Estimation code memo

Yoichi Sugita

March 24, 2024

Folder “EstimationCode” contains estimation codes and a test dataset. You can use either R or Matlab.The R code uses the following packages: tidyverse, splines, splines2, nloptr, modopt.matlab, grpnet. The Matlab code uses the optimization toolbox.

1 Disclaimer

This code was basically written in April 2023 and is older than the version we currently use. I chose this version because this mathematical memo is prepared only for this version. Please understand three points it implies. First, the code does not reproduce results in our working papers and presentations. Second, the code does not achieve the best performance. Please do not judge any performance of our estimator by this code. Third, the code may contain errors. The authors of the code are not responsible for any error or trouble that using the code may cause.

2 DGP

“DGP_AR20230426_CD.Rmd” simulates 1000 samples and each sample has 300 firms with four periods.

2.1 Demand function

Each firm faces the following HSA (homothetic utility with a single aggregator) demand system of a constant response demand function:

$$\tilde{\varphi}_t(y_{it}, a_t(\mathbf{y}_t), \epsilon_{it}) = \Phi_t + \delta + \beta \ln(\exp(\alpha(y_{it} - a_t(\mathbf{y}_t))) + \gamma \varepsilon_{it}) \tag{1}$$

where $\mathbf{y}_t = (y_{1t}, y_{2t}, ..., y_{Nt})$ is a vector of outputs and $a_t(\mathbf{y}_t)$ is the quantity index, and Φ_t is the log of (given) total expenditure given by $\exp(\Phi_t) = \sum_{i=1}^{N} \exp(\tilde{\varphi}_t(y_{it}, a_t(\mathbf{y}_t), \varepsilon_{it}))$. The demand shock$\epsilon_{it}$ follows an MA1 process:

$$\epsilon_{it} = \rho_\epsilon \xi_{it-1} + \xi_{it}$$

where $\xi_{it}$and ξ_{it-1} are independent uniform random variables. Thus, the CDF of ϵ_{it} is

$$F_\epsilon(t) = \begin{cases} 1 - \frac{(1+\rho_\epsilon - t)^2}{2\rho} & \text{if } t \in [1, 1+\rho_\epsilon] \\ t - \frac{\rho_\epsilon}{2} & \text{if } t \in [\rho_\epsilon, 1] \\ \frac{t^2}{2\rho_\epsilon} & \text{if } t \in [0, \rho_\epsilon]. \end{cases}$$

We set the parameters as $(\alpha, \beta, \gamma, \delta, \rho_\epsilon, \Phi) = (0.12, 8.3, 1, -12, 0.6, 10)$ and the number of firms $N =$ 300.From (1), the quantity index $a_t(\mathbf{y}_t)$ is implicitly defined as

$$1 = \sum_{i=1}^{N} \frac{\exp(\tilde{\varphi}_t(y_{it}, a_t(\mathbf{y}_t), \varepsilon_{it}))}{\exp(\Phi_t)} = \sum_{i=1}^{N} \exp[\delta + \beta \ln(\exp(\alpha(y_{it} - a_t(\mathbf{y}_t))) + \gamma \varepsilon_{it})]. \tag{2}$$

2.2 Cobb-Douglas production function

The production function takes the following Cobb-Douglas form:

$$y_{it} = \theta_m m_{it} + \theta_k k_{it} + \theta_l l_{it} + \omega_{it} \tag{3}$$

where parameters are chosen as $(\theta_m, \theta_k, \theta_l) = (0.4, 0.3, 0.3)$. TFP ω_{it} follows an AR1 process

$$\omega_{it} = \rho_\omega \omega_{it-1} + \eta_{it}, \eta_{it} \sim N(0, \sigma_\eta^2),$$

where parameters are chosen as $(\rho_\omega, \sigma_\eta) = (0.8, 1)$ and $\omega_{i0} \sim N(0, \sigma_\eta^2/(1-\rho_\omega^2))$. Capital and labor are predetermined and follows the following exogenous laws of motion:

$$k_{it} = \rho_k k_{it-1} + \rho_{k\omega}\omega_{it-1} + e_{kit}, e_{kit} \sim N(0, \sigma_{ke}^2),$$
$$l_{it} = \rho_l l_{it-1} + \rho_{l\omega}\omega_{it-1} + e_{lit}, e_{lit} \sim N(0, \sigma_{le}^2)$$

where parameters are chosen as $(\rho_k, \rho_{k\omega}, \sigma_{ke}; \rho_l, \rho_{l\omega}, \sigma_{le}) = (0.4, 0.1, 1; 0.4, 0.1, 1)$ and $k_{i0}, l_{i0} \sim N(0, 1)$.

For given capital and employment at period t, firm i chooses the material at period t, taking a_t as given:

$$m_{it}(a_t) \in \arg\max_m \exp(\tilde{\varphi}_t(y_{it}, a_t, \varepsilon_{it})) - \exp\left(p_t^m + m\right) \text{ s.t (1) and (3).}$$

We set $(p_{m1}, p_{m2}, p_{m3}, p_{m4}) = (1, 0.5, 1, 0.5)$. The first order condition becomes:

$$\begin{aligned} & \frac{\alpha\beta\theta_m \exp(\alpha\theta_m m_{it} + x_{1it} - \alpha a_t)}{\exp(\alpha\theta_m m_{it} + x_{1it} - \alpha a_t) + \gamma\varepsilon_{it}} \\ = & \frac{\exp\left(p_t^m + m_{it}\right)}{\exp(\Phi_t + \delta + \beta \ln(\exp(\alpha\theta_m m_{it} + x_{1it} - \alpha a_t) + \gamma\varepsilon_{it}))} \text{ for } i = 1, \dots, N \end{aligned}$$

where $x_{1it} := \alpha(\theta_k k_{it} + \theta_l l_{it} + \omega_{it})$. The log of the FOC is

$$\begin{aligned} \ln(\alpha\beta\theta_m) + \alpha\theta_m m_{it} + x_{1it} - \alpha a_t - p_t^m - m_{it} + \\ \Phi_t + \delta + (\beta - 1)\ln(\exp(\alpha\theta_m + x_{1it} - \alpha a_t) + \gamma\varepsilon_{it}) = 0 \end{aligned} \tag{4}$$

We obtain $\{m_{it}\}_{i=1}^N$ and a_t in an equilibrium that satisfy a system of $N+1$ equations (2) and (4) in the following two steps. In the first step, we obtain $m_{it}(a_t)$ from (4) for given a_t. In the second step, we choose an equilibriuma_t that satisfies

$$1 = \sum_{i=1}^N \exp[\delta + \beta \ln(\exp(\alpha\theta_m m_{it}(a_t) + x_{1it} - \alpha a_t) + \gamma\varepsilon_{it})].$$

2.3 Useful properties

Markup Since the partial derivative of revenue is

$$\frac{\partial\tilde{\varphi}_t(y_{it}, a_t(\mathbf{y}_t), \epsilon_{it})}{\partial y_{it}} = \frac{\alpha\beta \exp(\alpha(y_{it} - a_t(\mathbf{y}_t)))}{\exp(\alpha(y_{it} - a_t(\mathbf{y}_t))) + \gamma\varepsilon_{it}},$$

the markup is expressed as

$$\left(\frac{\partial\tilde{\varphi}_t(y_{it}, a_t(\mathbf{y}_t), \epsilon_{it})}{\partial y_{it}}\right)^{-1} = \frac{1}{\alpha\beta} + \frac{\gamma\varepsilon_{it}}{\alpha\beta \exp(\alpha(y_{it} - a_t(\mathbf{y}_t)))}.$$

Since the markup has to be greater than one., $\alpha\beta \geq 1$ is required for the profit maximization to hold for all ϵ_{it}.

Control function The first order condition (4) implies the material demand function is a function of x_{it}:

$$m_{it} = \mathbb{M}_t\left(\theta_k k_{it} + \theta_l l_{it} + \omega_{it}, u_{it}\right),$$

where $u_{it} = F_\epsilon(\epsilon_{it})$. Thus, the control function for ω_{it} becomes

$$\omega_{it} = \lambda_t\left(m_{it}, u_{it}\right) - \theta_k k_{it} - \theta_l l_{it}.$$

This implies

$$y_{it} = \theta_m m_{it} + \lambda_t\left(m_{it}, u_{it}\right).$$

The first step revenue function is

$$\begin{aligned} \varphi_t\left(y_{it}, u_{it}\right) &= \tilde{\varphi}_t\left(\theta_m m_{it} + \lambda_t\left(m_{it}, u_{it}\right), F_\epsilon^{-1}\left(u_{it}\right)\right) \\ &= \phi_t\left(m_{it}, u_{it}\right). \end{aligned}$$

3 Estimation: First Step

3.1 Moment Condition

The first step identifies $\phi_t\left(m_{it}, u_{it}\right)$ and $u_{it}$by using the IV quantile regression:

$$E\left[1\left\{r_{it} \leq \phi_t(m_{it}, u)\right\} - u | m_{it-2}\right] = 0. \text{ for any } u \in [0,1] \tag{5}$$

where $1\{\cdot\}$ is an indicator function.

A traditional approach to estimation may be as follows. We consider L equal sized partitions of $[0,1]$ and let $T \equiv \{\tau_1, \tau_2, ..., \tau_{L-1}\}$ be the set of $L-1$ partition points, e.g., $T = \{0.01, 0.02, ..., 0.99\}$

for $L = 100$. For each $\tau_l \in T$, we formulate ϕ by B-splines:

$$\begin{aligned} \phi_t(m_{it}, \tau_l) &= \sum_{s_1=1}^{S_1} B^m_{s_1}(m_{it})\beta_{s_1}(\tau_l) \\ &= B^m(m_{it})^T \beta(\tau_l) \end{aligned} \tag{6}$$

where $B^m(m_{it}) = \left(B^m_1(m_{it}), ..., B^m_{S_1}(m_{it})\right)^T$ is a S_1 vector of B-spline basis functions of m_{it} and $\beta(\tau_l)$ is a vector of parameters to be estimated. We can use $B^m(m_{it-2}) = \left(B^m_1(m_{t-2}), ..., B^m_{S_1}(m_{t-2})\right)^T$ as a vector of IVs for $B^m(m_{it})$.For each $\tau_l \in T$, we have S_1moment conditions:

$$E\left[\left(1\left\{r_{it} \leq B^m(m_{it})^T \beta(\tau_l)\right\} - \tau_l\right) B^m(m_{it-2})\right] = 0. \tag{7}$$

3.2 Smoothed GMM Quantile Regression

The IV quantile regression typically face two challenges. First, the moment condition includes a non-smooth function (indicator function). Second, the estimated function $\phi_t(m_{it}, u)$ may fail to be increasing in u.

We use the smoothed GMM quantile regression by Firpo, Galvao, Pinto, Poirier and Sanroman (2022). Firpo et al. (2022) introduce three modifications to (7).First, they replace the indicator function by a kernel smoothing function. Second, they model $\beta(\tau_l)$ as a flexible parametric function of τ_l. Finally, they estimate $\beta(\tau_l)$ using $S_1 \times (L-1)$ moment conditions simultaneously.

We model $\beta(\tau_l) = (\beta_1(\tau_l), ..., \beta_{S_1}(\tau_l))$ by B-splines: for each $s = 1, ..., S_1$

$$\beta_s(\tau_l) = \sum_{s_2=1}^{S_2} B^\tau_{s_2}(\tau_l)\,\alpha_{s,s_2}$$

where $\{B_\tau(\tau)\}$ are a $S_2 \times 1$ vector of B-spline basis functions of τ_l.That means

$$\beta(\tau_l) = \begin{pmatrix} \beta_1(\tau_l) \\ \beta_2(\tau_l) \\ \vdots \\ \beta_{S_1}(\tau_l) \end{pmatrix} = \begin{pmatrix} \sum_{s_2=1}^{S_2} B^\tau_{s_2}(\tau_l)\,\alpha_{1,s_2} \\ \sum_{s_2=1}^{S_2} B^\tau_{s_2}(\tau_l)\,\alpha_{2,s_2} \\ \vdots \\ \sum_{s_2=1}^{S_2} B^\tau_{s_2}(\tau_l)\,\alpha_{S_1,s_2} \end{pmatrix}.$$

Then, the model (6) is written as

$$
\begin{aligned}
\phi_t(m_{it}, \tau_l) &= \sum_{s_2=1}^{S_1} \sum_{s_2=1}^{S_2} B_{s_1}^m(m_{it}) B_{s_2}^\tau(\tau_l)\, \alpha_{s,s_2} \\
&\equiv B^\phi(m_{it}, \tau_l)^T \alpha. \qquad (8)
\end{aligned}
$$

where $B^\phi(m_{it}, \tau)$ is a $(S_1S_2) \times 1$ vector of products of B-spline basis functions and α is a $(S_1S_2) \times 1$ vector of parameters to be estimated. $B^\phi(m_t, \tau_l)^T$ in (8) is a row-wise Kronecker product of $B^m(m_{it})$ and $B^\phi(\tau_l)$

Following Firpo et al. (2022), we consider the following kernel CDF function used by Horowitz:

$$
K(\tau) = \left[\frac{1}{2} + \frac{105}{64}\left(\tau - \frac{5}{3}\tau^3 + \frac{7}{5}\tau^5 - \frac{3}{7}\tau^7\right)\right] 1\{\tau \in [-1, 1]\} + 1\{\tau > 1\}.
$$

Define

$$
p_\tau(m_{it}, r_{it}; \alpha) \equiv K\left(\frac{B^\phi(m_{it}, \tau)^T \alpha - r_{it}}{b_n}\right) - \tau
$$

where the bandwidth is chosen as $b_n = 1.06 \cdot \hat{\sigma}_r \cdot n^{-1/5}$. The moment condition (7) is written as

$$
E\left[p(m_{it}, r_{it}; \alpha, \tau_l) B^m(m_{it-2})\right] = 0 \text{ for each} \tau_l \in T. \qquad (9)
$$

The moment condition (9) has S_1 equations for each τ_l and $S_1 \times (L-1)$ equations in total.

We construct the GMM estimator as follows. Let

$$
p^L(m_{it}, r_{it}; \alpha) \equiv \begin{pmatrix} p_{\tau_1}(m_{it}, r_{it}; \alpha) \\ p_{\tau_2}(m_{it}, r_{it}; \alpha) \\ \vdots \\ p_{\tau_{L-1}}(m_{it}, r_{it}; \alpha) \end{pmatrix}
$$

and

$$
w^L(m_{it}, r_{it}, \alpha) = p^L(m_{it}, r_{it}; \alpha) \otimes B^m(m_{t-2}).
$$

The moment condition (9) is written as

$$
m^L(\alpha) = E\left[w^L(m_{it}, r_{it}, \alpha)\right] = 0.
$$

Let α_0 be the true value of α, which satisfies

$$\alpha_0 = \arg\min_{\alpha} m^L(\alpha)^T \Omega(\alpha_0)^{-1} m^L(\alpha)$$

where the weighting matrix is

$$\begin{aligned}
\Omega_L(\alpha) &= E\left[w^L(m_{it}, r_{it}, \alpha)\, w^L(m_{it}, r_{it}, \alpha)^T\right] \\
&= E\left[p^L(m_{it}, r_{it}, \alpha)\, p^L(m_{it}, r_{it}, \alpha)^T \otimes B^m(m_{it-2}) B^m(m_{it-2})^T\right].
\end{aligned}$$

Firpo et al. (2022) showed

$$\Omega_L(\alpha_0)^{-1} = \Sigma_L^{-1} \otimes \Sigma_M^{-1}$$

where

$$\begin{aligned}
\Sigma_L^{-1} &= E\left[p^L(Z;\alpha)\, p^L(Z;\alpha)^T\right]^{-1} \\
&= \frac{L}{\epsilon_2 - \epsilon_1}
\begin{pmatrix}
\frac{\epsilon_2-\epsilon_1}{\epsilon_1 L + \epsilon_2 - \epsilon_1} + 1, & -1 & 0 & \cdots & 0 & 0 & 0 \\
-1 & 2 & -1 & \cdots & 0 & 0 & 0 \\
0 & -1 & 2 & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & 2 & -1 & 0 \\
0 & 0 & 0 & \cdots & -1 & 2 & -1 \\
0 & 0 & 0 & \cdots & 0 & -1 & \frac{\epsilon_2-\epsilon_1}{(1-\epsilon_2)L + \epsilon_2 - \epsilon_1} + 1
\end{pmatrix}
\end{aligned}$$

and

$$\Sigma_M = E\left[B^m(m_{t-2}) B^m(m_{t-2})^T\right].$$

Note that the weighting matrix does not depend on the parameter α. Thus, the GMM estimator is

$$\alpha_0 = \arg\min_{\alpha} m^L(\alpha)^T \left[\Sigma_L^{-1} \otimes \Sigma_M^{-1}\right] m^L(\alpha)$$

GMM-QR estimator Let $\{r_{it}, B_m(m_{it})^T, B_m(m_{it-2})^T\}_{i=1}^N$ be the observed sample data. The GMM-QR estimator is

$$\alpha^{GMM} = \arg\min_{\alpha} \bar{w}_L(\alpha)^T \left[\Sigma_L^{-1} \otimes \hat{\Sigma}_M^{-1}\right] \bar{w}_L(\alpha) \tag{10}$$

where

$$\begin{aligned}\bar{w}_L(\alpha) &= \frac{1}{n}\sum_{i=1}^{n} w^L(m_{it}, r_{it}, \alpha) \\ &= \frac{1}{n}\sum_{i=1}^{n} p^L(m_{it}, r_{it}, \alpha) \otimes B^m(m_{it-2})\end{aligned}$$

and

$$\hat{\Sigma}_M = \frac{1}{n}\sum_{i=1}^{n} B^m(m_{it-2}) B^m(m_{it-2})^T.$$

Shape restriction An advantage of the GMM quantile regression is that we can impose the monotonicity of $\phi_t(m_t, u)$. It is also easy to compute the derivatives of B-splines. Both R and Matlab have functions to calculate the derivatives of B-spline basis functions. The partial derivative of $\phi_t(m_{it}, \tau)$ with respect to m is

$$\begin{aligned}\frac{\partial \phi_t(m_{it}, \tau)}{\partial m_{it}} &= \sum_{s_2=1}^{S_1}\sum_{s_2=1}^{S_2} dB_{s_1}^m(m_t) B_{s_2}^{\tau}(\tau_l)\, \alpha_{s_1 s_2} \\ &\equiv \partial_m B^{\phi}(m_{it}, \tau_l)^T \alpha \end{aligned} \tag{11}$$

where $dB_{s_1}^m(m_t)$ is the derivative of $B_{s_1}^m(m_t)$. Similarly, the partial derivative of $\phi_t(m_{it}, \tau)$ with respect to u is

$$\begin{aligned}\frac{\partial \phi_t(m_{it}, \tau_l)}{\partial u_{it}} &= \sum_{s_2=1}^{S_1}\sum_{s_2=1}^{S_2} B_{s_1}^m(m_t)\, dB_{s_2}^{\tau}(\tau_l)\, \alpha_{s_1 s_2} \\ &\equiv \partial_u B^{\phi}(m_t, \tau_l)^T \alpha\end{aligned}$$

where $dB^{\tau}_{s_2}(\tau)$ is the derivative of $B^{\tau}_{s_2}(\tau)$. We impose the monotonicity of $\phi_t(m_t, u)$ as linear constraints on α:

$$\begin{aligned} &\partial_m B^{\phi}(m_{it}, \tau_l)^T \alpha > \kappa \\ &\partial_u B^{\phi}(m_{it}, \tau_l)^T \alpha > \kappa \text{ for all } i = 1, .., n; \tau_l \in T. \end{aligned} \tag{12}$$

for a vector of some small positive number κ.

Optimization We calculate (10) with constraints (12). We choose an initial value of α from a OLS regression of r_{it} on $B^{\phi}(m_t, \tau_l)^T$. The default optimization algorithm in the R code uses the SQP (sequential quadratic programming) of the NLopt through the nloptr package; the matlab code uses the sqp-legacy algorithm in the fmin-con function in the optimization toolbox.

Estimate the demand shock Let U be a $L \times 100$ partition of $[0, 1]$ and let U be the set of its partition points. Once we obtain $\hat{\alpha}$, we estimate u_{it} by a grid search:

$$\hat{u}_{it} = \arg\min_{\tau \in U} B^{\phi}(m_{it}, \tau)^T \hat{\alpha}.$$

Estimate the demand shock at $t-1$ For the second step, we also estimate $\hat{u}_{it-1}$ by repeating the above steps for lagged data.

4 Second Step

4.1 Transformation model

In the second step, we estimate the following transformation model

$$\lambda_t(m_{it}, u_{it}) = \theta_l l_{it} - \theta_l \rho l_{it-1} + \theta_k k_{it} - \theta_k \rho k_{it-1} + \rho \lambda_{t-1}(m_{it-1}, u_{it-1}) + \eta_t.$$

We model $\lambda_t(m_{it}, u_{it})$ and $\lambda_{t-1}(m_{it-1}, u_{it-1})$ by B-splines:

$$\begin{aligned}
\lambda_t(m_{it}, u_{it}) &= \sum_{s_3=1}^{S_3} \sum_{s_4=1}^{S_4} B^m_{s_3}(m_{it}) B^\tau_{s_4}(\tau_l) \beta_{s_3 s_4} \\
&= B^\lambda(m_{it}, u_{it})^T \beta \\
\lambda_{t-1}(m_{it-1}, u_{it-1}) &= B^\lambda(m_{it-1}, u_{it-1})^T \gamma
\end{aligned} \tag{13}$$

where $B^\lambda(m_{it}, u_{it})$ and $B^\lambda(m_{it-1}, u_{it-1})$ are (the tensor product of) the B-spline basis vectors. The partial derivative of λ_t is a linear function of β:

$$\begin{aligned}
\frac{\partial \lambda_t(m_{it}, u_{it})}{\partial m_{it}} &= \sum_{s_3=1}^{S_3} \sum_{s_4=1}^{S_4} dB^m_{s_3}(m_{it}) B^\tau_{s_4}(\tau_l) \beta_{s_3 s_4} \\
&= \partial_m B^\lambda(m_{it}, u_{it})^T \beta
\end{aligned}$$

where $dB^m_{s_3}(m_{it})$ is the derivative of $B^m_{s_3}(m_{it})$.

Following Linton et al. (2008), we construct a profile likelihood (PL) estimator as follows. The conditional distribution of m_{it} given $v_{it} := (k_{it}, l_{it}, u_{it}, x_{it-1}, u_{it-1})$ can be written as

$$G_{m_t|v_t}(m_{it}|v_{it}) = G_\eta\left(\lambda_t(m_{it}, u_{it}) - \theta_l l_{it} + \theta_l \rho l_{it-1} - \theta_k k_{it} + \theta_k \rho k_{it-1} - \rho \lambda_{t-1}(m_{it-1}, u_{it-1})\right)$$

The conditional density of m_t given v_t is derived as

$$\begin{aligned}
g_{m_t|v_t}(m_{it}|v_{it}) &= g_{\eta_t}(\eta_{it}) \frac{\partial \lambda_t(m_{it}, u_{it})}{\partial m_{it}} \\
&= g_{\eta_t}(\eta_{it}) \partial_m B^\lambda(m_{it}, u_{it})^T \beta
\end{aligned}$$

We obtain estimates of η_{it} and $g_\eta(\cdot)$ as follows. For given parameter β, we obtain $\lambda_{it}(\beta) := \lambda_t(m_{it}, u_{it}; \beta)$ and estimate

$$\begin{aligned}
\lambda_{it}(\beta) &= \theta_l l_{it} - \theta_l \rho l_{it-1} + \theta_k k_{it} - \theta_k \rho k_{it-1} + B^\lambda(m_{it-1}, u_{it-1})^T \gamma \rho + \eta_{it} \\
&\equiv Z_{it}^T \delta + \eta_{it}
\end{aligned} \tag{14}$$

by OLS to obtain the residual $\eta_{it}(\beta)$ where $Z_{it}^T \equiv \left[k_{it}, k_{it-1}, l_{it}, l_{it-1}, B^\lambda(m_{it-1}, u_{it-1})^T\right]$. Using $\eta_{it}(\beta)$, we estimate $g_\eta(\cdot)$ by a kernel estimator:

$$\hat{g}_{\eta_t}(\eta_t;\beta) = \frac{1}{nb}\sum_{j=1}^{n} K\left(\frac{\eta_{it}(\beta) - \eta_{jt}(\beta)}{b}\right) \tag{15}$$

where K is a smooth kernel and b is a bandwidth. We choose the Gaussian kernel and chooses a rules of thumb bandwidth, $b = 1.06 \cdot \hat{\sigma}_\eta \cdot n^{-1/5}$ and $\hat{\sigma}_\eta$ is the standard deviation of $\eta_{it}(\beta)$.

The log-likelihood function can be written as

$$\sum_{i=1}^{n}\left\{\ln g_{m_t|\nu_t}(m_{it}|\nu_{it})\right\} = \sum_{i=1}^{n}\left\{\ln \hat{g}_{\eta_t}(\eta_{it}(\beta);\beta) + \ln\left(\partial_m B^{\lambda}(m_{it}, u_{it})^T \beta\right)\right\}.$$

For identification, we must impose the location and scale normalization of $\lambda_t(m_t, u_t)$:

$$\begin{aligned} \lambda_t(m_{0.25}, 0.5) &= 0 \\ \lambda_t(m_{0.75}, 0.5) &= 1 \end{aligned} \tag{16}$$

where $m_{0.25}$ and $m_{0.75}$ are the 25 percentile and the 75 percentile of m_{it}, respectively.

The PL estimator $\hat{\beta}$ is defined as

$$\hat{\beta} \in \arg\max_{\beta} \sum_{i=1}^{n}\left\{\ln \hat{g}_{\eta_t}(\eta_{it}(\beta);\beta) + \ln\left(\partial_m B^{\lambda}(m_{it}, u_{it})^T \beta\right)\right\} \text{ subject to (16).}$$

Optimization We calculate (10) with constraints (12). We choose an initial value of α from a OLS regression of r_{it} on $B^{\phi}(m_t, \tau_l)^T$. The default optimization algorithm in the R code uses the SQP (sequential quadratic programming) of the NLopt through the nloptr package; the matlab code uses the sqp-legacy algorithm in the fmin-con function in the optimization toolbox.

Standard errors The current version of the code does not calculate the standard errors of β. We plan to add them in a future version. Linton et al. (2008) derived asymptotic standard errors for the PL estimator. You may use it if you can ignore the estimation error in $\hat{u}_{it}$ from step 1.

4.2 Calculate parameters

With estimate $\hat{\beta}$, we obtain $(\hat{\theta}_k, \hat{\theta}_l, \hat{\rho})$ by the regression (14). The code calculates an OLS regression of $\lambda_{it}(\hat{\beta})$ on Z_{it}^T and estimate θ_l and θ_k from the coefficients of l_{it} and k_{it}. Alternatively, we may estimate $(\theta_k, \theta_l, \rho)$ in (14) by non-linear least square.

We calculate $\hat{\theta}_m$ as follows. Let $s_{it} \equiv \frac{\exp(p_{mt}+m_{it})}{\exp(r_{it})}$ be the material expenditure share. From the paper,

$$
\begin{aligned}
\frac{\partial f}{\partial m_{it}} &= \frac{\partial \mathbb{M}_t^{-1}}{\partial m_t}\left(\frac{\partial \phi_t}{\partial m_t} - s_{it}\right)^{-1}\frac{\partial \phi_t}{\partial m_t} - \frac{\partial \mathbb{M}_t^{-1}}{\partial m_t} \\
&= \left(\frac{s_{it}}{\frac{\partial \phi_t}{\partial m_t} - s_{it}}\right)\frac{\partial \lambda_t}{\partial m_{it}}.
\end{aligned}
$$

Using (11), we estimate

$$
\hat{\theta}_m = \text{median}\left\{\left(\frac{s_{it}}{\partial_m B(m_{it}, \hat{u}_{it})^T \hat{\alpha} - s_{it}}\right)\partial_m B^{\lambda}(m_{it}, u_{it})^T \hat{\beta}\right\} \tag{17}
$$

where $\partial_m B(m_{it}, \hat{u}_{it})^T \hat{\alpha}$ estimates $\frac{\partial \phi_t}{\partial m_t}$ from the first step.

In the code, we impose the constant return to scale to identify the scale parameter b. The final estimates of the production function parameters are

$$
\begin{aligned}
\tilde{\theta}_m &= \frac{\hat{\theta}_m}{\hat{\theta}_m + \hat{\theta}_k + \hat{\theta}_l} \\
\tilde{\theta}_k &= \frac{\hat{\theta}_k}{\hat{\theta}_m + \hat{\theta}_k + \hat{\theta}_l} \\
\tilde{\theta}_l &= \frac{\hat{\theta}_l}{\hat{\theta}_m + \hat{\theta}_k + \hat{\theta}_l}
\end{aligned}
$$

The TFP and markup are estimated as

$$
\begin{aligned}
\tilde{\omega}_{it} &= \frac{B^{\lambda}(m_{it}, u_{it})^T \hat{\beta} - \hat{\theta}_k k_{it} - \hat{\theta}_l l_{it}}{\hat{\theta}_m + \hat{\theta}_k + \hat{\theta}_l} \\
\tilde{\mu}_{it} &= \frac{\tilde{\theta}_m}{s_{it}}.
\end{aligned}
$$

5 Extensions and planned future updates

Standard errors The current version of the code does not calculate the standard errors of β. We plan to add them in a future version. Firpo et al. (2022) and Linton et al. (2008) derived asymptotic standard errors for the 1st step and the 2nd step, respectively.

Observable Demand Shifter We can add an observable demand shifter such as firm's export status, z_{it}. In step 1, the B spline model of ϕ_t in (8) is

$$\begin{aligned}\phi_t\left(m_{it}, z_{it}, u_{it}\right) &= \sum_{s_1=1}^{S_1}\sum_{s_2=1}^{S_2}\sum_{s_3=1}^{S_3} B_{s_1}^{m}\left(m_{it}\right) B_{s_2}^{z}\left(z_{it}\right) B_{s_3}^{\tau}\left(\tau_l\right) \alpha_{s_1,s_2,s_3} \\ &= B^{\phi}\left(m_{it}, z_{it}, u_{it}\right)^T \alpha.\end{aligned}$$

In step 2, the B spline model of λ_t and λ_{t-1} in (13) is

$$\begin{aligned}\lambda_t(m_{it}, z_{it} u_{it}) &= \sum_{s_4=1}^{S_4}\sum_{s_5=1}^{S_5}\sum_{s_6=1}^{S_6} B_{s_4}^{m}\left(m_{it}\right) B_{s_5}^{z}\left(z_{it}\right) B_{s_6}^{\tau}\left(\tau_l\right) \beta_{s_4,s_5,s_6} \\ &= B^{\lambda}(m_{it}, z_{it} u_{it})^T \beta \\ \lambda_{t-1}(m_{it-1}, z_{it-1}, u_{it-1}) &= B^{\lambda}(m_{it-1}, z_{it-1}, u_{it-1})^T \gamma\end{aligned}$$

Data periods The current code considers a dataset with 4 periods to deal with a MA1 demand shock. If the demand shock is an iid shock instead of MA1, then we can use m_{it-1} as an IV instead of m_{it-1}, so the data needs 3 periods.

If a dataset includes more than 4 periods, we can exploit them to improve the estimation of $(\theta_m, \theta_l, \theta_k)$. Suppose we have T periods data ($t = 1, ..., T$) and $T \geq 5$. We have $T_0 = T - 3$ sub-datasets each of which has 4 periods. Thus, we have $\hat{\beta}_t$ and $\lambda_t(\hat{\beta}_t)$ for T_0 periods and we may want to pool them to improve the estimation of $(\theta_m, \theta_l, \theta_k)$. However, to pool estimates from different periods, we need to adjust the difference in location parameters a_t and scale parameters b_t across time. One way to adjust them may be as follows. First, we assume the standard error of η_t is stable over time $\sigma_{\eta t} = \sigma_\eta$. Since each estimate $\hat{\sigma}_{\eta t}$ and the scale parameter b_t satisfy $b_t \hat{\sigma}_{\eta t} = \sigma_\eta$, we identify $b_1/b_t = \hat{\sigma}_{\eta 1}/\hat{\sigma}_{\eta t}$ ($t = 2, .., T_0$). Second, we multiply b_1/b_t to each variable in (13) ,e.g., $\tilde{k}_{it} \equiv (b_1/b_t)k_{it}$, $\tilde{l}_{it} \equiv (b_1/b_t)l_{it}$

and so on. We estimate

$$\lambda_t(\hat{\beta}_t) = \kappa_t + \delta_1 \tilde{k}_{it} + \delta_2 \tilde{k}_{it-1} + \delta_3 \tilde{l}_{it} + \delta_4 \tilde{l}_{it-1} + \tilde{B}^{\lambda}(m_{it-1}, u_{it-1})^T \delta_5 + \eta_{it} \tag{18}$$

by pooling $t = 1, ..., T_0$ and including time fixed effects κ_t to adjust location parameters a_t, and obtain $\hat{\theta}_k = -\hat{\delta}_1$ and $\hat{\theta}_l = -\hat{\delta}_3$. Third, we estimate $\hat{\theta}_{mt}$ for each $t = 1, ..., T_0$ from (17). Then, we adjust the scale parameter and pool them to obtain

$$\hat{\theta}_m = median\left\{\frac{b_1}{b_t}\hat{\theta}_{mt}\right\}.$$

Finally, we identify the scale parameter b_1 by $b_1 = \hat{\theta}_m + \hat{\theta}_k + \hat{\theta}_l$ and obtain

$$\tilde{\theta}_m = \frac{\hat{\theta}_m}{b_1}, \tilde{\theta}_k = \frac{\hat{\theta}_k}{b_1}, \text{ and } \tilde{\theta}_l = \frac{\hat{\theta}_l}{b_1}.$$

We will add such an analysis in a future version.

More flexible production function In the above method, both ϕ_t and λ_t are functions of (m_{it}, u_{it}) thanks to the Cobb-Douglas production function. This property holds for a more flexible production function as long it takes the following form:

$$y = f_1(m_{it}) + f_2(k_{it}, l_{it}) + \omega_{it}.$$

The first order condition shows that the control function becomes

$$\omega_{it} = \lambda_t(m_{it}, u_{it}) - f_2(k_{it}, l_{it})$$

and the first step revenue function becomes

$$\begin{aligned}\varphi_t(y_{it}, u_{it}) &= \tilde{\varphi}_t\left(\theta_m m_{it} + \lambda_t(m_{it}, u_{it}), F_{\epsilon}^{-1}(u_{it})\right) \\ &= \phi_t(m_{it}, u_{it}).\end{aligned}$$

It is not difficult to extend the above method to this class of production functions. We will add such an analysis in a future version.

Parametric demand function The above method considers a non-parametric demand function. We may estimate a parametric demand function by using estimated $(\hat{p}_{it}, \hat{y}_{it}, \hat{u}_{it})$. We will add such an analysis in a future version.

6 Appendix

6.1 Matrix Implementation of Step1

The”bs” function in the R code and the “bsmatrixn” the Matlab code produce the matrix of B-spline basis function.For m_t, they produce the matrix of B-spline basis functions for m_{it} as

$$\mathbf{B}_t^m \equiv \begin{pmatrix} B_1^m(m_{1t}) & \cdots & B_{S_1}^m(m_{1t}) \\ B_1^m(m_{2t}) & \cdots & B_{S_1}^m(m_{2t}) \\ \vdots & \ddots & \vdots \\ B_1^m(m_{nt}) & \cdots & B_{S_1}^m(m_{nt}) \end{pmatrix} = \begin{pmatrix} B^m(m_{1t})^T \\ B^m(m_{2t})^T \\ \vdots \\ B^m(m_{nt})^T \end{pmatrix}$$

We can specify the degree of the B-spline, the number of knots, and the position of knots. Similarly, we generate the matrices of B-spline basis functions for m_{it-2} and τ, respectively:

$$\mathbf{B}_{t-2}^m \equiv \begin{pmatrix} B_1^m(m_{1t-2}) & \cdots & B_{S_1}^m(m_{1t-2}) \\ B_1^m(m_{2t-2}) & \cdots & B_{S_1}^m(m_{2t-2}) \\ \vdots & \ddots & \vdots \\ B_1^m(m_{nt-2}) & \cdots & B_{S_1}^m(m_{nt-2}) \end{pmatrix} = \begin{pmatrix} B^m(m_{1t-2})^T \\ B^m(m_{2t-2})^T \\ \vdots \\ B^m(m_{nt-2})^T \end{pmatrix}$$

$$\mathbf{B}^\tau \equiv \begin{pmatrix} B_1^\tau(\tau_1) & \cdots & B_{S_2}^\tau(\tau_1) \\ B_1^\tau(\tau_2) & \cdots & B_{S_2}^\tau(\tau_2) \\ \vdots & \ddots & \vdots \\ B_1^\tau(\tau_{L-1}) & \cdots & B_{S_2}^\tau(\tau_{L-1}) \end{pmatrix}.$$

Obtain the data matrix of the B-spline basis of m_t and given τ_l, $B^\phi(m_{it}, \tau_l)$.

B-spline Define the data matrix of the B-spline basis of m_t for given τ_l as

$$
\underset{n\times K}{\mathbf{B}^{\phi}_{\tau_l}} = \begin{pmatrix} B^{\phi}(m_{1t},\tau_l)^T \\ B^{\phi}(m_{2t},\tau_l)^T \\ \vdots \\ B^{\phi}(m_{nt},\tau_l)^T \end{pmatrix}
$$

$$
= \left(\begin{matrix} B^m_1(m_{1t})B^\tau_1(\tau_l) & \cdots & B^m_1(m_{1t})B^\tau_{S_2}(\tau_l), & B^m_2(m_{1t})B^\tau_1(\tau_l) & \cdots & B^m_2(m_{1t})B^\tau_{S_2}(\tau_l), & \cdots \\ B^m_1(m_{2t})B^\tau_1(\tau_l) & \cdots & B^m_1(m_{2t})B^\tau_{S_2}(\tau_l), & B^m_2(m_{2t})B^\tau_1(\tau_l) & \cdots & B^m_2(m_{2t})B^\tau_{S_2}(\tau_l), & \cdots \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \cdots \\ B^m_1(m_{nt})B^\tau_1(\tau_l) & \cdots & B^m_1(m_{nt})B^\tau_{S_2}(\tau_l), & B^m_2(m_{nt})B^\tau_1(\tau_l) & \cdots & B^m_2(m_{nt})B^\tau_{S_2}(\tau_l), & \cdots \end{matrix} \right.
$$

$$
\left. \begin{matrix} \cdots & B^m_{S_1}(m_{1t})B^\tau_1(\tau_l) & \cdots & B^m_{S_1}(m_{1t})B^\tau_{S_2}(\tau_l) \\ \cdots & B^m_{S_1}(m_{2t})B^\tau_1(\tau_l) & \cdots & B^m_{S_1}(m_{2t})B^\tau_{S_2}(\tau_l) \\ \cdots & \vdots & \ddots & \vdots \\ \cdots & B^m_{S_1}(m_{nt})B^\tau_1(\tau_l) & \cdots & B^m_{S_1}(m_{nt})B^\tau_{S_2}(\tau_l) \end{matrix} \right).
$$

We express the data matrix for (8) as

$$
\underset{n(L-1)\times K}{\mathbf{B}^{\phi}} \equiv \begin{pmatrix} \mathbf{B}^{\phi}_{\tau_1} \\ \mathbf{B}^{\phi}_{\tau_2} \\ \vdots \\ \mathbf{B}^{\phi}_{\tau_{L-1}} \end{pmatrix} = \begin{pmatrix} B^{\phi}(m_{1t},\tau_1)^T \\ \vdots \\ B^{\phi}(m_{nt},\tau_1)^T \\ B^{\phi}(m_{1t},\tau_2)^T \\ \vdots \\ B^{\phi}(m_{nt},\tau_{L-1})^T \end{pmatrix}.
$$

We can calculate $\mathbf{B}^{\phi}$ by a row-wise Kronecker product (expressed by $*_{RK}$) of two matrices

$$\iota_n \otimes \mathbf{B}_t^m = \begin{pmatrix} \mathbf{B}_t^m \\ \mathbf{B}_t^m \\ \vdots \\ \mathbf{B}_t^m \end{pmatrix} \text{ and } \mathbf{B}^{\tau} \otimes \iota_n = \begin{pmatrix} B^{\tau}(\tau_1)^T \\ \vdots \\ B^{\tau}(\tau_1)^T \\ B^{\tau}(\tau_2)^T \\ \vdots \\ B^{\tau}(\tau_{L-1})^T \end{pmatrix}.$$

The Matlab code uses the Khatri-Rao product $*_{KR}$ (column-wise Kronecker product). Thus, the data matrix is

$$\begin{aligned} \underset{n(L-1)\times K}{\mathbf{B}^{\phi}} &= \left(\iota_n \otimes \mathbf{B}_t^m\right) *_{RK} \left(\mathbf{B}^{\tau} \otimes \iota_n\right) \\ &= \left[\left(\iota_n \otimes \mathbf{B}_t^m\right)^T *_{KR} \left(\mathbf{B}^{\tau} \otimes \iota_n\right)^T\right]^T. \end{aligned}$$

Variance Matrix $\hat{\Sigma}_M$ Since $\left(\mathbf{B}_{t-2}^m\right)^T = \begin{pmatrix} B^m(m_{1t-2}) & B^m(m_{2t-2}) & \cdots & B^m(m_{nt-2}) \end{pmatrix}$, we have

$$\begin{aligned} \left(\mathbf{B}_{t-2}^m\right)^T B_{t-2}^m &= \begin{pmatrix} B^m(m_{1t-2}) & B^m(m_{2t-2}) & \cdots & B^m(m_{nt-2}) \end{pmatrix} \begin{pmatrix} B^m(m_{1t-2})^T \\ B^m(m_{2t-2})^T \\ \vdots \\ B^m(m_{nt-2})^T \end{pmatrix} \\ &= \sum_{i=1}^{n} B^m(m_{it-2}) B^m(m_{it-2})^T. \end{aligned}$$

Therefore, the variance matrix is obtained as:

$$\hat{\Sigma}_M = \frac{1}{n}\left(\mathbf{B}_{t-2}^m\right)^T B_{t-2}^m.$$

Moment condition Define an operator $\mathbf{K}(x)$ for an vector $x = (x_1, ..., x_n)^T$ such that

$$\mathbf{K}(x) = \begin{pmatrix} K(x_1/b_n) \\ K(x_2/b_n) \\ \vdots \\ K(x_n/b_n) \end{pmatrix}.$$

Let

$$\begin{aligned}
\mathbf{p}_{\tau_l}(\alpha) &\equiv \begin{pmatrix} p_{\tau_l}(m_{1t}, r_{1t}, \alpha) \\ p_{\tau_l}(m_{2t}, r_{2t}, \alpha) \\ \vdots \\ p_{\tau_l}(m_{nt}, r_{nt}, \alpha) \end{pmatrix} \\
&= \begin{pmatrix} K\left(\frac{B^{\phi}(m_{1t},\tau_l)^T\alpha - r_{1t}}{b_n}\right) - \tau_l \\ K\left(\frac{B^{\phi}(m_{2t},\tau_l)^T\alpha - r_{2t}}{b_n}\right) - \tau_l \\ \vdots \\ K\left(\frac{B^{\phi}(m_{nt},\tau_l)^T\alpha - r_{nt}}{b_n}\right) - \tau_l \end{pmatrix} \\
&= \mathbf{K}\left(\mathbf{B}^{\phi}_{\tau_l}\alpha - r_t\right) - \tau_l \iota_n
\end{aligned}$$

where

$$r_t = \begin{pmatrix} r_{1t} \\ r_{2t} \\ \vdots \\ r_{nt} \end{pmatrix} \text{ and } \underset{n\times 1}{\iota_n} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}.$$

Note that

$$\left(\mathbf{B}^m_{t-2}\right)^T \mathbf{p}_{\tau_l}(\alpha) = \begin{pmatrix} B^m(m_{1t-2}) & B^m(m_{2t-2}) & \cdots & B^m(m_{nt-2}) \end{pmatrix} \begin{pmatrix} p_{\tau_l}(m_{1t}, r_{1t}, \alpha) \\ p_{\tau_l}(m_{2t}, r_{2t}, \alpha) \\ \vdots \\ p_{\tau_l}(m_{nt}, r_{nt}, \alpha) \end{pmatrix}$$

$$= \sum_{i=1}^{n} p_{\tau_l}(m_{it}, r_{it}, \alpha) B^m(m_{it-2})$$

since $p_\tau(m_{it}, r_{it}, \alpha)$ is a scalar.

Then, the moment condition is obtained by

$$\begin{aligned}
\underset{K(L-1)\times 1}{\bar{w}^L(\alpha)} &= \frac{1}{n}\sum_{i=1}^{n} p^L(m_{it}, r_{it}, \alpha) \otimes B^m(m_{it-2}) \\
&= \frac{1}{n}\begin{pmatrix} \sum_{i=1}^n p_{\tau_1}(m_{it}, r_{it}, \alpha) B^m(m_{it-2}) \\ \sum_{i=1}^n p_{\tau_2}(m_{it}, r_{it}, \alpha) B^m(m_{it-2}) \\ \vdots \\ \sum_{i=1}^n p_{\tau_{L-1}}(m_{it}, r_{it}, \alpha) B^m(m_{it-2}) \end{pmatrix} \\
&= \frac{1}{n}\begin{pmatrix} \left(\mathbf{B}^m_{t-2}\right)^T \mathbf{p}_{\tau_1}(\alpha) \\ \left(\mathbf{B}^m_{t-2}\right)^T \mathbf{p}_{\tau_2}(\alpha) \\ \vdots \\ \left(\mathbf{B}^m_{t-2}\right)^T \mathbf{p}_{\tau_{L-1}}(\alpha) \end{pmatrix} \\
&= \frac{1}{n}\begin{pmatrix} \underset{K\times n}{\left(\mathbf{B}^m_{t-2}\right)^T} & 0 & 0 & 0 \\ 0 & \left(\mathbf{B}^m_{t-2}\right)^T & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \left(\mathbf{B}^m_{t-2}\right)^T \end{pmatrix} \begin{pmatrix} \mathbf{p}_{\tau_1}(\alpha) \\ \mathbf{p}_{\tau_2}(\alpha) \\ \vdots \\ \mathbf{p}_{\tau_{L-1}}(\alpha) \end{pmatrix} \\
&= \frac{1}{n}\underbrace{\left(I_{L-1} \otimes \left(\mathbf{B}^m_{t-1}\right)\right)^T}_{K(L-1)\times n(L-1)} \underset{n(L-1)\times 1}{\mathbf{p}^L}
\end{aligned}$$

where $\mathbf{p}^L$ can be calculated as

$$\underset{n(L-1)\times 1}{\mathbf{p}^L} \equiv \begin{pmatrix} \mathbf{p}_{\tau_1}(\alpha) \\ \mathbf{p}_{\tau_2}(\alpha) \\ \vdots \\ \mathbf{p}_{\tau_{L-1}}(\alpha) \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{K}\left(\mathbf{B}^{\phi}_{\tau_1}\alpha - r\right) - \tau_1 \iota_n \\ \mathbf{K}\left(\mathbf{B}^{\phi}_{\tau_2}\alpha - r\right) - \tau_2 \iota_n \\ \vdots \\ \mathbf{K}\left(\mathbf{B}^{\phi}_{\tau_{L-1}}\alpha - r\right) - \tau_{L-1} \iota_n \end{pmatrix}$$

$$= \mathbf{K}\left(\mathbf{B}^{\phi}\alpha - \iota_{L-1} \otimes r\right) - \tau \otimes \iota_n$$

where $\tau \equiv (\tau_1, ..., \tau_{L-1})^T$.

Shape restriction The"dbs" function in the R code and the “d_basisn” function in the Matlab code produce the matrix of the derivative of B-spline basis function.

$$\mathbf{d_m B}^m_t \equiv \begin{pmatrix} dB^m_1(m_{1t}) & \cdots & dB^m_{S_1}(m_{1t}) \\ dB^m_1(m_{2t}) & \cdots & dB^m_{S_1}(m_{2t}) \\ \vdots & \ddots & \vdots \\ dB^m_1(m_{nt}) & \cdots & dB^m_{S_1}(m_{nt}) \end{pmatrix} = \begin{pmatrix} dB^m(m_{1t})^T \\ dB^m(m_{2t})^T \\ \vdots \\ dB^m(m_{nt})^T \end{pmatrix}.$$

$$\mathbf{d_u B}^{\tau} \equiv \begin{pmatrix} dB^{\tau}_1(\tau_1) & \cdots & dB^{\tau}_{S_2}(\tau_1) \\ dB^{\tau}_1(\tau_2) & \cdots & dB^{\tau}_{S_2}(\tau_2) \\ \vdots & \ddots & \vdots \\ dB^{\tau}_1(\tau_{L-1}) & \cdots & dB^{\tau}_{S_2}(\tau_{L-1}) \end{pmatrix}$$

Those functions choose the position of knots based on the quantile. We can calculate (8) in the matrix form by:

$$\underset{n(L-1)\times K}{\mathbf{d_m B^{\phi}}} \equiv \begin{pmatrix} \partial_m B^{\phi}(m_{1t},\tau_1)^T \\ \vdots \\ \partial_m B^{\phi}(m_{nt},\tau_1)^T \\ \partial_m B^{\phi}(m_{1t},\tau_2)^T \\ \vdots \\ \partial_m B^{\phi}(m_{nt},\tau_{L-1})^T \end{pmatrix} = \left[\left(\iota_n \otimes \mathbf{d_m B}_t^m\right)^T *_{KR} \left(\mathbf{B}^{\tau} \otimes \iota_n\right)^T \right]^T$$

$$\underset{n(L-1)\times K}{\mathbf{d_u B^{\phi}}} \equiv \begin{pmatrix} \partial_u B^{\phi}(m_{1t},\tau_1)^T \\ \vdots \\ \partial_u B^{\phi}(m_{nt},\tau_1)^T \\ \partial_u B^{\phi}(m_{1t},\tau_2)^T \\ \vdots \\ \partial_u B^{\phi}(m_{nt},\tau_{L-1})^T \end{pmatrix} = \left[\left(\iota_n \otimes \mathbf{B}_t^m\right)^T *_{KR} \left(\mathbf{d_u B}^{\tau} \otimes \iota_n\right)^T \right]^T.$$

Then, the monotonicity restriction of ϕ in (12) is expressed as

$$\begin{pmatrix} \mathbf{d_m B^{\phi}} \\ \mathbf{d_u B^{\phi}} \end{pmatrix} \alpha \geq 0.$$

6.2 Matrix Implementation of Step 2

In the code, we calculate $\hat{g}_{\eta_t}(\eta_{it}(\beta);\beta)$ as follows. We first define

$$\mathbf{X}_h \equiv \begin{pmatrix} l_{1t} & k_{1t} & l_{1t-1} & k_{1t-1} & X^{\lambda}_{1t-1} \\ l_{2t} & k_{2t} & l_{2t-1} & k_{2t-1} & X^{\lambda}_{2t-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ l_{nt} & k_{nt} & l_{nt-1} & k_{nt-1} & X^{\lambda}_{nt-1} \end{pmatrix} \text{ and } \mathbf{B}^{\lambda} \equiv \begin{pmatrix} B^{\lambda}(m_{1t},\hat{u}_{1t})^T \\ B^{\lambda}(m_{2t},\hat{u}_{2t})^T \\ \vdots \\ B^{\lambda}(m_{nt},\hat{u}_{nt})^T \end{pmatrix}$$

Then, for given β, the OLS estimates of $\eta_{it}(\beta)$ are calculated as

$$\begin{aligned}\begin{pmatrix}\eta_{1t}(\beta)\\ \eta_{2t}(\beta)\\ \vdots\\ \eta_{nt}(\beta)\end{pmatrix} &= \left(I - \mathbf{X}_h\left(\mathbf{X}_h^T\mathbf{X}_h\right)^{-1}\mathbf{X}_h^T\right)\begin{pmatrix}\lambda_{1t}(\beta)\\ \lambda_{2t}(\beta)\\ \vdots\\ \lambda_{nt}(\beta)\end{pmatrix}\\ &= \left(I - \mathbf{X}_h\left(\mathbf{X}_h^T\mathbf{X}_h\right)^{-1}\mathbf{X}_h^T\right)\mathbf{B}^\lambda\beta\\ &= \Psi\beta.\end{aligned}$$

The term $\eta_{it}(\beta) - \eta_{jt}(\beta)$ inside the kernel density (15) can be expressed as a linear function of β:

$$\begin{aligned}\begin{pmatrix}\eta_{1t}(\beta)-\eta_{jt}(\beta)\\ \eta_{2t}(\beta)-\eta_{jt}(\beta)\\ \vdots\\ \eta_{nt}(\beta)-\eta_{jt}(\beta)\end{pmatrix} &= \Psi\beta - \iota_n\Psi_j\beta\\ &= \left(\Psi - \iota_n\Psi_j\right)\beta\end{aligned}$$

Then, we put $\left(\Psi - \iota_n\Psi_j\right)\beta$ into the Gaussian kernel with the bandwidth and calculate it mean to obtain $\hat{g}_{\eta_t}(\eta_{it}(\beta);\beta)$.

References

Firpo, Sergio, Antonio F Galvao, Cristine Pinto, Alexandre Poirier, and Graciela Sanroman, "GMM quantile regression," *Journal of Econometrics*, 2022, *230* (2), 432–452.

Linton, Oliver, Stefan Sperlich, and Ingrid Van Keilegom, "Estimation of a semiparametric transformation model," *The Annals of Statistics*, 2008, *36* (2), 686–718.