Simulation and estimation of categorical model (E01_est_categ.sce)

```
// Estimation of categorical model
//   f(y(t)|u(t),y(t-1))
// ---------------------------------------
clc, clear, close, mode(0)

u=[1 1 2 1 2 2 1 2 2 1 1 1 2 1 2];
y=[1 1 2 1 2 1 2 1 2 1 2 1 2 2 1];
nd=length(y)-1;
S=1e-8*ones(2,4);                       // initial statistics
// [u(t),y(t-1)] 11 12 21 22
//               ----------
//      y(t)=1 | x  x  x  x
//      y(t)=2 | x  x  x  x

// cumulation of statistics
for t=2:nd
  k=2*(u(t)-1)+y(t-1);                  // active column of the statistics
  S(y(t),k)=S(y(t),k)+1;                // increment of the statistics
end

// point estimates of parameters
for i=1:4
  th(:,i)=S(:,i)/sum(S(:,i));
end
disp('point estimates of parameters',th)
```

Simulation and estimation of regression model (E02_est_regr.sce)

```
// Estimation of regression model
//   y(t)=a1*y(t-1)+b0*u(t)+e(t)
// ------------------------------------
clc, clear, close, mode(0)

nd=100;
a1=.6; b0=1;
y(1)=0;
u=rand(1,nd,'n')-1;
V=zeros(3,3); ka=0;

// time loop
for t=2:nd
  e(t)=rand(1,1,'n');
  y(t)=a1*y(t-1)+b0*u(t)+e(t);              //simulation

  Ps=[y(t) y(t-1) u(t)]';
  V=V+Ps*Ps';                               // statistics update
  ka=ka+1;
end

// results
Vy=V(1,1); Vyp=V(2:$,1); Vp=V(2:$,2:$);     // partitioning of V
th=inv(Vp)*Vyp                              // point estimates

set(scf(),'position',[800 100 600 400])
plot(1:nd,u,1:nd,y)
```

Simulation and estimation of regression model with LS (E03_est_regrLS.sce)

```
// LS estimation of regression model
// y(t)=a1*y(t-1)+b0*u(t)+e(t)
// ------------------------------------
clc, clear, close, mode(0)

nd=1000;
a1=.6; b0=1;
y(1)=0;
u=(rand(1,nd,'n')-1)+0*sin(2*%pi*(1:nd)/nd);
X=[]; Y=[];

// time loop
for t=2:nd
  e(t)=rand(1,1,'n');
  y(t)=a1*y(t-1)+b0*u(t)+e(t);          // simulation

  X=[X; [y(t-1) u(t)]];                 // construction of X
  Y=[Y; y(t)];                          // construction of Y
end

//results
th=inv(X'*X)*X'*Y                       // point estimate of reg. coef.

yp=X*th;
ep=y(2:$)-yp;
r=variance(ep)                          // point estimates of noise variance
```

Initialization and estimation of categorical model (E04_init_categ.sce)

```
// Estimation of coint with prior information
// ------------------------------------------
clc, clear, close, mode(0)

x=(rand(1,10,'u')<.6)+1
S=[10 10];
for t=1:length(x)
  S(x(t))=S(x(t))+1;
  th(t,:)=S/sum(S);
end
plot(th(:,1),'.:')
set(gca(),'data_bounds',[1 length(x) 0 1]);
```

Initialization and estimation of regression model (E05_init_regr.sce)

```
// Estimation of reg. model with prior information
//   y(t)=ay(t-1)+bu(t)+k+e(t)
// ----------------------------------------
clc, clear, close, mode(0)

nd=2000;
a=.6; b=1; k=3; sd=1;
u=rand(1,nd,'n')+1;
y(1)=0; th=[];
ka=100; //.0001;
aI=a; bI=b; kI=k*2;
V=[1 aI bI kI
   aI 1  0 0
   bI 0  1 0
   kI 0  0 1]*ka;
for t=2:nd
  y(t)=a*y(t-1)+b*u(t)+k+sd*rand(1,1,'n');

  Ps=[y(t) y(t-1) u(t) 1]';
  V=V+Ps*Ps';
  ka=ka+1;

  th(t,:)=inv(V(2:$,2:$))*V(2:$,1);
end
plot(th(:,1),'.:')
set(gca(),'data_bounds',[1 length(y) 0 1]);
```

Prediction with categorical model (E06_pred_categ.sce)

```
// Zero-step prediction with categorical model
//   f(y(t)|u(t),y(t-1))
// -------------------------------------
clc, clear, close, mode(0)

nd=20;
th=[.9 .1 .1 .9
    .1 .9 .9 .1];

// simulation
y(1)=1;
for t=2:nd
  u(t)=sum(cumsum([.4 .6])<rand(1,1,'u'))+1;
  i=2*(u(t)-1)+y(t-1);
  y(t)=sum(cumsum(th(:,i))<rand(1,1,'u'))+1;
end

// prediction
yp(1)=1;
for t=2:nd
  i=2*(u(t)-1)+y(t-1);
  yp(t)=sum(cumsum(th(:,i))<rand(1,1,'u'))+1;
end

plot(1:nd,y,'x:',1:nd,yp,'.:','markersize',8)
title 'Discrete signal and its prediction'
```

Prediction with regression model (E07_pred_regr.sce)

```
// Zero-step prediction with regression model
//   y(t)=bu(t)+a*y(t-1)+k+e(t)
// ------------------------------------
clc, clear, close, mode(0)

nd=120;
a=.7;
b=1;
k=-2;
sd=.1;
y(1)=1; u(2)=0;
aE=.5; bE=.1; kE=0;
V=1e-8*eye(4,4);
u=sin(6*%pi*(1:nd)/nd)+1;

for t=2:nd
  // prediction
  yp(t)=aE*y(t-1)+bE*u(t)+kE;

  // simulation
  y(t)=a*y(t-1)+b*u(t)+k+sd*rand(1,1,'n');

  // estimation
  Ps=[y(t) y(t-1) u(t) 1]';
  V=V+Ps*Ps';
  th=inv(V(2:$,2:$))*V(2:$,1);
  aE=th(1); bE=th(2); kE=th(3);
end

// results
set(scf(),'position',[800 100 600 400])
```

```
plot(1:nd,y,1:nd,yp)
title 'Continuous signal and its prediction'
legend('y','yp');

disp 'Simulated parameters'
a,b,k
disp 'Estimated parameters'
aE,bE,kE
```

Prediction with regression model + initialization (E08_pred_regr_ini.sce)

```
// Zero-step prediction with regression model
//   y(t)=bu(t)+a*y(t-1)+k+e(t)
// ------------------------------------
clc, clear, close, mode(0)

nd=120;
a=.7; b=1; k=-2; sd=.1;
y(1)=1; u(2)=0;
ka=1e-8;
aE=.5; bE=.1; kE=0;
V=eye(4,4); V(2:$,1)=[aE bE kE]'; V(1,2:$)=[aE bE kE]; V=ka*V;
u=sin(6*%pi*(1:nd)/nd)+1;

for t=2:nd
  // prediction
  yp(t)=aE*y(t-1)+bE*u(t)+kE;

  // simulation
  y(t)=a*y(t-1)+b*u(t)+k+sd*rand(1,1,'n');

  // estimation
  Ps=[y(t) y(t-1) u(t) 1]';
  V=V+Ps*Ps';
  th=inv(V(2:$,2:$))*V(2:$,1);
  aE=th(1); bE=th(2); kE=th(3);
end

// results
set(scf(),'position',[800 100 600 400])
plot(1:nd,y,1:nd,yp)
title 'Continuous signal and its prediction'
```

```
legend('y','yp');

disp 'Simulated parameters'
a,b,k
disp 'Estimated parameters'
aE,bE,kE
```

State estimation (E09_state_est.sce)

```
// State estimation
//   x(t)=Mx(t-1)+Nu(t)+w(t)
//   y(t)=Ax(t)+Bu(t)+v(t)
// ---------------------------------------
clc, clear, close(winsid()), mode(0), getd()

nd=300;
M=[.9 .1
   .2 .6];
N=[1
   -.5];
A=[.4 -.2];
B=.1;
x=[0
   0];
u=sin(6*%pi*(1:nd)/nd);

Rw=[.1 0;0 .2];
Rv=.1;
Rx=1000*eye(2,2);
xE=[0
    0];

for t=2:nd
  // simulation
  x=M*x+N*u(t)+[.1 0;0 .2]*rand(2,1,'n');
  y(t)=A*x+B*u(t)+.1*rand(1,1,'n');
  xt(:,t)=x;

  // estimation
  [xE,Rx,yp(t)]=Kalman(xE,y(t),u(t),M,N,A,B,Rw,Rv,Rx);
```

```
  xp(:,t)=xE;
end

set(scf(1),'position',[600 0 600 300])
plot([xt' xp'])
legend('xt1','xt2','xp1','xp2');
title 'State and its prediction'

set(scf(2),'position',[600 400 600 300])
plot([y yp])
legend('y','yp');
title 'Output and its prediction'
```

Classification with known model (E10_class.sce)

```
// Classification with known components
// -------------------------------------
clc, clear, close(winsid()), mode(0), getd()

nd=100;
k=[1 5];
for t=1:nd
  c(t)=sum(cumsum([.4 .6])<rand(1,1,'u'))+1;
  y(t)=k(c(t))+rand(1,1,'n');

  for j=1:2
    q(j)=GaussN(y(t),k(j),1);
  end
  w(:,t)=q/sum(q);
  [nill cp(t)]=max(w(:,t));
end

disp 'Accuracy of classification'
acc=sum(c==cp)/nd

set(scf(1),'position',[600 0 600 300])
plot(1:nd,c,'.:',1:nd,cp,'x:')
legend('c','cp');
title 'Pointer and its estimate'
```

Classification with unknown model (E11_class_est.sce)

```
// Classification with known components
// -------------------------------------
clc, clear, close(winsid()), mode(0), getd()

nd=100;
k=[1 5];
kE=[3 4];
ka=[1 1];
S=kE.*ka;
for t=1:nd
  c(t)=sum(cumsum([.4 .6])<rand(1,1,'u'))+1;
  y(t)=k(c(t))+rand(1,1,'n');

  for j=1:2
    q(j)=GaussN(y(t),kE(j),1);
  end
  w(:,t)=q/sum(q);
  [nill cp(t)]=max(w(:,t));
  for j=1:2
    S=S+w(j,t)*y(t);
    ka=ka+w(j,t);
    kE(j)=S(j)/ka(j);
  end
end

disp 'Accuracy of classification'
acc=sum(c==cp)/nd

set(scf(1),'position',[600 0 600 300])
plot(1:nd,c,'.:',1:nd,cp,'x:')
legend('c','cp');
```

```
title 'Pointer and its estimate'
```

Classification with Naive Bayes method - known model(E12_class_NB.sce)

```
// Classification NB with known components
// ---------------------------------------
clc, clear, close(winsid()), mode(0),
function c=pmult(a,b)
  // polynomial multiplication
  // c(1,1)=a(1,1)*b(1,1);
  // c(1,2)=a(1,1)*b(2,1);
  // c(1,3)=a(2,1)*b(1,1);
  // c(1,4)=a(2,1)*b(2,1);
  // c(2,1)=a(1,2)*b(1,2);
  // c(2,2)=a(1,2)*b(2,2);
  // c(2,3)=a(2,2)*b(1,2);
  // c(2,4)=a(2,2)*b(2,2);
  na=size(a,2);
  nb=size(b,2);
  n=size(a,1);
  k=0;
  for i=1:na
    for j=1:nb
      k=k+1;
      c(1:n,k)=a(:,i).*b(:,j);
    end
  end
endfunction

nd=100;
fx=[.3 .7;                              // [f(x1)
    .8 .2];                             //  f(x2)]
fx1Iy=[.3 .6                            // f(x1|y)
       .7 .4];
fx2Iy=[.99 .01                          // f(x2|y)
```

```
       .01 .99];
fy=[.7 .3];                              // f(y)

fxx=pmult(fx1Iy,fx2Iy);                  // f(x1|y)*f(x2|y)
fyx=[fxx(1,:)*fy(1)                      // f(x1|y)*f(x2|y)*f(y)
     fxx(2,:)*fy(2)];
for i=1:4
  fyIx(:,i)=fyx(:,i)/sum(fyx(:,i));      // normalization -> f(y|x)
end

// time loop
for t=1:nd
  // simulation
  for i=1:2
    x(i,t)=sum(cumsum(fx(i,:))<rand(1,1,'u'))+1;
  end
  k=2*(x(1,t)-1)+x(2,t);
  y(t)=sum(cumsum(fyIx(:,k))<rand(1,1,'u'))+1;

  // classification: xt=[x1t,x2t] is the measured data record
  py=fx1Iy(x(1,t),:).*fx2Iy(x(2,t),:).*fy;  // f(x1t|y)*f(x2t|y)*f(y)
  [nill,yp(t)]=max(py);
end

// accuracy of classification
acc=sum(y==yp)/nd
```

Classification with Naive Bayes - unknown model (E13_class_NB_est.sce)

```
// Classification NB with known components
// ---------------------------------------
clc, clear, close(winsid()), mode(0),
getd c:\functions

nd=100;
fx=[.3 .7;                              // [f(x1)
    .8 .2];                             //  f(x2)]

for t=1:nd
  // simulation
  for i=1:2
    x(i,t)=sum(cumsum(fx(i,:))<rand(1,1,'u'))+1;
  end
  if x(1,t)>x(2,t), y(t)=1; else y(t)=2; end
end

// estimation
s=find(y==1);                           // find indexes where y=1
x1=x(:,s);                              // set of x where y=1
m1=mean(x1,2);                          // mean of x1
s=find(y==2);                           // find indexes where y=2
x2=x(:,s);                              // set of x where y=2
m2=mean(x2,2);                          // mean of x2

// classification
for t=1:nd
  q(1)=GaussN(x(:,t),m1,.1*eye(2,2));   // proximity for x1
  q(2)=GaussN(x(:,t),m2,.1*eye(2,2));   // proximity for x2
  w=q/sum(q);                           // weights
  [nill,yp(t)]=max(w);                  // argument maxima w
```

```
end

// accuracy of classification
acc=sum(y==yp)/nd
```