韓國經營科學會誌
第34권 第4號
2009年 12月

1차 확률적 지배를 하는 최대수익 포트폴리오 가중치의 탐색에 관한 연구*

류 춘 호**

An Efficient Algorithm to Find Portfolio Weights for the First Degree Stochastic Dominance with Maximum Expected Return

Choonho Ryu**

■ Abstract ■

Unlike the mean-variance approach, the stochastic dominance approach is to form a portfolio that stochastically dominates a predetermined benchmark portfolio such as KOSPI. This study is to search a set of portfolio weights for the first-order stochastic dominance with maximum expected return by managing the constraint set and the objective function separately. A nonlinear programming algorithm was developed and tested with promising results against Korean stock market data sets.

Keyword : the First-order Stochastic Dominance, Portfolio Selection, Portfolio Weights, Maximum Expected Return, Nonlinear Programming

1. 서 론

불확실한 자산(risky asset)에 투자를 고려하는 사람들은 가급적 수익은 높으면서 위험은 낮은 투자를 원하기에, 대부분의 투자자들은 하나의 자산에만 투자하기보다는 여러 자산에 투자를 분산시킴으로써 투자에 따른 위험을 줄이려고 노력한다. 이럴 경우 가장 대표적으로 사용되고 있는 포트폴리오

논문접수일 : 2009년 11월 16일 논문게재확정일 : 2009년 12월 03일
논문수정일(1차 : 2009년 12월 03일)
* 이 논문은 2006학년도 홍익대학교 학술연구진흥비에 의하여 지원되었음.
** 홍익대학교 경영학부

구성 방법으로는 '평균-분산(mean-variance) 기준'을 들 수가 있다. 평균-분산 기준에 의하면, 포트폴리오 수익률의 표준편차를 투자자들이 느끼는 위험이라고 가정을 하고, 포트폴리오의 기대수익률이 높을수록 그리고 포트폴리오 수익률의 표준편차가 낮을수록 더 우수한 포트폴리오라고 간주하게 된다.

이러한 평균-분산 기준을 이용하면, 매우 드물게 일어나는 시장폭락 가능성을 투자자들이 염려하는 것과 같이 평균과 분산만으로는 실제 투자자들의 행동을 설명할 수 없을 경우, 투자자들이 느끼는 위험을 자산수익률의 표준편차만으로 설명하는 것은 적절하다고 하기가 곤란하다. 이러한 '평균-분산 기준'이 갖고 있는 문제를 보완할 수 있는 방법으로서 '확률적 지배(stochastic dominance)' 개념을 사용한 포트폴리오의 구성이 제시되었다.

확률적 지배 개념은 통계 분야는 물론, 경제, 재무, 마케팅, 의사결정 및 OR 분야에서도 많이 이용되고 개발되어 왔다. 그 중에서도 확률적 지배 개념을 포트폴리오 가중치 결정과정에 적용하는 것을 살펴보면, 류춘호, 신성환[3]은 확률적 지배기준이 평균-분산 기준과 비교하여 어떠한 장점이 있는지를 비교적 자세히 설명하였다. 류춘호[2]는 '2차 확률적 지배(the second-order stochastic dominance)'를 고려할 경우 최적화의 대상이 되는 비선형 목적함수(nonlinear objective function)의 수학적 특성을 분석하여 전체최적성(global optimality)과 국부최적성(local optimality)을 검토하고, 이 알고리즘을 주식시장으로부터의 실제 자료에 적용하여 그 효율성을 시험하였으며, 류춘호[1]는 '1차 확률적 지배(the first-order stochastic dominance)'의 경우에 대해서 포트폴리오의 가중치를 최적화하는 알고리즘을 개발하였다. McNamara[9]도 2차 확률적 지배 개념을 이용한 포트폴리오 구성을 고려하였지만, S&P500과 같은 시장지수와 무위험(risk-free)자산의 합성치를 2차 확률적으로 지배하면서 앞으로의 예상수익률이 최대가 되도록 하는 포트폴리오를 구성하기 위하여 이 문제를 선형계획법(linear programming ; LP)으로 정식화한 후 표본추출을 반복해 가는 모의실험(simulation)을 통해서 포트폴리오를 구성하였다는 점에서 앞의 연구와는 다소 차이가 있다고 하겠다.

다음 단계는 확률적 지배를 하는 포트폴리오 중에서 기대수익률이 가장 큰 것을 찾아내는 것인데, Dentcheva and Ruszczynski[5]는 2차 확률적 지배기준을 유지하는 함수의 볼록성(convexity)과 기대수익률의 선형성(linearity)을 이용해서 이 문제를 선형계획모형으로 정식화하여 최적해를 구하는 방법을 제시하였고, Ruszczynski and Vanderbei[12]는 2차 확률적 지배의 관점에서 지배당하지 않는 포트폴리오(nondominated portfolio)를 구성하는 방법을 선형계획모형으로 정식화하였으며, Dentcheva and Ruszczynski[4]는 2차 확률적 지배를 제약식으로 갖는 최적화 모형의 최적성 조건(optimality condition)과 쌍대이론(duality theory)을 제안하였다.

Kuosmanen[7]은 1차와 2차 확률적 지배를 하는 포트폴리오의 확률적 지배 여부를 알아보고 그 효율성을 측정하는 방법을 제시하였으며, Noyan and Ruszczynski[11]는 이산형(discrete) 자료에 대하여 1차 확률적 지배를 제약식으로 갖는 최적화 모형을 혼합 0-1 정수계획모형(mixed 0-1 integer programming model)으로 정식화한 후, 2차 확률적 지배 개념으로 완화하여 적합부등식(valid inequalities)을 이용하여 heuristic solution을 찾는 방법을 연구하였다.

본 연구에서는, 류춘호[1]가 보여 준 1차 확률적 지배기준의 확보가 매우 효율적이라는 점에 착안하여 1차 확률적 지배를 하는 포트폴리오 중 기대수익률이 가장 큰 것을 찾는 방법을 모색하고자 한다. 먼저 1차 확률적 지배를 하는 가중치(초기해)를 찾아내어 일단 1차 확률적 지배 조건을 확보한 후에는 그 속성을 유지하거나 그것으로부터 크게 벗어나지 않으면서 포트폴리오의 기대수익률이 최대가 되는 곳으로 해를 움직여 가는 방법을 고려해 보고자 한다. 이 방법은 선형함수(linear function)의 형태를 가지는 기대수익률의 최대화는 물론 비선형함수(nonlinear function)의 형태를 가지는 경우(예 : 포트폴리오의 분산 최소화)에도 적용할 수 있다는 점

에서 1차와 2차 확률적 지배의 경우 모두에 적용이 가능하며, 2차 확률적 지배에 대해 선형계획법으로 정식화하여 선형함수의 경우에만 가능하게 되어 있는 Dentcheva and Ruszczynski[5]의 방법보다 일반적이라고 할 수 있다.

다음 장에서 먼저 포트폴리오의 기대수익률(선형함수)을 최대화하는 것을 살펴보고 포트폴리오 수익률의 분산(비선형함수)을 최소화하는 경우에의 적용가능성도 함께 모색해 보고자 한다.

2. 가중치탐색 알고리즘

n개의 주식들의 T개의 시점에 대한 수익률이 주어져 있다고 할 때, n개의 주식으로 포트폴리오를 구성하려고 한다고 하자. 주식 i의 시점 t에서의 전기 대비 수익률을 x_{it}라고 하고, 주식 i에 부여하는 가중치를 ω_i라고 하며, 각 시점 t에 대해서 n개의 주식들의 수익률의 가중합계(포트폴리오)를 X_t라고 하면 다음과 같이 나타낼 수 있다.

$$X_t = \sum_{i=1}^{n} \omega_i x_{it}, \quad \text{where} \sum_{i=1}^{n} \omega_i = 1, \omega_i \in R \quad \forall i.$$

여기서 R은 실수를 의미한다. 이렇게 만든 T개의 X_t를 표본으로 간주하고 각각의 표본에 1/T의 확률을 똑같이 부여해서 실증분포(empirical distribution)를 만들어 낼 수가 있으며, 이것으로부터 X_t의 누적분포함수(cumulative distribution function) $F_x(\cdot)$를 구할 수가 있다. T개의 시점에 대한 KOSPI(Korea Composite Stock Price Index ; 한국종합주가지수)의 수익률 K_t에 대해서도 이와 같이 실증분포를 만들고 그로부터 누적분포함수 $F_k(\cdot)$를 구할 수 있다.

$$F_x(y) = \Pr\{X_t \leq y\} = \frac{p}{T}, \quad \text{where } X_{(p)} \leq y < X_{(p+1)},$$

$$F_k(y) = \Pr\{K_t \leq y\} = \frac{q}{T}, \quad \text{where } K_{(q)} \leq y < K_{(q+1)}.$$

여기서 $X_{(t)}$는 X_t의 순서통계량(order statistics)으로서 크기가 작은 것부터 큰 순서대로 배열했을 때 t번째의 X_t를 의미한다. 즉, $X_{(p)}$는 X_t의 p번째 순서통계량이고, $K_{(q)}$는 K_t의 q번째 순서통계량이 된다.

여기서 우리의 포트폴리오 X_t가 그것의 benchmark가 되는 KOSPI K_t를 확률적으로 지배할 수 있도록 어떻게 ω_i를 결정할 것인지, 그리고 나아가서 그렇게 만들어낸 확률적 지배 포트폴리오 중 최대의 기대수익률을 주는 포트폴리오를 어떻게 찾아낼 것인지가 주 관심사라고 할 수 있다.

2차 확률적 지배 포트폴리오는 위험회피적인 성향을 갖는 투자자만이 선택하고 위험선호적인 투자자는 선택하지 않는 반면에, 1차 확률적 지배 포트폴리오는 투자자의 위험성향이 위험회피적이든 위험선호적이든 관계없이 선택한다는 점에서 훨씬 더 강한 조건을 가지고 있다(Huang and Litzenberger [6], Levy[8]) 즉, 2차 확률적 지배는 효용함수(utility function)의 오목성(concavity)을 전제로 하지만, 1차 확률적 지배는 효용함수의 성질에 전혀 영향을 받지 않는다. 그러므로 1차 확률적 지배 포트폴리오는 언제나 2차 확률적 지배 포트폴리오가 되지만, 그 역은 성립하지 않는다. 그러므로 2차 확률적 지배 포트폴리오의 최대기대수익률은 1차 확률적 지배 포트폴리오의 최대기대수익률보다 항상 크거나 같으며, 따라서 전자는 후자의 상한(upper limit)이 된다고 할 수 있다.

KOSPI를 2차 확률적으로 지배하는 포트폴리오 중 기대수익률을 최대로 하는 가중치를 찾아내는 문제(P_2)는 아래와 같이 정식화할 수 있다.

$$\begin{aligned}
&\underset{\omega}{\text{Max}} \quad E(X) = \frac{1}{T}\sum_{t=1}^{T} X_t \\
(P_2) \quad &\text{s.t. } h(\omega) = \max_{h} \int_{-\infty}^{h} \{F_x(y) - F_k(y)\} dy \leq 0 \\
&\sum_{i=1}^{n} \omega_i = 1 \\
&w = (w_1, w_2, \cdots, w_n) \in R^n
\end{aligned}$$

여기서 첫 제약식의 함수는 비선형(nonlinear)이지만 류춘호[2]는 이 제약식을 0으로 하는 효율적인 알고리즘을 제시하였으며, 이를 이용하여 (P_2)의 최

적해를 찾는 알고리즘을 개발하였다.

그런데 Dentcheva and Ruszczynski[5]는 (P_2)의 실행가능영역이 볼록집합임을 증명하고 이 문제를 선형계획모형으로 정식화하였다. 그러나 이들이 제시하고 있는 LP의 규모는 각 주식별 수익률 자료의 개수가 커지면 제약식의 숫자가 급격히 커지기 때문에 따라서 컴퓨터의 실행시간(CPU time)이 많이 걸릴 수 있다는 단점도 함께 가지고 있다. 즉, N개의 주식에 대한 T시점의 자료를 사용한다면, 의사결정변수가 $(N+T^2)$개, 제약식이 $(T+T^2+1)$개나 된다. 예를 들어, 30개 주식의 일 년 동안의 일별수익률 자료(260일분)의 경우, 67,630개의 변수와 67,861개의 제약식을 가진 LP문제가 된다. 하지만 (P_2)는 본 논문에서 고려하고 있는 문제 (P_1)의 최적목적함수값에 대해 상한을 제공한다는 점에서 대단히 유용하다고 할 수 있다. 즉, 문제 (P_1)의 최적목적함수값을 $V(P_1)$이라고 하면, $V(P_1) \leq V(P_2)$ 관계가 항상 성립한다. 투자를 고려하고 있는 주식의 숫자나 각 주식의 수익률 자료 개수의 변화에 따라 컴퓨터 실행시간이 어떻게 변화하는지, 그리고 본 논문에서 제시한 알고리즘과 어떤 차이를 보이는지에 대해서는 다음 장에서 살펴보기로 한다.

KOSPI를 1차 확률적으로 지배하는 포트폴리오 중 기대수익률을 최대로 하는 가중치를 찾아내는 문제 (P_1)은 아래와 같이 정식화할 수 있다.

$$
\begin{aligned}
&\underset{\omega}{\text{Max}} \quad E(X) = \frac{1}{T}\sum_{t=1}^{T} X_t \\
(P_1) \quad &\text{s.t. } g(\omega) = \max_{h} \{F_x(h) - F_k(h)\} \leqq 0 \\
&\sum_{i=1}^{n} \omega_i = 1 \\
&w = (w_1, w_2, \cdots, w_n) \in R^n
\end{aligned}
$$

그런데 제약식의 $g(w)$는 볼록함수(convex function)도 아니고 전역에서 미분가능(everywhere differentiable)하지도 않기 때문에 알고리즘을 개발하기가 어려웠는데, 류춘호[1]는 $g(w)$를 아래와 같이 다시 정의하여 전역에서 미분가능하게 만들었다.

$$g(\omega) = \int_{-\infty}^{\infty} \{F_x(y) - F_k(y)\}^+ \, dy$$

여기서 $\{F(\cdot)\}^+$는 함수가 0보다 크거나 같은 부분을 의미한다. 이렇게 (P_1)의 첫 번째 제약식을 미분가능하게 바꿈으로써 아래와 같이 (P)를 새로 정의하도록 한다.

$$
\begin{aligned}
&\underset{\omega}{\text{Max}} \quad E(X) = \frac{1}{T}\sum_{t=1}^{T} X_t \\
(P_1) \quad &\text{s.t. } g(\omega) = \int_{-\infty}^{\infty} \{F_x(y) - F_k(y)\}^+ \, dy = 0 \\
&\sum_{i=1}^{n} \omega_i = 1 \\
&w = (w_1, w_2, \cdots, w_n) \in R^n
\end{aligned}
$$

여기서 $g(w)$는 볼록함수는 아니나 전역에서 미분가능하기 때문에 그 값을 0으로 만드는 w를 구하기만 하면 실행가능하며, $f(w)$는 w의 선형함수이기 때문에 알고리즘의 개발이 한결 수월해진다. 즉, 위의 문제는 비볼록집합(non-convex set)인 실행가능영역(feasible region) 위에서 볼록함수인 선형함수를 최대화하는 문제가 된다.

류춘호[1]는 $g(w)$가 볼록함수는 아니지만 $g(w)$의 이론적인 최소값인 0으로 하는(즉, 1차 확률적 지배를 하는) w를 효율적으로 구할 수 있는 알고리즘을 제시하였는데, 본 논문에서는 이를 이용하여 최대의 기대수익률을 찾아내는 알고리즘을 개발하고자 한다. 즉, 시작점에서 출발하여 우선 실행가능영역으로 진입을 하고나서(1차 확률적 지배를 확보하고 나서) 목적함수인 $f(w)$를 증가시키고 그 결과가 실행가능영역을 벗어나면 다시 실행가능영역을 찾아 들어가고 거기서 다시 목적함수를 증가시키는 과정을 계속 반복한다.

이 경우 실행가능영역으로 진입하기 위해서는 $g(w)$를 최소화하는(즉, 0으로 만드는) 과정이 필요한데, $g(w)$는 전역에서 미분가능하므로 도함수기법(gradient method, Minoux[10])을 적용할 수 있다. 이 기법은 $g(w)$의 일차 도함수를 이용하여 축차적으로 다음 해를 구하게 되는데, 현재의 해 w^{old}에서의 일

차 도함수를 $\nabla g(\omega^{old})$라고 하면 새로운 해 w^{new}는 아래와 같이 구할 수가 있다.

$$\omega^{new} = \omega^{old} - \lambda \frac{g(\omega^{old})}{\|\nabla g(\omega^{old})\|^2} \cdot \nabla g(\omega^{old})$$

이와 같이 알고리즘을 적용하여 $g(w)$의 값이 이론적인 최소값인 0이면 최적해에 도달한 것이 된다. 그러나 $g(w)$의 값이 0이 아닌 경우에는 이대로는 최소값으로 수렴하지 않으므로, 일정 회수를 반복해도 $g(w)$의 값이 작아지지 않을 경우는 λ의 값을 줄여가면서 최소값을 찾아갈 수 있도록 조정을 한다.

그러나 $f(w)$를 최대화하는 과정은 이와는 조금 다르게 진행이 된다. (P)에서 $g(w)$는 볼록함수가 아니기 때문에 (P)의 실행가능영역도 볼록집합이 될 수가 없으므로, (P)는 볼록집합이 아닌 실행가능영역 위에서 선형함수인 $f(w)$를 최대화하는 문제가 된다. $f(w)$는 선형함수이기 때문에 쉽게 일차 도함수를 구할 수가 있으므로 $f(w)$를 최대화하는 과정에도 도함수기법을 적용할 수가 있다. 현재의 해 w^k에서의 $f(w)$의 일차 도함수를 $\nabla f(\omega^k)$라고 하면 새로운 해 w^{k+1}은 아래와 같이 구할 수가 있다.

$$\omega^{k+1} = \omega^k + \lambda \frac{f(\omega^*) - f(\omega^k)}{\|\nabla f(\omega^k)\|^2} \cdot \nabla f(\omega^k)$$

여기서 w^*는 (P)의 최적해를 의미하는데, 현재로서는 최적해를 알 수가 없으므로 $f(w^*)$ 대신 (P_2)의 최적해로부터 얻은 최대 기대수익률[$V(P_2)$]을 사용하고자 한다. 물론 $V(P_2)$는 $f(w^*)$와 같지는 않지만 $f(w^*)$의 상한이라는 점에서 효용성이 있으며, 두 값의 차이가 크지 않다면 대단히 유용한 대체값이 될 수가 있다. 만일 두 값의 차이가 상대적으로 크거나 실행가능영역의 비볼록성(non-convexity)으로 인하여 현재의 해가 최적해로부터 먼 곳에 있을 경우에는 $V(P_2)$를 사용하는 것이 새로운 해가 실행가능영역으로부터 너무 멀리 벗어나도록 할 수가 있으므로, 일정 회수를 반복해도 $f(w)$의 값이 커지지 않을 경우는 λ의 값을 줄여가면서 새로운 해를 찾아갈 수 있도록 조정을 한다.

만일 기대수익률을 최대로 하는 포트폴리오를 구하는 대신 포트폴리오 수익률의 표준편차를 최소로 하는 1차 확률적 지배 포트폴리오를 구하고자 한다면, 위의 알고리즘을 조금만 수정하면 가능하게 될 수 있다. 즉, $g(w)$를 최소화하는 과정은 그대로 적용하고, $f(w)$를 최대화하는 과정을 $\sigma(w)$를 최소화하는 과정으로 대체하면 된다. 이 경우 $\sigma(w)$는 w의 2차 함수로 표시되므로 비교적 쉽게 수정할 수가 있을 것이다.

아울러 포트폴리오의 기대수익률[$EX(w)$]과 표준편차[$\sigma(w)$]를 동시에 고려하는 경우도 생각해 볼 수 있는데, 가령 기대수익률과 표준편차에 가중치를 각각 α와 β로 준다면 목적함수를 $Max_w\ \alpha EX(w) - \beta\sigma(w)$로 정의할 수가 있고 이 함수는 w의 2차 함수이므로, 역시 위의 알고리즘의 연장선상에서 수정이 가능할 것이다.

이와 같이 본 논문에서 제시하는 알고리즘은 단지 1차 확률적 지배 포트폴리오의 기대수익률을 최대로 하는 가중치를 구하는 것은 물론이고, 수익률의 표준편차 등과 같은 다른 기준들을 최대화 또는 최소화할 경우에도 그 적용가능성이 크다는 점에서 의의가 있다고 하겠다.

3. 예 제

다음과 같이 세 개의 주식과 KOSPI에 대해 5일 동안의 일별수익률이 존재하는 조그마한 예제에 위의 알고리즘을 적용하여 어떻게 움직이는지를 살펴보고자 한다.

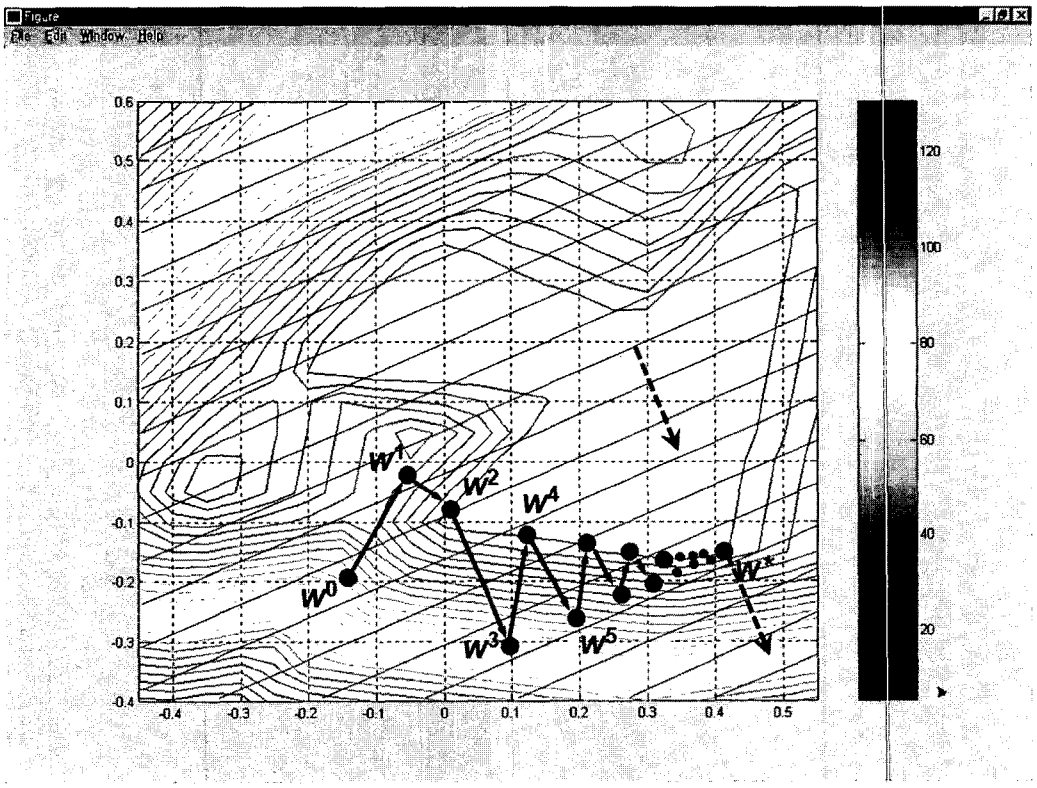
〈표 1〉 기간별 수익률

기 간	A	B	C	KOSPI
1	0.0206	0.1807	-0.0049	-0.0240
2	0.1119	-0.0816	0.0100	-0.0225
3	0.0029	0.0022	-0.0009	0.0235
4	0.0119	-0.0197	0.0396	0.0207
5	-0.0601	0.0111	0.0000	-0.0067

주식 A, B, C로 포트폴리오를 구성하고자 할 경우, 세 주식에 부여되는 가중치를 각각 w_1, w_2, w_3라고 하자. 모든 가중치의 합은 1이어야 하므로 $w_3 = 1 - w_1 - w_2$로 치환할 수가 있고, w_1과 w_2만 결정하면 충분하므로, $g(w)$는 (w_1, w_2)-평면상에 나타낼 수가 있다. [그림 1]을 보면, $g(w)$를 등고선으로 표시하였는데, 문제 (P)의 실행가능영역은 $g(w)$의 값이 0인 부분으로서 등고선의 맨 안쪽에 있는 영역이 되며, 이는 볼록집합이 아님을 분명히 보여주고 있다. 빗금이 등간격으로 그려져 있는데 이는 목적함수인 $f(w)$의 등고선을 나타내고 있으며, 점선으로 표시된 화살표의 방향이 $f(w)$가 증가하는 방향, 즉, $f(w)$의 일차 도함수 방향이 된다.

본 연구에서 제시한 알고리즘은 첫 시작점 w^0에서 출발하여 $g(w)$를 최소화하는 과정을 적용하여 w^1을 거쳐 w^2에 도달하게 된다. 이때 $g(w^2)$의 값이 0이 되면서 w^2는 (P)의 실행가능영역으로 진입하게 된다. 그 다음으로 목적함수인 $f(w)$를 증가시키기 위하여 $f(w)$를 최대화하는 과정을 적용하여 w^3로 이동하게 된다. 그러나 w^3는 실행가능영역 밖에 있어서 실행불가능이므로, 다시 $g(w)$를 최소화하는 과정을 적용하여 실행가능한 w^4로 이동하게 된다. 이 과정을 지속적으로 반복해서 결국에는 w^*에 도달하게 된다([그림 1] 참조).

본 논문에서 제시한 알고리즘은 1차 확률적 지배 포트폴리오의 기대수익률을 최대로 하는 가중치를 찾는 것이어서 2차 확률적 지배 포트폴리오의 기대수익률을 최대로 하는 Dentcheva and Ruszczynski [5]의 선형계획법 알고리즘과는 평면비교를 할 수는 없다. 그러나 본 논문에서 제시한 방법을 2차 확률적 지배 포트폴리오의 기대수익률을 최대로 하는 과정에 동일하게 적용하였을 경우를 살펴보면 본 논



[그림 1] 알고리즘의 이행 과정

문의 알고리즘의 효율성을 짐작할 수 있을 것이다.

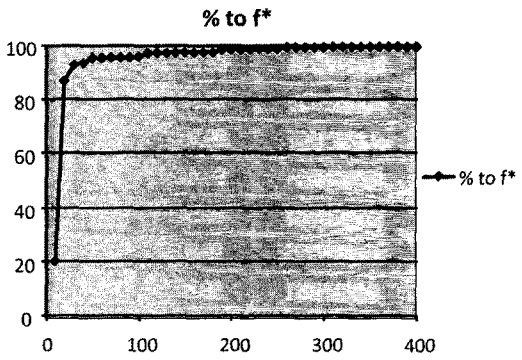
30개 주식의 260일 간의 일별수익률 자료에 대하여 본 연구에서 제시한 1차 확률적 지배 알고리즘을 2차 확률적 지배 알고리즘에 적용하여, 반복(iteration) 횟수와 그에 따른 목적함수 및 컴퓨터 실행시간의 변화를 살펴보면 <표 2>와 같다. 여기서 f^*는 2차 확률적 지배를 하는 포트폴리오의 최대 기대수익률을 나타내며, Dentcheva and Ruszczynski[5]의 알고리즘을 이용하여 GAMS/CPLEX로 구한 값(0.02631575)이다. CPU(sec.)은 IBM PC(Pentium III, 2.34GHz, 2GB RAM)에서 FORTRAN을 이용했을 때의 CPU time(단위 : 초)이다. n은 반복횟수(iteration number)를 나타내고, $f(w^n)$는 반복횟수(n)에 따른 목적함수값을 나타내며, % to f^*는 최적목적함수값에 대한 $f(w^n)$의 %를 나타내고 있다.

〈표 2〉 알고리즘의 반복횟수와 목적함수값의 변화

n	$f(w^n)$	% to f^*	CPU(sec.)
1	-	-	0.01
3	0.00530937	20.176	0.01
10	0.00530937	20.176	0.01
20	0.02280345	86.653	0.01
30	0.02433614	92.477	0.05
40	0.02453646	93.239	0.05
50	0.02503317	95.126	0.05
100	0.02518308	95.696	0.15
200	0.02583738	98.182	0.30
300	0.02603712	98.941	0.45
400	0.02611837	99.250	0.65
500	0.02616094	99.412	0.80
750	0.02619538	99.543	1.19
1,000	0.02621225	99.607	1.59
2,000	0.02628011	99.865	3.25
3,000	0.02629224	99.911	4.84
4,000	0.02630009	99.940	6.49
5,000	0.02630183	99.947	8.14
7,500	0.02630883	99.974	12.23
10,000	0.02631006	99.978	16.28
20,000	0.02631267	99.988	32.71
30,000	0.02631328	99.991	49.15
40,000	0.02631356	99.992	65.58
50,000	0.02631373	99.992	81.97
60,000	0.02631381	99.993	98.40
67,157	0.02631382	99.993	110.09

이 표를 살펴보면, 이 알고리즘이 불과 50번의 반복(0.05초)만에 95%를 넘어섰으며 최적해 근처의 99% 이내에서 110.09초의 시간 대부분을 보내고 있는 걸 알 수가 있다. work station에서 GAMS/CPLEX의 실행시간이 579.17초인 걸 감안하면 1차 확률적 지배의 경우를 위해 개발한 알고리즘이 2차 확률적 지배의 경우에도 효율성이 높음을 알 수가 있다.

[그림 2]는 반복횟수 400회까지의 % to f^*를 표시한 것인데, 이 알고리즘이 얼마나 빨리 최적해에 근접하는지를 보여주고 있다. 만일 최적해에 상당히 근접한(예를 들어, 99% 이내에 도달한) 해로서도 충분하다면, 대부분의 CPU time을 절약할 수도 있을 것이다.



[그림 2] 알고리즘 진행 초기의 목적함수값의 변화

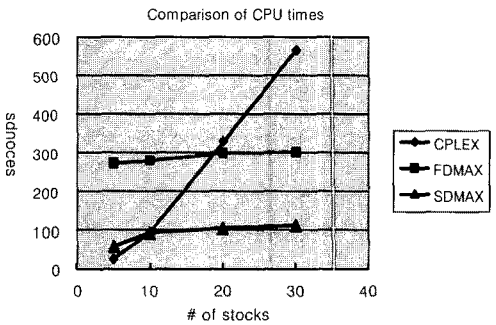
4. 실험 결과

본 논문에서 제시한 1차 확률적으로 지배하는 포트폴리오 중 기대수익률을 최대로 하는 가중치를 찾아내는 알고리즘의 효율성이 어느 정도인지를 알아보기 위하여, 한국의 주식시장으로부터 종목별로 우량하다고 여겨지는 기업들을 50개 골라서 각 기업

의 1년간(260일)의 실제 일별 주식수익률 자료를 구하여 본 연구에서 제시한 알고리즘을 적용하였다.

<표 3>은 5개, 10개, 20개, 30개 회사의 경우에 대해서 6개월(130개)과 1년(260개)의 주식수익률 자료를 선택하되, 각각의 경우에 대하여 주식 조합을 5번에 걸쳐서 임의로 골라 2차 확률적 지배 알고리즘을 적용해 본 결과의 평균치를 보여주고 있다. N은 주식의 개수, T는 주식수익률의 개수를 의미하며, 'LP problem size'는 Dentcheva and Ruszczynski[5]의 알고리즘을 적용할 경우의 문제의 규모를 나타내는 것으로서 '제약'은 제약식의 개수를, '변수'는 의사결정변수의 개수를, 그리고 '≠0'는 제약식에서 0이 아닌 계수의 개수를 의미한다. 'LP/CPLEX'는 이를 GAMS/CPLEX(HP/Unix workstation)로 풀었을 때의 반복횟수와 CPU time(초)을 나타내고, '2SD/FORTRAN'은 2차 확률적 지배 알고리즘을 IBM PC (Pentium III, 2.34GHz, 2GB RAM)에서 FORTRAN을 이용했을 때의 반복횟수, 최종목적함수값의 최적목적함수값(LP)과의 비교 및 CPU time(초)을 나타내고 있다. 130개와 260개의 데이터에 대해서 주식의 수가 증가하면서 LP의 시간은 상당히 가파르게 증가하는 반면, 본 연구에서 제시한 알고리즘은 CPU time의 증가정도가 완만한 것을 알 수 있다.

[그림 3]은 1년(260개)의 주식수익률 데이터에 대해서 2차 확률적 지배 포트폴리오의 최대수익률을 구하는 GAMS/CPLEX의 CPU time(CPLEX)과, 1차 확률적 지배 포트폴리오의 최대수익률을 구하는 본 논문의 알고리즘의 CPU time(FDMAX), 그리고 이 알고리즘을 2차 확률적 지배 포트폴리오의 최대수익률을 구하는 것에 적용하였을 때의 CPU time (SDMAX)에 대해 주식의 숫자가 늘어감에 따른 변화를 대조적으로 보여주고 있다. 이 그림에서 볼 수 있듯이, 주식의 수가 증가하면서 LP의 시간은 상당히 가파르게 증가하는 반면에, 본 연구에서 제시한 알고리즘은 큰 변화가 없는 것을 알 수 있다.



[그림 3] 고려주식의 숫자에 따른 CPU time의 비교

<표 4>는 동일한 자료에 대해서 1차 확률적 지배 알고리즘을 적용한 결과를 보여주고 있다. 그런데 1차 확률적 지배 문제의 최적해는 LP로 구할 수가 없기 때문에, 그 상한값인 2차 확률적 지배 문제의 최적목적함수값과 본 논문에서 제시한 1차 확률

〈표 3〉 2차 확률적 지배 알고리즘의 계산 결과

N	T	LP problem size			LP/CPLEX		2SD/FORTRAN		
		제약	변수	≠0	iter.#	CPU	iter.#	%	CPU
5	130	17,032	16,906	113,345	7,814	2.3	68,191	99.99	27.4
10	130	17,032	16,911	199,649	9,548	4.7	78,831	99.98	33.6
20	130	17,032	16,921	358,321	10,779	12.0	93,139	99.97	42.7
30	130	17,032	16,931	520,346	13,131	23.7	100,000	99.93	49.7
5	260	67,862	67,606	457,403	32,988	24.6	37,076	99.99	56.8
10	260	67,862	67,611	796,141	40,938	93.6	58,245	99.99	92.4
20	260	67,862	67,621	1,442,105	63,639	328.5	61,825	99.98	104.6
30	260	67,862	67,631	2,094,673	77,372	564.9	63,952	99.97	112.7

적 지배 알고리즘으로 구한 최종해의 기대수익률을 비교해보고자 한다. 즉, f^*는 GAMS/CPLEX로 구한 2차 확률적 지배 문제의 최적목적함수값을 의미하며, iter. #는 반복횟수를 의미한다. $f(w^*)$는 본 연구에서 제시한 1차 확률적 지배 알고리즘의 최종목적함수값이고, % to f^*는 $f(w^*)$가 f^* 대비 몇 %나 되는지를 의미하며, $f(w^0)$는 이 알고리즘이 (P)의 실행가능영역으로 들어가면서 처음으로 얻은 초기해의 목적함수값을 보여주고 있다. $f(w^*)/f(w^0)$는 최종해의 기대수익률이 초기해의 기대수익률의 몇 배나 되는지를 나타내고, CPU는 CPU time(초)을 나타내고 있다. 물론 이 모든 수치는 임의로 생성한 5개 데이터의 평균값을 보여주고 있다.

우선 2차 확률적 지배 포트폴리오의 최대 기대수익률에 대한 1차 확률적 지배 포트폴리오의 최대 기대수익률의 %(% to f^*)는 6개월(130개)의 데이터보다는 1년(260개)의 데이터가 높으며, 고려하는 주식의 수가 늘어날수록 높아지는 경향을 보이고 있다. 이것은 두 가지 해석이 가능한데, 첫째는 데이터 숫자의 크기와 무관하게 1, 2차 확률적 지배 포트폴리오의 최대 기대수익률의 차이가 비슷할 경우, 데이터의 숫자가 커질수록 본 논문이 제시한 알고리즘이 더 잘 작동하는 것으로 볼 수도 있고, 둘째는 데이터 숫자의 크기와 무관하게 본 논문이 제시한 알고리즘이 비슷하게 작동할 경우, 데이터의 숫자가 커질수록 1, 2차 확률적 지배 포트폴리오의 최대 기대수익률의 차이가 점점 더 가까워지는 것으로 볼 수도 있다는 점이다.

그리고 1차 확률적 지배 포트폴리오를 하나 찾아내는 것도 의미가 있는 일이지만 그들 중 기대수익률이 제일 좋은 것을 찾아낼 수 있다는 것은 비록 그것이 최적해가 아닐 지라도 실질적인 개선이 된다는 점에서 본 논문에 의의를 부여할 수가 있는데, 최종해의 기대수익률이 초기해의 기대수익률보다 최대 10배 가까이 좋아지는 걸 보여주고 있다.

아울러 2차 확률적 지배 포트폴리오의 경우와 비슷하게 2차 확률적 지배 포트폴리오의 경우에서도, 주식의 수가 증가하더라도 CPU time은 전체적으로 거의 증가하지 않는다고 할 만큼 약간의 증가만을 보여주고 있다는 점에서, 데이터 숫자의 크기에 영향을 많이 받는 다른 알고리즘과는 달리 본 논문이 제시한 알고리즘은 안정성이 크다고 할 수 있을 것이다.

이상에서 살펴본 바와 같이 본 논문이 제시한 1차 확률적 지배를 하는 포트폴리오의 기대수익률을 최대로 하는 가중치를 구하는 알고리즘은 비교적 간단한 구조를 가지고 있지만 효율성이 좋은 것으로 나타났다.

5. 결론 및 향후 연구방향

이상에서 우리는 변수들의 실증분포(empirical distribution)로부터 KOSPI를 1차 확률적으로 지배하는 포트폴리오 중 기대수익을 최대로 하는 가중치를 구하는 알고리즘을 개발하여, 실제 주식시장의

〈표 4〉 1차 확률적 지배 알고리즘의 계산결과

N	T	f^*	iter. #	$f(w^*)$	% to f^*	$f(w^0)$	$f(w^*)/f(w^0)$	CPU
5	130	0.015877	72,716	0.012856	80.07	0.009826	1.34	57.3
10	130	0.020525	100,000	0.017205	80.17	0.007489	2.25	77.8
20	130	0.042320	100,000	0.036945	87.18	0.007160	5.22	78.0
30	130	0.063365	100,000	0.058423	92.21	0.006257	9.66	80.4
5	260	0.009902	85,711	0.008866	89.94	0.007728	1.16	272.8
10	260	0.013530	84,762	0.012444	91.29	0.006800	1.84	280.5
20	260	0.022275	100,000	0.021642	97.16	0.007362	2.94	298.7
30	260	0.026688	100,000	0.026163	98.01	0.005666	4.69	302.4

데이터를 가지고 알고리즘의 효율성을 실험해 보았다. 그 결과 2차 확률적 지배 포트폴리오의 최대 기대수익률에 대한 1차 확률적 지배 포트폴리오의 최대 기대수익률의 %는 고려하는 주식의 수가 늘어날수록 높아지는 경향을 보이고 있고, 이 알고리즘이 구한 최종해의 기대수익률이 초기해의 기대수익률에 비해 많이 좋아진다는 의미 있는 결과를 보여주고 있다.

이 알고리즘은 주식투자 분야에는 물론이고, 변수들의 실증분포를 가지고 1차 확률적 지배(통상적인 확률적 지배)를 하는 가중치를 구하고자 하는 곳에도 이 알고리즘이 그대로 적용될 수 있을 것이라는 점에서 그 활용가능성도 기대해 볼 수가 있다.

그리고 1차 확률적 지배는 위험회피성향을 가정하고 있는 2차 확률적 지배와는 달리 위험성향과 무관하게 선호되는 포트폴리오를 구성할 수 있다는 사실은 이 알고리즘의 실제 적용가능성을 더 높여준다고 할 수 있다.

아울러 본 논문에서 제시한 알고리즘은 1차 확률적 지배 포트폴리오의 기대수익률을 최대로 하는 가중치를 구하는 경우에 활용가능성이 높은 것은 물론이고, 나아가서 1차 확률적 지배 포트폴리오의 기대수익률의 표준편차 등을 최소화할 경우에도 그 적용가능성이 크다는 점에서 의의가 있다고 하겠다.

하지만 본 논문의 알고리즘은 1차 확률적 지배 포트폴리오의 최대 기대수익률을 구하기 위해서 2차 확률적 지배 포트폴리오의 최대 기대수익률을 선형계획모형으로부터 제공 받아 사용해야 한다는 한계점이 있다.

향후 연구방향으로는, 알고리즘 측면에서 다양한 자료에 대해서 이 알고리즘을 실험해 보면서 초기해의 선정, 스텝사이즈의 감소 등에 대한 개선을 모색해 보고, 이러한 최적가중치에 의해 구성되는 포트폴리오가 갖는 여러 가지 통계적 특성들을 살펴보고자 한다. 그리고 가중치가 음인 경우는 양인 경우보다 현실적으로 실행이 쉽지가 않기 때문에, 이 알고리즘이 모든 가중치가 양이라는 제약 하에서는 어떻게 수정되어야 하는지도 검토의 대상이다. 아울러 본 논문에서 제시한 알고리즘을 포트폴리오의 기대수익률 이외에 수익률의 표준편차나 그 둘의 조합을 최소화 또는 최대화하는 가중치를 구하는 경우에도 적용할 수 있도록 하기 위해서 어떠한 수정이 필요한 지, 그리고 실험결과는 어떻게 나오는지도 관심의 대상이다.

참 고 문 헌

[1] 류춘호, "1차 확률적 지배를 하는 포트폴리오 가중치의 탐색에 관한 연구", 「한국경영과학회지」, 제28권 제1호(2003), pp.25-36.

[2] 류춘호, "2차 확률적 지배를 하는 가중치의 탐색에 관한 연구 : 주식투자의 경우를 중심으로", 「경영학연구」, 제28권 제1호(1999), pp.223-239.

[3] 류춘호, 신성환, "최적 포트폴리오 구성에 관한 연구", 「경영연구」, 홍익대학교 경영연구소, 제22권(1997), pp.363-378.

[4] Dentcheva, D. and A. Ruszczynski, "Optimality and Duality Theory for Stochastic Optimization Problems with Nonlinear Dominance Constraints," *Mathematical Programming*, Vol.99, No.2(2004), pp.329-350.

[5] Dentcheva, D. and A. Ruszczynski, "Optimization with Stochastic Dominance Constraints," *SIAM Journal of Optimization*, Vol.14, No.2(2003), pp.548-566.

[6] Huang, C. and R.H. Litzenberger, *Foundations for Financial Economics*, Englewood Cliffs, NJ, Prentice Hall, Inc., 1988.

[7] Kuosmanen, T., "Efficient Diversification According to Stochastic Dominance Criteria," *Management Science*, Vol.50, No.10(2004), pp.1390-1406.

[8] Levy, H., "Stochastic Dominance and Expected Utility : Survey and Analysis," *Management Science*, Vol.38, No.4(1992), pp.555-591.

[9] McNamara, J.R., "Portfolio Selection Using Stochastic Dominance Criteria," *Decision Sciences*, Vol.29, No.4(1998), pp.785-801.

[10] Minoux, M., *Mathematical Programming: Theory and Algorithms*, New York, NY, John Wiley and Sons, Ltd., 1986.

[11] Noyan, N. and A. Ruszczynski, "Valid Inequalities and Restrictions for Stochastic Programming Problems with First Order Stochastic Dominance Constraints," *Mathematical Programming*, Vol.114, No.2(2008), pp.249-275.

[12] Ruszczynski, A. and R.J. Vanderbei, "Frontiers of Stochastically Nondominated Portfolios," *Econometrica*, Vol.71, No.4(2003), pp. 1287-1297.