Gustman – Steinmeier Retirement Model

First of all, the author would like to appreciate the generosity of Gustman and Steinmeier in making the programs available.

Structural models are an invaluable tool for estimating the behavioral effects of proposed Social Security reforms and their effects on economic well-being. They enable researchers to estimate underlying preference parameters, and to estimate how a utility maximizing household would respond to the reform, given its preferences. But they are difficult and time-consuming to build. Only a few researchers have chosen to construct structural models of the retirement decision, and many important aspects of the retirement decision have yet to be modeled structurally.

Gustman and Steinmeier have constructed some of the most sophisticated structural models of the retirement decision (Gustman and Steinmeier 2005, 2007, 2008, 2009, and 2010). They have generously offered to make their code used in Gustman and Steinmeier (2008) available to academic researchers through the Center for Retirement Research at Boston College (CRR). In this paper, Gustman and Steinmeier estimate a stochastic dynamic structural retirement model that contains many of the key features that researchers might want, including part-time work, stochastic asset returns, and heterogeneous rates of time preference.

The user manual will carefully document the programs of Gustman and Steinmeier model and explain how to use the rource on CRR’s website to learn the model.

1. Getting started

It is anticipated that users will be experienced programmers, with some knowledge of the *Health and Retirement Study* data, relevant Social Security rules and models of savings and labor supply. CRR staff will provide phone and e-mail technical support to researchers using the code.

a) Hardware and software

The C++ programs for GS model can run on both 32-bit and 64-bit windows. However, the data preparation program GSDataPrep.cpp could only run on 32-bit due to the DOS prompt commands written inside of that program.

All C++ programs are debugged and compiled by Microsoft Visual Studio 2013, and need to be compiled again if the codes are modified.

b) Data

* HRS data

The programs use both HRS restricted and unrestricted data. The unrestricted data are freely available to registered users of the HRS. The restricted data are only available to qualified users. This page on the HRS website explains the procedure for obtaining the restricted data: [hrsonline.isr.umich.edu/index.php?p=resdat](http://hrsonline.isr.umich.edu/index.php?p=resdat)

The programs require the following unrestricted data files (.da) and dictionary (.dct) files

"w2track”, "health", "employer", "househld", "w2cs", "w2a", "w2b", "w2d", "w2e", "w2fa", "w2fb", "w2fc", "w2j", "w2k", "w2v", "w2c", "w2n", "w2r", "w2s", "w2g", "w2h", "h96pr\_r", "h96cs\_r", "h96cs\_h", "h96a\_r", "h96b\_r", "h96c\_r", "h96pc\_r", "h96d\_h", "h96e\_r", "h96f\_h", "h96g\_r", "h96h\_r", "h96j\_r", "h96j\_h", "h96m\_r", "h96n\_r", "h96n\_h", "h96r\_r", "h96t\_r", "h98pr\_r", "h98cs\_h", "h98cs\_r", "h98a\_r", "h98b\_r", "h98c\_r", "h98pc\_r" , "h98d\_h", "h98d\_r", "h98e\_r", "h98f\_h", "h98g\_r", "h98h\_r", "h98j\_h", "h98j\_r", "h98m\_r", "h98n\_h", "h98n\_r", "h98r\_r”, "h98t\_r", "h00pr\_r", "h00cs\_h", "h00cs\_r", "h00a\_r", "h00b\_r", "h00c\_r", "h00pc\_r", "h00d\_h", "h00d\_r", "h00e\_r", "h00f\_h", "h00g\_r", "h00h\_r", "h00j\_h", "h00j\_r", "h00m\_r", "h00n\_h", "h00n\_r", "h00r\_r", "h00t\_r", "h02pr\_h", "h02pr\_r", "h02a\_h", "h02a\_r", "h02b\_r", "h02c\_r", "h02d\_r", "h02e\_h", "h02f\_r", "h02g\_r", "h02h\_h", "h02j\_r", "h02k\_r", "h02l\_r", "h02m1\_r", "h02m2\_r", "h02n\_r", "h02p\_r", "h02q\_h", "h02r\_h", "h02s\_r", "h02t\_r", "h02v\_r", "h02w\_r", "h02y\_r" and “track31.da”(only need .da file)

Those files from HRS wave 1 to wave 6 are available on HRS website by July 2015, except “w2track” and “track31” which are obsolete tracker files no longer available online (our solution is to ask HRS to hold those files on their website under the section of researchers contribution). Users could download those files from HRS website, and then save to c:\\GS\\HRS\\waveX, where X=1-6 according to corresponding waves. However, in the future, if HRS modifies the data file in early waves, the program may not work properly. (The change of amount or format of variables in the data files may cause the discrepancies in field width, which is a problem for C++ program to precisely allocate memory buffer)

The programs require the following restricted data:

* ssaear3.da, save to c:\\GS\\SSA\\

HRS covered Earning Records Wave 1 1991 Social Security Information Version 3.1 (including 9539 samples with records of annual earnings and covered earnings quarters, separate for year 1951-1991 and total for year 1937-1950)

* Partic-F.all, Partic-G.all and Partic-H.all, save to c:\\GS\\Pension

Employer pension records (Participant Files). Each line contains information such as the date of birth and the date of hire for each participant for which a pension is being evaluated. Section F is for the current job. Section G is for the last job. Section H is for a previous job lasting at least 5 years.

* Other files
* COH0015.95T, COH1655.95T, and COH5690.95T, save to c:\\GS\\

Mortality tables for cohort 1900-1915, 1916-1955 and 1956-1990

* winshell.h, winshell.ico, winshell.inc, winshell.rc, winshell64.h and winshell64.inc, save to c:\\GS\\winshell\\

C++ header and resource files, shared by Tom Steinmeier and offered by the author.

* Calc6cDB.exe and Calc6cDC.exe, save to c:\\GS\\Pension

Pension Calculator Program in 1998, including three modes to run in DOS prompt command lines. See attached file “PensionCalculatorUserGuide.pdf” for more detail.

* DPMI16BI.OVL, Param, and RTM.exe, and save to c:\\GS\\Pension

Other files needed for pension calculation, shared by Tom Steinmeier and offered by the author. See attached file “PensionCalculatorUserGuide.pdf” for more detail.

2. Run Gustman and Steinmeier’s C++ Program

There are three C++ programs. HRS92-02.cpp merges HRS 92-02 unrestricted data files. GSDataPrep.cpp cleans the data and collect other data from restricted HRS files such as Social Security earnings record and employer pension records. GSModel.cpp is the model to run the data file prepared by previous programs. All program files and data files should be organized by the directories written in the program (as mentioned above).

The following gives a detail description for each of the program. The author put comments inside of each program with following notations. EDIT means the original program is modified at this place to fit current format; UPDATE means the program needs to be updated at this place to fit more waves; INFO offers more comments to help reader understand the program. Any changes in the codes may require to compile again. After debugging and compiling, to run the program, open the executables file, choose from the pull-down menu whether you want the log output to go to the screen, file, or both, and whether you want the log to be replaced or appended, and then click start form the menu.

1. HRS92-02.cpp

This C++ program is used to merge HRS unrestricted data in c:\\GS\\HRS\\waveX by HRS identifier variable HHIDPN. (see comments inside of the program for more detail)

The output of this program is c:\\GS\\hrs92-02.dat, a data file including 12652 samples (individuals in wave 1) and c:\\GS\\hrs92-02.dic, the corresponding dictionary file.

See HRS92-02.OUT as an example of log output file.

1. GSDataPrep.cpp

This C++ program includes the following processes:

* Read in merged unrestricted HRS data. (c:\\GS\\hrs92-02.dat)
* Clean the data and prepare personal variables, job history variables, panel variables, retirement status variables.
* Calculate wage streams using restricted Social Security earning records ssaear3.da
* Calculate pension amount using files mentioned above under folder c:\\GS\\Pension\\. The C++ program will use those files to generate batch files and run them in DOS mode. The pension calculation programs are 16-bit, so this part could only be run with 32-bit windows, not 64-bit.

See attached file “Memo\_PensionCalculation” for more detail.

* Read in mortality tables for cohort 1900-1915, 1916-1955 and 1956-1990 and generate survival table c:\\GS\\SurvTabs.dat for 1900-1990.
* Calculate Social Security amount
* Save sample information to c:\\GS\\GSretirement.dat, which will be the input data file for GS model. This is a binary file, with first four bytes are an integer for the number of observations, and the rest of the file are the observations.

Assumptions: inflation is 4%; real wage growth rate plus inflation is 5.1%; discount rate is 6.3%. Subsititute assumptions by global searching 1.04, 1.051 or 1.063 with own assumptions.

See GSDataPrep.OUT as an example of log output file.

1. GSModel.cpp

This is the model part. It has several modes determined by a variable called “mode” in the main routine.

* Mode 0 = test mode: test the program on a synthetic individual

When mode is set to a value of zero, as it currently is, the model operates in test mode on a synthetic individual whose characteristics are given in the subroutine setData. When started, the program will ask for a range of thread numbers. The number of threads is given by nthreads in the main routine and should be equal to the lesser of the numbers of observations (one in the test case) or the number of cores in the machine (twice that number if the cores are multithreaded). For instance, if the machine has four multithreaded cores, nthreads should be set equal to 8, and the range of threads should be entered as 0 7. But in test mode the range should be entered as 0 0. If the program is run on multiple machines, the program should be started on the slave machines first and the range of threads on those machines will be something like 8 11, numbered consecutively after the range of threads on the master machine. Each slave machine should have a unique range of thread values. The master machine will always have a range of threads starting with 0.

See GSModel\_0.OUT as an example of log output file.

* Mode 1 = iteration mode: a single function evaluation given the initial values of parameters in one iteration.

The program read in GSRetirementData.dat for a data structure called ORIGDATA which could be found in the inpt routine. The routine calcPrep calculates several values for the structure CALCSTRUCT which is used to pass values among routines. The routine rhocalc calculates a value of the time preference variable consistent with observed wealth for retirement at the observed or expected retirement age. The routine calcModel solves the model for the decision variables as a function of the state variables, and the routine getMoments does the simulations.

See GSModel\_1.OUT as an example of log output file.

* Mode 2 = estimation and simulation mode:

Mode 2 is basically repeating the iteration process in mode but slightly changing the values of parameters to minimize the function distance called q. After that, the program runs simulation using the estimated parameters.

See GSModel\_2.OUT as an example of log output file.

3. Learn GS Model in MATLAB

For the purose of easy learning the model, the author also offers the MATLAB version of the program which keeps the original variables and functions’ names in C++ program and replicates the same results (see log file MATLAB\_GSModel.log from MATLAB and OUT file GSModel.out from C++). The flow chart and manual document for MATLAB program are very helpful to fully understand the original C++ program for GS Model. The MATLAB code is much more intuitive than C++ and will help users understand the estimation technique, though users will not want to run the MATLAB code in estimation mode as it is much slower than C++. See the files and user manual in MATLAB section for more detail.