ANNEX 2

The computer program of the geometrical method of dating of star configurations by their proper movement taking into account the systematic errors of the catalogue

```
{\small \tt
====================================
program perebor; \{written in Pascal under Delphi4.0\}
uses Math;
const
   nstar1 = 300; \{limit of the number of stars in the configuration \}
   pi = 3.1415926536;     \{value $\pi$\}
   deltaGM = 5; \{ scope of search gamma around $\gamma_{stat}$ in search of optimum turn
              (in minutes of arc)\}
   deltaBM = 30; \{ scope of search beta around zero in search of optimum turn (in minutes)\}
   gstepM = 1.0; \{step of search of optimum point on gamma (in minutes)\}
   bstepM = 1.0; \{step of search of optimum point on beta (in minutes)\}
   eps = 30; \{vicinity of capture for the count of stars close by their latitude(in minutes)\}
   d8 = 900000; \{maximum distance allowed from the star to the closest one of the 8 named stars \}
type
   cr1=record
      nb                                  : integer;
      a,d,va,vd,l,b,cb,sb,Mbs5,Malm       : real;
      obozn                               : string;
      end;
var
   co                            : array[1..nstar1] of cr1;
   ah,am,asec,dg,dm,ds,va,vd,lg,lm,bg,bm,e,ce,se,
   lx,clx,slx,bx,cbx,sbx,ly,cly,sly,by,cby,sby,
   e1,se1,ce1,ft,ps,mg,maxb1,maxb2,angle,cangle,sangle,
   x,y,gr,deltl,ymin,ymax,gstep,bstep,cgstep,
   sgstep,cbstep,sbstep,bmax,gamma0,beta0,dl0,dist0,
   cminmax,cc,fmax,fminmax,fx,y1,dist1,dBm,dBmm,
   deltaG,cdeltaG,sdeltaG,deltaB,cdeltaB,sdeltaB,
   cGstat,sGstat,xd1,xd2,d8rad,epsrad                          : real;
   stt,stm,stf            : array [1..nstar1,1..6] of real;
   Gstat         : array [1..30] of real; \{value $\gamma_{stat}$ calculated with statistics estimate
                                          procedure \}
   zv,zvv        : array [1..nstar1] of integer;
   id            : array [1..nstar1] of integer; \{attribute of keeping the star due the proximity
                                                  to the 8-stars kernel:
0 - strike, 1 - keep\}
   agamt,cgamt,sgamt,abett,cbett,sbett      : real;
   nb,i,j,t,t1,t2,nstar,Ngamma,Nbeta,Ng0,Nb0,ig,ib,Nstep,
```

```
    Iok,Itek,NBmm,NBm,jj,jj1,i8                                  : integer;
    f,f1,f2                 : text;
    konec                          : char;
\{*********************************************************\}
\{*                                vvod                                *\}
\{*********************************************************\}
procedure vvod;
var    i                    : integer;
       Mbs5,Malm            : real;
       ob                   : string;
begin
assign(f1,'result.txt');
rewrite(f1);
assign(f2,'sig-max.txt');
rewrite(f2);
writeln(f1,'       *** Program perebor.pas ***');
writeln('      *** Program perebor.pas ***');
assign(f,'fast.txt'); \{fast.txt - Input file with stars \}
reset(f);
\{*********** reading data **************\}
nstar:=0;
while not eof(f) do
   begin \{while\}
   nstar:= nstar+1;
   i:=nstar;
   readln(f,nb,ah,am,asec,dg,dm,ds,Mbs5,va,vd,lg,lm,bg,bm,Malm,ob);
           \{++++++ structure of the data line in file fast.txt ++++++++\}
           \{ nb - number of star in BS5,                                                    \}
           \{ ah - direct ascension (hours),                                                 \}
           \{ am - direct ascension (minutes of the hour) NO SIGN,                           \}
           \{ asec - direct ascension (seconds of the hour) NO SIGN,           \}
           \{ dg - declination (degrees),                                                    \}
           \{ dm - declination (minutes of arc), NO SIGN                                     \}
           \{ ds - declination (secondes of arc), NO SIGN                                    \}
           \{ va - speed of proper movement in the direct ascension,           \}
           \{    aligned to equator ("/year),                                                \}
           \{ vd - speed of proper movement in declination                     \}
           \{     ("/year),                                                                  \}
           \{ lg - longitude in Almagest (degrees),                            \}
           \{ lm - longitude in Almagest (minutes), NONNEGATIVE                              \}
           \{ bg - latitude in Almagest (degrees),                                           \}
           \{ bm - latitude in Almagest (minutes)    NO SIGN                                 \}
           \{ Mbs5 - magnitude (luminosity)in BS5                                            \}
           \{ Malm - magnitude (luminosity)in Almagest                                       \}
           \{ ob - modern name (definition) of the star                                      \}
   if (ah<0) then
              begin
              am:= -am;
              asec:=-asec;
              end;
    if (dg<0) then
              begin
              dm:= -dm;
              ds:=- ds;
              end;
    if (bg<0) then bm:= -bm;
   co[i].nb:=nb;
   co[i].a:=pi*(ah+am/60+asec/3600)/12;
   co[i].d:=pi*(dg+dm/60+ds/3600)/180;
   co[i].va:=va*pi/6480.0; \{translation of the speeds of proper movement: \}
   co[i].vd:=vd*pi/6480.0; \{seconds/year->radians/100years              \}
   co[i].l:=pi*(lg+lm/60)/180;
   co[i].b:=pi*(bg+bm/60)/180;
   co[i].Malm:=Malm;
   co[i].Mbs5:=Mbs5;
   co[i].obozn:=ob;
   co[i].cb:=cos(co[i].b);
```

```
    co[i].sb:=sin(co[i].b);
    if co[i].cb <> 0 then
         co[i].va:=co[i].va/co[i].cb;\{from now on the speed is NOT aligned to the equator \}
    writeln(f1,nb:4,' ',ah:4:0,' ',am:6:2,' ',
            dg:4:0,' ',dm:6:2,' ',
            lg:4:0,' ',lm:4:0,' ',bg:4:0,' ',bm:4:0,' ',
            Malm:3:1,'      ',Mbs5:3:1,'      ',ob);
    writeln(nb:4,' ',ah:4:0,' ',am:6:2,' ',
            dg:4:0,' ',dm:6:2,' ',
            lg:4:0,' ',lm:4:0,' ',bg:4:0,' ',bm:4:0,' ',
            Malm:3:1,'      ',Mbs5:3:1,'      ',ob);
    end;   \{while\}
writeln('nstar= ',nstar);
writeln(f1,'FAST.TXT:      nstar= ',nstar);
writeln(f1);
\{for i:=1 to nstar do
     writeln(f1,co[i].nb:4:0,' ',co[i].a:7:5,' ',co[i].d:7:5,
             ' ',co[i].l:7:5,' ',co[i].b:7:5);       \}
writeln('VVOD' );
end; \{vvod\}
\{************************************************\}
\{*                TURN                          *\}
\{************************************************\}
procedure turn;
    \{lx (clx, slx) - longitude(cos, sin) before the turn,
     bx (cbx, sbx) - latitude(cos, sin) before the turn,
     ly (cly, sly) - longitude(cos, sin) after the turn,
     by (cby, sby) - latitude(cos, sin) after the turn,
     angle (cangle,sangle) - angle (cos,sin) of the turn\}
var
     c,x,y     : real;
begin \{turn\}
    sby:= -slx*cbx*sangle + sbx*cangle;
    cby:= sqrt(1 - sqr(sby));
    if sby=1 then by:= pi/2
            else by:= arctan(sby/cby);
    c:= cbx*clx;
    if c = 0 then
            begin
            if cbx*cangle+slx*sbx*sangle > 0 then     ly := lx
                                                      else     ly:=lx-pi;
            if cbx = 0 then ly:= pi/2;
            end
        else \{if c is not equal zero \}
        begin
        ly:= (slx*cbx*cangle + sbx*sangle)/c;
        ly:= arctan(ly);
        if ly < 0 then ly:= ly + pi;
        \{if ly > pi then writeln('!!!!!!!!!!!');      \}
     \{---------------------------------------------\}
     \{If the star is in the circle on the sphere that has as its diameter the arc of the length of
       angle connecting the new and the old poles, then the module of the difference of its old and new
       longitude is closer to pi, than to zero. If the star is outside such a circle then the module
       difference of its longitude is closer to 0 than to pi\}
        y:=pi/2 - bx;
        x:=angle*cos(lx+pi/2); \{To facilitate calculation an estimate is used.
                                Actually: angle*cos(lx+pi/2) <= x <= angle \}
        if y>x    then
                   begin
                   if abs(abs(lx-ly)-pi)<pi/2 then ly:=ly+pi;
                   end
                else
                   begin
                   if abs(lx-ly)<pi/2 then ly:=ly+pi;
                   end;
     \{---------------------------------------------\}
        end; \{if c is not equal zero\}
```

```
    cly:= cos(ly);
    sly:= sin(ly);
    if ly > 2*pi then ly:=ly-2*pi;
    if ly < 0 then ly:=ly+2*pi;
end;   \{turn\}
\{************************************************************\}
\{*            RECALCULATION FOR MOMENT IN TIME T            *\}
\{************************************************************\}
procedure pereschet;
var i: integer;
       z,zz: real;
     \{result: stt[i,1] = l
               stt[i,2] = cos(l)
              stt[i,3] = sin(l)
              stt[i,4] = b
              stt[i,5] = cos(b)
              stt[i,6] = sin(b)
              where l,b - ecliptical coordinates of the star in epoch t
                          (taking its proper movement into account)\}
     begin \{pereschet\}
     for i:= 1 to nstar do
       begin \{for i\}
       lx := co[i].a + t1*co[i].va;
       clx:= cos(lx);
       slx:= sin(lx);
       bx := co[i].d + t1*co[i].vd;
       sbx:= sin(bx);
       cbx:= sqrt(1 - sqr(sbx));
       cangle:= ce;
       sangle:= se;
       angle:=e;
       turn;
       bx := by;
       cbx:= cby;
       sbx:= sby;
       lx:= ly - ft;
       if lx < 0 then lx:= lx + 2*pi;
       clx:= cos(lx);
       slx:= sin(lx);
       cangle:= ce1;
       sangle:= se1;
       angle:=e1;
       turn;
       stt[i,4]:= by;
       stt[i,5]:= cby;
       stt[i,6]:= sby;
       lx:= ly + ft + ps;
       if lx > 2*pi         then lx:= lx - 2*pi;
       if lx <= -2*pi then lx:= lx + 2*pi;
       if lx > 2*pi         then lx:= lx - 2*pi;
       if lx <= -2*pi then lx:= lx + 2*pi;
       stt[i,1]:= lx;
       stt[i,2]:= cos(lx);
       stt[i,3]:= sin(lx);
     \{-----------------
       zz:=mg/60;
       z:= (stt[i,1]-co[i].l)*zz;
       writeln(f1,co[i].nb:4,'      ','L= ',lx*zz:5:3,';       B= ',by*zz:5:3);
       writeln(co[i].nb:4,'         ','L= ',lx*zz:5:3,';       B= ',by*zz:5:3);
       if abs(z)> 20 then
       begin
writeln(f1,'dL=',z:10:1,'(gr); i= ',co[i].nb,' L-alm=',co[i].l*zz:6:2,
                                                    ' B-alm=',co[i].b*zz:6:2);
writeln('','dL=',z:10:1,'(gr); i= ',co[i].nb,' L-alm=',co[i].l*zz:6:2,
                                                    ' B-alm=',co[i].b*zz:6:2);
       end;
       z:= (stt[i,4]-co[i].b)*mg;
```

```
       if abs(z)> 300 then
       begin
       writeln(f1,'        ','dB= ',z:10:1,'(min);    i= ',i);
       writeln('  ','dB= ',z:10:1,'(min);    i= ',i);
       end;
       ------------------   \}
       end;   \{for i\}
end; \{pereschet\}
\{************************************************************\}
\{*  DIST (distance between points on the sphere in radians) *\}
\{************************************************************\}
function dist(L1:real;B1:real;L2:real;B2:real) : real;
   \{L1,B1 - longitude and latitude of the first point,
    L2,B2 - longitude and latitude of the second point \}
var
    X1,X2,Y1,Y2,Z1,Z2,DE,DSIN,DTAN           : real;
begin \{dist\}
        X1 := COS(B1)*COS(L1);
        Y1 := COS(B1)*SIN(L1);
        Z1 := SIN(B1);
        X2 := COS(B2)*COS(L2);
        Y2 := COS(B2)*SIN(L2);
        Z2 := SIN(B2);
        DE:=SQRT(SQR(X1-X2)+SQR(Y1-Y2)+SQR(Z1-Z2));
        DSIN:= DE/2;
        DTAN:=DSIN/SQRT(1.0-SQR(DSIN));
        Result:= 2.0*ARCTAN(DTAN);
end;\{dist\}
\{************************************************************\}
\{                      MAIN PROGRAM                          \}
\{************************************************************\}
begin \{program\}
\{*********************************\}
   vvod;  \{stars data input from file fast.txt\}
\{*********************************\}
   mg:= 180.0*60.0/pi;    \{ratio for recalculation from arc minutes into radians and reverse \}
   e:=pi*(23+27/60+8.26/3600)/180; \{angle of inclinitation of eclipticto equator for t=0\}
   se:=sin(e);
   ce:=cos(e);
   d8rad:=d8/mg;
   epsrad:=eps/mg;
   \{----------------------\}
      for i:=1 to n star do
      begin
      xd1:=10;
      for i8:=1 to 8 do  \{8 stars of the informative kernel must stand in the beginning!\}
         begin
         xd2:=dist(co[i8].a,co[i8].d,co[i].a,co[i].d);
    \{  writeln(f1,co[i].nb,'       dist (min) = ',xd2*mg:4:1);          \}
         if xd2 < xd1 then xd1:=xd2;
         end;
         xd2:=xd1*mg/60;
   \{  writeln(f1,co[i].nb,'        dist (grad) = ',xd2:4:1); \}
         if xd1 < d8rad then id[i]:=1 else id[i]:=0;
         end;
   \{----------------------\}
   gstep:=gstepM/mg;
   bstep:=bstepM/mg;
   cgstep:= cos(gstep);
   sgstep:= sin(gstep);
   cbstep:= cos(bstep);
   sbstep:= sin(bstep);
   deltaG:=deltaGM/mg;
   cdeltaG:= cos(deltaG);
   sdeltaG:= sin(deltaG);
   deltaB:=deltaBM/mg;
   cdeltaB:= cos(deltaB);
```

```
   sdeltaB:= sin(deltaB);
   Ngamma:=Trunc(deltaG/gstep); \{number of steps on gamma to one side\}
   Nbeta:= Trunc(deltaB/bstep);     \{ number of steps on beta to one side\}
   Gstat[1]:= 30.5/mg;
   Gstat[2]:= 29.5/mg;
   Gstat[3]:= 28.5/mg;
   Gstat[4]:= 27.5/mg;
   Gstat[5]:= 27.0/mg;
   Gstat[6]:= 26.0/mg;
   Gstat[7]:= 25.2/mg;
   Gstat[8]:= 24.4/mg;
   Gstat[9]:= 23.5/mg;
   Gstat[10]:= 22.6/mg;
   Gstat[11]:= 21.8/mg;
   Gstat[12]:= 21.0/mg;
   Gstat[13]:= 20.4/mg;
   Gstat[14]:= 19.5/mg;
   Gstat[15]:= 18.8/mg;
   Gstat[16]:= 18.0/mg;
   Gstat[17]:= 17.2/mg;
   Gstat[18]:= 16.4/mg;
   Gstat[19]:= 15.8/mg;
   Gstat[20]:= 15.0/mg;
   Gstat[21]:= 14.4/mg;
   Gstat[22]:= 13.8/mg;
   Gstat[23]:= 13.1/mg;
   Gstat[24]:= 12.5/mg;
   Gstat[25]:= 12.0/mg;
   Gstat[26]:= 11.5/mg;
   Gstat[27]:= 11.1/mg;
   Gstat[28]:= 10.8/mg;
   Gstat[29]:= 10.5/mg;
   Gstat[30]:= 10.2/mg;
    writeln(f2,' t          ','sigma          ','maxB','       N-in-eps');
   for t:=1 to 30 do \{time cycle with 1=century step\}
   begin \{for t\}
    \{ writeln(f1,'T = ',t:2);
   writeln(f1);  \}
   writeln('T = ',t:2);
   writeln;
   t1:=-t;
   e1:=(pi/648000.0)*(47.070559+(-0.033769+0.00005*t1)*t1)*t1;
   se1:=sin(e1);
   ce1:=cos(e1);
   ft:=(pi/180.0)*(174+52/60.0 -t1*870.0798/3600.0+t1*t1*0.024578/3600.0);
   ps:=(pi/648000.0)*(5026.872+(1.131358+0.000102*t1)*t1)*t1;
\{*************************************************************        \}
   pereschet;  \{recalculation of star coordinates in epoch t \}
\{*************************************************************        \}
   cGstat:=cos(Gstat[t]);
   sGstat:=sin(Gstat[t]);
   angle:= Gstat[t]-deltaG;
   cangle:= cdeltaG*cGstat+sdeltaG*sGstat;
   sangle:= sGstat*cdeltaG -sdeltaG*cGstat; \{ in the beginning current angle of turn on gamma
                                            is set Gstat[t]-deltaG \}
 \{cgamt,sgamt - cosinus and sinus of accumulated angle of turn on gamma\}
 \{cbett,sbett - cosinus and sinus of accumulated angle of turn on beta\}
   bmax:=1;    \{preparation for minimum rotation for maximum stellar latitude non-alignment\}
   dBmm:=1;    \{preparation for minimum rotation for medium stellar latitude non-alignment \}
   Nbmm:=0;   \{preparation for maximum number of stars by turns landing in eps' - vicinity of
               Almagest star \}
   gamma0:=0; \{preparation for optimum turn on gamma\}
   beta0:=0;  \{preparation for optimum turn on beta\}
   dl0:=0; \{preparation for spread on longitude with minimax on latitude \}
   dist0:=0;   \{preparation non-alignment on arc with minimax on latitude \}
   for ig:=-Ngamma to Ngamma do
   begin \{for ig - turn along\}
```

```
   \{ writeln('ig = ',ig); \}
   i:=1;
      while (i <= nstar) do
      begin \{while i<=nstar\}
      lx := stt[i,1];
      clx:= stt[i,2];
      slx:= stt[i,3];
      bx := stt[i,4];
      cbx:= stt[i,5];
      sbx:= stt[i,6];
      turn;
      if ly > 3.0*pi/2.0 then x:= ly-2.0*pi else x:=ly;
      stm[i,1]:= x+pi/2;
      stm[i,2]:= -sly;
      stm[i,3]:= cly;
      stm[i,4]:= by;
      stm[i,5]:= cby;
      stm[i,6]:= sby;
      i:=i+1;
      end; \{while i<=nstar\}
      agamt:=angle;
      cgamt:=cangle;
      sgamt:=sangle;      \{record the accumulated angle on gamma,
                           to use after a completed cycle of across turns\}
      angle:= -deltaB;
      cangle:= cdeltaB;
sangle:= -sdeltaB; \{in the beginning of the cycle of turns set the angle equal to -deltaB\}
      for ib:= -Nbeta to Nbeta do
      begin \{for ib - across turn\}
         i:=1;
         maxb1:=0.0;
         ymin:=7.0;
         ymax:=-7.0;
         dBm:=0;
         Nbm :=0;
         while (i <= nstar)       do
         begin \{while i<=nstar\}
         lx := stm[i,1];
         clx:= stm[i,2];
         slx:= stm[i,3];
         bx := stm[i,4];
         cbx:= stm[i,5];
         sbx:= stm[i,6];
         turn;
         stf[i,2]:=by;
         stf[i,3]:=cby;
         if ly < pi/2 then y:=ly + 2*pi else y:= ly;
         stf[i,1]:=ly - pi/2;
         y:= y - pi/2 - co[i].l;
         if y < -pi then y:=y+2*pi
                         else if y> pi then y:= y-2*pi;
         if y < -pi then y:= y+2*pi
                         else if y>pi then y:=y-2*pi;
         y1:=y*cby;
         if abs(y1)>0.5 then
            begin
            writeln(f1,'dL*cosB=',y1:10:5,'(rad); N(BS5)=',co[i].nb:4);
            writeln('dL*cosB=',y1:10:5,'(rad); N(BS5)=',co[i].nb:4);
            writeln(f1,'cosB=',cby:10:5);
            writeln('cosB=',cby:10:5);
            x:=mg/60;
            writeln(f1,'by=',by*x:9:2,'    ly=',ly*x:9:2);
            writeln('by=',by*x:9:2,'       ly=',ly*x:9:2);
            writeln(f1,'L-alm=',co[i].l*x:9:2,'     B-alm=',co[i].b*x:9:2);
            writeln('L-alm=',co[i].l*x:9:2,'        B-alm=',co[i].b*x:9:2);
            readln(konec);
            end;
```

```
          stf[i,4]:=y;
          if y < ymin then ymin:= y;
          if y > ymax then ymax:= y;
 \{----1-st case: kernel of 8 stars is always kept ------\}
          maxb2:= abs(by - co[i].b);
          if (id[i]=1) and (maxb2 < epsrad) then
                                    begin
                                    dBm:=dBm+sqr(maxb2);
                                    NBm:=NBm+1;
                                    zv[NBm]:=i;
                                    end;
          if maxb2 > maxb1 then
               begin
               maxb1:= maxb2;
               Itek:=i
               end;
          i:= i+1;
          end; \{while i<=nstar\}
          dBm:=sqrt(dBm/NBm);
\{-----2-nd case: the kernel is not separated when kept -----\}
     \{     maxb2:= abs(by - co[i].b);
          dBm:=dBm+sqr(maxb2);
          if maxb2*mg<eps then NBm:=NBm+1;
          if maxb2 > maxb1 then
               begin
               maxb1:= maxb2;
               Itek:=i
               end;
          i:= i+1;
          end; \{while i<=nstar\}
     \{    dBm:=sqrt(dBm/nstar);                      \}
     \{-----------end of 2 cases--------\}
          \{=====================================================
          deltL:=(ymin+ymax)/2; \}\{- previous calculation of optimal twist \}
          \{Improved calculation of optimal twist on longitude:
          Look maximum on C minimum on i of value
          cos(B)*[abs(dL(i) - C],
          where B - maximum of Almagest latitude and calculated altitude,
          dL(i) - difference between calculated and Almagest longitude for i star.
          Resulting C produces the value of optimal twist deltL       \}
          x:=0.01;
          y:=ymax-ymin;
          Nstep:=Trunc(y/x);
          cminmax:=ymin;
          cc:=ymin;
          fminmax:=7;
          for i:=1 to Nstep do
              begin
              cc:=cc+x;
              fmax:=0;
              for j:=1 to nstar do
              begin
              fx:=Min(stf[j,3],co[j].cb);
              fx:=fx*abs(stf[j,4]-cc);
              if fx > fmax then fmax:=fx;
              end;
              if fmax < fminmax then
                                        begin
                                        fminmax:=fmax;
                                        deltL:=cc;
                                        end;
              end;
          \{=====================================================\}
          \{if (maxb1 < bmax) then \}
          if (dBm < dBmm) then     \{ <- one of three versions is chosen \}
           \{ if (NBm > NBmm) then \}
          begin
```

```
          bmax:=maxb1;
          Iok:=Itek;
          Ng0:=ig;
          Nb0:=ib;
          dBmm:=dBm;
          NBmm:=NBm;
                  for jj:=1 to NBm do
                  begin
                  zvv[jj]:=zv[jj];
                  end;
          gr:=0.0;
          for i:=1 to nstar do
             begin
             x:= (stf[i,4]-deltL)*Min(stf[i,3],co[i].cb);
             x:= sqr(x);
             y:=sqr(stf[i,2] - co[i].b);
             x:=sqrt(x+y);
             if x > gr then gr:=x;
             end;
          dist0:=gr;
          end; \{if maxb1<bmax, if dBm < dBmm or if (NBm > NBmm)\}
          abett:=angle;
          cbett:=cangle;
          sbett:=sangle;
          angle:=angle+bstep;
          cangle:= cbett*cbstep - sbett*sbstep;
          sangle:= sbett*cbstep + cbett*sbstep;
       end;    \{for ib -  turn across\}
       angle:= agamt+gstep;
       cangle:= cgamt*cgstep - sgamt*sgstep;
       sangle:= sgamt*cgstep + cgamt*sgstep;
       end; \{for ig - turn along\}
   \{*************************************************************\}
                  \{save and print file \}
   gamma0:= (Ng0*gstep+Gstat[t])*mg;
   beta0:=Nb0*bstep*mg;
   bmax:=bmax*mg;
   dist0:=dist0*mg;
   dBmm:=dBmm*mg;
   t2:=1900-t*100;
   writeln(f1,'================================================');
   writeln(f1,'Max distance to inf. kernel allowed = ',d8,'(min)');
   writeln(f1,'   ',t2:2,'           ',bmax:4:1,' (',co[Iok].nb:4,
   ')     ',gamma0:4:1,'             ',beta0:4:1,'              ',dist0:4:1);
 writeln(f1,'     sigma=',dBmm:4:1,' Nstars= ',NBmm,'(',eps,'min close)');
   for jj:=1 to NBmm do
        begin
        jj1:=zvv[jj];
        writeln(f1,co[jj1].nb,'      ',co[jj1].obozn);
        end;
   writeln(f2,t2:2,'       ',dBmm:4:1,'     ',bmax:4:1,'      ',NBmm);
   writeln('*** T = ',t2:2,'       ***');
   writeln('Max distance to inf. kernel allowed = ',d8,'(min)');
   writeln('dBmax=',bmax:4:1,' i=',co[Iok].nb:4,' gamma=',gamma0:4:1,
            '  beta= ',beta0:4:1,'         dist=',dist0:4:1);
 writeln('sigma=',dBmm:4:1,'; Nstars (',eps,' min close)=',NBmm);
   end;   \{for t\}
close(f1);
close(f2);
writeln('Enter any character');
readln(konec);
end.
==========================================
\vspace{1cm}
```

EXAMPLES OF INPUT FILES FOR PERESCHET PROGRAM (FAST.TXT)

```
\vspace{0.5cm}
Contents of columns in input data file FAST.TXT for PERESCHET program:
1 column - number of star in catalogues BS4, BS5;
2 column - direct ascension RA 1900 in BS5: hours;
3 column - direct ascension RA 1900 in BS5: minutes;
4 column - direct ascension RA 1900 in BS5: seconds;
5 column - declination DEC 1900 in BS5: degrees;
6 column - declination DEC 1900 in BS5: minutes;
7 column - declination DEC 1900 in BS5: seconds;
8 column - star magnitude in BS5;
9 column - speed of proper movement in RA1900, aligned to equator (in BS4);
10 column - speed of proper movement in DEC1900, aligned to equator (in BS4);
11 column - longitude in Almagest;
12 column - latitude in Almagest;
13 column - brightness in Almagest;
14 column - modern name of star in BS5.
\vspace{0.7cm}
```

1. Data file: 8 stars of informative kernel of Almagest.

```
\vspace{0.4cm}
}
{\footnotesize \tt
5340  14  11  06.0  +19  42  11  -0.04  -1.098  -1.999  177  00  +31  30 1.~ 16Alp Boo
1708  05  09  18.0  +45  53  47  ~0.08  +0.080  -0.423  ~55  00  +22  30 1.~ 13Alp Aur
3982  10  03  02.8  +12  27  22  ~1.35  -0.249  +0.003  122  30  ~~0  10 1.~ 32Alp Leo
2943  07  34  04.0  +05  28  53  ~0.38  -0.706  -1.029  ~89  10  -16  10 1.~ 10Alp CMi
5056  13  19  55.4  -10  38  22  ~0.98  -0.043  -0.033  176  40  ~-2  ~0 1.~ 67Alp Vir
6134  16  23  16.4  -26  12  36  ~0.96  -0.007  -0.023  222  40  ~-4  ~0 2.~ 21Alp Sco
7001  18  33  33.1  +38  41  26  ~0.03  +0.200  +0.285  257  20  ~62  ~0 1.~ ~3Alp Lyr
3449  08  37  29.9  +21  49  42  ~4.66  -0.103  -0.043  100  20  ~~2  40 3.7 43Gam Cnc
}
{\small \tt
\vspace{0.7cm}
```

2. Data file: named stars from A, ZodA, B, ZodB, M, are rapid (>=0.1"/year in RA1900 or DEC1900) and isolated ones among stars of comparable brightness, resulting in their unambiguous identity in Almagest catalogue. The 8 stars Almagest informative kernel is added.

```
}
\vspace{0.4cm}
{\footnotesize \tt
5340 14 11 06.0 +19 42 11  -0.04 -1.098 -1.999 177 00 +31 30 1.~  16Alp Boo
1708 05 09 18.0 +45 53 47  ~0.08 +0.080 -0.423 ~55 00 +22 30 1.~  13Alp Aur
3982 10 03 02.8 +12 27 22  ~1.35 -0.249 +0.003 122 30 ~~0 10 1.~  32Alp Leo
2943 07 34 04.0 +05 28 53  ~0.38 -0.706 -1.029 ~89 10 -16 10 1.~  10Alp CMi
5056 13 19 55.4 -10 38 22  ~0.98 -0.043 -0.033 176 40 ~-2 ~0 1.~  67Alp Vir
6134 16 23 16.4 -26 12 36  ~0.96 -0.007 -0.023 222 40 ~-4 ~0 2.~  21Alp Sco
7001 18 33 33.1 +38 41 26  ~0.03 +0.200 +0.285 257 20 ~62 ~0 1.~  ~3Alp Lyr
3449 08 37 29.9 +21 49 42  ~4.66 -0.103 -0.043 100 20 ~~2 40 3.7  43Gam Cnc
~~15 00 03 13.0 +28 32 18  ~2.06 +0.137 -0.158 347 50 +26 00 2.3  21Alp And
~~21 00 03 50.2 +58 35 54  ~2.27 +0.526 -0.177 ~~7 50 +51 40 3.~  11Bet Cas
~219 00 43 03.0 +57 17 06  ~3.44 +1.101 -0.521 ~13 00 +47 50 4.~  24Eta Cas
~337 01 04 07.8 +35 05 26  ~2.06 +0.179 -0.109 ~~3 50 +26 20 3.~  43Bet And
~403 01 19 16.1 +59 42 56  ~2.68 +0.300 -0.045 ~20 40 +45 30 3.~  37Del Cas
~544 01 47 22.7 +29 05 30  ~3.41 +0.010 -0.229 ~11 00 +16 30 3.~  ~2Alp Tri
~545 01 48 02.4 +18 48 21  ~4.83 +0.078 -0.108 ~~6 40 ~+7 20 3.3  5Gam1Ari
~553 01 49 06.8 +20 19 09  ~2.64 +0.097 -0.108 ~~7 40 ~+8 20 3.~  ~6Bet Ari
~941 03 02 44.8 +44 28 43  ~3.80 +0.178 -0.153 ~30 30 +27 00 4.~  27Kap Per
~951 03 05 54.5 +19 20 55  ~4.35 +0.151 -0.007 ~23 50 ~+1 40 4.~  57Del Ari
```

```
1346 04 14 06.0 +15 23 11  ~3.65 +0.116 -0.024 ~39 00 ~-5 45 3.3  54Gam Tau
1409 04 22 46.5 +18 57 31  ~3.53 +0.108 -0.036 ~41 50 ~-3 00 3.3  74Eps Tau
1457 04 30 10.9 +16 18 30  ~0.85 +0.065 -0.189 ~42 40 ~-5 10 1.~  87Alp Tau
1791 05 19 58.1 +28 31 23  ~1.65 +0.025 -0.175 ~55 40 ~+5 00 3.~  112Bet Tau
2821 07 19 30.9 +27 59 49  ~3.79 -0.121 -0.088 ~82 00 ~+5 30 4.~  60Iot Gem
2990 07 39 11.8 +28 16 04  ~1.14 -0.627 -0.051 ~86 40 ~+6 15 2.~  78Bet Gem
3323 08 21 57.5 +61 03 09  ~3.36 -0.131 -0.110 ~85 20 +39 50 4.~  1Omi UMa
3461 08 39 00.1 +18 31 19  ~3.94 -0.017 -0.233 101 20 ~-0 10 3.7  47Del Cnc
3569 08 52 21.8 +48 26 04  ~3.14 -0.443 -0.235 ~95 30 +29 20 3.~  ~9Iot UMa
3852 09 35 48.8 +10 20 50  ~3.52 -0.143 -0.041 117 20 ~-4 10 4.~  14Omi Leo
3905 09 47 04.6 +26 28 41  ~3.88 -0.215 -0.060 114 20 +12 00 3.~  24Mu Leo
4033 10 11 04.0 +43 24 50  ~3.45 -0.165 -0.043 112 40 +29 20 3.~  33Lam UMa
4301 10 57 33.6 +62 17 27  ~1.79 -0.118 -0.071 107 40 +49 00 2.~  50Alp UMa
4357 11 08 47.4 +21 04 18  ~2.56 +0.143 -0.135 134 10 +13 40 2.3  68Del Leo
4534 11 43 57.5 +15 07 52  ~2.14 -0.497 -0.119 144 30 +11 50 1.3  94Bet Leo
4660 12 10 28.7 +57 35 18  ~3.31 +0.102 +0.004 123 10 +51 00 3.~  69Del UMa
4785 12 28 59.6 +41 54 03  ~4.26 -0.707 +0.288 140 10 +41 20 5.~  ~8Bet CVn
4825 12 36 35.5 -00 54 03  ~3.68 -0.568 +0.008 163 10 ~+2 50 3.~  29Gam Vir
4905 12 49 37.8 +56 30 09  ~1.77 +0.109 -0.010 132 10 +53 30 2.~  77Eps UMa
5107 13 29 35.8 -00 05 05  ~3.37 -0.286 +0.036 174 50 ~+8 40 3.~  79Zet Vir
5191 13 43 36.0 +49 48 45  ~1.86 -0.124 -0.014 149 50 +54 00 2.~  85Eta UMa
5235 13 49 55.3 +18 53 56  ~2.68 -0.064 -0.363 171 20 +28 00 3.~  ~8Eta Boo
5350 14 12 37.4 +51 49 42  ~4.75 -0.154 +0.088 154 10 +58 20 5.~  21Iot Boo
5404 14 21 47.5 +52 18 47  ~4.05 -0.242 -0.400 155 20 +60 10 5.~  23The Boo
5435 14 28 03.0 +38 44 44  ~3.03 -0.116 +0.149 169 40 +49 00 3.~  27Gam Boo
5487 14 37 47.3 -05 13 25  ~3.88 +0.105 -0.321 192 40 ~+9 50 4.~  107Mu Vir
5531 14 45 20.7 -15 37 34  ~2.75 -0.108 -0.071 198 00 ~~0 40 2.~  ~9Alp2Lib
5747 15 23 42.3 +29 27 01  ~3.68 -0.179 +0.083 191 40 +46 30 3.7  ~3Bet CrB
5793 15 30 27.2 +27 03 04  ~2.23 +0.120 -0.091 194 40 +44 30 1.7  ~5Alp CrB
5854 15 39 20.5 +06 44 25  ~2.65 +0.136 +0.044 204 20 +25 20 3.~  24Alp Ser
6056 16 09 06.2 -03 26 13  ~2.74 -0.048 -0.145 215 00 +17 00 3.~  ~1Del Oph
6241 16 43 41.1 -34 06 42  ~2.29 -0.610 -0.255 228 30 -11 00 3.~  26Eps Sco
6410 17 10 55.4 +24 57 25  ~3.14 -0.023 -0.157 226 40 +48 00 3.~  65Del Her
6556 17 30 17.5 +12 37 58  ~2.08 +0.117 -0.227 234 50 +36 00 2.7  55Alp Oph
6603 17 38 31.9 +04 36 32  ~2.77 -0.042 +0.159 238 00 +27 15 3.7  60Bet Oph
6879 18 17 32.0 -34 25 55  ~1.85 -0.032 -0.125 248 00 -10 50 3.~  20Eps Sgr
7557 19 45 54.2 +08 36 15  ~0.77 +0.537 +0.387 273 50 +29 10 1.7  53Alp Aql
7602 19 50 24.0 +06 09 25  ~3.71 +0.048 -0.482 274 50 +27 10 3.~  60Bet Aql
7882 20 32 51.5 +14 14 50  ~3.63 +0.112 -0.031 288 30 +32 00 3.3  ~6Bet Del
7949 20 42 09.8 +33 35 44  ~2.46 +0.355 +0.329 300 50 +49 30 3.~  53Eps Cyg
8162 21 16 11.5 +62 09 43  ~2.44 +0.150 +0.052 346 40 +69 00 3.~  ~5Alp Cep
8264 21 32 25.7 -08 18 10  ~4.69 +0.113 -0.023 297 20 ~+6 15 5.~  23Xi Aqr
8278 21 34 33.1 -17 06 51  ~3.68 +0.188 -0.022 294 50 ~-2 10 3.~  40Gam Cap
8322 21 41 31.3 -16 34 52  ~2.87 +0.262 -0.294 296 20 ~-2 00 3.~  49Del Cap
8417 22 00 53.7 +64 08 26  ~4.29 +0.208 +0.089 358 30 +65 30 5.~  17Xi Cep
8499 22 11 33.4 -08 16 53  ~4.16 +0.117 -0.019 306 10 ~+3 00 4.~  43The Aqr
8518 22 16 29.5 -01 53 29  ~3.84 +0.129 +0.012 309 30 ~+8 45 3.~  48Gam Aqr
8684 22 45 10.5 +24 04 25  ~3.48 +0.148 -0.036 327 00 +29 30 4.~  48Mu Peg
8775 22 58 55.5 +27 32 25  ~2.42 +0.188 +0.142 332 10 +31 00 2.3  53Bet Peg
8974 23 35 14.3 +77 04 27  ~3.21 -0.065 +0.156 ~33 00 +64 15 4.~  35Gam Cep
}
```