Documentation for Within-Module Estimation (runmicroblp.m)

Main Steps

1. Data work (**step0_datawork.m**)
   - Output dimensions of data for diagnostic purposes
2. Linear estimation (step1_microblp with options set to only run linear estimation - **step1_microblp**(0,0,0))
   - Output linear estimates
3. Test non-linear objective function (step1_microblp with options set to only test objective function - **step1_microblp**(0,0,1))
   - Output calculation times
4. Check non-linear gradient functions (step1_microblp with options set to check gradient function - **step1_microblp**(0,1,1))
   - Output calculation times and gradient check results
5. Run non-linear estimation (step1_microblp with options set to check gradient function - **step1_microblp**(1,0,1))
   - Output calculation times and non-linear estimation results

1 Data work (step0_datawork.m)

1. Imports data or simulates and cleans simulated data (**0ai_generatedata**.m and **0aii_cleansimulateddata**.m)
2. Imposes sample limits:
   (a) Keeps a random subsample of `maxT` markets, with priority to markets that we observe households shopping in.
   (b) Aligns identifiers across datasets sample limits imposed (**0b_alignidentifiers**.m).
   (c) Exits if we have too few markets (less than 5) with any non-outside goods.
3. Preps data for estimation:
   (a) Generates "draws" - quadrature points - to represent each market income distribution (**0c_generatedraws**.m).
   (b) Demeans income so that linear parameters can be compared across models with and without non-linear parameters.
   (c) Differences data from outside good and drops outside good observations.
   (d) Generates, rescales and selects (through pca transformation) instruments (including price instruments and brand dummies).
   (e) Dimension work required for non-linear estimation
      i. Generate subscript matrices
      ii. Group markets into roughly equally sized sets of observations for sequential or parallel processing of independent components of constraint and Hessian matrices.

   - Note: Product and brand counts are calculated *including* the outside good.

1.1 Simulation Code (0ai_generatemicrodata.m):

$$u_{ijt} = \exp\left((\alpha_i \gamma_i (\beta_j + \xi_{jt}) - \alpha_i \log p_{jt})\right)$$

where prices are $p = (\beta_j + \xi_{jt}) - min(\beta_j + \xi_{jt}) + u^1_{jt}$ and price instruments are defined as $z = u^2_{jt} + |u^1_{jt}|$, for $\log u^1_{jt} \sim N(0.1, 1)$, $u^2_{jt} \sim N(0, 0.1)$, and $\xi_{jt} \sim N(0, \sigma_{jt})$.

Shares are calculated as:

$$v_{jt} = s_{jt} = \frac{1}{N_t} \sum_i \frac{u_{ijt}}{\sum_{j \in J_t} u_{ijt}}$$

1.2 Cleaning Simulated Data (0aii_cleansimulateddata.m):

- Sales shares are defined as

$$s_{jt} = \frac{1}{N_t} \sum_i \frac{v_{jt}}{1 + \sum_{j \in J_t} v_{jt}}$$

  - including the outside good observations

- Price censoring:
  - Observations with prices greater than 5 standard deviations from the mean within a product are removed from the sample. Markets where these prices were observed are removed from the sample.
  - Observations with prices greater than 5 standard deviations from the mean within a module are removed from the remaining sample. Markets where these prices were observed are removed from the sample.

- Outside good:
  - Products whose average sales share across markets where they are sold is in the lowest 10th percentile for the sample are allocated to the "outside good."
  - All products in the brand with the lowest average sales shares across markets where it is sold are allocated to the "outside good."
  - Markets that don't sell any "outside good" product or at least two "inside good" products are dropped from the sample.
  - Price, instrument, and share data for the outside good is calculated as the average value for the (non-logged) price and price instruments and the sum of the observed market shares.
  - The outside good is allocated brand 1.

1.3 Identifier creation (0b_alignidentifiers.m):

- Make sure that market, product and brand identifiers are 1:T, 1:J, and 1:B and aligned across market and household data.
- Drop markets that don't sell an outside good or whose product count is under 2.
- Accumulate and append outside good data as one observation per market.
  - Note:

$$s_{ijt} = \frac{u_{ijt}}{\sum_{j \in J_t} u_{ijt}} = \frac{u_{ijt}}{\sum_{j \in J_t^O} u_{ijt} + \sum_{j \in J_t^I} u_{ijt}}$$

    * Suppose that $\beta_{jt} = \beta_{0t}$ for all $j \in J_t^O$ and $P_{0t} = \left[\sum_{j \in J_t^O} (p_{jt})^{-\alpha_i}\right]^{-\frac{1}{\alpha_i}}$, then:

* Therefore, we have that the log share of any product j relative to the outside good is:

$$\ln s_{ijt} - \ln s_{i0t} \quad = \quad \ln \tilde{u}_{ijt}$$

where $\tilde{u}_{ijt} = u_{ijt} / \left(\sum_{j \in J_t^O} u_{ijt} \right) = \exp \left(\alpha_i \gamma_i \left(\beta_{jt} - \beta_{0t} \right) - \alpha_i \left(\log p_{jt} - \log P_{0t} \right) \right)$.

* We can also express the share of any product j in terms of relative product qualities and prices since:

$$s_{ijt} \quad = \quad \frac{\tilde{u}_{ijt}}{1 + \sum_{j \in J_t^I} \tilde{u}_{ijt}}$$

* In practice, we proxy for P_{0t} with a sales-weighted average of the prices paid in market t.
* All data is differenced from market fixed effects in order to adjust for differing quality and measurement error of prices of the outside good across markets.

2 Within-Module Estimation Sub-Program (step1_microblp.m)

This program loops around the 4 models to be estimated and, depending on the input arguments (objfuntest, checkgradients, and nlestimate), performs some cumulative subset of the following:

1. Generates linear estimates (**step2_thetalin_twostep**.m)
2. Sets starting values (**step3_startandrescale.m**)
   (a) Makes sure that elasticities (`alpha0, alpha1, gamma`) are within bounds.
   (b) Times objective function (**step 4a_gmmobjective.m**) and check that it is defined at starting values. Set Knitro objective function scaling.
3. Identifies deltas implied by starting elasticity values (**step3a_alldeltas**, **step3b_solvefordeltas**).
4. Runs non-linear estimation, calculate efficient weighting matrix, and re-run estimation with this weighting matrix (**step4a_gmmobjective**).
   (a) If corner solution or negative exit flag, re-estimate using a grid of starting values (i.e., manual multistart).
5. Outputs estimates and standard errors.

2.1 Linear estimation (step2_thetalin_twostep.m)

* Variables:
  - B_{jt} = data.bdums(:,2:end): one brand dummy for each brand except the outside good
  - $\tilde{x}_{jt} = x_{jt} - \bar{x}_t$ data.relx: demeaned log price
  - $\tilde{z}_{jt} = z_{jt} - \bar{z}_t$ data.relz: rescaled differenced (log) price instruments
  - $\tilde{y}_t = y_t - \ln(37, 500)$ data.dmmktlninc: “demeaned” market log income
  - σ_t^y = data.vlninc: standard deviation of income within a market t
  - Z_{jt} = data.relZ: dummies for all, including, outside good (acts as a constant) + differenced (log) price instruments in level then interacted with mkt mean income, mkt mean income^2, and mkt sd(income)
    * If $K_{jt} = \begin{bmatrix} B_{jt} & \tilde{z}_{jt} \end{bmatrix}$, $Z_{jt} = \begin{bmatrix} K_{jt} & K_{jt} y_t & K_{jt} y_t^2 & K_{jt} \sigma_t^y \end{bmatrix}$
  - A = data.relA = (data.relZ’ data.relZ)^-1

- Estimation procedure:
  - For simplest case, $\alpha_1, \gamma = 0$: $\widetilde{\ln s_{jt}} = \alpha^0 \widetilde{\beta_j} - \alpha^0 \widetilde{\ln p_{jt}} + \varepsilon_{jt}$ where $\varepsilon_{jt} = \left(\widetilde{\delta_{jt}} - \widetilde{\beta_{jt}}\right) \nu_{jt}$ and $\widetilde{x_{jt}} = x_{jt} - \bar{x}_t$
    * ivest = **ivregression**(rellnsjt, relX, data.relZ, data.relA)
      · $relX_{jt} = \begin{bmatrix} B_{jt} & \widetilde{x_{jt}} \end{bmatrix}$ = relX: dummies for all but outside good + differenced log pricerelx includes dummies for all but the outside good (and no constant)
      · relZ and relA are as defined above.
    * $\hat{\alpha}_0^{lin}$=-ivest(end,1)
    * $\hat{\beta}_j^{lin}$=[ivest(1:(end-1))-ivest(1))/alpha0]
  - For case where $\alpha_1, \gamma \neq 0$: $\widetilde{\ln s_{jt}} = \alpha_t \gamma_t \widetilde{\delta_{jt}} - \alpha_t \widetilde{\ln p_{jt}} + \varepsilon_{jt}$ where $\alpha_t = \alpha^0 + \alpha^1 \tilde{y}_t$ and $\gamma_t = 1 + \gamma \tilde{y}_t$
    * Step 1: $\widetilde{\ln s_{jt}} = \alpha^0 \widetilde{\beta_j} - \alpha^0 \widetilde{\ln p_{jt}} + \varepsilon_{jt}$ where $\varepsilon_{jt} = \left(\widetilde{\delta_{jt}} - \widetilde{\beta_{jt}}\right) \nu_{jt}$ and $\widetilde{x_{jt}} = x_{jt} - \bar{x}_t$
      · ivest = ivregression(rellnsjt, relX, data.relZ, data.relA)
      · $relX_{jt} = \begin{bmatrix} B_{jt} & \widetilde{x_{jt}} \end{bmatrix}$ = relX: dummies for all but outside good + differenced log pricerelx includes dummies for all but the outside good (and no constant)
      · relZ and relA are as defined above.
      · $\hat{\alpha}_0^{lin}$=-ivest(end,1)
      · $\hat{\beta}_j^{lin}$=[ivest(1:(end-1))-ivest(1))/alpha0]

$$\begin{aligned}\widetilde{\ln s_{jt}} &= (1 + \gamma \tilde{y}_t)\left(\hat{\alpha^0} + \alpha^1 \tilde{y}_t\right) \hat{\tilde{\beta}}_j - \left(\hat{\alpha^0} + \alpha^1 \tilde{y}_t\right) \widetilde{\ln p_{jt}} + \mu_{jt} \\ &= \hat{\alpha^0}\left(\hat{\tilde{\beta}}_j - \widetilde{\ln p_{jt}}\right) + \gamma \tilde{y}_t \hat{\alpha^0} \hat{\tilde{\beta}}_j + \alpha^1 \tilde{y}_t \left(\hat{\tilde{\beta}}_j - \widetilde{\ln p_{jt}}\right) + \gamma \alpha^1 (\tilde{y}_t)^2 \hat{\tilde{\beta}}_j + \mu_{jt}\end{aligned}$$

    * Step 2: $\widetilde{\ln s_{jt}} - \text{Control}_{jt} = \gamma \tilde{y}_t \hat{\alpha^0} \hat{\tilde{\beta}}_j + \alpha^1 \tilde{y}_t \left(\hat{\tilde{\beta}}_j - \widetilde{\ln p_{jt}}\right) + \alpha^1 \gamma (\tilde{y}_t)^2 \hat{\tilde{\beta}}_j + \mu_{jt}$ where $\text{Control}_{jt} = \hat{\alpha^0}\left(\hat{\tilde{\beta}}_j - \widetilde{\ln p_{jt}}\right)$
      · ivest2=**ivregression**(rellnsjt2,relX2,relZ2,relA2)
      · relX2 = $\begin{bmatrix} \tilde{y}_t \hat{\alpha^0} \hat{\tilde{\beta}}_j & \tilde{y}_t \left(\hat{\tilde{\beta}}_j - \widetilde{\ln p_{jt}}\right) & (\tilde{y}_t)^2 \hat{\tilde{\beta}}_j \end{bmatrix}$
      · relZ2 = $\begin{bmatrix} \tilde{y}_t \hat{\alpha^0} \hat{\tilde{\beta}}_j & \tilde{y}_t \left(\hat{\tilde{\beta}}_j - \tilde{z_{jt}}\right) & (\tilde{y}_t)^2 \hat{\tilde{\beta}}_j \end{bmatrix}$
      · relA2= (data.relZ2' data.relZ2)^-1

2.2 Set starting values (step3_startandrescale.m)

1. Makes sure that elasticities (`alpha0, alpha1, gamma`) are within bounds.
2. Rescales starting values (**rescale.m**).
3. Times objective function (**step 4a_gmmobjective.m**) and checks that it is defined at starting values. Set Knitro objective function scaling.

2.3 Minimize non-linear objective function (4a_gmmobjective.m)

$$G(\delta;\theta)^{'}WG(\delta;\theta)$$

where:

$$G(\delta;\theta) = \begin{bmatrix} G_1(\delta;\theta) \\ G_2(\delta;\theta) \end{bmatrix} = \begin{bmatrix} Z'\xi(\delta) \\ \mathbf{E}[\delta_{j^*_{it}} Y_i|\delta,\theta] \\ \mathbf{E}[\tilde{p}_{j^*_{it}} Y_i|\delta,\theta] \end{bmatrix}$$

1. Identifies `delta` (mean utility) values using fixed point algorithm (**step3a_alldeltas**.m which calls **step3b_solvefordeltas**.m sequentially or in parallel).

   - For a set of markets (specified in ids.nTset, ids.Tsetmin, and ids.Tsetmax created in step0_dataprep.m), the code calculates the product-level market shares predicted by the model for a given θ and ξ: $s(\xi;\theta)$ and the difference between these predicted market shares and the empirical market shares. The fixed point is identified using a constrained optimization routine with constant (zero) objective function. The constraint (and Jacobian) are defined in **step3c_cons_s_par** (using shares from: **step3ci_rc_shares_multmkt**.m and **step3cii_Fdelta_par**.m). The Hessian is defined in **step3d_d2Fddelta2_par**.m.
   - For starting values, the code uses the fact that $\ln s_{jt} - \ln s_{0t} = \alpha_{im}\gamma_{im}\delta_{jt} - \alpha_{im}\tilde{x}_{jt}$ where $\tilde{x}_{jt} = x_{jt} - \bar{x}_t$, $\delta^0_{jt} = \left((\ln s_{jt} - \ln s_{0t}) + \hat{\alpha}^{0,lin}_m \tilde{x}_{jt}\right)/\hat{\alpha}^{0,lin}_m$

2. Extracts xi and calculate **macro moment**.

$$G_1(\delta;\theta) = Z'\xi(\delta) = Z'\left(\delta - (X'X)^{-1}X'\delta\right)$$

   (a) Pre-multiply `delta` by `data.pre'` (XX) to extract implied: `beta = data.pre'*delta`.

   (b) Pre-multiply quality shock by instruments to calculate macro moment: `g = data.relZ'*(delta - beta_hat(data.brandid))`

   (c) Calculate Jacobian matrix and gradient function ($\nabla_{\theta,\delta} g_1$) using **step3c_cons_s_par** (shares from **step3ci_rc_shares_multmkt**.m and **step3cii_Fdelta_par**.m for constraint).

3. Calculates **micro moments.**

$$\begin{aligned} G_2(\theta) \quad = \quad & \frac{1}{N}\sum_{t\in T}\sum_{h_j\in H_{jt}}\left(Y_{ht} - \frac{1}{N}\sum_{t\in T}\sum_{h_j\in H_{jt}} Y_{ht}\right)\left(\left(x_{hjt} - \left(\frac{\sum_j x_{jt}s_{hjt}(\delta;\theta)}{\sum_j s_{hjt}(\delta;\theta)}\right)\right)\right. \\ & \left.\ldots - \frac{1}{N}\sum_{t\in T}\sum_{h_j\in H_{jt}}\left(x_{hjt} - \left(\frac{\sum_j x_{jt}s_{hjt}(\delta;\theta)}{\sum_j s_{hjt}(\delta;\theta)}\right)\right)\right) \end{aligned}$$

   where for x_{hjt} equal to either p_{hjt}, the relative unit value paid by a household h for a product j that they purchased in market t, or δ_{hjt}, the perceived quality of the product j that they purchased in market t. H_t denotes the set of sample households observed shopping in each market $t \in T$ and H_{jt} denotes the set of sample households who purchase product j in market t. N is the number of household-product purchases observed. Y_{ht} denotes the income of sample household h in market t and $s_{hjt}(\delta;\theta)$ denotes the expected probability that sample household h purchases product j in market t.

   - **step4bi_cons_g2_par**.m calculates constraint and gradient functions $\nabla_{\theta,\delta} g_2, \nabla_{\theta,\delta} g_3$) to prepare inputs (using shares from **step3ci_rc_shares_multmkt.m**).