Lista 1 - Organização Industrial I

Gustavo Henrique

2025-05-22

Parte I: Demanda logit

Considerando s_{jt} como a participação de mercado da firma j no mercado t e s_{0t} como a participação da opção externa, o componente fixo de utilidade é modelado por:

$$\delta_{jt} = log(s_{jt}) - log(s_{0t}) = \beta_{0j} + \beta_1 x_{jt}^{conv} + \beta_2 x_{jt}^{spec} - \alpha p_{jt} + \xi_{jt}$$

onde x_{jt}^{conv} e x_{jt}^{spec} são características do produto (canais convencionais e especiais), p_{jt} é o preço e ξ_{jt} representa choques não observados.

Estratégia de Identificação

Para lidar com a endogeneidade dos preços, foram utilizados:

- Variáveis exógenas do modelo;
- **Instrumento de Hausman**: Média dos preços praticados pela firma em outras cidades da mesma região.

As hipóteses para validade do instrumento são:

- *Relevância*: Custos regionais são correlacionados entre cidades da mesma região;
- *Exclusão*: Choques específicos de custo e demanda não são correlacionados entre cidades, após controlar pelos preços.

Programação

Abaixo segue código para geração dos resultados:

```
## Limpando a base
data = data %>%
  # Gerando dummies para cada firma
  mutate(j1 = case_when(
    j == 1 ~ 1,
    T ~ 0
  ),
  j2 = case_when(
```

```r
    j == 2 ~ 1,
    T ~ 0
  )) %>%
  group_by(t) %>%
  # Gerando variável da % de consumidores que escolhem a "outside option"
  mutate(ms0 = 1 - sum(ms)) %>%
  group_by(r, j) %>%
  # Gerando o instrumento (média dos preços da firma em outras cidades da mesma região)
  mutate(price_iv = (sum(price) - price) / (n() - 1)) %>%
  # Gerando o componente fixo da utilidade
  mutate(delta = log(ms)-log(ms0))

## Estimação
# OLS
ols_reg = ivreg(delta ~ j1 + j2 + channels + channels_spec + price - 1, data=data)

# IV
iv_reg = ivreg(delta ~ j1 + j2 + channels + channels_spec + price - 1 | j1 + j2 + channels + channels_sp

## Visualização dos resultados
suppressWarnings(stargazer(ols_reg, iv_reg, type = 'text'))
```

```
##
## ============================================================
##                                   Dependent variable:
##                               ----------------------------
##                                          delta
##                                    (1)            (2)
## ------------------------------------------------------------
## j1                                -0.068         0.210*
##                                   (0.124)        (0.126)
##
## j2                              -0.447***        -0.189
##                                   (0.119)        (0.121)
##
## channels                         0.016***       0.016***
##                                   (0.001)        (0.001)
##
## channels_spec                    0.125***       0.130***
##                                   (0.018)        (0.018)
##
## price                           -0.046***      -0.053***
##                                   (0.002)        (0.002)
##
## ------------------------------------------------------------
## Observations                        586            586
## R2                                 0.854          0.852
## Adjusted R2                        0.853          0.851
## Residual Std. Error (df = 581)     0.609          0.614
## ============================================================
## Note:                           *p<0.1; **p<0.05; ***p<0.01
```

Interpretação dos resultados

Como podemos ver na tabela acima, temos $\hat{\alpha}_{OLS} = 0.046 < 0.053 = \hat{\alpha}_{IV}$, ou seja, a estimação via OLS subestima o efeito do preço. Podemos interpretar este resultado como evidência de que parte da variação em δ é explicada por fatores regionais não observáveis, de modo que o instrumento de Hausman corrige esse viés.

Parte II: Demanda Logit com coeficientes aleatórios

Para capturar heterogeneidade dos consumidores, ampliamos o modelo com coeficientes aleatórios. Além do instrumento de Hausman, incluímos como instrumentos o número de canais convencionais e especiais oferecidos pela firma rival na mesma cidade.

Cálculo do Market Share

Dado um vetor de componentes fixos δ, o market share esperado da firma j na cidade t é:

$$s_{jt}(\sigma, \delta) = \int \frac{exp(\delta_{jt} + \sigma\eta_{it}p_{jt})}{1 + exp(\delta_{1t} + \sigma\eta_{it}p_{1t}) + exp(\delta_{2t} + \sigma\eta_{it}p_{2t})} dF_{\eta_{it}}$$

Na prática, aproximamos a integral por simulação com $H_s = 100$ sorteios de η para cada cidade:

$$\tilde{s}_{jt}(\sigma, \delta) = \frac{1}{H_s} \sum_{i=1}^{H_s} \frac{exp(\delta_{jt} + \sigma\eta_{it}p_{jt})}{1 + exp(\delta_{1t} + \sigma\eta_{it}p_{1t}) + exp(\delta_{2t} + \sigma\eta_{it}p_{2t})}$$

Estimação via GMM

A estimação dos parâmetros é feita via GMM, minimizando:

$$J(\beta, \sigma) = (\delta(\sigma) - X\beta)' Z(Z'Z)^{-1} Z'(\delta(\sigma) - X\beta)$$

onde, para cada σ, calculamos:

$$\beta(\delta(\sigma)) = (X'Z(Z'Z)^{-1}Z'X)^{-1} X'Z(Z'Z)^{-1}Z'\delta(\sigma)$$

Programação

Abaixo segue código para geração dos resultados:

```r
# Construindo os novos instrumentos
data = data %>%
  group_by(t)%>%
  # Instrumento de canais convencionais
  mutate(channels_iv = case_when(
    j == 1 ~ channels[j == 2],
    T ~ channels[j == 1]
  ),
  # Instrumento de canais especiais
```

```
  channels_spec_iv = case_when(
    j == 1 ~ channels_spec[j == 2],
    T ~ channels_spec[j == 1]
    ))

# Gerando a matriz X
X = as.matrix(data[,c("j1", "j2", "channels", "channels_spec", "price")])

# Gerando a matriz de instrumentos
Z = as.matrix(data[,c("j1", "j2", "channels", "channels_spec", "price_iv", "channels_iv", "channels_spe


# Extraindo uma amostra de 100 indivíduos por cidade
set.seed(11)
n = length(unique(data$t))
ind = 100
eta = rnorm(n*ind)
eta = array(rep(eta,each=2),c(n*2,ind))

# Criando uma função para estimação da integral
integral_f = function(i,sigma,delta){
  # Encontrando linha da firma 1
  j = i - 1*(i%%2 == 0)
  # Gerando a lista
  lista = exp(delta[i] + sigma*data$price[i]*eta[i,])/(1 + exp(delta[j] + sigma*data$price[j]*eta[j,]) +
  # Retornando a média
  return(mean(lista))
}

# Criando uma função que atualiza o delta, dado o sigma
delta_f = function(sigma){
  delta_old = rep(1,n*2)
  delta = rep(0,n*2)
  iter = 0
  maxiter = 10000
  while (max(abs(delta-delta_old)) > 1e-10) {
    iter = iter+1
    delta_old = delta
    # estimando a integral em cada linha
    integ = sapply(1:(n*2),function(i){integral_f(i,sigma,delta)})
    # atualizando o delta
    delta = delta + log(data$ms) - log(integ)
  }
  return(delta)
}

# Criando uma função linear que retorna (beta_{0,1}, beta_{0,2}, beta_1, beta_2, -alpha)
coef_f = function(delta){
  return(solve(t(X) %*% Z %*% solve(t(Z) %*% Z) %*% t(Z) %*% X) %*% t(X) %*% Z %*% solve(t(Z) %*% Z) %*%

# Criando uma função gmm para dado sigma
gmm_f = function(sigma){
  delta = delta_f(sigma)
```

```r
  return(t(delta - X %*% coef_f(delta)) %*% Z %*% solve(t(Z) %*% Z) %*% t(Z) %*% (delta - X %*% coef_f(d
}

# Aplicando a função gmm no grid de sigma
gmm_grid = sapply(seq(0, 0.2, length.out = 40), gmm_f)

# Calculando o sigma
sigma_hat = seq(0, 0.2, length.out = 40)[which.min(gmm_grid)]

# Calculando os coeficientes
coef_hat = coef_f(delta_f(sigma_hat))

# Imprimindo o valor do sigma estimado
sigma_hat
```

```
## [1] 0.06666667
```

```r
# Imprimindo o valor dos coeficientes
coef_hat
```

```
##                     [,1]
## j1             1.96979864
## j2             1.54005239
## channels       0.02338246
## channels_spec  0.18974676
## price         -0.11977059
```

Resultados

Dos resultados acima, temos que o valor ótimo encontrado para σ foi $\hat{\sigma} = 0.066$, com estimativas finais:

- $\hat{\beta}_{01} = 2.050$
- $\hat{\beta}_{02} = 1.622$
- $\hat{\beta}_1 = 0.023$
- $\hat{\beta}_2 = 0.199$
- $\hat{\alpha} = 0.121$