응용통계연구 제10권 1호
1997년 3월 pp. 163-175

선형추세무관 블록계획법의 생성[1)]

박 동 권[2)] , 김 형 문[3)]

요 약

많은 산업체나 농업 현장에서 실험이 행해질 때 블록내 실험단위에서의 처리가 시간적 또는 공간적으로 제약을 받는 경우가 빈번히 발생하게 되는데 이러한 경우 처리는 확률화에 따르기보다는 조직적인 순서에 따라 행해지게 된다. 이 때 처리 효과가 블록내에서 나타날 수 있는 시간적 또는 공간적 추세에 독립적으로 추정되도록 순서가 설계된 실험계획을 추세무관 블록계획이라 부른다. 본 논문에서는 먼저 추세무관 블록계획의 성질을 살펴 그 필요성에 관해 약술하고 다음으로 선형추세무관 블록계획의 생성을 위한 두 알고리즘을 소개한다. 하나는 Bradley와 Odeh(1988)에 의해 고안된 알고리즘을 보완하였고, 다음은 존재하는 모든 가능한 계획을 발생시키는 알고리즘을 제시하였다.

1. 서 론

실험에는 결과에 영향을 미칠 수 있는 모든 요인을 고려해야 하지만 현실적으로는 여러 가지 원인으로 불가능한 경우가 대부분이어서 부득이 실험에 고려되지 않은 요인들이 있게 마련이다. 이 경우 실험외적 요인의 영향을 상쇄시켜 동등화가 이루어지도록 한 원리가 확률화(randomization)이다. 이러한 확률화는 실험계획에서 가장 중요한 원리로 인정받고 있으나 때로는 조직적으로 실험을 할 필요가 있다. 예로서 한 공장에서 실험 할 경우 시간의 흐름에 따른 장비의 특성 변화 혹은 실험자의 피로도에 의해 반응치가 영향을 받을 수 있다. 이와 같이 많은 산업체나 농업 현장에서는 실험단위에 시간적 또는 공간적인 순서에 따라 처리가 행해지는 경우가 빈번히 발생하게 된다. 이러한 경우 처리는 확률화에 따르기보다는 조직적인(systematic) 순서에 의해 행해지게 된다. 이러한 조직적인 순서 중 처리 효과가 실험에서 나타날 수 있는 시간적 또는 공간적 추세에 독립적으로 추정되도록 순서가 설계된 실험계획을 추세무관(trend-free) 실험계획이라 부른다.

추세무관 실험계획법은 크게 요인실험에서 실험순서를 정하는 경우와 단일 요인이 블록내에서 추세에 무관하도록 배치되는 추세무관 블록계획(trend-free block designs; TFB)의 두 가지 경우로 나누어져 연구가 이루어져 왔다. 물론 두 경우가 명백히 분리되지는 않지만 편의상 둘로 구분하여 여기서는 후자의 경우만을 다루도록 하자. 전자의 경우 예로서 요인의 수가 6인

1) 이 논문은 1996년도 연세대학교 학술연구비 지원에 의하여 이루어진 것임
2) (220-710) 강원도 원주시 흥업면 매지리 234, 연세대학교 통계학과 부교수.
3) 서울시 연세대학교 상경대학 응용통계학과 석사과정 졸업.

2^{6-1}-부분요인실험의 경우 '어떻게 선택된 32개의 처리조합이 추세 무관하도록 실험 순서가 정해질 수 있는가?' 등을 연구하게 된다. Daniel과 Wilcoxon(1966)이 시간이나 거리에서 추세가 있을 때 요인 효과를 보다 더 잘 추정하기 위하여 요인 처리조합을 실험단위들에 할당하는 추세무관 요인실험에 관해 다루기 시작하였다. 요인실험의 경우에 대한 자세한 내용은 Coster와 Cheng(1988)의 논문을 참조하기 바란다.

우리의 관심이 되는 후자의 경우를 살펴보면 최초로 Cox(1951, 1952)에 의해 제시된 후 Box(1952)는 양적 요인의 수준을 선택하는 문제를 다루었고, Hill(1960)은 추세가 있을 때 질적, 양적 요인의 효과를 연구하기 위하여 Cox와 Box의 계획법을 조합하였다. 그 후 Yeh와 Bradley(1983)는 TFB의 존재에 관한 추측(conjecture)을 제시하였고, Chai와 Majumdar(1993)는 이러한 추측이 성립하는 필요충분조건을 제시함으로서 TFB의 설계에 기초를 제공하였다. 본 연구에서는 먼저 추세무관 블록계획법의 개념과 성질들에 대해서 살펴본 후 선형추세만이 고려된 선형추세무관 블록계획법(linear TFB ;LTFB)을 발생시키는 알고리즘을 고안한다. 본 연구에서는 두 가지 알고리즘을 제안하였다. 하나는 Bradley와 Odeh(1988)에 의해 제시된 알고리즘을 보완한 것이며 다른 하나는 존재하는 모든 가능한 LTFB를 생성시키는 것이다.

2. 추세무관 블록계획법의 여러 가지 성질들

본 연구에서는 블록내에 있는 처리들을 비교하기 위하여 한 블록내의 처리의 수가 일정(k)하고, 어떠한 처리도 동일 블록에서 두 번 이상 실험되지 않는 이분(binary)계획이고, v개의 처리의 반복수가 같고(r), 연결된(connected) 계획법으로 국한한다. 그리고 각각의 블록내에 있는 실험구들에 공통의 다항추세가 존재한다고 가정하여 추세들은 p개의 직교다항식(orthogonal polynomial) ϕ_α (α=1,..., p)로 나타낼 수 있다. 따라서 적절한 고정효과 모형(fixed-effects model)은 블록계획법의 모형에 추세항을 포함시킴으로써 아래와 같이 표현될 수 있다.

$$y_{jt} = \mu + \sum_{i=1}^{v} \delta_{jt}^{i}\tau_i + \beta_j + \sum_{\alpha=1}^{p} \theta_\alpha \phi_\alpha(t) + \varepsilon_{jt} \qquad (2.1)$$

위에서 y_{jt}는 j (=1,..., b) 번째 블록의 t (=1,..., k)번째 실험구에서의 관찰값이고 μ, τ_i, β_j, θ_α는 각각 평균, 처리, 블록, 추세모수이다. 오차항 ε_{jt}은 평균 0, 분산 σ^2의 정규분포를 따른다고 가정한다. 직교다항계수 $\phi_\alpha(t)$는 식

$$\sum_{t=1}^{k} \phi_\alpha(t) = 0, \quad \sum_{t=1}^{k} \phi_\alpha(t)\phi_\beta(t) = \begin{cases} 1 & \text{만약 } \alpha=\beta \text{ 이면} \\ 0 & \text{만약 } \alpha\neq\beta \text{ 이면} \end{cases} \qquad (2.2)$$

을 만족하고,

$$\delta_{jt}^{i} = \begin{cases} 1 & \text{만약 처리 } i\text{가 실험구 } (j,t)\text{에서 실험될 경우} \\ 0 & \text{그밖의 경우} \end{cases}$$

를 나타낸다. p=1이면 LTFB가 된다.

위 (2.1) 모형을 행렬로 표시하면

$$\boldsymbol{y} = X\xi + \boldsymbol{\varepsilon} = X_\mu \mu + X_\tau \boldsymbol{\tau} + X_\beta \boldsymbol{\beta} + X_\theta \boldsymbol{\theta} + \boldsymbol{\varepsilon} \tag{2.3}$$

$$\boldsymbol{\tau}^t = (\tau_1, \cdots, \tau_v),\quad \boldsymbol{\beta}^t = (\beta_1, \cdots, \beta_b),\quad \boldsymbol{\theta}^t = (\theta_1, \cdots, \theta_p)$$
$$\xi^t = (\mu, \boldsymbol{\tau}^t, \boldsymbol{\beta}^t, \boldsymbol{\theta}^t),\quad X = (X_\mu, X_\tau, X_\beta, X_\theta),$$
$$X_\mu = \mathbf{1}_{bk},\quad X_\tau^t = (\Delta_1^t, \cdots, \Delta_b^t),\quad X_\beta = I_b \otimes \mathbf{1}_k,\quad X_\theta = \mathbf{1}_b \otimes \Phi$$

로서 다시 쓸 수 있는데 여기에서 $\otimes$는 Kronecker 곱하기이며 Δ_j는 δ_{jt}^i를 행 t의 i번째 원소로 가지는 $k \times v$행렬이고, Φ는 $\phi_\alpha(t)$를 t행, α열의 원소로 가지는 $k \times p$행렬이다. 정보행렬(information matrix)은 N을 빈도행렬(incidence matrix)이라 하면

$$C = rI_v - \frac{1}{k} NN^t - \frac{1}{b} \Delta_+^t \Phi \Phi^t \Delta_+ \tag{2.4}$$

로 구해진다. 여기서 $\Delta_+ = \sum_j \Delta_j$를 나타낸다.

위 (2.1) (혹은 (2.3)) 모형에서 처리제곱합과 블록제곱합이 추세 효과가 모형에 포함되지 않은 모형에서와 같은 제곱합을 가지면 처리 효과는 추세와 독립적으로 추정될 수 있다. 다시 말해서 $R(\boldsymbol{\tau} \mid \mu, \boldsymbol{\beta}, \boldsymbol{\theta})$을 모형 (2.1) 아래서 조정된 처리제곱합이라 하고, θ=0 일 때 조정된 처리제곱합을 $R(\boldsymbol{\tau} \mid \mu, \boldsymbol{\beta})$라 할 때 Bradley와 Yeh(1980)는

$$R(\boldsymbol{\tau} \mid \mu, \boldsymbol{\beta}, \boldsymbol{\theta}) = R(\boldsymbol{\tau} \mid \mu, \boldsymbol{\beta}) \tag{2.5}$$

을 만족하면 그 블록계획은 추세 무관하다고 정의하였다. 또한, 그들은 한 블록계획이 추세무관하기 위한 다음의 필요충분조건을 제시하였다.

정리 2.1 (Bradley와 Yeh, 1980) 한 블록계획법이 추세무관 블록계획법이 되기 위한 필요충분조건은 다음과 같다.

$$X_\tau^t X_\theta = \mathbf{0} \tag{2.6}$$

정리 2.1의 필요충분조건에서 $X_\tau^t = (\Delta_1^t, \cdots, \Delta_b^t)$이고 $X_\theta = \mathbf{1}_b \otimes \Phi$ 이며 $\Delta_+^t = \sum_{j=1}^b \Delta_j^t$ 이다. 따라서 조건 (2.6)은 Δ_j이 δ_{jt}^i를 행 t의 i번째 원소로 가지는 $k \times v$행렬이고 Φ이 $\phi_\alpha(t)$를 t행, α열의 원소로 가지는 $k \times p$행렬이므로

$$X_\tau^t X_\theta = \sum_{j=1}^b \Delta_j^t \Phi = \Delta_+^t \Phi = \sum_{j=1}^b \sum_{t=1}^k \delta_{jt}^i \phi_\alpha(t) = \mathbf{0} \tag{2.7}$$

와 같아짐을 알 수 있다. 정리 2.1의 필요충분조건이 $\boldsymbol{y}$의 분포에 의존하지 않으므로 정리 2.1은 확률효과모형(random effect model) 또는 혼합모형일 때도 성립한다. 또한 정리 2.1은 이분

계획법이 아닐 때에도 성립하고 블록 안에 있는 처리의 수가 다를 때에도 성립함을 Lin과 Dean(1991)이 증명한 바 있다. 모형 (2.1)에서 주어진 블록계획법이 추세무관 블록계획법이 되면 분산분석표는 <표 1>과 같이 간단하게 구해진다. <표 1>에서

$$B=X^t_\beta y,\quad W=X^t_\theta y,\quad G=X^t_\mu y,\quad T=X^t_\tau y,\quad C_0=rI-\frac{1}{k}NN^t,$$

$$Q=T-\frac{1}{k}NB-\frac{1}{b}X^t_\tau X_\theta W,\quad F=\frac{1}{k}\,\xi^t X^t\left(X_\beta X^t_\beta-\frac{1}{b}X_\mu X^t_\mu\right)X\xi$$

, $d(A)$는 행렬 A의 차수(rank)를 나타내며 추세블록제곱합은 $\frac{W^tW}{b}$이 된다. SAS를 이용하여 선형추세무관 블록계획법의 자료를 분석하려면 실험구 위치를 공변수(covariate)로 이용하면 된다.

추세를 고려한 블록계획법의 정보행렬 (2.4)에서 TFB의 조건 (2.7)이 만족하면 추세가 고려되지 않은 블록계획법의 정보행렬로 줄어듦을 알 수 있다. 따라서, TFB의 여러 가지 성질들은 추세가 고려되지 않은 블록계획의 경우와 같이 유지됨을 알 수 있다. 예로서 LTFB의 경우 추세무관 완비 블록계획법과 추세무관 균형불완비블록계획법(BIBD)은 모든 연결된 계획법(v,b,r,k)내에서 최적(optimal)계획법이 된다.

<표 1> 추세무관 블록계획법의 분산분석표

변 동 요 인	자 유 도	제 곱 합	기대평균제곱합
블록(미조정)	$b-1$	$\left(B^tB-\frac{1}{b}G^2\right)/k$	$\sigma^2+F/(b-1)$
처리(조정)	$d(C_0)$	$Q^tC_0^-Q$	$\sigma^2+\tau^tC_0\tau/d(C_0)$
추세항 1	1	$\frac{W_1^{\,2}}{b}$	$\sigma^2+b\theta_1^2$
⋮	⋮	⋮	⋮
추세항 p	1	$\frac{W_p^{\,2}}{b}$	$\sigma^2+b\theta_p^2$
오차항	$bk-b-p-d(C_0)$	전변동 제곱합 − 나머지 모든 제곱합	σ^2
전변동	$bk-1$	$y^ty-\frac{G^2}{bk}$	

또한, 블록내에서 처리들을 확률화하는 공분산분석과 비교하여 추세무관 블록계획법이 처리대비의 분산 기준에서 보면 보다 효율적임도 쉽게 증명이 가능하다.

그러나, 한 블록계획법이 주어졌을 때 블록내에서 실험구의 위치를 바꿈으로서 항상 TFB의 설계가 가능하지는 않다. 따라서 어떤 조건하에서 가능한지가 연구의 대상이 되는데 이 문제의 답을 주기 위해 Yeh와 Bradley(1983)는 다음과 같은 추측(conjecture)을 하였다.

가정된 추세가 일차일 때 모수조합(v, b, r, k)이 주어진 한 연결된 블록계획법이 블록내에 있는 실험구 위치를 바꿈으로써 항상 LTFB가 되기 위한 필요충분조건은 '$r(k+1)/2$가 정수'이다.

위 추측에서 필요조건은 증명이 되었으나 충분조건은 늘 성립하지 않음이 Stufken(1988)에 의해 예로서 입증되었다. 이 문제의 해답은 Chai와 Majumdar(1993)에 의해 k가 짝수이거나 BIBD인 경우만 성립함이 증명되었다. 이러한 사실은 3장에서 알고리즘이 보완되는 데 이용된다.

3. 알고리즘

3.1 Bradley와 Odeh의 알고리즘의 개요

먼저 기존의 Bradley와 Odeh의 알고리즘의 중요 절차를 살펴보면 다음과 같다. LTFB는 한 주어진 블록계획법에서 처리들을 적절히 배치하여 모든 i=1,..., v에 대하여 조건 (2.7) $\sum_{j=1}^{b}\sum_{t=1}^{k}\delta_{jt}^{i}\phi_1(t)$=0 이 만족되는 경우 생성된다. 먼저 각각의 블록에 있는 처리들을 블록내의 실험구에 무작위로 배치한 후 실험구에 있는 처리들을 쌍으로 바꾸면서 위 조건이 만족되면 생성된 계획법을 출력한다. 이를 위해 Q를 아래와 같이 정의하자.

$$Q = \sum_{i=1}^{v}\left[\sum_{j=1}^{b}\sum_{t=1}^{k}\delta_{jt}^{i}\phi_1(t)\right]^2 \tag{2.8}$$

이 알고리즘은 목적함수 Q의 반복적인 최소화에 기초하고 있다. 조건 ' r(k+1)/2가 정수이다'가 만족되면 알고리즘은 쌍으로 처리들을 바꾸면서 Q가 0이 되면 한 LTFB를 출력하고, 만약 Q가 0이 되지 않는다면(추측이 충분조건은 아님을 기억하기 바람) 멈춤 규칙에 의해 쌍으로 처리들을 바꾸는 반복을 멈추면서 가장 작은 Q를 가지는 계획법을 출력한다.

처리시간의 단축과 수렴성을 얻기 위해 두개의 교환 체계가 필요하다. 첫 번째는 최초로 처리들을 실험구에 무작위로 배치한 계획법의 Q값을 Q_0 라고 하고 쌍으로 처리들을 바꾸면서 Q값이 여러개 나올 수 있다. 만약 $Q > Q_0$이면 바로 전의 계획법은 기각되고 Q_0가 계속 유지될 것이다. $Q < Q_0$일 때 Q=0이 되면 LTFB가 출력된다. $Q < Q_0$면서 Q=0 이 되지 않으면 Q를 Q_0로 다시 정의하고 쌍으로 처리들을 바꾸는 과정이 계속된다. 다음으로 $Q = Q_0$이면 두 번째 체계(system)가 요구된다. $Q = Q_0$일 때 한 블록내에서 쌍으로 바뀐 것만 다른 두개의 계획법 d_1과 d_2가 얻어지며 그 블록을 b_0라고 하자. 그런 다음 d_2의 b_0안에서 다시 쌍으로 바꾸면서 $Q < Q_0$이 되는가 검사한다. 위 부등식에서 <의 효과는 d_1과 d_2가 반복되는 무한루

프를 방지한다. 이 때 $Q \leq Q_0$의 값을 가지는 d_2는 첫 번째 체계로 복귀한다. 만약 모든 탐색 후 d_2가 $Q < Q_0$의 값을 갖지 않을 때는 d_1으로 복귀되어 d_2의 탐색방법과 동일하게 조사된다. 비록 d_1과 d_2만 조사되었지만 첫 번째와 두 번째 체계의 효과는 다른 블록내에서 쌍으로 바꾸는 방법과 함께 b_0안에서 모든 가능한 교환 방법을 고려하는 것이다. 실제로 Q_0가 감소될 때 $Q = Q_0$이 되는 많은 계획법이 있으며 d_1과 d_2가 전부 조사되기 전에 $Q < Q_0$ 이 되는 경우가 일어난다.

주어진 블록계획법에서 LTFB가 존재하지 않을 때 이 알고리즘은 수렴하지 않으며 멈춤 규칙에 의해 가장 작은 Q를 가지는 계획법이 출력된다.

3.2 수정된 알고리즘

Bradley와 Odeh의 알고리즘은 몇 가지 보완할 점을 지니고 있어 문제점을 제기한 후 이에 대한 해결방안을 제시하였다. 우선 처리들을 블록내에 있는 실험구에 배치할 때 무작위로 배치하기 때문에 같은 계획법이 반복해서 생성될 수 있고 또한 LTFB가 존재하는데도 꼭 LTFB만 찾아주지 않을 수 있다. 이러한 문제점을 해결하기 위해 우선은 처음 계획법과 그 후 발생되는 계획법에서 계획법 뒷부분의 처리의 위치가 같은 것의 개수가 일정 개수 이상이면 다시 무작위로 처리를 배치하는 무작위 점검(randomness check)을 행함으로서 생성된 계획의 반복수가 줄어들거나 또는 더 많은 LTFB를 생성하게 하였다. 무작위 점검을 행함으로서 대부분의 경우 LTFB를 생성하게 되나 LTFB의 존재의 필요충분조건을 만족하는 경우 완벽을 위하여 LTFB가 생성될 때까지 무작위 점검을 계속함으로서 생성되는 계획이 모두 LTFB가 되도록 하였다. 물론 추측이 성립하는 필요충분조건이 블록크기가 짝수일 때와 BIBD일 때에만 증명이 되어 있어서 보다 폭넓은 적용을 위하여 앞으로 이에 대한 보완이 필요하다고 하겠다.

효율적인 측면에서 본다면 잘못된 자료를 입력하면 잘못된 결과가 나올 수 있는데도 이런 경우에 대한 대책이 없어서 보완대책을 삽입하였다. 즉, 자료 화일에 대한 점검을 하였다. 불완비 블록계획법인 경우 자료화일에서 입력된 r과 k가 처음 배치된 계획법에서의 r과 k와 맞는지 검사하였다.

위의 결과를 토대로 [부록 1]의 프로그램은 Bradley와 Odeh의 프로그램에서 중요하게 달라진 부분들을 표시한 것이며 첫 번째는 자료화일에 대한 점검부분이며 두 번째 부분은 무작위 점검에 대한 것이며 마지막은 필요충분조건들을 삽입한 것이다.

[부록 2]에는 예로서 네 모수 조합 아래서 두 프로그램에 따라 생성된 블록계획에 대한 결과의 비교가 나타나 있다. 결과를 보면 수정된 알고리즘하에서 생성된 블록계획들이 반복된 계획의 수가 적고, 또한 더 많은 LTFB(Q=0)를 생성함을 볼 수 있어 개선되었음을 볼 수 있다. 소요시간 역시 단축되었다. 부록에서 나타난 블록계획들은 블록간의 순서가 바뀌어도 추세무관에는 영향을 미치지 않아 같은 블록계획으로 간주된다.

3.3 모든 가능한 선형추세무관 블록계획법의 생성

우리는 3.2절에서 LTFB들을 [부록 1]에 제시된 프로그램을 이용하여 발생시켜 보았다. 실제로 적용할 경우 우리는 각각의 모수조합(v, b, r, k)에 따라 존재하는 모든 가능한 LTFB를 안다면 선택의 폭이 넓어진다고 할 수 있다. 따라서 각각의 모수조합(v, b, r, k)에 따라 존재하는 모든 가능한 LTFB를 생성할 필요가 있다. 이러한 각각의 모수 조합에 대해서 존재하는 모든 LTFB를 생성시키는 알고리즘을 제시하면 다음과 같다.

먼저 자료화일에 대한 점검을 한다. 자료화일에서 입력된 r과 k가 처음 배치된 계획법에서의 r과 k와 맞는지 검사한다. 그런 후 k개의 처리들을 순서대로 한 블록 내에서 나열할 수 있는 모든 경우의 수가 k!이므로 한 블록(set라 하자)이 각각 k!개의 가능한 블록을 생성된다. 그런 다음 한 set로부터 각각 하나씩 b개를 선택하면 한 블록계획이 구성된다. 이와 같이 총 $(k!)^b$개의 블록계획을 만든 후 그 블록계획법이 LTFB가 되는지 점검 한 후 LTFB인 경우에 한하여서만 출력한다.

프로그램에서는 프로그램 작성시 필요한 $k!$의 일반적인 알고리즘을 얻기 어려워서 b와 k를 고정시켜야만 가능하며 실제의 프로그램은 프로그램이 길고 주어진 알고리즘을 따라 쉽게 작성할 수 있어서 여기서는 생략한다.

아래의 표는 $2 \le b \le 10$이고 $2 \le k \le 5$인 범위에서 몇 개의 모수조합(v, b, r, k)에 따라 존재하는 모든 가능한 LTFB의 갯수(단위:개)에 대해서 작성하였다.

<표 2 - 1> 완비 블록계획법에서 모든 가능한 LTFB의 개수

	(2,2,2,2)	(3,3,3,3)	(2,4,4,2)	(3,5,5,3)	(4,3,3,4)	(2,3,3,2)
총 개수	1	2	1	6	0	0

<표 2 - 2> 불완비 블록계획법에서 모든 가능한 LTFB의 개수

	(4,4,3,3)	(5,10,4,2)	(7,7,3,3)	(4,6,3,2)	(4,10,5,2)
총 개수	24	24	48	0	0

<표 2-1>에서 앞의 4개의 계획법은 'r(k+1)/2가 정수이다'라는 조건이 만족되는 경우이며 나머지는 만족되지 않는 경우이다. <표 2-2>에서 앞의 3개의 계획법은 'r(k+1)/2가 정수이다'라는 조건이 만족되며 BIBD이며 4번째 계획법은 BIBD이나 조건이 만족되지 않는 경우이며, 마지막 계획법은 BIBD도 아니면서 조건도 만족되지 않는 경우이다.

4. 결 론

LTFB는 처리 효과를 추세와 독립적으로 추정할 수 있는 계획법이다. 본 논문에서는 Bradley와 Odeh(1988)에 의해 제시된 LTFB를 발생시키는 알고리즘을 보완하였고 존재하는 모든 가능한 LTFB를 출력하는 알고리즘을 제시하였다. 그렇지만 지금까지는 추측이 성립하는 필요충분조건이 완비 블록계획법과 블록크기가 짝수일 때와 균형불완비 블록계획법일 때에만 증명이 되어있어서 보다 폭넓은 적용을 위하여 이에 대한 일반화가 요구되며 2차 이상의 추세 무관 블록계획법을 설계하는 정리도 향 후 연구과제이다.

참 고 문 헌

[1] Box, G.E.P. (1952). Multi-Factor Designs of First Order, *Biometrika*, Vol.39, 49-57.

[2] Bradley, R.A. and Yeh, C.M. (1980). Trend-Free Block Designs : Theory, *The Annals of Statistics*, Vol. 8, No. 4, 883-893.

[3] Bradley, R.A. and Odeh, R.E. (1988). A Generating Algorithm for Linear Trend-Free and Nearly Linear Trend-Free Block Designs, *Communications in Statistics* - Simulations, 17(4), 1259-1280.

[4] Chai, F.S. and Majumdar, D. (1993). On the Yeh-Bradley Conjecture on Linear Trend-Free Block Designs, *The Annals of Statistics*, Vol. 21, No. 4, 2087-2097.

[5] Coster, D.C. and Cheng, C.S. (1988). Minimum Cost Trend-Free Run Orders of Fractional Factorial Designs, *The Annals of Statistics*, Vol. 16, No. 3, 1188-1205.

[6] Cox, D.R. (1951). Some Systematic Experimental Designs, *Biometrika*, Vol.38, 312-323.

[7] Cox, D.R. (1952). Some Recent Work on Systematic Experimental Designs, *Journal of Royal Statistical Society, B*, Vol.14, 211-219.

[8] Daniel, C. and Wilcoxon, F. (1966). Factorial 2^{p-q} Designs Robust Against Linear and Quadratic Trends, *Technometrics*, Vol.8, 259-278.

[9] Hill, H.M. (1960). Experimental Designs to Adjust for Time Trends, *Technometrics*, Vol.2, 67-82.

[10] Lin, M. and Dean, A.M. (1991), Trend-Free Block Designs for Varietal and Factorial Experiments, *The Annals of Statistics*, Vol. 19, No. 3, 1582-1596.

[11] Stufken, J. (1988), On the Existence of Linear Trend-Free Block Designs, *Communications in Statistics*-Theory Methods, 17(11), 3857-3863.

[12] Yeh, C.M. and Bradley, R.A. (1983), Trend-Free Block Design : Existence and Construction Results, *Communications in Statistics* - Theory Methods, 12(1), 1-24.

[부록 1] : 수정된 알고리즘에 대한 프로그램

```
c     data file checking
      if (k .lt. v) then
         do 81 j = 1, b
           do 81 i = 1, v
             h(i, j) = 0
             do 81 is = 1, k
               if (t(is, j) .eq. i) h(i, j) = h(i, j) + 1
81       continue
        do 82 i = 1, v
             isoftrt = 0
          do 83 j = 1, b
             isoftrt = isoftrt + h(i,j)
83        continue
          if (isoftrt .ne. r) then
            write(6,'(1x,/(a)/)') ' Incorrect Data Input : Check ! '
            write(3,'(1x,/(a)/)') ' Incorrect Data Input : Check ! '
            stop
          endif
82      continue
        do 84 j = 1, b
             isofblc = 0
          do 85 i = 1, v
             isofblc = isofblc + h(i,j)
85        continue
          if (isofblc .ne. k) then
            write(6,'(1x,/(a)/)') ' Incorrect Data Input : Check ! '
            write(3,'(1x,/(a)/)') ' Incorrect Data Input : Check ! '
            stop
          endif
84      continue
      endif

c     Randomness check
        if ( ioo .gt. 1) then
           ijj = 0
           do 41 j_1 = int(b/2), b
             do 42 u_1 = 1, k
               if ( ww(u_1, j_1) .eq. w(u_1, j_1)) ijj = ijj + 1
42           continue
41         continue
        if ( ijj .gt. int((1.0/2)*( b * k )) ) then
           ioo = ioo + 1
           go to 1000
        endif
        endif

c Necessary & Sufficient condition for LTF design
        if ( k .lt. v ) then
          if ((( mod(r*(k+1),2) .eq. 0 ) .and. ( mod(k,2) .eq. 0 )) .or.
     *        (( mod(r*(k+1),2) .eq. 0 ) .and. (iramda .ge. 0))) then
             do 211 j = 1, b
               do 211 i = 1, k
211              t(i, j) = w(i, j)
             go to 1000
          endif
        else
          if ( mod(r*(k+1),2) .eq. 0 ) go to 1000
        endif
```

[부록 2] : LTFB의 예

(1) $v=5, b=10, r=4, k=2, \lambda=1$인 BIBD 10개 생성

수정 알고리즘에 의한 결과		기존 알고리즘에 의한 결과	
Design 1 (q=0)	Design 6 (q=0)	Design 1 (q=0)	Design 6 (q=8)
블록 1 : (2 1)	(2 1)	(2 1)	(2 1)
블록 2 : (1 4)	(1 4)	(1 4)	(1 4)
블록 3 : (3 4)	(3 4)	(4 3)	(3 4)
블록 4 : (3 2)	(2 3)	(3 2)	(3 2)
블록 5 : (2 5)	(5 2)	(2 5)	(2 5)
블록 6 : (5 3)	(3 5)	(3 5)	(5 3)
블록 7 : (1 3)	(1 3)	(1 3)	(3 1)
블록 8 : (5 1)	(5 1)	(5 1)	(1 5)
블록 9 : (4 5)	(4 5)	(5 4)	(4 5)
블록 10 : (4 2)	(4 2)	(4 2)	(4 2)
Design 2 (q=0)	Design 7 (q=0)	Design 2 (q=0)	Design 7 (q=0)
(1 2)	(2 1)	(2 1)	(1 2)
(4 1)	(4 1)	(4 1)	(4 1)
(4 3)	(3 4)	(4 3)	(4 3)
(3 2)	(3 2)	(3 2)	(2 3)
(2 5)	(5 2)	(5 2)	(5 2)
(5 3)	(5 3)	(3 5)	(3 5)
(3 1)	(1 3)	(1 3)	(3 1)
(1 5)	(1 5)	(1 5)	(1 5)
(5 4)	(4 5)	(5 4)	(5 4)
(2 4)	(2 4)	(2 4)	(2 4)
Design 3 (q=0)	Design 8 (q=0)	Design 3 (q=0)	Design 8 (q=0)
(2 1)	(1 2)	(2 1)	(2 1)
(1 4)	(4 1)	(4 1)	(1 4)
(3 4)	(3 4)	(4 3)	(4 3)
(2 3)	(2 3)	(3 2)	(3 2)
(5 2)	(5 2)	(5 2)	(5 2)
(3 5)	(5 3)	(3 5)	(5 3)
(1 3)	(3 1)	(1 3)	(3 1)
(5 1)	(1 5)	(1 5)	(1 5)
(4 5)	(4 5)	(5 4)	(4 5)
(4 2)	(2 4)	(2 4)	(2 4)
Design 4 (q=0)	Design 9 (q=0)	Design 4 (q=8)	Design 9 (q=0)
(1 2)	(2 1)	(1 2)	(2 1)
(4 1)	(4 1)	(1 4)	(4 1)
(4 3)	(3 4)	(4 3)	(3 4)
(3 2)	(3 2)	(2 3)	(3 2)
(2 5)	(5 2)	(5 2)	(5 2)
(3 5)	(5 3)	(3 5)	(5 3)
(1 3)	(1 3)	(3 1)	(1 3)
(5 1)	(1 5)	(5 1)	(1 5)
(5 4)	(4 5)	(5 4)	(4 5)
(2 4)	(2 4)	(2 4)	(2 4)
Design 5 (q=0)	Design 10 (q=0)	Design 5 (q=0)	Design 10 (q=0)
(1 2)	(1 2)	(2 1)	(2 1)
(4 1)	(1 4)	(1 4)	(1 4)
(3 4)	(4 3)	(4 3)	(3 4)
(2 3)	(2 3)	(3 2)	(2 3)
(2 5)	(2 5)	(5 2)	(5 2)
(3 5)	(3 5)	(5 3)	(5 3)
(1 3)	(3 1)	(3 1)	(3 1)
(5 1)	(5 1)	(1 5)	(1 5)
(5 4)	(5 4)	(4 5)	(4 5)
(4 2)	(4 2)	(2 4)	(4 2)
반복되는 것 : Design 3과 6, 7과 9		반복되는 것 : Design 2와 3, 5와 8 LTFB가 아닌것 : Design 4, 6	

(2) $v=7, b=7, r=3, k=3, \lambda=1$인 BIBD 10개 생성

수정된 알고리즘에 의한 결과		기존 알고리즘에 의한 결과	
Design 1 (q=0)	Design 6 (q=0)	Design 1 (q=2)	Design 6 (q=0)
(4 1 2)	(2 4 1)	(1 4 2)	(4 2 1)
(2 5 3)	(3 2 5)	(2 3 5)	(3 5 2)
(3 4 6)	(6 3 4)	(4 6 3)	(6 3 4)
(5 7 4)	(4 5 7)	(5 7 4)	(7 4 5)
(1 6 5)	(5 1 6)	(5 1 6)	(5 1 6)
(6 2 7)	(7 6 2)	(6 2 7)	(2 6 7)
(7 3 1)	(1 7 3)	(7 3 1)	(1 7 3)
Design 2 (q=0)	Design 7 (q=0)	Design 2 (q=2)	Design 7 (q=0)
(2 4 1)	(1 2 4)	(2 1 4)	(2 4 1)
(3 5 2)	(3 5 2)	(3 2 5)	(5 3 2)
(4 6 3)	(6 4 3)	(6 4 3)	(3 6 4)
(5 7 4)	(4 7 5)	(4 5 7)	(4 7 5)
(6 1 5)	(5 1 6)	(5 1 6)	(1 5 6)
(7 2 6)	(2 6 7)	(7 6 2)	(6 2 7)
(1 3 7)	(7 3 1)	(7 3 1)	(7 1 3)
Design 3 (q=0)	Design 8 (q=0)	Design 3 (q=0)	Design 8 (q=0)
(1 4 2)	(1 2 4)	(2 4 1)	(1 4 2)
(2 5 3)	(5 3 2)	(5 3 2)	(2 3 5)
(6 3 4)	(3 4 6)	(3 6 4)	(4 6 3)
(4 7 5)	(4 7 5)	(4 7 5)	(5 7 4)
(5 6 1)	(6 5 1)	(1 5 6)	(6 5 1)
(7 2 6)	(2 6 7)	(6 2 7)	(7 2 6)
(3 1 7)	(7 1 3)	(7 1 3)	(3 1 7)
Design 4 (q=0)	Design 9 (q=0)	Design 4 (q=0)	Design 9 (q=2)
(1 2 4)	(2 1 4)	(2 4 1)	(4 1 2)
(5 3 2)	(3 5 2)	(3 2 5)	(2 3 5)
(4 6 3)	(6 4 3)	(6 3 4)	(6 4 3)
(7 4 5)	(4 7 5)	(4 5 7)	(7 5 4)
(6 5 1)	(5 6 1)	(5 1 6)	(5 1 6)
(2 7 6)	(7 2 6)	(7 6 2)	(2 6 7)
(3 1 7)	(1 3 7)	(1 7 3)	(3 7 1)
Design 5 (q=0)	Design 10 (q=0)	Design 5 (q=0)	Design 10 (q=0)
(2 4 1)	(4 1 2)	(2 4 1)	(1 2 4)
(5 3 2)	(5 2 3)	(5 3 2)	(5 3 2)
(3 6 4)	(6 3 4)	(3 6 4)	(3 4 6)
(4 7 5)	(7 4 5)	(4 7 5)	(4 7 5)
(1 5 6)	(1 5 6)	(1 5 6)	(6 5 1)
(6 2 7)	(2 6 7)	(6 2 7)	(2 6 7)
(7 1 3)	(3 7 1)	(7 1 3)	(7 1 3)
반복되는 것 : 없음 LTFB가 아닌 것 : 없음		반복되는 것 : Design 3과 5와 7 LTFB이 아닌 것 : Design 1과 2와 9	

(3) $v=4, b=4, r=3, k=3, \lambda=2$인 BIBD 10개 생성

수정된 알고리즘에 의한 결과		기존 알고리즘에 의한 결과	
Design 1 (q=0)	Design 6 (q=0)	Design 1 (q=0)	Design 6 (q=0)
(3 1 2)	(2 1 3)	(1 3 2)	(1 2 3)
(4 2 1)	(1 2 4)	(2 4 1)	(2 1 4)
(1 4 3)	(3 4 1)	(4 1 3)	(4 3 1)
(2 3 4)	(4 3 2)	(3 2 4)	(3 4 2)
Design 2 (q=0)	Design 7 (q=0)	Design 2 (q=0)	Design 7 (q=0)
(2 1 3)	(1 3 2)	(2 3 1)	(3 2 1)
(1 4 2)	(2 4 1)	(1 2 4)	(4 1 2)
(4 3 1)	(4 1 3)	(4 1 3)	(1 4 3)
(3 2 4)	(3 2 4)	(3 4 2)	(2 3 4)
Design 3 (q=0)	Design 8 (q=0)	Design 3 (q=0)	Design 8 (q=0)
(1 3 2)	(1 3 2)	(1 2 3)	(2 3 1)
(2 1 4)	(4 2 1)	(2 1 4)	(4 1 2)
(3 4 1)	(3 1 4)	(3 4 1)	(1 4 3)
(4 2 3)	(2 4 3)	(4 3 2)	(3 2 4)
Design 4 (q=0)	Design 9 (q=0)	Design 4 (q=0)	Design 9 (q=0)
(1 3 2)	(3 1 2)	(2 3 1)	(3 2 1)
(2 1 4)	(2 4 1)	(4 1 2)	(4 1 2)
(3 4 1)	(1 3 4)	(1 4 3)	(1 3 4)
(4 2 3)	(4 2 3)	(3 2 4)	(2 4 3)
Design 5 (q=0)	Design 10 (q=0)	Design 5 (q=0)	Design 10 (q=0)
(3 2 1)	(2 3 1)	(1 2 3)	(1 2 3)
(4 1 2)	(1 4 2)	(2 1 4)	(2 1 4)
(1 4 3)	(3 1 4)	(3 4 1)	(4 3 1)
(2 3 4)	(4 2 3)	(4 3 2)	(3 4 2)
반복 되는 것 : Design 3과 4		반복 되는 것 : Design 3과 5, 4와 8, 6과 10	

(4) $v=3, b=5, r=5, k=3$인 완비 블록계획 6개 생성

수정된 알고리즘에 의한 결과		기존 알고리즘에 의한 결과	
Design 1 (q=0)	Design 4 (q=0)	Design 1 (q=0)	Design 4 (q=0)
블록 1 : (1 3 2)	(2 1 3)	(2 3 1)	(3 1 2)
블록 2 : (1 3 2)	(3 1 2)	(3 2 1)	(2 1 3)
블록 3 : (3 2 1)	(1 3 2)	(2 1 3)	(2 1 3)
블록 4 : (2 1 3)	(2 1 3)	(1 3 2)	(1 3 2)
블록 5 : (2 3 1)	(3 2 1)	(1 3 2)	(3 2 1)
Design 2 (q=0)	Design 5 (q=0)	Design 2 (q=0)	Design 5 (q=0)
(2 3 1)	(2 3 1)	(2 1 3)	(1 3 2)
(2 3 1)	(3 1 2)	(2 1 3)	(1 3 2)
(3 1 2)	(1 2 3)	(3 2 1)	(2 1 3)
(1 3 2)	(1 2 3)	(1 3 2)	(3 2 1)
(1 2 3)	(3 2 1)	(3 1 2)	(2 3 1)
Design 3 (q=0)	Design 6 (q=0)	Design 3 (q=2)	Design 6 (q=0)
(3 1 2)	(3 1 2)	(2 1 3)	(1 2 3)
(1 2 3)	(3 1 2)	(2 1 3)	(2 3 1)
(3 1 2)	(2 3 1)	(3 1 2)	(3 1 2)
(2 3 1)	(2 1 3)	(3 1 2)	(1 2 3)
(2 1 3)	(1 2 3)	(3 2 1)	(3 2 1)
반복되는 것 : Design 3과 6		LTFB이 아닌것 : Design 3 반복되는 것 : Design 1과 5, 2와 4	

Generation of Linear Trend-Free Block Designs[1)]

DongKwon Park[2)], HyoungMoon Kim[3)]

Abstract

Randomization of the run order within a block is a technique commonly employed by the experimenters of block designs to avoid biases in the estimates of the effects of interest. In practice, however the experimental responses are sometimes affected by the spatial or temporal position of the experimental units within a block. In such cases, it is preferable to use a systematic ordering of the treatments. It is often possible to find an ordering which will allow the estimation of treatment effects independently of any trend is known as a trend-free block designs. In many industrial and agricultural experiments, treatments are applied to experimental units sequentially in time or space. This paper begins with a review of concepts and properties of trend-free designs. We, then devise algorithms to generate linear trend-free designs. We extend and modify the existing algorithm which is given by Bradley and Odeh(1988). Also, the algorithm which generate all possible linear trend-free designs is provided.

1) This research is supported by a Yonsei University Faculty Research Grant for 96-70.

2) Associated Professor, Department of Statistics, Yonsei University, Wonju City, Kangwon-Do, Korea.

3) Master, Department of Applied Statistics, Yonsei University, Seoul, Korea.