키는 m 단위로 얻는다.)

$$\text{BMI} = \frac{\text{몸무게}}{\text{키}^2} \tag{1.1}$$

로 계산하려면

```
BMI = WT/HT**2;
```

와 같이 연산을 정의하면 된다. 만일 키가 cm 단위로 측정되었으면

```
BMI = WT/(HT/100)**2;
```

으로 계산할 수 있다. 각종 연산에서 사용하는 연산자나 내장함수는 1.4.1절을 참고하기로 한다. 참고로 BMI는 Quetlet's index로 불리기도 한다.

1.2. 자료의 선언과 입력

자료의 선언은 DATA 문으로 한다. 사용법은

```
DATA sas_data options ;
```

인데 이 문장은 하나의 자료를 만들어 저장하겠다는 뜻이며 *sas_data*는 사용자가 정해주는 자료의 이름이다. 자료의 이름은 이름에 점(.)이 있는 경우와 그렇지 않은 경우로 구분되는데 한 번 점이 있는 이름을 사용하면 점이 없는 자료의 이름은 모두 WORK.을 가정한다. 즉

```
DATA A;
DATA B.DATA;
```

로 같이 두 개의 자료를 만들면 처음 자료를 부를 때는 반드시 WORK.A라고 해야한다. 즉, 점(.)을 사용하지 않은 자료는 WORK.을 붙여서 사용한다.

*options*에는

KEEP 뒤에서 읽어들일 변수 중에서 지금 만들어지는 자료에 포함할 변수의 이름을 준다.

DROP 뒤에서 읽어들일 변수 중에서 지금 만들어지는 자료에 포함하지 않을 변수의 이름을 준다.

LABEL 만드는 데이터 세트(SAS dataset)의 설명을 설정한다.

TYPE 만일 자료가 아주 특이한 형태, 즉 분산-공분산 행렬 등일 때 사용한다.

등을 사용할 수 있다. 예를 들어

```
DATA A(DROP= MON X1-X10 LABEL=A SUBSET OF THE DATA);
```

이면 WORK.A라는 SAS 데이터 세트(자료)가 만들어지며 이 자료의 이름은 LABEL에 주어진 'A SUBSET OF THE DATA'가 된다. 뒤에서 INPUT, INFILE 명령 등에서 읽은 변수 중 MON, X1부터 X10은 저장하지 않는다. SAS에서 연속적으로 번호가 붙은 변수는 -(마이너스 부호)를 사용하여 한 번에 설정가능한다. 위의 경우 X1-X10은 X1, X2, ... , X10을 의미한다.

데이터 세트는 한 번에 여러 개를 만들 수 있다. 예를 들어

```
DATA PAST(KEEP=YR1998) CURRENT(DROP=YR1998);
```

이면 뒤에서 읽을 자료 중에서 변수 YR1998은 PAST라는 이름의 데이터 세트에, 나머지 변수는 CURRENT라는 데이터 세트에 저장된다. 이 경우 나중에 YR1998을 포함한 변수들이 SET 명령이나 INPUT 명령 등으로 읽혀질 예정이다.

1.3. 자료의 입력

자료의 입력은 INPUT 명령어로 한다. 이 명령어는

```
INPUT var_list
```

로 사용하여 변수의 이름을 var_list에 주는 방식으로 사용할 수 있다. 이 방식을 사용하려면 자료가 최소한 하나 이상의 빈 칸으로 띄어져 있어야 한다. 변수의 값이 문자열이면 변수 이름 뒤에 $를 붙여야한다. 예를 들어

```
INPUT YIELD LOC $ X1;
```

은 숫자로 된 변수 YIELD와 X1 및 문자열인 변수 LOC를 읽겠다는 뜻이다.

결측치: SAS에서 결측치는 점(.)으로 표시한다. 점(.)이 아닌 특정한 자료의 값을 결측치로 설정하고자 하면 MISSING 문을 이용하여 다른 값에 대해서도 결측치로 정할 수 있다. 예를 들어

```
1 DATA A;
2 MISSING A V;
3 INPUT X Y @@;
4 CARDS;
5 1 2   2 A   3 V   4 5   5 7 . 6
6 ;
7 PROC FREQ;
8  TABLE X Y;
9 RUN;
```

로 할 경우에는 Y 값이 A와 V인 경우에는 결측치로 취급한다. 위 프로그램의 결과는

```
1                          Cumulative  Cumulative
2         X   Frequency    Percent   Frequency    Percent
3         ------------------------------------------------
4         1           1       20.0           1       20.0
5         2           1       20.0           2       40.0
6         3           1       20.0           3       60.0
7         4           1       20.0           4       80.0
8         5           1       20.0           5      100.0

9                     Frequency Missing = 1
```

```
                                        Cumulative  Cumulative
        Y    Frequency     Percent     Frequency      Percent
        -----------------------------------------------------
        2            1        25.0             1         25.0
        5            1        25.0             2         50.0
        6            1        25.0             3         75.0
        7            1        25.0             4        100.0

                      Frequency Missing = 2
```

이다.

만일 자료가 빈 칸이 없이 특정한 열에 줄을 맞추어 코딩되어 있으면

INPUT *var format*

으로 사용하여 자료가 열을 맞춘 상태라는 것을 알려줄 수 있다. 이 경우는 인접하는 값들 사이에 빈 공간이 없어도 된다. 예를 들어

```
INPUT SEX $ 7 X2 8-10 Y 1-3;
```

라고 하면 변수 SEX는 한 문자인 문자열 자료이며 일곱 번째 열에서 자료를 읽고 X2는 8,9,10 열에 맞추어진 자료이며 Y는 1,2,3 열에 맞추어진 자료인데 둘 다 숫자이다. 이 경우에 빈 칸은 숫자로 읽는 변수인 경우 0이 있는 것으로 생각한다. 만일 숫자로 읽을 변수 값이 소숫점이 필요하면 소숫점 이하의 자리수를 정할 수 있다. 예를 들어

```
INPUT SEX $ 7 X2 8-10 1 Y 1-3 2;
```

과 같이 소숫점 이하 자리수를 1과 2로 주면 X2는 8열에서 10열까지의 숫자를 읽어서 소숫점 앞에 8열과 9열에서 읽은 두 숫자를, 소숫점 뒤에 10열에서 읽은 숫자를 붙여서 숫자를 만든다. 이 때 원래 자료에 소숫점이 있으면 원래 자료의 소숫점이 우선한다.

1.3.1 SAS 명령문 안의 자료

SAS 명령문 안에 자료가 있을 때에는 CARDS;로 한다. 자료 입력이 다 끝난 뒤에는 ; 을 붙여준다.

1.3.2 자료 파일로 따로 만든 자료

만일 자료 파일이 따로 준비되어 있다면 이 자료 파일은 INFILE 문으로 읽을 수 있다. INFILE 문은

INFILE *file_name*;

로 사용한다. 예를 들어

```
INFILE 'a:/data/data1.dat';
```

와 같이 사용한다.

1.4. 연산

1.4.1 산술연산

SAS에서의 연산은 대개의 컴퓨터 언어에서 사용하는 것과 비슷한 방법으로 한다. 사칙 연산은 +, -, * 및 /로 덧셈, 뺄셈, 곱셈 및 나눗셈을 한다. ** 는 지수이다. 즉

```
Y = X**3;
```

의 결과로 만들어진 Y는 X의 세제곱이다.

이외에도 SAS의 내장함수를 사용할 수 있는데 내장함수는 표 1.1에 정리하였다.

1.4.2 논리연산

논리 연산은 주로 IF 문에서 많이 사용하는데 어떤 조건이 맞으면 참, 틀리면 거짓의 값을 갖는 연산을 말한다. 논리 연산자는 표 1.2에 정리하였다.

1.5. IF-THEN/ELSE 문

IF 문은 IF 다음의 조건이 참일 때 THEN 뒤에 주어진 명령을 실행하고 그렇지 않은 경우에는 ELSE 뒤에 주어진 명령을 실행하는 문장이다. 예를 들어

```
IF YEAR > 1995 THEN MODEL=1;
ELSE MODEL=0;
```

문의 결과는 변수 YEAR 값이 1995보다 크면 MODEL이라는 변수값을 1로 하고 그렇지 않으면 MODEL 값이 0이 된다. IF 문의 조건이 참일 때 수행할 명령이 여러 개 있는 경우에는 DO 문을 이용하면 된다. (1.6 절 참조.) ELSE 문은 생략될 수 있으며 그런 경우에는 IF 문에서 준 조건이 맞지 않는 경우에는 아무 일도 일어나지 않는다. THEN 문장 마저 생략될 경우에는 SAS 자료의 부분 집합을 취하고자 할 때 사용되는데 예를 들면 다음과 같다.

```
DATA JUNIOR;
SET POP;
IF AGE <=18;
```

표 1.1: SAS 내장 함수

함수	함수 사용 결과
ABS(X)	X의 절대값
MAX(X)	X의 최대값
MIN(X)	X의 최소값
MOD(X,Y)	X를 Y로 나눈 나머지 값.
SIGN(X)	X의 부호. X의 값이 0이면 0.
SQRT(X)	X의 양의 제곱근. 음수인 X 값에 대해서는 .을 줌.
LOG(X)	X의 자연 로그
LOG2(X)	밑이 2인 X의 로그
LOG10(X)	밑이 10인 X의 로그
GAMMA(X)	Gamma 함수. 즉 $\int_0^\infty t^{x-1}e^{-t}dt$
DIGAMMA(X)	Digamma 함수. 즉 $\log'$(GAMMA(X))
EXP(X)	e^x.
NORMAL(*value*)	표준정규분포로부터의 난수. *value*에는 0이나 5, 6, 7 자리 수의 홀수를 사용할 수 있다. (초기치로 사용됨)
SIN(X)	sine 함수 값. X의 값은 radian임
COS(X)	cosine 함수 값. X의 값은 radian임
TAN(X)	tangent 함수 값. X의 값은 radian임

표 1.2: SAS의 비교 연산자

연산자	연산 내용
= 또는 EQ	같다.
^= 또는 NE	같지 않다.
> 또는 GT	크다.
>= 또는 GE	같거나 크다.
< 또는 LT	작다.
<= 또는 LE	같거나 작다.
AND 또는 &	그리고
OR 또는 |	또는
^ 또는 NOT	부정. 아님.

위의 예는 POP라는 SAS 데이터 세트에서 나이가 18세 이하인 경우만 뽑아서 JUNIOR라는 데이터 세트를 만든 것이다.

1.6. DO 명령

DO 명령은 END 명령이 나올 때까지의 명령을 반복하라는 명령이다. 예를 들어

```
IF X <= Y THEN DO;
  X = X**2;
  Y = Y + 3;
END;
```

라고 하면 X의 값이 Y의 값보다 같거나 작으면 X는 제곱을 하고 Y에는 3을 더해준다.

DO 문은 반복문(loop)로서 사용할 수도 있는데 이 때는 DO 다음에 루프의 조건을 준다. 예를 들어

```
DO X = 1 TO  100;
  OUPUT;
END;
```

라고 하면 지금 만드는 SAS 자료에 X라는 변수가 만들어지고 길이는 100이며 값은 1부터 100이 된다. 이런 루프에서 증가시킬 값이 다르면

```
DO X = 1 TO 100 BY 2;
```

증분이 값이 다르게 줄 수 있다. 이 때 X의 값은 1, 3, 5,. . . , 99가 된다. 루프의 증분에 특별한 규칙이 없는 경우에는 해당 값을 직접 줄 수도 있다.

```
DO X = 1, 5, 8, 11, 57;
```

이라고 하면 X의 값을 1, 5, 8, 11, 57로 바꾸면서 END; 문이 나올 때까지 반복을 하게 된다.

1.7. OUTPUT 명령

OUTPUT 명령은 SAS 자료에 현재의 값을 변수값으로 갖는 변수를 만드는 명령이다. 예를 들면

```
DATA A;
 DO X = 1 TO 10;
   IF X ^> 5 THEN Y = 0;
   ELSE Y = 1;
   OUTPUT;
 END;
```

```
PROC FREQ DATA =A;
 TABLES X Y;
RUN;
```

의 결과로 변수 X와 Y가 만들어지며 그 결과는 다음과 같다.

```
                                Cumulative  Cumulative
     X   Frequency   Percent   Frequency    Percent
    ------------------------------------------------
     1           1      10.0           1       10.0
     2           1      10.0           2       20.0
     3           1      10.0           3       30.0
     4           1      10.0           4       40.0
     5           1      10.0           5       50.0
     6           1      10.0           6       60.0
     7           1      10.0           7       70.0
     8           1      10.0           8       80.0
     9           1      10.0           9       90.0
    10           1      10.0          10      100.0

                               Cumulative  Cumulative
    Y   Frequency   Percent   Frequency    Percent
  ------------------------------------------------
    0           5      50.0           5       50.0
    1           5      50.0          10      100.0
```

이는 다음과 같이 (특히 실험계획법 자료에서) 하나의 수준에서 여러 개의 관측치를 갖는 경우에 유용하게 사용할 수 있다.

```
DATA CRD;
INPUT FACTOR YIELD1-YIELD4;
YIELD = YIELD1; OUTPUT;
YIELD = YIELD2; OUTPUT;
YIELD = YIELD3; OUTPUT;
YIELD = YIELD4; OUTPUT;
DROP YIELD1-YIELD4;
CARDS;
1  2 3 4 6
2  3 5 6 7
3  3 4 5 6
;
PROC PRINT DATA=CRD;
```

```
 VAR FACTOR YIELD;
RUN;
```

의 결과는

```
 OBS    FACTOR    YIELD
   1        1        2
   2        1        3
   3        1        4
   4        1        6
   5        2        3
   6        2        5
   7        2        6
   8        2        7
   9        3        3
  10        3        4
  11        3        5
  12        3        6
```

이다. 위의 예는 세 개의 수준에서 반복수가 4인 실험자료를 읽어서 하나의 변수로 만든 것이다. 여기서 YIELD를 만들고 난 후에는 필요없는 변수인 YIELD1부터 YIELD4는 SAS 자료에서 없애도 되므로 DROP 명령을 사용하였다. 이와 유사하게 다음과 같이 만들어 볼 수도 있다.

```
DATA A;
 DO X =  1 To 3;
 INPUT Y@@;
  OUTPUT;
 END;
 CARDS;
2 3 3    3 5 4
4 6 5    6 7 6
;

PROC PRINT; VAR X Y; RUN;
```

두 개의 SAS 자료로 따로 출력을 만들 수도 있다.

```
DATA A B;
 DO X = 1 TO 10;
   IF X ^> 5 THEN OUTPUT A;
   ELSE OUTPUT B;
 END;
```

```
6 PROC PRINT DATA=A;
7 RUN;

8 PROC PRINT DATA=WORK.B;
9 RUN;
```

라고 하면 SAS 자료 A는 1부터 5까지, B는 6부터 10까지의 값을 갖는 변수 X를 포함하게 된다.

1.8. DELETE와 DROP 명령

DELETE와 DROP은 각각 필요없는 관측치와 변수를 삭제하는 명령이다. 예를 들어

```
1 IF X > Y THEN DELETE;
```

라고 하면 X의 값이 Y의 값보다 큰 경우는 모두 삭제한다.

DROP 문은 특정한 변수를 삭제하는 명령문이다. 예를 들어

```
1 DROP X1-X10;
```

의 결과는 SAS 자료에서 열개의 변수 X1에서 X10까지를 삭제하는 것이다.

1.9. KEEP 명령

KEEP 명령은 DROP 명령과 반대로 작용하는 것으로 KEEP 명령에서 준 변수를 제외한 나머지 변수는 모두 DROP하게 하는 명령이다. 예를 들어

```
1 DATA C;
2 SET A;
3 KEEP X1-X5
```

라고 하면 SAS 데이터 세트 C는 데이터 세트 A에서 X1부터 X5 이외의 변수는 모두 삭제한 자료가 된다.

1.10. SET 명령

SET 명령은 이미 만들어져 있는 다른 SAS 데이터 세트를 불러오는 명령이다. 사용법은

SET *sasdata*;

인데 *sasdata*에는 불러올 SAS 데이터 세트의 이름을 준다. 만일 이름을 주지 않으면 가장 최근에 만든 SAS 데이터 세트를 불러온다.

둘 이상의 SAS 자료를 불러올 수도 있는데 그런 경우에는 자료를 이름이 나오는 순서대로 위에서 아래로 붙여서 새 자료를 만든다. 따라서 두 개의 자료가 n_1과 n_2 개의 관측치로 되어 있다면 이 둘을 SET 명령으로 불어 온 결과로 만들어진 자료는 $n_1 + n_2$개의 관측치를 갖는다. 변수 이름이 같은 것은 하나의 변수 이름으로 저장하고 서로 다른 변수 이름이 있는 경우에는 각각의 변수를 만든다. 이 때 변수가 없는 쪽에는 결측치로 취급한다. 예를 들어

```
DATA A;
 INPUT X Y @@;
 CARDS;
 1 2 1 3 1 4 1 5 1 6
;

DATA B;
 INPUT X Z @@;
 CARDS;
 2 6 2 7 2 9 2 10
 ;

DATA C;
 SET A B;

PROC PRINT DATA =C;
RUN;
```

의 결과로 만들어진 SAS 출력의 결과는 다음과 같다.

```
             OBS    X    Y     Z

               1    1    2     .
               2    1    3     .
               3    1    4     .
               4    1    5     .
               5    1    6     .
               6    2    .     6
               7    2    .     7
               8    2    .     9
               9    2    .    10
```

나머지 변수는 해당되는 곳에만 만든다. 위의 SAS 프로그램에서 데이터 B의 변수 이름을 둘 다 X와 Y라고 하고[1] 이들 변수 값을 출력하여 보아라.

[1]이 경우 두 데이터세트의 변수의 이름이 모두 같아진다.

1.11. MERGE 명령

SET이 둘 이상의 자료를 합하여 하나로 만들 때 자료의 숫자가 늘어나는 방향으로 자료가 합해지는데 MERGE 명령은 둘 이상을 합하여 변수의 개수가 늘어나는 경우에 사용한다. 다음 예를 보자.

```
DATA SCORE;
 INPUT ID GRADE$ @@;
 CARDS;
 1 A 2 A 3 B 4 C 5 D 6 F
;
DATA GENDER;
 INPUT ID SEX$ @@;
 CARDS;
 1 M 2 M 3 M 4 F 5 F 6 F
;
DATA ALL;
 MERGE SCORE GENDER;
;
PROC PRINT DATA=ALL;
  VAR ID  SEX GRADE;
RUN;
```

이 명령의 결과는 다음과 같다.

```
   OBS    ID    SEX    GRADE
    1      1     M       A
    2      2     M       A
    3      3     M       B
    4      4     F       C
    5      5     F       D
    6      6     F       F
```

위의 경우 ID의 변숫값이 두 데이터 세트 SCORE와 GRADE에서 같은 순서로 같은 값이 있으므로 MERGE하는데 문제가 없었다. 하지만 만일 자료가 ID에 따라서 정렬이 되어 있지 않거나 일부 ID는 한 데이터 세트에만 있는 등의 경우 위와 같이 MERGE 할 경우에는 문제가 발생한다. 예를 들어

```
DATA SCORE;
 INPUT ID GRADE$ @@;
 CARDS;
 3 B 4 C 5 D 6 F 1 A 2 A
;
DATA GENDER;
```

OBS	ID	SEX	GRADE
1	1	M	A
2	2	M	A
3	3	M	B
4	4	F	C
5	5	F	D
6	6		F
7	7	F	

그림 1.2: 키(key) 값에 따른 match merging의 결과

```
 INPUT ID SEX$ @@;
 CARDS;
 1 M 2 M 3 M 4 F 5 F 7 F
;
DATA ALL2;
 MERGE SCORE GENDER;
RUN;
PROC PRINT DATA=ALL2;
  VAR ID  SEX GRADE;
RUN;
```

인 경우 MERGE의 결과는 앞의 경우와 달라지며, 같은 ID 값에 의한 매치(match)도 생기지 않는다. 이 경우 BY를 사용하여 다음과 같이

```
PROC PRINT DATA=ALL;
  VAR ID  SEX GRADE;
  BY ID;
RUN;
```

를 실행한다. 이 때 BY를 사용하기 위해서는 사전에 두 데이터 세트를 sort 하여야 하며 이를 위해서 SORT 프로시저를 추가하여야 한다. 전체 명령은 다음과 같다.

```
DATA SCORE;
 INPUT ID GRADE$ @@;
 CARDS;
 3 B 4 C 5 D 6 F 1 A  2 A
;
DATA GENDER;
 INPUT ID SEX$ @@;
 CARDS;
 1 M 2 M 3 M 4 F 5 F 7 F
```

```
;

PROC SORT DATA=SCORE; BY ID;
RUN;
PROC SORT DATA=GENDER; BY ID;
RUN;

DATA ALL2;
 MERGE SCORE GENDER;
 BY ID;
;
PROC PRINT DATA=ALL2 ;
  VAR ID  SEX GRADE;
RUN;
```

를 사용하면 그림 1.2와 같이 원하는 출력이 얻어진다.

1.12. SAS의 데이터 출력

1.10절이나 1.11절과 같이 두 개 또는 그 이상의 데이터 세트를 합한 경우나 새 변수를 계산한 경우 등에서는 새 자료를 별도의 데이터 파일로 저장할 필요가 있는 경우가 있다. 데이터 세트의 자료를 별도의 텍스트 파일로 저장하고자 할 때는 FILE 명령과 PUT 명령의 조합으로 사용하며 이들은

```
FILE 'path\filename' DLM='delimiter' ENCODING='encoding';
PUT var1 var2 ···;
```

로 사용하며

path\filename 에는 저장할 파일의 경로와 이름을 설정하고

delimiter 는 각 열을 구분할 문자(구분자)를 설정한다. DLM이 생략되면 구분자는 빈칸이 사용된다.

encoding 에는 인코딩에 사용할 규격을 설정한다. 한국어인 경우 예전엔 'euc-kr', 최근엔 'utf-8'이 사용되는 것이 바람직하다.

var1 등에는 이 파일에 저장할 변수를 설정한다.

보기 1.1. 1.11절의 데이터 세트 ALL2를 인쇄해보자.

```
DATA _NULL_;
 SET ALL2;
 FILE 'd:\htex\sas\utf-8\textout.txt' DLM=',';
```

```
 PUT ID GRADE SEX;
RUN;
```

이 명령의 결과는 새 파일 `textout.txt` 파일이 설정된 경로에 만들어지며 이 파일의 내용은

```
1,A,M
2,A,M
3,B,M
4,C,F
5,D,F
6,F,
7, ,F
```

이다.[2]

PUT 명령에서 formatted 출력도 가능하며 이에 대한 자세한 내용은 SAS 매뉴얼을 참고하기로 한다. ■

1.13. 레이블링

변수의 이름과 값에 따로 변수의 설명 및 각각의 값에 대한 설명을 추가할 수 있다.

1.13.1 변수 레이블링

변수 레이블링(labeling)이란 변수이름에 변수의 설명을 추가해주는 기능이다. 이 기능을 사용하여 변수이름에 설명을 설정할 때는 DATA 스텝에서 LABEL 명령으로 변수명 = 변수 설명을 아래의 예와 같이 나열하여 실행한다.

```
DATA  auto2;
  LABEL  ID  = "Student ID Number"
         SEX = "Gender Reported"
         GRADE = "Grade s/he got";
RUN;
```

1.13.2 변숫값 레이블링

변숫값 레이블링은 주로 범주형 자료에서 자료의 각 값에 대한 설명을 추가하는 것을 말한다. 변숫값 레이블링은 PROC FORMAT에서 VALUE 명령으로 아래의 예와 같이 설정한다.

[2] 이와 같이 콤마(,)로 값이 구분된 자료의 형식을 CSV(Comma Separated Values)라고 한다.

테이블:SEX * GRADE

SEX(Gender Reported)	GRADE(Grade s/he got)					
	A	B	C	D	Failed	합계
Female	0 0.00 0.00 0.00	0 0.00 0.00 0.00	1 20.00 50.00 100.00	1 20.00 50.00 100.00	0 0.00 0.00 .	2 40.00
Male	2 40.00 66.67 100.00	1 20.00 33.33 100.00	0 0.00 0.00 0.00	0 0.00 0.00 0.00	0 0.00 0.00 .	3 60.00
합계	2 40.00	1 20.00	1 20.00	1 20.00	0 0.00	5 100.00

결측값 빈도 = 2

그림 1.3: 변수명 및 변숫값 레이블링을 설정한 경우의 출력 결과

```
PROC FORMAT;
   VALUE  sex 0 ="Male"
              1 = "Female";
   VALUE  $model "Cad."   ="Cadillac (GM)"
                 "Chev."  ="Chevrolet (GM)"
                 "Datsun" ="Datsun (Nissan)";
RUN;
```

위의 설정에서 보는 것과 같이 수치형 자료는 포맷 형식에 $가 없지만 문자형 변수는 $로 시작한다. 이 FORMAT 프로시저만으로는 출력에 변숫값 레이블링이 반영되지 않으며 변숫값 레이블링을 반영하려면 각 프로시저에서 FORMAT 명령을 주어서 어떤 변수에 어떤 포맷을 사용할지 설정하여야 한다. 예를 들어 gender 변수엔 FORMAT 프로시저에서 정의한 sex를 사용한다면(마지막의 마침표가 필요함에 주의)

```
FORMAT gender sex.
```

로 FORMAT문을 사용하여야 한다.

앞의 데이터 세트 ALL2를 사용하여 변수명 및 변숫값 레이블링을 다음과 같이 정의하고 마지막에 교차표를 얻어보면

```
DATA  ALL3;
 SET ALL2;
  LABEL  ID  = "Student ID Number"
         SEX = "Gender Reported"
         GRADE = "Grade s/he got";
RUN;
PROC FORMAT;
```

```
   VALUE  IDlbl 7 = "Graduated";
   VALUE  $sexlbl "M" ="Male"
               "F" = "Female";
   VALUE  $gradelbl "F"   ="Failed";
RUN;
PROC FREQ DATA=ALL3;
 TABLE sex * grade;
 FORMAT sex  $sexlbl.
        grade  $gradelbl.;
RUN;
```

결과는 그림 1.3과 같다. 위 명령에서

1. 변수명 레이블링는 DATA 스텝에서
2. FORMAT 프로시저에서 변숫값 레이블링을 정의하고
3. 실제 분석 프로시저(이 경우 FREQ)에서 FORMAT 문을 사용하여

그림 1.3의 출력에서 변수명 및 변숫값 레이블링 얻어짐을 알 수 있다.

1.14. 다중응답의 처리

다중응답은 한 문항에서 두 개이상을 선택한 경우를 말한다. 예를 들어

> 좋아하는 과일을 모두 고르시오
> ① 사과 ② 귤 ③ 바나나

와 같은 문항에서 사용자는 하나도 고르지 않을 수도 있고 세 개 모두 선택할 수도 있다. 이와 같은 문항은 코딩할 때는 각 보기에서 선택하였는지 선택하지 않았는지를 표현하기 위해 보기의 갯수만큼 문항이 만들어지며 위의 경우 한 문항이지만 세 개의 보기가 있으므로 세 개의 변수로 코딩되며 각 변수는 선택여부에 따라 0과 1, 1과 2 또는 1과 결측치 등으로 코딩한다.

보기 1.2. 다음은 성별과 세 개의 메뉴 중 좋아하는 메뉴를 선택하도록 한 결과(중복 선택 가능; 1=선택됨, 0=선택되지 않음)를 SAS를 사용하여 빈도표나 교차표를 얻어보고자 한다. 이 때 원하는 빈도수는 각 메뉴를 선택한 사람의 수이다.

```
DATA A;
  INPUT  gender$ menu1-menu3;
  CARDS;
  M  1 0 0
  F  1 1 1
  M  1 0 1
  F  1 0 0
```

OBS	gender	menu
1	M	1
2	F	1
3	F	2
4	F	3
5	M	1
6	M	3
7	F	1
8	M	3
9	F	2
10	F	3

빈도
백분율
행 백분율
칼럼 백분율

테이블:gender * menu

gender	menu 1	menu 2	menu 3	합계
F	2 20.00 33.33 50.00	2 20.00 33.33 100.00	2 20.00 33.33 50.00	6 60.00
M	2 20.00 50.00 50.00	0 0.00 0.00 0.00	2 20.00 50.00 50.00	4 40.00
합계	4 40.00	2 20.00	4 40.00	10 100.00

그림 1.4: SAS의 다중응답에 대한 분석

```
  M  0 0 1
  F  0 1 1
  ;
DATA B(DROP=menu1-menu3);
  SET A;
   IF (menu1 EQ 1) THEN DO;
     menu = 1; OUTPUT;
   END;
   IF (menu2 EQ 1) THEN DO;
     menu = 2; OUTPUT;
   END;
   IF (menu3 EQ 1) THEN DO;
     menu = 3; OUTPUT;
   END;

PROC PRINT DATA = b;
 VAR gender menu;
RUN;

PROC FREQ;
 TABLE gender * menu;
 RUN;
```

이 명령의 결과는 그림 1.4이며 이 그림의 왼쪽 그림은 새로 만든 데이터 세트를 인쇄한 것이고, 오른쪽 출력은 성별에 따른 각 메뉴를 선택한 사람의 빈도수이다. ■

1.15. t-test

```
PROC TTEST options;
CLASS c_var;
VAR vars;
PAIRED p_list;
```

으로 사용하며

CLASS 에는 독립 2-표본인 평균추론에서 그룹표시를 설정한다. 따라서 독립 2-표본인 경우에만 설정한다.

VAR 은 1-표본 또는 독립 2-표본에서 평균을 추론할 변수를 설정한다. 따라서 짝비교인 경우엔 설정하지 않는다.

PAIRED 짝비교인 경우에만 설정한다.

이며 각 옵션 및 설정은

options 에는

DATA= 사용할 데이터 세트 이름을 설정한다.

ALPHA= 신뢰구간의 신뢰도를 설정한다.

H0= 귀무가설하에서의 값을 설정한다. 기본값은 0이다.

SIDES= 단측검정인지 양측검정인지 등을 설정하며 2는 양측검정(기본값), L은 단측검정(작다), U도 단측검정(크다)를 설정한다.

TOST(lower, upper) Equivalence 검정에서 얼마이상 차이가 나야 차이가 나는 것으로 할지 설정하며 한 값만 설정하면 upper 값으로 간주한다.

PLOTS 는 자료의 분포 등을 확인하기 위한 그림을 설정할 때 사용한다. PLOTS를 사용하기 위해서는 ODS 그래프를 활성화를 먼저 하여야 한다. 예를 들면,

```
ODS GRAPHICS ON;
PROC TTEST H0=0 SIDES=U PLOTS = (interval);
 PAIRED before*after;
RUN;
ODS GRAPHICS ON;
```

이다. 요구할 수 있는 그림은

PLOTS=(*plot_list*);

으로 사용하며 예를 들면

```
plots=(histogram boxplot interval qq profiles agreement)
```

로 사용한다. 위의 경우는 요구할 수 있는 모든 그림의 종류이며 이 경우엔 `all`로 대치할 수 있다. 이들 그림 중 `profiles`와 `agreement`는 짝비교인 경우(`PAIRED`)에 적용되며 나머지는 모두에 적용된다. 각 옵션에 대한 그림의 종류는 다음과 같다.

`qq` 는 Quantile-Quantile 산점도를 그려 자료가(자료의 차이가) 정규분포인지 확인할 수 있도록 한다.

`interval` 모평균에 대한 신뢰구간의 그림을 제공한다.

`profiles` 짝비교인 경우 전과 후의 값을 연결한 산점도를 그려 전후 차이를 그림으로 확인할 수 있게 한다.

`agreement` 짝비교인 경우 가로축과 세로축에 각각 전과 후의 값을 사용한 산점도를 그려 전과 후의 차이를 볼 수 있도록 한다.

c_var 독립 이표본 검정에서 그룹을 표시하는 변수를 설정한다.

p_list 짝비교를 할 경우 짝을 설정한다. 짝을 설정하는 방법은

PAIRED 명령	결과
`PAIRED A*B;`	`A`−`B`
`PAIRED A*B C*D;`	`A`−`B` and `C`−`D`
`PAIRED (A B)*(C D);`	`A`−`C`, `A`−`D`, `B`−`C`, and `B`−`D`
`PAIRED (A1-A2):(B1-B2);`	`A1`−`B1` and `A2`−`B2`

이다.

보기 1.3. 다음은 BMI 자료에 대한 t-검정: 일표본 양측검정, 일표본 단측검정 및 독립이표본 검정을 실행한 보기이다.

```
1  DATA BMI;
2   INFILE 'd:/htex/sas/utf-8/sas/bmi.txt';
3   INPUT height weight year religion$ gender$ marriage$;
4   bmi = weight/(height/100)**2;
5   age = 2000 - year;
6   cbmi = .;
7   IF (bmi < 18.5) THEN cbmi=1;
8   IF (bmi >= 18.5 && bmi < 23) THEN cbmi=2;
9   IF (bmi >= 23 && bmi < 25) THEN cbmi=3;
10  IF (bmi >= 25 && bmi < 30) THEN cbmi=4;
11  IF (bmi >= 30) THEN cbmi=5;
12 RUN;

13 PROC FORMAT;
14  VALUE cbmiv 1 = 'Underweighted'
```

N	Mean	Std Dev	Std Err	Minimum	Maximum
177	162.1	5.4727	0.4114	150.0	180.0

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
162.1	161.3	162.9	5.4727	4.9558	6.1110

DF	t Value	Pr > |t|
176	5.10	<.0001

그림 1.5: 일표본 양측검정의 결과

```
15              2 = 'Normal'
16              3 = 'Overweighted'
17              4 = 'Slight Obesity'
18              5 = 'Obsesity';
19  VALUE $sexlbl "M" ="Male"
20                "F" = "Female";
21 RUN;

22 PROC TTEST H0 = 160;
23  VAR HEIGHT;
24 RUN;

25 PROC TTEST H0 = 160 SIDES=L;
26  VAR HEIGHT;
27 RUN;

28 PROC TTEST;
29  CLASS gender;
30  VAR bmi;
31  FORMAT gender $sexlbl.;
32 RUN;
```

첫 번째 TTTEST 프로시저는 키에 대해 귀무가설 $H_0 : \mu = 160$ 대 대립가설 $H_1 : \mu \neq 160$를 실행한 것으로 결과는 그림 1.5와 같으며 두 번째 TTTEST 프로시저는 키에 대해 귀무가설 $H_0 : \mu = 160$ 대 대립가설 $H_1 : \mu < 160$를 실행한 것으로 결과는 그림 1.6과 같다. 세 번째 TTTEST 프로시저는 남자와 여자의 체질량지수(BMI)가 차이가 있는지 검정한 것으로 μ_M과 μ_F를 각각 남자와 여자의 체질량지수의 평균이라 할 때 $H_0 : \mu_M - \mu_F = 0$ 대 대립가설 $H_0 : \mu_M - \mu_F \neq 0$를 실행한 것으로 결과는 그림 1.7과 같다.■

보기 1.4. 짝비교 예제로 전과 후를 비교해 보자.

N	Mean	Std Dev	Std Err	Minimum	Maximum
177	162.1	5.4727	0.4114	150.0	180.0

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
162.1	-Infty	162.8	5.4727	4.9558	6.1110

DF	t Value	Pr < t
176	5.10	1.0000

그림 1.6: 일표본 단측검정의 결과

gender	N	Mean	Std Dev	Std Err	Minimum	Maximum
Female	158	20.0114	1.8123	0.1442	16.0037	25.5367
Male	19	22.7344	2.4869	0.5705	18.4139	25.9516
Diff (1-2)		-2.7230	1.8928	0.4596		

gender	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev	
Female		20.0114	19.7267	20.2962	1.8123	1.6321	2.0376
Male		22.7344	21.5358	23.9331	2.4869	1.8791	3.6777
Diff (1-2)	Pooled	-2.7230	-3.6301	-1.8159	1.8928	1.7136	2.1142
Diff (1-2)	Satterthwaite	-2.7230	-3.9491	-1.4969			

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	175	-5.92	<.0001
Satterthwaite	Unequal	20.363	-4.63	0.0002

Equality of Variances				
Method	Num DF	Den DF	F Value	Pr > F
Folded F	18	157	1.88	0.0415

그림 1.7: 독립 이표본 양측검정의 결과

```
1 DATA Speak;
2   INPUT before after @@;
3  CARDS;
4   5 4 1 0 0 0 4 2 4 1 5 3 3 3 3 1 2 2 4 1
5 ;
6 RUN;
```

N	Mean	Std Dev	Std Err	Minimum	Maximum
10	1.4000	1.1738	0.3712	0	3.0000

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
1.4000	0.7196	Infty	1.1738	0.8074	2.1429

DF	t Value	Pr > t
9	3.77	0.0022

그림 1.8: 짝비교의 결과

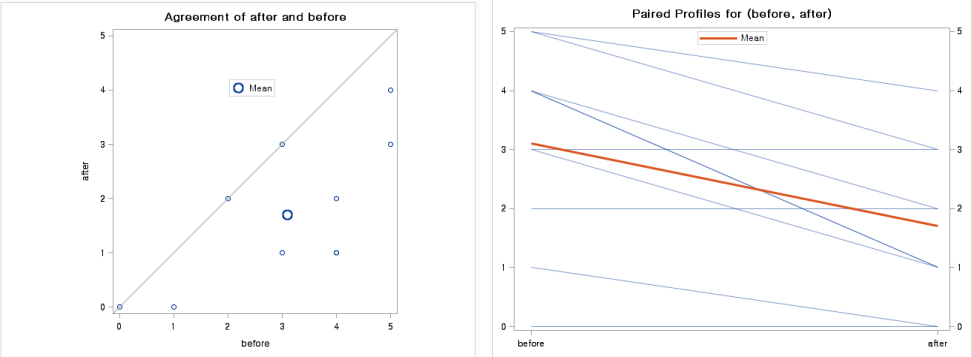

그림 1.9: Agreement plot과 profile plot

```
7 PROC TTEST H0=0 SIDES=U;
8  PAIRED before*after;
9 RUN;
```

그림 1.9는 자료의 전과 후의 값에 대한 agreement plot과 profile plot이다. ■

1.16. 분산분석

SAS 분산분석은 GLM 프로시저를 사용할 수 있으며 사용법은 아래와 같다.

Parameter	Estimate		Standard Error	t Value	Pr > |t|	95% Confidence Limits	
Intercept	79.80000000	B	3.67083097	21.74	<.0001	72.01818598	87.58181402
pgm 1	-5.40000000	B	5.19133894	-1.04	0.3137	-16.40514693	5.60514693
pgm 2	-1.60000000	B	5.19133894	-0.31	0.7619	-12.60514693	9.40514693
pgm 3	-11.20000000	B	5.19133894	-2.16	0.0465	-22.20514693	-0.19485307
pgm 4	0.00000000	B	.	.	.	.	.

그림 1.10: SOLUTION 옵션과 CLPARM 옵션을 동시에 사용한 결과

```
PROC GML g_options;
   CLASS c_var;
   MODEL resp(event_opt) = c_var1 c_var2 ... / model_opt ;
   MEANS effect / means_opt;
   CONTRAST 'label' constants / con_opt;
   ESITMATE 'label' constants / est_opt;
   LSMEANS effect / lsm_opt;
```

과 같이 사용하며 옵션과 변수의 설정은 다음과 같다.

g_options 에는 DATA, PLOTS 등의 옵션을 사용하여 각각 사용할 데이터세트와 필요한 그림을 얻을 수 있다. PLOTS에는 ALL, NONE, BOXPLOT, DIAGNOSTICS, LINESPLOT 등을 사용할 수 있다.

CLASS 모형에서 처리를 나타내는 변수를 설정한다.

MODEL 문은 종속변수와 독립변수를 사용하여 분석 모형을 설정한다. *는 상호작용을 의미하며 |는 주효과와 상호작용 모두를 의미한다.

model_opt 에는

SOLUTION 옵션을 사용하면 각 수준별 평균을 구할 수 있는데 이때 각 수준별 평균은 특정한 범주를 기준으로 그 값과의 차이를 출력하게 된다. 그림 ??은 처리 pgm의 값이 4일때를 기준으로 각 수준별 평균의 차이를 계산한 것으로 수준 4에서의 평균은 INTERCEPT와 수준 4의 값(0)과의 합인 79.8이며 수준1일 때는 79.8과 -5.4의 합인 74.4 등이다.

CLPARM 옵션을 사용하면 GLM 프로시저의 출력에서 신뢰구간을 계산한다. 그림 fig:glmsolution의 오른쪽 출력인 신뢰구간은 MODEL 문에서 CLPARM 옵션을 사용하여 추가로 만들어진 출력이며 이 출력에 의하면 수준 4에서의 평균에 대한 95% 신뢰구간은 $(72.02, 87.58)$이며 수준 1의 평균 $\mu_1 - \mu_4$의 신뢰구간은 $(-16.41, 5.61)$ 등이다. CLPARM 옵션을 사용하면 기본값으로

95% 신뢰구간이 계산되는데 만일 다른 신뢰도를 사용하고자 하면 ALPHA 옵션을 추가로 사용하여 신뢰도를 설정할 수 있다. 예를 들어

```
MODEL resp=pgm / CLPARM ALPHA=0.01;
```

을 사용하면 99% 신뢰구간을 얻을 수 있다(즉, ALPHA의 기본값은 0.05).

등을 설정할 수 있다.

CONTRAST 는 대비검정을 하기 위한 설정으로 *label*에는 적절한 이름을 설정하고 constants에는 대비에 사용하는 계수를 설정한다.

means_opt 에는 다중비교와 등분산성이 성립하지 않을 때의 추론을 얻도록 설정할 수 있다.

DEPONLY 종속변수의 평균 계산만 한다.

LSD Least Squares Distance 다중비교를 출력한다.

TUKEY Tukey 다중비교를 실행한다.

BON Bonferroni 다중비교를 계산한다.

SCHEFFE Scheffè 다중비교를 얻는다.

LINES 다중비교의 결과를 그림으로 출력한다.

HOVTEST 그룹간 등분산성 검정(Levene 검정)을 얻는다.

WELCH 수준간 등분산성이 성립하지 않을 때 사용할 수 있는 Welch 검정을 수행한다.

ESTIMATE 설정은 대비와 같은 방법으로 하며 주어진 constants에 대한 선형조합의 추정값과 대비검정의 결과(이 경우 t-검정)을 출력한다. 이때 MODEL 문에서 CLPARM 옵션이 주어지면 신뢰구간이 출력된다. 사용법은

ESTIMATE 'label' [INTERCEPT 1] $c\_var\ c_1 c_2 \cdots c_a$;

이며 한 수준에서의 평균값을 얻으려면 INTERCEPT 1 옵션을 사용하고 해당 수준에 1을 나머지는 0을 설정하고 수준간 차에 대한 추론이 필요하면 해당 수준 위치에 1과 -1을, 나머지 수준은 0을 설정한다. 예를 들어 4개의 수준이 있는 일원배치에서 수준1의 평균, 수준 1과 2의 차이, 수준 1과 4의 차이를 얻고자 한다면

```
ESTIMATE 'pgm1' INTERCEPT 1 pgm 1 0 0 0;
ESTIMATE 'pgm1 - pgm2' pgm 1 -1 0 0;
ESTIMATE 'pgm1 - pgm4' pgm 1 0 0 -1;
```

Parameter	Estimate	Standard Error	t Value	Pr > |t|	95% Confidence Limits	
pgm1	74.4000000	3.67083097	20.27	<.0001	66.6181860	82.1818140
pgm1 - pgm2	-3.8000000	5.19133894	-0.73	0.4748	-14.8051469	7.2051469
pgm1 - pgm4	-5.4000000	5.19133894	-1.04	0.3137	-16.4051469	5.6051469

그림 1.11: ESTIMATE 문과 CLPARM 옵션을 동시에 사용한 결과

을 명령할 수 있고 결과는 그림 glmestimate과 같다. label에는 각 결과를 표시하기 위한 적절한 문자열을 설정하여 출력에서 구분할 수 있도록 한다. 이 출력에서 보이는 신뢰구간은 MODEL 문에서 CLPARM 옵션을 설정하여 추가로 얻는 출력이다.

LSMEANS 처리변수 effect의 각 수준에 따른 평균값을 계산한다. LSMEANS에는 PLOTS를 추가하여 필요한 그림을 그릴 수 있다. 예를 들어

```
1 LSMEANS edu /PLOTS=MEANPLOT(CL);
```

은 edu의 각 수준에 따른 평균값의 산점도와 신뢰구간을 표시하는 그림을 얻게 된다. MEANPLOT의 옵션으로는 위의 CL에 더하여 CONNECT, CLBAND, ASCENDING, DESCENDING 등을 추가로 설정할 수 있다.

1.17. 로지스틱 회귀분석

SAS에서 이항, 다항 및 proportional 로지스틱회귀분석은 모두 같은 프로시저로 계산결과를 얻을 수 있다.

1. 이항 로지스틱은 반응변수의 가능한 값이 두 개이면 별도의 설정없이 얻어진다. 기준이 되는 값의 설정은 ORDER나 DESC 옵션 등으로 설정한다.
2. 다항 로지스틱은 MODEL 문에서 옵션 LINK에 GLOGIT을 설정한다. 반응변수의 기준이 되는 값은 ORDER나 DESC 옵션 등으로 설정한다.
3. 비례 오즈 로지스틱은 반응변수의 서로 다른 값의 개수가 두 개 이상이면 기본으로 계산한다.

LOGISTIC 프로시저는 다음

```
PROC LOGISTIC options;
  CLASS c_var c_opt;
  MODEL resp(event_opt) = x1 x2 ... /m_opt;
  OUTPUT o_opt;
```