응용통계연구 제 8권 1호
1995년 3월 pp. 75-87.

컴퓨터그래픽스에 의한 이원 분산분석1)

허 문 열2)

요 약

데이타 분석 과정에서 일부 데이타를 변화시키거나 또는 적용하고자 하는 통계적 모형의 성격을 변화시킬 때 이러한 변화가 분석 결과에 어떤 영향을 미치는가를 즉시에 알아보기위해 동적그래픽스의 기법을 이용한다. 본 논문에서는 분산분석 과정을 동적그래픽스로 구현하는 방법과 이를 구현하는 데 필요한 여러가지 형태의 플롯을 설명하고 이원 분산분석 모형에서 구현한 동적 그래픽스 소프트웨어를 소개한다.

1. 동적그래픽스

데이타를 플롯에 의해 표현하면 통계적 분석만으로 얻기 어려운 많은 정보를 얻을 수가 있다. 더욱이 최근에는 정적인 방법에 의해 플롯을 표현하는 한계에서 벗어나 동적인 방법에 의해 플롯을 표현하므로서 데이타 분석에 새로운 방향을 제시하고있다. 통계적인 의미에서 동적 그래픽스는 데이타를 변화시키거나 플롯의 성격을 변화시킬 때 이 결과가 순간적으로 플롯에 영향을 주는 것을 의미한다. 동적 그래픽스를 통계적 분석에서 이용하는 목적은 다음의 두 가지로 집약할 수있다.

1) 데이타의 변화가 플롯에 미치는 영향의 탐색

예: 일련의 관측값들을 이용하여 히스토그램을 그리거나 통계량을 계산할 때 이중에서 하나 또는 일부의 관측값을 변화시키므로써 이 관측값들의 변화가 히스토그램 또는 통계량에 미치는 영향을 조사하고자 하는 경우.

2) 플롯의 성격 변화가 플롯에 미치는 영향의 탐색

예: 히스토그램의 모양이 막대의 수에 따라 변화하는 상황을 파악하고자 할 때, 또는 varimax 법에 의해 요인 분석을 수행할 때 축의 회전 각도를 변화할 때 이것이 모형에 미치는 영향을 분석하고자 하는 경우.

동적 그래픽스는 다변량 자료의 분석이나 통계적 모형의 진단 분야에서 특히 유용성을 발휘하였다(1, 2, 4, 6). 그러나 분산분석의 경우 범주형 자료로 이루어지는 요인들의 결합 내에서 반응값을 갖는 자료를 고려하여야 하는 특수성 때문에 데이타를 그래프로 표현하는 방법이 매우 제한적이었다. 본 논문에서는 분산분석에서 동적그래픽스를 구현하는 방법들을 제시하고 데이타를 시각적으로 표현하는 새로운 방법인 FEDF와 더불어 교호작용 플롯, side-by-side 플롯

1) 본연구는 1992년 한국학술진흥재단의 연구소 지원 학술연구 조성비에 의해 연구되었음.
2) (110-745) 서울시 종로구 명륜동3가 53, 성균관대학교 경상대학 통계학과.

등 동적그래픽스를 위해 필요한 여러 가지 플롯들과 이원 분산분석 모형에서 개발된 동적그래픽스 소프트웨어를 소개한다.

2. 분산분석 모형에서 동적그래픽스의 방법

Becker등(2)은 동적 그래픽스를 위한 방법으로 다음과 같은 방법들을 제시하였다.

라벨링(labeling)과 위치지정(locating)
삭제(deleting)
연결(linking)

분산분석 모형을 위한 동적그래픽스 구현에서도 이러한 방법들이 기본적으로 적용되어야 한다. 그러나 분산분석 모형의 특수성을 고려할 때 이상의 방법들을 적용하는 과정에서 적절한 보완이 요구되며 몇 가지 방법이 추가로 필요하다. 이들을 정리하면 다음과 같다.

라벨링과 위치지정: 플롯에서 선택된 점에 라벨을 붙여 주고 또 관심이 있는 점의 위치를 플롯에서 지정해 주는 기능이다. 분산분석 모형의 경우 각 관측값은 해당 요인들이 정해진 수준을 취할 때 나타나는 반응값으로 생각할 수 있다. 따라서 라벨링의 기능은 선택한 관측값이 어떤 요인의 어떤 수준인가를 보여 주어야 한다. 위치지정은 관심이 되는 관측값이 전체 데이타 중에서 어떤 위치에 있는가를 지정해 주는 것이다. 분산분석의 경우 관심이 있는 요인의 수준을 선정하면 여기에 해당하는 관측값들이 전체 데이타 중에서 어떤 곳에 위치하고 있는 가를 보여주어야 한다. 여기서 O_{ij}를 요인 i의 수준 j에 속하는 관측값(들)이라고 하면 요인의 수준을 선정하는 기능은 다음 두 가지를 모두 포함하여야 한다.

(O_{ij} 또는 $O_{ij'}$)
(O_{ij} 그리고 $O_{ij'}$)

이동과 삭제: 하나의 관측 점을 마우스로 선택하고 이를 다른 위치로 변화시키므로 서 이 관측점의 값을 다른 값으로 대치시키는 방법이다. 이를 이용하면 하나의 관측값을 다른 값으로 변화시킬 때 분산분석에 미치는 영향을 분석할 수 있다. 또한 분산분석에서 하나의 관측값을 해당 셀의 평균값으로 대체하면 이 관측값을 삭제하는 효과를 가져오므로 이동 방법을 적용하므로써 관측값의 삭제가 분산분석에 미치는 영향을 조사할 수 있다.

연결: 연결은 선택된 관측값(들)을 다른 플롯(들)에 연결하는 기능이다. 분산분석의 경우 전체 데이타를 표현하는 플롯에서 어떤 관측값을 택했을 때 이 관측값을 각 요인의 효과 플롯이나 교호작용 플롯, 그리고 잔차 플롯 등에 연결시켜 주거나 거꾸로 잔차 플롯에서 어떤 점을 선택하면 이것이 다른 플롯 등에 연결시켜 주어야한다. 각 플롯들에 대해서는 다음 절에서 설명한다.

플롯형식의 변화: 이원 배치의 예를 들면 A 요인을 중심으로 하여 나타낸 교호작용 플롯을 B 요인을 중심으로 하여 다시 표현하는 기능이다.

모형변화: 유의하지 않은 인자를 모형에서 제거하고 새로운 모형을 만드는 기능이다.

3. FEDF

데이타를 플롯으로 표현하는 기본적인 방법에는 히스토그램과 상자플롯 등이 있다. 이들 그림을 이용하면 데이타의 분포 형태를 직관적으로 알 수 있다. 그러나 히스토그램은 막대의 수를 변화시키거나 출발점의 선택에 따라 플롯의 형태가 많이 변하기 때문에 이를 이용할 때는 세심한 주의를 기울여야 한다. 이에 반해 상자플롯은 그리는 사람에 관계없이 동일한 모양이 된다. 다만 상자플롯은 3개의 통계량(중앙값, 상-하 사분위수)에 기본을 두고 그린 것이어서 이보다 더 자세한 정보가 필요한 경우 제한적이며 상자플롯을 그리기 위해서는 최소 5개 이상의 관측값이 있어야 의미가 있다. 분산분석의 경우 각 수준에서 반응 값이 많지 않을 수가 있으므로(예를 들어 5개 이하) 상자플롯은 제한적이다. 또한 동적그래픽스를 구현하기 위해서는 각각의 관측값을 조정할 필요가 있으므로 히스토그램과 상자플롯 등은 이용에 한계가 있다. 이를 보완하는 방법으로 누적분포함수(empirical distribution function - EDF)를 생각할 수 있다. 누적분포함수는 단조 증가함수이며 하한과 상한이 있고 그리는 사람에 따라 변하지 않는다.

EDF를 중앙값 까지만 그리고 중앙값 이후부터는 y=1/2에 대칭으로 꺽는 경우를 생각해 보자. 이를 꺽은누적분포함수 (flipped EDF - FEDF)라고 부르기로 한다. FEDF의 공식은 다음과 같다.

$$FEDF = \begin{matrix} EDF, & x \leq \text{중앙값} \\ 1-EDF, & x > \text{중앙값} \end{matrix}$$

여기서 EDF는 Hoaglin 등(1991, HMT 187쪽)에서 이용하고있는 다음의 공식을 이용한다.

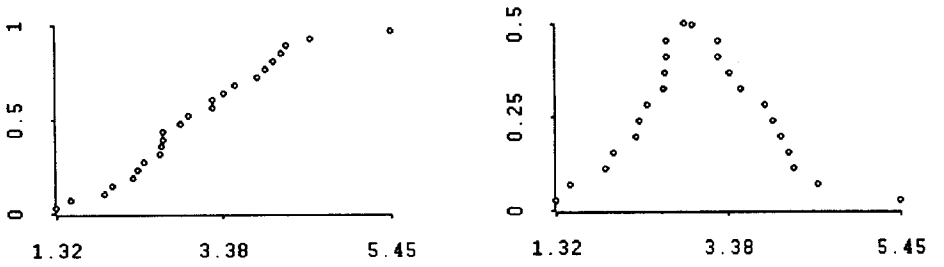
$$EDF = \frac{(i-1/3)}{(n+1/3)}, i = 1,...,n$$

예 1 노트북 컴퓨터의 밧데리 수명이 CPU타입, DISPLAY 타입에 따라 어떤 영향을 받는가를 알아보기 위해 1993년 10월호 BYTE지에 게재된 62개의 노트북 컴퓨터에 대한 성능평가자료(ROLL CALL OF PORTABLES TESTED)중 24개를 택해 조사한 것이 다음과 같다.

1.32 1.50 1.92 2.02 2.28 2.33 2.42 2.62 2.63 2.65 2.65 2.87
2.97 3.27 3.27 3.40 3.53 3.82 3.92 4.02 4.12 4.17 4.47 5.45

이 데이타를 EDF와 FEDF로 표현한 것이 <그림 1>에 나타나 있다. FEDF는 다음과 같은 장점을 갖고 있다.

그리기 쉽다.
특이값의 판단이 쉽다.
기초 통계량을 쉽게 구할 수 있다.
분포가 대칭인가를 곧 판단할 수 있다.



(a) (b)

<그림 1> notebok 데이타의 EDF(a)와 FEDF(b). 여기서 Q_p는 p-위수를 나타낸다.

각 관측값의 행동을 파악할 수 있다.
동점 관측값에도 유효하다.
표본크기가 작아도 (3개 이상) 유효하다.

FEDF를 이용하면 우리가 관심이 있는 특정한 관측값이 어느 곳에 위치한가를 바로 찾아낼 수 있으며 이 관측값이 전체 데이타에서 어떤 위치에 있는가를 곧 알 수 있다. 또한 Y-축의 .25에서 수평한 선을 긋고 FEDF가 이와 맞나는 점에서 수선을 그어 X-축과 맞나는 두 점을 찾으면 이 것이 상사분위 수 및 하사분위 수가 된다. 이런 방법에 의해 기초 통계량을 쉽게 구할 수 있다. 따라서 이를 이용하면 상자플롯에서 제공하는 정보를 곧 알 수 있다. 또 동점(tie)이 발생하더라도 이를 순서대로 배열하고 각 관측값에서 1/(n+1/3) 씩 Y-축의 높이를 증가(또는 감소)시키면 각 관측값들이 겹치지 않게 나타난다. 다만 FEDF는 밀도함수와 관계가 없으며 밀도함수에 익숙해 있는 관점에서 이를 비교할 경우 혼동의 우려가 있을 수 있다.

4. 분산분석에 적용될 수 있는 여러 가지 플롯

분산분석의 경우에는 회귀분석과 달리 분석과정을 그래픽스에 의해 탐색할 수 있는 범위가 제한적이다. 최근에 탐색적 통계분석 방법이 개발되면서 그래픽스 방법에 의해 분산분석형 데이타를 탐색하고자 하는 노력이 많아졌다(5).

분산분석에서 많이 이용되고 있는 그래픽스 표현방법은 상자플롯이다. 상자플롯은 하나의 요인에 대해서만 유효하다. 이에 대해 HMT에서 소개되고 있는 side-by-side 플롯은 여러 개의 요인이 있을 때 이를 동시에 표현해 줄 수 있는 것이 특징이다. 교호작용 플롯은 두 요인에 대한 교호작용을 판단할 때 유용하다. S-Plus 등의 통계 팩키지에서는 진단플롯을 위해 정규확률지를 이용하기도 한다. 그러나 정규확률지를 이용하려면 표본크기가 작지 않아야 의미가 있으

며 분산분석의 경우 각 수준에서 반응값의 수가 작은 경우가 많다.

교호작용 플롯은 원래 각 수준의 평균값들을 연결하도록 제안되었다. 그러나 이를 동적그래픽스에서 구현하려면 각 수준에서 평균값과 더불어 모든 관측값들을 표현하는 것이 더 효율적이다. 노트북 데이타를 통해 여기서 이용하는 플롯들을 설명한다.

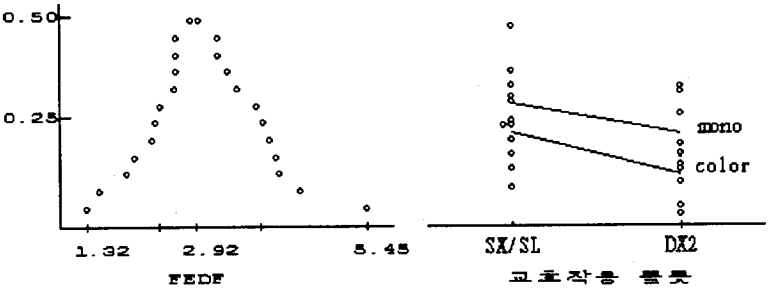
예 2 노트북 데이타를 CPU 타입과 디스플레이 타입의 두 요인으로 구분하여 각 요인의 효과와 교호작용의 효과, 그리고 잔차를 구하면 표1과 같다. 편의를 위해 A를 CPU 타입, B를 디스플레이 타입이라 하면 A1, A2를 각 각 SX/SL 와 DX2, B1과 B2를 각 각 mono와 color라고 하자. 또 (A1, B1)은 셀 (A1, B1)을 나타낸다. (이 표의 형식은 HMT의 101 쪽을 따른다)

밧데리 수명을 FEDF와 교호작용 플롯으로 나타낸 것이 <그림 2>에 나타나 있다. FEDF 플롯의 X-축에는 5개의 티크마크(tic mark)가 있다. 이들은 최소값, 하사분위수, 중앙값, 상사분위수, 최대값을 나타낸다. 이들을 "5 가 통계량"이라고 부르기로 한다. 동적그래픽스의 구현을 위해서 각 관측값의 위치를 알아야 하기 때문에 각 수준에서 각 관측값들을 선 플롯(line 플롯)에 의해 나타냈다.

교호작용 플롯에는 Y-축의 값을 명시하지 않았다. 그러나 본 논문에서는 교호작용 플롯이 항상 FEDF와 함께 나타나고 FEDF에는 5가 통계량이 X축에 나타나기 때문에 Y-축의 스케일을 명시하지 않아도 FEDF를 참고하면 교호작용 플롯의 스케일을 알 수 있다.

<표1> (예 2)의 각 요인에 대한 효과

		B1	B2
	3.07	0.39	-0.39
A1	0.40	-0.07	0.07
A2	-0.40	0.07	-0.07
A1		-0.52 0.38 -1.14 -0.52 0.13 1.67	1.32 -0.82 -0.18 0.25 0.67 -1.23
A2		1.00 0.90 -1.10 -0.70 0.41 -0.50	-0.71 0.44 0.66 -0.89 0.08 0.43



<그림 2> 노트북 데이타의 FEDF와 교호작용 플롯. FEDF의 x-축에 있는 5개의 티크마크는 5가 통계량에 해당된다.

5. 이원배치 모형의 분석을 위한 동적그래픽스의 구현

이원배치 모형에서 데이타와 모형을 변화시킬 때 분석 결과가 어떻게 변화하는가를 화면에서 즉시 관찰하기 위해 화면의 구성을 두개로 나누었다.

첫번 째 창(window)은 이원배치 데이타 자체를 표현하는 데 중점을 두었으며 두 번째 창에는 분석 결과를 표현하는데 중점을 두었다. 이를 위해 첫번 째 창에는 데이타 자체에 대한 FEDF와 교호작용 플롯을 배치하였다. 이를 "데이타창"으로 부르기로 한다. 또 여기에는 각 플롯들을 조정하는 5개의 메뉴와 함께 데이타나 모형의 변화에 따라 변화하는 F-비의 변화가 주어져 있다. 여기서 F-비의 변화를 간단히 줄여서 F-변화라고 한다. F-비는 F-A=MSA/MSE, F-B=MSB/MSE, F-AB=MSAB/MSE의 3 가지를 생각할 수 있다.

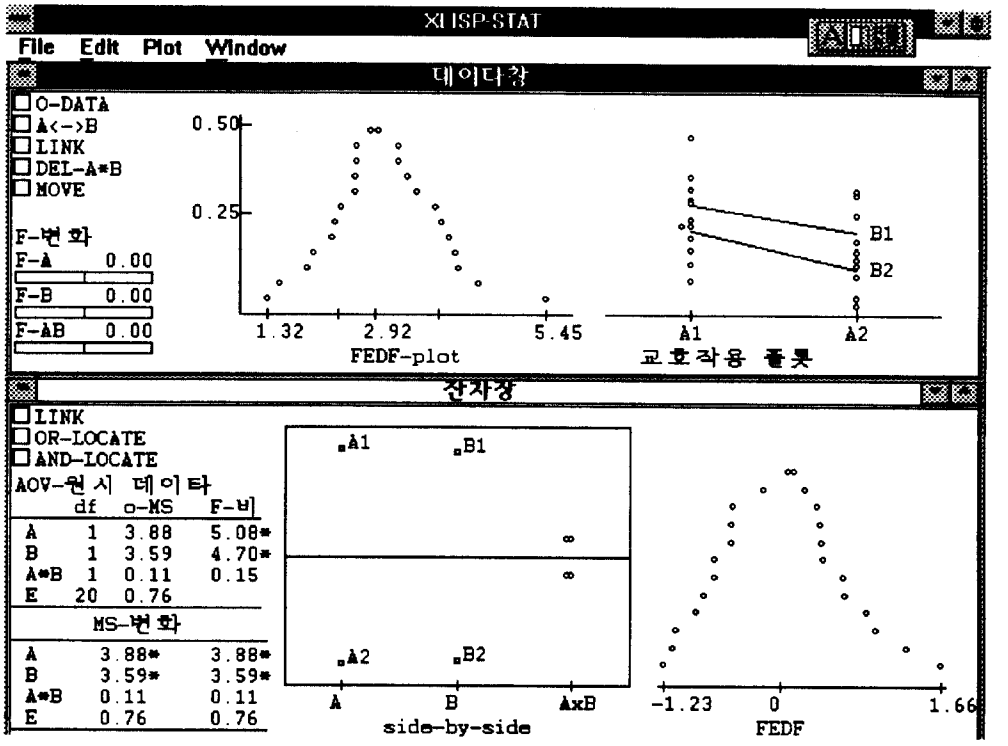
두번 째 창에는 side-by-side 플롯과 잔차 FEDF를 배치하였다. 이 창에는 플롯을 조정하는 3개의 메뉴와 함께 원시 데이타에 대한 분산분석표가 주어져 있으며 데이타나 모형을 변화시킬 때 이 변화에 따라 변화하는 MS-변화량(바로 전 데이타에 대응하는 MS와 데이타나 모형이 변하므로서 나타나는 MS)이 주어져 있다. 이 창을 "잔차창"으로 부르기로 한다.

처음 두개의 창이 그려질 때에는 원시 데이타를 이용하며 이 경우 데이타나 모형의 변화가 없으므로 데이타창에서 보여 주는 F-변화가 없고, 잔차창의 분산분석표와 MS변화량에 나타난 두 개의 MS는 같다(그림 3).

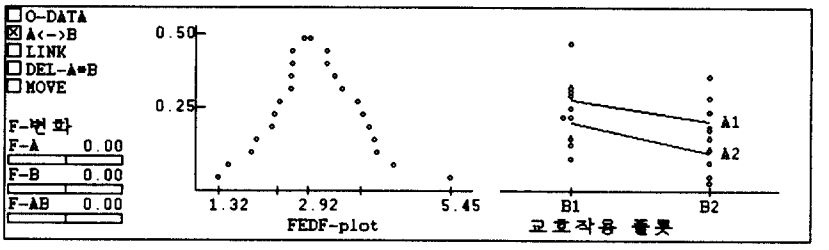
각 창의 메뉴 작동 원리는 먼저 메뉴를 클릭하면 이 후 이루어지는 모든 마우스의 움직임이 이 메뉴의 환경 하에서 이루어지게 되어있다. 예를 들어 어떤 점을 다른 위치로 옮기고자 하면 먼저 MOVE 메뉴를 지정하고 해당 점을 마우스로 끌어서 옮기고자 하는 위치에 놓으면 된다. 이 상태에서 또 다른 점의 움직임도 처리할 수 있다. 각 메뉴에 대해 자세히 설명하기로 한다.

데이타창의 메뉴

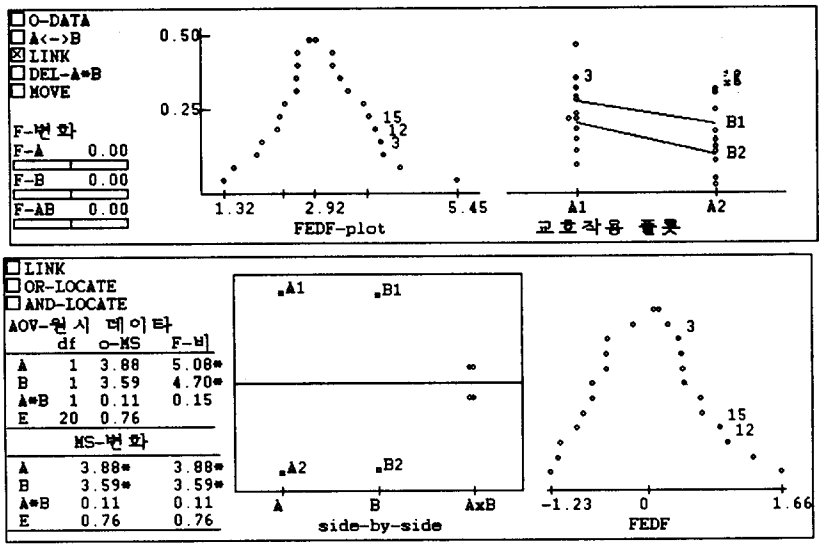
O-DATA: 이 메뉴를 택하면 모든 플롯과 분석절차에 원시 데이타를 적용한다.



<그림 3> 640 x 480 해상도를 갖는 화면에서 데이타 창과 잔차창을 동시에 띄우면 화면이 꽉 찬다. 데이타창에는 이원 배치 데이타에 대한 FEDF와 교호작용 플롯, 그리고 데이타와 도형을 조정하는 5개의 메뉴가 있고 데이타나 도형의 변화에 따라 변화하는 F-변화량이 나타나있다. 잔차창에는 이원 배치 데이타에 대한 side-by-side 플롯과 잔차 FEDF, 그리고 각 플롯들을 연결시키는 3개의 메뉴가 있고, 원시 데이타에 대한 분산분석표와 데이타나 모형을 변화시키므로서 변화하는 MS통계량이 주어져 있다.



<그림 4> A<->B 메뉴를 택하면 교호작용 플롯에서 x-축에 표시된 요인이 바뀌면서 플롯이 다시 그려진다.



<그림 5> LINK메뉴를 택하고 데이타창의 FEDF에서 세 점을 택한 결과 이 점들이 다른 플롯의 어떤 점에 위치해 있는 가를 보여주고 있다. 또 side-by-side 플롯을 보면 이 3 점들은 요인 B에서는 B1 수준에 속해있는 것을 알 수 있고 A 요인의 경우 A1 수준과 A2 수준에 걸쳐 속해 있는 것을 알 수 있다. 그러나 교호작용 플롯을 참고하면 3번 관측값은 A1 수준에, 12번과 15번 관측값은 A2수준에 속하는 것을 알 수 있다. (교호작용 플롯에 12, 15 숫자가 겹쳐서 나타났다)

A<->B: 교호작용 플롯에서 x-축에 나타나 있는 요인을 변화시킨다. 예를 들어 <그림 3>에는 x-축에 A요인을 기준으로 하여 교호작용 플롯이 나타나 있으나 A<->B 메뉴를 택하므로서 B요인을 기준으로 하여 <그림 4>와 같이 교호효과 플롯이 다시 그려진다.

LINK: 하나의 플롯에 나타나 있는 점(하나 또는 몇 개의 점들의 집합)이 다른 플롯에서는 어떤 위치에 나타나 있는가를 보여 준다. 예를 들어 FEDF에서 3개의 점을 택한 결과 이 점들의 위치가 다른 플롯에는 어떻게 나타나는가를 살펴보자. <그림 5>에 의하면 이 점들은 B 요인의 경우 B1 수준에 해당되며 (CPU는 DX2 타입이고 디스플레이는 color) A 요인의 경우 모든 수준에 걸쳐있는 것을 알 수 있다. 그러나 교호작용 플롯을 참고하면 3개의 점 중에 3번으로 표시되어 있는 점(3번째 관측값)만 A1 수준에 속하고 나머지 두개의 점들(12번과 15번)은 A2 수준에 속하는 것을 알 수 있다. 또한 잔차 FEDF를 참고하면 이원 분산분석 모형에 적합시킨 결과

이 3 점들의 잔차의 상대 위치를 알 수 있다. 물론 여기서 LINK할 점들은 교호작용 플롯에서 택해도 된다. 따라서 LINK 메뉴를 이용하면 2 절에서 설명한 라벨링 효과도 알아볼 수 있다.

DEL-A*B: 전통적인 분산분석에서는 교호효과가 유의하지 않으면 교호효과의 제곱합을 오차 제곱합에 합쳐서 새로운 모형을 세운다. 이 메뉴를 택하면 교호효과가 없는 모형을 만들어 모

든 과정을 다시 수행하게 된다. <그림 5>의 경우 잔차창에 나타나 있는 분산분석표를 참조하면 교호작용이 유의하지 않은 것으로 나타났다(5% 수준에서 유의하면 F-비옆에 *를, 1% 에서 유의하면 **를 표시하였다). 따라서 교호작용을 오차항에 합치는 것이 타당하므로 DEL-A*B 메뉴를 택한다. 이 메뉴를 택하면 데이타창과 잔차창은 <그림 6>과 같이 변한다. 데이타창의 FEDF와 교호작용 플롯은 변화가 일어나지 않지만교호작용의 MSE가 오차의 MSE와 합쳐지기 때문에 F-변화가 일어난다. 이 변화가 데이타창의 F-변화에 나타나 있다. 여기서 F-변화가 음으로 나타나면 막대의 중앙에서 좌측으로 변화량 만큼 검정 띠가 그려지고 양의 방향으로 변화가 있으면 가운데에서 우측으로 변화량 만큼 검정 띠가 그려진다. 변화량의 최대는 3, 최소는 -3 으로 하였다.(분자의 자유도 3, 분모의 자유도가 5인 경우 F.8 = 2.25, F.95 = 5.41로서 3.16의 변화가 나타난다) 이 데이타의 경우 교호작용을 제거한 후의 F-변화를 살펴보면 A요인의 경우 0.21이 증가하였고 B요인의 경우 0.20이 증가하였다. A*B의 경우 0.15가 감소한 것은 교호작용을 제거하기 전 F-비가 0.15였기 때문이다.

잔차창에서는 교호작용을 제거하므로써 많은 변화가 나타나는 것을 알 수 있다. 우선 side-by-side 플롯에서 A*B에 해당하는 모든 항이 모두 0으로 변한다. 또한 잔차 FEDF 형태가 약간 변한 것을 알 수 있다. <그림 5>를 참고하면 교호작용을 포함한 경우 잔차가 -1.23에서 1.66까지 분포되어 있지만 <그림 6>을 참고하면 교호작용을 제거한 경우 -1.20에서 1.60까지 분포되어 있다. MS-변화를 보면 교호작용 MSAB는 0.11에서 0으로, MSE는 0.76에서 0.73으로 변화한 것을 알 수 있다. 이 데이타의 경우 교호작용을 제거하여도 요인들의 유의성에는 차이가 없는 것을 알 수 있다.

MOVE: 하나의 관측값을 현재의 위치에서 다른 위치로 움직이므로써 이 변화가 모형과 분석결과에 어떤 영향을 미치는가를 알고자 할 때 이용한다. 예를 들어 24개의 노트북 중에서 가장 밧데리 수명이 긴 것은 Zenith SL/25 mono 디스플레이로서 5.45시간이다. LINK 메뉴를 이용하여 이 관측값이 속한 셀을 조사하면 (A1, B1) - SX/SL CPU와 mono 디스플레이 - 인 것을 알 수 있다. 만약 이 값을 셀 (A1, B1)의 평균으로 바꾼다면 결과가 어떻게 변화할 것인가? (이렇게 변화시키는 것은 해당 점을 삭제하는 것과 같은 결과를 가져온다) 이를 위해 MOVE 메뉴를 택하고 이 점을 셀 (A1,B1)의 평균점의 위치에 옮기면 된다. (A1,B1)의 평균의 위치는 교호작용 플롯의 A1 수준에서 B1 수준의 평균 선이 맞나는 점이다. 이렇게 작동한 결과가 <그림 7>에 나타나 있다. 움직인 점은 모든 플롯에서 검정 색 점으로 표시(highlight, 또는 이를 간단히 hilite로 표기하기도 한다)되어 있다.

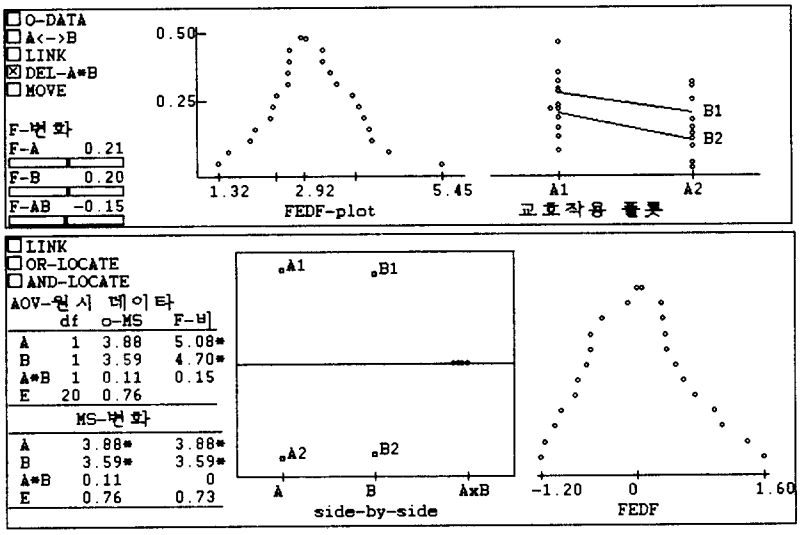
<그림 7>의 데이타창을 참고하면 MOVE 메뉴 밑에 "5.45 -> 3.62"가 있다. 이것은 5.45에서 3.62로 한 점의 MOVE 가 일어났다는 뜻이다. (셀 (A1, B1)속한 관측값의 정확한 평균은 3.61이지만 여기서는 마우스로 이 위치를 찾아가기 때문에 약간의 오차가 발생할 수 있다). 교호작용 플롯에서 점을 움직이고 나면 해당 셀의 평균이 변하기 때문에 움직인 점은 (A1,B1)의 평균점과 일치하지 않는다. 데이타창의 F-변화를 참고하면 A, B 두 요인은 각 각 0.83과 0.84 만큼 F-비가 줄어든 것을 알 수 있다. 잔차창의 MS 변화를 참고하면 앞에서 지정한 한 점의 변화로 MSA 는 3.88* 에서 2.55로 MSB는 3.59*에서 2.31로, 그리고 MSE는 0.76에서 0.60으로 변화한 것을 알 수 있다. 즉, 이 점을 움직이기 전에는 A, B 두 요인 모두 5% 수준에서 유의하던 것이 이 점을 움직이고 난 후에는 두 요인 모두 유의하지 않은 것으로 나타났다. 또한 <그림 7>에 나타난 잔차 FEDF와 <그림 6>의 원시 데이타 잔차 FEDF (교호작용을 제거한 모형)을 비

교하면 원래에는 잔차가 -1.20 에서 1.60까지 분포되어 있던 것이 점 하나를 움직이므로 서 -1.23에서 1.32의 범위에서 분포되어 있는 것을 알 수 있다.

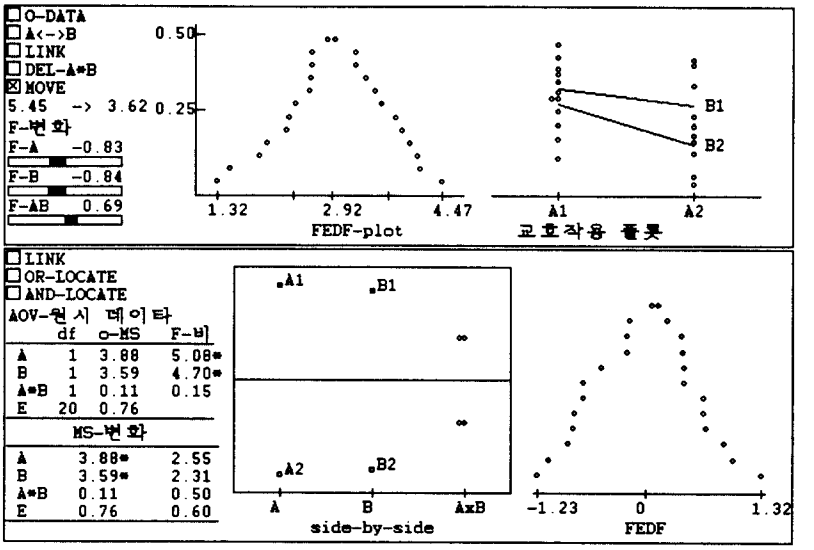
잔차창에는 side-by-side 플롯과 잔차에 대한 FEDF가 나타나 있다. 이 창에 나타난 메뉴는 다음과 같다.

LINK: 이 메뉴의 기능은 데이타창의 LINK 메뉴와 같다. 예를 들어 잔차창에 주어진 잔차 FEDF 플롯에서 절대값이 가장 큰 잔차는 다른 플롯에서 어떤 점에 해당되는가를 알고자 할 때, 또는 side-by-side 플롯에서 교호작용 효과가 음의 값으로 나타난 점들은 다른 플롯에서 어떤 점으로 나타나는가를 알고자 할 때 이를 이용하면 편리하다.

OR-LOCATE/AND-LOCATE: A요인이나 B요인의 각 수준에 있는 점을 잔차창이나 데이타창의 FEDF 또는 교호작용 플롯과 연결시켜 준다. 예를 들어 셀 (A1, B1)에 있는 관측값들을 잔차 FEDF나 교호작용 플롯과 연결시키려면 AND-LOCATE 메뉴를 택하고 A1과 B1에서 마우스를 클릭한다. 만약 AND-LOCATE 메뉴를 택하고 셀 A1, A2에서 클릭을 하면 여기에 해당하는 관측값이 없으므로 아무 반응이 없다. 또 A1이나 B1수준에 있는 모든 관측값들을 다른 플롯에 연결하려면 OR-LOCATE 메뉴를 택하고 A1과 B1에서 마우스를 클릭한다.



<그림 6> DEL-A*B메뉴를 택하면 분산분석 모형에서 교호작용이 제거되고 A*B 요인의 SS가 E요인의 SS로 합쳐지면서 F-변화가 일어난다. 잔차창을 참고하면 MSAB가 0으로 변화하고 MSE는 .76에서 .73으로 줄었다. side-by-side 플롯에서 A*B 요인에 해당되는 항은 모두 0으로 처리되고 잔차 FEDF도 교호작용이 포함되어있는 모형의 잔차 FEDF (그림 5 참조)와 다른 것을 알 수 있다.



<그림 7> MOVE 메뉴를 택하고 교호작용 플롯에서 가장 큰점(이것은 FEDF에서 가장 큰점과 같고 이 값은 5.45이다)을 이 점이 속한 셀 (A1, B1)의 평균위치로 옮기면 A 요인과 B 요인의 F-비가 0.83과 0.84만큼 감소하고, AB 요인의 경우 0.69만큼 증가하며 5% 수준에서 유의하던 A, B 요인이 모두 유의하지 않게 변화한 것을 알 수 있다.

6. 마치는 말

여기서 개발된 소프트웨어는 PC에서 수행되게 되었으며 이것은 640x480 해상도를 갖는 화면을 기준으로 설계되었다. 여기서 사용한 개발도구는 XLISP-STAT(7)이다. XLISP-STAT은 한 가지 폰트 만을 제공하기 때문에 각 창의 글자 크기를 작게 조정할 수가 없으며 따라서 이 화면에서는 본 논문에서 다루고 있는 두개의 창(데이타 창과 잔차 창)이 최대 한계이다. 그러나 화면의 해상도를 더 늘린다면 (예를 들어 1024x768) 각 창에 더 많은 정보를 실을 수도 있고 새로운 창을 추가 할 수도 있다. 다만 동적그래픽스 환경은 여러개의 창이 동시에 한 화면에 나타나야 하며 이들이 같이 움직여야 하기 때문에 화면에 정보가 많으면 그만큼 속도가 더 빠른 컴퓨터가 필요하게 된다.

각 셀의 관측값이 불균형인 경우 HMT에서 제시한 side-by-side 플롯의 적용이 어렵고 유일한 분산분석이 존재하지 않기 때문에 여기서는 각 셀이 균형인 경우로 제한하였다. 그러나 현실적인 문제에서는 불균형인 실험이 많이 나타나므로 이를 보완하는 작업이 필요하다. 이원 분산분석의 제한에서 벗어나 일반적인 경우에도 적용될 수있는 시스템에 대해서는 현재 연구가 진행 중이다.

여기서 사용한 개발도구인 XLISP-STAT은 일단 숙달이되면 프로그램하기가 편리한 강력한

도구로 이용할 수있다. 그러나 사용자에게는 효율적인 소프트웨어가 기계에게는 매우 부담을 주는 일반적인 원리가 여기에도 적용된다. 더우기 동적 그래픽스는 데이타나 모형에 변화를 주면 이 효과가 그 즉시 화면에 나타나야 하므로 매우 강력한 컴퓨터가 필요하다. 다행히 컴퓨터 기술이 급격히 발달하여 속도가 빠른 하드웨어가 값싸게 공급되기 때문에 이러한 문제가 어느 정도 해결되고있다. 이 논문을 위해 개발된 소프트웨어는 전체 크기가 50K byte 로서 약 40쪽이다. 관심있는 독자에게는 무상 배포하고자 한다.

참고문헌

[1] 김성수 (1992). *Regression Diagnostics Using Dynamic Graphical Methods,* 박사학위 논문, 서울대학교

[2] Becker, R. A., Cleveland, W. S. and Wilks, A. R. (1988). Dynamic Graphics for Data Amnalysis, 1-50, *Dynamic Graphics for Statistics,* edited by Cleveland, W. S. and M. E. McGill, Belmont, CA: Wadsworth.

[3] BYTE (1993). ROLL CALL OF PORTABLES TESTED, PP. 192-193, Oct.

[4] Easton, G. S. (1994). A simple dynamic graphical diagnostic method for almost any model, *Journal of American Statistical Association,* Vol. 89, 201-207.

[5] Hoaglin, D. C., Mosteller, F. and John W. Tukey (1991). *Fundamentals of Exploratory Analysis of Variance,* Wiley.

[6] Nagel, M. and Ostermann, R. (1993). Dynamic Graphics for Discrete Data, *Computational Statistics,* Vol. 8, 197-205.

[7] Tierney, L. (1990). *LISP-STAT,* John Wiley & Sons.

Computer Graphics Approach to Two-way ANOVA

Huh, Moon Yul[3)]

Abstract

Computer graphics approach is a powerful tool when we are to explore the effects of the change of a part of the data, or the effects of the alteration of the characteristics of the statistical model currently employed. The paper describes the methods to implement dynamic graphics for the process of analysis of variance, and the methods to graphically represent ANOVA type data. The paper then describes a dynamic graphics software developed by the author for two-way ANOVA model.

3) Department of Statistics, Sung Kyun Kwan University, Seoul 110-745, KOREA.