# Week 7: Logit Estimation

ResEcon 703: Topics in Advanced Econometrics

Matt Woerman
University of Massachusetts Amherst

## Agenda

Last week

- Maximum likelihood estimation

This week's topics

- Logit model recap
- Logit estimation
- Alternate logit estimation methods
- Logit estimation R example

This week's reading

- Train textbook, chapters 3.7–3.8

## Logit Model Recap

## Course So Far

So far we have covered

- Structural econometric models
- Discrete choice framework
- Random utility model
- Logit model
- Maximum likelihood estimation
- Numerical optimization

We can finally put all these pieces together and estimate a structural econometric model ourselves!

But first, a recap of the random utility and logit models

## Random Utility Model Recap

A decision maker chooses the alternative that maximizes utility

- A decision maker, $n$, faces a choice among $J$ discrete alternatives
- Alternative $j$ provides utility $U_{nj}$ (where $j = 1, \dots, J$)
- $n$ chooses $i$ if and only if $U_{ni} > U_{nj}\ \forall j \neq i$

We (the econometricians) do not observe utility $U_{nj}$, so we model it as being composed of

- $V_{nj}$: Utility from observed attributes
- $\varepsilon_{nj}$: Utility from unobserved attributes, which we treat as random

$$U_{nj} = V_{nj} + \varepsilon_{nj}$$

The probability that decision maker $n$ chooses alternative $i$ is

$$\begin{aligned} P_{ni} &= \Pr(U_{ni} > U_{nj}\ \forall j \neq i) \\ &= \int_{\boldsymbol{\varepsilon}} \mathbb{1}(\varepsilon_{nj} - \varepsilon_{ni} < V_{ni} - V_{nj}\ \forall j \neq i) f(\boldsymbol{\varepsilon}_n) d\boldsymbol{\varepsilon}_n \end{aligned}$$

## Logit Model Recap

The logit model makes a simple (but sometimes overly strong) assumption about the joint density of unobserved utility, $f(\varepsilon_n)$

$$\varepsilon_{nj} \sim \text{i.i.d. type I extreme value (Gumbel) with } Var(\varepsilon_{nj}) = \frac{\pi^2}{6}$$

This assumption yields a closed-form expression for choice probabilities

$$P_{ni} = \frac{e^{V_{ni}}}{\sum_j e^{V_{nj}}}$$

These choice probabilities make estimation relatively easy, but they imply rigid and often unrealistic substitution patterns

- Independence of irrelevant alternatives (IIA)
- Proportional substitution

## Logit Estimation

## Probability Mass Function of Logit Model

We observe the following data about a discrete choice setting

- $\boldsymbol{y}$: Vector of choices
  - $y_{ni} = 1$ if and only if decision maker $n$ chooses alternative $i$
- $\boldsymbol{X}$: Matrix of data including attributes of alternatives and information on decision makers

Under the random utility model, $y_{ni}$ is a binary (Bernoulli) random variable with conditional probability mass function equal to the choice probability

$$\Pr(y_{ni} = 1 \mid \boldsymbol{x}_n, \boldsymbol{\theta}) = P_{ni}(\boldsymbol{x}_n, \boldsymbol{\theta})$$

Under the logit assumption, this PMF and choice probability are

$$\Pr(y_{ni} = 1 \mid \boldsymbol{x}_n, \boldsymbol{\theta}) = P_{ni}(\boldsymbol{x}_n, \boldsymbol{\theta}) = \frac{e^{V_{ni}(\boldsymbol{x}_{ni}, \boldsymbol{\theta})}}{\sum_j e^{V_{nj}(\boldsymbol{x}_{nj}, \boldsymbol{\theta})}}$$

We will use this distributional assumption to estimate the MLE, $\widehat{\boldsymbol{\theta}}$

## Joint Probability Mass Function of Logit Model

A decision maker chooses exactly one alternative, so the probability of observing the decision maker's vector of choices equals the choice probability of the observed choice

$$\Pr(\boldsymbol{y}_n \mid \boldsymbol{x}_n, \boldsymbol{\theta}) = \prod_{i=1}^{J} [P_{ni}(\boldsymbol{x}_n, \boldsymbol{\theta})]^{y_{ni}}$$

Assuming each decision maker is independent, the probability of observing the entire vector of choices is

$$\Pr(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta}) = \prod_{n=1}^{N} \prod_{i=1}^{J} [P_{ni}(\boldsymbol{x}_n, \boldsymbol{\theta})]^{y_{ni}}$$

Under the logit assumption, this joint PMF is

$$\Pr(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta}) = \prod_{n=1}^{N} \prod_{i=1}^{J} \left[ \frac{e^{V_{ni}(\boldsymbol{x}_{ni}, \boldsymbol{\theta})}}{\sum_j e^{V_{nj}(\boldsymbol{x}_{nj}, \boldsymbol{\theta})}} \right]^{y_{ni}}$$

## Logit Likelihood Function

The joint PMF implies that we know $\boldsymbol{X}$ and $\boldsymbol{\theta}$ and want to know probabilities of $\boldsymbol{y}$, but we actually know $\boldsymbol{y}$ and $\boldsymbol{X}$ and want to estimate $\boldsymbol{\theta}$

We simply switch the conditioning on this joint PMF to yield the likelihood function of the parameters, $\boldsymbol{\theta}$, conditional on the data, $\boldsymbol{y}$ and $\boldsymbol{X}$

$$L(\boldsymbol{\theta} \mid \boldsymbol{y}, \boldsymbol{X}) = \prod_{n=1}^{N} \prod_{i=1}^{J} [P_{ni}(\boldsymbol{x}_n, \boldsymbol{\theta})]^{y_{ni}}$$

Under the logit assumption, this likelihood function is

$$L(\boldsymbol{\theta} \mid \boldsymbol{y}, \boldsymbol{X}) = \prod_{n=1}^{N} \prod_{i=1}^{J} \left[ \frac{e^{V_{ni}(\boldsymbol{x}_{ni}, \boldsymbol{\theta})}}{\sum_j e^{V_{nj}(\boldsymbol{x}_{nj}, \boldsymbol{\theta})}} \right]^{y_{ni}}$$

## Logit Log-Likelihood Function

Then the log-likelihood function of the parameters, $\boldsymbol{\theta}$, conditional on the data, $\boldsymbol{y}$ and $\boldsymbol{X}$, is

$$\ln L(\boldsymbol{\theta} \mid \boldsymbol{y}, \boldsymbol{X}) = \sum_{n=1}^{N} \sum_{i=1}^{J} y_{ni} \ln P_{ni}(\boldsymbol{x}_n, \boldsymbol{\theta})$$

Under the logit assumption, this likelihood function is

$$\ln L(\boldsymbol{\theta} \mid \boldsymbol{y}, \boldsymbol{X}) = \sum_{n=1}^{N} \sum_{i=1}^{J} y_{ni} \ln \left[ \frac{e^{V_{ni}(\boldsymbol{x}_{ni}, \boldsymbol{\theta})}}{\sum_j e^{V_{nj}(\boldsymbol{x}_{nj}, \boldsymbol{\theta})}} \right]$$

If we further assume that representative utility is liner, $V_{ni} = \beta' \boldsymbol{x}_{ni}$, then this log-likelihood function is

$$\ln L(\beta \mid \boldsymbol{y}, \boldsymbol{X}) = \sum_{n=1}^{N} \sum_{i=1}^{J} y_{ni} \ln \left[ \frac{e^{\beta' \boldsymbol{x}_{ni}}}{\sum_j e^{\beta' \boldsymbol{x}_{nj}}} \right]$$

# Logit Maximum Likelihood Estimator

The maximum likelihood estimator is the set of parameters that maximizes this log-likelihood function

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \ln L(\boldsymbol{\theta} \mid \boldsymbol{y}, \boldsymbol{X})$$

These parameters make it most likely to generate the choices that you observe, conditional on the data describing the choice setting

- The MLE makes the choice probability close to one for chosen alternatives and close to zero for alternatives that are not chosen

The first-order conditions for maximization are

$$\frac{\partial \ln L(\boldsymbol{\theta} \mid \boldsymbol{y}, \boldsymbol{X})}{\partial \boldsymbol{\theta}} = \mathbf{0}$$

These first-order conditions do not have a closed-form solution, so we will need to use numerical optimization to find the MLE

## Alternate Logit Estimation Methods

## Logit Moment Conditions and GMM

If we assume representative utility is linear, $V_{ni} = \beta' \boldsymbol{x}_{ni}$, then the logit MLE first-order conditions are equivalent to

$$\sum_{n=1}^{N} \sum_{i=1}^{J} \left[ y_{ni} - P_{ni}(\boldsymbol{x}_n, \beta) \right] \boldsymbol{x}_n = \boldsymbol{0}$$

This expression implies the orthogonality of econometric residuals, $y_{ni} - P_{ni}(\boldsymbol{x}_n, \beta)$, and the data, $\boldsymbol{x}_n$

- Analogous to OLS finding the parameters that make the residuals orthogonal to the data

These equalities are sample moment conditions that we could use to estimate the logit parameters by GMM

- More on GMM next week

## Endogeneity in the Logit Model

The previous slides require that the data describing the choice settings, $\boldsymbol{X}$, are exogenous

- If the data are endogenous to the choice, then coefficients may be inconsistent
  - The institution is the same as endogeneity in an OLS regression

If we have some exogenous instruments that generate variation in our data, we can use a kind of instrumental variables method to estimate consistent parameters

- We will construct moment conditions using these instruments, rather than the data, and solve with GMM
- More on this topic next week

## Estimation Using a Subset of Alternatives

In some settings, it is computationally infeasible to consider the full choice set for every decision maker

- We can instead consider only a subset of the alternatives available to each decision maker

If we sample the alternatives for each decision maker in such a way that each alternative has an equal probability of being considered, we can use the standard ML method from the previous slides

- In this case, the MLE is consistent but not efficient

If we sample with unequal probabilities, we need to make a small adjustment to estimation to account for this unequal sampling

- We may want unequal probabilities when some alternatives have very low take-up
- See the Train textbook for details of this the estimation method

## Choice-Based Sampling of Decision Makers

The previous slides require that the sampling of decision makers is random (or at least exogenous to the choice)

- If the sample is endogenous to the choice, then coefficients may be inconsistent
    - Example: What could go wrong if you take a survey of commute choices, but you take the survey at the bus stop?

In some case, we may want to oversample some choices

- If an important alternative has low take-up, we may want to oversample those decision makers to obtain more information about that choice

We can recover consistent estimates from a choice-based sample of decision makers, but estimation is more complex and depends on the specific sampling method used

- See the Train textbook for more details and references

## Logit Estimation R Example

## Multinomial Choice Example

We are studying how consumers make choices about expensive and highly energy-consuming appliances in their homes, but now with different data

- We have (real) data on 900 households in California and the type of heating system in their home. Each household has the following choice set, and we observe the following data

Choice set

- `gc`: gas central
- `gr`: gas room
- `ec`: electric central
- `er`: electric room
- `hp`: heat pump

Alternative-specific data

- `ic`: installation cost
- `oc`: annual operating cost

Household demographic data

- `income`: annual income
- `agehed`: age of household head
- `rooms`: number of rooms
- `region`: home location

## Load Dataset

The `Heating` dataset is part of the `mlogit` package, so we can load it using the `data()` function

```r
## Load tidyverse and mlogit
library(tidyverse)
library(mlogit)
## Load dataset from mlogit package
data('Heating', package = 'mlogit')
## Rename dataset to lowercase
heating <- Heating
```

# Dataset

```
## Look at dataset
tibble(heating)
## # A tibble: 900 x 16
##    idcase depvar ic.gc ic.gr ic.ec ic.er ic.hp oc.gc oc.gr oc.ec oc.er
##     <dbl> <fct>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1      1 gc      866   963.  860.  996. 1136.  200.  152.  553.  506.
##  2      2 gc      728.  759.  797.  895.  969.  169.  169.  520.  486.
##  3      3 gc      599.  783.  720.  900. 1048.  166.  138.  439.  405.
##  4      4 er      835.  793.  761.  831. 1049.  181.  147.  483   425.
##  5      5 er      756.  846.  859.  986.  883.  175.  139.  404.  390.
##  6      6 gc      666.  842.  694.  863.  859.  136.  141.  398.  371.
##  7      7 gc      670.  941.  634.  952. 1087.  192.  148.  478.  446.
##  8      8 gc      778. 1022.  813. 1012.  990.  188.  159.  502.  465.
##  9      9 gc      928. 1212.  876. 1025. 1232.  169.  190.  553.  452.
## 10     10 gc      683. 1045.  776.  874.  878.  176.  136.  532.  472.
## # ... with 890 more rows, and 5 more variables: oc.hp <dbl>,
## #   income <dbl>, agehed <dbl>, rooms <dbl>, region <fct>
```

## Random Utility Model of Heating System Choice

We model the utility to household $n$ of installing heating system $j$ as

$$U_{nj} = V_{nj} + \varepsilon_{nj}$$

where $V_{nj}$ depends on the data about alternative $j$ and household $n$

The probability that household $n$ installs heating system $i$ is

$$P_{ni} = \int_{\boldsymbol{\varepsilon}} \mathbb{1}(\varepsilon_{nj} - \varepsilon_{ni} < V_{ni} - V_{nj} \; \forall j \neq i) f(\boldsymbol{\varepsilon}_n) d\boldsymbol{\varepsilon}_n$$

Under the logit assumption, these choice probabilities simplify to

$$P_{ni} = \frac{e^{V_{ni}}}{\sum_j e^{V_{nj}}}$$

## Representative Utility of Heating System Choice

What might affect the utility of the different heating systems?

- Installation cost
- Annual operating cost
- Heating system technology
    - Gas systems might be preferred to electric systems
    - Central systems might be preferred to room systems
- Anything else?

We model the representative utility of heating system $j$ to household $n$ as

$$V_{nj} = \alpha_j + \beta_1 IC_{nj} + \beta_2 OC_{nj}$$

## Multinomial Logit Model of Heating System Choice

Under the logit assumption, the choice probability that household $n$ installs heating system $i$ is

$$P_{ni} = \frac{e^{V_{ni}}}{\sum_j e^{V_{nj}}}$$

We model the representative utility of heating system $j$ to household $n$ as

$$V_{nj} = \alpha_j + \beta_1 IC_{nj} + \beta_2 OC_{nj}$$

Substituting this representative utility into the choice probabilities gives

$$P_{ni} = \frac{e^{\alpha_i + \beta_1 IC_{ni} + \beta_2 OC_{ni}}}{\sum_j e^{\alpha_j + \beta_1 IC_{nj} + \beta_2 OC_{nj}}}$$

We have six parameters to estimate using maximum likelihood

## Maximum Likelihood Estimator of Logit Model

The likelihood function of the parameters is

$$L(\boldsymbol{\theta}) = \prod_{n=1}^{N} \prod_{i=1}^{J} P_{ni}^{y_{ni}}$$

and the log-likelihood function of the parameters is

$$\ln L(\boldsymbol{\theta}) = \sum_{n=1}^{N} \sum_{i=1}^{J} y_{ni} \ln P_{ni}$$

The maximum likleihood estimator of these parameters is

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \ln L(\boldsymbol{\theta})$$

We will use numerical optimization to find the parameters values that maximize the log-likelihood function

## Numerical Optimization for MLE

We want to find the set of parameters, $\widehat{\boldsymbol{\theta}}$, that maximize the log-likelihood function, $\ln L(\boldsymbol{\theta})$

1. Begin with some initial parameter values, $\boldsymbol{\theta}^0$
2. Check if you can "walk up" to a higher value
3. If so, take a step in the right direction to $\boldsymbol{\theta}^1$
4. Repeat steps (2) and (3), stepping from $\boldsymbol{\theta}^s$ to $\boldsymbol{\theta}^{s+1}$ until you reach the maximum

The `optim()` function in R will perform this numerical optimization for us

- We just have to give the `optim()` function two things:
    - Some initial parameter values, $\boldsymbol{\theta}^0$
    - A function that will take those parameters as an argument and calculate the log-likelihood, $\ln L(\boldsymbol{\theta})$
    - (And sometimes additional information to fine-tune the optimization procedure and output)

## Optimization in R

```r
### Optimization in R
## Help file for the optimization function, optim
?optim
## Arguments for optim function
optim(par, fn, gr, ..., method, lower, upper, control, hessian)
```

`optim()` requires that you create a function, `fn`, that

1. Takes a set of parameters and other arguments as inputs
2. Calculates your objective function given those parameters
3. Returns this value of the objective function

You also have to give `optim()` arguments for

- `par`: starting parameter values
- `...`: dataset and other things needed by your function
- `method`: optimization algorithm
  - I recommend `method = 'BFGS'` for our estimation

`optim()` will find the parameters that minimize the objective function

- To maximize, minimize the negative of the objective function

## Steps to Calculate the Logit Log-Likelihood

The logit log-likelihood function is

$$\ln L(\boldsymbol{\theta}) = \sum_{n=1}^{N} \sum_{i=1}^{J} y_{ni} \ln P_{ni}$$

where the choice probabilities for our model are

$$P_{ni} = \frac{e^{\alpha_i + \beta_1 IC_{ni} + \beta_2 OC_{ni}}}{\sum_j e^{\alpha_j + \beta_1 IC_{nj} + \beta_2 OC_{nj}}}$$

Steps to calculate the logit log-likelihood conditional on $\boldsymbol{\theta}$

1. Calculate the representative utility of each alternative-household
2. Calculate the choice probability of each alternative-household
3. Calculate the log of the choice probability for each chosen alternative
4. Calculate the negative of the log-likelihood

## Function to Calculate Logit Log-Likelihood

```r
## Function to calculate log-likelihood for heating choice
ll_fn_1 <- function(params, data){
  ## Extract individual parameters with descriptive names
  alpha_ec <- params[1]
  alpha_er <- params[2]
  alpha_gc <- params[3]
  alpha_gr <- params[4]
  beta_1 <- params[5]
  beta_2 <- params[6]
  ## Calculate representative utility for each alternative given the parameters
  model_data <- data %>%
    mutate(utility_ec = alpha_ec + beta_1 * ic.ec + beta_2 * oc.ec,
           utility_er = alpha_er + beta_1 * ic.er + beta_2 * oc.er,
           utility_gc = alpha_gc + beta_1 * ic.gc + beta_2 * oc.gc,
           utility_gr = alpha_gr + beta_1 * ic.gr + beta_2 * oc.gr,
           utility_hp = beta_1 * ic.hp + beta_2 * oc.hp)
  ## Calculate logit choice probability denominator given the parameters
  model_data <- model_data %>%
    mutate(prob_denom = exp(utility_ec) + exp(utility_er) + exp(utility_gc) +
             exp(utility_gr) + exp(utility_hp))
  ## Calculate logit choice probability for each alt given the parameters
  model_data <- model_data %>%
    mutate(prob_ec = exp(utility_ec) / prob_denom,
           prob_er = exp(utility_er) / prob_denom,
           prob_gc = exp(utility_gc) / prob_denom,
```

## Function to Calculate Logit Log-Likelihood

```
  ## Calculate logit choice probability for each alt given the parameters
  model_data <- model_data %>%
    mutate(prob_ec = exp(utility_ec) / prob_denom,
           prob_er = exp(utility_er) / prob_denom,
           prob_gc = exp(utility_gc) / prob_denom,
           prob_gr = exp(utility_gr) / prob_denom,
           prob_hp = exp(utility_hp) / prob_denom)
  ## Calculate logit choice probability for chosen alt given the parameters
  model_data <- model_data %>%
    mutate(prob_choice = prob_ec * (depvar == 'ec') +
             prob_er * (depvar == 'er') + prob_gc * (depvar == 'gc') +
             prob_gr * (depvar == 'gr') + prob_hp * (depvar == 'hp'))
  ## Calculate log of logit choice probability for chosen alt given the params
  model_data <- model_data %>%
    mutate(log_prob = log(prob_choice))
  ## Calculate the log-likelihood for these parameters
  ll <- sum(model_data$log_prob)
  return(-ll)
}
```

## Estimate Logit Model

```r
## Arguments for optim function
optim(par, fn, gr, ..., method, lower, upper, control, hessian)
```

```r
## Maximize the log-likelihood function
model_1 <- optim(par = rep(0, 6), fn = ll_fn_1, data = heating,
                 method = 'BFGS', hessian = TRUE)
```

## Estimation Results

```
## Show optimization results
model_1
## $par
## [1]  1.811460030  1.986682245  1.661661689  0.255606904 -0.001603602 -0.007689238
##
## $value
## [1] 1008.337
##
## $counts
## function gradient
##       76       12
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##             [,1]         [,2]         [,3]         [,4]         [,5]        [,6]
## [1,]    58.682876    -6.268791    -39.79257    -9.042317    -974.0483   15359.361
## [2,]    -6.268791    74.971017    -52.12634   -11.874898   11305.5416   16493.310
## [3,]   -39.792565   -52.126345    205.37492   -81.829543  -31025.7987  -24006.728
## [4,]    -9.042317   -11.874898    -81.82954   109.932020   10530.5731   -7914.243
## [5,]  -974.048313 11305.541577 -31025.79874 10530.573092 8813294.1157 2781602.803
## [6,] 15359.360795 16493.309579 -24006.72752 -7914.243457 2781602.8033 9017721.751
```

# Maximum Likelihood Estimator and Hessian

```
## Show MLE parameters
model_1$par
## [1]  1.811460030  1.986682245  1.661661689  0.255606904 -0.001603602 -0.007689238

## Show Hessian at the MLE
model_1$hessian
##             [,1]        [,2]        [,3]        [,4]         [,5]        [,6]
## [1,]   58.682876   -6.268791   -39.79257   -9.042317    -974.0483   15359.361
## [2,]   -6.268791   74.971017   -52.12634  -11.874898   11305.5416   16493.310
## [3,]  -39.792565  -52.126345   205.37492  -81.829543  -31025.7987  -24006.728
## [4,]   -9.042317  -11.874898   -81.82954  109.932020   10530.5731   -7914.243
## [5,] -974.048313 11305.541577 -31025.79874 10530.573092 8813294.1157 2781602.803
## [6,] 15359.360795 16493.309579 -24006.72752 -7914.243457 2781602.8033 9017721.751
```

## Variance-Covariance Matrix

$$\widehat{Var}(\widehat{\boldsymbol{\theta}}) = \left\{ -\frac{\partial^2 \ln L(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}'}\bigg|_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}} \right\}^{-1}$$

```
## Calculate MLE variance estimator
model_1$hessian %>%
  solve()
##               [,1]          [,2]         [,3]          [,4]          [,5]          [,6]
## [1,]  1.980437e-01  1.424109e-01 1.860571e-02 -5.521102e-03  9.512776e-05 -5.824413e-04
## [2,]  1.424109e-01  1.284812e-01 7.940390e-03 -5.846949e-03  3.493597e-05 -4.723201e-04
## [3,]  1.860571e-02  7.940390e-03 5.124370e-02  3.792083e-02  9.747095e-05  9.342135e-05
## [4,] -5.521102e-03 -5.846949e-03 3.792083e-02  4.244292e-02  4.399295e-05  1.447288e-04
## [5,]  9.512776e-05  3.493597e-05 9.747095e-05  4.399295e-05  3.843712e-07 -4.639197e-08
## [6,] -5.824413e-04 -4.723201e-04 9.342135e-05  1.447288e-04 -4.639197e-08  2.356832e-06
```

## Standard Errors, Z-Stats, and P-Values

$$\widehat{\boldsymbol{\theta}} \overset{a}{\sim} \mathcal{N}\left(\boldsymbol{\theta}_0, I(\boldsymbol{\theta}_0)^{-1}\right)$$

```
## Calculate MLE standard errors
model_1_se <- model_1$hessian %>%
  solve() %>%
  diag() %>%
  sqrt()
model_1_se
## [1] 0.4450210320 0.3584427563 0.2263707190 0.2060168046 0.0006199767 0.0015351975

## Calculate parameter z-stats
model_1_zstat <- model_1$par / model_1_se
model_1_zstat
## [1]  4.070504  5.542537  7.340444  1.240709 -2.586552 -5.008631

## Calculate parameter p-values
model_1_pvalue <- 2 * pnorm(q = -abs(model_1_zstat))
model_1_pvalue
## [1] 4.691147e-05 2.981204e-08 2.128857e-13 2.147133e-01 9.694147e-03 5.481861e-07
```

## Another Way to Calculate Logit Log-Likelihood

We can make our function to calculate log-likelihood more efficient if we work with a long dataset

```r
## Pivot heating dataset into a long dataset
heating_long <- heating %>%
  pivot_longer(contains('.')) %>%
  separate(name, c('name', 'alt')) %>%
  pivot_wider() %>%
  mutate(choice = (depvar == alt)) %>%
  select(-depvar) %>%
  arrange(idcase, alt)
## Add columns of ones to long dataset for alt-specific constant terms
heating_long <- heating_long %>%
  mutate(ind_ec = 1 * (alt == 'ec'),
         ind_er = 1 * (alt == 'er'),
         ind_gc = 1 * (alt == 'gc'),
         ind_gr = 1 * (alt == 'gr'))
```

## Dataset in Long Format

```
## Look at long dataset
heating_long %>%
  select(idcase, alt, choice, ic, oc, starts_with('ind'))
## # A tibble: 4,500 x 9
##    idcase alt   choice    ic    oc ind_ec ind_er ind_gc ind_gr
##     <dbl> <chr> <lgl>  <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
##  1      1 ec    FALSE   860.  553.      1      0      0      0
##  2      1 er    FALSE   996.  506.      0      1      0      0
##  3      1 gc    TRUE    866   200.      0      0      1      0
##  4      1 gr    FALSE   963.  152.      0      0      0      1
##  5      1 hp    FALSE  1136.  238.      0      0      0      0
##  6      2 ec    FALSE   797.  520.      1      0      0      0
##  7      2 er    FALSE   895.  486.      0      1      0      0
##  8      2 gc    TRUE    728.  169.      0      0      1      0
##  9      2 gr    FALSE   759.  169.      0      0      0      1
## 10      2 hp    FALSE   969.  199.      0      0      0      0
## # ... with 4,490 more rows
```

# A Different Function to Calculate Logit Log-Likelihood

```r
## Function to calculate log-likelihood for heating choice
ll_fn_2 <- function(params, data){
  ## Extract individual parameters with descriptive names
  alpha_ec <- params[1]
  alpha_er <- params[2]
  alpha_gc <- params[3]
  alpha_gr <- params[4]
  beta_1 <- params[5]
  beta_2 <- params[6]
  ## Calculate representative utility for each alternative given the parameters
  model_data <- data %>%
    mutate(utility = alpha_ec * ind_ec + alpha_er * ind_er +
             alpha_gc * ind_gc + alpha_gr * ind_gr +
             beta_1 * ic + beta_2 * oc)
  ## Calculate logit probability denominator given the parameters
  model_data <- model_data %>%
    group_by(idcase) %>%
    mutate(prob_denom = sum(exp(utility))) %>%
    ungroup()
  ## Calculate logit choice probability for each alt given the parameters
  model_data <- model_data %>%
    mutate(prob = exp(utility) / prob_denom)
  ## Keep logit choice probability for only the chosen alt given the parameters
  model_data <- model_data %>%
    filter(choice == TRUE)
  ## Calculate log of logit choice probability for chosen alt given the params
  model_data <- model_data %>%
    mutate(log_prob = log(prob))
  ## Calculate the log-likelihood for these parameters
  ll <- sum(model_data$log_prob)
  return(-ll)
}
```

## Estimate Logit Model

```r
## Maximize the log-likelihood function
model_2 <- optim(par = rep(0, 6), fn = ll_fn_2, data = heating_long,
                 method = 'BFGS', hessian = TRUE)
```

## Estimation Results

```
## Show optimization results
model_2
## $par
## [1]  1.811460030  1.986682245  1.661661689  0.255606904 -0.001603602 -0.007689238
##
## $value
## [1] 1008.337
##
## $counts
## function gradient
##       76       12
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##              [,1]         [,2]         [,3]         [,4]         [,5]        [,6]
## [1,]    58.682876    -6.268791    -39.79257    -9.042317    -974.0483   15359.361
## [2,]    -6.268791    74.971017    -52.12634   -11.874898   11305.5416   16493.310
## [3,]   -39.792565   -52.126345    205.37492   -81.829543  -31025.7987  -24006.728
## [4,]    -9.042317   -11.874898    -81.82954   109.932020   10530.5731   -7914.243
## [5,]  -974.048313 11305.541577 -31025.79874 10530.573092 8813294.1157 2781602.803
## [6,] 15359.360795 16493.309579 -24006.72752 -7914.243457 2781602.8033 9017721.751
```

# Maximum Likelihood Estimator and Hessian

```
## Show MLE parameters
model_2$par
## [1]  1.811460030  1.986682245  1.661661689  0.255606904 -0.001603602 -0.007689238

## Show Hessian at the MLE
model_2$hessian
##             [,1]         [,2]         [,3]         [,4]         [,5]        [,6]
## [1,]    58.682876    -6.268791    -39.79257    -9.042317    -974.0483   15359.361
## [2,]    -6.268791    74.971017    -52.12634   -11.874898   11305.5416   16493.310
## [3,]   -39.792565   -52.126345    205.37492   -81.829543  -31025.7987  -24006.728
## [4,]    -9.042317   -11.874898    -81.82954   109.932020   10530.5731   -7914.243
## [5,]  -974.048313 11305.541577 -31025.79874 10530.573092 8813294.1157 2781602.803
## [6,] 15359.360795 16493.309579 -24006.72752 -7914.243457 2781602.8033 9017721.751
```

## Variance-Covariance Matrix

```
## Calculate MLE variance estimator
model_2$hessian %>%
  solve()
##               [,1]          [,2]         [,3]          [,4]          [,5]          [,6]
## [1,]  1.980437e-01  1.424109e-01 1.860571e-02 -5.521103e-03  9.512776e-05 -5.824413e-04
## [2,]  1.424109e-01  1.284812e-01 7.940390e-03 -5.846949e-03  3.493597e-05 -4.723201e-04
## [3,]  1.860571e-02  7.940390e-03 5.124370e-02  3.792083e-02  9.747095e-05  9.342135e-05
## [4,] -5.521103e-03 -5.846949e-03 3.792083e-02  4.244292e-02  4.399295e-05  1.447288e-04
## [5,]  9.512776e-05  3.493597e-05 9.747095e-05  4.399295e-05  3.843712e-07 -4.639197e-08
## [6,] -5.824413e-04 -4.723201e-04 9.342135e-05  1.447288e-04 -4.639197e-08  2.356832e-06
```

## Standard Errors, Z-Stats, and P-Values

```r
model_2_se <- model_2$hessian %>%
  solve() %>%
  diag() %>%
  sqrt()
model_2_se
## [1] 0.4450210322 0.3584427563 0.2263707192 0.2060168049 0.0006199767 0.0015351975

## Calculate parameter z-stats
model_2_zstat <- model_2$par / model_2_se
model_2_zstat
## [1]  4.070504  5.542537  7.340444  1.240709 -2.586552 -5.008631

## Calculate parameter p-values
model_2_pvalue <- 2 * pnorm(q = -abs(model_2_zstat))
model_2_pvalue
## [1] 4.691147e-05 2.981204e-08 2.128857e-13 2.147133e-01 9.694147e-03 5.481861e-07
```

## Yet Another Function to Calculate Logit Log-Likelihood

We can make our function to calculate log-likelihood even more efficient if we work with matrices

```r
## Function to calculate log-likelihood for heating choice
ll_fn_3 <- function(params, data){
  ## Select data for X and convert to a matrix [(N * J) x K]
  X <- data %>%
    select(starts_with('ind'), ic, oc) %>%
    as.matrix()
  ## Calculate representative utility for each alternative [(N * J) x 1]
  utility <- X %*% params
  ## Convert utility to a wide matrix [N x J]
  utility <- matrix(data = utility, ncol = 5, byrow = TRUE)
  ## Calculate logit probability denominator [N x 1]
  prob_denom <- rowSums(exp(utility))
  ## Calculate logit choice probability for each alternative [N x J]
  prob <- exp(utility) / prob_denom
  ## Select data for y and convert to a wide matrix [N x J]
  y <- matrix(data = data$choice, ncol = 5, byrow = TRUE)
  ## Calculate logit choice probability for the chosen alternatives [N x 1]
  prob_choice <- rowSums(y * prob)
  ## Calculate log of logit choice probability for chosen alternatives [N x 1]
  log_prob <- log(prob_choice)
  ## Calculate the log-likelihood [1 x 1]
  ll <- sum(log_prob)
  return(-ll)
}
```

## Estimate Logit Model

```r
## Maximize the log-likelihood function
model_3 <- optim(par = rep(0, 6), fn = ll_fn_3, data = heating_long,
                 method = 'BFGS', hessian = TRUE)
```

# Estimation Results

```
## Show optimization results
model_3
## $par
## [1]  1.811460030  1.986682245  1.661661689  0.255606904 -0.001603602 -0.007689238
##
## $value
## [1] 1008.337
##
## $counts
## function gradient
##       76       12
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##              [,1]         [,2]         [,3]         [,4]         [,5]        [,6]
## [1,]    58.682876    -6.268791    -39.79257    -9.042317    -974.0483   15359.361
## [2,]    -6.268791    74.971017    -52.12634   -11.874898   11305.5416   16493.310
## [3,]   -39.792565   -52.126345    205.37492   -81.829543  -31025.7987  -24006.728
## [4,]    -9.042317   -11.874898    -81.82954   109.932020   10530.5731   -7914.243
## [5,]  -974.048313 11305.541577 -31025.79874 10530.573092 8813294.1157 2781602.803
## [6,] 15359.360795 16493.309579 -24006.72752 -7914.243457 2781602.8033 9017721.751
```

# Maximum Likelihood Estimator and Hessian

```
## Show MLE parameters
model_3$par
## [1]  1.811460030  1.986682245  1.661661689  0.255606904 -0.001603602 -0.007689238

## Show Hessian at the MLE
model_3$hessian
##             [,1]         [,2]         [,3]         [,4]         [,5]        [,6]
## [1,]    58.682876    -6.268791    -39.79257    -9.042317    -974.0483   15359.361
## [2,]    -6.268791    74.971017    -52.12634   -11.874898   11305.5416   16493.310
## [3,]   -39.792565   -52.126345    205.37492   -81.829543  -31025.7987  -24006.728
## [4,]    -9.042317   -11.874898    -81.82954   109.932020   10530.5731   -7914.243
## [5,]  -974.048313 11305.541577 -31025.79874 10530.573092 8813294.1157 2781602.803
## [6,] 15359.360795 16493.309579 -24006.72752 -7914.243457 2781602.8033 9017721.751
```

# Variance-Covariance Matrix

```
## Calculate MLE variance estimator
model_3$hessian %>%
  solve()
##               [,1]          [,2]         [,3]          [,4]          [,5]          [,6]
## [1,]  1.980437e-01  1.424109e-01 1.860571e-02 -5.521103e-03  9.512776e-05 -5.824413e-04
## [2,]  1.424109e-01  1.284812e-01 7.940390e-03 -5.846949e-03  3.493597e-05 -4.723201e-04
## [3,]  1.860571e-02  7.940390e-03 5.124370e-02  3.792083e-02  9.747095e-05  9.342135e-05
## [4,] -5.521103e-03 -5.846949e-03 3.792083e-02  4.244292e-02  4.399295e-05  1.447288e-04
## [5,]  9.512776e-05  3.493597e-05 9.747095e-05  4.399295e-05  3.843712e-07 -4.639197e-08
## [6,] -5.824413e-04 -4.723201e-04 9.342135e-05  1.447288e-04 -4.639197e-08  2.356832e-06
```

## Standard Errors, Z-Stats, and P-Values

```
## Calculate MLE standard errors
model_3_se <- model_3$hessian %>%
  solve() %>%
  diag() %>%
  sqrt()
model_3_se
## [1] 0.4450210322 0.3584427563 0.2263707192 0.2060168049 0.0006199767 0.0015351975

## Calculate parameter z-stats
model_3_zstat <- model_3$par / model_3_se
model_3_zstat
## [1]  4.070504  5.542537  7.340444  1.240709 -2.586552 -5.008631

## Calculate parameter p-values
model_3_pvalue <- 2 * pnorm(q = -abs(model_3_zstat))
model_3_pvalue
## [1] 4.691147e-05 2.981204e-08 2.128857e-13 2.147133e-01 9.694147e-03 5.481861e-07
```

## Likelihood Ratio Test

We can test hypotheses about the model parameters using the likelihood ratio test

We will test if these households have heating technology preferences

- Are all of the the alternative-specific intercepts equal zero?

$$H_0 : \begin{pmatrix} \alpha_{ec} \\ \alpha_{er} \\ \alpha_{gc} \\ \alpha_{gr} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

## Likelihood Ratio Test Statistic

The likelihood ratio test statistic is

$$-2\ln\lambda = 2\left(\ln L(\widehat{\boldsymbol{\theta}}_U) - \ln L(\widehat{\boldsymbol{\theta}}_R)\right)$$

and is distributed $\chi^2$ with degrees of freedom equal to the number of model restrictions

$$-2\ln\lambda \sim \chi^2(4)$$

We need both the unrestricted MLE, $\widehat{\boldsymbol{\theta}}_U$, and the restricted MLE, $\widehat{\boldsymbol{\theta}}_R$, to calculate this test statistic

- $\widehat{\boldsymbol{\theta}}_U$: MLE from the full model that we have already estimated
- $\widehat{\boldsymbol{\theta}}_R$: MLE from a model with no alternative-specific intercepts

We need to find the MLE of a model with representative utility

$$V_{nj} = \beta_1 IC_{nj} + \beta_2 OC_{nj}$$

# Function to Calculate Restricted Log-Likelihood

```r
## Function to calculate restricted log-likelihood for heating choice
ll_fn_2_rest <- function(params, data){
  ## Extract individual parameters with descriptive names
  beta_1 <- params[1]
  beta_2 <- params[2]
  ## Calculate representative utility for each alternative given the parameters
  model_data <- data %>%
    mutate(utility = beta_1 * ic + beta_2 * oc)
  ## Calculate logit probability denominator given the parameters
  model_data <- model_data %>%
    group_by(idcase) %>%
    mutate(prob_denom = sum(exp(utility))) %>%
    ungroup()
  ## Calculate logit choice probability for each alt given the parameters
  model_data <- model_data %>%
    mutate(prob = exp(utility) / prob_denom)
  ## Keep logit choice probability for only the chosen alt given the parameters
  model_data <- model_data %>%
    filter(choice == TRUE)
  ## Calculate log of logit choice probability for chosen alt given the params
  model_data <- model_data %>%
    mutate(log_prob = log(prob))
  ## Calculate the log-likelihood for these parameters
  ll <- sum(model_data$log_prob)
  return(-ll)
}
```

# Estimate Logit Model

```r
## Maximize the log-likelihood function
model_2_rest <- optim(par = rep(0, 2), fn = ll_fn_2_rest,
                      data = heating_long,
                      method = 'BFGS', hessian = TRUE)
```

## Estimation Results

```
## Show optimization results
model_2_rest
## $par
## [1] -0.006232787 -0.004580007
##
## $value
## [1] 1095.237
##
## $counts
## function gradient
##      112        9
##
## $convergence
## [1] 0
##
## $message
## NULL
##
## $hessian
##           [,1]      [,2]
## [1,] 8075639.7  473398.1
## [2,]  473398.1 9730685.6
```

## Likelihood Ratio Test Statistic

The likelihood ratio test statistic is

$$-2 \ln \lambda = 2 \left( \ln L(\widehat{\boldsymbol{\theta}}_U) - \ln L(\widehat{\boldsymbol{\theta}}_R) \right)$$

```
## Look at negative of log-likelihood value for each model
model_2$value
## [1] 1008.337

model_2_rest$value
## [1] 1095.237

## Calculate likelihood ratio test statistic
lr_stat <- 2 * (-model_2$value - -model_2_rest$value)
lr_stat
## [1] 173.8003
```

## Likelihood Ratio Test

The likelihood ratio test statistic is distributed $\chi^2$ with degrees of freedom equal to the number of model restrictions

$$-2 \ln \lambda \sim \chi^2(4)$$

```
## Find chi-squared critical value for 4 degrees of freedom
qchisq(0.95, 4)
## [1] 9.487729

## Test if likelihood ratio test statistic is greater than critical value
lr_stat > qchisq(0.95, 4)
## [1] TRUE

## Calculate p-value of test
1 - pchisq(lr_stat, 4)
## [1] 0
```