# Market analysis with mergersim 2
## *(Preliminary and incomplete)**

Jonas Björnerstedt and Frank Verboven

February 8, 2015

**Abstract**

In Björnerstedt & Verboven (2014) an implementation of merger simulation with the Stata package mergersim was introduced. In this paper extensions to the package are presented, implemented in the package mergersim2. The package mergersim2 implements more demand specifications, including three level nests and nest specific parameters in the nested logit, and the PCAIDS demand model of Epstein & Rubinfeld. Demand can be calibrated from a specification of demand elasticities. It also extends the merger analysis to include partial ownership and more general specifications of firm conduct. In addition upward pricing pressure (UPP) can be calculated as well as SSNIP tests. The package has also been extended to better allow for other kinds of market analysis.

Björnerstedt & Verboven (2014) introduced a Stata package, mergersim, to facilitate the quantative analysis of mergers using merger simulation methods. Other kinds of market analysis, relevant in competition policy analysis were not addressed by the program, however. With the development of mergersim2, the package now incorporates a new set of tools for market analysis. The purpose of this document is to show how the new features can be used, and to provide the underlying analysis. It is intended to be read in conjunction with Björnerstedt & Verboven (2014), which explains the basic functionality of mergersim.

The primary goal of mergersim was to create a platform for merger simulation. As demand analysis and the tools in the program could be used for other problems in market analysis in general and competition policy in particular, mergersim2 incorporates many changes in this direction. The new features focus on the analysis market power, pricing and extending the set of demand models of mergersim. There is also a more complicated example includet, that illustrates handling of uncertainty with a parametric bootstrap in the setting of merger simulations.

The new features of mergersim2 are presented in the next section. The following section contain presentation of the analysis and derivations used in

*Updated versions of the documentation are available at: `www.bjornersted.org/stata/mergersim2/new_features.pdf`.

these features. Sections 2 and 3 show how upward pricing pressure and SSNIP-tests are calculated in a multi-product setting. Analysis of demand with joint ownership is presented in section 4. The appendices document the derivations of three level nested logit demand, and the matrix algebra used for the share jacobian in merger simulation.

# 1 Using mergersim 2

The new features of mergersim2 are implemented in the framework of the previous version of the program, using new subcomands and new options to existing subcommands. It is an extension of mergersim, with complete backwards compatibility.

## 1.1 Program Status

The new version of mergersim is nearing completion. Currently there are a few new features that have not been completed. Updated versions can be installed as indicated below, and the latest version of the documentation is available at: `www.bjornersted.org/stata/mergersim2/new_features.pdf`.

## 1.2 Installation

The new program mergersim2 is implemented as a new version of mergersim, documented in Björnerstedt & Verboven (2014). Due to limitations of Stata, both programs *cannot* be installed on the same computer[1]. To install mergersim2, the following commands are executed in Stata:

```
net from http://www.bjornerstedt.org/stata/mergersim2
net install mergersim
```

Once installed, mergersim2 is updated with the Stata command:

```
adoupdate, update
```

## 1.3 Example market

Examples will be based on the example dataset of automobile sales in Europe cars1.dta, included with mergersim. The mergersim initialization and estimation commands will be the same for all examples. For expositional purposes we have chosen not to discuss questions of endogeneity of prices and group shares, using the Stata command `xtreg`. In any real application the usere must of course handle these issues properly, using for example `xtivreg` instead.

[1] If a previous version of mergersim has been installed, it can be removed before installing mergersim2 with the Stata command `ado uninstall mergersim`.

UPP
Unweighted averages by product and market

| | firm1 | firm2 |
|---|---|---|
| UPP | .5530756 | 1.126897 |
| UPP3 | .5530756 | 1.126897 |
| FP | 1.275078 | 1.918926 |
| Simulation | .5567027 | 1.132987 |

Figure 1: UPP results

```
mergersim init, nests(segment domestic) price(price) ///
  quantity(qu) marketsize(MSIZE) firm(firm)
xtreg M_ls price M_lsjh M_lshg horsepower fuel width height domestic ///
  year country2-country5, fe
```

Market analysis and simulation will in all cases be on the year 2008, in Germary (country 3). For example mergersim market is constrained to this market by the command:

```
mergersim market if year == 1998 & country == 3
```

## 1.4 UPP and Consumer Surplus

When performing a merger simulation with mergersim2, the Upward Pricing Pressure (UPP) can be also be calculated, using the `upp` option to `mergersim simulate`. With the command

```
mergersim simulate if year == 1998 & country == 3 , ///
seller(15) buyer(26) detail upp
```

The output of `mergersim simulate` is given by Figure 7. Note that in the current version the absolute price change is reported in the table, not percentage changes.

The detail option specifies that market shares before and after the merger are also displayed. In addition, the Herfindahl (HHI), C4 and C8 concentration ideces are included. With mergersim2 the consumer and producer surplus are also included in this detailed output, as shown in Figure 2.

The code used can be found in example2.do.

## 1.5 Market analysis

Simulation of market outcomes can be of importance in many other contexts, however. Changes in product characteristics or marginal costs, for example by the imposition of a sales tax. In the context of competition policy, one can wish

| | Pre-merger | Post-merger |
|---|---|---|
| HHI: | 1501 | 1972 |
| C4: | 66.07 | 71.50 |
| C8: | 86.21 | 88.01 |

| | Change |
|---|---|
| Consumer surplus: | -1,839,750 |
| Producer surplus: | 1,303,353 |

Figure 2: Concentration indeces and surplus

to study for exampel abuse of dominance in analyzing to what extent raising rivals' costs increases sales and profits of the dominant firm.

Given a demand specification, `mergersim simulate` can be used to study the equilibrium effect of a change in market conditions. In mergersim2 the options `buyer()` and `seller()` are now optional, only to be used if changes in ownership are to be studied. The command `mergersim simulate` can be used not only to study the effects of changes in ownership, but also the effects of for instance

1. Changes in costs, using the `newcosts()` option
2. Changes in the property of products, such as horsepower, using the `newdelta()` option
3. Introduction or removal of products
4. Changes in conduct, using the `newconduct()` option

We will illustrate the first three features, with a market simulation after both cost and product changes, and with a firm exiting the market.

The command `mergersim market` calculates various variables that can be used in market analysis.

M_costs contains the marginal costs per product, calculated based on the estimated demand and assumptions on market competition. They are the costs that would result in observed prices and demand.

M_delta contains the average utility of each product in each market excluding the price effect. In the current example it includes the effect of horsepower, fuel, width and height, as well as market specific constants.

To generate these variables in the market to be studied, after initializing mergersim and estimating demand as discussed above, the following command is used:

```
mergersim market if year == 1998 & country == 3
```

To increase costs by 5%, a new variable is generated

```
gen costs2 = 1.05*M_costs
```

Similarly, we create a new variable for the new utility of each product. Here an increase in horsepower of 50% for VW results in higher utility for these products. It is a product of the following terms:

1. the increase in horsepower: `horsepower*0.5`
2. the estimated marginal effect of horsepower on utility: `_b[horsepower]`
3. a dummy for VW: `(firm==26)`

```
gen delta2 = M_delta + horsepower*0.5*_b[horsepower]*(firm==26)
```

Once these variables have been created, a market simulation can be performed

```
mergersim simulate if year == 1998 & country == 3 & firm != 4, ///
  newdelta(delta2) newcosts(costs2)
```

By excluding firm 4 from the simulation, Fiat exits the market. The new costs and utility are included with the options on the second line of the command. The results of the simulation are shown in Figure 3.

The code used can be found in example2.do.

## 1.6 SSNIP and Analysis of Demand

After estimating or specifying demand parameters, mergersim can perform a SSNIP test. With costs calculated based on the market competition assumption, the profitability of a price increase of a set of products can be evaluated. To perform a SSNIP test of a 5 percent price increase, the following command is used:

```
mergersim demand if year == 1998 & country == 3 , ssnip(0.05)
```

The resultant output is given in Figure 4.

### 1.6.1 Analysis of Demand

The estimated or specified demand function can be studied with mergersim2. With the `mergersim demand` command, one can study the non-equilibrium effect on demand of a set of prices. To see the effect of a 5 percent price increase in prices of products in Germany, the following commands can be used

Merger Simulation

| | Buyer | Seller |
|---|---|---|
| Firm | | |
| Marginal cost savings | 0 | 0 |

Simulation method: Newton
Periods/markets: 1
Number of iterations: 6
Max price change in last it: .00001

Prices
Unweighted averages by firm

| firm code | Pre-merger | Post-merger | Relative change |
|---|---|---|---|
| BMW | 17.946 | 18.565 | 0.035 |
| Ford | 13.093 | 13.482 | 0.027 |
| Honda | 15.778 | 16.523 | 0.046 |
| Hyundai | 12.912 | 13.454 | 0.041 |
| Kia | 11.276 | 11.738 | 0.040 |
| Mazda | 14.229 | 14.819 | 0.041 |
| Mercedes | 20.114 | 20.634 | 0.029 |
| Mitsubishi | 15.832 | 16.487 | 0.041 |
| Nissan | 15.101 | 15.850 | 0.047 |
| GM | 19.921 | 20.718 | 0.036 |
| PSA | 16.397 | 17.108 | 0.043 |
| Renault | 15.292 | 15.977 | 0.045 |
| Suzuki | 9.225 | 9.588 | 0.039 |
| Toyota | 13.019 | 13.569 | 0.041 |
| VW | 17.182 | 18.638 | 0.089 |
| Volvo | 22.149 | 23.224 | 0.048 |
| Daewoo | 13.483 | 14.061 | 0.042 |

Variables generated: M_price2 M_quantity2 M_price_ch (Other M_ variables dropped)

Figure 3: Market simulation with cost, utility and product changes

```
. mergersim demand if year == 1998 & country == 3, ssnip(0.05)

Profits prior and after price increase
```

| firm code | (sum) M_profit | (sum) M_profit3 |
|---|---|---|
| BMW | 745,382 | 988,017 |
| Fiat | 328,022 | 400,018 |
| Ford | 777,287 | 955,204 |
| Honda | 81,092 | 104,869 |
| Hyundai | 38,844 | 50,455 |
| Kia | 20,261 | 26,175 |
| Mazda | 178,098 | 227,652 |
| Mercedes | 1,522,057 | 1,828,528 |
| Mitsubishi | 106,710 | 138,946 |
| Nissan | 175,887 | 223,457 |
| GM | 1,411,487 | 1,760,316 |
| PSA | 243,041 | 306,396 |
| Renault | 390,740 | 471,796 |
| Suzuki | 38,877 | 50,298 |
| Toyota | 185,975 | 233,975 |
| VW | 3,241,081 | 3,804,455 |
| Volvo | 94,982 | 122,965 |
| Daewoo | 41,239 | 54,464 |
| Total | 9,621,060 | 11747986 |

```
Variables generated: M_quantity3 M_profit M_profit3
```

Figure 4: SSNIP test, firm profits

Prices and quantities prior and after price change

| firm code | (mean) price | (mean) newp | (sum) qu |
|---|---|---|---|
| BMW | 17.94551 | 18.84278 | 231820 |
| Fiat | 15.33784 | 16.10473 | 133697 |
| Ford | 13.09282 | 13.74746 | 298808 |
| Honda | 15.77822 | 16.56713 | 36298 |
| Hyundai | 12.91196 | 13.55755 | 18756 |
| Kia | 11.27581 | 11.8396 | 9868 |
| Mazda | 14.22884 | 14.94028 | 78876 |
| Mercedes | 20.11445 | 21.12017 | 312500 |
| Mitsubishi | 15.83209 | 16.62369 | 48134 |
| Nissan | 15.10065 | 15.85568 | 79487 |
| GM | 19.921 | 20.91705 | 521581 |
| PSA | 16.39682 | 17.21666 | 107815 |
| Renault | 15.29199 | 16.05659 | 158788 |
| Suzuki | 9.224823 | 9.686064 | 18635 |
| Toyota | 13.01877 | 13.66971 | 83910 |
| VW | 17.18183 | 18.04092 | 940416 |
| Volvo | 22.14901 | 23.25646 | 38854 |
| Daewoo | 13.48337 | 14.15754 | 19822 |

Figure 5: Demand

```
generate newprice = 1.05*princ if country == 3
mergersim demand if country == 3 , price(newprice)
```

The generate command creates a new price variable with German prices five percent higher. The mergersim demand command generates the table in Figure 5.

The code is in example2.do.

## 1.7 Demand specification in mergersim2

Nesting in mergersim has been extended to three level nests.

Nest specific parameters.

The rather tedious derivation of shares, equation to be estimated and share jacobian for three level nests are given in Appendix B.

### 1.7.1 PCAIDS demand

Epstein & Rubinfeld (2002) introduced a demand model that could be calibrated to a two specified elasticities and market demand. The PCAIDS demand specification simply assumes that diversion to other brands is in proportion to the market shares of that brand. In mergersim2 demand can be specified as PCAIDS, using the `demand()` option of `mergersim init`. The following code

```
Merger Simulation

                                    Simulation method: Newton
                      Buyer    Seller   Periods/markets: 1
Firm                  1        2        Number of iterations: 5
Marginal cost savings 0        0        Max price change in last it: 1.2e-07

Prices
Unweighted averages by firm
```

| firm | Pre-merger | Post-merger | Relative change |
|---|---|---|---|
| 1 | 1.000 | 1.138 | 0.138 |
| 2 | 1.000 | 1.108 | 0.108 |
| 3 | 1.000 | 1.041 | 0.041 |

Figure 6: Merger with PCAIDS demand

replicates the calculations of Epstein & Rubinfeld (2002, p 895). It is assumed that there are three firms each producing a single good, all with initial price $p = 1$. Prices and value shares $q$ are given by the following table:

| firm | p | q |
|---|---|---|
| 1 | 1 | 0.2 |
| 2 | 1 | 0.3 |
| 3 | 1 | 0.5 |

Elasticities are given by . The following code generates the dataset and evaluates a merger between firms 1 and 2.

```
matrix data = (1, 1, 0.2 \ 2, 1, 0.3 \ 3, 1, 0.5 )
matrix colnames data = firm p q
svmat data , names(col)

mergersim init, demand(pcaids) elasticities(-1 -3) price(p) quantity(q) firm(firm)
mergersim simulate , buyer(1) seller(2)
```

The results are shown in Figure 6.

## 1.8 Elasticities

The new version of mergersim can calculate elasticities for price increases within and between groups as specified by a grouping variable in the dataset. The segment variable in the example cars1 dataset for example contains information on

Group elasticities based over variable: segment

| | c1 | c2 | c3 | c4 | c5 |
|---|---|---|---|---|---|
| r1 | -.437361 | .0158611 | .0158611 | .0158611 | .0158611 |
| r2 | .0334207 | -.6119596 | .0334207 | .0334207 | .0334207 |
| r3 | .0308328 | .0308328 | -.8092644 | .0308328 | .0308328 |
| r4 | .0144502 | .0144502 | .0144502 | -1.045716 | .0144502 |
| r5 | .0212202 | .0212202 | .0212202 | .0212202 | -1.311138 |

Figure 7: Calculated group elasticities

the car segment of each car brand. To see how percentage changes for all products in one segment affect demand in different car segments, the groupelasticities option can be used as shown below.[2]

```
mergersim market if year == 1998 & country == 3 , ///
groupelasticities(segment)
```

The resulting output of group elasticities by car segment for Germany in 1998 is shown in Figure 7.

The elasticity matrix is saved in the r(groupelasticities) matrix in the output returned by mergersim market. Product level elasticites are saved in the r(elas) matrix. The code used can be found in example2.do.

### 1.8.1 Calibration from elasticities

An alternative to estimating the nested logit parameters $\alpha$ and $\sigma_i$ is calibration - to explicity specify them. In `mergersim market` the following code is used.

```
mergersim market if year == 1998 & country == 3, elasticities(-5.5 0.4 0.03)
```

For a two level nested logit demand model as above, three elasticities are specified:

1. the average own price elasticity
2. the average cross price elasticity with respect to another product in the same subgroup
3. the average cross price elasticity with respect to another product in the same group

Note that as the elasticities depend on market shares, the $\alpha$ and $\sigma_i$ that correspond to a given set of elasticities will vary by market. See Appendix C for a derivation. For this reason calibration from `elasticity` requires the specification of a single market, in the example above Germany (country 3) in 1998.

[2] In the syntax of Stata, /// at the end of a line indicates that the command continues on the next line.

### 1.9 Estout support

The estout package provides a simple way of creating tables of results.[3] By specifying the option `estout([name])` in `mergersim init`, results of estimation, market analysis and merger simulation are stored. To present tables of stored results, the estout can be used as indicated by the table.

| **Description** | **estout command** |
|---|---|
| Display all estimates | estout |
| Display estimates starting with C | estout C* |
| Display price changes | estout , cells(price_ch) |
| Display elasticities | estout , cells(elasticities) |

## 2 Merger simulation and uncertainty

In this section we will illustrate how to obtain confidence intervals on a merger simulation using a parametric bootstrap. In order to do this, a little more knowledge of Stata programming is required. Specifically, matrices, local macros, temporary files, and a programming loop (forwhile) are used. See the Stata Users Guide for more information on these topics.

After initializing and estimating, mergersim market is invoked to create matrices of the estimate (M_demand_b) and variance-covariance matrix (M_demand_V) for the three parameters that will be randomly sampled: $\alpha$, $\sigma_1$ and $\sigma_2$.

```
mergersim init, nests(segment domestic) price(price) ///
  quantity(qu) marketsize(MSIZE) firm(firm)
xtreg M_ls price M_lsjh M_lshg horsepower fuel width height domestic year country2-country5
mergersim market
```

Now two temporary files are created, one with the market to be studied, the other an empty dataset to which each simulation will be appended.

```
keep if year == 1998 & country == 3
tempfile market bootstrap
save ‘market’
drop if 1 // Save empty dataset
save ‘bootstrap’ , replace
```

A matrix params is created with ndraws random draws (here set to 100) using the matrices created by mergersim market. Setting a seed for the pseudo random number generator ensures that the same draws are used, enabling reproduction of the calculations.

```
local ndraws 100
set seed 1
drawnorm alpha sigma1 sigma2, n(‘ndraws’) means(M_demand_b) cov(M_demand_V) clear
mkmat alpha sigma1 sigma2, matrix(params)
```

[3] To install the estout package, use the command `ssc install estout`.

Using the *ndraws* random draws, merger simulations for each are appended to the temporary dataset *bootstrap*.

```
forvalues i = 1/‘ndraws’ {
use ‘market’ , clear
local alpha = params[‘i’,1]
local sigma1 = params[‘i’,2]
local sigma2 = params[‘i’,3]

quietly mergersim init, nests(segment domestic) price(price) quantity(qu) ///
marketsize(MSIZE) firm(firm) alpha(‘alpha’) sigmas(‘sigma1’ ‘sigma2’)
quietly mergersim simulate , seller(15) buyer(26)

quietly append using ‘bootstrap’
quietly save ‘bootstrap’ , replace
}

tabstat M_price_ch , by(firm) statistics(mean sd) format(%9.3f)
```

# 3 Joint ownership in merger simulation

In this section we discuss how one can handle issues of joint ownership in mergersim2. Market analysis or mergers where there is joint ownership complicates the analysis. We will discuss briefly how joint ownership can be handled conceptually, and then present the implementation in mergersim2.

Let $p_m$, $q_m$ and $c_m$ denote vectors of prices, quantities and marginal costs per product. Calculation of costs and equilibrium prices in a market are based on the first order condition of profit maximization.

$$R_m \odot D_m \left(p_m - c_m\right) + q_m = 0$$

The matrix $D_m$ is the demand jacobian where element $(i, j)$ is the derivative of the demand for good $i$ with respect to price $j$. The ownership matrix $R_m$ in market $m$ identifies which products belong to the same firm, with element $(i, j)$ set to 1 if they have the same owner, otherwise zero.

The ownership matrix $R$ is created from the firm variable specified by the user by first creation of a binary product matrix $F$, with rows for each product and columns for each firm, with a 1 if the firm owns the product. For example, with 3 products and 2 firms the matrix

$$F_m = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

indicates that firm 2 owns two goods. We then have

$$R_m = F_m F_m'$$

Assume that the market has three firms, with a product each. If firm is owned by firm 1 and 2 with shares$\alpha$ and $1-\alpha$ respectively. Partial ownership can be incorporated by letting the ownership materix depend on product ownership $F_m$ and firm ownership $S$ matrices:

$$R_m = F_m S F_m'$$

In the simple setting with three firms each owning one product, we have $R_m = S$. The element $(i, j)$ of $S$ indicates to what extent firm $i$ includes the effect of profits of firm $j$ in its profit maximization.

For an owning firm, if profits of the subsidiary are split according to ownership shares, the revenues of the subsidiary should enter the maximization problem with the same weight $\alpha$. With joint ownership, how profits are maximized in the jointly owned firm matters. Three different assumptions come to mind.

- With *total profit maximization,* prices of the jointly owned firm are set to maximize the total profit, the idea being that with an optimal contract between owners such pricing would be chosen. (Whether such contracting is possible or legal is another question.)

$$S_T = \begin{pmatrix} 1 & 0 & a \\ 0 & 1 & 1-a \\ 1 & 1 & 1 \end{pmatrix}$$

- With *weighted profit maximization* the subsidiary maximizes the weighted sum of the profit functions of itself and the owners , with weights corresponding to ownership shares.

$$S_W = \begin{pmatrix} 1 & 0 & a \\ 0 & 1 & 1-a \\ a & 1-a & 1 \end{pmatrix}$$

With *own profit maximization* the subsidiary maximizes its own profits.

$$S_O = \begin{pmatrix} 1 & 0 & a \\ 0 & 1 & 1-a \\ 0 & 0 & 1 \end{pmatrix}$$

In the following, we will focus on *weighted* profit maximization.

With more complicated ownership structures than in the example above, creating an ownership matrix becomes more complicated. Consider the ownership shares of 4 goods and 3 firms. Product 3 is jointly owned by 1 and 2, and product 4 is owned by 1 and 3 as in the following table

| | **1** | **2** | **3** |
|---|---|---|---|
| **1** | 1 | | |
| **2** | | 1 | |
| **3** | a | 1-a | |
| **4** | b | | 1-b |

The question is whether indirect ownership should be taken into account. With weighted profit maximization, defining the ownership matrix as above, we obtain

$$S_W = \begin{pmatrix} 1 & 0 & a & b \\ 0 & 1 & 1-a & 0 \\ a & 1-a & 1 & 1-b \\ b & 0 & 1-b & 1 \end{pmatrix}$$

Here the effect that firm 4 has on firm 2 is ignored by firm 4, as the ownership is indirect. In order for this effect to be taken into account, the corresponding cell has to be changed:

$$S_{W'} = \begin{pmatrix} 1 & 0 & a & b \\ 0 & 1 & 1-a & 0 \\ a & 1-a & 1 & 1-b \\ b & \mathbf{(1-a)(1-b)} & 1-b & 1 \end{pmatrix}$$

## 3.1 Generation of ownership in mergersim

In order to implement joint ownership in a simple way, it is advantageous to distinguish between products ownership by firms and the ownership of other firms by firms. The $p \times f$ product matrix $F$ defines which firm (or subsiduary) that produces the product, created from the firm categorical variable in the dataset. The $r \times r$ firm matrix $S$ defines joint ownership.

In mergersim 1 the firm ownership matrix $S$ is implicitly an identity matrix, unless the conduct option is specified. In this case, the off-diagonal elements of $S$ are set to the value $\theta$ that the user specifies.

There are two methods of creating joint ownership in mergersim 2. The ownership matrix can be created by the user as a Stata matrix with the name M_ownership. If a matrix with this name exists it will be automatically used by mergersim. Alternatively mergersim can assist in the creation of the matrix. Joint ownership is specified in a Stata dataset with three variables in the following order.

| Firm | Owner | Share |
|---|---|---|
| 3 | 1 | $\alpha$ |
| 3 | 2 | $1-\alpha$ |

Several joint ownerships can be specified in the dataset. Mergersim replaces the off-diagonal zero elements$(3,1)$ and $(1,3)$ in the identity matrix with $\alpha$ and $(3,2)$ and $(2,3)$ with $1-\alpha$. Ownership is specified in mergersim market with the option `ownership([name of dataset])`.

If the structure of joint ownership changes over time, this can be also be handled. Assume that the ownership shares of firm 3 changes to $\beta$ and $1-\beta$ at some time $t$ in the data. Let a new owner 4 replace firm 3 as owner in all observations after $t$ and use the ownership dataset

| Firm | Owner | Share |
|---|---|---|
| 3 | 1 | $\alpha$ |
| 3 | 2 | $1-\alpha$ |
| 4 | 1 | $\beta$ |
| 4 | 2 | $1-\beta$ |

Such a new firm can also be used to model a merger from partial ownership to complete ownership by setting $\beta = 1$ above, letting firm 4 aquire firm 3 in the merger.

A conduct parameter can be specified in mergersim to capture the degree of competition between firms. Specifying conduct $\theta$ replaces all off-diagonal elements in the ownership matrix $S$ by $\theta$. The ownership dataset can also be used to specify different conduct parameters, rather than joint ownership. To see what happens if firm 1 and 2 take the profit of the other firm into account by $\theta$, we use the following ownership dataset:

| Firm | Owner | Share |
|---|---|---|
| 2 | 1 | $\theta$ |

The effect of this specification is that only elements$(1,2)$ and $(2,1)$ of $S$ are set to $\theta$.

The ownership matrix is used to specify how firms take the profits of other firms into account and the product ownership is used to specify which firm owns what. Assumptions on conduct thus affect the ownership matrix, with the possibility of pre- and post merger assumptions.

- Note that the implementation of joint ownership in mergersim2 has not yet been completed.

# Appendix

In the appendix, calculations of muliproduct UPP, shares and share Jacobian for three level nests and calibration from elasticities are presented.

## A Multiproduct UPP

Let $\Delta q\left(p\right)$ be the demand jacobian for all products of all firms, $D_{11}$ be the submatrix of $\Delta q\left(p\right)$ of elements where $i$ and $j$ are products of firm 1 and $D_{12}$ be the submatrix of terms where $i$ is a product of firm 1 and $j$ is a product of firm 2. (As all derivatives will be at pre-merger prices, the dependence of these submatrices on $p$ is surpressed.) The FOC for the set of products of firm 1 can be expressed as

$$D_{11}(p_1 - c_1) + q_1 = 0 \tag{1}$$

As above, subscripts indicate sub-vectors of the products that belong to firm 1 prior to the merger. If firm 1 buys firm 2, these equations are modified as follows

$$D_{11}(p_1 - (1 - e_1)c_1) + D_{12}(p_2 - c_2) + q_1 \tag{2}$$

At pre-merger prices this expression will not be zero. Subtracting, we can define the vector:

$$UPP_{12} = -e_1 c_1 - D_{11}^{-1} D_{12}(p_2 - c_2)$$

The invertability of $D_{11}$ is guaranteed by the existence of a unique pre-merger equilibrium. The $j \times k$ matrix

$$Div_{12} = -D_{11}^{-1} D_{12}$$

are the diversion ratios for each product $j$ of firm 1 and $k$ of firm 2.

We can, along the lines of Schmalensee also define

$$UPP_{12}^{*} = -e_1 c_1 - D_{11}^{-1} D_{12}(p_2 - (1 - e_2)\, c_2)$$

### A.1 Interpretation

Let be the optimal price conditional on $p_2$ and other firms

$$D_{11}(p_1 + \Delta p_1 - (1 - e_1)c_1) + D_{12}(p_2 - c_2) + q_1 = 0$$

Substituting $q_1$ from (1), we get

$$D_{11}(\Delta p_1 + e_1 c_1) + D_{12}(p_2 - c_2) = 0$$

Thus

$$UPP_{12} = \Delta p_1$$

$UPP_{12}$ is thus the optimal price change for the products of firm 1, given that prices of other products including the products formerly owned by firm 2 are unchanged. The measure is the absolute price change rather than the relative change measured in merger simulations, as the focus is on the direction of change rather than the magnitude.

## A.2 UPP and first iteration of fixed point

The fixed point algorithm solves the following equation for prices post-merger:

$$\begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix} \begin{pmatrix} p_1 - c_1 \\ p_2 - c_2 \end{pmatrix} + \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = 0$$

The FOC for other products are solved seperately as ownership creates a block diagonal matrix $D$.

Block inverting the matrix $D$, the prices of the pre-merger products in the first iteration of the fixed point algorithm is:

$$p_1 = (1-e_1)c_1 - \left(D_{11} - D_{12}D_{22}^{-1}D_{21}\right)^{-1} q_1 - D_{11}^{-1}D_{12}\left(-D_{22} + D_{21}D_{11}^{-1}D_{12}\right)^{-1} q_2$$

The difference between UPP and the first iteration for firm 1 is that for the new products, UPP takes the prices as given by the pre-merger level, whereas the first stage fixed point calculates these as being optimal given the prices of other firms.

The difference between these expressions are given by the second terms within the two parenthesises with inverses ($D_{12}D_{22}^{-1}D_{21}$ and $D_{21}D_{11}^{-1}D_{12}$). This can be seen as follows. Setting these to zero and using the fact that the pre-merger FOC for products 2 can be written as

$$D_{22}\left(p_2 - c_2\right) = -q_2$$

we get the UPP equation above.

## A.3 Average UPP

Assuming that firm 1 has the same marginal costs for all products $c_1 = \bar{c}_1 i_1$, and firm 2 has the same markup $\bar{m}_2 i_2 = p_2 - c_2$ on all products, where$i_k$ is a column vector of ones corresponding to the number of products of firm $k$, we can define the vector:

$$-\frac{(i_1' D_{11} i_1)}{J_1}\bar{c}_1 e_1 - \frac{(i_1' D_{12} i_2)}{J_1}\bar{m}_2$$

where $J_1$ is the number of products of firm 1. Under these assumptions we can get an average UPP that depends on a scalar diversion ratio.

$$AUPP_{12} = -\bar{c}_1 e_1 - \frac{(i_1' D_{12} i_2)}{(i_1' D_{11} i_1)}\bar{m}_2 = -\bar{c}_1 e_1 - \frac{(i_1' D_{12} i_2)}{(i_1' D_{11} i_1)}\bar{m}_2$$

We can then define the diversion ratio as

$$Div_{12} = -\frac{i_1' D_{12} i_2}{i_1' D_{11} i_1}$$

Both $UPP_{12}$ and $AUPP_{12}$ correspond to the single market $UPP_{12}$ when both firms 1 and 2 are single product firms. Note that

$$Div_{12} = -\frac{i_1' \Delta q\left(p\right) i_2}{i_1' \Delta q\left(p\right) i_1}$$

if vectors $i_k$ are redefined as containing 1 if product $j$ belongs to $k$, otherwise 0.

# B Nested logit derivations

This section presents derivations for share jacobians and log share values used in estimation,. for the two- and three-level nested logit demand specification.

$$I_j = \exp\left(\delta_j / \left(1 - \sigma_h\right)\right)$$

$$I_{h(j)} = \sum_{k \in H_j} I_k$$

$$I_{g(j)} = \sum_{h \in G_j} \left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}}$$

$$I = \sum_{g \in G} I_g^{1-\sigma_g}$$

Shares in a two level nested logit are given by:

$$s_j = \frac{\exp\left(\delta_j / \left(1 - \sigma_h\right)\right)}{I_{h(j)}} \frac{\left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}}}{I_{g(j)}} \frac{I_{g(j)}^{1-\sigma_g}}{1+I} \tag{3}$$

Note that considering an outer $\sigma_f = 0$ makes the equation symmetric in each fraction above. Summing products of the subgroup $h(j)$ we get the share of the subgroup:

$$s_{h(j)} = \frac{\left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}}}{I_{g(j)}} \frac{I_{g(j)}^{1-\sigma_g}}{1+I} \tag{4}$$

Similarly

$$s_{g(j)} = \frac{I_{g(j)}^{1-\sigma_g}}{1+I} \tag{5}$$

Thus

$$s_j = \frac{s_j}{s_{h(j)}} \frac{s_{h(j)}}{s_{g(j)}} s_{g(j)} = s_{jh} s_{hg} s_g$$

By dividing the previous equations we obtain the share of subgroup $h(j)$ of the group $g(j)$ that it belongs to:

$$s_{hg} = \frac{\left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}}}{I_{g(j)}}$$

## B.1 Two level nested Jacobian

$$\frac{dI_{h(j)}}{d\delta_j} = \frac{1}{1-\sigma_h} I_j$$

$$\frac{dI_{h(j)}}{d\delta_j} I_{h(j)}^{-1} = \frac{1}{1-\sigma_h} s_{jh}$$

$$I_{g(j)} = \sum_{h \in G_j} \left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}}$$

$$\frac{dI_{g(j)}}{d\delta_j} = \frac{1-\sigma_h}{1-\sigma_g} \left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}} I_{h(j)}^{-1} \frac{dI_{h(j)}}{d\delta_j} = \frac{1}{1-\sigma_g} s_{jh} \left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}}$$

$$\frac{dI_{g(j)}}{d\delta_j} I_{g(j)}^{-1} = \frac{1}{1-\sigma_g} s_{jh} s_{hg} = \frac{1}{1-\sigma_g} s_{jg}$$

from the definition of $s_{hg}$ above.

To reduce the number of terms in taking derivatives we rewrite

$$s_j = \exp\left(\delta_j / \left(1-\sigma_h\right)\right) \left[\left(I_{h(j)}^{1-\sigma_h}\right)^{\left(\frac{1}{1-\sigma_g} - \frac{1}{1-\sigma_h}\right)}\right] \left[I_{g(j)}^{-\sigma_g}\right] \frac{1}{1+I}$$

Let $F_i\left(\delta_j\right)$ denote each of the four terms in this product. Then

$$\frac{\partial F_1\left(\delta_j\right)}{\partial \delta_j} = \frac{1}{1-\sigma_h} F_1$$

$$\frac{\partial F_2\left(\delta_j\right)}{\partial \delta_j} = \frac{\partial I_{h(j)}}{\partial \delta_j} \left(1-\sigma_h\right) \left(\frac{1}{1-\sigma_g} - \frac{1}{1-\sigma_h}\right) I_{h(j)}^{-1} F_2$$

Using the derivative $\frac{dI_{h(j)}}{d\delta_j} I_{h(j)}^{-1}$ above:

$$\frac{\partial F_2\left(\delta_j\right)}{\partial \delta_j} = s_{jh} \left(\frac{1}{1-\sigma_g} - \frac{1}{1-\sigma_h}\right) F_2$$

Using the derivative $\frac{dI_{g(j)}}{d\delta_j} I_{g(j)}^{-1}$ above:

$$\frac{\partial F_3\left(\delta_j\right)}{\partial \delta_j} = -\sigma_g \frac{\partial I_{g(j)}}{\partial \delta_j} I_{g(j)}^{-1} F_3 = \frac{-\sigma_g}{1-\sigma_g} s_{jg} F_3$$

$$\frac{\partial F_4\left(\delta_j\right)}{\partial \delta_j} = -\left(1-\sigma_g\right) I_{g(j)}^{1-\sigma_g} I_{g(j)}^{-1} \frac{\partial I_{g(j)}}{\partial \delta_j} F_4^2 = -s_{jg} \left(I_{g(j)}^{1-\sigma_g} F_4\right) F_4 = -s_j F_4$$

The share derivative is given by the product rule:

$$\frac{\partial s_j}{\partial \delta_j} = \sum_{i=1}^{4} \left(\frac{\partial F_i\left(\delta_j\right)}{\partial \delta_j} \prod_{k \neq i} F_k\right)$$

As each $\frac{\partial F_i}{\partial \delta_j}$ contains $F_i$, we can factor out the product $s_j$, and obtain

$$\frac{\partial s_j}{\partial \delta_j} = \left(\frac{1}{1-\sigma_h} - \left(\frac{1}{1-\sigma_h} - \frac{1}{1-\sigma_g}\right) s_h - \frac{\sigma_g}{1-\sigma_g} s_{jg} - s_j\right) s_j$$

## B.2 Three level nest jacobian

To reduce the number of terms in taking derivatives we rewrite

$$s_j = \exp\left(\delta_j / \left(1-\sigma_h\right)\right)\left[\left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}-1}\right]\left[\left(I_{g(j)}\right)^{\frac{1-\sigma_g}{1-\sigma_f}-1}\right]\left[I_{f(j)}^{-\sigma_f}\right]\frac{1}{1+I}$$

Let $F_i\left(\delta_j\right)$ denote each of the five terms in this product. Then $F_1$ and $F_2$ are defined as above with

$$\frac{\partial F_1\left(\delta_j\right)}{\partial \delta_j} = \frac{1}{1-\sigma_h}F_1$$

$$\frac{\partial F_2\left(\delta_j\right)}{\partial \delta_j} = s_{jh}\left(\frac{1}{1-\sigma_g}-\frac{1}{1-\sigma_h}\right)F_2$$

We can calculate the derivative $\frac{dI_{m(j)}}{d\delta_j}I_{m(j)}^{-1}$ as above:

$$I_{f(j)} = \sum_{h\in F_j}\left(I_{g(j)}\right)^{\frac{1-\sigma_g}{1-\sigma_f}}$$

$$\frac{dI_{f(j)}}{d\delta_j} = \frac{1-\sigma_g}{1-\sigma_f}\left(I_{g(j)}\right)^{\frac{1-\sigma_g}{1-\sigma_f}}I_{g(j)}^{-1}\frac{dI_{g(j)}}{d\delta_j} = \frac{1}{1-\sigma_f}s_{jg}\left(I_{g(j)}\right)^{\frac{1-\sigma_g}{1-\sigma_f}}$$

$$\frac{dI_{f(j)}}{d\delta_j}I_{f(j)}^{-1} = \frac{1}{1-\sigma_f}s_{jf}$$

$$\frac{\partial F_2\left(\delta_j\right)}{\partial \delta_j} = \frac{\partial I_{h(j)}}{\partial \delta_j}\left(1-\sigma_h\right)\left(\frac{1}{1-\sigma_g}-\frac{1}{1-\sigma_h}\right)I_{h(j)}^{-1}F_2$$

$$\frac{\partial F_2\left(\delta_j\right)}{\partial \delta_j} = s_{jh}\left(\frac{1}{1-\sigma_g}-\frac{1}{1-\sigma_h}\right)F_2$$

$$\frac{\partial F_3\left(\delta_j\right)}{\partial \delta_j} = s_{jg}\left(\frac{1}{1-\sigma_f}-\frac{1}{1-\sigma_g}\right)F_3$$

Changing index

$$\frac{\partial F_4\left(\delta_j\right)}{\partial \delta_j} = -\sigma_f\frac{\partial I_{f(j)}}{\partial \delta_j}I_{f(j)}^{-1}F_4 = \frac{-\sigma_f}{1-\sigma_f}s_{jf}F_4$$

$$\frac{\partial F_5\left(\delta_j\right)}{\partial \delta_j} = -\left(1-\sigma_f\right)I_{f(j)}^{1-\sigma_f}I_{f(j)}^{-1}\frac{\partial I_{f(j)}}{\partial \delta_j}F_5^2 = -s_{jf}\left(I_{f(j)}^{1-\sigma_f}F_5\right)F_5 = -s_jF_5$$

The share derivative is given by the product rule:

$$\frac{\partial s_j}{\partial \delta_j} = \sum_{i=1}^{5}\left(\frac{\partial F_i\left(\delta_j\right)}{\partial \delta_j}\prod_{k\neq i}F_k\right)$$

Insering the definitions of $F_i$ and $\frac{\partial F_i(\delta_j)}{\partial \delta_j}$ and simplifying, we obtain

$$\frac{\partial s_j}{\partial \delta_j} = \left(\frac{1}{1-\sigma_h}-\left(\frac{1}{1-\sigma_h}-\frac{1}{1-\sigma_g}\right)s_{jh}-\left(\frac{1}{1-\sigma_g}-\frac{1}{1-\sigma_f}\right)s_{jg}-\frac{\sigma_f}{1-\sigma_f}s_{jf}-s_j\right)s_j \tag{6}$$

(As each $\frac{\partial F_i}{\partial \delta_j}$ contains $F_i$, we can factor out the product $s_j = \prod_k F_k$.)

## B.3 Log-share Parameters Used in Estimation

To obtain the three level linear equation to estimate, start from the share equation (3)

$$s_j = \frac{\exp\left(\delta_j/\left(1-\sigma_h\right)\right)}{I_{h(j)}} \frac{\left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}}}{I_{g(j)}} \frac{I_{g(j)}^{1-\sigma_g}}{1+I}$$

above

$$s_j/s_0 = \frac{\exp\left(\delta_j/\left(1-\sigma_h\right)\right)}{I_{h(j)}} \frac{\left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}}}{I_{g(j)}} I_{g(j)}^{1-\sigma_g}$$

Using

$$\exp\left(\delta_j/\left(1-\sigma_h\right)\right) = \exp\left(\delta_j\right)\exp\left(\delta_j\frac{\sigma_h}{1-\sigma_h}\right)$$

we get

$$s_j/s_0 = \exp\left(\delta_j\right)\exp\left(\delta_j\frac{\sigma_h}{1-\sigma_h}\right)\left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}-1} I_{g(j)}^{-\sigma_g}$$

Multiplying by $I_{h(j)}^{-\sigma_h} I_{h(j)}^{\sigma_h}$

$$s_j/s_0 = \exp\left(\delta_j\right)\left[\exp\left(\frac{\delta_j}{1-\sigma_h}\right) I_{h(j)}^{-1}\right]^{\sigma_h} I_{h(j)}^{\sigma_h}\left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}-1} I_{g(j)}^{-\sigma_g}$$

$$s_j/s_0 = \exp\left(\delta_j\right)\left[\exp\left(\frac{\delta_j}{1-\sigma_h}\right) I_{h(j)}^{-1}\right]^{\sigma_h}\left[\left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}} I_{g(j)}^{-1}\right]^{\sigma_g}$$

Substituting the definitions

$$s_j/s_0 = \exp\left(\delta_j\right) s_{jh}^{\sigma_h} s_{hg}^{\sigma_g}$$

The population equation to be estimated is given by

$$\log\left(s_j/s_0\right) = \delta_j + \sigma_h \log s_{jh} + \sigma_g \log s_{hg}$$

### B.3.1 Three level nests

In the three level nestm the corresponding equation to (3) is given by

$$s_j = \exp\left(\delta_j/\left(1-\sigma_h\right)\right)\left[\left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}-1}\right]\left[\left(I_{g(j)}\right)^{\frac{1-\sigma_g}{1-\sigma_f}-1}\right]\left[I_{f(j)}^{-\sigma_f}\right]\frac{1}{1+I}$$

$$s_j = \frac{\exp\left(\delta_j/\left(1-\sigma_h\right)\right)}{I_{h(j)}} \frac{\left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}}}{I_{g(j)}} \frac{\left(I_{g(j)}^{1-\sigma_g}\right)^{\frac{1}{1-\sigma_f}}}{I_{f(j)}} \frac{I_{f(j)}^{1-\sigma_f}}{1+I}$$

$$s_j/s_0 = \exp\left(\delta_j\right)\exp\left(\delta_j\frac{\sigma_h}{1-\sigma_h}\right)\left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}-1}\left(I_{g(j)}\right)^{\frac{1-\sigma_g}{1-\sigma_f}-1} I_{f(j)}^{-\sigma_f}$$

$$s_j/s_0 = \exp(\delta_j)\left[\exp\left(\delta_j \frac{\sigma_h}{1-\sigma_h}\right) I_{h(j)}^{-\sigma_h}\right] I_{h(j)}^{\sigma_h} \left(I_{h(j)}\right)^{\frac{1-\sigma_h}{1-\sigma_g}-1} \left(I_{g(j)}\right)^{\frac{1-\sigma_g}{1-\sigma_f}-1} I_{f(j)}^{-\sigma_f}$$

$$s_j/s_0 = \exp(\delta_j)\left[\exp\left(\delta_j \frac{1}{1-\sigma_h}\right) I_{h(j)}^{-1}\right]^{\sigma_h} \left[\left(I_{h(j)}^{1-\sigma_h}\right)^{\frac{1}{1-\sigma_g}} I_{g(j)}^{-1}\right]^{\sigma_g} \left[\left(I_{g(j)}^{1-\sigma_g}\right)^{\frac{1}{1-\sigma_f}} I_{f(j)}^{-1}\right]^{\sigma_f}$$

$$s_j/s_0 = \exp(\delta_j)\, s_{jh}^{\sigma_h} s_{hg}^{\sigma_g} s_{gf}^{\sigma_f}$$

Thus the equation to be estimatedis given by

$$\log(s_j/s_0) = \delta_j + \sigma_h \log s_{jh} + \sigma_g \log s_{hg} + \sigma_f \log s_{gf}$$

# C Elasticities

This section contains the analysis of nested logit demand and elasticities. To calibrate nested logit model from elasticities, the functional dependence of parameters $\alpha$ and $\sigma_i$ on these elasticities have to be derived. Expressions for group elasticites, used in mergersim market are also presented. Note that calibration from elasticites in the three level nested logit are not supported yet.

## C.1 Calibration from elasticities

With a one level nested logit, elasticities calculated from quantities are given by:

$$e_{jj} = \alpha\left(\frac{1}{1-\sigma} - \frac{\sigma}{(1-\sigma)}\frac{q_j}{q_{g_j}} - \frac{q_j}{m_1}\right) p_j$$

$$e_{jk} = -\alpha\left(\frac{\sigma}{(1-\sigma)}\frac{q_j}{q_{g_j}} + \frac{q_j}{m_1}\right) p_j$$

Let

$$\bar{z} = \frac{1}{J}\sum_j \frac{p_j q_j}{q_{g(j)}} = \frac{1}{J}\sum_{g\in G}\frac{\bar{v}_g}{\bar{q}_g}$$

Unweighted average elasticities are then

$$\bar{e}_{jj} = \alpha\left(\frac{1}{1-\sigma}\bar{p} - \frac{\sigma}{(1-\sigma)}\bar{z} - \frac{1}{m_1}\bar{v}\right)$$

$$\bar{e}_{jk} = -\alpha\left(\frac{\sigma}{1-\sigma}\bar{z} + \frac{1}{m_1}\bar{v}\right)$$

Defining $\bar{p}$, $A$ and $B$ as the three sums above

$$C_1 = \frac{\alpha}{1-\sigma} = \frac{\bar{e}_{jj} - \bar{e}_{jk}}{\bar{p}}$$

and thus

$$\bar{e}_{jk} = -\sigma C_1 \bar{z} - (1-\sigma) C_1 \bar{z}$$

$$\bar{e}_{jk} + C_1 \bar{v} = \sigma C_1 \left( \bar{v} - \bar{z} \right)$$

The one level nested logit parameters can thus be expressed in terms of shares

$$\sigma = \frac{\bar{e}_{jk}/C_1 + \bar{v}}{\bar{v} - \bar{z}} = \left( 1 - \frac{\bar{p}}{\bar{v}} \frac{\bar{e}_{jk}}{\bar{e}_{jk} - \bar{e}_{jj}} \right) \frac{\bar{v}}{\bar{v} - \bar{z}}$$

$$\alpha = (1 - \sigma) C_1 = \frac{C_1 \bar{z} - \bar{e}_{jk}}{\bar{v} - \bar{z}}$$

$$-\alpha = \frac{\bar{e}_{jk}}{\bar{v} - \bar{z}} \left( 1 + \frac{\bar{e}_{jj} - \bar{e}_{jk}}{\bar{e}_{jk}} \frac{\bar{z}}{\bar{p}} \right)$$

### C.1.1 Two level nests

Basing calculations on shares in the two level unit demand logit, we have:

$$\bar{e}_{jj} = \alpha \left( \frac{1}{1 - \sigma_1} \bar{p} - \left( \frac{1}{1 - \sigma_1} - \frac{1}{1 - \sigma_2} \right) p \cdot s_{hg}/J - \frac{\sigma_2}{1 - \sigma_2} p \cdot s_g/J - p \cdot s/J \right)$$

$$\bar{e}_{jk} = -\alpha \left( \left( \frac{1}{1 - \sigma_1} - \frac{1}{1 - \sigma_2} \right) p \cdot s_{hg}/J + \frac{\sigma_2}{1 - \sigma_2} p \cdot s_g/J + p \cdot s/J \right)$$

$$\bar{e}_{jl} = -\alpha \left( \frac{\sigma_2}{1 - \sigma_2} p \cdot s_{hg}/J + p \cdot s/J \right)$$

Let $Z_g = p \cdot s_g/J$, $Z_{hg} = p \cdot s_{hg}/J$, and $Z_j = p \cdot s/J$.

$$\bar{e}_{jj} = \alpha \left( \frac{1}{1 - \sigma_1} \bar{p} - \left( \frac{1}{1 - \sigma_1} - \frac{1}{1 - \sigma_2} \right) Z_{hg} - \frac{\sigma_2}{1 - \sigma_2} Z_g - Z_j \right)$$

We now have

$$C_1 = \frac{\alpha}{1 - \sigma_1} = \frac{\bar{e}_{jj} - \bar{e}_{jk}}{\bar{p}}$$

and

$$\bar{e}_{jl} - \bar{e}_{jk} = \alpha \left( \frac{1}{1 - \sigma_1} - \frac{1}{1 - \sigma_2} \right) Z_{hg}$$

$$\bar{e}_{jl} - \bar{e}_{jk} = \left( C_1 - \frac{\alpha}{1 - \sigma_2} \right) Z_{hg}$$

$$C_2 = \frac{\alpha}{1 - \sigma_2} = C_1 - \frac{\bar{e}_{jl} - \bar{e}_{jk}}{Z_{hg}}$$

$$\bar{e}_{jl} = -\sigma_2 \frac{\alpha}{1 - \sigma_2} Z_g - \frac{\alpha}{1 - \sigma_2} (1 - \sigma_2) Z_j$$

$$\bar{e}_{jl} = -\sigma_2 C_2 Z_g - C_2 (1 - \sigma_2) Z_j$$

$$\bar{e}_{jl}/C_2 + Z_j = -\sigma_2 Z_{gg} + \sigma_2 Z_j$$

Thus the solution is

$$\sigma_2 = \frac{\bar{e}_{jl}/C_2 - Z_j}{Z_j - Z_g}$$

$$\alpha = (1 - \sigma_2) C_2$$
$$\alpha = (1 - \sigma_1) C_1 = (1 - \sigma_2) C_2$$
$$\sigma_1 = 1 - \frac{(1 - \sigma_2) C_2}{C_1}$$

### C.1.2 CES

For CES demand, the $C_1$ and $Z_*$ parameters above have to be modified as follows

$$\bar{e}_{jj} = \alpha \left( \frac{1}{1 - \sigma_1} - \left( \frac{1}{1 - \sigma_1} - \frac{1}{1 - \sigma_2} \right) s_{hg}/J - \frac{\sigma_2}{1 - \sigma_2} s_g/J - s/J \right) - 1$$

In this case

$$C_1 = \bar{e}_{jj} - \bar{e}_{jk} + 1 = \frac{\alpha}{1 - \sigma}$$

Also $Z_{jh} = \sum_j s_{jh}/J$, $Z_{hg} = \sum_j s_{hg}/J$, and $Z_j = \sum_j s_j/J$. In the formulas above $\bar{p}$ is replaced with 1.

## C.2 Group elasticities

Group elasticities are calculated based on a categorical variable that defines which group a product belongs to. The variable is transformed to a $g \times n$ group membership matrix $G$, where element $(i, j)$ is 1 if product $j$ is in group $i$. The matrix of product elasticities can be expressed in terms of the share Jacobian $D$ as:

$$E = \hat{p} D' \hat{s}^{-1}$$

where the $\hat{}$-operator is used to indicate diagonalizing a vector. Similarly, the group elasticities are given by

$$E_g = G \hat{p} D' G' \widehat{Gs}^{-1}$$