GAUSS Code Documentation:

The Age-Profile of Mobility Measures: An application to earnings in West Germany

by

Mark Trede
Seminar für Wirtschafts- und Sozialstatistik
Universität zu Köln
50923 Köln
Germany
Tel: +49-221-4702283
Fax: +49-221-4705074
E-mail: trede@wiso.uni-koeln.de

July 1997

1 Introduction

This paper describes the GAUSS procedures and programs used for estimating age-profiles of earnings mobility. Most of the GAUSS code is documented in a unified way. The comments at the beginning of each procedure give a brief explanation of (a) what the procedure is supposed to do, (b) which inputs are required, (c) what the output looks like. Most procedures are therefore self-explaining, at least if seen in connection with the journal article.

Some of the programs are rather general and can be applied for various purposes while other programs and procedures are very specific. This holds in particular for the bootstrap programs. Hopefully this documentation helps adapt the programs and procedures to other situations.

The procedures and programs listed in this paper are sometimes slightly different from the computer readable form in the archive. This is only to make the text more compact. The proper code is identical.

If there are any problems please do not hesitate to contact me.

2 Kernel Density Estimation for Inequality Reduction Mobility Measures

The next subsection is concerned with the Weighted Average of Rounded Points (WARP) algorithm. Subsection 2.2 describes the point estimation of the mobility age profile and subsection 2.3 deals with interval estimation.

2.1 The WARP-Algorithm

The WARP algorithm yields rapid approximations of kernel density estimates. The algorithm for one dimensional densities is explained in depth in Härdle (1991). Two dimensional densities may, of course, also be approximated by WARP. This is done by the procedure `warpnrm2.g` (WARP with normal (Gaussian) kernel for 2 dimensions).

The procedure `warpnrm2.g` closely follows the WARP approach as described in the appendix of the journal article. Some comments not included in the computer readable form of this procedure are given below. The inputs `v1` and `v2` can be used to restrict the area for which the density is approximated (in other words, you can freely choose the grid of bins). However, the probability mass under the approximated density function is 1 even if the relevant area is restricted; any restriction therefore amounts to conditioning.

There are two methods of finding the density approximation if the counts and the weights are already given. The first one considers each bin separately and looks at its neighbouring bins. If the neighbouring bins are not empty the density of the current bin is increased according to the distance of the neighbour (the weight) and according to the number of observations in that bin. The second method works the other way round. Each non-empty bin "spreads out" probability mass to its neighbours. The more observations there are in the current bin the more the neighbouring densities are increased. Of course, both methods arrive at the same result. However, the second approach seems to be preferable in terms of computer time. It is taken in this procedure.

```
/* Two-dimensional WARPing of a kernel density estimation using a
   multiplicative Gaussian kernel
    Inputs:  x   (n,2)-matrix of data of the two variables
             h   (2,1)-vector; bandwidths (same as in ordinary kernel
                 density estimation)
             M   (2,1)-vector; binning parameter
             v1  (2,1)-vector; upper and lower boundary of pertinent range
                 for the first variable (first column of x)
                 OR scalar; then v1=minc(x)-4*h[1]|maxc(x)+4*h[1]
             v2  same as v1 but for second variable
    Outputs:     (p,1)-vector of evaluation points (x-coordinates)
                 (q,1)-vector of evaluation points (y-coordinates)
                 (p,q)-matrix of estimated densities */

proc(3)=warpnrm2(x,h,M,v1,v2);
  local a,b,c,d,f,i,j,j1,j2,ja1,ja2,je1,je2,k1,k2,l1,l2,s1,s2,t,w;
  a=minc(x)-8*h;                    @ lower limits of the margin @
  b=maxc(x)+8*h;                    @ upper limits of the margin @
  d=h./M;                           @ parameters d @
  s1=seqa(a[1],d[1],ceil((b[1]-a[1])/d[1])+1); @ x-coordinates of
                                                 mid points @
  s2=seqa(a[2],d[2],ceil((b[2]-a[2])/d[2])+1); @ y-coordinates of
                                                 mid points @
  t=rows(s1)|rows(s2);
  c=zeros(t[1],t[2]);               @ initiate bin counts variable @
  f=c;                              @ initiate density approximations
                                      variable @

  if rows(v1)==1;                   @ define relevant range of x-coord. @
    v1=(minc(x[.,1])-4*h[1]-1e-9)|(maxc(x[.,1])+4*h[1]);
  else;
    v1[1]=v1[1]-1e-9;               @ -1e-9 is needed for counting below @
  endif;
  if rows(v2)==1;                   @ define relevant range of y-coord. @
    v2=(minc(x[.,2])-4*h[2]-1e-9)|(maxc(x[.,2])+4*h[2]);
  else;
    v2[1]=v2[1]-1e-9;
  endif;

  i=1;
  do until i==t[2];                 @ this loop performs the counting @
    i=i+1;                          @ of observations within each bin @
    j=indexcat(x[.,2],s2[i-1:i]);
    if scalmiss(j);
      c[.,i]=zeros(t[1],1);
    else;
      c[.,i]=counts(x[j,1],s1);
    endif;
  endo;
```

```
  l1=pdfn(seqa(-4*M[1],1,8*M[1]+1)./M[1]);       @ these rows define    @
  l2=pdfn(seqa(-4*M[2],1,8*M[2]+1)./M[2]);       @ the grid of weights, @
  w=(l1./sumc(l1).*M[1]).*(l2./sumc(l2).*M[2])'; @ stored in w          @
  k1=seqa(-4*M[1],1,8*M[1]+1);      @ k1 and k2 are vectors of @
  k2=seqa(-4*M[2],1,8*M[2]+1);      @ indices to be used below @
  j1=indexcat(s1,v1);
  j2=indexcat(s2,v2);
  ja1=j1[1];                        @ indices of first mid points @
  ja2=j2[1];                        @ to be evaluated             @
  je1=j1[rows(j1)];                 @ indices of last mid points  @
  je2=j2[rows(j2)];                 @ to be evaluated             @

  i=ja1-4*M[1]-1;                   @ the double loop creates the       @
  do until i==je1+4*M[1];           @ matrix of density approximations, @
    i=i+1;                          @ if a bin is not empty the neigh-  @
    j=ja2-4*M[2]-1;                 @ bouring density estimates are     @
    do until j==je2+4*M[2];         @ increased according to the bins   @
      j=j+1;                        @ weights.                          @
      if c[i,j].>0;
        f[k1+i,k2'+j]=f[k1+i,k2'+j]+c[i,j].*w;
      endif;
    endo;
  endo;

  f=f[ja1:je1,ja2:je2]/(rows(x)*h[1]*h[2]); @ normalization of f @
  retp(s1[ja1:je1],s2[ja2:je2],f);
endp;
```

Example 1 *Generate $n = 1000$ realizations from the bivariate standard normal distribution. Compute the two dimensional kernel density estimate by WARP with bandwidths $h_1 = 0.2$, $h_2 = 0.4$ and $M_1 = 4$, $M_2 = 6$ and plot it:*

```
x=rndn(1000,2);
h=0.2|0.4;
M=4|6;
{s1,s2,f}=warpnrm2(x,h,M,0,0);
surface(s1',s2,f');
```

Example 2 *Use the same set of parameters as in the previous example. Perform the WARP algorithm for other values of n and consider the impact on computing time:*

```
h=0.2|0.4; M=4|6;
x=rndn(n,2);
h0=hsec;
{s1,s2,f}=warpnrm2(x,h,M,0,0);
etstr(hsec-h0);
```

The required computer time will roughly increase proportionally with n. Playing around with the parameters M reveals their strong impact on computing time. Restricting the range also has an impact (just change the set of arguments of the `warpnrm2` *procedure to, say,* `warpnrm2(x,h,M,-3|3,-3|3)`*).*

2.2 Point Estimation of the Mobility Age Profile

There are three procedures for estimating mobility age profiles from the approximated joint density functions.[1] These procedures are `c_mst.g`, `c_msa.g` and `c_msv.g` (the `c` stands for "conditional", the `ms` for "mobility index by Shorrocks", `t`, `a` and `v` for Theil's index, Atkinson's index, and coefficient of variation). All procedures need the same inputs — apart, of course, from the parameters specific to the generalized Theil measures and Atkinson's measures. The required inputs are the points of density evaluation in both periods and for aggregate income as well as matrices containing the two dimensional density approximations for (a) income in the first period and age, (b) income in the second period and age, (c) aggregate income and age. Note that the three matrices must have the same number of columns. However, since the same vector of ages is included in every WARP approximation the matrices will automatically have the same number of columns.

Some comments not included in the archive version of the procedures are given in the listing below.

```
/* Estimating the inequality reduction mobility measure based on
   Theil's generalized inequality measures conditional on age using
   (Gaussian) WARPing density estimation
   Inputs:  l   (p,1)-vector of parameters for the Theil measure
            sy1 (n1,1)-vector, points of density evaluation in the
                first period
            sy2 (n2,1)-vector, same for second period
            sya (na,1)-vector, same for aggregate income
            f1  (n1,k)-matrix, (unconditional) densities at incomes sy1
                and various ages
            f2  (n2,k)-matrix, same for second period
            fa  (na,k)-matrix, same for aggregate income
   Output:      (k,p)-maxtrix of conditional mobility measures,
                one row per age, one column per parameter
   Remarks:     All the inputs except the vector l are returns of the
                WARPing algorithm "warpnrm2.g". If the evaluation points
                are negative they are deleted. */

proc c_mst(l,sy1,sy2,sya,f1,f2,fa);
  local d1,d2,da,i,m1,m2,ma,t1,t2,ta,z;
  d1=sy1[2]-sy1[1];          @                   @
  d2=sy2[2]-sy2[1];          @ recompute bin widths @
  da=sya[2]-sya[1];          @                   @

  f1=f1./(d1.*sumc(f1)');    @                             @
  f2=f2./(d2.*sumc(f2)');    @ compute the conditional densities @
  fa=fa./(da.*sumc(fa)');    @                             @

  sy1=sy1+d1/2;              @ shift the mid points of the bins      @
  sy2=sy2+d2/2;              @ (this is not really necessary but may @
  sya=sya+da/2;              @ improve accuracy of approximations)   @
```

[1]Of course, the same approach might be used to rapidly approximate inequality age profiles (see e.g. Pudney, 1993). The procedures are constructed in the same way as those discussed in this paper.

```
  f1=delif(f1,sy1.<=0); sy1=delif(sy1,sy1.<=0); @                         @
  f2=delif(f2,sy2.<=0); sy2=delif(sy2,sy2.<=0); @ delete negative values @
  fa=delif(fa,sya.<=0); sya=delif(sya,sya.<=0); @                         @

  m1=sumc(sy1.*f1.*d1);       @                                  @
  m2=sumc(sy2.*f2.*d2);       @ calculate the conditional means @
  ma=sumc(sya.*fa.*da);       @ (by numerical integration)      @
  z=zeros(cols(f1),rows(l));

  i=0;
  do until i==rows(l);        @ loop over all parameters l      @
    i=i+1;

    t1=sumc(((sy1./m1')^(l[i]+1)).*f1.*d1)-1; @ compute the Theil     @
    t2=sumc(((sy2./m2')^(l[i]+1)).*f2.*d2)-1; @ measures by numerical @
    ta=sumc(((sya./ma')^(l[i]+1)).*fa.*da)-1; @ integration           @

    z[.,i]=1-(ma.*ta)./(m1.*t1+m2.*t2);

  endo;
  retp(z);
endp;
```

The following two procedures are structured analogously to `c_mst.g`. Thus, the comments are omitted. The procedure `c_msa.g` computes mobility age profiles for mobility measures based on Atkinson's inequality measures.

```
/* Estimating the inequality reduction mobility measure based on
   Atkinson's inequality measures conditional on age using (Gaussian)
   WARPing density estimation

   Inputs:  e   (p,1)-vector of parameters for the Atkinson measure
            sy1 (n1,1)-vector, points of density evaluation in the
                first period
            sy2 (n2,1)-vector, same for second period
            sya (na,1)-vector, same for aggregate income
            f1  (n1,k)-matrix, (unconditional) densities at incomes sy1
                and various ages
            f2  (n2,k)-matrix, same for second period
            fa  (na,k)-matrix, same for aggregate income
   Output:      (k,p)-maxtrix of conditional mobility measures,
                one row per age, one column per parameter
   Remarks:     All the inputs except the vector e are returns of the
                WARPing algorithm "warpnrm2.g". If the evaluation points
                are negative they are deleted. */

proc c_msa(e,sy1,sy2,sya,f1,f2,fa);
  local a1,a2,aa,d1,d2,da,i,m1,m2,ma,z;
  d1=sy1[2]-sy1[1];
  d2=sy2[2]-sy2[1];
```

```
  da=sya[2]-sya[1];
  f1=f1./(d1.*sumc(f1)');
  f2=f2./(d2.*sumc(f2)');
  fa=fa./(da.*sumc(fa)');
  sy1=sy1+d1/2;
  sy2=sy2+d2/2;
  sya=sya+da/2;
  f1=delif(f1,sy1.<=0); sy1=delif(sy1,sy1.<=0);
  f2=delif(f2,sy2.<=0); sy2=delif(sy2,sy2.<=0);
  fa=delif(fa,sya.<=0); sya=delif(sya,sya.<=0);
  m1=sumc(sy1.*f1.*d1);
  m2=sumc(sy2.*f2.*d2);
  ma=sumc(sya.*fa.*da);
  z=zeros(cols(f1),rows(e));
  i=0;
  do until i==rows(e);
    i=i+1;
    a1=1-sumc(((sy1./m1')^(1-e[i])).*f1.*d1)^(1/(1-e[i]));
    a2=1-sumc(((sy2./m2')^(1-e[i])).*f2.*d2)^(1/(1-e[i]));
    aa=1-sumc(((sya./ma')^(1-e[i])).*fa.*da)^(1/(1-e[i]));
    z[.,i]=1-(ma.*aa)./(m1.*a1+m2.*a2);
  endo;
  retp(z);
endp;
```

The procedure `c_msv.g` computes mobility age profiles for mobility measures based on the coefficient of variation.

```
/* Estimating the mobility based on the coefficient of variation
   conditional on age using (Gaussian) WARPing density estimation

   Inputs:  sy1 (n1,1)-vector, points of density evaluation in the
            first period
            sy2 (n2,1)-vector, same for second period
            sya (na,1)-vector, same for aggregate income
            f1  (n1,k)-matrix, (unconditional) densities at incomes sy1
                and various ages
            f2  (n2,k)-matrix, same for second period
            fa  (na,k)-matrix, same for aggregate income
   Output:      (k,1)-maxtrix of conditional mobility measures.
   Remarks:     All the inputs are returns of the WARPing algorithm
                "warpnrm2.g". Negative values are deleted. */

proc c_msv(sy1,sy2,sya,f1,f2,fa);
  local d1,d2,da,m1,m2,ma,v1,v2,va;
  d1=sy1[2]-sy1[1];
  d2=sy2[2]-sy2[1];
  da=sya[2]-sya[1];
  f1=f1./(d1.*sumc(f1)');
  f2=f2./(d2.*sumc(f2)');
```

```
  fa=fa./(da.*sumc(fa)');
  sy1=sy1+d1/2;
  sy2=sy2+d2/2;
  sya=sya+da/2;
  m1=sumc(sy1.*f1.*d1);
  m2=sumc(sy2.*f2.*d2);
  ma=sumc(sya.*fa.*da);
  v1=sumc(((sy1-m1')^2).*f1.*d1);
  v2=sumc(((sy2-m2')^2).*f2.*d2);
  va=sumc(((sya-ma')^2).*fa.*da);
  retp(1-sqrt(va)./(sqrt(v1)+sqrt(v2)));
endp;
```

Example 3 *Let* `y` *denote a* $(2000, 2)$*-matrix of incomes in the first and second period for 2000 persons. Let* `a` *be the* $(2000, 1)$ *vector of ages of the 2000 persons. The bandwidths of the Gaussian kernels are, say,* $h_1 = h_2 = 2000$ *and* $h_a = 3$*. Notice that due to the Gaussian kernel the bandwidth of the aggregate income,* h_A *has to be* $h_A = \sqrt{h_1^2 + h_2^2}$ *or, if* $h_1 = h_2$*,* $h_A = \sqrt{2}h_1$*. The WARP parameters are* $M_1 = M_2 = 5$ *and* $M_a = 6$*. The WARP parameter for aggregate income is arbitrarily set to* $M_A = M_1 = 5$*. The mobility age profile of the mobility measure based on the coefficient of variation is found in the following way:*

```
{sy1,sa1,f1}=warpnrm2(y[.,1]~a,2000|3,5|6,0,0);
{sy2,sa2,f2}=warpnrm2(y[.,2]~a,2000|3,5|6,0,0);
{sya,saa,fa}=warpnrm2((y[.,1]+y[.,2])~a,(sqrt(2)*2000)|3,5|6,0,0);
map=c_msv(sy1,sy2,sya,f1,f2,fa);
```

By construction, `sa1`*,* `sa2` *and* `saa` *are identical. The mobility age profile (*`map`*) can be plotted by typing* `xy(sa1,map)`*.*

2.3 Interval Estimation of the Mobility Age Profile

Interval estimation of the mobility age profile relies on bootstrap methods. The observations (consisting of income in both periods and age) are resampled a large number of times. Each time a new mobility age profile is computed as described above. Finally, the bundle of profiles is analyzed, either to find an area in which a high proportion (say 95 per cent) of the profiles is entirely included (confidence bands), or to find pointwise confidence intervals.

It is important to note that the computational effort for deriving confidence bands is considerable as a large number of mobility age profiles has to be constructed and evaluated. Therefore, it is not advisable to compute confidence bands unless the computer is very powerful.

An important though very simple procedure is `resample.g`:

```
/* Resampling (with replacement)
   Input:   x  (n,k) matrix
   Output:     (n,k) matrix, the rows are resampled. */

proc resample(x);
  local smpl,n;
  n=rows(x);
```

```
  smpl=floor(n*rndu(n,1)+1);
  retp(x[smpl,.]);
endp;
```

For each re-sample a new mobility profile is computed using one of the procedures `c_msa`, `c_mst` or `c_msv`. The profiles are stored as column vectors and put next to each other to form a matrix. The following procedure (`confband.g`) derives confidence bands from the matrix of mobility age profiles.

```
/* Calculates nonparametric confidence bands
   Inputs: x  (n,k) matrix, each column is a curve
              (mobility age profile)
           p  scalar; confidence level (e.g. 0.95)
   Output:    (n,2) matrix, the first column contains the lower bounds,
              the second the upper bounds of the confidence band */

proc confband(x,p);
  local i,k,n,o,o0,u,u0,y;
  n=rows(x);
  k=cols(x);
  y=sortmat(x')';
  u=y[.,1];
  o=y[.,k];
  i=1;
  do while sumc(sumc((x.<u).or (x.>o)).==0)>k*p;
    i=i+1;
    u0=u;
    o0=o;
    u=y[.,i];
    o=y[.,k+1-i];
  endo;
  retp(u0~o0);
endp;
```

The procedure `confband.g` calls a procedure `sortmat.g`. The latter simply sorts each column of a matrix in ascending order. The code of `sortmat.g` is

```
/* Sorts each column of a matrix
   Input:   x  (n,k) matrix
   Output:     (n,k) matrix, each column sorted. */

proc sortmat(x);
  local i;
  i=0;
  do until i==cols(x);
    i=i+1;
    x[.,i]=sortc(x[.,i],1);
  endo;
  retp(x);
endp;
```

Example 4 *Suppose you have a (n, k) matrix of k resampled mobility age profiles, called* `map`. *Then* `cb=confband(map,0.95)` *yields the upper and lower boundaries of the 0.95 confidence band.*

For practical applications these procedures have to be implemented in specifically designed programs. Since there is such a wide variety of interesting questions there is as yet no program fit for general applications. The final example lists the code used for the calculations reported in the article.

Example 5 *There are two programs involved. The first one computes the bootstrap replications of the mobility age curves: The $(n, 2)$ matrix* `y0` *contains the incomes in both periods of all individuals, and the $(n, 1)$ vector* `a0` *contains their ages.*

```
/* Program for computing mobility-age-curves */

B=1000;                  @ number of bootstrap replications @
M=10|2;                  @ Parameters @
h=2000|2;                @ Bandwidths for single period incomes @
ha=sqrt(2)*h[1];         @ Bandwidth for aggregate income @
da=ha./M[1];
d=h./M;

i=0;
do until i==B;           @ Bootstrap loop @
  i=i+1;

  x=resample(y0~a0);     @ Resampling of rows @

  {sy1,sa1,f1}=warpnrm2(x[.,1 3],h,M,0,20|60);
  {sy2,sa2,f2}=warpnrm2(x[.,2 3],h,M,0,20|60);
  {sya,saa,fa}=warpnrm2((x[.,1]+x[.,2])~x[.,3],ha|h[2],M,0,20|60);

  mth=c_mst(-0.8,sy1,sy2,sya,f1,f2,fa); @ Compute profile             @
                                        @ here: based on Theils index @
  output file=zmst.dat on;              @ Save results                @
  mth';                                 @ to the file "zmst.dat"      @
  output off;
endo;
```

Depending on the number of observations this program will consume a rather large amount of computer time. Finally, the file `zmst.dat` *contains 1000 mobility age profiles, each from age 20 up to age 60 (i.e., 41 years).*

The second program evaluates the data produced by the above program:

```
/* Program for evaluating data; pointwise confidence intervals
   and confidence bands */

a=seqa(20,1,41);              @ vector of ages          @
```

```
load x[]=zmst.dat;              @ load profile bundle      @
x=reshape(x,rows(x)/41,41)';    @ put it into right shape  @
b=confband(x,0.95);             @ compute confidence bands @
x=sortmat(x');                  @ compute pointwise        @
c=x[25 975,.]';                 @   confidence intervals   @

xlabel("Age");
ylabel("M]Th(-0.8)[(t,t+1)");
scale(20|60,0|0.3);
_pltype=6|6|1|1;
xy(a,b~c);                      @ plot output              @
```

References

Härdle, W. (1991), *Smoothing Techniques: With Implementation in S*, New York: Springer.

Pudney, S. (1993), Income and Wealth Inequality and the Life Cycle: A Nonparametric Analysis for China, *Journal of Applied Econometrics* 8: 249–276.