23rd Sep 2022

行動モデル夏の学校

# MNL, NL, RL コードの説明

小林里瑳

# Multinomial Logit Model

# MNLモデルのパラメータ推定

```r
### Multinomial Logit model estimation

### Create the data array from the existing dataset
Data <- read.csv("C:/R/Data_Clean_English.csv",header=TRUE)
## Count the number of data rows
hh <- nrow(Data)

## Set the initial values of the parameters (the number in parenthesis corresponds to the number of parameters to be estimated)
b0 <- numeric(5)
```

1. まずはデータを読み込む
2. データ数を数える
3. パラメータ数の初期値(=0)を代入したベクトル を作る

   今回はパラメータ数を5にするので、要素が5 の0ベクトルを作る

# MNLモデルのパラメータ推定

```r
##### Define the log-likelihood function of the logit model#####

fr <- function(x) {
  ### declare the parameters###
  ## Alternative specific constants
  b1 <- x[1]
  b2 <- x[2]
  b3 <- x[3]
  b4 <- x[4]

  ## Travel time to destination
  d1 <- x[5]

  ## declare the log-likelihood variable, set value to 0
  LL = 0

  ### For this choice problem, we consider the following 5 modes???
  ## 鉄道(train)
  ## バス(bus)
  ## 自動車(car)
  ## 自転車(bike)
  ## 徒歩(walk)

  ## calculate the utility function: :introduce the desired explanatory variables in the function
                                  # time           # fare      # constant
  train  <- Data$ModeAvailableTrain*exp(d1*Data$TotalTimeTrain/100  +b1*matrix(1,nrow =hh,ncol=1))
  bus    <- Data$ModeAvailableBus  *exp(d1*Data$TotalTimeBus/100    +b2*matrix(1,nrow =hh,ncol=1))
  car    <- Data$ModeAvailableCar  *exp(d1*Data$TimeCar/100         +b3*matrix(1,nrow =hh,ncol=1))
  bike   <- Data$ModeAvailableBike *exp(d1*Data$TimeBike/100        +b4*matrix(1,nrow =hh,ncol=1))
  walk   <- Data$ModeAvailableWalk *exp(d1*Data$TimeWalk/100                                    )
```

ベクトルxを入れると尤度を出す関数frを作る

今回の効用関数

$$V_{train} = b1 + d1 * \text{鉄道所要時間}/100$$
$$V_{bus} = b2 + d1 * \text{バス所要時間}/100$$
$$V_{car} = b3 + d1 * \text{自動車所要時間}/100$$
$$V_{bike} = b4 + d1 * \text{自転車所要時間}/100$$
$$V_{walk} = \quad +d1 * \text{徒歩所要時間}/100$$

MNLの確率

$$P(i) = \frac{\exp(\mu V_i)}{\sum_{j \in A} \exp(\mu V_i)}$$

# MNLモデルのパラメータ推定

```r
### Calculate the choice probabilities
## calculate the Inclusive Value (the denominator of the choice probabilities equation)
deno <- (car + train + bus + bike + walk)

## Calculate indiviudal choice probabilities
Ptrain <- Data$ModeAvailableTrain*(train / deno)
Pbus   <- Data$ModeAvailableTrain  *(bus   / deno)
Pcar   <- Data$ModeAvailableCar    *(car   / deno)
Pbike  <- Data$ModeAvailableBike   *(bike  / deno)
Pwalk  <- Data$ModeAvailableWalk   *(walk  / deno)

## Avoid problems stemming from choice probabilities becoming zero.
Ptrain <- (Ptrain!=0)*Ptrain + (Ptrain==0)
Pbus   <- (Pbus!=0)*Pbus     + (Pbus==0)
Pcar   <- (Pcar  !=0)*Pcar   + (Pcar  ==0)
Pbike  <- (Pbike  !=0)*Pbike + (Pbike  ==0)
Pwalk  <- (Pwalk!=0)*Pwalk   + (Pwalk  ==0)



## Choice results
Ctrain   <- Data$MainModeENG =="Rail"
Cbus     <- Data$MainModeENG =="Bus"
Ccar     <- Data$MainModeENG =="Car"
Cbike    <- Data$MainModeENG =="Bicycle"
Cwalk    <- Data$MainModeENG =="Walk"

## Calculate the Log-likelihood function
LL <- colSums(Ctrain*log(Ptrain) + Cbus*log(Pbus) +
                Ccar  *log(Pcar)   + Cbike  *log(Pbike) +Cwalk *log(Pwalk))
```

$$P(i) = \frac{\exp(\mu V_i)}{\sum_{j \in A} \exp(\mu V_i)}$$

$$P(i) = \frac{\exp(\mu V_i)}{\sum_{j \in A} \exp(\mu V_i)}$$

選択確率が0だと尤度も0になる

すると$\ln 0 = -\infty$ になってしまいエラーが出る

選択確率0の時，1にする処理をする

$$LL(\boldsymbol{\theta}) = \sum_{n=1}^{N} \sum_{i} y_{in} \log P_n(i|\boldsymbol{\theta})$$

# MNLモデルのパラメータ推定

```
##Parameter optimization
res <- optim(b0,fr,gr=NULL, method = "Nelder-Mead", hessian = TRUE, control=list(fnscale=-1))
```

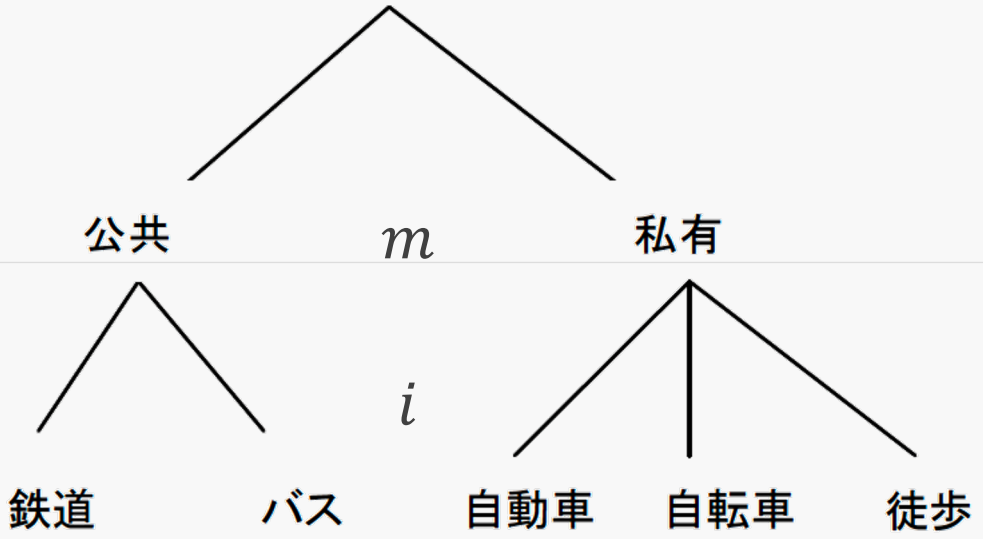
- 最適化関数optim
  - optim(par, fn, gr = NULL, method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN"), lower = -Inf, upper = Inf, control = list(), hessian = FALSE, ...)
  - Par というパラメータを初期値としてfnを最小化するようにパラメータを動かしながら反復して探索する（最大化の場合はfnscale =-1）
  - 探索方法をmethod から選ぶ
    - “Nelder-Mead”法：関数値だけを用い頑健(初期値の選択に敏感)だが遅い
    - “BFGS”法：準ニュートン法．関数値と勾配関数を関数の曲面近似に使う
    - “CG”法：共役勾配法．破綻しやすいがメモリ使用量が少ない
    - “L-BFGS-B”法：**変数の上限・下限を設定した**準ニュートン法
    - “SANN”法：確率的手法である焼きなまし法．関数値だけを用い遅い．

# MNLモデルのパラメータ推定

```r
## Parmeter estimation、Hessian matrix calculation
b   <- res$par
hhh <- res$hessian

## Calculate the t-statistic
tval <- b/sqrt(-diag(solve(hhh)))

## L(0), Log-Likelihood when all parameters are 0
L0 <- fr(b0)
## LL, maximium likelihood
LL <- res$value

##### Output #####
print(res)
## L(0)
print(L0)
## LL
print(LL)
##rho-square
print((L0-LL)/L0)
## adjusted rho-square
print((L0-(LL-length(b)))/L0)
##estimated parameter values
print(b)
## t-statistic
print(tval)
```

- t 値
  - 母集団の平均値の仮説検定に利用
  - |t| >1.96 で5%有意
  - ヘッセ行列の逆行列の対角成分の平方根が分母
- 修正済み尤度比
  - $\frac{L0-(LL-\text{パラメータ数})}{L0}$

# Nested Logit Model



# NLモデルのパラメータ推定

```
fr <- function(x) {
  ### declare the parameters###
  ## Alternative specific constants
  b1 <- x[1]
  b2 <- x[2]
  b3 <- x[3]
  b4 <- x[4]

  ## Travel time to destination
  d1 <- x[5]

  ## Fare
  f1 <- x[6]

  ##Scale parameter
  pa <- x[7]
```

MNL同様にベクトルxを入れると尤度を出す関数frを作る

x[1]~x[4] は定数項（MNLと同じ）

x[5] は時間：上位ネストと下位ネストの組み合わせで決まる効用

x[6] は運賃：上位ネストと下位ネストの組み合わせで決まる効用（今回は考慮しない）

x[7] はスケールパラメータ
NLではスケールパラメータを推定
スケールパラメータが1ならばMNLと同等

# NLモデルのパラメータ推定

- 各選択肢の効用の定義

```
train  <- Data$ModeAvailableTrain*exp(d1*Data$TotalTimeTrain/100  +f1*Data$FareTrain/1000  +b1*matrix(1,nrow =hh,ncol=1))
bus    <- Data$ModeAvailableBus  *exp(d1*Data$TotalTimeBus/100    +f1*Data$FareBus/1000    +b2*matrix(1,nrow =hh,ncol=1))
car    <- Data$ModeAvailableCar  *exp(d1*Data$TimeCar/100                                  +b3*matrix(1,nrow =hh,ncol=1))
bike   <- Data$ModeAvailableBike *exp(d1*Data$TimeBike/100                                 +b4*matrix(1,nrow =hh,ncol=1))
walk   <- Data$ModeAvailableWalk *exp(d1*Data$TimeWalk/100                                                            )
```

- 今回はf1で公共交通の運賃を定義しています

# NLモデルのパラメータ推定

ネストの作成：上位ネストの計算 $\tilde{V}_{in} = \frac{1}{\mu_{in}} \ln \sum_{i \in C_{in}} \exp(\mu_{in} V_i)$ $P(m) = \frac{\exp(\mu_m \tilde{V}_l)}{\sum_l \exp(\mu_m \tilde{V}_l)}$

```
48    ### Construct nests.First, calculate public or private choice probabilityP(LV1)
49    ## setting of log-sum variables
50    logsum.public  <- log( ( (train+bus)!=0) * (train + bus) + ((train+bus)==0))
51    logsum.private <- log( ( (car + bike + walk)!=0) * (car + bike + walk) + ((car + bike + walk)==0))
52
```

公共／私有の選択についてログサム変数を設定

```
53    ##Calculate public or private choice probability with these log-sum variables
54    nume.public   <- (logsum.public != 0)*exp(pa*logsum.public) + (logsum.public == 0)
55    nume.private  <- (logsum.private != 0)*exp(pa*logsum.private) + (logsum.private == 0)
```

ログサム変数を使って公共／私有の選択確率の分子を計算

```
57    deno <- nume.public + nume.private
58    P.public  <- nume.public  / deno
59    P.private <- nume.private / deno
```

分母を計算し(deno)，公共／私有の選択確率を計算

# NLモデルのパラメータ推定

ネストの作成：上位ネスト内の条件付確率

$$P(i \mid m) = \frac{\exp(\mu_m V_i)}{\sum_j \exp(\mu_m V_j)}$$

```
61    ### Next, calculate conditional probability.P(LV2|LV1)
62    ## conditional probability under public choice
63    deno.public     <- train + bus
64    P.train.public  <- train / ((deno.public!=0)*deno.public + (deno.public==0))
65    P.bus.public    <- bus   / ((deno.public!=0)*deno.public + (deno.public==0))
```

公共ー鉄道／バス　というネスト構造
公共を条件とした鉄道, 公共を条件としたバスの選択確率をそれぞれ計算

```
67    ## conditional probability under private choice
68    deno.private    <- car + bike + walk
69    P.car.private   <- car   / ((deno.private!=0)*deno.private + (deno.private==0))
70    P.bike.private  <- bike  / ((deno.private!=0)*deno.private + (deno.private==0))
71    P.walk.private  <- walk  / ((deno.private!=0)*deno.private + (deno.private==0))
```

私有ー車／自転車／徒歩　というネスト構造
私有を条件とした車，自転車，徒歩の選択確率をそれぞれ計算

# NLモデルのパラメータ推定

同時確率の計算

```
### Finally, calculate joint probability. P(LV1, LV2) = P(LV2|LV1)*P(LV1)
P.train <- P.train.public * P.public
P.bus   <- P.bus.public   * P.public
P.car   <- P.car.private  * P.private
P.bike  <- P.bike.private * P.private
P.walk  <- P.walk.private * P.private
```

$$P(i,m) = P(i \mid m)P(m) = \frac{\exp(\mu_m V_i)}{\sum_j \exp(\mu_m V_j)} \frac{\exp(\mu_m \tilde{V}_l)}{\sum_l \exp(\mu_m \tilde{V}_l)}$$

ネストの作り方を含めたNLモデルの詳細については2日目の倉内先生の資料を参照ください

# NLモデルのパラメータ推定

- 最適化関数optim による最尤推定

```
res <- optim(b0,fr, method = "L-BFGS-B",
             lower = c(-Inf,-Inf,-Inf,-Inf,-Inf,-Inf,0),
             upper = c(Inf,Inf,Inf,Inf,Inf,Inf,1),
             hessian = TRUE, control=list(fnscale=-1))
```

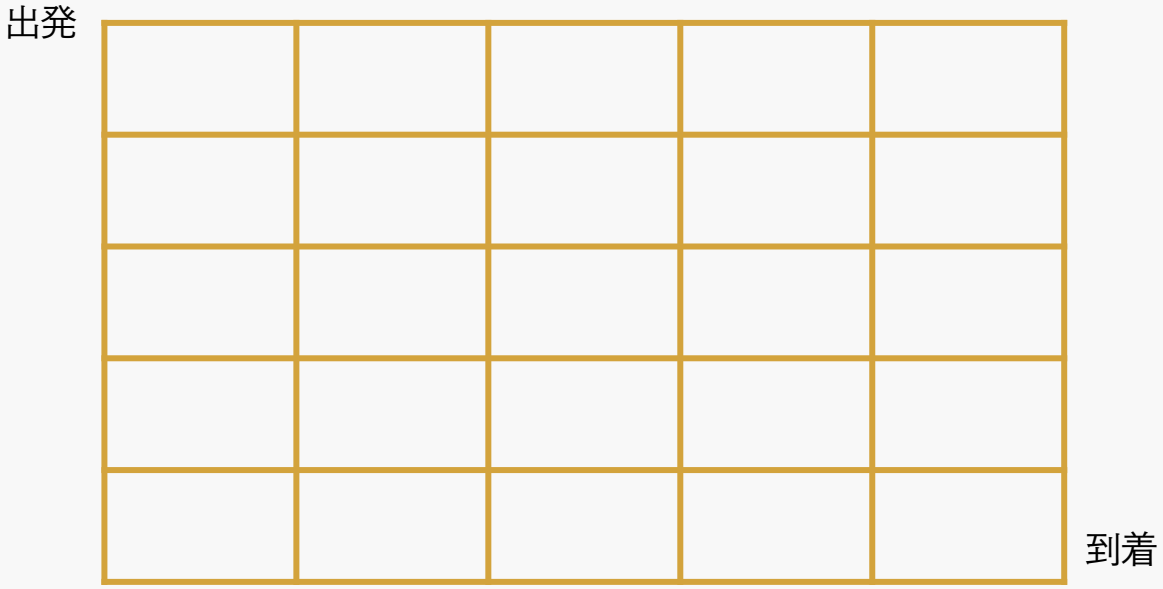
- ここでは7番目のパラメータがスケールパラメータなので，
  変数に下限０上限１を設定して，準ニュートン法でパラメータを探索
- スケールパラメータが(0,1)の範囲に収まらない場合はネスト構造が不適切
- 同様に１に貼り付いている場合は不適切であると同時にMNLと同義

# Recursive Logit Model

Fosgerau, Mogens, Emma Frejinger, and Anders Karlstrom. "A link based network route choice model with unrestricted choice set." *Transportation Research Part B: Methodological* 56 (2013): 70-80.

# はじめに

- 自宅から最寄り駅に行くルートをランダム効用最大化理論で考える
- MNL モデルでは経路の羅列が不可能（組み合わせ爆発を起こすため）



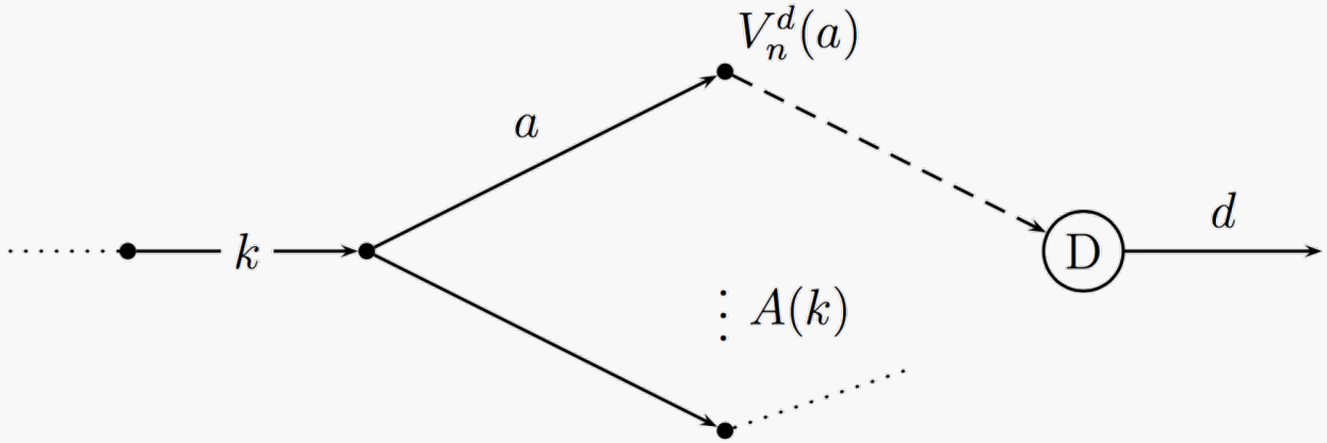

- 解決方法
  - 代表的なルートのみ抽出する：明示的な経路集合を定める
  - 逐次的な行動モデルを考えることで明示的な経路列挙を必要としない

# Recursive Logit モデルとは

- 即時効用，誤差項，目的地までの期待最大効用に従って各ノードで選択
  - 選択そのものはMNL モデル
- **逐次的**かつ**再帰的**な方法で経路が選択される
  - 逐次的：リンク選択の積み重ねが経路全体の選択になると考える
  - 再帰的：次の状態だけでなく将来の状態まで考えた効用の式…Bellman 方程式の利用
  - マルコフ過程に従うと仮定する
- 選択肢は無限集合であり，ループを含む経路を選択する可能性がある
  - 道路ネットワークが与えられている場合，経路集合を設定しなくとも良い
    - アクティビティパス選択などネットワークを限定する必要がある場合はまた別

# 定式化

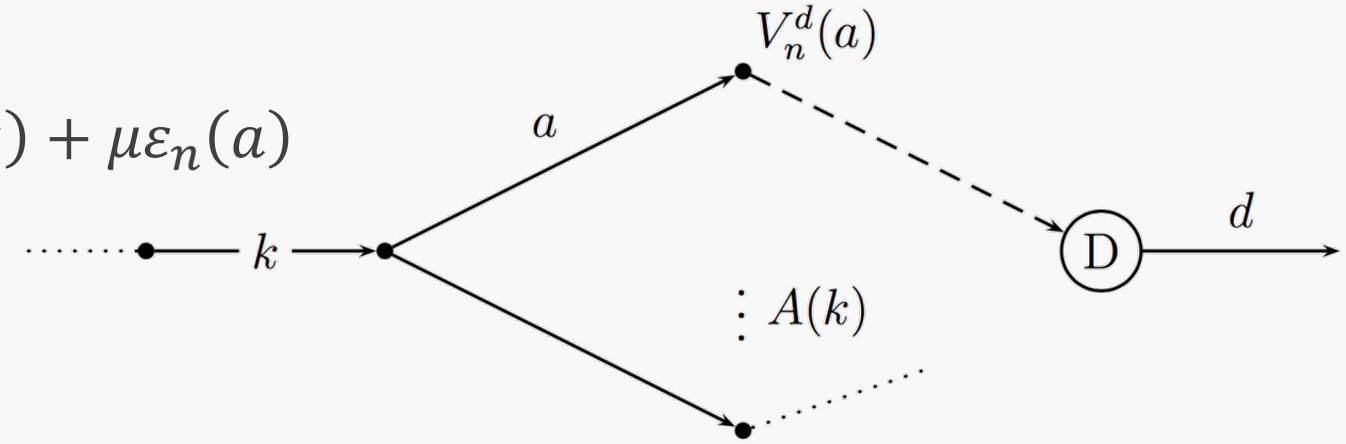
- 有向連結グラフ（ネットワーク）$\mathcal{G} = (\mathcal{S}, \mathcal{A})$ を考える
  - $\mathcal{S}$: ノードの集合，$\mathcal{A}$: リンクの集合
  - リンク $k, a \in \mathcal{A}$
- リンク$k$ の終点から出るリンクの集合 $\mathcal{A}(k)$，　$a \in \mathcal{A}(k) \in \mathcal{A}$




- OからD の経路はリンクの列$[k_0 \ldots k_I]$ で表される $\forall i < I, k_{i+1} \in \mathcal{A}(k_i)$,

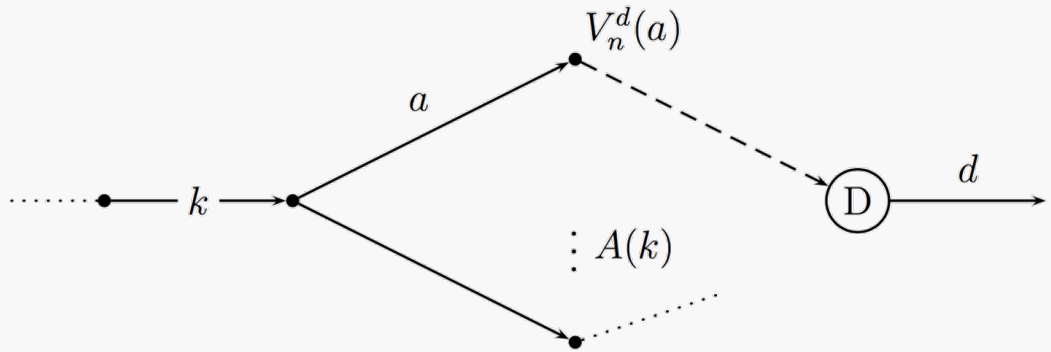
# 定式化

- 効用関数
  - 即時効用　$u_n(a|k) = v_n(a|k) + \mu\varepsilon_n(a)$
  - 期待効用　$V_n^d(a)$
- 終点のダミーリンク $d$
  - 目的地ノード(D) からダミーリンク $d$ を追加し，吸収状態を定義
  - 全リンクの集合を$\tilde{\mathcal{A}}^d = \mathcal{A} \cup d$
  - 目的地ノードを終点とする全てのリンク$k$に対して$v_n(d|k) = 0$ を仮定する
- 主体はマルコフ過程に従って次の状態を選択する
  - 各リンク$k$ でランダム効用$\varepsilon_n(a)$を観測
  - 即時効用と期待効用の和で表される効用を最大化する　$v_n(a|k) + \mu\varepsilon_n(a) + V_n^d(a)$



# 定式化

- 効用関数
  - 即時効用　$u_n(a|k) = v_n(a|k) + \mu\varepsilon_n(a)$
    - 旅行者$n$がリンク$k$にいる状態で選択肢集合$\mathcal{A}(k)$からリンク$a$を選択する
    - 確定項　$v_n(a|k) = v_n(x_{n,a|k}\,;\,\beta)$
    - 誤差項　$\mu\varepsilon_n(a)$：平均値0のガンベル分布を仮定
  - 期待効用　$V_n^d(a)$
    - リンク$a$を選択した場合の目的地Dから伸びるダミーリンク$d$までの期待効用



# 定式化

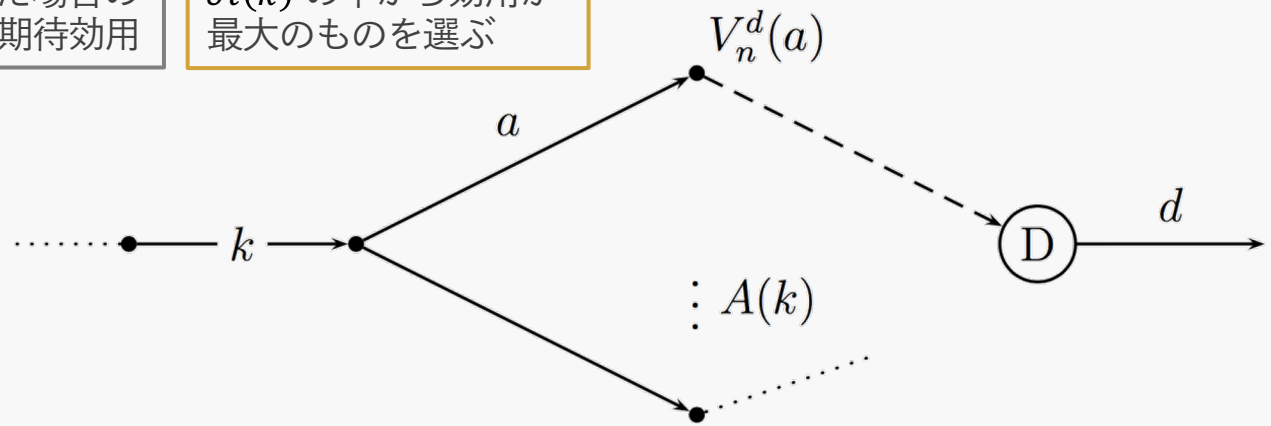
- 効用関数
  - 即時効用　$u_n(a|k) = v_n(a|k) + \mu\varepsilon_n(a)$
  - 期待効用　$V_n^d(a)$
- 価値関数 $V_n^d(k)$: Bellman 方程式と呼ぶ

リンク $k$ において次にリンク$a$ を選んだ場合の効用

$$V_n^d(k) = \mathrm{E}\left[ \max_{a \in \mathcal{A}(k)} \left( v_n(a|k) + V_n^d(a) + \mu\varepsilon_n(a) \right) \right] \quad \forall k \in \mathcal{A}$$

リンク$k$ を選んだ場合の目的地$d$ までの期待効用

$\mathcal{A}(k)$ の中から効用が最大のものを選ぶ



# 選択確率

- リンク $k$ における次のリンク $a$ の選択確率はMNL モデルで表される

$$P_n^d(a|k) = \frac{\exp\left(\frac{1}{\mu}\left(v_n(a|k) + V_n^d(a)\right)\right)}{\sum_{a' \in \mathcal{A}(k)} \exp\left(\frac{1}{\mu}\left(v_n(a'|k) + V_n^d(a')\right)\right)} \quad \text{(RL 3)}$$

- 価値関数は次のように書くことができる

$$V_n^d(k) = \begin{cases} \mu \ln \sum_{a \in \mathcal{A}(k)} \delta(a|k) \exp\left(\frac{1}{\mu}(v_n(a|k) + V_n^d(a))\right) & \forall k \in \mathcal{A} \\ 0 & k = d \end{cases} \quad \text{(RL 4)}$$

$$\delta(a|k) = \begin{cases} 1 & if\ a \in \mathcal{A}(k) \\ 0 & otherwise \end{cases}$$

# 価値関数の求め方

- Bellman 方程式を解く $V_n^d(k) = \mathrm{E}\left[ \max_{a \in \mathcal{A}(k)} \left( v_n(a|k) + V_n^d(a) + \mu\varepsilon_n(a) \right) \right]$
  - 式(RL 4) の両辺に対数をとって変換する

$$e^{\frac{1}{\mu}V^d(k)} = \begin{cases} \sum_{a \in \mathcal{A}(k)} \delta(a|k) e^{\frac{1}{\mu}(v(a|k)+V^d(a))} & \forall k \in \mathcal{A} \\ 1 & k = d \end{cases} \qquad \text{(RL 5)}$$

  - さらにこれを行列式で定義する
  - 以下に示す要素$M$ から構成される行列 $\boldsymbol{M}$ ($|\mathcal{A}| \times |\mathcal{A}|$) を定義する

$$M_{ka} = \begin{cases} \delta(a|k) e^{\frac{1}{\mu}(v(a|k))} & a \in \mathcal{A}(k) \\ 0 & otherwise \end{cases} \qquad \text{(RL 6)}$$

# 価値関数の求め方

- Bellman 方程式を解く $V_n^d(k) = \mathrm{E}\left[ \max_{a \in \mathcal{A}(k)} \left( v_n(a|k) + V_n^d(a) + \mu\varepsilon_n(a) \right) \right]$
  - さらに以下の要素からなるベクトル$\boldsymbol{b}$ ($|\mathcal{A}|\times 1$)と$\boldsymbol{z}$ ($|\mathcal{A}|\times 1$)を定義する

$$b_k = \begin{cases} 1 & k \neq d \\ 0 & k = d \end{cases} \qquad z_k = e^{\frac{1}{\mu}V^d(k)}$$

  - すると式(RL 5) は以下のように簡潔な線形方程式で書き下すことができる $\mathbf{I}$ は単位行列

$$\mathbf{z} = \mathbf{M}\mathbf{z} + \mathbf{b} \Leftrightarrow (\mathbf{I} - \mathbf{M})\mathbf{z} = \mathbf{b} \qquad \text{(RL 7)}$$

  - $\mathbf{z}$ が解を持つ条件は$\mathbf{I} - \mathbf{M}$ が逆行列を持つこと
    - 逆行列を持たない可能性もある．持つかどうかはパスの数や瞬間効用のサイズに依存する

---

$$e^{\frac{1}{\mu}V^d(k)} = \begin{cases} \sum_{a \in \mathcal{A}(k)} \delta(a|k) e^{\frac{1}{\mu}(v(a|k)+V^d(a))} & \forall k \in \mathcal{A} \\ 1 & k = d \end{cases} \qquad \text{(RL 5)}$$

# 遷移確率の求め方

- 選択確率に対応する行列$\boldsymbol{P}$ を作る
  - 目的地に対するリンクの選択確率は出発点に依存しないことに注意
  - 式(3) を$\mathcal{A}$ 内のリンクについて定義した行列$\boldsymbol{P}$ に整理する
  - 状態 $k$ に対応する行は以下の通り． $\odot$は要素ごとの積， $\boldsymbol{M}_k$は$\mathbf{M}$の$k$ 列目を表す

$$\boldsymbol{P}_k = \frac{\boldsymbol{M}_k \odot \boldsymbol{z}^T}{\boldsymbol{M}_k \boldsymbol{z}} \qquad \text{(RL 8)}$$

---

行列 $\boldsymbol{M}$ の要素

$$M_{ka} = \begin{cases} \delta(a|k) e^{\frac{1}{\mu}(v(a|k))} & a \in \mathcal{A}(k) \\ 0 & otherwise \end{cases} \qquad \text{(RL 6)}$$

# 経路選択確率

- 経路選択確率
  - 経路$\sigma = \{k_i\}_{i=0}^{I} = \{k_0, k_1, \ldots, k_I\}$ について考える　$k_0$ は出発リンク，　$k_{\mathrm{I}} = d$　とする
  - 経路$\sigma$ の選択確率は

$$P(\sigma) = \prod_{i=0}^{I-1} P(k_{i+1}|k_i) = \prod_{i=0}^{I-1} \frac{e^{\frac{1}{\mu}\left(v(k_{i+1}|k_i)+V(k_{i+1})\right)}}{\sum_{a\in A(k)} e^{\frac{1}{\mu}\left(v(a|k_i)+V(a)\right)}}$$

$$= \prod_{i=0}^{I-1} e^{\frac{1}{\mu}\left(v(k_{i+1}|k_i)+V(k_{i+1})-V(k_i)\right)}$$

$$= e^{-\frac{1}{\mu}V(k_0)} \prod_{i=0}^{I-1} e^{\frac{1}{\mu}v(k_{i+1}|k_i)} \qquad \text{(RL 9)}$$

# 経路選択確率

- 経路選択確率
  - $v(\sigma) = \sum_{i=0}^{I-1} v(k_{i+1}|k_i)$ と置くと

$$P(\sigma) = e^{-\frac{1}{\mu}V(k_0)} \prod_{i=0}^{I-1} e^{\frac{1}{\mu}v(k_{i+1}|k_i)}$$

$$= \frac{e^{\frac{1}{\mu}v(\sigma)}}{e^{\frac{1}{\mu}V(k_0)}} = \frac{e^{\frac{1}{\mu}v(\sigma)}}{\sum_{\sigma' \in \Omega} e^{\frac{1}{\mu}v(\sigma')}} \quad \text{(RL 10)}$$

  - ここで$\Omega$ は全選択肢の集合
  - 式(10) の分子は経路$\sigma$の確定項，分母は可能な経路の確定項の和であるため，このモデルは無限の選択肢を持つMNL モデルと同等
  - 従ってIIA 特性を持つ

# Discounted Recursive Logit Model (Oyama, Hato, 2017)

- Bellman方程式に時間割引率$\beta$を導入

$$V_n^d(k) = \max_{s_{j+1} \in A(k)} E\left[\sum_{t=j}^{\infty} \beta^{\,t-j} u(s_{j+1} s_j; \theta)\right]$$

$$= E\left[\max_{s_{j+1} \in A(k)} v_n(s_{j+1}|s_j) + \beta V_n^d(s_{j+1}) + \mu\varepsilon_n(s_{j+1})\right]$$

- 時間割引率を導入し，将来価値の低減，未来の不確実性や近視眼的な選択を表現

$$P_n^d(s_{j+1}|s_j) = \frac{\exp\left(\frac{1}{\mu}(v_n(s_{j+1}|s_j) + \beta V_n^d(s_{j+1}))\right)}{\sum_{s_{j+1}' \in A(s_j)} \exp\left(\frac{1}{\mu}(v_n(s'_{j+1}|s_j)\beta V_n^d(s_{j+1}'))\right)}$$

$$e^{\frac{V_n^d(s_j)}{\mu}} = \begin{cases} \sum_{s_{j+1} \in \mathcal{A}(s_j)} \delta(s_{j+1}|s_j) \exp\left(\frac{1}{\mu}(v_n(s_{j+1}|s_j) + \beta V_n^d(s_{j+1}))\right), & s_j \in \mathcal{A} \\ 1, & s_j = d \end{cases}$$

# Discounted Recursive Logit Model (Oyama, Hato, 2017)

- 時間割引率を含んだ価値関数の計算方法

行列 $\boldsymbol{M}$ の要素

$$M_{s_j s_{j+1}} = \begin{cases} \delta(s_{j+1}|s_j) e^{\frac{1}{\mu}(v(s_{j+1}|s_j))} & \forall s_j \in \mathcal{A} \\ 0 & otherwise \end{cases}$$

ベクトル $\boldsymbol{z}$ の要素

$$z_{s_j} = e^{\frac{1}{\mu}V^d(s_j)}$$

RLモデルと異なり，価値関数の解は非線形方程式系に従うことになる

$$z_{s_j} = \begin{cases} \sum_{s_{j+1} \in \mathcal{A}} M_{s_j s_{j+1}} (z_{s_{j+1}})^{\beta}, & \forall s_j \in \mathcal{A} \\ 1, & s_j = d \end{cases} \qquad \text{(dRL 1)}$$

式(dRL 1)を行列表記で書くと以下の通り

$$\mathbf{z} = \mathbf{M}\mathbf{X}(\mathbf{z}) + \mathbf{b} \qquad \text{(dRL 2)}$$

# Discounted Recursive Logit Model (Oyama, Hato, 2017)

- 時間割引率を含んだ価値関数の計算方法

行列 $\mathbf{M}$ の要素

$$M_{s_j s_{j+1}} = \begin{cases} \delta(s_{j+1}|s_j) e^{\frac{1}{\mu}(v(s_{j+1}|s_j))} & \forall s_j \in \mathcal{A} \\ 0 & otherwise \end{cases}$$

ベクトル $\mathbf{z}$ の要素

$$z_{s_j} = e^{\frac{1}{\mu} V^d(s_j)}$$

---

$$\mathbf{z} = \mathbf{M}\mathbf{X}(\mathbf{z}) + \mathbf{b} \qquad \text{(dRL 2)}$$

$\mathbf{X}(\mathbf{z})$ は$|\mathcal{A}|\times|\mathcal{A}|$の行列で要素$X(z)_{s_j} = (z_{s_j})^{\beta}$

式(dRL 2)を解くには価値関数が不動点に達するまで反復計算する必要がある

# Discounted Recursive Logit Model (Oyama, Hato, 2017)

- 時間割引率を含んだ価値関数の計算方法→不動点までの反復計算

$$\mathbf{z} = \mathbf{M}\mathbf{X}(\mathbf{z}) + \mathbf{b} \qquad \text{(dRL 2)}$$

$\mathbf{X}(\mathbf{z})$ は$|\mathcal{A}|\times|\mathcal{A}|$の行列で要素$X(z)_{s_j} = (z_{s_j})^{\beta}$

反復計算の手順（Mai, Fosgerau, and Frejinger, 2015 ）

1. ベクトル $\mathbf{z}^{(0)}$ として初期化する
2. $\mathbf{z}^{(1)} = \mathbf{M}\mathbf{X}\left(\mathbf{z}^{(0)}\right) + \mathbf{b}$ として更新する
3. $\left|\mathbf{z}^{(n+1)} - \mathbf{z}^{(n)}\right| < \gamma$ となるまで2を繰り返す

# dRL モデルの推定：コードの概要

データのインプット

価値関数（V, z）の初期値の設定

対数尤度関数の定義

- 選択確率の計算　$P_k = \frac{\boldsymbol{M}_k \odot \boldsymbol{z}^T}{\boldsymbol{M}_k \boldsymbol{z}}$
- 対数尤度の計算

価値関数の定義

- 時間割引率を導入しているため価値関数z を閾値まで求解

最尤推定

価値関数の計算

収束するまで繰り返し

# dRL モデルの推定：データのインプット

- インプットデータとして最低限必要なもの
  - 経路データ

    … トリップID／発リンク／着リンク／終点リンクの列からなる経路の集合

| TripID | Link | NextLink | Absorption |
|---|---|---|---|
| 23010 | 10035 | 114 | 368 |
| 23010 | 114 | 683 | 368 |
| 23010 | 683 | 684 | 368 |

  - ネットワークデータ

    … リンクの接続関係を表す

| k | a |
|---|---|
| 1 | 10001 |
| 2 | 6 |

  - リンクデータ

    … リンクid ごとの説明変数を格納　モデルの説明変数となる

| StateID | StartNodeID | EndNodeID | Length | type | cross |
|---|---|---|---|---|---|
| 1 | 61 | 62 | 73.43643748 | 14 | 0 |
| 2 | 61 | 63 | 14.60142731 | 14 | 0 |
| 3 | 61 | 66 | 18.12913213 | 16 | 0 |
| 4 | 66 | 67 | 22.22027616 | 16 | 0 |

# dRL モデルの推定：対数尤度関数の定義

配布コード：RL_Estimation_Assignment_Node.R を参照ください！

```
fr <- function(x) {

  LL <- 0
  for(d in 1:D){
    dn <- (1:L)[linkid == ddata[d]]

    dd <- ddata[d]
    Id <- Ilist[[d]]

    # 価値関数
    z <- array(exp(Theta* V[,d]), dim = c(L, 1))
    ZD <-  array(rep(z), dim = c(L, L))
    ZD <- t(ZD)

    # 瞬間効用
    M <- Id * Mset(x)
    M[, dn] <- exp(0)
```

←終点リンクd ごとに計算している

← $z_k = e^{\frac{1}{\mu}V^d(k)}$ を計算

```
    # 選択確率を計算
    Mz <- (M %*% z != 0) * (M %*% z) + (M %*% z == 0)
    MZ <- array(rep(Mz), dim = c(L, L))
    p <- (M * ZD) / MZ
    p <- (p == 0) * 1 + (p != 0) * p
```

選択確率の計算

← $P_k = \frac{\boldsymbol{M}_k \odot \boldsymbol{z}^T}{\boldsymbol{M}_k \boldsymbol{z}}$ を計算

# dRL モデルの推定：価値関数の計算（newV）

```
newV <- function(x) {

  z <- matrix(1, nrow = L, ncol = D) #exp(Vd)
  V <- matrix(0, nrow = L, ncol = D)
  i <- 0

  for (d in 1:D){
    kd <- (1:L)[linkid == ddata[d]]
    z[kd, d] <- 1

    M <- matrix(0, nrow = L, ncol = L)
    B <- matrix(0, nrow = L, ncol = 1)
    B[kd, 1] <- 1

    Id <- Ilist[[d]]

    # 瞬間効用Mを更新
    for (k in 1:L){
      for (a in 1:L){
        Ika <- Id[k, a]
        if(Ika == 1){
          if(a == kd){
            M[k, a] <- 1
          }else{
            M[k, a] <- Mset(x)[k, a]
          }
        }
      }
    }
```

関数名：newV の中身

←終点リンクd ごとに計算している

← **M**と**b**を定義

← 即時効用 **M** を更新

# dRL モデルの推定：価値関数の計算（newV）

$$\mathbf{z} = \mathbf{M}\mathbf{X}(\mathbf{z}) + \mathbf{b} \qquad \text{(dRL 2)}$$

$\mathbf{X}(\mathbf{z})$ は$|\mathcal{A}|\times|\mathcal{A}|$の行列で要素$X(z)_{s_j} = (z_{s_j})^{\beta}$

```
#価値関数zの求解
while(dL >= 0.01){
  zii <- zi
  Xz <- zii^Theta
  zi <- M %*% Xz + B
  dL <- sum(abs(zii - zi))
}
```

反復計算の手順（Mai, Fosgerau, and Frejinger, 2015 ）

1. ベクトル $\mathbf{z}^{(0)}$ として初期化する
2. $\mathbf{z}^{(1)} = \mathbf{M}\mathbf{X}\left(\mathbf{z}^{(0)}\right) + \mathbf{b}$ として更新する
3. $\left|\mathbf{z}^{(n+1)} - \mathbf{z}^{(n)}\right| < \gamma$ となるまで2を繰り返す

# dRL モデルの推定：最尤推定と価値関数の更新

```
while(dL>=0.01){
  n <- n + 1

  b0 <- b
  z <- z0
  V <- V0

  res <- optim(b0, fr, method = "Nelder-Mead", hessian = TRUE, control=list(fnscale=-1))

  b <- res$par
  cat("b  =")
  print(b)
  cat("lnL=")
  print(res$value)

  #updating value funciton
  z0 <- newV(b)

  #convergence condition
  zz <- (z == 0) * 1 + (z != 0) * z
  V <- log(zz)
  zz0 <- (z0 == 0) * 1 + (z0 != 0) * z0
  V0 <- log(zz0)
  dL <- sum(abs(b - b0)) # NFXP アルゴリズム

  cat("dL =")
  print(dL)
}
```

←最尤推定の実施

←パラメータの更新

←価値関数の求解
（newVの実行）

←価値関数の更新

# RLモデル, DRL モデルを用いた研究の例

- 経路選択モデル（RLモデル）
  - Fosgerau, Mogens, Emma Frejinger, and Anders Karlstrom. **"A link based network route choice model with unrestricted choice set."** *Transportation Research Part B: Methodological* 56 (2013): 70-80.
- 災害時におけるドライバーの経路選択モデル（DRLモデル）
  - Oyama, Yuki, and Eiji Hato. **"A discounted recursive logit model for dynamic gridlock network analysis."** Transportation Research Part C: Emerging Technologies 85 (2017): 509-527.
- Activity-travel スケジューリングの時空間経路選択モデル（mixed RLモデル）
  - Zimmermann, Maëlle, et al. **"Capturing correlation with a mixed recursive logit model for activity-travel scheduling."** *Transportation Research Part C: Emerging Technologies* 93 (2018): 273-291.
- 歩行者環境におけるアクセシビリティ分析のための歩行者経路選択モデル（RLモデル）
  - Zheng LIANG*, Ka Fai NG, Hong K. LO **“A Link-Based Approach For Measuring Walking Accessibility”,** 2022