Modeling and Figures: Level-by-Level

Tate Huffman

2/8/2022

Overview

This is part of a series of RMarkdown files that will break down the code contained in the document at the end of the fall. This file in particular will write the functions necessary to run the O'Flaherty and Siow model, run the model, and update the figures used in the thesis.

This differs from the file dated February 1, 2022 because this will instead be fitting a series of models that are each based on a one-step promotion sequence; e.g., fitting the model to Low-A to High-A promotions, or Double-A to Triple-A. This is done to both address possible noise and inaccuracy from the previous iteration of the model, and also to potentially explain some of its more noticeable disparities. For example, if predicted exit probabilities are significantly less accurate at lower-level sequences than at higher levels, it may serve as support for the idea that baseball's economic system is more taxing for lower-level players, in turn helping springboard into a discussion of baseball's labor economics in general, and MLB's antitrust exemption more specifically.

Model Details

We want to minimize the objective function

$$Q = \sum_{i=1}^{20} [P_i - \pi_i(\Omega)]^2$$

with respect to the parameters of Ω. The values of i correspond to the exit or promotion of the individual in a given time period: for $1 \leq i \leq 10$, this denotes an exit in year i, and for $11 \leq i \leq 20$, this denotes a promotion of this person in year $i - 10$ (e.g., $i = 12$ means a promotion for the individual in year 2). P_i is the observed probability of a worker exiting or being promoting in period i.

$\pi_i(\Omega)$ is the predicted probability of an exit or promotion in period i, where $\Omega = (\alpha, \tau, \theta_0, \mu, \sigma)$. These variables, respectively, correspond to: the probability of a Type A (able) worker producing a good signal; the probability of a Type B (unable) worker producing a good signal; the ex ante probability of a new worker being Type A; the mean of the worker promotion threshold θ_u; and its standard deviation.

When simplified, we see the probability of an exit in period i as

$$\pi_i = [\theta_0 \alpha^{i-1}(1-\alpha) + (1-\theta_0)\tau^{i-1}(1-\tau)] \int_{\theta(i-1,i-1)}^{1} g(\theta_u; \mu, \sigma) d\theta_u$$

and the probability of a promotion in period i as

$$\pi_{i+10} = [\theta_0 \alpha^{i} + (1-\theta_0)\tau^{i}] \int_{\theta(i-1,i-1)}^{\theta(i,i)} g(\theta_u; \mu, \sigma) d\theta_u$$

where $g(\theta_u; \mu, \sigma)$ is the beta distribution of θ_u. Additionally, $\theta(x, n)$ represents the probability of a worker producing x good signals in n periods, which by Bayes' rule can be represented as

$$\theta(x,n) = \frac{\theta_0 \alpha^x (1-\alpha)^{n-x}}{\theta_0 \alpha^x (1-\alpha)^{n-x} + (1-\theta_0)\tau^x (1-\tau)^{n-x}}$$

Model Application

From above, actually running the model for this thesis requires the computation of a few integrals, in addition to minimizing the objective function Q with respect to the parameters of Ω. Functions for these computations are below.

```
# Want to define the function pi_i(omega) that can calculate it for each value of i

# Calculating bounds for the beta density integral

theta_bounds <- function(n, theta_0, alpha, tau){

  # n <- theta_inputs[1]
  # theta_0 <- theta_inputs[2]
  # alpha <- theta_inputs[3]
  # tau <- theta_inputs[4]

  num <- theta_0*(alpha^n)
  den <- theta_0*(alpha^n) + (1-theta_0)*(tau^n)

  return(num / den)
}

# Calculating the beta density integral
# Bounds change depending on whether it's an entry or exit

beta_integral <- function(exit, n, a, b, theta_0, alpha, tau){

  # exit <- beta_inputs[1]
  # n <- beta_inputs[2]
  # a <- beta_inputs[3]
  # b <- beta_inputs[4]
  # theta_0 <- beta_inputs[5]
  # alpha <- beta_inputs[6]
  # tau <- beta_inputs[7]

  bound_lower <- theta_bounds(n-1, theta_0, alpha, tau)
  bound_upper <- if_else(exit, 1, theta_bounds(n, theta_0, alpha, tau))

  integrand <- function(x){

    num <- (x^(a-1)) * ((1-x)^(b-1))
    den <- beta(a, b)

    return(num / den)
  }
```

```r
  return(integrate(integrand, lower = bound_lower, upper = bound_upper)$value)
}

# Gets the multiplication term

mult_term <- function(exit, n, theta_0, alpha, tau){

  # exit <- mult_inputs[1]
  # n <- mult_inputs[2]
  # theta_0 <- mult_inputs[3]
  # alpha <- mult_inputs[4]
  # tau <- mult_inputs[5]

  val <- if_else(exit,
                 theta_0*(alpha^(n-1))*(1-alpha) + (1-theta_0)*(tau^(n-1))*(1-tau),
                 theta_0*(alpha^n) + (1-theta_0)*(tau^n))

  return(val)
}

# Main function to optimize, using functions created above

fn_optim <- function(input_vals){ # making this a vector for optimization

  a <- input_vals[1]
  b <- input_vals[2]
  theta_0 <- input_vals[3]
  alpha <- input_vals[4]
  tau <- input_vals[5]

  fn_val <- 0

  for(i in 1:(2*n_years)){ # USE YEAR INSTEAD OF CELL!

    # Filtering the observed data to the cell in question

    data <- player_mvmt %>%
      filter(cell == i)

    # Isolating variables for ease of use

    obsv <- data$pct # observed probability
    cell <- data$cell # cell index
    yr <- data$year # year of the data
    exit <- if_else(cell <= n_years, T, F) # whether it's an exit period

    # Value for predicted probability

    add_val <- mult_term(exit, yr, theta_0, alpha, tau) *
      beta_integral(exit, yr, a, b, theta_0, alpha, tau)

    fn_val <- fn_val + (obsv - add_val)^2
  }
  return(fn_val)
}

# Inequality function

fn_ineq <- function(input_vals){

  return(input_vals[3] - input_vals[2]) # ensures that alpha >= tau

}
```

After declaring these functions, we want to actually run the model. We do so by creating a set of initial guess parameters, which have upper and lower sanity-check bounds (to be altered?). The version run here works off of these constraints, and commented out is an unconstrained version that is less accurate but is retained for potential use.