

# Adaptivní řízení s regresním modelem

- regresní model s vektorovým vstupem a výstupem
- model obecného řádu s neznámými parametry
- separace identifikace a řízení
- realizace adaptivního řízení při ustupujícím horizontu

V programu se nejprve provede předběžná identifikace modelu z apriorních dat a spočtou bodové odhady parametrů modelu.

potom, v časové smyčce, se regresní model převede do stavového tvaru. Provádí optimalizace, tj. výpočet řídicího předpisu na celém intervalu řízení, a to proti směru času. Nakonec se realizuje aplikace posledního (tedy aktuálního) předpisu pro řízení. Po jeho aplikaci se generuje (změří) nový výstup a s novými daty se přepočtou bodové odhady parametrů.

Použitý regresní model

$$y_t = b_0 u_t + a_1 y_{t-1} + b_1 u_{t-1} + \dots + a_n y_{t-n} + b_n u_{t-n} + k + e_t.$$

Model se určí zadáním parametrů  $y$ ,  $u$  a konstanty.

**Předpoklady:** Konstantní parametry modelu pro simulaci.

**Sci značení:**  $M$ ,  $N$  - parametry stavového modelu,  $O_m$  - penalizace.

**Úloha:** Řízení s mnohorozměrným regresním modelem.

## Poznámka

*Realizuje se kvadraticky optimální řízení které má sledovat předem daný průběh. Tomuto průběhu se říká žádaná hodnota nebo set-point.*

*Při odhadu během řízení mohou nastat potíže s vybuděním soustavy.*

## Doporučené experimenty

1. Adaptivní řízení ve velice složitá úloha, která vyžaduje určitou inicializaci (předběžný odhad parametrů) tak, aby řízení již od začátku fungovalo alespoň trochu "rozumně". Pokud tomu tak není, může se celá uzavřená smyčka tak vybudit (většinou se generují obrovská čísla, ta se v počítači zaokrouhlí a tím se naruší konzistence systému), že se řízení rozpadne a nekonverguje.
  - (a) Zkuste měnit délku počátečního odhadu  $ni$  a sledujte kvalitu řídicího procesu (zejména na začátku řízení).
  - (b) Dále lze experimentovat s počátečním nastavením informační matice  $V$  (tady nastavena jako diagonální s diagonálou 0.1) a statistikou  $ka$  (čím je větší, tím více se prosazují počáteční hodnoty parametrů).  
Pozor: Matice  $V$  musí být symetrická (viz její přepočet).

2. Pro implementaci algoritmu adaptivního řízení se využívá metoda ustupujícího horizontu. Délka horizontu se zadává jako  $Con.Cy.nh$  a je předvolena na hodnotu 3. Měla by být tak velká, aby se počítaný řídicí zákon stačil ustálit. Zkuste měnit její hodnotu a sledujte kvalitu řízení.
3. V kritériu řízení se penalizuje kvadrát výstupu a kvadrát přírůstku řídicí veličiny, a to s vahou  $la$  (určuje významnost penalizace řízení proti penalizaci výstupu).
  - (a) Zkuste měnit hodnotu  $la \in (0, 1)$  a udělejte závěry o vlivu na kvalitu řízení.
  - (b) Zkuste rozmyslet, jak je třeba změnit penalizační matici  $Om$ , aby se penalizovaly celé hodnoty řízení, nikoliv přírůstky. Porovnejte řízení s penalizací celé řídicí veličiny a jejich přírůstků.
4. Pomocí parametru  $Iadapt$  je možno přepínat řízení adaptivní a řízení se známými parametry. Testujte a porovnejte.

### Program

```
// ADAPTIVE CONTROL WITH DYNAMIC REG. MODEL WITH CONSTANT
// multivariate input and output
// control with setpoint
// penalization of input increments
// preliminary (prior) estimation
// -----
[u,t,n]=file();           // find working directory
chdir(dirname(n(1)));     // set working directory
clear("u","t","n")      // clear auxiliary data
exec("ScIntro.sce",-1),mode(0) // intro to sesion

ni=50;                   // preliminary estimation
nd=150;                  // control

I_adapt=0;              // adaptive control  1=yes, 0=no,

Sim.Cy.ord=2;           // model order,    else fix prior est.
Sim.Cy.th(1).a=[.3 -.2; -.05 -.3];
Sim.Cy.th(2).a=[.1 -.1; .02 -.1];
Sim.Cy.thb0=[1 -.4; 1.3 -.4];
Sim.Cy.th(1).b=[.1 0;0 .2];
Sim.Cy.th(2).b=[.1 0;0 .1];
Sim.Cy.thk=[1; -1];
Sim.Cy.sd=.5;           // noise stdev

ord=Sim.Cy.ord;
ny=size(Sim.Cy.th(1).a,1); // dimension of y
nu=size(Sim.Cy.th(1).b,2)/(1+ord); // and u

th=Sim.Cy.thb0;
for i=1:ord             // parameters -> vector th
```

```

    th=[th Sim.Cy.th(i).a Sim.Cy.th(i).b];
end
th=[th Sim.Cy.thk];
k=Sim.Cy.thk;
sd=Sim.Cy.sd;

Con.Cy.nh=3;          // length of control interval
Con.Cy.om=1;          // penalty: y(t)^2
Con.Cy.la=.01;        // penalty: (u(t)-u(t-1))^2
nh=Con.Cy.nh;

[XX,XX,XX,nx,ny,nu]=reg2st(th,ord);
                    // penalization matrices
Om=zeros(nx,nx);
Om(1:ny,1:ny)=Con.Cy.om*eye(ny,ny);    // output
nn=ny+nu;
for i=1:nu            // input increment
    Om((ny+i),(ny+i))=Con.Cy.la;        // (only for ord>1 !!)
    Om((ny+i+nn),(ny+i+nn))=Con.Cy.la;
    Om((ny+i),(ny+i+nn))=-Con.Cy.la;
    Om((ny+i+nn),(ny+i))=-Con.Cy.la;
end

// setpoint
g=genstp(ny,nd,[3;5],[.95;.92],[-5;10]);

// Prior estimation
yi=zeros(ny,ni);
ui=rand(nu,ni,'norm');
V=.01*eye(nx+ny+nu,nx+ny+nu);
for i=(ord+1):ni
    ps=genps(ord,i,yi,ui);
    yi(:,i)=th*ps + .1*rand(ny,1,'norm');
    Ps=[yi(:,i); ps];
    V=V+Ps*Ps';
end

// TIME LOOP
S=list();
y=zeros(ny,nd); y(:,1:2)=[10 10;-10 -10];
u=zeros(nu,nd);
R=.00001*eye(nx,nx);    // regularizace
for t=(ord+1):(nd-nh)
    // Estimation
    if I_adapt==1
        ps=genps(ord,t,y,u);
        Ps=[y(:,t); ps];
        V=V+Ps*Ps';
        theta=v2thN(V/t,ny);
    end
end

```

```

else
    theta=th';
end

[M,N,A,nx,ny,nu]=reg2st(theta',ord);

// Generation of control law
for i=nh:-1:1
    M(1:ny,$)=-g(:,t+i)+k;
    T=R+0m;
    A=N'*T*N;
    B=N'*T*M;
    C=M'*T*M;
    S(t)=inv(A)*B;
    R=C-S(t)'*A*S(t);
end
// Generation of optimal control
x=genph(ord,t,y,u);
u(:,t)=-S(t)*x;
y(:,t)=th*[u(:,t); x]+.01*rand(ny,1,'norm');
end

// Results
s=1:(nd-nh);
subplot(211)
plot(s,y(1,s),s,u(1,s),'--',s,g(1,s),'m.','markersize',2)
legend('output','input','setpoint',4);
subplot(212)
plot(s,y(2,s),s,u(2,s),'--',s,g(2,s),'c.','markersize',2)
set(gcf(),'position',[100 50 600 500])
legend('output','input','setpoint',1);

```