6.2 フィルター　トレンドとサイクルの分解(2)　　ver.0.1

データがある時系列のモデルにしたがっているものとして、そこから Deterministic トレンド＋Stochastic トレンド＋Cyclical トレンド部分に分離するのが Beveridge-Nelson 分解である。いま前章と同じく D:にデータ datafile17.txt があるものとする。まずは、一番単純な AR(1)モデルからはじめよう。

AR(1)を仮定したケース

Beveridge-Nelson タイプの Decomposition においては、もとの対数系列 Yt を

Yt=TDt＋TSt＋Ct

というふうに３つの成分に分解します。基本的に、Yt の階差が AR(1)にしたがっていると仮定して、その係数μとρを求めた上で残差を計算します。これをもとに、TDt をμをもとに線形直線として求めて、もとの対数系列 Yt からこれを引いた変動する部分を TSt＋Ct と考えます。TSt はランダムウオークを仮定してプラスマイナスを万遍なく変動する残差の和を考えます。その際のスケールパラメータには AR(1)の時の長期インパクトの乗数を用いてこれをかけます。最後に残ったのが Ct ということになります。

プログラム

```
new; cls;
load data[91,2]=D:/datafile17.txt;
y=data[.,2];
call bnar1(y);

proc bnar1(y);
    local n1,dy,x,n,b,e,rho,iter,length,start,finish,i,step,grid,j;
    local ystar,xstar,bstar,rhomax,rhomax1,mu,res,psi,t,TDt,TSt,Ct;
    n1=rows(y);
    y=ln(y);
    dy=y[2:n1]-y[1:n1-1];
    x=ones(n1-1,1);
/* AR(1) process of ln difference to get mu and rho */
    n=rows(x);
    b=inv(x'x)*x'dy;
    e=dy-x*b;
    rho=e[1:n-1,.]'e[2:n,.]/e[1:n-1,.]'e[1:n-1,.];
    print "Iteration:      #          rho";
    print "                    1 ";; print rho;
```

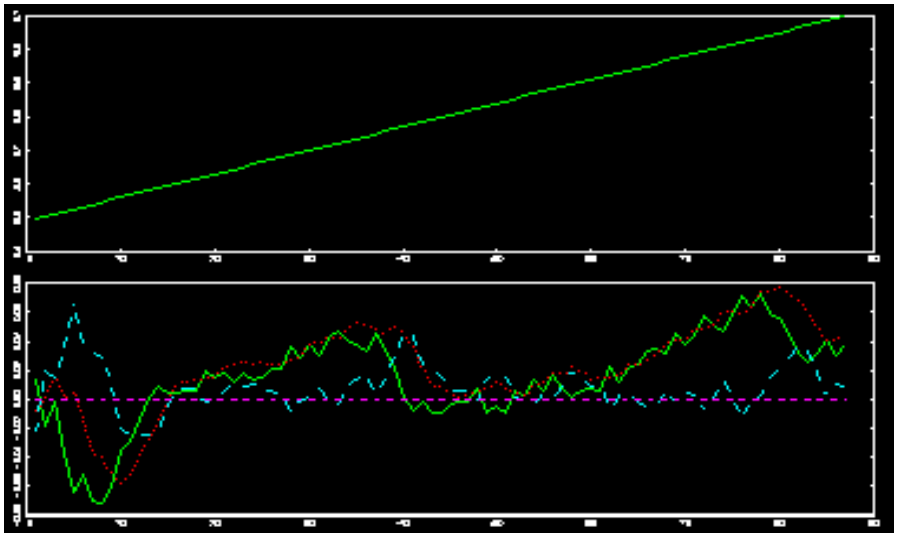
```
    iter=100; length=18;
    start=-0.9; finish=0.9;
    i=1; rhomax1=0;
    do while i<=iter;
        step=(finish-start)/length;
        grid=(-1e256).*ones(length+1,1);
        rho=start;
        j=1;
        do while j<=length+1;
            ystar=sqrt(1-rho^2)*dy[1,.]|dy[2:n,.]-rho*dy[1:n-1,.];
            xstar=sqrt(1-rho^2)*x[1,.]|x[2:n,.]-rho*x[1:n-1,.];
            bstar=inv(xstar'xstar)*xstar'ystar;
            e=ystar-xstar*bstar;
            grid[j]=e'e;
            j=j+1; rho=rho+step;
        endo;
        rhomax=start+(minindc(grid)-1)*step;
        if abs(rhomax-rhomax1)<1e-8;
            break;
        endif;
        rhomax1=rhomax;
        start=rhomax-step; finish=rhomax+step;
        i=i+1; print/rz i;;print rhomax;
    endo;
    rho=rhomax1;
    ystar=sqrt(1-rho^2)*dy[1,.]|dy[2:n,.]-rho*dy[1:n-1,.];
    xstar=sqrt(1-rho^2)*x[1,.]|x[2:n,.]-rho*x[1:n-1,.];
    mu=inv(xstar'xstar)*xstar'ystar;
    print " mu=" mu;
    print "rho=" rho;
    res=(dy[2:n1-1]-mu)-rho*(dy[1:n1-2]-mu);
    psi=1/(1-rho);
    print "psi=" psi;
/* Main BN given mu,rho,psi,res and y */
/* Notice that y is original ln of y. */
    t=rows(res);
```

```
    TDt=y[2]+mu*seqa(1,1,t-2);
    TSt=psi*cumsumc(res[3:t]);
    Ct=y[3:t]-TDt-TSt;
/* Graph */
    library pgraph;
    graphset;
    begwind;
    window(2,1,1);
    setwind(1);
        xy(seqa(1,1,t-2),TDt);
    setwind(2);
        xy(seqa(1,1,t-2),TSt~Ct~(y[3:t]-TDt)~zeros(t-2,1));
    endwind;
    retp(TDt~TSt~Ct);
endp;
```

画面表示

```
mu=      0.0070130868
rho=        0.41298955
psi=         1.7035472
```

グラフ表示



上のように、Deterministic トレンド TDt は上図のように直線になる。これは線形のタイムトレンドのように直線になる。階差を取る前の対数データからこの直線のトレンド部分を引いたものが、下図の赤い部分である。その部分を Stochastic トレンド TSt と Cyclical コンポーネント Ct とに分解している。赤とピンクの間の直線トレンドとの差の部分を、緑の

Deterministic 部分と青色の Cyclical コンポーネント部分に分離している。

上のプログラムでは、インプットには何も変換しない系列ｙを入れて、内部で対数変換や階差をとっています。関数前半の冗長な部分はただ単に AR(1)の係数をμとρを求めるものであって、方法は任意です。上では仮に系列相関の章で行なったのと同じ Hildreth-Lu の Grid Search 法にｙ側に対数階差をｘ側に１ばかりの系列を入れてμとρを求めています。分離すると面倒なので関数のなかに Grid Search を内在化させてしまっています。その後の短い部分が、Beveridge-Nelson に相当するアルゴリズムです。長期乗数には、ここでは計算簡略化のため、１からρを引いたものの逆数を用いています。最後に、TDt の直線の軌跡をグラフ上図に、もとの対数系列 Yt からこの TDt を引いたものの残りを赤線で下図に示した上で、ピンクの０の基準線との間隔を TSt と Ct にその変動を分離しています。

ARMA(1,1)（　ARIMA(1,1,1)　）を仮定したケース

同様のことを ARMA(1,1)について、その係数μ、φ、θに基づいて、Beveridge-Nelson 分解を試みる。やり方はほとんど同じで、長期乗数のところの計算が若干異なるのみである。ARMA(1,1)の係数の計算には、Light 版を考慮して、単純に残差平方和を最小する方法で QNewton 関数を用いて解いている。

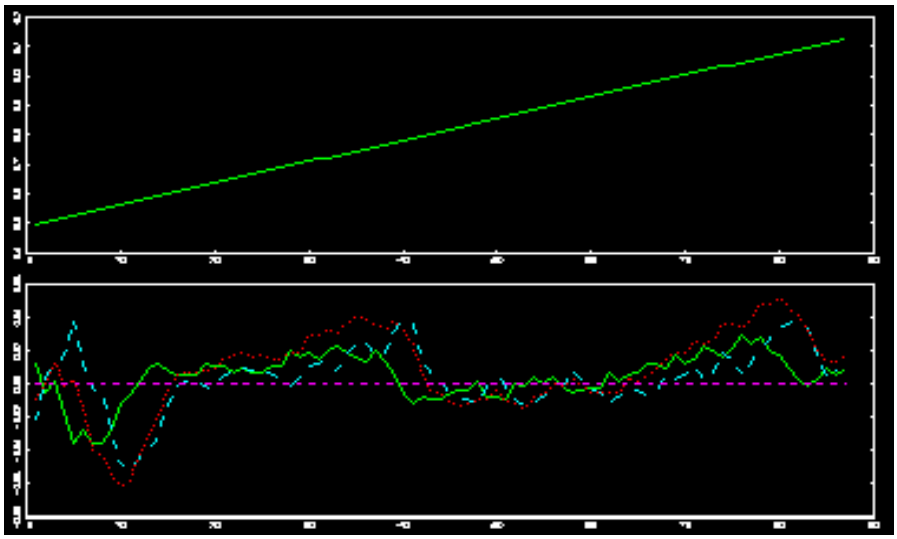
プログラム

```
new; cls;
load data[91,2]=D:/datafile17.txt;
y=data[.,2];
n=rows(y);                  /* Global value passing through */
y=ln(y);                    /* Global vector                */
dy=y[2:n]-y[1:n-1];         /* Global vector                */
datax=dy~ones(n-1,1); /* Global matrix                      */
call bnarma11(y);

proc bnarma11(y);
    local start,x,f,g,ret,mu,phi,theta,u,res,i,psi,t,TDt,TSt,Ct;
/* ARMA(1,1) process to get mu, phi and theta,       */
/* which are approximates by minimization of RSS. */
    start={0,0,0};
    screen off;
    {x,f,g,ret}=QNewton(&arma11,start);
    screen on;
    mu=x[1]; phi=x[2]; theta=x[3];
    u=(dy[2:n-1]-mu)-phi*(dy[1:n-2]-mu);
```

```
    res=zeros(n-2,1);
    res[1]=u[1];
    i=2;
    do while i<=n-2;
        res[i]=u[i]+theta*res[i-1];
        i=i+1;
    endo;
    psi=(1-theta)/(1-phi);
    print "     mu=" mu;
    print "   phi=" phi;
    print "theta=" theta;
    print "   psi=" psi;
    psi=1;
/* Main BN */
    t=rows(res);
    TDt=y[2]+mu*seqa(1,1,t-2);
    TSt=psi*cumsumc(res[3:t]);
    Ct=y[3:t]-TDt-TSt;
/* Graph */
    library pgraph;
    graphset;
    begwind;
    window(2,1,1);
    setwind(1);
        xy(seqa(1,1,t-2),TDt);
    setwind(2);
        xy(seqa(1,1,t-2),TSt~Ct~(y[3:t]-TDt)~zeros(t-2,1));
    endwind;
    retp(TDt~TSt~Ct);
endp;

proc arma11(b);         @ Must be single-value for QNewton. @
    local n1,e,u,v,i;
    n1=rows(datax);
    u=zeros(n1,1); v=zeros(n1,1);
    e=datax[.,1]-datax[.,2]*b[1];
```

```
    u[1]=sqrt(1-b[2]^2)*e[1];
    i=2;
    do while i<=n1;
        u[i]=e[i]-b[2]*e[i-1];
        i=i+1;
    endo;
    v[1]=u[1];
    i=2;
    do while i<=n1;
        v[i]=u[i]+b[3]*v[i-1];
        i=i+1;
    endo;
    retp(v'v);
endp;
```

グラフ表示



対数系列の階差を取ったデータの ARMA(1,1)による係数を、簡易に、残差平方和の最小２乗原理で求めたものなので、不正確かもしれない。また、そもそも ARMA(1,1)が残差を求めるのに適しているのかに問題があるかもしれない。

なお、この Beveridge-Nelson 分解では、常にサイクル Ct と Stochastic トレンド TSt は完全相関の関係にある。なぜならば、もとの系列から線形の Deterministic トレンドを引いた部分を単純に引き算で TSt と Ct に分けているためである。

その他の分析では、対数階差データの ARMA(2,2)が用いられる（これには、Morley の状態空間モデルを用いた ARIMA(2,1,2)のプログラムを詳しくは参照してほしい。そこでは、百倍されたデータが使われていて、μの大きさはその分大きくなるので注意のこと）。

ARIMA(2,1,2)を仮定したケース（参考　ライブラリ optmum が必要）

以下は Morley 博士のページにあるプログラムを動作可能なものにしたものである。太字のところを中心に変更を加えればサイクルのグラフまでたどりつく。

プログラム（参考）

```
new; cls;
library   optmum, pgraph;
/*
**     This program is a general modification of "arima212.opt" & "bn.txt"
**     in    Morley, Nelson, and Zivot "Why Are Beveridge-Nelson and
**     Unobserved-Component Decompositions of GDP So Different?"
*/
format /m1 /rd 9,6;
/* load data[206,1]=D:/lngdpq.txt; Original dataset is natural log of series. */
load data[91,2]=D:/datafile17.txt; data=ln(data[.,2]);
n=rows(data);
yy= 100*data;
dyy=yy[2:n]-yy[1:n-1];
t=rows(dyy);

START=1;
PRMTR_IN={0,0,0,0,0,0};
{xout,fout,gout,cout}=optmum(&lik_fcn,PRMTR_IN);
prm_fnl=trans(xout);
hessn0=hessp(&lik_fcn,xout);
cov0=inv(hessn0);
grdn_fnl=gradfd(&TRANS,xout);
cov=grdn_fnl*cov0*grdn_fnl';
SD =sqrt(diag(cov));
print "Estimated parameters are:" prm_fnl';
DATA = filter(prm_fnl);

/* BN Decomposition Below */
data=dyy~DATA;
mu=prm_fnl[4];
phi1=prm_fnl[2];
phi2=prm_fnl[3];
```

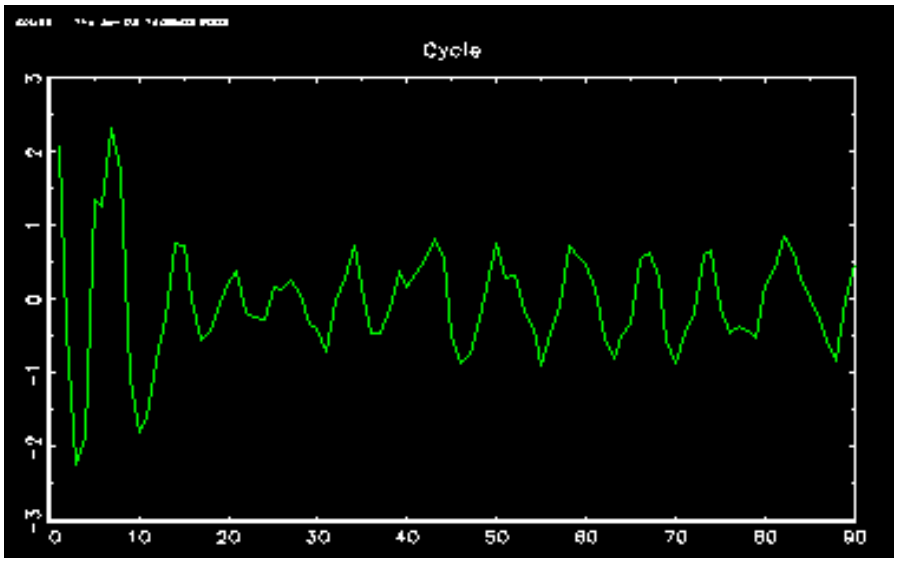
```
theta1=prm_fnl[5];
theta2=prm_fnl[6];
x_mat=zeros(n-1,4);
x_mat[.,1]=data[1:n-1,1]-mu;
x_mat[.,2]=0|data[1:n-2,1]-mu;
x_mat[.,3]=data[1:n-1,2];
x_mat[.,4]=0|data[1:n-2,2];
F=phi1~phi2~theta1~theta2|
      1~0~0~0|
      0~0~0~0|
      0~0~1~0;
BN=(-F*inv(eye(4)-F)*x_mat')';
print BN[.,1];

/* Graph */
graphset;
    xindex=seqa(1,1,rows(BN));
    title("Cycle");
    xy(xindex, BN[.,1]);

proc lik_fcn(PRMTR1);
       local   SN,SE,SNE,F,F1,H,Q,Q1,R,BETA_LL,P_LL,BETA_TL, P_TL;
       local BETA_TT,P_TT,G, P_LL1, vt,ft, VAL,LIK_MAT,J_ITER;
       local phi1,phi2,vecpll,prmtr,muvec,mu,theta1,theta2;
       PRMTR=TRANS(PRMTR1);
       LOCATE 20,1;   PRMTR';
       SE=PRMTR[1];
       phi1=PRMTR[2];
       phi2=PRMTR[3];
       mu=PRMTR[4];
       theta1=PRMTR[5];
       theta2=PRMTR[6];
       F=(phi1~phi2~theta1~theta2)|
         (1~0~0~0)|
         (0~0~0~0)|
         (0~0~1~0);
```

```
        H=1~0~0~0;
        Q=SE^2;
        G=1|0|1|0;
        R= 0;
        BETA_LL=0|0|0|0;
            P_LL = inv( (eye(16)-F.*.F) )*vec(G*Q*G');
            P_LL=   reshape(P_LL,4,4);
       LIK_MAT=ZEROS(T,1);
J_ITER = 1;
DO UNTIL J_ITER>T;
        BETA_TL = F*BETA_LL ;
        P_TL = F*P_LL*F' + G*Q*G';
        vt=dyy[j_iter,1] - mu - H * BETA_TL ;
        ft= H * P_TL * H' + R;
        BETA_TT= BETA_TL + P_TL * H' * inv(ft) * vt;
        P_TT= P_TL - P_TL * H' * inv(ft) * H * P_TL;
        LIK_MAT[J_ITER,1] =   0.5*(LN(2*pi*ft) + vt^2/ft);
        BETA_LL=BETA_TT;
        P_LL=P_TT;
        J_ITER = J_ITER+1;
    ENDO;
    VAL = SUMC(LIK_MAT[START:T]);
    LOCATE 2,20;   VAL;
    RETP(VAL);
endp;

proc filter(PRMTR1);
       local SN,SE,SNE,F,F1,H,Q,Q1,R,BETA_LL,P_LL,BETA_TL;
       local P_TL,BETA_TT,P_TT,G,P_LL1, vt,ft, VAL,LIK_MAT;
       local J_ITER, phi1,phi2,vecpll,prmtr,muvec,mu,theta1,theta2,BETA_MAT;
       BETA_MAT=ZEROS(T,1);
       LIK_MAT=ZEROS(T,1);
       PRMTR=PRMTR1;
       LOCATE 20,1;   PRMTR';
       SE=PRMTR[1];
       phi1=PRMTR[2];
```

```
        phi2=PRMTR[3];
        mu=PRMTR[4];
        theta1=PRMTR[5];
        theta2=PRMTR[6];
        F=(phi1~phi2~theta1~theta2)|
          (1~0~0~0)|
          (0~0~0~0)|
          (0~0~1~0);
        H=1~0~0~0;
        Q=SE^2;
       G=1|0|1|0;
        R= 0;
       BETA_LL=0|0|0|0;
              P_LL = inv( (eye(16)-F.*.F) )*vec(G*Q*G');
              P_LL=   reshape(P_LL,4,4);
     J_ITER = 1;
     DO UNTIL J_ITER>T;
         BETA_TL = F*BETA_LL ;
         P_TL = F*P_LL*F' + G*Q*G';
         vt=dyy[j_iter,1] - mu - H * BETA_TL ;
         ft= H * P_TL * H' + R;
         BETA_TT= BETA_TL + P_TL * H' * inv(ft) * vt;
         P_TT= P_TL - P_TL * H' * inv(ft) * H * P_TL;
         LIK_MAT[J_ITER,1] =   0.5*(LN(2*pi*ft) + vt^2/ft);
         BETA_MAT[J_ITER,.]=BETA_TT[3,1];
         BETA_LL=BETA_TT;
         P_LL=P_TT;
     J_ITER = J_ITER+1;
     ENDO;
     RETP(BETA_MAT[START:T,.]);
endp;

Proc trans(c0);
        local c1,p,varcov,aaa,ccc,e1,e2;
        c1 = c0;
        c1[1]=exp(-c0[1]);
```

```
        aaa=c0[2]./(1+abs(c0[2]));
        ccc=(1-abs(aaa))*c0[3]./(1+abs(c0[3]))+abs(aaa)-aaa^2;
        c1[2]=2*aaa;
        c1[3]= -1* (aaa^2+ccc) ;
        retp(c1);
endp;
```

グラフ表示（参考）



上のプログラムでは、Morley 博士のサイトにある状態空間モデルを用いた arima212.opt とｂｎのプログラムを１つにして、D ドライブにデータをおいてそれにフィルターをかけた上でサイクル成分計算してそのグラフを描かせたものである。アウトプットの ON/OFF の命令と、係数以外の表示は削除してある。上のプログラムは、寄せ集めのプログラムらしくコードの重複や未整理な部分がある。もう少し整理したい。データとしてはすでに自然対数をとったものをインプットとして入れる。