2.1 Personal Consumption Data

Personal Consumption at 92 Categories, in Current and Constant Dollars

Personal consumption data is taken from detailed unpublished NIPA data (H:\GBANKS\NIPA\UNPUB96$\AUNPUB) and aggregated to 92 categories both in current dollars and in 1996 dollars. The list of 92 categories can be found in the file \IDLIFT\MODEL\PCE.TTL, or in section 6 of this book. A 1987$ series is created for the estimation routine called *Symcon*. While a number of files are used to compile the consumption and price data, all are called when the user runs MAKEPCE.ADD. With *G7*, the process may be as simple a clicking "Run" on the menu. Before running the program, the user must carefully review MAKEPCE.ADD and the files it calls. Few changes will be needed to prepare the program for use, but two are critical. First, the paths guiding the program to data files and to other program files must be updated, as well as paths to output files. Second, dates specified by "tdates," "fdates," and "gdates," and also dates contained in "matpr" and similar commands, must be updated as new data becomes available. Carefully review all files before proceeding.

MAKEPCE.ADD constructs data for 92 types of consumption goods from NIPA data. Currently, this data is in 1996 dollars. This file, together with NIPA product listings and definitions, defines the product categories used in *Symcon* and in *LIFT*. After computing real and nominal consumption and prices for each sector, \IDLIFT\EQ\PCE\MAKEPCE.ADD calls MKCONDAT.ADD to create data in 1987 dollars. Next, the file MKCONDAT.SAV is created as a check file for the user to review. MKCODAT1.ADD and MKPDAT.ADD are called to create data files for consumption and prices. MAKEVAM.ADD is called to create a databank (Vam file) containing nominal and real consumption. Next, several macro variables are computed by calling \IDLIFT\MACROVAR\GETAPC.ADD to create average prices of consumption and by calling \IDLIFT\MACROVAR\PCEH.ADD to calculate macro variables for health care spending. Finally, \IDLIFT\MACROVAR\MKEXTRA.BAT and \IDLIFT\MACROVAR\DOP.BAT compile macro data and store it in a data bank; see Section 2.19 for how to create the EXTRA.DAT file and incorporate it into the MACRO.CBK databank. All of these steps and many more are executed by running MAKEPCE.ADD; see the file for more details.

Data for the Cross-Sectional Calculations

Several data files are needed to calculate the cross-sectional equations in Symcon and *LIFT*. CSCOEF.PRN contains demographic and income bracket coefficients; this file never needs to be changed. DEMOGS.DAT should be updated as new data becomes available. It includes the following demographic variables:

ncent	Region = North Central
south	Region = South

west	Region = West
college	Education of Household Head = College
twoy	Working Status of Spouse = Employed
fs1	Family Size = 1
fs2	Family Size = 2
fs5	Family Size = 5 and up
head1	Age of Head of Household < 35
head2	Age of Head of Household > 55

MKGPOP.DAT creates data series for eight population brackets, labeled GPOP1 to GPOP8. The file is created by MKGPOP.ADD using data from the \IDLIFT\MACROVAR\MACRO.CBK databank. The file must be given the following heading and terminated with a semi-colon:

```
# GPOPS.DAT
0 59 100 1 8 8
#Date   gpop1   gpop2   gpop3   gpop4  gpop5   gpop6   gpop7   GPOP8
1959.00 20.170  34.560  13.020  22.020 24.600  22.340  24.870  16.250
…
2000.00 18.870  39.690  19.900  36.370 41.850  42.530  41.280  4.840;
```

Aggregate income is divided into five income brackets by the program PCE\MKSLICE\MKSLICE.CPP. This program requires three data files to be stored in the PCE directory: population (PT.DAT), created by MKPT.ADD; total consumption expenditures in current prices (TOTEX.DAT), created by MKTOTEX.ADD; average expenditures per capita in 1987 dollars (AVGINC87.DAT), created by MKAVGINC.ADD; and a series of "alpha" parameters stored in ALPHA.DAT, forecast using Jeff Janoska's regression equation (see \IDLIFT\EQ\PCE\JEFF\ALPHA.REG). "totex", "pt," and "avginc" are read as vectors; hence, all headings and dates must be erased before proceeding. The "ADD" files listed above save the data with "NEW" extensions; the modified files must be saved with a "DAT" extension. Note that no comments are permitted in these files. PT.DAT, for example, contains a single column of data:

```
205060.000
…
275330.094
```

To run MKSLICE.CPP, change the parameter for the last year of data, and then recompile at the DOS prompt with "bump mkslice" and execute with "mkslice." The program also runs in Borland Builder. The result of running MKSLICE.CPP is \MKSLICE\SLICE.OUT, which must then be converted (by hand) to the "matdata" format and stored as SLICES.MAT in the PCE directory.

```
# SLICES.MAT
0 70 100 1 5 8
 1970    6313    1653     608      82       6
…
 2000    6413    2531    3357    3053    2054
```

Finally, the program CSTAR.CPP reads SLICES.MAT, DEMOGS.DAT, GPOPS.DAT, and CSCOEF.PRN and creates CSTAR.DAT and POPUL.DAT. The variables labeled "cstar" are initial guesses for "per capita" consumption for each product. These guesses are based only on demographic and income data. No price effects or dynamics are considered; these effects will be introduced in *Symcon* as a second stage. The estimates in fact are not per capita but rather "per popul," where "popul" is a weighted population variable calculated for each product. To run CSTAR.CPP, type "bump cstar," if the program has not yet been compiled, and then type "cstar." This program also runs in Borland Builder.

2.2 Estimating Consumption Functions with *Symcon*

A Brief Overview of the Consumption Equations

Symcon is a program that estimates the Perhaps Adequate Demand System (PADS) introduced by Almon in 1979 and 1996. Horst introduced a minor extension in 2002. Estimation is performed in two stages. In the first stage, cross sectional data is employed to estimate

$$C_i^* = \left(a_i + \sum_{k=1}^{K} b_{i,k} Y_k + \sum_{l=1}^{L} d_{i,l} D_l \right) \left(\sum_{g=1}^{G} w_{i,g} n_g \right)$$

where:
C^*_i = household consumption expenditures on good *i*
Y_k = the amount of per-capita income (expenditures within income category *k*
D_l = dummy variable used to show membership in the j^{th} demographic group
n_g = number of household members in age category *g*
w_g = adult equivalency weights (estimated parameters)
K = the number of income groups
L = the number of demographic categories
G = the number of age groups
a, b, d = estimated parameters

The dependent variable, "Cstar," is the product of two functions. The first includes a piecewise-linear Engle curve and a series of demographic dummy variables. The second is a weighted sum of family members; it allows unique weights to be assigned to members of each age group. If we divide both sides of the equation by the terms in the second set of parentheses, we see that this method is a generalization of per-capita estimation techniques.

Devine (1983) and Chao (1995) estimated the above equations. The adult equivalency weights were revised by Horst (2002). Li Ding will provide the next revision of these equations in forthcoming work. Current parameter estimates are contained in the file CSCOEF.PRN. While these estimates are dated, very likely there is

nothing for the user to revise when updating the consumption equations. Attention should be given instead to the next set of equations. See Section 2.1 for instructions on calculating Cstar using existing parameter estimates and new data.

The first stage ignored price effects and dynamics. These are introduced in the second stage, which is estimated with time series data. The consumption functions are

$$x_i(t) = \left(\alpha_i + \beta_i \frac{y(t)}{P(t)} + \sum_k^{K_i} \theta_{i,k} T_{i,k} \right) \prod_{n=1}^{N} p_n^{\delta_{i,n}}$$

$$P = \prod_{n=1}^{N} p_n^{s_n}$$

where $x_i(t)$ is per capita consumption of product *i* in period *t*; α, β, δ, and θ are parameters; y(t) is a measure of nominal per capita income or expenditures; and T are K_i additional variables important to product *i*. p_n is the price of product *n*; s_n is the budget share of product *n* in the base period; and P is the overall consumer price index. The estimated form of the model is

$$x_i(t) = \left(\alpha_i + \beta_i \frac{C_i^*(t)}{P(t)} + \phi_i \Delta \left[\frac{C_i^*(t)}{P(t)} \right] + \sum_k^{K_i} \theta_{i,k} T_{i,k} \right) \left(\frac{p_i}{P} \right)^{\lambda_i} \prod_{n=1}^{N} \left(\frac{p_i}{p_n} \right)^{-\lambda_n s_n} \left(\frac{p_i}{p_G} \right)^{-\mu_G} \left(\frac{p_i}{p_g} \right)^{-\nu_g}$$

where:
C^*_i = expenditure estimates from the first stage
p_G = the average price index of group *G*
p_g = the average price of subgroup *g*
λ_i = the individual good price response parameter
μ_G = the group price response parameter
ν_g = the subgroup price response parameter

Δ is the first-difference operator. Remaining parameters and variables are described above. In this form, consumption products have been organized into groups and subgroups to reduce the number of parameters. See Almon (1996) for derivation and additional description of these equations. While Inforum uses the two-stage approach in its *LIFT* model, disposable income or total consumption expenditures may be used instead of Cstar. Examples of additional linear terms (*T*) include a linear time trend, interest rates, and transfer payments.

Data for the Estimation of the Consumption Equations

Most of the data work needed to estimate the consumption functions was described in Section 2.1. Some editing of those files is required. Diligence is required since the estimation program will fail or will yield strange results if the data is not formatted correctly. In particular, the user will get strange results if the starting points are not specified correctly for each data series. Hence, a portion of each data file is

presented here. See Almon (1996), Horst (2002), and later sections of this manual for additional information.

Consumption data is prepared in MKCONDAT.ADD (see Section 2.1) and is contained in CONSUM.DAT. The format required by *Symcon* is

```
# CONSUM.DAT
92 sectors
30 years of data
1971 first year
1987 base year
#" Date"  "    pce1"  "   pce2"   "  pce3"  "   pce4"      ...  "  pce20"
1971.000   52258.945  28804.439   9108.316  20581.307      ...  13421.920
1972.000   50840.324  29633.695   9366.997  20420.533      ...  13584.856
...
2000.000   49619.402  32711.504  27060.127  26841.818      ...  10388.730
#
#"  Date"  "  pce21"  "   pce22"  " pce23"  "  pce24"      ...  "  pce40"
1971.000    8913.791   3898.473  17219.934  14147.279      ...  19445.283
...
2000.000   50777.707  21231.059  45974.867  21366.672      ...  10901.466
#
...
#
#"  Date"  "  pce81"  "   pce82"  " pce83"  "  pce84"      ...  "  pce92"
1971.000    4101.631   7758.426   3640.559   9527.256      ...  -1512.823
...
2000.000   15300.828  16144.146  11284.676  49216.137      ...  -8240.905
```

Price data is prepared in MKCONDAT.ADD and is contained in PRICES.DAT. The format required by *Symcon* is

```
# Prices
#" Date"  "cprices1" "cprices2" "cprices3" "cprices4"     ... "cprices20"
1971          0.4311     0.4293     0.5390     0.4011     ...     0.3225
1972          0.4757     0.4378     0.5424     0.4265     ...     0.3290
...
2000          1.2885     1.4764     1.3313     1.4839     ...     1.4422
#
...
#
#" Date" "cprices81" "cprices82" "cprices83" "cprices84"  ... "cprices92"
1971         0.4713      0.4492      0.4041      0.3996   ...     0.5486
2000         1.1467      1.6885      1.4849      1.5015   ...     1.0251
```

Population data is prepared in CSTAR.CPP and is contained in POPUL.DAT. The format required by *Symcon* is

```
98 Number of populations
#   wpop1   wpop2   wpop3   wpop4   wpop5   wpop6   wpop7   ...   wpop20
70    192     192     192     192     192     192     192   ...      165
71    194     194     194     194     194     194     194   ...      167
...
100   262     262     262     262     262     262     262   ...      225
#
...
#
#  wpop81  wpop82  wpop83  wpop84  wpop85  wpop86  wpop87   ...   wpop98
```

```
70    203     192     203     192     141     231     375   ...      189
...
100   273     250     273     250     192     301     432   ...      264
```

Cstar data is prepared in CSTAR.CPP and is contained in CSTAR.DAT. Note that the starting date is one year earlier than for other series to allow the program to calculate changes in Cstar. The format required by *Symcon* is

```
98 Number of cstar series
#   cstar1  cstar2  cstar3  cstar4  cstar5  cstar6  cstar7   ...  cstar20
70  381799  381799  381799  381799  381799  381799  381799   ...     5920
71  389328  389328  389328  389328  389328  389328  389328   ...     6174
...
100 706163  706163  706163  706163  706163  706163  706163   ...    23308
#
...
#
#  cstar81 cstar82 cstar83 cstar84 cstar85 cstar86 cstar87   ...  cstar98
70   10376   20746   21944   20746   23767    4006    6519   ...     3039
...
100  39791   97645  106518   97645   76080   18774   26935   ...     6428
```

The last data file need to run Symcon is TEMPI.DAT. The data in this file is prepared by TEMPI.ADD. Be careful to heed the warning printed by this file: correct and uncomment the number of linear terms, and then comment the title line. Failure to do this can cause many problems. Also, make sure data exists for the first year of each series. In addition to storing text data for use in *Symcon*, this program also prepares the IDTEMPI.SAV and TEMPI.* databanks required by *IdBuild* for use in *LIFT*; additional information about these files will be provided later. Here is a sample TEMPI.ADD file, which prepares five variables (called *T* in Equations 2.2 and 2.3). The variables are a linear time trend, the Treasury bill rate, hospital and medical insurance benefits, housing stock, and the "Cstar" estimate for electronics repair and rental.

```
#  TEMPI.ADD
vammode a
zap
tdates 1971 2000
fdates 1971 2000

cbk c:\idlift\macrovar\macro a
ba c:\idlift\eq\constr\constreg c
hbk h:\gbanks\NIPA\a96$\NIPAA d
add c:\idlift\eq\pce\cstar_mat.dat
add c:\idlift\eq\pce\consum_mat.dat
add c:\idlift\eq\pce\prices_mat.dat

# These variables are included on the rhs of equations
# in the next section: get them into the databank.
do{ f pce%1 = pce%1 }(1-92)
do{ f cst%1 = c.cst%1 }(1-19,25)
do{ f cstar%1 = cstar%1 }(1-98)
do{ f cprices%1 = cprices%1 }(1-92)
f rtb = rtb
f trphmi = trphmi

# These equations will be included in heart.cpp
```

```
save c:\idlift\eq\pce\idtempi.sav       # save the equations
do{ f tempo%1 = 0.0 }(1-30)             # DO NOT CHANGE THIS LINE!!
f tempo1  = 70 + @cum(tttt,1.,0.)       # A linear time trend.
f tempo2  = rtb                         # Treasury Bill Rate
f tempo3  = trphmi                      # Benefits: Hosp. and Medical Ins.
                                        # Housing Stock
f tempo4 = @cum(housStk,cst1+cst2+cst3+cst4,.03)/@cum(ub03,1.,.03)
f tempo5 = cstar98                      # Electronics repair and rental
save off

vam c:\idlift\model\novbase b  # A bit of trickery needed for IdBuild.
dvam b
do{ f pce%1 = b.pce%1 }(1-92)
do{ f cst%1 = b.cst%1 }(1-19,25)
dos copy c:\idlift\eq\pce\ws.* c:\idlift\eq\pce\tempi.*     # copy the database

save c:\idlift\eq\pce\tempi.dat         # save the data as text
ic 5 linear terms.  Need to uncomment this line, comment the titles line.
ic Make sure number above is correct, and fill in any missing information
ic in first lines below.  If the number of terms is >20, be sure to comment
ic out dates in second group.
matty 71 100 tempo1 1 tempo2 3 tempo3 tempo4 tempo5;
save off
```

The format required for TEMPI.DAT is

```
5 linear terms.  Need to uncomment this line, comment the titles line.
# Make sure number above is correct, and fill in any missing information
# in first lines below.  If the number of terms is >20, be sure to comment
# out dates in second group.
#
#  Date    tempo1     tempo2     tempo3       tempo4     tempo5
   1971    71.000      4.340      7.800   156451.125   3124.000
   1972    72.000      4.069      8.600   170335.594   3260.000
...
   2000   100.000      2.830    215.900   185077.812   6428.000
```

Configuration Files for the Estimation of the Consumption Equations

Three remaining files must be provided to run *Symcon*. The first two, GROUPS.TTL and SOFTCON.DAT, are described in Almon (1996), although several changes are described here.

A sample GROUPS.TTL file is

```
# Groups.ttl.  Columns are
# 1  The consumption category number
#    2  The group number
#       3  The subgroup number
#          4  The weighted population number to be used with this category
#             5  The income (Cstar) variable to be used with this category
#                6 Include in system (sensitive = 1)
#                  7 The title of the category
  1  1  1  1  1  1 Meat
  2  1  1  2  2  1 Dairy products
...
```

```
 15  0  0 15 15  1 Tobacco
 16  2  0 16 16  1 Footwear
 17  2  0 17 17  1 Clothing, Women's & girls'
...
 60  8  2 60 60  1 New autos
 61  8  2 61 61  1 Net purchases of used autos
...
 68  8  3 68 68  1 Mass transit
 69  8  3 69 69  1 Taxicab
...
 91  0  0 91 91  0 Less: exp in U.S. by foreigers
 92  0  0 92 92  0 Less: HH insur benefits
```

This file links each consumption category (column 1) to a particular Cstar (5) and Popul (4) series, or to other income and population data if the two-stage approach is not employed. The title of each consumption product is provided in column 7. Column 6 allows price effects to be ignored; that is, only the first part (the linear terms in Equation 2.3) is estimated for those goods. Products that are sensitive to price effects may be combined into groups (column 2) and subgroups (3).

The second configuration file required for *Symcon* is SOFTCON.DAT. The data in this file impose a version of the soft constraints described in Almon's The Craft of Economic Modeling (3th edition, 1994). For each parameter in the model except the constant, there is a pair of values: the first is a reasonable value for the parameter, and the second is a "magic number." A larger magic number gives more weight to the reasonable value and less weight to the data. In the original version of *Symcon*, as reported in Almon (1996), a magic number of "1" gives approximately equal weight to the constraint and to the data. This remains true for the income and price terms, but it is not true for the other linear terms. In the lines below, the first pair ({.2, 2}) is for the income parameter (Inc) for the first product (Meat). A small positive value (.2) is believed reasonable for the income parameter, and a fairly small magic number is sufficient to bring the actual parameter estimate "close" to .2. Clearly, "close" and "reasonable" are subjective terms, and the process of obtaining acceptable estimates for all parameters may require many iterations. Additional description of the process will follow. Constraints for any additional linear terms (*T* in Equations 2.2-2.3) are listed consecutively at the end of each line. If only the third term is included for product 3, for example, than constraints {0,0} must be provided for terms one and two. Mistakes and confusion can be avoided by providing constraints for all terms, as is shown here.

```
# SOFTCON.DAT
#sec Title   Inc Dinc  lamda   mu   nu   Ti1   Ti2   Ti3  Ti4  Ti5
1     Meat  .2 2 0 15  1   1  0  0  0 0  0 .1  0 0   0 0  0 0  0  0
2    Dairy  .1 1 0  0  .5 .5  0  0  0 0  0 .1  0 0   0 0  0 0  0  0
...
15 Tobacco  .1 1 0  1  .5  0  0  0  0 0  0 .1  0 0   0 0  0 0  0  0
16 Footwear  0 0 0  1  .5  0  0  0  0 0  0  0  0 0   0 0  0 0  0  0
17 Clo, W&g  0 0 0 .5  .5  0  0  0  0 0  0  0  0 0   0 0  0 0  0  0
...
60 New auto  1 5 0 .5  .4 .5 .1  1 -1 2  0 .1 -.1 1  0 0  0 0  0  0
61 Used car  0 0 0  0  .5 .4  0  0  0 0  0  0  0 0   0 0  0 0  0  0
...
68 M trans  .1 1 0  0   1  1  0  0  0 0  0  0  0 0   0 0  0 0  0  0
69 Taxicab  .1 1 0 10   1 15  0  0  0 0  0  6  0 0   0 0  0 0  0  0
...
```

```
91 Less:exp .1 0 0  1   0  0  0  0  0 0  0  0  0 0   0 0  0 0  0  0
92 Less: HH .1 0 0  0  .1  0  0  0  0 0  0  0  0 0   0 0  0 0  0  0
```

The final file required for estimating the consumption functions is TEMPI.CFG. The body of this file lists each product number followed by the index numbers of the linear terms included for that product. The syntax for each line is

```
<product number> <first term> [second term] [third term] [....]
```

where <> indicates required input and [] indicates optional input. In the following sample file, Term 1 is included for Product 15, and Terms 1 and 4 are included for Product 37.

Two commands may be given in this file. They are listed at the top here, but one or more of each command can be given at any point in the file. The first is the "check" command. As will be discussed later, the program automatically checks the income, change in income, and price parameters. For example, income parameters should be positive. If they are not, it is reported in the output file. The user may also request checking of the parameters for the linear terms, and he does so with the "check" command. The syntax for the command is

```
<trend number> <:> [lower bound] <,> [upper bound].
```

The first line in this file provides examples of the three types of checking this command provides. The line is

```
check 1:-3.0,3.0   2:,0.00   3:0.0,
```

indicating that the user should be notified if the following conditions are violated for any product:

```
-3.0  ≤  Parameter for Term 1  ≤  3.0
         Parameter for Term 2  ≤  0.0
 0.0  ≤  Parameter for Term 3.
```

The second command is called "pop." It indicates whether a linear term should be converted to per capita units before estimation. Recall that in the two-stage estimation process, "per capita" is really "per popul," where "popul" is a weighted population estimate unique to each product. Hence, such scaling must be done by the program and not by the user. Time trends and interest rates, for example, should not be converted. On the other hand, transfer payments, housing stock, and spending on electronics should be converted to per capita units. This can be done easily with the "pop" command. The syntax is

```
<pop> [Term i1] [Term i2] [Term i3 <-> Term ij] ...
```

where the "pop" command is followed by a listing of individual term numbers (e.g. 3) or a range of term numbers (e.g. 4-5). These examples correspond to the command provided in the following code. The format is flexible; any number of commands may be

given with any number of products or ranges of products listed for each command. A sample TEMPI.CFG, with all of these commands and features, is provided here:

```
# TEMPI.CFG
check  1:-3.0,3.0   2:,0.00   3:0.0,
check  4:0.0,    5:0.0,
pop    3 4-5
1      1
2      1
...
15     1
16     1
17     1
...
37     1      4
...
45     1      3
...
60     1      2
61     1      2
...
68     1
69     1
...
79     1      5
...
91     1
92     1.
```

Estimation of the Consumption Equations

Once the above files have been prepared, the user is ready to estimate the consumption functions (Equation 2.3). However, this manuals serves only as a guide for preparing the files and for other practical matters. The reader is encouraged to study the papers describing the development of PADS (Almon 1979 and 1996).

The five data files and three configuration files listed immediately above must be stored in the same directory as the program *Symbild2.exe* (this is a later version of the original estimation program *Symcon*). The program is run by entering `symbild2` at the DOS prompt. The program can be run in "debug" mode, where all input data is displayed on the screen, by typing

```
symbild2 d.
```

Typically the program will be run in batch mode to eliminate the need for continual user input. The command for batch mode is given by

```
symbild2 b [<iterations>] [<tolerance>]
```

where [] indicates optional input. If values are not supplied, a default maximum of 50 iterations will be performed before the program stops or asks for user input. Similarly, the default error tolerance is 5.0. Often it is easiest to insert such lines in a batch file. For example, the RUN.BAT file is called by typing

```
run [extension for output files]
```

where an optional extension for output files may be specified. For example, the output file TABLES.OUT may be renamed TABLES.1 to indicate it holds results for the first version of the model. Also copied are fitted and actual consumption series and the input files SOFTCON.DAT, TEMPI.DAT, and TEMPI.CFG. These input files are most likely to change in the development of the demand system.

```
rem run.bat
cls
symbild2 b 150 2
if "%1"=="" goto end
copy tables.out tables.%1
copy fits.dat fits.%1
copy softcon.dat softcon.%1
copy tempi.cfg tempi.%1
copy tempi.dat tempi.%1
:end
```

Examination and Interpretation of Estimation Results

Three output files are recorded when estimation is complete: FITS.DAT, CONSUM.PAD, and TABLES.OUT. FITS.DAT records actual and estimated per capita (actually per "popul") PCE for each product. These series are recorded in the "matdata" format and can be examined in *G7* with the program FITS.ADD. CONSUM.PAD records the parameters and other specifications for simulation in *LIFT*; this will be described later.

TABLES.OUT contains estimation results in readable form. Several new tools are introduced here to improve speed and accuracy of user input in the estimation process, but the user still must dedicate significant time to examining the results contained in this file. When problems are discovered (typically parameters of the wrong sign or magnitude), the user must alter the constraints or magic numbers in SOFTCON.DAT and perhaps alter the linear terms specified in TEMPI.CFG. After the changes have been made, Symcon must be run again and TABLES.OUT examined once more. This iterative process is speeded by putting "check" commands in TEMPI.CFG (as described above); any violation of these commands is reported here.

TABLES.OUT contains several sections. First, the *mu* and *nu* parameters, which determine cross-price elasticities, are reported, followed by parameters for income, own price, and the linear terms, as well as other specifications and statistics. These are described in Almon (1996). This output is followed by the results of automatic and user-specified problem checking. *Symcon* automatically checks whether elasticity with respect to income is positive and whether own-price elasticity is negative. Elasticity with respect to the change in income must be either positive or, if negative, smaller in magnitude than the parameter on income. Parameters for each linear term (*T* in Equations 2.2-2.3) must not violate the "check" conditions specified in TEMPI.CFG. All such violations are reported here. The user is left to make corrections.

Typically, the user includes relatively few linear terms for each product. In fact, all linear terms are included automatically for each product, but parameters for terms that are not included in TEMPI.CFG are forced to approximately zero with a soft constraint and a very large magic number {0, 999999999}. Occasionally, this magic number is too small, and the parameters for the excluded terms are not close to zero. In such cases, the product number, the linear terms, and the parameters are reported. If it is deemed necessary, the user can provide a larger magic number for those terms. The constraint itself (0.0) cannot be changed, and only magic numbers greater than the default (999999999) are read.

2.3 Forecasting and Simulation

Preparing for Simulation: Bringing Files into LIFT

A number of data and parameter files are needed to employ the estimates from *Symcon* in the general equilibrium *LIFT* model. The format and starting dates required of each data file differs for *Symcon* and *LIFT*. Because this frequently is a source of problems when performing simulations, as it is when estimating the equations, a detailed listing of each file is provided. Following the instructions on data preparation are a listing and description of other programs that must be run before running the *LIFT* model.

First, the file CONSUM.PAD must be copied to the \IDLIFT\MODEL\EQUATION\ subdirectory. The file is created by *Symcon* and should not be modified. The file CSCOEF.EQN must also be stored in this subdirectory. It is prepared from the CSCOEF.PRN file described above. It contains all the parameters listed there in the *Interdyme* "EQN" format. It rarely is modified. An example is shown here.

```
98 24 2000
cs   1  a 24
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
0.0  362.6  0.2259  0.0908   0.0666  0.0772  0.0396  -109.1   -81.5   ...
cs   2  a 24
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
0.0  362.6  0.2259  0.0908   0.0666  0.0772  0.0396  -109.1   -81.5   ...
...
cs  98  a 24
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
0.0  -7.00  0.0033       0   0.0023       0  0.0003    0.67    1.56   ...
```

The following files need to be copied to the \IDLIFT\MODEL\DAT\ subdirectory: CPRICES.DAT, PCE.DAT, PCECU.DAT, and SLICES.DAT. All data must begin in 1972 and must be in the "VMATDAT" or "VDATA" format; otherwise, problems will occur. If they are not in the proper format, then either alter the files by hand or read them into *G7* and print them in the proper style. Files are listed here for clarity.

SLICES.DAT must be in the “VMATDATA” format:

```
# SLICES.DAT
vmatdat r 1 29 1 5 7
slices  1972 1973 1974 1975 1976 1977 1978 1979 1980 ... 1999 2000
 1972    6345    1864     882     157      16
 1973    6365    2004    1040     171      14
 ...
 2000    6413    2531    3357    3053    2054;
```

PCE.DAT, PCECU.DAT, and CPRICES.DAT must be in the “VDATA” format:

```
# PCE.DAT
# 1 Meat
vdata pce1
    1972    50840.3   46300.7   47831.9   50325.8   52929.0
    ...
    1997    44195.6   47063.4   49536.7   49619.4
...
# 92 Less: HH insur benefits
vdata pce92
    1972    -1450.3   -1668.3   -1944.1   -2159.3   -2039.8
    ...
    1997    -6270.3   -7446.6   -7801.1   -8240.9
```

The following files need alteration as they are copied from the \IDLIFT\EQ\PCE directory to the \IDLIFT\MODEL\DATA directory: POPUL.DAT, TEMPI.DAT, CSTAR.DAT, and GPOPS.DAT copied as GPOPOLD.DAT with the variable name “gpopold,” and DEMOGS.DAT with the variable name “demog.” Earlier versions of *LIFT* also included GPOPS.DAT with the “gpop” variable name, but this file may not be used; you may wish to include it. A few lines of each are listed here for clarity:

```
# TEMPI.DAT
vmatdat r 1 29 1 20 5
tempi 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 ... 2000
#   Date     tempo1      tempo2    tempo3     tempo4     tempo5
    1972     72.000       4.069     8.600 170335.594   3260.000
    1973     73.000       7.027     9.700 174143.078   3381.000
...
    2000    100.000       2.830   215.900 185077.812   6428.000;


# POPUL.DAT
vmatdat r 1 29 1 20 3
popul 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 ... 2000
72      197     197     197     197     197     197     197       ... 169
...
100      262     262     262     262     262     262     262       ... 225;
...
vmatdat r 1 29 81 98 3
popul 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983  ... 2000
72      208     195     208     195     145     243     381       ...  193
...
100      273     250     273     250     192     301     432       ...  264;


# CSTAR.DAT
vmatdat r 1 29 1 20 3
```

```
cstar 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 ... 2000
72   401108  401108  401108  401108  401108  401108  401108  401108 ... 6566
...
100   706163  706163  706163  706163  706163  706163  706163  706163 ... 23308;
vmatdat r 1 29 21 40 3
cstar 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 ... 2000
72     6462    6579   19069   18838   22604  113393     414   30519   ... 14008
...
100    25185   26222   46933   48911   88499  184798    2869  137214 ... 32528;
...
vmatdat r 1 29 81 98 3
cstar 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 ... 2000
72    11677   23399   24906   23399   26166    4833    7592    4833   ...  3260
...
100    39791   97645  106518   97645   76080   18774   26935   18774  ... 6428;

# GPOPOLD.DAT
vmatdat r 1 29 1 8 8
gpopold 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 ... 1999 2000
#  Date   gpop1   gpop2   gpop3   gpop4   gpop5   gpop6   gpop7   gpop8
1972.000  17.100  39.950  20.300  33.390  23.540  23.730  30.870  21.020
...
2000.000  18.870  39.690  19.900  36.370  41.850  42.530  41.280  34.840;

# DEMOGS.DAT
vmatdat r 1 29 1 10 7
demog 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 ... 2000
#  Date  ncent   south  west  college   twoy   fs1   fs2   fs5  ...  head3
   1972  0.275   0.313 0.173    0.139  0.347 0.183 0.292 0.192  ...  0.366
   1973  0.274   0.316 0.175    0.143  0.354 0.185 0.302 0.183  ...  0.361
...
   2000  0.229   0.356 0.225    0.232  0.547 0.261 0.321 0.098  ...  0.349;
```

These data files also should be copied to the \IDLIFT\MODEL\HIS\ subdirectory with a "HIS" extension in place of the "DAT" extension. Once these files have been prepared, several other programs must be run. The first is \MODEL\MAKEVAM.BAT to load text data from the \MODEL\DATA\ directory into a VAM databank called HIST.VAM. Normally, no changes are necessary for either the MAKEVAM.ADD file (called by MAKEVAM.BAT) or the VAM.CFG file, which is used to reserve memory when constructing the VAM file. The portion of VAM.CFG corresponding to consumption is shown here for completeness.

```
#  Vectors for PADS Consumption Functions:
popul   98  1 0 pce.ttl                # Weighted populations
cstar   98  1 2 pce.ttl                # cstar terms
cprices 92  1 0 pce.ttl                # Consumption prices
demog   10  1 0 demog.ttl              # Demographic variables
gpopold  8  1 0 gpop.ttl               # Population groups used in PCE
slices   5  1 0 slices.ttl             # Income slices used in PCE
tempi   30  1 0 two.ttl                # time trends
```

Next, *IdBuild* must be run to prepare the macro variables and the macro equations. Equations for the "tempo" macro variables specified in TEMPI.ADD were saved in \EQ\PCE\IDTEMPI.SAV when TEMPI.ADD was run in *G7*. The data required for these equations were saved in the \EQ\PCE\TEMPI databank. These files are read by

IdBuild as specified in \IDLIFT\MODEL\MASTER. Several lines of the MASTER file are shown here:

```
# This includes the file to calculate linear terms in consumption
ba \idlift\eq\pce\tempi
isvector pce,cst,cstar
iadd \idlift\eq\pce\idtempi.sav
isvector clear
```

IdBuild can be run by typing

```
c:\pdg\idbuild c:\idlift\model\master
```

at the DOS prompt, where the path in the first segment indicates the location of the executable *IdBuild* program, and the second indicates the location of the MASTER file. The data are compiled into a single databank of macro variables, and the equations are translated into C++ code. They are stored as the function *idtempif()* in the HEART.H and HEART.CPP files. A function built into *LIFT*, called *Tempi()* and stored in TEMPI.CPP, calls the *idtempif()* function that the model builder constructs using *IdBuild*. The purpose of *Tempi()* is very simple: it copies the "tempo" macro variables defined in TEMPI.ADD and *idtempif()* to the "tempi" vector established in VAM.CFG. This is slightly confusing; fortunately, little adjustment by the user is required. However, if vector variables are employed on the right-hand side of equations in TEMPI.ADD (and consequently in TEMPI.SAV and *idtempif()*), then they must be listed after the "isvector" command in the MASTER file. Also, the vectors must be passed from the main *LIFT* program to the *Tempi()* function. Hence, the declaration of *Tempi()*, located in \IDLIFT\MODEL\IDLIFT.H; the function call, located in MODEL.CPP; and the function itself must be altered accordingly. The function declaration, for example, is

```
short Tempi(int ntempi, Vector& tempi, Vector& pce, Vector& cst,
      Vector& cstar);
```

indicating that elements from the pce, cst, and cstar vectors are employed in the equations. The integer "ntempi" and the vector "tempi" are always passed and the corresponding code should not be altered.

The final step required before running *LIFT* is to run \IDLIFT\MODEL\RUN.BAT (this is very different than \IDLIFT\EQ\PCE\RUN.BAT). The primary purpose of this program is to copy databanks to be used in the model and to run the fixer programs FIXER and MACFIXER. Of course, consumption demand and price variables can be fixed, but little discussion of these topics is provided here; see the IdLift User Guide (2001), the *Interdyme* manual, or other *Inforum* literature for discussion and instructions. Note, however, that the "alpha" variable used to construct "slices" currently is projected using a fix in \IDLIFT\MODEL\MACROFIX.MFX. Check to ensure that the model variables match those used by MKSLICE.CPP. Also, make sure that the rho fixes and other fixes have been updated. Finally, update \IDLIFT\MODEL\LASTDATA to inform the model of where the historical data ends for vector variables. Note that the last data point for PCE is specified in

\IDLIFT\MODEL\EQUATION\CONSUM.PAD (first line, second value). If you wish for PCE estimation to begin in another year, change this value and the value in LASTDATA.

Once these steps have been completed, run the model by typing *idlbild* at the DOS prompt. Consumption forecasts can be examined using \EQ\PCE\SHOWCONS.SHW. If several alternative simulations are to be compared, use \EQ\PCE\COMPRUNS.SH. If the forecasts or simulations seem unreasonable and the data is entered correctly, then adjust the soft constraints and repeat the estimation process and all following steps, or introduce or modify the fixes, execute \MODEL\RUN.BAT, and rerun the model.

2.4 REFERENCES

Almon, Clopper. "A System of Consumption Functions and its Estimation for Belgium." *Southern Economic Journal*, vol. 46, No. 1, pp. 85-106. July 1979.

---. "A Perhaps Adequate Demand System with Application to France, Italy, Spain, and the USA." Inforum Working Paper #96-007. 1996.

---. The Craft of Economic Modeling: Part 1. 3rd ed. Interindustry Economic Research Fund, Inc. 1994.

Chao, Chang-yu I. A Cross-Sectional and Time-Series Analysis of Household Consumption and a Forecast of Personal Consumption. Ph.D. dissertation. 1995.

Devine, Paul. Forecasting Personal Consumption Expenditures from Cross-Section and Time-Series Data. Ph.D. dissertation. 1983.

Horst, Ronald. forthcoming Inforum Working Paper. 2002.

IdLift User Guide Version 1.3. Interindustry Economic Research Fund, Inc. 2001.

"Interdyme: A Package of Programs for Building Interindustry Dynamic Macroeconomic Models." Inforum. 2000.