Odhad směsi se smíšenými daty

Pod názvem smíšená data máme na mysli data, která obsahují jak spojité y_t tak i diskrétní z_t veličiny. Běžné směsi obsahují dva typy modelů. Jednak jsou to komponenty (ať už spojité nebo diskrétní) a model ukazovátka c_t. Tady budeme mít tři druhy modelů: spojité komponenty, diskrétní komponenty a model ukazovátka. Spojité komponenty budou modelovat spojité veličiny (v závislosti na diskrétních) a jejich parametry označíme $\Theta_{c,z}$, diskrétní komponenty budou diskrétní modely pro diskrétní veličiny, s parametrem $\vartheta_{z|c}$ a model ukazovátka bude také diskrétní s parametrem α_c.

Algoritmus odhadu odvodíme opět rozvojem sdružené pravděpodobnosti $\mathcal{J}$ pro neznámé veličiny, kde $d(t-1) = \{y(t-1), z(t-1)\}$

$$\begin{aligned} \mathcal{J} &= f(y_t, z_t, c_t, \Theta, \vartheta, \alpha | d(t-1)) = \\ &= f(y_t | z_t, c_t, \Theta) f(z_t | c_t, \vartheta) f(c_t | \alpha) f(\Theta, \vartheta, \alpha | d(t-1)) \end{aligned} \tag{1}$$

kde prvé tři hustoty pravděpodobnosti (pravděpodobnostní funkce) odpovídají jmenovaným modelům. Tyto modely napíšeme podrobněji.

Modely

Modely spojitých veličin

$$y_t = \Theta_{z_t, c_t} + e_t$$

tedy např. pro dimenzi y_t tři, dvě hodnoty diskrétní veličiny a pět komponent budeme mít 5*2 = 10 modelů, každý bude mít parametr jako dvourozměrný vektor. Tedy

$$\begin{bmatrix} y_{1;t} \\ y_{2;t} \end{bmatrix} = \begin{bmatrix} \Theta_1 \\ \Theta_2 \end{bmatrix}_{z_t, c_t} + \begin{bmatrix} e_{1;t} \\ e_{2;t} \end{bmatrix}, \quad \text{pro } z_t = 1, 2 \text{ a } c_t = 1, 2, \cdots, 5$$

Poznámka

Pro inicializaci musíme tedy zadat 10 dvouprvkových vektorů jako počáteční parametry.

Modely diskrétních veličin

Uvedeme jako příklad se zavedenými dimenzemi. Model bude

	$z_t = 1$	$z_t = 2$
$c_t = 1$	$\vartheta_{1\|1}$	$\vartheta_{2\|1}$
$c_t = 2$	$\vartheta_{1\|2}$	$\vartheta_{2\|2}$
$\cdots$	$\cdots$	$\cdots$
$c_t = 5$	$\vartheta_{1\|5}$	$\vartheta_{2\|5}$

Model ukazovátka

Opět jako příklad

$c_t = 1$	α_1
$c_t = 2$	α_2
$\cdots$	$\cdots$
$c_t = 5$	α_5

Výpočet vah

Váhy w_t jsou hodnoty pravděpodobnostní funkce $f\left(c_t|d\left(t\right)\right)$. Tu určíme ze sdružené pravděpodobnosti 1 takto

$$f\left(c_t|d\left(t\right)\right) \propto \int_{\Theta^*}\int_{\vartheta^*}\int_{\alpha^*} f\left(y_t, z_t, c_t, \Theta, \vartheta, \alpha|d\left(t-1\right)\right) d\alpha d\vartheta d\Theta =$$

$$= \int_{\Theta^*}\int_{\vartheta^*}\int_{\alpha^*} f\left(y_t|c_t, z_t, \Theta\right) f\left(z_t|c_t, \vartheta\right) f\left(c_t|\alpha\right) f\left(\Theta, \vartheta\alpha|d\left(t-1\right)\right) =$$

$$= \int_{\Theta^*} f\left(y_t|c_t, z_t, \Theta\right) f\left(\Theta|d\left(t-1\right)\right) d\Theta \int_{\vartheta^*} f\left(z_t|c_t, \vartheta\right) f\left(\vartheta|d\left(t-1\right)\right) d\vartheta \int_{\alpha^*} f\left(c_t|\alpha\right) f\left(\alpha|d\left(t-1\right)\right) d\alpha$$

Tedy váhy se dostanou součinem odhadnutého spojitého modelu, diskrétního modelu a modelu ukazovátka. Spojité modely generují proximity, diskrétní modely jsou reprezentovány svými maticemi parametrů.

Statistiky

Pro odhad zavedeme **statistiky** a uvedeme jejich **přepočty**

Spojité komponenty

$$V_{c,z_t;t} = V_{c,z_t;t-1} + w_c \Psi\Psi', \quad \kappa_{c;t} = \kappa_{c;t-1} + w_c$$

kde $\Psi = \left[y_t^{'}, 1\right]'$.

Diskrétní komponenty

$$U_{z_t|c;t} = U_{z_t|c;t-1} + w_c$$

Model pointru

$$\nu_{c;t} = \nu_{c;t-1} + w_c$$

.. to celé pro $c = 1, 2, \cdots, nc$.

Poznámka

Protože z_t známe (měříme), provedeme vzorce vždy jen pro jeho jednu hodnotu. Pointer c_t neznáme (neměříme), vzorce tedy musíme provést pro všechny jeho hodnoty a pro každou hodnotu se použije váha w_c.

Bodové odhady parametrů

Protože jednotlivé modely jsou standardní spojité nebo diskrétní modely, výpočet parametrů bude také standardní - tj. pro spojité modely rozdělíme informační matici a z jejích submatic spočteme parametry; pro diskrétní modely spočívá výpočet parametrů v normalizaci statistiky tak, aby součty prvků v řádcích byly rovny jedné.

Detaily algoritmu budeme sledovat přímo za programem, který následuje:

```
// Mixture estimation - static components and pointer model
// - 14 continuous + 2 disctere  variables
// - random initialization (prepared for expert knowledge)
// - display by marginals - select v1,v2 (variables) and z (discrete)
exec("ScIntro.sce",-1),mode(0)          // intro to sesion
rand('seed',0)
function mixGr(v1,v2,z)
  // plot and print of clusters
  // v1,v2 - variables for marginal
  // z     - value of discrete variable
  disp('Centers of selected marginals - all components')
  set(scf(100),'position',[850 50 600 400])
  for i=1:nc
    m=psi2row([z,i],[nz,nc]);
    if ~isempty(yC(m))
      plot(yC(m)(v1,:),yC(m)(v2,:),tx(m),'markersize',3)
      disp(Est.Cy(m).th([v1,v2]),'Cent '+string(m)+', (z = '+string(z)+')')
    end
  end
  title('Variables '+string(v1)+', '+string(v2)+'  and  zt = '+string(z))
endfunction

nd=1350;            // number of data: all = 1350
nc=5;               // number of components
I_estCov=0;         // estimation of noise covariances ? 0|1 no|yes

// DATA ============================================================
load _data/d_con.dat d_con;         // HERE, THE DATA ARE LOADED
load _data/d_ord.dat d_ord;
load _data/d_nom.dat d_nom;

// DATA ASSIGNMENT (time runs in rows)
[dsc,mD,sD]=scal([d_con';d_ord']); // scaling
// by scalling we transform data to standard range -> init. by rand is possible
yt=dsc;                        // scaled continuous and ordinal data
yu=unscal(yt,mD,sD);           // unscaled data
[nv,nd]=size(yt);
dd=d_nom(:,2);
bn=max(dd,'r');            // nominal data
zz=dd';
for i=1:nd
  zt(1,i)=psi2row(zz(:,i),bn); // coding 2var -> 1var
end
nz=max(zt);                    // number of values of nominal variable

// MODEL INITIALIZATION
// random generation of initial centers
for j=1:nz     // components are indexed by z and c
for k=1:nc
```

```
  i=psi2row([j,k],[nz,nc]);
  Est.Cy(i).th=rand(1,nv,'u'); // HERE SET THE INITIAL CENTERS OF COMPONENTS
  // for good clustering it is necessary to set better initial centers
  Est.Cy(i).sd=0.01*eye(nv,nv); // covariances of components
  Est.Cy(i).cv=Est.Cy(i).sd**2;
  Ps=[Est.Cy(i).th 1]';       // initial parameters
  Est.Cy(i).V=Ps*Ps';         // statistics
end
end
Est.ka=ones(1,nc);            // counter
// initial discrete pars
Est.Cz.V=rand(nz,nc,'u')+1;
Est.Cz.th=fnorm(Est.Cz.V,2);
// initial pointer probabilities
Est.Cp.V=ones(1,nc);          // pointer statistics
Est.Cp.th=fnorm(ones(1,nc));  // pointer parameter
Est.ct(1)=1;                  // initial poiner value
w=fnorm(ones(1,nc));          // weights
tht=list(); for i=1:nc, tht(i)=[]; end
wt=zeros(nc*nz,nd);
ii=zeros(nc,nd);

// ESTIMATION ==========================================================
printf(' ')
kk=ceil(nd/10); printf('\n  2 4 6 8 |\n ')
for t=1:nd                    // time loop
if t/kk==fix(t/kk), printf('.'); end
  // proximities continuous and ordinal
  for k=1:nc
    i=psi2row([k,zt(t)],[nc,nz]);
    [xxx,G(k)]=GaussN(yt(:,t),Est.Cy(i).th,Est.Cy(i).cv); // likelihood
    G(k)=G(k)*Est.Cz.th(zt(t));
    ii(k,t)=i;
  end
  Lq=G-max(G);
  q=exp(Lq(:)');

  ww=q'.*Est.Cp.th'; w=ww/sum(ww);          // generation of weights
  wt(ii(:,t),t)=w;

  // Update of statistic
  Ps=[yt(:,t)' 1];                          // extended reg.vec.
  for k=1:nc
    i=psi2row([k,zt(t)],[nc,nz]);           // selected components (zt)
    Est.Cy(i).V=Est.Cy(i).V+w(k)*Ps'*Ps;    // information matrix
    Est.ka(k)=Est.ka(k)+w(k);               // counter
    Est.Cz.V(zt(t),k)=Est.Cz.V(zt(t),k)+w(k);           // discrete stat
    Est.Cp.V(k)=Est.Cp.V(k)+w(k);           // pointer statistics
```

```
    // Point estimates
    Vyy=Est.Cy(i).V(1:nv,1:nv);              // part Vyy - y.y'
    Vy=Est.Cy(i).V($,1:nv);                  // part Vy  - psi.y
    V1=Est.Cy(i).V($,$);                     // part V1  - psi.psi'
    Est.Cy(i).th=inv(V1+1e-8*eye(V1))*Vy;    // pt.est. - reg.coef.
    if t>nd+1, I_estCov=1; end               // DELAYED ESTIMATION OF COV
    if I_estCov~=0
      // pt.est. of noise covariance - used or not
      Est.Cy(i).cv=(Vyy-Vy'*inv(V1+1e-8*eye(V1))*Vy)/Est.ka(k);
    end
  tht(k)=[tht(k) Est.Cy(k).th'];
  end
  Est.Cz.th(zt(t),:)=fnorm(Est.Cz.V(zt(t),:)); // nominal variables

  Est.Cp.th=fnorm(Est.Cp.V,2);     // pt.est. of pointer parameter
  [ss,Est.ct(1,t)]=max(w);         // store of c
end
[xxx,cE]=max(wt,'r');
Est0=Est;
save est0.dat Est0;                // save for repetitive running

// RESULTS
disp ' '; disp(vals(Est.ct),'Frequencies of pointer values')

// plot of weights (last 100 weights)
set(scf(1),'position',[50 50 400 300]);
plot(wt(:,($-100):$)')
title('Weights - check of functionality')

// plot of of evolution of parameter estimates
if 0                // activate by 1
for i=1:nc
  scf(10+i);
  plot(tht(i)')
  title('comp '+string(i))
end
end

// Detection of centers of components in original scale
// - for each z, nc-points from 14-dimensional space is stored
// - type Th + Enter to see the centers
Th=list();
for j=1:nz
  zz=[];
  for k=1:nc
    i=psi2row([j,k],[nz,nc]);
    zp=unscal(Est.Cy(i).th',mD,sD)';
    zz=[zz; zp];
  end
```

```
  jz=row2psi(j,bn);
  Cz(jz(1),jz(2)).c=zz;
  Th(j)=zz;
end

tx1=['bo';'ro';'go';'mo';'ko';'co';'yo'];
tx2=['bx';'rx';'gx';'mx';'kx';'cx';'yx'];
tx3=['b+';'r+';'g+';'m+';'k+';'c+';'y+'];
tx=[tx1;tx2;tx3];

yC=list();
for k=1:nz
  for i=1:nc
    m=psi2row([k,i],[nz,nc]);
    j=find((cE==i)&(zt==k));
    yC(m)=yt(:,j);
  end
end

v1=1; v2=4; z=1;        // set marginal variables and discrete z
mixGr(v1,v2,z)

// Parameters of all nc*nz components (for scaled data)
set(scf(111),'position',[200 10 600 600]);
title('Components -> down, discrete variable -> right')
for i=1:nc*nz
  subplot(nc,nz,i)
  bar(Est.Cy(i).th)
end
```

Popis programu:

// DATA

Program pracuje s reálnými daty. Je zde 14 veličin spojitých y_t a 2 diskrétní z_t. Z těch 14 spojitých jsou dvě sice diskrétní, ale s více než 10 hodnotami, takže je možno je počítat mezi spojité. V programu jsou nataženy jako d_con, d_ord a d_nom. Diskrétní veličiny jsou kódovány do jedné pomocí funkce psi2row(zz(:,i),bn). Spojité veličiny jsou škálovány tak, aby mely střední hodnotu 0 a rozptyl 1.

// MODEL INITIALIZATION

Inicializace je nejdůležitější částí každého odhadování modelu směsi distribucí. Zde je inicializace automatická, protože program pracuje s velkým množstvím veličin a s voleným počtem komponent (nc je počet komponent, nv je počet spojitých veličin a nz je počet hodnot kódované diskrétní veličiny).

Provedené škálování spojitých veličin pomáhá při inicializaci. Víme totiž, že veličiny se pohybují kolem nuly se směrodatnou odchylkou ± 1. Počáteční hodnoty parametrů můžeme tedy volit

jednoduše pomocí funkce rand(). Podle počátečních parametrů je pak pro spojité veličiny ještě třeba nastavit statistiky. To uděláme takto: pro dané počáteční parametry Θ_0 vytvoříme rozšířený regresní vektor $\Psi = \left[\Theta_0^{'}, 1\right]^{'}$ (tak, jako by hodnoty parametru byla data). Informační matici vytvoříme takto $V = \Psi\Psi'$.

Vysvětlení: *Po rozdělení informační matice dostaneme* $V_y = \Theta_0\Theta_0^{'}$, $V_{yp} = \Theta_0$ *a* $V_p = 1$. *Odhad parametrů je* $\hat{\Theta} = V_1^{-1}V_{yp} = \Theta_0$

Na zbytku inicializace už tolik nezáleží.

Pozor: Pokud by výsledky nebyly dobré, je třeba počáteční hodnoty parametrů "ušít na míru". O tom je úloha Inicializace směsi. Tady to ale bude dosti náročné, protože počet parametrů je tolik, kolik se součin počtu hodnot diskrétních proměnných krát nastavený počet komponent. Tady pro $nc = 5$ a $nz = 6$ to bude 30 parametrů (5-ti prvkových vektorů).

// ESTIMATION

Vlastní odhad běží v časové smyčce se třemi hlavními body

1. Výpočet vah w pro jednotlivé komponenty, kde hlavním prvkem jsou tzv. proximity - tj. hodnoty hustoty pravděpodobnosti příslušné komponenty s dosazenými existujícími bodovými odhady parametrů a aktuálně změřenými daty, tj. $f_c\left(y_t|\hat{\Theta}_{c;t-1}\right)$, kde $c = 1, 2, \cdots, nc$ (pro všechny komponenty).
2. Přepočet statistik, kde data se přidávají s příslušnou váhou.
3. Výpočet bodových odhadů.

Tyto body se provádějí ve dvojí variantě: pro model spojitých veličin a pro model diskrétních veličin. Model pointru je společný.

// RESULTS

S výsledky je hlavní potíž. Pracujeme ve 14-ti rozměrném prostoru, navíc ještě indexovaném 6-ti hodnotami diskrétní veličiny. Co tedy můžeme prohlížet?

Prvním ukazatelem kvality odhadu je časový průběh vah w uchovaný ve matici wt a průběh vývoje parametrů (center komponent) v matici tht.

Další, co lze sledovat, jsou dvourozměrné marginál dat - tedy z celé mnohorozměrné distribuce vybereme jen dvě veličiny a sledujeme řezy distribucí ve těchto dvou veličinách. Vybrané veličiny jsou označené $v1$ a $v2$. Dále pak musíme vybrat hodnotu diskrétní veličiny pro kterou chceme klastry zobrazit. Ta je označena z. Kreslí se původní (neškálovaná) data, obarvená pole toho, kdo které komponenty se klasifikují. Toto zobrazení a tisk příslušných souřadnic center komponent se provádí ve funkci mixGr(v1,v2,z) , která je definována v úvodu programu. Při běhu se použije s přednastavenými hodnotami a lze ji dále spouštět např. takto: close,mixGr(1,5,2), kde close zavírá předchozí graf a prohlížíme veličiny 1 a 5 pro $z = 2$.

Poslední část programu zobrazuje hodnoty všech odhadnutých parametrů (center komponent). Zde můžeme sledovat, zda se některé komponenty nepřekrývají.