5.23 モンテカルロシミュレーション 分散減少法(1)　　ver.0.1

対照変量法 Antithetic Variates

これまでのシミュレーション方法を使っていって全く問題はないのだが、少ないシミュレーションの回数で手早く済ませようということになると従来の方法はストレートすぎて、もとになる乱数が若干 Bias があるケースには相当数のシミュレーションを重ねて収束させることが必要になってくる。それに対して、計算機がまだ十分に発達していなかった時代にいろいろとその問題を解決しようとしたのが、これから説明しようとする様々な分散減少法というやり方である。推定値の分散を減少させる。すなわち、同じ分散になるのには、ずっと少ない回数で済むという考え方である。

これから説明する Antithetic variates というやり方は、その分散減少法の中でも比較的手軽にできる方法の１つである。乱数を発生させる際に、もとの乱数からある期待値のまわりで負に相関する乱数を変換することによって用いることで、そのペアを考える（あるいはペアの集まりを考える）ことによって、推定値の散らばりを最初から抑えた形で計算しようというものである。

方法その１（正規対照変量法）

基本的なアルゴリズム

１）$X \sim N(\mu, \sigma^2)$にしたがう乱数を発生させる。

２）$X^* = 2\mu - X$の変換をして、Ｘに負の相関をするＸ*を作成する。

３）今求めたい推定値が$\theta = h(X)$という関数の場合、$\theta^{av} = (h(X) + h(X^*))/2$を計算。

４）θ^{av}について平均および分散を計算する。

以下では、乱数Ｘが平均２分散１の正規分布にしたがうものとするとき、関数ｈをそのＸを２乗するものとして、推定値θの分散が実際に減少しているか確かめてみる。

プログラム

```
new; cls;
n=5000;
fn h(x)=x^2;

/* regular simulation */
mu=2;
X=mu+rndn(2*n,1);
theta=h(X);
print "mean=" meanc(theta);
print " var=" vcx(theta);
```

```
/* antithetic variates */
X1=X[1:n];                          /* Use the same random numbers. */
X2=2*mu-X1;                         /* This means that 2-rndn(n,1).     */
theta=(h(X1)+h(X2))/2;
print "Antithetic Variates:";
print "mean=" meanc(theta);
print " var=" vcx(theta);
```

画面表示

```
mean=          4.9416678
 var=         17.793733
Antithetic Variates:
mean=          5.0073890
 var=         1.9354647
```

実際に、分散の値は格段に減少していることが見てとれる。つまり、ここでは、ｎが 5000 個のケースを扱って（２ｎ個の乱数に対する）その平均をみているが、通常どおりやった分散でよしとするのならば、シミュレーションの回数（ここではｎ）をもっと減らせることを意味する。実際、対照変量法の方はペアで 2ｎ個の乱数（つまりｎ個分の乱数に対する）をｈによって変換した値の平均をみているだけで足りている。それでも、分散は格段に小さい。関数ｈのところをいろいろ取り換えたり、乱数の平均μを変化させたりするとどうなるのかを実際に確かめよう。

今度は、共通の乱数の数ｎ（これはシミュレーションの数でもある）に対して、ｎが１からｎになるまでの間に、その推定値の分散がどのように変化するのかを目で見て確かめるために、理論的な値に対して、ｎが 1,2,3,...となるにしたがって平均値がどのように変化するのかをプロットしてみよう。

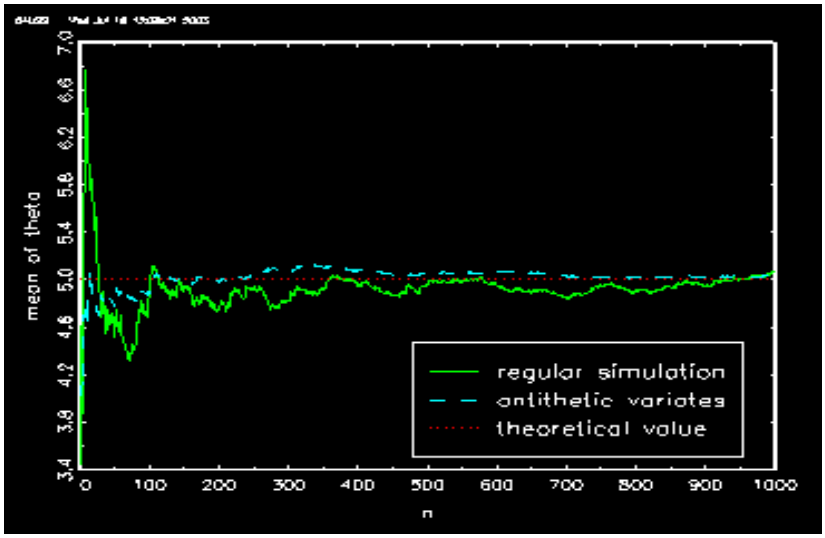
プログラム

```
new; cls;
n=1000;
fn h(x)=x^2;

/* regular simulation */
mu=2;
X=mu+rndn(n,1);
theta1=h(X);
/* antithetic variates */
```

```
X1=X;
X2=2*mu-X1;
theta2=(h(X1)+h(X2))/2;
/* graph */
library pgraph;
graphset;
ns=seqa(1,1, n);
xlabel("n");
ylabel("mean of theta");
_plegctl={2 7 4.5 1};
_plegstr="regular simulation¥000antithetic variates¥000theoretical value";
xy(ns,cumsumc(theta1)./ns~cumsumc(theta2)./ns~(mu^2+1)*ones(n,1));
```

グラフ表示



理論値は、２乗和１個分だから平均１にもとの正規乱数の平均μの２乗が加わったものである。実線の通常のシミュレーションによるθの平均は乱数作成の回数ｎが相当数要しても理論値に収束しないのに対して、antithetic variates によるシミュレーションでは比較的少ない乱数作成回数ｎで理論値にほぼ収束しているのがわかる。この現象は、通常のシミュレーションによるθの分散が antithetic variates による分散よりも大きいこととほぼ同じ意味である。相当数のシミュレーションを重ねても通常のシミュレーションでは比較的大きな分散のままである、すなわち、まだ散らばりが大きいということを表している。

方法その２（一般的な関数に対する対照変量法）

基本的なアルゴリズム

１）U～U(0,1)にしたがう乱数を発生させる。

２）負の相関をするように逆関数法で$F^{-1}(U)$および$F^{-1}(1-U)$を作成する。

３）求めたい推定値が$\theta = F^{-1}(U)$の場合、$\theta^{av} = (F^{-1}(U) + F^{-1}(1-U))/2$を計算。

４）θ^{av}について平均および分散を計算する。

以下では、パラメータλ＝５のときの指数分布の平均をみてみよう。これまでと同様に、乱数の個数（シミュレーションの回数）ｎが 1,2,3,...と増えるにしたがって平均値がどのように変化するかをプロットしてみる。

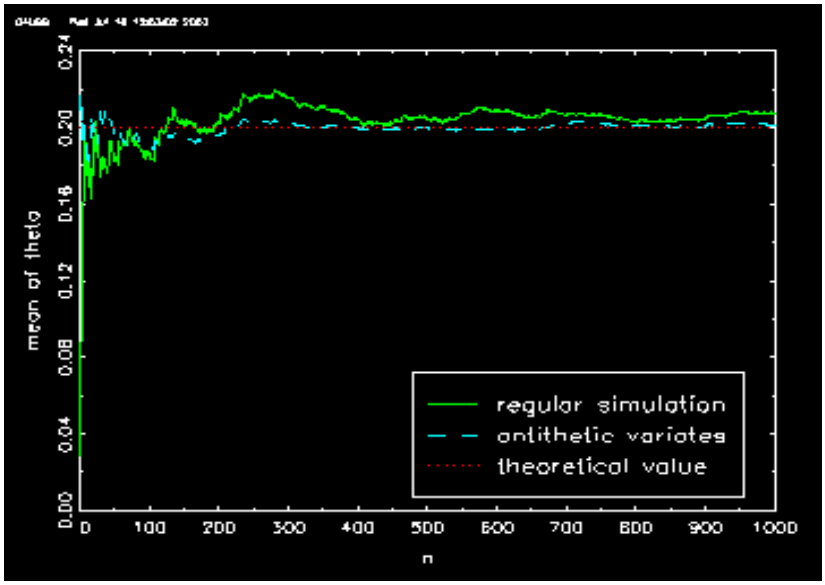
プログラム

```
new; cls;
n=1000;
lambda=5;
fn Finv(x)=-1/lambda*ln(1-x);          /* exponential distribution */

/* regular simulation */
U=rndu(n,1);
theta1=Finv(U);
/* antithetic variates */
theta2=(Finv(U)+Finv(1-U))/2;
/* graph */
library pgraph;
graphset;
ns=seqa(1,1,n);
xlabel("n");
ylabel("mean of theta");
_plegctl={2 7 4.5 1};
_plegstr="regular simulation¥000antithetic variates¥000theoretical value";
xy(ns,cumsumc(theta1)./ns~cumsumc(theta2)./ns~(1/lambda)*ones(n,1));
```

上のプログラムの最後でマージしているように、パラメータλの時の指数分布の平均は、その逆数 1/λとなる。この理論値にいかに収束していくかをこのプログラムは説明している。早く収束すれば、その分だけ分散も小さいということが一般に言える。なお、指数分布の逆関数は、$F^{-1}(X) = -1/\lambda \ln(1-X)$である。なお、自然対数の中は対称であるからＸでも構わないが、まさにこのことを利用してＸに一様乱数をｎ個与えて計算している。

画面表示



理論値である１/λに対照変量法の方がはやく収束するさまが見てとれる。通常のシミュレーションではこの回数では、まだ足りないのがわかる。

幾何ブラウン運動（満期Ｔだけのケース）

ここでは、通常の幾何ブラウン運動の公式の変形である満期Ｔの関係式

$$S(t_T) = e^{\nu T + \sigma \varepsilon(t_i)\sqrt{T}} S(t_0)$$

$$where \quad \nu = \mu - \frac{\sigma^2}{2}$$

にしたがってシミュレーションを行なう。この場合、ε(ｔi)～N(0,1)であるのだが、この標準正規乱数は本来１つだけであって、満期Ｔ期の推定値を得るために times 回分の乱数を用いている。μの部分は書物によってはｒとなっていることもあるので注意が必要である。初期値 S(t_0)から満期の S(t_T)を計算するのである。

基本的なアルゴリズム

１）ｅ～N(0,1)にしたがう標準正規乱数を発生させる。

２）通常の公式にしたがって S(t_T)を計算する。また、公式のｅの部分を-ｅに代えて同様に負の相関をする S(t_T)*を計算する。

３）θ^{av}＝（S(t_T)＋S(t_T)*）/２を計算する。

４）これをtimes回繰り返して（またはtimes個の標準正規乱数ｅに対して一度に計算をして）θ^{av}について平均および分散を計算する。

プログラム

```
new; cls;
S0=100; mu=0.10; sig=0.40; T=1; times=10000;
call gbrantit(S0,mu,sig,T,times);

proc gbrantit(S0,mu,sig,T,times);
    local e,ST,ST1,ST2,STa;
/* regular simulation */
    e=rndn(times,1);
    ST=exp((mu-(sig^2)/2)*T+sig*e*sqrt(T))*S0;
/* antithetic variates */
    ST1=exp((mu-(sig^2)/2)*T+sig*e*sqrt(T))*S0;
    ST2=exp((mu-(sig^2)/2)*T+sig*(-e)*sqrt(T))*S0;
    STa=(ST1+ST2)/2;
/* statistics */
    print "regular simulation:";
    print "   E(ST)=" meanc(ST);
    print "Var(ST)=" vcx(ST);
    print;
    print "antithetic variates:";
    print "   E(ST)=" meanc(STa);
    print "Var(ST)=" vcx(STa);
    retp( meanc(STa) );
endp;
```

画面表示

```
regular simulation:
   E(ST)=          110.84523
Var(ST)=          2112.6502

antithetic variates:
   E(ST)=          110.44698
Var(ST)=          152.37986
```

上のプログラムでは、先にまとめて標準正規乱数ｅを times 個作成しておいて、ｅに対するものと-ｅに対するものを足して２で割ったものを計算している。ここでも、分散は減少していることがわかる。上のプログラムを exponential の中味が平均 (mu-(sig^2)/2)*T、標準偏差が sig*sqrt(T)の正規分布にしたがうものとして、冒頭のケースと同じ変換を行な

うとすると、以下のプログラムは上のプログラムと全く同値である。

プログラム

```
new; cls;
S0=100; mu=0.10; sig=0.40; T=1; times=10000;
call gbrantit(S0,mu,sig,T,times);

proc gbrantit(S0,mu,sig,T,times);
    local e,ST,ST1,ST2,STa,X;
/* regular simulation */
    e=rndn(times,1);
    X=(mu-(sig^2)/2)*T+sig*sqrt(T)*e;
    ST=exp(X)*S0;
/* antithetic variates */
    ST1=exp(X)*S0;
    X=2*(mu-(sig^2)/2)*T-X;
    ST2=exp(X)*S0;
    STa=(ST1+ST2)/2;
/* statistics */
    print "regular simulation:";
    print "   E(ST)=" meanc(ST);
    print "Var(ST)=" vcx(ST);
    print;
    print "antithetic variates:";
    print "   E(ST)=" meanc(STa);
    print "Var(ST)=" vcx(STa);
    retp( meanc(STa) );
endp;
```

基本的なアルゴリズム

１）ｅ～N(0,1)にしたがう標準正規乱数を発生させる。

２）通常の公式にしたがって $S(t_T)$を計算する。また、公式の exponential の中味の部分をＸとみたてて、平均 (mu-(sig^2)/2)*T 標準偏差が sig*sqrt(T)の正規分布にしたがうものとして、Ｘ*＝２μ－Ｘの変換をして負の相関をする $S(t_T)$*を計算する。

３）θ^{av}＝（$S(t_T)$＋$S(t_T)$*）/２を計算する。

４）これをtimes回繰り返して（またはtimes個の標準正規乱数ｅに対して一度に計算をして）θ^{av}について平均および分散を計算する。

幾何ブラウン運動（antithetic variates pair）

もとの標準正規乱数の符合を入れかえることによって、期待値が同じで互いに逆相関するようなペアーを考えて、それをもとに幾何ブラウン運動を生成すれば、できた２つのパスは、上下対称な形となる。ここでは、幾何ブラウン運動の標準的な公式に立ち戻って考えることにしよう。間隔Δｔ（１年 252 営業日であるとするとここは 1/252）に対して、

$$\ln S(t_{k+1}) - \ln S(t_k) = \nu\Delta t + \sigma\varepsilon(t_k)\sqrt{\Delta t}$$

$$S(t_{k+1}) = e^{\nu\Delta t + \sigma\varepsilon(t_k)\sqrt{\Delta t}} S(t_k)$$

初日ｔ$_0$の初期値Ｓ(ｔ$_0$)からｎ日のＳ(ｔ$_n$)をＳ(ｔ$_{k+1}$)とＳ(ｔ$_k$)の漸化式によって計算していく。

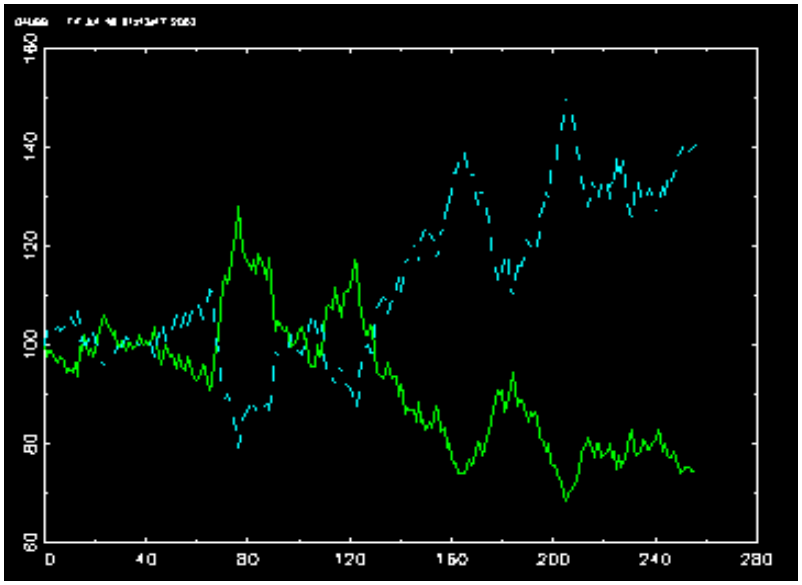
プログラム

```
new; cls;
S0=100; mu=0.10; sig=0.40; delt=1/252; n=252;
call gbravpair(S0,mu,sig,delt,n);

proc gbravpair(S0,mu,sig,delt,n);
    local e,i,e,S1,S2;
/* antithetic variates pair */
    S1=S0|zeros(n,1);
    S2=S0|zeros(n,1);
    e=rndn(n,1);
    i=1;
    do while i<=n;
        S1[i+1]=exp((mu-(sig^2)/2)*delt+sig*e[i]*sqrt(delt))*S1[i];
        S2[i+1]=exp((mu-(sig^2)/2)*delt+sig*(-e[i])*sqrt(delt))*S2[i];
        i=i+1;
    endo;
/* graph */
    library pgraph;
    graphset;
    xy(seqa(0,1,n+1),S1~S2);
    retp(S1~S2);
endp;
```

なお、以下のペアの経路は、乱数の出具合によって、どのようにも変化する。どのような経路をとるにしても、両者は上下対称になっていることがわかるであろう。

グラフ表示



幾何ブラウン運動（通常の S(ｔ)から S(t+1)を生成する方法）

　初期値 S0 から乱数によって満期Ｔ期だけの値をたくさん作ってその平均をとるのではなくて、S(t-1)から S(t)を作成する作業を続ける通常の幾何ブラウン運動の生成を考えて、その最終Ｔ期だけの平均を考えると次のようになる。ちょうど上の antithetic variates pair をシミュレーションの回数 times 回分のペアーだけ作成することになる。

プログラム

```
new; cls;
S0=100; mu=0.10; sig=0.40; delt=1/252; n=252; times=10000;
call gbrantit(S0,mu,sig,delt,n,times);

proc gbrantit(S0,mu,sig,delt,n,times);
    local e,S,ST,i,j,e,S1,S2,STa;
/* regular simulation */
    ST=zeros(times,1);
    j=1;
    do while j<=times;
        S=S0|zeros(n,1);
        e=rndn(n,1);
        i=1;
        do while i<=n;
            S[i+1]=exp((mu-(sig^2)/2)*delt+sig*e[i]*sqrt(delt))*S[i];
            i=i+1;
```

```
        endo;
        ST[j]=S[n+1];
        j=j+1;
    endo;
/* antithetic variates */
    STa=zeros(times,1);
    j=1;
    do while j<=times;
        S1=S0|zeros(n,1);
        S2=S0|zeros(n,1);
        e=rndn(n,1);
        i=1;
        do while i<=n;
            S1[i+1]=exp((mu-(sig^2)/2)*delt+sig*e[i]*sqrt(delt))*S1[i];
            S2[i+1]=exp((mu-(sig^2)/2)*delt+sig*(-e[i])*sqrt(delt))*S2[i];
            i=i+1;
        endo;
        STa[j]=(S1[n+1]+S2[n+1])/2;
        j=j+1;
    endo;
/* statistics */
    print "regular simulation:";
    print "    E(ST)=" meanc(ST);
    print "Var(ST)=" vcx(ST);
    print;
    print "antithetic variates:";
    print "    E(ST)=" meanc(STa);
    print "Var(ST)=" vcx(STa);
    retp( meanc(STa) );
endp;
```

画面表示

```
regular simulation:
   E(ST)=          111.18457
Var(ST)=          2102.8086
```

antithetic variates:

E(ST)= 110.44332

Var(ST)= 157.21711

上の計算では、通常のシミュレーションと antithetic variates によるものとの乱数は異なるものを用いている。なお、満期T期の際と同じようにプログラムを書き替えれば以下のようになる。全く上とやっていることは同じことである。

プログラム

```
new; cls;
S0=100; mu=0.10; sig=0.40; delt=1/252; n=252; times=10000;
call gbrantit(S0,mu,sig,delt,n,times);

proc gbrantit(S0,mu,sig,delt,n,times);
    local e,S,ST,i,j,e,S1,S2,STa,X,Xhat;
/* regular simulation */
    ST=zeros(times,1);
    j=1;
    do while j<=times;
        S=S0|zeros(n,1);
        e=rndn(n,1);
        X=(mu-(sig^2)/2)*delt+sig*sqrt(delt)*e;
        i=1;
        do while i<=n;
            S[i+1]=exp(X[i])*S[i];
            i=i+1;
        endo;
        ST[j]=S[n+1];
        j=j+1;
    endo;
/* antithetic variates */
    STa=zeros(times,1);
    j=1;
    do while j<=times;
        S1=S0|zeros(n,1);
        S2=S0|zeros(n,1);
        e=rndn(n,1);
```

```
        X=(mu-(sig^2)/2)*delt+sig*sqrt(delt)*e;
        Xhat=2*(mu-(sig^2)/2)*delt-X;
        i=1;
        do while i<=n;
            S1[i+1]=exp(X[i])*S1[i];
            S2[i+1]=exp(Xhat[i])*S2[i];
            i=i+1;
        endo;
        STa[j]=(S1[n+1]+S2[n+1])/2;
        j=j+1;
    endo;
/* statistics */
    print "regular simulation:";
    print "    E(ST)=" meanc(ST);
    print "Var(ST)=" vcx(ST);
    print;
    print "antithetic variates:";
    print "    E(ST)=" meanc(STa);
    print "Var(ST)=" vcx(STa);
    retp( meanc(STa) );
endp;
```

応用例（Asian Options）

ここでは、行使価格Kのときの標準的なアジア型オプションを考えよう。その最終期nにおけるペイオフの現在価値は、

$$Y = e^{-\mu n \Delta t} \max\left(0, \sum_{i=1}^{n} \frac{S(i)}{n} - K\right)$$

となる。S(i)のところに幾何ブラウン運動過程を取り入れて、その現在価値の平均を求めてみよう。合わせてその時の現在価値の分散を計算して、antithetic variates を用いると格段に減少していることを示そう。

プログラム

```
new; cls;
S0=30; mu=0.10; sig=0.2; K=27; n=100; delt=1/252; times=10000;
call asian(S0,mu,sig,K,delt,n,times);

proc asian(S0,mu,sig,K,delt,n,times);
```

```
    local Y,Ya,Y1,Y2,i,j,X,Xhat,S1,S2,e;
    Y=zeros(times,1); Ya=zeros(times,1);
    j=1;
    do while j<=times;
        S1=S0|zeros(n,1); S2=S0|zeros(n,1);
        e=rndn(n,1);
        X=(mu-(sig^2)/2)*delt+sig*sqrt(delt)*e;
        Xhat=2*(mu-(sig^2)/2)*delt-X;
        i=1;
        do while i<=n;
            S1[i+1]=exp(X[i])*S1[i];
            S2[i+1]=exp(Xhat[i])*S2[i];
            i=i+1;
        endo;
        Y1=exp(-mu*n*delt)*maxc(0|sumc(S1[2:n+1])/n-K);   /* present value         */
        Y2=exp(-mu*n*delt)*maxc(0|sumc(S2[2:n+1])/n-K);   /* present value         */
        Y[j]=Y1; Ya[j]=(Y1+Y2)/2;                         /* plug them in at last */
        j=j+1;
    endo;
/* statistics */
    print "Asian Option";
    print "regular simulation:";
    print "E[Y]=" meanc(Y);
    print "Var[Y]:" vcx(Y);
    print;
    print "antithetic variates:";
    print "E[Y]=" meanc(Ya);
    print "Var[Y]:" vcx(Ya);
    retp( meanc(Ya) );
endp;
```

画面表示

```
Asian Option
regular simulation:
E[Y]=          3.5002525
Var[Y]:         4.3607840
```

antithetic variates:
E[Y]=　　3.5018587
Var[Y]:　　0.072277474

上のプログラムでは慣例にしたがって、exponential の中味のレベルで変換を行なっているが、標準正規乱数ｅのレベルで単純にマイナス値にすることで負の相関を出すことも可能である。その場合には、X と Xhat の変数を消去してから該当部分を

```
e=rndn(n,1);
i=1;
do while i<=n;
    S1[i+1]=exp((mu-(sig^2)/2)*delt+sig*sqrt(delt)*e[i])*S1[i];
    S2[i+1]=exp((mu-(sig^2)/2)*delt+sig*sqrt(delt)*(-e[i]))*S2[i];
    i=i+1;
endo;
```

というように書き換えることで済む。なお、ｅは exponential の括弧の中の最後でもσの後でも同じことである。上の書き換えによる計算で結果は全く変わらない。

その他のアジア型のオプションである行使価格Ｋに依存しないタイプの

$$Y = e^{-\mu n\Delta t}\max\left(0, S(n) - \sum_{i=1}^{n}\frac{S(i)}{n}\right)$$

に対しても上のプログラムにおいて、Y1 と Y2 を求める計算式を書き換えて、行使価格Ｋをなくすことによって簡単に計算できるであろう。例えば、該当部分を

```
Y1=exp(-mu*n*delt)*maxc(0|S1[n+1]-sumc(S1[2:n+1])/n);
Y2=exp(-mu*n*delt)*maxc(0|S2[n+1]-sumc(S2[2:n+1])/n);
```

と変更する。インプットのＫは必要なくなる。その他のタイプのオプションも同様な計算を加えるだけで簡単に計算できる。ただし、こうした打ち切りの値を含むオプションの計算では、ペアのもう片方は打ち切りの値がゼロよりも下の値をとって最大値の評価の時に０が選択されている可能性もある。十分な回数行なわれている場合には、そのようなケースも含めて裏表万遍無く平均がとられるが、打ち切りの割合の相対的な大きさによってはシミュレーションの回数が必ずしも多くないケースには分散減少のそもそもの効果があらわれないこともある。また、この方法の要件としては「関数の単調性」が仮定されている。そうでない部分があるケースでは、数学的にそもそも分散減少とならないのか、確率論的に分散減少される部分が打ち勝って分散減少が達成されているのかに留意したい。