Discrete Choice Model

Huan Deng

April 29, 2025

Outline

Overview

- Today we will discuss discrete choice models, which set the stage for the selection model and demand model that we will discuss later.
- We will focus on static discrete choice models. Dynamic discrete choice model (DDCM) is beyond the scope of this course.
- We will discuss:
  - The structure of the discrete choice models
  - Prominent discrete choice models (Logit, GEV, and Probit)
  - Estimation techniques
- Today's lecture follows Train (2009), which in my opinion is the best introduction to discrete choice models.

An Introduction to Discrete Choice Models

Model Setup

The Choice Set

Choice Set: The set of alternatives among which a decision-maker chooses.

- Must be:
  1. **Mutually exclusive:** Choosing one alternative implies not choosing another.
  2. **Exhaustive:** All possible alternatives must be included.
  3. **Finite in number**.
- Researcher can re-define alternatives to ensure mutual exclusivity and exhaustiveness (e.g., add a "none of these" option).
- The number of alternatives must be finite.

Choice Set: Examples

- Work or not work
- Work for less than 5 hours a week, 5-20 hours a week, 20-40 hours, more than 40 hours
- Married to a spouse with college degree, married to a spouse without a college degree, remain single.
- Buying a gasoline car, an electric car, or anything else.
- Using X, Wechat, QQ, Facebook, or all others.

Derivation of Choice Probabilities

Random Utility Maximization (RUM):

- Each decision-maker n faces a set of J alternatives.
- Utility from alternative j is U_{nj}.
- Decision-maker chooses the alternative with highest utility:

$$\text{Choose } i \quad \text{if and only if} \quad U_{ni} > U_{nj} \quad \forall j \neq i.$$

- Researcher observes part of U_{nj} as V_{nj}, and treats the remainder as unobserved:

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

Interpretation of the Error

- The above decomposition is general: ε_{nj} is defined as the difference between the true utility and the part captured by researchers.
- The researcher doesn't know ε_{nj} and treat the errors as random variables.
- The researcher places a distribution on the errors, reflecting his/her belief about the values the unobserved part can take.
- This also serves to fit the data: conditional on the observed part V_{nj}, we observe people making different choices, thus we introduce the error terms to capture this.

Choice Probability (General Form)

The probability that decision-maker n chooses alternative i:

$$P_{ni} = \Pr\big(U_{ni} > U_{nj}\,\forall j \neq i\big) = \Pr\Big(\big(V_{ni}+\varepsilon_{ni}\big) > \big(V_{nj}+\varepsilon_{nj}\big)\,\forall j \neq i\Big).$$

Equivalently,

$$P_{ni} = \Pr\big(\varepsilon_{nj} - \varepsilon_{ni} < V_{ni} - V_{nj}\,\forall j \neq i\big).$$

Different discrete choice models arise from different assumptions on the joint distribution of the $\varepsilon_n = (\varepsilon_{n1}, \ldots, \varepsilon_{nJ})$.

Specific Models: Preview

- **Logit:** Assumes iid Extreme Value (Gumbel) errors.
  - Yields closed-form choice probabilities.
  - Restricts correlation patterns across alternatives.
- **GEV (Generalized Extreme Value):**
  - More general correlation structure (e.g. Nested Logit).
  - Still usually has closed-form solutions.
- **Probit:** Assumes joint normality of error terms.
  - Allows any pattern of correlation via covariance matrix.
  - Usually requires numerical simulation (no closed form).
- **Mixed Logit:**
  - Very flexible; any distribution can be approximated.
  - Often estimated by simulation.
- I will discuss the first three today.

An Introduction to Discrete Choice Models

Identification and Normalization

Identification: Only Differences in Utility Matter

- Adding a constant k to all alternatives' utilities does not change the choice:

$$U_{nj} \quad \rightarrow \quad U_{nj} + k \quad \text{does not affect which } U_{nj} \text{ is largest.}$$

- Hence, only utility *differences* are identified.
- **Alternative-specific constants:** Only relative levels matter. One can set one constant to 0 (the "base" alternative).
- **Sociodemographic variables:** Must enter in a way that distinguishes alternatives (e.g. via interactions with alternative attributes).

Identification: Scale of Utility Is Arbitrary

- Multiplying all utilities by a positive constant $\lambda > 0$ does not change the chosen alternative:

$$U_{nj} \quad \rightarrow \quad \lambda \, U_{nj}.$$

- Typically, we normalize by fixing the variance of the unobserved term (often 1).
- Sometimes we also normalize one parameter from V_{nj} (to 1)
- In Logit, we often set $\mathrm{Var}(\varepsilon_{nj}) = \frac{\pi^2}{6}$ for convenience.
- When error variances differ by segment (e.g. different regions), we estimate a *scale parameter* relative to a base segment.

Aggregation

- Discrete choice models are defined at the individual level.
- We often want *market shares* or *aggregate outcomes*.
- Nonlinear nature means:

$$\overline{P_i} \neq P_i(\overline{x}).$$

(i.e., probability at the average x is not the average of probabilities.)

Methods of Aggregation

1) Sample Enumeration

$$\hat{N}_i = \sum_{n=1}^{N} w_n P_{ni},$$

where w_n is the weight for individual n.

2) Segmentation

- Partition the population into segments with unique (x_{nj}, s_n).
- Calculate P_i for each segment and multiply by the segment size.

Forecasting

- The same aggregation approaches apply for future predictions.
- Adjust explanatory variables or segment shares to match anticipated future conditions (income growth, demographics, etc.).
- Apply the choice model to the updated data or updated segment counts.
- The result is an estimated future share for each alternative.

Conclusion

- **Discrete choice models** focus on the probability of each alternative's being chosen, grounded in a RUM framework.
- All major models (Logit, GEV, Probit, Mixed Logit) derive from assumptions on distribution of unobserved utility.
- **Key points:**
  - Only utility *differences* matter.
  - Scale of utility is arbitrary; must normalize error variance.
- Provides a flexible basis for understanding and predicting choice behavior across many fields.

Canonical Models

The Logit Model

Overview

- **Logit** is one of the most widely used discrete choice models.
- Assumes each alternative j has utility:

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

  where ε_{nj} are i.i.d. extreme value (Gumbel) random terms.
- Closed-form choice probabilities make logit computationally simple, especially when you have a complicated model and some parts can be greatly simplified by the closed-form.
- Key property: *independence from irrelevant alternatives* (IIA).
- IIA put strong restrictions on substitution patterns and can be unpleasant for some research questions.

Choice Probabilities: Derivation

- Assume errors ε_{nj} are **i.i.d. extreme value**:

$$f(\varepsilon_{nj}) = e^{-\varepsilon_{nj}} e^{-e^{-\varepsilon_{nj}}}, \quad F(\varepsilon_{nj}) = e^{-e^{-\varepsilon_{nj}}}.$$

- Variance of each ε_{nj} is $\pi^2/6$.
- **Logit probability** that n chooses i:

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

- Linear-in-parameters specification:

$$V_{nj} = \beta^\top x_{nj} \implies P_{ni} = \frac{\exp(\beta^\top x_{ni})}{\sum_j \exp(\beta^\top x_{nj})}.$$

Interpretation

- The *sigmoid* shape implies:
  - P_{ni} is highest when V_{ni} is large relative to V_{nj}, $j \neq i$.
  - As $V_{ni} \to \infty$, $P_{ni} \to 1$; as $V_{ni} \to -\infty$, $P_{ni} \to 0$.
- Useful to compare *ratios* of probabilities:

$$\frac{P_{ni}}{P_{nk}} = \frac{e^{V_{ni}}}{e^{V_{nk}}} = e^{V_{ni} - V_{nk}}.$$

- The logit model's closed form arises directly from the *extreme value* distribution of errors.

The Scale Parameter

- Utility can be written as: $U^*_{nj} = V_{nj} + \varepsilon^*_{nj}$, with variance $\sigma^2(\pi^2/6)$.
- Dividing by σ yields $U_{nj} = \frac{V_{nj}}{\sigma} + \varepsilon_{nj}$ where ε_{nj} now has variance $\pi^2/6$.
- Probability becomes:

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

- **Only β/σ is identified**, not β and σ separately.
- Interpretation: Larger σ implies more noise (unobserved utility), so the β estimated is "scaled down" by σ.

Independence from Irrelevant Alternatives (IIA)

- For logit, $\frac{P_{ni}}{P_{nk}} = e^{V_{ni} - V_{nk}}$ does **not** depend on any other alternative m.
- Introduced alternative (or changed attributes of other alternatives) does not affect the ratio P_{ni}/P_{nk}.
- **Red-bus–blue-bus problem:**
  - Before introducing blue bus, market share for subway and red bus are both 0.5.
  - After introducing the red bus, now the share of bus (blue plus red) is $\frac{2}{3}$.
  - Since blue and red buses are perfect substitutes, we expect that the share of subway should still be 0.5, but the Logit model predicts $\frac{1}{3}$

Substitution Patterns

- **Proportional Substitution:** Improving one alternative draws proportionally from all others:

$$\text{Cross-elasticity of } P_{ni} \text{ w.r.t. } z_{nj} = -\beta_z \, z_{nj} \, P_{nj}.$$

- All cross-elasticities (for $i \neq j$) are identical.
- In many real-world applications, we suspect more nuanced patterns (e.g., "similar" goods compete more heavily).
- $\Longrightarrow$ IIA imposes strong restrictions on substitution patterns.

Repeated Choices Over Time

- Logit can be applied to **panel data** if errors are assumed i.i.d. across *both* alternatives *and* time.
- Then each choice occasion t just adds an observation to the likelihood:

$$U_{njt} = V_{njt} + \varepsilon_{njt}, \quad \varepsilon_{njt} \sim \text{i.i.d. extreme value.}$$

- State dependence (e.g. lagged utility or choice) can be included in V_{njt} if desired, but unobserved factors remain independent over time.
- If we suspect correlation in unobserved portions across time, we need more flexible approaches (probit, mixed logit).
- This explains why Logit model is not a popular choice for life-cycle models.

Consumer Surplus in Logit

- Expected consumer surplus in money metric:

$$E(\mathrm{CS}_n) = \frac{1}{\alpha_n} Emax\{V_{nj} + \varepsilon_{nj}\}$$
$$= \frac{1}{\alpha_n}[\ln\left(\sum_j e^{V_{nj}}\right) + \gamma] \quad (1)$$

  where α_n is the marginal utility of income (negative of the cost coefficient), and $\gamma \simeq 0.5772$ is the Euler constant..

- Change in expected consumer surplus when conditions change from 0 to 1:

$$\Delta E(\mathrm{CS}_n) = \frac{1}{\alpha_n}\left[\ln\left(\sum_j e^{V_{nj}^1}\right) - \ln\left(\sum_j e^{V_{nj}^0}\right)\right].$$

- "Log-sum" term: very convenient in *policy analysis*.
- Assumes α_n is constant w.r.t. income over the relevant range.

Derivatives and Elasticities

Direct Effects:

$$\frac{\partial P_{ni}}{\partial z_{ni}} = \frac{\partial V_{ni}}{\partial z_{ni}} P_{ni}(1 - P_{ni}).$$

If $V_{ni} = \beta_z \, z_{ni}$, then

$$\frac{\partial P_{ni}}{\partial z_{ni}} = \beta_z \, P_{ni}(1 - P_{ni}).$$

Cross Effects:

$$\frac{\partial P_{ni}}{\partial z_{nj}} = - \frac{\partial V_{nj}}{\partial z_{nj}} \, P_{ni} P_{nj}, \quad i \neq j.$$

Elasticities:

$$E^i_{z_{ni}} = \frac{\partial P_{ni}}{\partial z_{ni}} \frac{z_{ni}}{P_{ni}} \quad \text{and} \quad E^i_{z_{nj}} = \frac{\partial P_{ni}}{\partial z_{nj}} \frac{z_{nj}}{P_{ni}}.$$

Estimation

- **Maximum likelihood** estimation is standard:

$$\mathcal{L}(\beta) = \prod_{n=1}^{N} \prod_{j=1}^{J} [P_{nj}(\beta)]^{y_{nj}},$$

  where $y_{nj} = 1$ if n chose j else 0.
- We ususally work with the log likelihood:

$$\mathcal{LL}(\beta) = \sum_{n} \sum_{j} y_{nj} \ln P_{nj}(\beta).$$

- In practice, log-likelihood is concave in linear-in-parameters logit, making estimation straightforward.
- **Subsets of alternatives:** If IIA holds, we can estimate on a reduced set of alternatives without bias, saving computation.
- **Choice-based sampling:** More complex weighting schemes needed if sample is not exogenous (see Manski & McFadden).

Estimation

- For convenience, let the representative utility be linear in parameters: $V_{nj} = \beta' x_{nj}$, then the first-order condition is:

$$\sum_n \sum_i (y_{ni} - P_{ni}) x_{ni} = 0. \tag{2}$$

- Rearranging and dividing both sides by N, we have:

$$\frac{1}{N} \sum_n \sum_i y_{ni} x_{ni} = \frac{1}{N} \sum_n \sum_i P_{ni} x_{ni}. \tag{3}$$

- The maximum likelihood estimates of β are those that make the predicted average of each explanatory variable equal to the observed average in the sample.
- Suppose we include alternative-specific intercept, then we have:

$$\frac{1}{N} \sum_n \sum_i y_{ni} d_i^j = \frac{1}{N} \sum_n \sum_i P_{ni} d_i^j \tag{4}$$

so the predicted share is equal to the observed share: $S_j = \hat{S}_j$

Conclusion

- Logit model is powerful and easy to implement, with a closed-form solution.
- Key assumption: *i.i.d. extreme value* errors $\Rightarrow$ IIA property.
- Handles systematic (observed) taste variation but not random (unobserved) taste variation.
- Proportional substitution is a strong limitation in some contexts.
- Many benefits (log-sum for consumer surplus, simple estimation), but *be aware* of potential misspecifications (IIA, no correlation over time).

Canonical Models

GEV

Overview of GEV Models

- GEV models extend the logit framework by allowing correlated error terms across alternatives.
- They maintain a closed-form expression for choice probabilities (no simulation needed).
- Examples of GEV models:
  - Nested Logit (NL)
  - Paired Combinatorial Logit (PCL)
  - Heteroskedastic Logit (HEV)
  - More general specifications (Cross-Nested Logit, Generalized Nested Logit, etc.)
- Retain key features of random utility maximization while relaxing the strict Independence from Irrelevant Alternatives (IIA).

Nested Logit: The Basic Idea

- Partition the set of J alternatives into K *nests*.
- Within each nest, unobserved utility terms are correlated (common unobserved factors).
- Between nests, unobserved utility terms are uncorrelated.
- Example: `Auto Alone` & `Carpool` as one nest, `Bus` & `Rail` as another, capturing correlations within auto modes vs. transit modes.

Nested Logit Choice Probabilities

Representative utility: $U_{nj} = V_{nj} + \varepsilon_{nj}$.
Assumption: ε_{nj} has a GEV distribution allowing correlation within nests: $\exp\left(-\sum_{k=1}^{K}\left(\sum_{j\in B_k} e^{-\varepsilon_{nj}/\lambda_k}\right)^{\lambda_k}\right)$

Two-level nested logit probability:

$$P_{ni} = \frac{e^{\frac{V_{ni}}{\lambda_k}}\left(\sum_{j\in B_k} e^{\frac{V_{nj}}{\lambda_k}}\right)^{\lambda_k - 1}}{\sum_{\ell=1}^{K}\left(\sum_{j\in B_\ell} e^{\frac{V_{nj}}{\lambda_\ell}}\right)^{\lambda_\ell}} \quad \text{for } i \in B_k.$$

- B_k = set (nest) of alternatives in nest k.
- $\lambda_k \in (0, 1]$ measures correlation in nest k; $\lambda_k = 1$ recovers logit within that nest.
- If $\lambda_k = 1$ for all k, the model collapses to standard logit (no correlation).

IIA and IIN in Nested Logit Models

- The ratio of choice probabilities:

$$\frac{P_{ni}}{P_{nm}} = \frac{e^{V_{ni}/\lambda_k}\left(\sum_{j\in B_k} e^{V_{nj}/\lambda_k}\right)^{\lambda_k-1}}{e^{V_{nm}/\lambda_\ell}\left(\sum_{j\in B_\ell} e^{V_{nj}/\lambda_\ell}\right)^{\lambda_\ell-1}}. \tag{5}$$

- If $k = \ell$ (i.e., i and m are in the same nest), then the factors in parentheses cancel out and we have:

$$\frac{P_{ni}}{P_{nm}} = \frac{e^{V_{ni}/\lambda_k}}{e^{V_{nm}/\lambda_k}}. \tag{6}$$

  IIA holds for alternatives within the same nest.

- For alternatives in different nests, IIA no longer holds, but IIN holds ("independence from irrelevant nests")
- So for nested Logit, IIA holds over alternatives in each nest and IIN holds over alternatives in different nests.

Interpretation via Two Logits

- Decompose P_{ni} into $P_{nB_k} \times P_{ni|B_k}$:

$$P_{nB_k} = \frac{\exp\left(W_{nk} + \lambda_k I_{nk}\right)}{\sum_{\ell=1}^{K} \exp\left(W_{n\ell} + \lambda_\ell I_{n\ell}\right)}, \qquad P_{ni|B_k} = \frac{\exp\left(Y_{ni}/\lambda_k\right)}{\sum_{j \in B_k} \exp\left(Y_{nj}/\lambda_k\right)},$$

  where:
  - W_{nk} = common part of V_{nj} for nest k,
  - Y_{nj} = alternative-specific part,
  - $I_{nk} = \ln \sum_{j \in B_k} e^{Y_{nj}/\lambda_k}$ = *inclusive value* of nest k.
- P_{nB_k} is a logit over nests; $P_{ni|B_k}$ is a logit within nest k.
- λ_k (the *log-sum coefficient*) connects them, capturing the correlation structure.

Three-Level Nested Logit

- Extension: partition alternatives into *nests*, and then sub-partition each nest into *subnests*.
- Probability decomposes into a product of three logit formulas (top, middle, bottom).
- Each level has its own "inclusive value" and coefficient capturing correlation within subnests and nests.
- Example: Housing choice:
  - **Level 1:** Neighborhood choice,
  - **Level 2:** Number of bedrooms (subnests),
  - **Level 3:** Specific housing unit.
- Requires $0 < \lambda_k < 1$ (and < 1 for sub-level coefficients) for consistency with utility maximization over all attributes.

Overlapping Nests

- In standard nested logit, each alternative belongs to exactly one nest.
- Sometimes an alternative shares features with multiple nests.
- **Cross-Nested Logit (CNL):** extends nested logit to allow each alternative to be in multiple nests, often *partially* (via "allocation parameters").
- **Examples:**
  - Carpool is partly "auto" (car-based) and partly "transit-like" (not fully flexible).
  - Destination choice might share partial features of multiple neighborhoods.
- Allows more flexible correlation patterns in unobserved utility.

Paired Combinatorial Logit (PCL)

- Proposed by Chu (1981, 1989); special case of cross-nested approach.
- Each **pair** of alternatives forms a "nest."
- Parameter λ_{ij} measures correlation (or independence) between alternatives i and j.
- The choice probability:

$$P_{ni} = \frac{\sum_{j \neq i} e^{V_{ni}/\lambda_{ij}} \left(e^{V_{ni}/\lambda_{ij}} + e^{V_{nj}/\lambda_{ij}}\right)^{\lambda_{ij}-1}}{\sum_{k=1}^{J-1} \sum_{\ell=k+1}^{J} \left(e^{V_{nk}/\lambda_{k\ell}} + e^{V_{n\ell}/\lambda_{k\ell}}\right)^{\lambda_{k\ell}}}.$$

- Extremely flexible, but many parameters: need constraints/structure to keep it tractable if J is large.

Generalized Nested Logit (GNL)

- Wen & Koppelman (2001) introduced GNL as a unifying framework for cross-nested models.
- Each alternative j may belong to multiple nests k with *allocation parameters* $\alpha_{jk} \geq 0$, summing to 1 over all k for a given j.
- Coefficients λ_k define scale within each nest.
- Probability:

$$P_{ni} = \frac{\sum_k \left(\alpha_{ik}\, e^{V_{ni}}\right)^{\frac{1}{\lambda_k}} \left(\sum_{j \in B_k} \left(\alpha_{jk}\, e^{V_{nj}}\right)^{\frac{1}{\lambda_k}}\right)^{\lambda_k - 1}}{\sum_{\ell=1}^{K} \left(\sum_{j \in B_\ell} \left(\alpha_{j\ell}\, e^{V_{nj}}\right)^{\frac{1}{\lambda_\ell}}\right)^{\lambda_\ell}}.$$

- Captures partial membership in multiple nests, generalizing nested logit and PCL.

Heteroskedastic Logit (HEV)

- Also a GEV-type specification; no correlation, but *different variances* of errors by alternative.
- Utility: $U_{nj} = V_{nj} + \varepsilon_{nj}$, with ε_{nj} i.i.d. extreme value but $\mathrm{Var}(\varepsilon_{nj}) = (\theta_j \pi)^2/6$.
- Probability needs 1-D integration (no closed-form), typically handled by quadrature or simple simulation:

$$P_{ni} = \int \left[\prod_{j \neq i} \exp\Big(-\exp\big(-[V_{ni} - V_{nj} + \theta_i\, w]/\theta_j\big) \Big) \right] \exp(-w)\, \exp\big(-e^{-w}\big)\, dw, \tag{7}$$

  where $w = \varepsilon_{ni}/\theta_i$.
- Useful when we suspect different degrees of "noise" for each alternative, but want to keep them uncorrelated.

Generating GEV Models via Function G

- McFadden (1978): define a nonnegative, homogeneous function $G(Y_1, \ldots, Y_J)$ with specific partial derivative properties.
- Then the GEV choice probability is:

$$P_i = \frac{Y_i \frac{\partial G}{\partial Y_i}}{G\big(Y_1, \ldots, Y_J\big)}, \quad \text{where } Y_j = \exp(V_j).$$

- Different choices of G yield different model structures:
  - $G = \sum Y_j \implies$ standard logit.
  - $G = \sum_{k=1}^{K} \left(\sum_{j \in B_k} Y_j^{1/\lambda_k} \right)^{\lambda_k} \implies$ nested logit.
  - $G = \sum_{k=1}^{J-1} \sum_{\ell=k+1}^{J} \left(Y_k^{1/\lambda_{k\ell}} + Y_\ell^{1/\lambda_{k\ell}} \right)^{\lambda_{k\ell}} \implies$ PCL.
  - . . . and so forth for other variants.
- Allows building custom GEV models with specific substitution patterns.

Summary

- GEV models form a broad class of discrete choice models, generalizing logit by allowing complex correlation patterns.
- **Nested Logit** remains the most commonly used GEV form.
- Nested Logit has IIA within nests and IIN across nests.

Canonical Models

Probit Model

Why Probit?

- **Limitations of Logit:**
  1. No *random taste variation* (unless fully captured by observed variables).
  2. Restrictive *substitution patterns* (IIA).
  3. Cannot handle *correlated errors* over time for the same individual (panel data).
- **Probit Model** addresses all three:
  - Allows *normally distributed random tastes*.
  - Flexible *substitution* via a general covariance structure.
  - Handles *arbitrary correlation* over time (panel data).
- Limitation: Unobserved part **normally** distributed, no closed-form for the choice probabilities, so simulation is necessary.

Choice Probabilities

Utility: $U_{nj} = V_{nj} + \varepsilon_{nj}$, with $\varepsilon_n = \langle \varepsilon_{n1}, \ldots, \varepsilon_{nJ} \rangle$.

- $\varepsilon_n \sim \mathcal{N}(0, \Sigma)$ (a multivariate normal).
- Probability of choosing i:

$$P_{ni} = \Pr(U_{ni} > U_{nj}, \forall j \neq i) = \Pr(\varepsilon_{ni} - \varepsilon_{nj} > V_{nj} - V_{ni}, \forall j \neq i).$$

- Integrals typically do not have a closed form:

$$P_{ni} = \int I\big(V_{ni} + \varepsilon_{ni} > V_{nj} + \varepsilon_{nj}, \forall j \neq i \big) \, \phi(\varepsilon_n) \, d\varepsilon_n.$$

- Must be approximated numerically (simulation).

Identification & Normalization

- **Level of utility** does not matter (adding constant k to all alternatives).
- **Scale of utility** does not matter (multiplying utility by $\lambda > 0$).
- For probit, these do *not* get normalized automatically (unlike logit).
- Must fix the scale by, e.g.:
  1. Setting variance of one error difference to 1.
  2. Or setting one diagonal element of Σ (or difference-based $\tilde{\Sigma}_i$) to 1.
- After normalization, only $\frac{J(J-1)}{2} - 1$ independent parameters remain for the covariance matrix of J alternatives (removing level & scale).

Random Coefficients (Taste Variation)

- Allow β_n to vary across individuals:

$$U_{nj} = \beta_n^\top x_{nj} + \varepsilon_{nj}, \quad \beta_n \sim \mathcal{N}(b, W).$$

- Then $U_{nj} = b^\top x_{nj} + (\beta_n - b)^\top x_{nj} + \varepsilon_{nj} = b^\top x_{nj} + \eta_{nj}$.
- η_{nj} has a *normal* covariance structure depending on x_{nj} & W.
- $\Rightarrow$ flexible representation of heterogeneity in tastes, provided tastes are normal.

Flexible Substitution Patterns

- Probit does not have the IIA constraint:

  $\frac{P_{ni}}{P_{nk}}$ can depend on attributes or existence of other alternatives.

- Covariance Σ (or structured version) dictates the substitution pattern.
  - Full Σ = complete flexibility but parameters can be less interpretable.
  - Structured Σ = fewer parameters, direct interpretation (e.g. correlation based on route overlap, etc.).
- By estimating Σ, probit adapts to the empirically revealed substitution patterns.

Probit with Panel Data

- Decision-maker n faces J alternatives in T periods.
- Utility:

$$U_{njt} = V_{njt} + \varepsilon_{njt}, \quad \varepsilon_n \sim \mathcal{N}(0, \Sigma),$$

  with dimension $(J \times T)$ in the covariance.
- Can capture correlation in unobserved factors over time *and* across alternatives.
- If T is large or J is large, dimension is high $\Rightarrow$ simulation is crucial.

Why Simulation?

- Typically we don't have **closed-form solutions**
- Must approximate $\int I(\cdot)\,\phi(\varepsilon)\,d\varepsilon$ via *Monte Carlo simulation*.
- Key is to draw ε from $\mathcal{N}(0, \Sigma)$:
  1. Compute Choleski factor L such that $L\,L^{\top} = \Sigma$.
  2. Generate standard normal vector $\eta \sim \mathcal{N}(0, I)$.
  3. Set $\varepsilon = L\,\eta$ to get $\varepsilon \sim \mathcal{N}(0, \Sigma)$.
- Evaluate whether a draw ε leads to alternative i being chosen. Repeat many times, average the results.

Accept–Reject Simulator

- **Concept:** Among R draws $\{\varepsilon^r\}_{r=1}^R$ from $N(0, \Sigma)$, check fraction that satisfies $U_{ni} > U_{nj}, \forall j \neq i$.

$$P_{ni} \approx \frac{1}{R} \sum_{r=1}^{R} I\left(V_{ni} + \varepsilon_{ni}^r > V_{nj} + \varepsilon_{nj}^r \, \forall j \neq i\right).$$

- **Issues for maximum simulated likelihood:**
  - Probability can be zero for all draws, leading to $\ln(0)$ in log-likelihood.
  - Step function in parameters $\Rightarrow$ derivatives not well-defined, optimization problems.

Smoothed Accept–Reject Simulator

- Smooths the indicator function, e.g. using logistic or normal cdf:

$$I(\Delta) \approx \frac{1}{1 + \exp(-c\,\Delta)}, \quad \Delta = (U_{ni} - U_{nj}),$$

  for some scaling c.
- Avoids zero-probability issues and is differentiable. But for large c, it approaches standard accept-reject, losing smoothness.
- Not as popular as the GHK simulator for Probit, but conceptually simpler than accept–reject and helps with zero-prob issue.

GHK Simulator (Geweke–Hajivassiliou–Keane)

- By far the most commonly used simulator for Multinomial Probit.
- **Key idea:** Factor the *multivariate* normal cdf into a product of conditional univariate cdfs, each easily simulated.
- Example for probability that $U_{n1} > U_{n2}, U_{n1} > U_{n3}, \ldots$:
  1. Order the integration bounds via a recursive scheme.
  2. Draw ε_{n1} from its marginal normal distribution.
  3. Draw ε_{n2} conditional on ε_{n1} from the conditional normal, etc.
- Highly efficient, no zero-prob issues, smooth in parameters.
- Implementation details in Train's book (Sec 5.6.3)

Estimation by Maximum Simulated Likelihood

- True log-likelihood:

$$LL(\beta) = \sum_{n=1}^{N} \sum_{j=1}^{J} y_{nj} \ln P_{nj}(\beta).$$

- No closed form for $P_{nj} \Rightarrow$ use *simulated* probability $\hat{P}_{nj}$:

$$\hat{LL}(\beta) = \sum_{n=1}^{N} \sum_{j=1}^{J} y_{nj} \ln \hat{P}_{nj}(\beta).$$

- Maximize $\hat{\ell}(\beta)$ over β (covariance parameters included).
- GHK-based $\hat{P}_{nj}$ is smooth $\Rightarrow$ standard gradient-based optimization works well.
- Under mild conditions, $\hat{\beta}$ is consistent, asymptotically normal as draws $R \to \infty$.

Conclusion

- Multinomial Probit is a powerful alternative to Logit:
  - Accommodates random coefficients.
  - Allows for flexible substitution and correlation patterns.
  - Handles repeated choices with serial correlation in errors.
- Drawback: normal distribution can be restrictive for some applications.
- Computation requires simulation (GHK approach is prevalent).

Estimation

Numeric Optimization

Motivation

- Many discrete choice models require maximizing a *log-likelihood function* for estimation.
- With complex models, no closed-form solution for the maximum exists.
- We rely on **numerical optimization methods** – iterative procedures that "search" for the maximum.
- I only cover basic methods here. Optimization is a huge topic that requires an independent course on it.

Notation

- $\text{LL}(\beta) =$ *average* log-likelihood (divided by sample size N).

$$\text{LL}(\beta) = \frac{1}{N} \sum_{n=1}^{N} \ln P_n(\beta).$$

- Goal: Find $\hat{\beta}$ that *maximizes* $\text{LL}(\beta)$.
- $\mathbf{g}_t =$ gradient vector of LL at iteration t.

$$\mathbf{g}_t = \left. \frac{\partial \text{LL}(\beta)}{\partial \beta} \right|_{\beta_t}.$$

- $\mathbf{H}_t =$ Hessian matrix of second derivatives (size $K \times K$).

Newton–Raphson (NR)

- **Idea:** Use a 2^{nd}-order Taylor expansion to approximate LL near current estimate.
- Update step:

$$\beta_{t+1} = \beta_t - \mathbf{H}_t^{-1}\,\mathbf{g}_t,$$

- $\mathbf{H}_t$ is negative definite if LL is globally concave, ensuring an upward step.
- **Pros:** Quadratic convergence if LL is near-quadratic, can be very fast.
- **Cons:** Must compute $\mathbf{H}_t$ every iteration (can be expensive), no guaranteed ascent if LL not globally concave.

BHHH (Berndt, Hall, Hall, Hausman)

- Recognizes $\mathrm{LL}(\beta)$ is a sum over N individuals:

$$\mathrm{LL}(\beta) = \frac{1}{N} \sum_{n=1}^{N} \ln P_n(\beta).$$

- Let $\mathbf{s}_n(\beta) = \nabla \ln P_n(\beta)$, i.e., score for each observation.
- Define $\mathbf{B}_t = \frac{1}{N} \sum_n \mathbf{s}_n(\beta_t)\, \mathbf{s}_n(\beta_t)^\top$.
- Iteration:

$$\beta_{t+1} = \beta_t + \lambda\, \mathbf{B}_t^{-1}\, \mathbf{g}_t,$$

- $\mathbf{B}_t$ is *positivedefinite* $\implies$ guaranteed ascent (with proper step size λ).
- Much cheaper than NR (no second derivatives).
- Fisher's information identity guarantees that B_t is a good approximation of the Hessian Matrix.

BFGS and DFP Methods

- **DFP** = Davidon–Fletcher–Powell, **BFGS** = Broyden–Fletcher–Goldfarb–Shanno.
- Both use *updates* to approximate Hessian from gradient changes across iterations (arc Hessian).
- $\mathbf{H}_t$ is approximated and refined each step with new gradient info.
- **BFGS** typically outperforms DFP in practice, widely used for general optimization.
- Effective especially when LL is *not* near-quadratic far from optimum.

Other Methods: Steepest Ascent

$$\beta_{t+1} = \beta_t + \lambda \mathbf{g}_t.$$

- Moves in direction of gradient, guaranteeing maximum increase *per small step*.
- Often slow convergence – step can be small near optimum.
- Rarely used alone, but valuable conceptually.

Convergence Criteria

- In theory, $\nabla \text{LL}(\hat{\beta}) = 0$. In practice, we accept $\hat{\beta}$ when it's "close enough."
- Common approach: Check if

$$m_t = \mathbf{g}_t^\top \left(-\mathbf{H}_t^{-1}\right) \mathbf{g}_t < \epsilon$$

  for small ϵ (e.g., 10^{-4}).
- Alternatively: Check if each component of $\mathbf{g}_t$ is small or if $\Delta\beta_{t+1}$ is tiny.
- If objective $\text{LL}(\beta_{t+1}) - \text{LL}(\beta_t) < \epsilon$, might also signal near convergence *but* can be misleading if step size is incorrectly chosen.

Local vs. Global Maximum

- Some log-likelihoods are not globally concave: they can contain multiple local maxima.
- Numerical methods might converge to a local maximum depending on starting values.
- **Practical tip:** Use different starting points; compare final solutions.
- If two local maxima differ, check which has higher LL to find the *global* maximum.
- Non-convex optimization is hard by nature and no method guarantees that we reach the global optimum.

Asymptotic Variance of $\hat{\beta}$

- For a *correctly specified* model:

$$\sqrt{N}(\hat{\beta} - \beta^*) \xrightarrow{d} \mathcal{N}\Big(0,\ (-\mathbf{H})^{-1}\Big).$$

- $\mathbf{H} \equiv$ expected Hessian in the population, β^* is true param vector.
- In practice, we estimate $\mathbf{H}$ by the sample Hessian or use $\mathbf{B}$ or $\mathbf{W}$ (outer product or covariance of scores) – all converge to the same matrix at the true parameters.

$$\text{Var}(\hat{\beta}) \ = \ \big(-\mathbf{H}^{-1}\big)/N.$$

Robust Covariance, Misspecification, & Bootstrapping

- If model is *misspecified*, let $\mathbf{V}$ = Cov of scores, $\mathbf{H}$ = average Hessian. Then asymptotic $\text{Var}(\hat{\beta}) = \mathbf{H}^{-1}\mathbf{V}\mathbf{H}^{-1}/N$.
- **Bootstrapping:** Another approach to assess standard errors:
  1. Re-sample the dataset (with replacement) many times.
  2. Re-estimate the model on each sample.
  3. Compute variance of estimates across these re-estimations.
- Useful to check distribution of estimates without heavy reliance on asymptotic formulas.

Suggestions

- BFGS or its close cousin L-BFGS is the default choice for many optimization problems in economics.
- But if your problem is complicated or high-dimensional, then might consider SGD or ADAM, which are popular in ML.
- Sometimes you might benefit by combining different algorithms. One combination I have tried is ADAM+BFGS.
- If your problem is simple and small, then every algorithm works.
- If your problem is complicated or large, then you might need to try out many algorithms
- Most packages compute derivatives using finite difference, which introduces errors. Use Autodiff whenever possible, another reason to consider Julia (but Python also offers autodiff).

Estimation

How to Draw from Densities?

Introduction

- Simulation entails drawing random realizations of ε from a density $f(\varepsilon)$.
- We compute desired statistics (e.g. choice probabilities, integrals) by averaging over these draws.
- Most programming/statistical environments provide routines for:
  - $\mathcal{N}(0,1)$: standard normal draws
  - Uniform(0,1)
- These are *pseudo-random* (deterministic with a seed), but often good enough to mimic true randomness.
- From these bases, can generate many other distributions.

Transformations of Standard Normal

- For a normal with mean b and variance σ^2:

$$\varepsilon \;=\; b \;+\; \sigma\,\eta, \quad \eta \sim \mathcal{N}(0,1).$$

- For a *lognormal*:

$$\varepsilon \;=\; \exp(b \;+\; \sigma\,\eta).$$

- E.g. if ε is lognormal(b, σ^2), then b and σ relate to ε's mean and variance in specific ways.

Inverse Cumulative for Univariate Densities

- If $F(\varepsilon)$ is invertible, we can draw μ from Uniform(0,1) and then set
$$\varepsilon = F^{-1}(\mu).$$
- E.g. extreme value distribution:
$F(\varepsilon) = e^{-e^{-\varepsilon}} \implies \varepsilon = -\ln\big[-\ln(\mu)\big]$.
- This is super useful and can let us draw from all univariate distributions, including truncated distributions.
- **Limitation:** only works univariately (for multi-dim, F^{-1} isn't unique).

Truncated Univariate Densities

- Suppose a variable ε is truncated: $a \leq \varepsilon \leq b$.
- Then $F_{truncated}(\varepsilon) = \frac{F(\varepsilon) - F(a)}{F(b) - F(a)}$ for $a \leq \varepsilon \leq b$, else 0;
- We can draw from the truncated distribution using the below procedure:
  - Draw μ from Uniform(0,1).
  - Define $\mu^* = (1 - \mu)\, F(a) \; + \; \mu\, F(b)$
  - Then $\varepsilon = F^{-1}(\mu^*)$ is in $[a, b]$.

Choleski Transformation for Multivariate Normal

- For $\varepsilon \sim \mathcal{N}(\mathbf{b}, \Sigma)$ in K dims:
  1. Compute L s.t. $L\,L^{\top} = \Sigma$ (Choleski factor).
  2. Draw η from $\mathcal{N}(0, I_K)$ (i.e. K independent standard normals).
  3. Set $\varepsilon = \mathbf{b} + L\,\eta$.
- ε now has covariance Σ and mean $\mathbf{b}$.

Accept–Reject for Truncated Multivariate Densities

- If ε has a complicated truncated region, can use *accept–reject*:
  1. Draw from untruncated $g(\varepsilon)$.
  2. If ε is in desired region, accept (keep it). Else reject, draw again.
- Problem: if truncation region is small, many rejections occur.
- Alternatively, might specify specialized approaches (Gibbs, etc.) if feasible.

Importance Sampling

- Goal: draw from density $f(\varepsilon)$, but easier to draw from $g(\varepsilon)$.
- Weighted draws:

$$\text{weight} = \frac{f(\varepsilon)}{g(\varepsilon)}.$$

- E.g. for integrating $\int t(\varepsilon) f(\varepsilon) d\varepsilon$, simulate as:

$$\frac{1}{R} \sum_{r=1}^{R} t(\varepsilon^r) \, \frac{f(\varepsilon^r)}{g(\varepsilon^r)}, \quad \varepsilon^r \text{ drawn from } g(\cdot).$$

- Must have support of g cover that of f, and $f(\varepsilon)/g(\varepsilon)$ must be finite.

Gibbs Sampling

- Useful when it's easy to draw from $f(\varepsilon_i \mid \varepsilon_{-i})$ but not from $f(\varepsilon_1, \ldots, \varepsilon_K)$ jointly.
- For 2D example with random vector $(\varepsilon_1, \varepsilon_2)$:
  1. Start with some guess $\varepsilon_1^{(0)}$.
  2. Draw $\varepsilon_2^{(0)} \sim f(\varepsilon_2 \mid \varepsilon_1^{(0)})$.
  3. Then $\varepsilon_1^{(1)} \sim f(\varepsilon_1 \mid \varepsilon_2^{(0)})$, etc.
- After enough iterations, $(\varepsilon_1, \varepsilon_2)$ behave like draws from the joint distribution.
- Especially popular in Bayesian sampling contexts.

Metropolis–Hastings (MH) Algorithm

- General-purpose method: can (in principle) handle *any* density $f(\varepsilon)$, even if normalizing constant unknown.
- **Algorithm:**
  1. Start with $\varepsilon^{(0)}$.
  2. Propose $\tilde{\varepsilon} = \varepsilon^{(t)} + \eta$ (draw η from some $g(\eta)$).
  3. Accept with probability $\min\left(1, \frac{f(\tilde{\varepsilon})}{f(\varepsilon^{(t)})}\right)$; otherwise, stay at $\varepsilon^{(t)}$.
- Over many iterations, ε sequence approximates f.
- Implementation details (choice of g, burn-in, etc.) can be intricate.

Motivation for Variance Reduction

- Purely random draws can clump, cause suboptimal coverage of the domain.
- **Goal:** reduce the variance of the simulator for a given number of draws.
- Methods:
  - Antithetics
  - Systematic sampling
  - Halton sequences (and other low-discrepancy sets)
  - Randomized Haltons
- Typically yields better integration than standard pseudo-random draws.

Antithetics

- Take a draw ε^r from $f(\cdot)$, then create $\varepsilon^{r'}$ by mirroring (e.g. for a symmetric dist around 0, $\varepsilon^{r'} = -\varepsilon^r$).
- This induces strong negative correlation, reducing variance in the average.
- Also can do multi-dimensional sign flipping:

$$\langle \varepsilon_a^r, \varepsilon_b^r \rangle \longrightarrow \langle -\varepsilon_a^r, \varepsilon_b^r \rangle, \langle \varepsilon_a^r, -\varepsilon_b^r \rangle, \ldots$$

- Typically improves coverage, especially with fewer draws.

Systematic Sampling

- Divide domain into grid segments; sample exactly one point from each segment.
- E.g. uniform(0,1):
  - If 4 draws needed, we do $\mu, 0.25 + \mu, 0.5 + \mu, 0.75 + \mu$, with $\mu \in [0, 0.25]$.
- Extends to multi-dim by dividing each dimension into intervals.
- Often combined with sign flipping or transformations for normal, etc.
- Balances coverage vs. randomization (need enough random draws for asymptotics).

Halton Sequences

- Constructed using prime bases – each prime for one dimension.
- Provide better coverage in fewer draws, plus negative correlation across observations.
- For multi-dimensional, use primes e.g. 2,3,5,... for each dimension.
- Usually discard first few elements to remove initial correlation.

Halton Sequence: an Example from Train (2009)

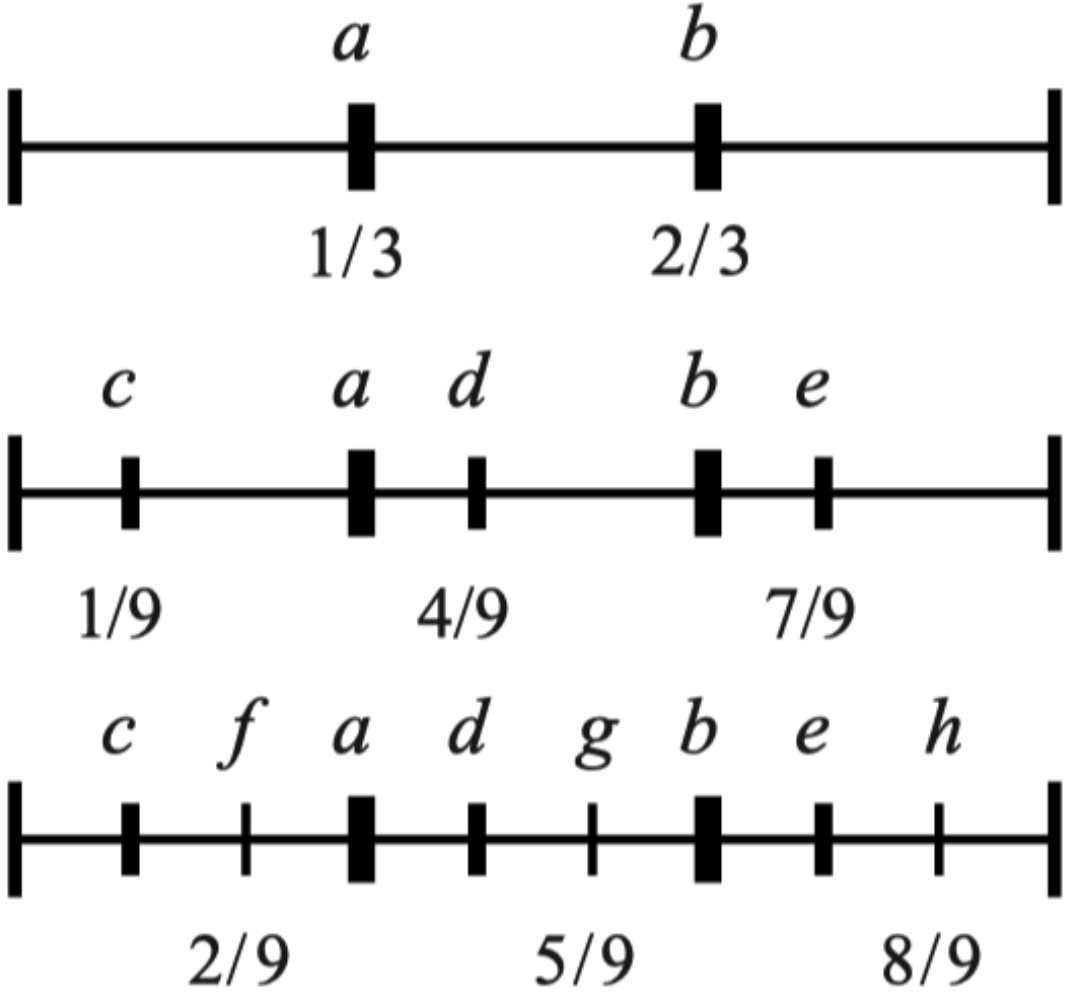

Figure 9.11. Halton sequence for prime 3.

Randomized Halton Sequences

- Halton draws are *systematic* – not truly random.
- If we want pseudo-randomness plus Halton coverage:

$$\text{randomized Halton} \;=\; \text{Halton}(r) \;+\; \mu$$

- Shift each Halton element by a random $\mu \in [0, 1]$, ff the resulting element exceeds 1, subtract 1 from it.
- Retains nice coverage but gains a random offset.

Summary

- Efficient simulation for discrete choice models relies on *good draws*.
- Purely random draws are straightforward but can require many draws to achieve precision.
- **Variance reduction:**
  - *Antithetics*, *systematic sampling*, *Halton*, *etc.* usually yield smaller simulation error for the same number of draws.
- Remember to check the documents for relevant packages when you need random draws.

Estimation

Simulation-based Estimation

Motivation

- Many discrete choice models yield choice probabilities with no closed-form expression.
- We approximate these probabilities by simulation.
- The next question: **how do we estimate parameters from these simulated probabilities?**
- We consider three esimtation methods:
  - Maximum Simulated Likelihood (MSL)
  - Method of Simulated Moments (MSM)
  - Method of Simulated Scores (MSS)
- We compare these methods, derive conditions for consistency, asymptotic normality, and efficiency.

Review: Traditional ML and MOM

Maximum Likelihood (ML):

$$\mathrm{LL}(\theta) = \sum_{n=1}^{N} \ln P_n(\theta),$$

where $P_n(\theta)$ is the exact choice probability for observation n.

Method of Moments (MOM):

$$\sum_{n=1}^{N} \left[d_{nj} - P_{nj}(\theta) \right] z_{nj} = 0,$$

for a set of instruments z_{nj}.

- ML solves $\sum_n \nabla \ln P_n(\theta) = 0$, typically consistent and efficient if model is correct.
- MOM is an alternative – can be less efficient unless ideal instruments are used.

Three Approaches with Simulation

(1) Maximum Simulated Likelihood (MSL)

- Replace $P_n(\theta)$ by simulated $\hat{P}_n(\theta)$ in log-likelihood.
- Then maximize $\sum_n \ln \hat{P}_n(\theta)$ to find $\hat{\theta}$.

(2) Method of Simulated Moments (MSM)

- Replace $P_n(\theta)$ by $\hat{P}_n(\theta)$ in the residuals $[d_{nj} - P_{nj}(\theta)]$, then set sample correlations $= 0$.
- $\sum_{n,j} \left[d_{nj} - \hat{P}_{nj}(\theta) \right] z_{nj} = 0$.

(3) Method of Simulated Scores (MSS)

- Replace the *scores* $s_n(\theta) = \nabla \ln P_n(\theta)$ with simulated versions $\hat{s}_n(\theta)$.
- Solve $\sum_n \hat{s}_n(\theta) = 0$.

Key Results: MSL

MSL:

$$\text{SLL}(\theta) = \sum_{n=1}^{N} \ln \hat{P}_n(\theta).$$

- Even if $\hat{P}_n(\theta)$ is unbiased for $P_n(\theta)$, the *log* introduces bias.
- **If number of draws (R) is fixed**, MSL is *inconsistent*.
- **If R grows with sample size** N, MSL becomes consistent, and:
  - If $R \gg \sqrt{N}$, MSL is asymptotically equivalent to full ML (efficient).
  - If R grows slower than $\sqrt{N}$, MSL is consistent but not asymptotically normal.

Key Results: MSM

- MOM-style approach:

$$\sum_{n,j} \left[d_{nj} - \hat{P}_{nj}(\theta) \right] z_{nj} = 0.$$

- $\hat{P}_{nj}(\theta)$ enters linearly, so no inherent bias from $\ln(\hat{P}_n)$.
- **If R is fixed**, still consistent & asymptotically normal (unlike MSL).
- Efficiency typically $<$ MSL unless *ideal instruments* used (which correspond to partial derivatives of $\ln P_n(\theta)$).

Key Results: MSS

- Instead of $\ln \hat{P}_n(\theta)$, we simulate *scores* $s_n(\theta) = \frac{\partial}{\partial \theta} \ln P_n(\theta)$.
- If these simulated scores are unbiased, no log-bias arises.
- **If R fixed**, consistent & asymptotic normal.
- **If R grows**, asymptotically equivalent to ML with no extra conditions on how fast R grows (unlike MSL which needs $R > \sqrt{N}$).
- Downside: implementing unbiased score simulators (e.g. accept/reject approach) can be *numerically difficult*.

How R vs N Impacts Consistency & Efficiency

MSL (log of simulated probs)

- **R fixed:** Not consistent.
- **R grows slower than** $\sqrt{N}$: consistent, but not normal. **R** grows faster than $\sqrt{N}$: consistent, asymptotically normal, *efficient*.

MSM (simulated residuals linearly)

- **R fixed:** consistent & normal, but typically less efficient if instruments not ideal.
- **R** grows: approaches efficiency of nonsimulated

MSS (simulated scores directly)

- **R fixed:** consistent & normal, if unbiased score simulator used. **R** grows: fully efficient (equivalent to ML) for any growth rate.
- Implementation can be numerically tough.

Conclusion

- Simulation-based estimators must handle the inherent *simulation bias* & *simulation noise*.
- **MSL** typically easier to implement, but *not consistent* if R is fixed – needs $R \to \infty$ suitably fast.
- **MSM** avoids log-bias, thus consistent even for fixed R, but not fully efficient unless using ideal instruments.
- **MSS** with unbiased scores is best theoretically (consistent with fixed R, fully efficient if R grows), but can be *hard to implement* in practice.
- Knowing these trade-offs helps selecting appropriate approach for discrete choice estimations requiring simulation.