

Concording U.S. Harmonized System Categories Over Time*

Justin R. Pierce[†]
Georgetown University

Peter K. Schott[‡]
Yale School of Management & NBER

March 2009

Abstract

This paper: outlines an algorithm for concording U.S. ten-digit Harmonized System export and import codes over time; describes the concordances we construct for 1989 to 2004; and provides Stata code that can be used to construct similar concordances for arbitrary beginning and ending years from 1989 to 2007.

JEL classification: F1

*Schott thanks the National Science Foundation (grants SES-0241474 and SES-0550190) and the Yale School of Management for research support. We thank Julie Linden of the Yale University Social Sciences Library for generous help in securing the U.S. trade data. We thank Mayumi Hairston Escalante for advice and for generous provision of unpublished Census “new-obsolete” files. The research in this paper was conducted while the authors were Special Sworn Status researchers at the U.S. Census Bureau, Center for Economic Studies. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the U.S. Census Bureau.

[†]37th & O Streets, NW, Washington, DC, 20057, tel: (202) 230-9611, email: jrp45@georgetown.edu.

[‡]135 Prospect Street, New Haven, CT 06520, tel: (203) 436-4260, fax: (203) 432-6974, email: peter.schott@yale.edu.

1. Introduction

This paper serves three purposes. First, it outlines an algorithm for concording ten-digit U.S. Harmonized System (HS) product codes over time. Second, it describes how this algorithm can be used to construct an export- or import-code concordance for any arbitrary beginning and ending years from 1989 to 2007. Finally, it summarizes the 1989 to 2004 HS concordances used in Bernard, Jensen, Redding and Schott's (2009) analysis of the margins of U.S. trade and provides statistics illustrating the prevalence of changes in HS codes during that time interval. We note that though the official names of U.S. export and import product codes are "Schedule B" and "Harmonized Tariff Schedule" codes, respectively, we refer to both generically as HS codes in this paper.

Section 2 provides a brief description of HS codes. Section 3 describes the data used to construct our concordance. Section 4 outlines our algorithm. Section 5 summarizes the 1989 to 2004 concordance. An appendix contains the Stata computer code and describes the input files used to generate concordances.

2. Brief Description of HS Codes

U.S. HS codes are based on the Harmonized System established by the World Customs Organization (WCO). The WCO assigns 6-digit codes for general categories, and countries adopting the system then define their own codes to capture commodities at more detailed levels. In the United States, the most detailed level of disaggregation is ten digits. In this paper, we refer to ten-digit codes as "product" or "goods" categories. U.S. export codes are administered by the United States Census Bureau (Census). U.S. import codes are administered by the U.S. International Trade Commission (USITC).

Changes to U.S. export or import product codes can occur via three routes: changes by the WCO to the official list of international six-digit prefixes; U.S. legislation that affects U.S. eight-digit codes (imports only); and changes by the Committee for Statistical Annotation of Tariff Schedules (known as the "484(f) Committee") to statistical ten-digit codes.¹

HS codes are updated for several reasons. The WCO, for example, makes adjustments to eliminate six-digit roots that capture little or no trade, with a goal of having trade roughly balanced across codes. In addition, the 484(f) Committee may split a single HS code into several new codes in order to report import or export data at a more detailed level. Similarly, producers may petition one of the official bodies noted above for code changes to obtain a higher profile for the goods they export or import.

3. Data

Each year, Census publishes documents outlining the HS codes that have become "obsolete" and the "new" codes that will take their place. We refer to these documents as Census' "new-obsolete" files. For exports, HS-code changes take effect annually in January; for imports, they can occur within as well as across years. New-obsolete files for years before 1997 are available only in hard copy and were transcribed into electronic form as part of the construction of our concordance. These files as well as electronic versions of subsequent files were obtained from Mayumi Hairston Escalante at Census. The most recent new-obsolete files are currently posted on the Census website.²

¹See http://www.census.gov/foreign-trade/aip/comb_seminar_pres.ppt, and www.census.gov/foreign-trade/faq/sb/sb0008.html for more detail.

²See <http://www.census.gov/foreign-trade/schedules/b/#obsolete> and <http://www.usitc.gov/tata/hts/index.htm>, respectively.

We use the terms “simple” and “complex” to describe the two basic changes to HS codes that can occur in a new-obsolete file.³ Simple changes make no adjustments to the actual items covered by a particular code, they just swap one ten-digit code for another. There are several possible reasons for a one-to-one re-numbering, including:

1. To align the Schedule B and HTS codes where Census finds their descriptions are the same;
2. To differentiate the Schedule B and HTS codes where Census has found them to be different;
3. To correct errors by reclassifying a commodity under a different subheading;
4. To maintain the level of statistical detail after a revision of the 6- or 8-digit codes; and
5. To accommodate a new numbering pattern, usually the result of another code being broken out.

In contrast to simple changes, complex changes alter the mix of items captured by a particular code. For these changes, the items formerly encompassed by one or more “obsolete” codes are distributed to one or more “new” codes. In 2002, for example, various types of waste oil, which previously were grouped with the fresh oils to which they were most similar, were given their own HS codes. As a result, the (now obsolete) former fresh oil product categories were linked to the new waste oil categories from which they emerged.

For each set of new-obsolete mappings in a particular new-obsolete file, we construct a synthetic HS code which we refer to as a “setyear” (`setyr` in our Stata code). This synthetic code records both the count of the change since the first change in 1989 and an identifier for when it takes place. Formally, for exports, it is defined as the count of the particular mapping plus the four-digit year in which the change occurs divided by 10,000. For imports, it is the count of the particular mapping plus six-digit year-month in which the change occurs divided by 1,000,000. The very first setyears for exports and imports, for example, are equal to 1.1989 and 1.198906.

Table 1 summarizes the number of new-obsolete mappings in the raw data for export and import codes, respectively. Results for export codes are displayed in the left panel while those for import codes are displayed in the middle and right panels. The first column of each panel notes the year-month in which the noted changes take place. The second and third columns report the total number of retired and replacement codes encompassed by the number of sets reported in column four. Note that the number of sets in column four is smaller than the numbers of HS codes in columns two and three because multiple codes are often involved in a particular change. The fifth column reports the number of changes that are “simple” in the sense outlined above.

As indicated in the table, HS codes are updated unevenly in the sense that some years (e.g., 2002) encompass substantially more changes than others (e.g. 2000).

4. Developing an HS Concordance

Concording HS codes over time is complicated by the existence of chains of HS-code changes across months and years into “family trees”. There are two basic types of family tree. We refer to the first case, displayed in Figure 1, generically as a “growing family tree”. In this case, code a from period t may become obsolete and be mapped to new codes b and c in period $t + 1$. Then, in period $t + 2$, codes b and c may become obsolete and be mapped to new codes e and f , and g and h ,

³Some new-obsolete files contain “blanket” mappings, our term for mappings that include codes ending in a series of X’s, e.g., 8486XXXXXX. We drop these observations.

Table 1: HS Code Changes by Year-Month

Exports					Imports									
Date	Obsolete	New	Sets	Simple	Date	Obsolete	New	Sets	Simple	Date	Obsolete	New	Sets	Simple
1989_01	234	310	157	92	1989_06	2	12	2	0	1998_01	52	85	47	18
1990_01	156	201	96	60	1989_07	112	196	91	27	1998_03	4	8	2	0
1991_01	186	313	131	34	1990_01	346	724	295	15	1998_04	3	3	3	3
1992_01	37	60	29	9	1990_05	16	20	16	12	1998_07	6	8	6	4
1993_01	64	126	60	19	1990_07	133	256	119	25	1998_08	9	23	9	0
1994_01	109	181	77	25	1990_08	38	49	30	17	1999_01	81	88	53	16
1995_01	137	205	113	63	1990_10	70	121	47	6	1999_07	54	70	33	5
1996_01	787	1,071	532	349	1991_01	69	194	45	0	2000_01	16	29	13	0
1997_01	216	232	145	107	1991_02	15	24	15	6	2000_03	11	30	11	0
1998_01	128	138	101	76	1991_05	11	20	11	2	2000_04	10	17	7	0
1999_01	23	29	22	17	1991_07	247	393	190	77	2000_07	6	13	6	1
2000_01	6	15	6	0	1992_01	85	138	50	0	2000_12	24	45	24	3
2001_01	16	9	7	0	1992_05	28	29	28	27	2001_01	119	113	55	1
2002_01	717	1,031	531	323	1992_07	117	194	109	42	2001_07	19	25	9	3
2003_01	97	87	81	74	1993_01	135	218	74	7	2002_01	1,122	1,542	874	595
2004_01	11	14	10	5	1993_02	42	51	42	33	2002_07	86	84	66	49
2005_01	43	82	38	8	1993_06	3	5	2	0	2002_08	5	10	5	0
2006_01	3	4	2	0	1993_07	7	8	7	6	2003_01	26	44	20	0
2007_01	1,140	1,030	821	631	1993_08	33	53	25	0	2003_02	1	2	1	0
					1993_11	8	10	2	0	2003_04	5	4	4	3
					1993_12	1	2	1	0	2003_07	45	67	37	11
					1994_01	667	1,082	468	176	2004_01	46	38	23	2
					1994_04	13	43	13	0	2004_02	5	7	4	0
					1994_06	66	112	47	0	2004_04	4	4	2	0
					1995_01	1,933	2,187	1,162	555	2004_07	44	87	37	1
					1995_07	38	73	31	0	2005_01	42	72	39	11
					1995_09	77	168	33	12	2005_07	32	45	26	9
					1996_01	1,164	1,485	798	523	2005_11	4	8	4	0
					1996_06	5	8	5	4	2006_01	19	38	19	0
					1996_07	4	12	4	0	2006_03	2	2	2	2
					1996_11	18	31	18	3	2006_04	4	5	4	3
					1997_01	148	198	107	66	2006_06	49	58	9	0
					1997_02	11	11	11	11	2006_07	63	59	35	0
					1997_06	18	33	18	3	2007_01	2,026	1,896	1543	1,220
					1997_07	231	319	190	89	2007_07	25	35	16	3
					1997_08	55	65	33	1					

Notes: Table reports changes to export (left panel) and import (middle and right panel) HS codes in noted year-month. Obsolete is number of codes retired from prior year. New is number of codes replacing these retirements. Sets is a count of the overall number of obsolete-new matches. Simple refers to re-numberings of individual codes.

respectively. Our concordance of the period t to period $t + 2$ HS codes assigns a common synthetic code to all HS codes in a growing family tree. Such an assignment may result in potentially many more HS codes being mapped to a given synthetic code in the final year of the concordance than in the first year.⁴

The second type of family tree, which we refer to generically as a “shrinking family tree”, is displayed in Figure 2. In this case, codes a and b , and c and d , from period t separately become obsolete and mapped to codes e and f , respectively, in period $t + 1$. Then, in period $t + 2$, codes e and f become obsolete and are assigned to new code g . In this case, the number of HS codes mapped to the family’s common synthetic code declines over time.⁵

The algorithm we develop for concording HS codes between arbitrary beginning and ending year-months accounts for both types of family trees, as well as combinations of the two types. Though specific details about how the algorithm is implemented can be determined by examining

⁴In 1997, for example, 7802000000 is mapped to 7802000030 and 7802000060. In a 1996 to 1997 concordance, we would assign a single synthetic HS code to all of these actual HS codes.

⁵In 1997, for example, 8506800010 and 8506800050 are mapped to 8506800000. In a 1996 to 1997 concordance, we would assign a single synthetic HS code to all of these actual HS codes.

the Stata code in the Appendix, the basic steps are as follows:

1. Read in raw obsolete-new mappings;
2. Assign a single setyear to each obsolete-new mapping appearing in the raw files;
3. Choose a beginning and end year for the concordance;
4. Identify family trees extending between the beginning and end years of the concordance; and
5. Assign all members of a family tree the minimum setyear among family members within the time-frame of the concordance. Note that the part of the setyear after the decimal point identifies the year in which the family tree starts (i.e., period t in Figures 1 and 2 above). In the Stata code below, a separate variable (named `effyr`) identifies the year in which a particular new-obsolete mapping occurs.⁶

Step four is accomplished by successively merging subsequent obsolete-new mappings to all periods' new-obsolete mappings between the beginning and end years of the concordance. To bridge codes used from 1989 to 2004 for example, the chained file is constructed as follows. First, merge the new codes in the 1990 file to the obsolete codes in 1991 file, dropping any codes that are unique to 1991. Second, merge the obsolete codes in the 1992 file to the new codes in the previously merged 1990-1991 file, again dropping any codes unique to 1992. And so on. Note that this successive merging has to be done starting with every year-month between the beginning and ending year-month because chains can begin in any year-month, and they would be missed otherwise given the dropping just mentioned. After these chains are created, they are appended into a single file and added to all obsolete-new mappings that are not parts of a chain.

5. A 1989-to-2004 Concordance

This section describes the 1989 to 2004 concordance used by Bernard, Jensen, Redding and Schott (2009) in their analysis of the margins of U.S. trade. The first and second columns of Table 2 summarize total U.S. exports in 1989 and 2004 and the total number of HS product categories exported in those two years, respectively. Columns three and four provide analogous detail with respect to U.S. imports. As indicated in the table, (nominal) exports more than double while (nominal) imports more than triple over the fifteen-year interval. The number of pre-concorded export and import HS codes observed in each year of data, by contrast, grow 13 percent and 21 percent, respectively.

Table 2: Trade in 1989 and 2004

	Exports		Imports	
	Value	Codes	Vale	Codes
1989	354	7,853	468	13,941
2004	818	8,859	1,460	16,836

Notes: Export and import values in billions of U.S. dollars. Number of codes refers to number of original ten-digit HS categories in the raw trade data.

⁶For example, in 1998 export code 8531800035 from 1997 is mapped to code 8531804000. Then, in 2002, codes 8531804000 and 8527908015 from 2001 are mapped into 8527908600. The `setyr` for the family is 1404.1998. The integer part of this `setyr` indicates that the first mapping in the family, from 8531800035 to 8531804000, is the 1404th mapping since 1989. The part after the decimal point indicates it occurs in 1998. The `effyr` for the two mappings are 1998 and 2002, respectively.

Table 3 reports two decompositions of export and import codes. The first three rows of the Table show how many of the original HS codes in each year survive versus being replaced by synthetic codes. The remaining rows in the table decompose the actual plus synthetic codes that remain after the concordance into those which are common across years and those which are idiosyncratic to a particular year.

Table 3: Distribution of Product Codes in Matched 1989 to 2004 U.S. Trade Data

	Exports				Imports			
	1989		2004		1989		2004	
Original HS Codes	7,853	100	8,859	100	13,941	100	16,836	100
Not Replaced by Synthetic Codes	5,349	68	5,341	60	8,585	62	8,508	51
Replaced by Synthetic Codes	2,504	32	3,518	40	5,356	38	8,328	49
Actual + Synthetic Codes After Concordance	6,978	89	6,971	79	12,262	88	12,240	73
Actual Codes	5,349	68	5,341	60	8,585	62	8,508	51
Common to both years	5,318	68	5,318	60	8,240	59	8,240	49
Appear in only one year	31	0	23	0	345	2	268	2
Synthetic Codes	1,629	21	1,630	18	3,677	26	3,732	22
Common to both years	1,624	21	1,624	18	3,570	26	3,570	21
Appear in only one year	5	0	6	0	107	1	162	1

Notes: Table decomposes the number of original HS codes in each year into those replaced by a synthetic code versus not, and total surviving HS plus synthetic codes in each year into noted sub-groups. All replacements are with respect to a 1989 to 2004 concordance. Even columns display values as a percent of first row in preceding column.

Of the 7,853 original HS codes appearing in the 1989 U.S. export data, for example, 2,504 are replaced by synthetic codes. Since the same synthetic code is often assigned to more than one original code, the resulting concorded dataset contains 6,978 actual plus synthetic codes. Of these, 5,349 and 1,629 are actual and synthetic, respectively. Each of these totals, in turn, can be broken down into actual codes which are common to both 1989 and 2004 (5,318), synthetic codes that are common to both 1989 and 2004 (1,624), actual codes unique to 1989 (31) and synthetic codes that are unique to 1989 (5). These breakdowns reveal that the number of actual and synthetic export and import goods actually added and dropped between 1989 and 2004 is relatively small.

The values of U.S. exports and imports associated with each of the cells in Table 3 are reported in Table 4. As indicated below, synthetic codes account for the majority of import value in both 1989 and 2004.

Table 4: Distribution of Value in Matched 1989 to 2004 Trade Data

	Exports				Imports			
	1989		2004		1989		2004	
Original HS Codes	353,766	100	817,936	100	468,012	100	1,460,160	100
Not Replaced by Synthetic Codes	206,556	58	428,571	52	178,545	38	550,049	38
Replaced by Synthetic Codes	147,210	42	389,365	48	289,467	62	910,111	62
Actual + Synthetic Codes After Concordance	353,766	100	817,936	100	468,012	100	1,460,160	100
Actual Codes	206,555	58	428,571	52	178,545	38	550,049	38
Common to both years	188,832	53	408,903	50	175,517	38	537,508	37
Appear in only one year	17,723	5	19,668	2	3,028	1	12,541	1
Synthetic Codes	147,210	42	389,366	48	289,466	62	910,111	62
Common to both years	147,143	42	388,971	48	288,273	62	906,775	62
Appear in only one year	67	0	395	0	1,193	0	3,336	0

Notes: Table decomposes U.S. export and import value according to whether HS codes are original or synthetic. All replacements are with respect to a 1989 to 2004 concordance. Values are in millions of U.S. dollars. Even columns display values as a percent of first row in preceding column.

Tables 3 and 4 also underscore the prevalence of changes in HS codes over time. As of 2004, 49 percent of import products and 40 percent of export products had been involved in an HS code change. Moreover, trade in products with code changes accounted for 62 percent of the value of

U.S. imports and 48 percent of the value of U.S. exports in 2004. Tracking changes in HS codes over time, therefore, is important in any empirical research using international trade data classified by HS codes, and critical when studying topics such as new product introduction.⁷

6. Conclusion

This paper has presented an algorithm for concording ten-digit U.S. Harmonized System (HS) product codes over time and described how the algorithm can be used to create concordances with arbitrary beginning and end years. Furthermore, in summarizing the 1989 to 2004 concordance used in Bernard, Jensen, Redding and Schott (2009) it has illustrated the prevalence of changes in HS codes over time and the importance of tracking these changes when conducting empirical research in international trade.

⁷We note that two features of Census' new-obsolete mappings complicate the identification of new product introductions (e.g., iPods). First, new HS codes always emerge from predecessor HS codes. Second, new HS codes' emergence may take place an unknown period of time after an underlying good has been introduced. Statistical agencies may wait to establish a new HS category until it reaches a certain size or until manufactures apply sufficient lobbying.

References

Bernard, Andrew B., J. Bradford Jensen, Stephen J. Redding and Peter K. Schott. 2009. “The Margins of U.S. Trade (Long Version).” NBER Working Paper 14662.

A Appendix

This appendix provides Stata code that can be used to create HS concordances. It also describes the input and output files associated with this code. These files are contained in the zip file `hs_over_time_20090302.zip`, which is available on the authors' websites. The two sections of code below contain our algorithm for creating export and import HS concordances for arbitrary beginning and ending year-months between 1989 and 2007. Those comfortable with Stata programming should find it relatively easy to manipulate. Those unfamiliar with Stata programming can instead use one of the output files described below.

Each program requires as an input a data file containing the raw new-obsolete mappings discussed in the main text. These input files are named `sch_b_concordances_20081101_02.dta` and `hts_concordances_20081101_02.dta`, respectively, with the string after the “_” reflecting the version date of the file. The basic structure of these input files resembles the raw new-obsolete files, i.e., each set of obsolete HS codes is followed by the new set of HS codes into which they map. They are posted to the same website where this paper is found and contain the following variables:

- `obsolete`: old HS codes that become obsolete as of effective date;
- `new`: new HS codes associated with the obsolete codes;
- `setyr`: synthetic code to which new and obsolete codes belong, as defined in main text; and
- `effyr`: date the mapping is effective.

The first two sections of code below produce the output files that can be used to concord HS codes in U.S. import and export data, as demonstrated in the last section of this Appendix. Specifically, the code produces output files `sch_b_concordances_VER_BEG_END.dta` and `hts_concordances_VER_BEG_END.dta`, where `VER`, `BEG` and `END` reflect user-defined version dates (currently 20081101) and beginning-end years (exports: 1989_2007) or year-months (imports: 198906_200707), respectively. These concordances include the same variables as the input files, but with `setyr` and `effyr` standardized across family trees, as described in Section 4 above. Variables in the concordance output files include:

- `obsolete`: obsolete HS code;
- `new`: corresponding new HS code;
- `setyr`: synthetic code linking this mapping to all mappings in its family tree;
- `effyr`: year (export) or year-month (import) in which the particular new-obsolete mapping first appears in the raw data.

For those unfamiliar with Stata programming we also provide two additional output files in `.txt` format. These output files, named `setyr_x_1989_2007.txt` and `setyr_m_1989_2007.txt`, provide a record of every HS code associated with every `setyr` that appears in the 1989-2007 concorded data. The first column of each file lists the `setyr`'s, sorted from low to high. Each additional column lists the actual HS codes appearing in a particular year of the trade data that should be replaced by the `setyr`. These actual HS codes also are sorted from low to high in each year. To concord U.S. trade data from 1989 to 2007, one would just replace all codes listed in the table with the synthetic `setyr`, and then collapse the data according to these `setyr`'s. HS codes not appearing in these output files are consistent across all years of the data.

A1. *Stata Code for Schedule B Concordance*

```

**1 Preliminaries
clear
set more off
set mem 1000m

**2 Create a file that chains years together
** Note that to chain you have to always match later years to earlier years. That is the reason
** that the second loop below is nested
** Note that you must set the local variables for the beg and end year you want;
** these locals govern both this and the next section.
local b = 1989
local e = 2004
local b1 = `b'+1
set more off
quietly {
    *chop up the data in the main file created above year and rename the vars for
    *the merging to take place in the next loop
    forvalues y=`b'/'e' {
        use sch_b_concordances_20081101_02, clear
        keep if effyr==`y'
        rename new new`y'
        rename obsolete obs`y'
        rename setyr setyr`y'
        rename effyr effyr`y'
        order obs`y' new`y'
        sort obs`y'
        save temp_xchain_`y', replace
    }
    *use the chopped up files from above to chain the obs-new matches across years.
    *the goal is to find news from subsequent years that modify new's from earlier years
    *the joinby command assumes all possible trees from a given origin are captured
    *note that after the inside loop, which matches subsequent years to a given year, we drop
    *observations unless they are chained, i.e., unless the merge code = 3
    forvalues s=`b'/'e' {
        use temp_xchain_`s', clear
        rename obs`s' obs
        forvalues t=`b'/'e' {
            if `t'>`s' {
                noisily display [`s'] " " [`t']
                rename new`s' obs`t'
                sort obs`t'
                joinby obs`t' using temp_xchain_`t', unmatched(master)
                noisily tab _merge
                drop if _merge==2
                rename _merge _m`s't'
                rename obs`t' new`s'
            }
        }
    }
}

```

```

    }
  }
  gen _mjunk=0
  egen idx = rowmax(_m*)
  noisily tab idx
  keep if idx==3
  sort obs
  drop _m*
  save temp2_xchain_ 's', replace
}
}

**3 Assign single setyear to all members of a family
**put the above chains, each of which starts with a different year from 1989 to 2004, back
**together into one file for the whole sample period;
**challenge here is to set a single setyr for all "families" revealed by the chain;
**note that there are two cases for a "family". in the first, all members sprout from the same
**obsolete code in some year. in the second, two sub-families in an early year are joined by a
**common code of set of codes in a subsequent year.
**the iteration of min commands below takes care of both cases by searching for the setyr for
**a family that covers all of its members.
use temp2_xchain_ 'b', clear
forvalues y='b1'/'e' {
  append using temp2_xchain_ 'y'
}
keep obs new* setyr* effyr*
capture duplicates drop
egen double setyr = rowmin(setyr*)
egen nchain = rownonmiss(new*)
rename obs obsolete
order obs setyr
sort obs
save temp2_xchain, replace
use temp2_xchain, clear
drop setyr effyr*
egen t1 = seq(), by(obs)
reshape long new setyr, i(obs t1) j(effyr)
drop if new==. & setyr==.
drop t1 nchain
duplicates drop obs effyr new setyr, force
egen osd=sd(setyr), by(obs)
egen nsd=sd(setyr), by(new)
sum nsd osd
drop osd nsd
*Now add back in the obs-new observations that are not part of chains (from section 2)
*have to add these in before the min loop below in case a non-chain obs-pair is part of a family
sort obsolete new effyr
merge obsolete new effyr using sch_b_concordances_20081101_02

```

```

drop if effyr<'b' | effyr>'e'
tab _merge
drop _merge
*now start family identification loop
egen double t1 = min(setyr), by(obs)
rename setyr oldsetyr
local zzz = 2
local stop = 0
while 'stop'==0 {
  quietly {
    noisily display ['zzz']
    local zlag = 'zzz'-1
    if mod('zzz',2)==0 {
      egen double t'zzz' = min(t'zlag'), by(new)
    }
    if mod('zzz',2)~0 {
      egen double t'zzz' = min(t'zlag'), by(obs)
    }
    compare t'zzz' t'zlag'
    gen idx = t'zzz'==t'zlag'
    tab idx
    local stop = r(r)==1
    local zzz = 'zzz'+1
    display r(r) " " ['stop']
    drop idx
  }
}
local yyy = 'zzz'-1
gen double setyr = t'yyy'
keep obs effyr new setyr
duplicates drop
sort obsolete new effyr
save sch_b_concordances_20081101_'b'_'e', replace
lerase temp*.dta tn.dta to.dta sch_b*_01.dta sch_b*_02.dta

```

A2. Stata Code for HS Concordance

```

**1 Preliminaries
clear
set more off
set mem 1000m

**2 Create a file that chains years together
** Note that to chain you have to always match later years to earlier years. That is the reason
** that the second loop below is nested
** Note that you must set the local variables for the beg and end year you want;

```

** these locals govern both this and the next section.

local b = 1989.06

local e = 2004.07

local list1 = "1989.06 1989.07 1990.01 1990.05 1990.07 1990.08 1990.10 1991.01 1991.02 1991.05 1991.07
1992.01 1992.05 1992.07 1993.01 1993.02 1993.06 1993.07 1993.08 1993.11 1993.12 1994.01 1994.04 1994.06
1995.01 1995.07 1995.09 1996.01 1996.06 1996.07"

local list2 = "1996.11 1997.01 1997.02 1997.06 1997.07 1997.08 1998.01 1998.03 1998.04 1998.07 1998.08
1999.01 1999.07 2000.01 2000.03 2000.04 2000.07 2000.12 2001.01 2001.07 2002.01 2002.07 2002.08 2003.01
2003.02 2003.04 2003.07 2004.01 2004.02 2004.04"

local list3 = "2004.07 2005.01 2005.07 2005.11 2006.01 2006.03 2006.04 2006.06 2006.07 2007.01 2007.07"

set more off

quietly {

*chop up the data in the main file created above year and rename the vars for

*the merging to take place in the next loop; have to do this for every year-month

*because chains below need to start, iteratively, with each year-month

foreach y in 'list1' 'list2' 'list3' {

noisily display ['y']

local yn = int('y'*100)

use hts_concordances_20081101_02, clear

keep if effyr=='y'

rename new new'yn'

rename obsolete obs'yn'

rename setyr setyr'yn'

rename effyr effyr'yn'

order obs'yn' new'yn'

sort obs'yn'

save temp_xchain_'yn', replace

}

*use the chopped up files from above to chain the obs-new matches across years.

*the goal is to find new's from subsequent years that modify new's from earlier years

*the joinby command assumes all possible trees from a given origin are captured

*note that after the inside loop, which matches subsequent years to a given year, we drop

*observations unless they are chained, i.e., unless the merge code = 3

foreach s in 'list1' 'list2' 'list3' {

local sn = int('s'*100)

if 's'>='b' & 's'<='e' {

use temp_xchain_'sn', clear

rename obs'sn' obs

foreach t in 'list1' 'list2' 'list3' {

if 't'>'s' & 't'<='e' {

noisily display ['s'] " " ['t']

local tn = int('t'*100)

rename new'sn' obs'tn'

sort obs'tn'

joinby obs'tn' using temp_xchain_'tn', unmatched(master)

noisily tab _merge

drop if _merge==2

rename _merge _m'sn'tn'

```

        rename obs`tn` new`sn`
    }
}
gen _mjunk=0
egen idx = rowmax(_m*)
noisily tab idx
keep if idx==3
sort obs
drop _m*
save temp2_xchain_`sn`, replace
}
}
}

```

```

**3 Assign single setyear to all members of a family
**put the above chains, each of which starts with a different year from 1989 to 2004, back
**together into one file for the whole sample period;
**challenge here is to set a single setyr for all "families" revealed by the chain;
**note that there are two cases for a "family". in the first, all members sprout from the same
**obsolete code in some year. in the second, two sub-families in an early year are joined by a
**common code of set of codes in a subsequent year.
**the iteration of min commands below takes care of both cases by searching for the setyr for
**a family that covers all of its members.set more off
local b = 1989.06
local e = 2004.07
local b1 = 1989.01
local list1 = "1989.06 1989.07 1990.01 1990.05 1990.07 1990.08 1990.10 1991.01 1991.02 1991.05 1991.07
1992.01 1992.05 1992.07 1993.01 1993.02 1993.06 1993.07 1993.08 1993.11 1993.12 1994.01 1994.04 1994.06
1995.01 1995.07 1995.09 1996.01 1996.06 1996.07"
local list2 = "1996.11 1997.01 1997.02 1997.06 1997.07 1997.08 1998.01 1998.03 1998.04 1998.07 1998.08
1999.01 1999.07 2000.01 2000.03 2000.04 2000.07 2000.12 2001.01 2001.07 2002.01 2002.07 2002.08 2003.01
2003.02 2003.04 2003.07 2004.01 2004.02 2004.04"
local list3 = "2004.07 2005.01 2005.07 2005.11 2006.01 2006.03 2006.04 2006.06 2006.07 2007.01 2007.07"
local bn = int(`b`*100)
local en = int(`e`*100)
local b1n = int(`b1`*100)
use temp2_xchain_`bn`, clear
foreach y in `list1' `list2' `list3' {
    if `y`>`b` & `y`<=`e' {
        local yn = int(`y`*100)
        display [`y']
        append using temp2_xchain_`yn`
    }
}
}
keep obs new* setyr* effyr*
capture duplicates drop
egen double setyr = rowmin(setyr*)
egen nchain = rownonmiss(new*)

```

```

rename obs obsolete
order obs setyr
sort obs
save temp2_xchain, replace
use temp2_xchain, clear
drop setyr effyr*
egen t1 = seq(), by(obs)
reshape long new setyr, i(obs t1) j(effyr)
rename effyr t2
gen double effyr = t2/100
drop if new==. & setyr==.
drop t1 nchain t2
duplicates drop
egen osd=sd(setyr), by(obs)
egen nsd=sd(setyr), by(new)
sum nsd osd
drop osd nsd
*Now add back in the obsolete-new observations that are not part of chains. These come from section 1
*have to add these in before the min loop below in case a non-chain obs-pair is part of a family
sort obsolete new effyr
merge obsolete new effyr using hts_concordances_20081101_02
drop if effyr<'b' | effyr>'e'
tab _merge
drop _merge
*now start family identification loop
egen double t1 = min(setyr), by(obs)
rename setyr oldsetyr
local zzz = 2
local stop = 0
while 'stop'==0 {
  quietly {
    noisily display ['zzz']
    local zlag = 'zzz'-1
    *mod(x,y) = x - y*int(x/y).
    if mod('zzz',2)==0 {
      egen double t'zzz' = min(t'zlag'), by(new)
    }
    if mod('zzz',2)~=0 {
      egen double t'zzz' = min(t'zlag'), by(obs)
    }
    compare t'zzz' t'zlag'
    gen idx = t'zzz'==t'zlag'
    tab idx
    local stop = r(r)==1
    local zzz = 'zzz'+1
    noisily display r(r) " " ['stop']
    drop idx
  }
}

```

```

}
local yyy = 'zzz'-1
gen double setyr = t'yyy'
keep obs effyr new setyr
rename effyr effyrmo
gen effyr = int(effyrmo)
duplicates drop
sort obsolete new effyrmo
save hts_concordances_20081101_ 'bn' 'en', replace
!erase temp*.dta tn.dta to.dta hts*_01.dta hts*_02.dta

```

A3. Stata Code for Implementing Concordance in U.S. Trade Data

```

/*
Note that you must change the use and save commands below
depending on whether you are concording export or import data
*/

quietly {
  forvalues y=1989/2004 {
    local ylead = 'y'+1
    noisily display " "

    noisily display " "
    noisily display "NEW LOOP " ['y']
    noisily display " "
    noisily display " "

    *get obsolete-new files ready
    *temp_obsolete is used to assign setyrs to codes that are last used in year y
    *basically want to insure against the code ever becoming obsolete, i.e., it being
    *an obsolete code in any year after the year of the loop
    *note the input file varies depending on whether import or export data
    *use sch_b_concordances_20081101_1989_2004, clear
    use hts_concordances_20081101_1989_2004, clear
    keep if effyr>='ylead'
    keep obsolete setyr
    drop if obsolete==.
    capture duplicates drop
    sort obsolete
    save temp_obsolete, replace
    *temp_new is used to assign setyrs to codes that are new in year y

    *basically want to insure against this code ever having been a new code prior to this
    *year; if so, need to assign it a setyr
    *use sch_b_concordances_20081101_1989_2004, clear
    use hts_concordances_20081101_1989_2004, clear

```

```

keep if effyr<='ylong'
keep new setyr
drop if new==.
duplicates drop
sort new
save temp_new, replace
*read in data and collapse to appropriate level

*assume trade file is called exports_Y or imports_Y, where Y=year
*assume file contains v=value, hs1=hs code, country1=us country code,
*year and month
*use exports_'y', clear
use imports_'y', clear
rename all _val_yr v
destring commodity, force g(hs1)
gen year = 'y'
gen month = int(uniform()*12) + 1
gen rp = uniform()>0.5
destring cty_code, force g(country1)
gen alpha1 = 1
collapse (sum) v, by(hs1 country1 month year)
format hs1 %15.0f
*merge in obsolete-code family identifiers
rename hs1 obsolete
sort obsolete
merge obsolete using temp_obsolete, keep(setyr)
noisily tab _merge
drop if _merge==2
drop _merge
rename obsolete hs1
*merge in new-code family identifiers
rename hs1 new
sort new
merge new using temp_new, keep(setyr) update
noisily tab _merge
drop if _merge==2
drop _merge
save exports_'y'_concorded_precollapse, clear
save imports_'y'_concorded_precollapse, clear
rename new hs1
*resent hs codes to family identifiers where appropriate
replace hs1=setyr if setyr~=
collapse (sum) v, by(hs1 country1 month year)
*save exports_'y'_concorded, replace
save imports_'y'_concorded, replace
}
}

```

```

*create files matching actual codes to setyr's by year
forvalues y=1989/2004 {
    *use exports_'y'_concorded_precollapse, replace
    use imports_'y'_concorded_precollapse, replace
    rename hs1 hs'y'
    drop v
    drop if setyr==.
    sort setyr hs'y'
    *save junk_x_'y', replace
    save junk_m_'y', replace
}
*use junk_x_1989, replace
use junk_m_1989, replace
forvalues y=1990/2004 {
    display ['y']
    *merge setyr using junk_x_'y'
    merge setyr using junk_m_'y'
    tab _merge
    drop _merge
    order setyr
    sort setyr hs'y'
}
forvalues y=1989/2004 {
    egen i'y' = tag(setyr hs'y')
    replace hs'y'=. if i'y'==0
    drop i'y'
}
*now sort each column within setyr
egen xx = seq()
drop xx
reshape long hs, i(xx setyr) j(year)
sort year setyr hs
drop xx
egen xx = seq(), by(year)
reshape wide hs, i(xx setyr) j(year)
drop xx
*save setyr_x_1989_2004, replace
save setyr_m_1989_2004, replace

```