スタートアップゼミ#3

離散選択モデル入門

M1 飯塚卓哉

シラバス

4/11　行動データ分析基礎

4/18　最短経路探索

4/25　離散選択モデル

GW休み

5/9　都市形成史分析1

5/16　課題発表①

5/23　均衡配分

5/30　アクティビティモデル

6/6　課題発表②

6/13　都市経済モデル

6/20　都市形成史分析2

6/27　課題発表③

目次

- 離散選択モデル基礎
- Logit Model
- Logit Modelのパラメータ推定

- R導入
- MNLモデルのパラメータ推定

離散選択モデル基礎

■ 行動モデルとは？

さまざまな**人の選択行動＝意思決定**を表現するモデル

ex）離散選択モデル，アクティビティモデル

意思決定者

個人，家計，組織...

選択肢

$$Choice\ set \in \boldsymbol{Univarsal\ set}$$

意思決定の際に考慮に入れる選択肢

意思決定者の環境によって定義されている

説明変数

選択肢の望ましさを表す特徴量ベクトル

ex）旅行時間，コスト...→ LOS(Level of Service)

意思決定ルール

意思決定を行うメカニズム

ex）優位性，規則，効用

ランダム効用最大化

- 「効用」という一つの尺度を表す関数を考える
- 意思決定するときは，トレードオフの概念に基づく
- 合理的に行動する
  - 効用の高い選択肢を選択する
- **ランダム効用**＝確率的に変化する効用を考える
  - Why? → 観察される行動は以下の要素を含んでいる
    1. 観測不可能な説明変数（好みなど）
    2. 測定時の誤差，不完全な情報

    etc...

ランダム効用最大化

効用関数

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

確定項　　誤差項

$$\begin{aligned} P(i|C_n) &= \Pr(U_{in} \geq U_{jn}, \forall j \in C_n) \\ &= \Pr(V_{in} + \varepsilon_{in} \geq V_{jn} + \varepsilon_{jn}, \forall j \in C_n) \\ &= \Pr\left(\varepsilon_{jn} - \varepsilon_{in} \leq V_{in} - V_{jn}, \forall j \in C_n\right) \\ &= \Pr(\varepsilon_n \leq V_n, \forall j \in C_n) \end{aligned}$$

確定項の値だけで確率が決まる!

C_n：個人nの選択肢

ランダム効用最大化

効用関数

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

基本は線形関数

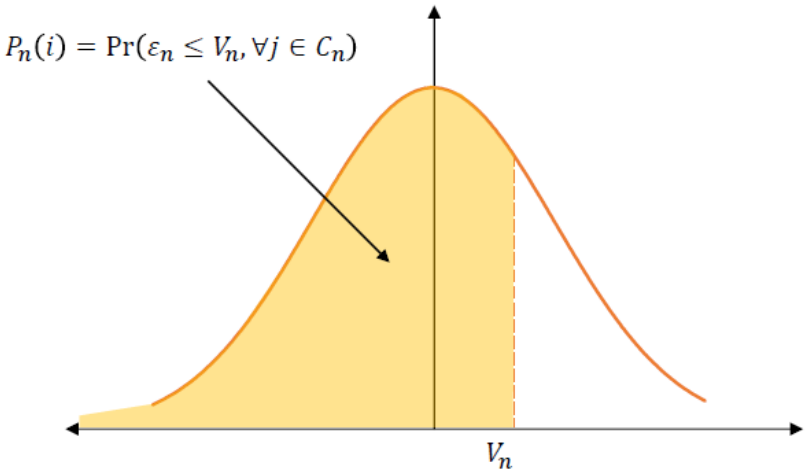
$$V_{in} = \beta_1 x_{in1} + \beta_2 x_{in2} + \cdots + \beta_K x_{inK}$$

x_{ink}：説明変数
β_k：パラメータ（どのくらい効いてるか）

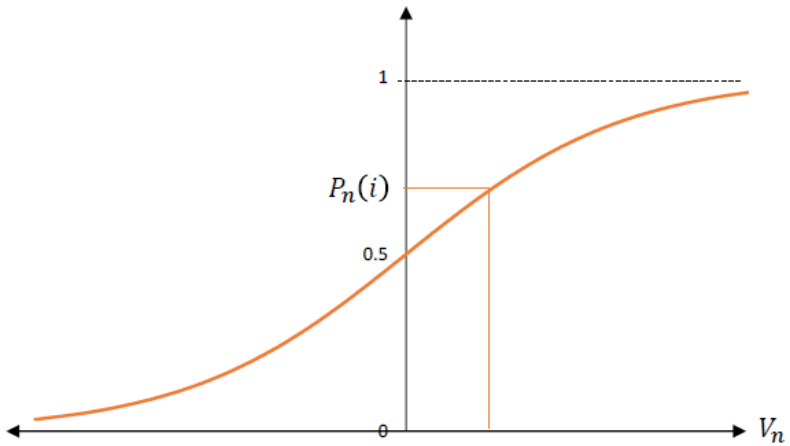
ある確率分布に従う確率変数
確率分布によって異なるモデルとなる

- 正規分布 → Probit model
- **ガンベル分布 → Logit model**

二項選択モデル：Probit Model



正規分布の確率密度関数



選択肢 i の選択確率

- 選択肢 i の選択確率

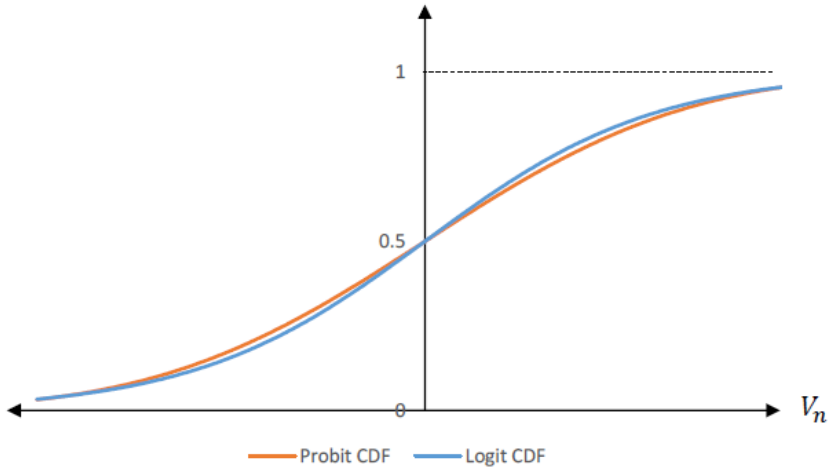
$$P_n(i) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{(V_n)/\sigma} exp\left[-\frac{1}{2}\left(\frac{\varepsilon}{\sigma}\right)^2\right] d\varepsilon = \Phi\left(\frac{(V_n)}{\sigma}\right)$$

選択確率式はcloced-formとならない

→ 選択確率式を導くのが大変

二項選択モデル：Logit Model

- 誤差項ε_{in}, ε_{jn}にi.i.d. ガンベル分布を仮定
- 確率式はProbit Modelの良い近似となる
- さらに確率式はclosed-formで書けるため，便利



- 選択肢 i の選択確率式

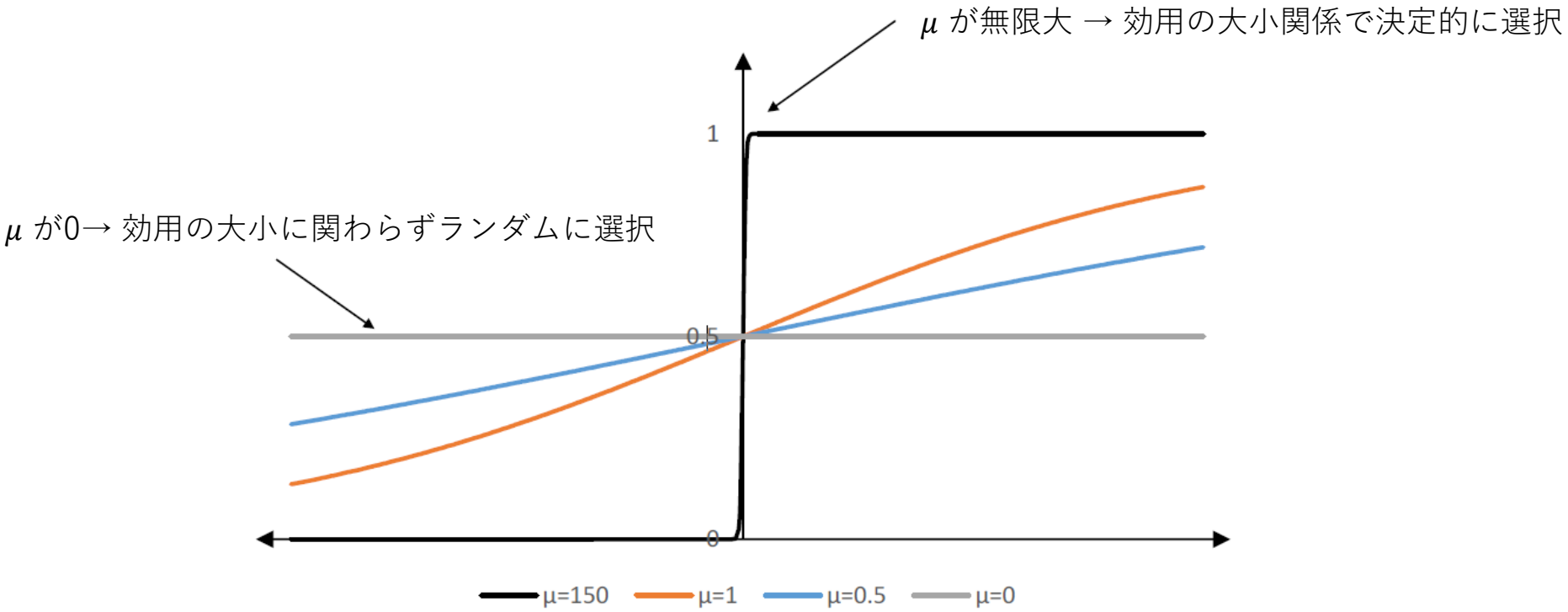
$$P_n(i) = \frac{\exp(\mu V_{in})}{\exp(\mu V_{in}) + \exp(\mu V_{jn})} = \frac{1}{1 + \exp\left(-\mu\left(V_{in} - V_{jn}\right)\right)}$$

where μ is a scale parameter

↑これの導出は簡単

スケールパラメータとは？

μの値による選択確率



μ の大きさ＝誤差項がどれだけ効くか

Logit Model

説明変数　V を構成する変数

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

確定項の値だけで確率が決まる!

選択肢固有定数項

- J個の選択肢があるとき最大J-1個の選択肢につけることができ，一つは0に基準化する
- 基準化した選択肢に対する他の説明変数以外の要素の平均的な影響を反映する

パラメータ（どれくらい効くか）　説明変数（Level Of Service）

$$V_{car} = ASC_{car} + \beta_{time} TravelTime + \beta_{cost}\left(\frac{Cost}{Income}\right) + \gamma_{worker} Worker$$

$$V_{car} = \quad 0 \quad + \beta_{time} TravelTime + \beta_{cost}\left(\frac{Cost}{Income}\right) + 0$$

個人属性

- 単独で説明変数に用いる場合，選択肢の内一つは0に基準化する
- 他の説明変数と関連しているとき，基準化は不要

Multinomial Logit Model

- 選択肢数が3個以上
- 誤差項の仮定
  - 各選択肢の誤差項の確率は独立で同一な分布に従う（I.I.D） (Independently and identically distributed)
  - ロケーションパラメータη（普通0），スケールパラメータ$\mu > 0$（普通1）のガンベル分布を仮定

選択確率

$$P(i) = Pr\left[V_{in} + \varepsilon_{in} \geq \max_{j \in C_n, j \neq i} (V_{jn} + \varepsilon_{jn})\right]$$

$$P(i) = \frac{1}{1 + \exp\left(-\mu(V_n^* - V_{in})\right)} = \frac{\exp(\mu V_{in})}{\sum_{j \in C} \exp\left(\mu V_{jn}\right)}$$

I.I.A特性（Independence of Irrelevant Alternatives）

説明変数に関係なく，選択確率が決まるとする

0.50　　0.50

I.I.A特性（Independence of Irrelevant Alternatives）

説明変数に関係なく，選択確率が決まるとする

0.50　　0.50

0.33　　0.33　　0.33

I.I.A特性（Independence of Irrelevant Alternatives）

説明変数に関係なく，選択確率が決まるとする

0.50　　0.50

0.33　　0.33　　0.33

現実にはこの方が妥当

0.50　　0.25　　0.25

相関のある選択肢がある場合に選択確率の妥当性を失う

→ I.I.A特性を緩和するために…

ex) NLモデル，CNLモデル，Path Size Logitモデル　etc…

ロジットモデルの強みと弱み

- 観測されうる行動原理（コストや時間，個人属性）は表現できるが，それ以外は表現できない（好み，その日の気分）

- I.I.A特性がある→モデルの拡張

- 観測されない行動原理の影響が小さくなるほどのサンプル数，繰り返し選択のデータがあれば，行動原理を良く表現できるモデル

Logit Modelのパラメータ推定

最尤推定法

あるサンプルの行動データ → その母集団の行動原理を知る

＝行動モデルのパラメータを求める

尤度：ある前提条件に従って結果が出現する場合に、逆に観察結果からみて前提条件が「何々であった」と推測する尤もらしさを表す数値

尤度関数（一般形）

$$L_n(\beta|y_n,x_n)=\prod_{n=1}^{N} f(y_n|\beta,x_n)$$

前提条件がβのときにy_nという結果が観察されるのはこれくらい尤もらしいですよという値

対数尤度関数の最大化

$$Max\ LL=\max_{\hat{\beta}_n}\sum_{n=1}^{N} log f(y_n|\beta,x_n)$$

Logit Modelのパラメータ推定

最尤推定法

一般的には，尤度関数は個人 n が観察された選択肢を選択する確率として定義される

$$L_n(\beta_1, \beta_2, \dots \beta_K) = \prod_{n=1}^{N} \prod_{i} P_n(i)^{y_{in}}$$

y_{in}は選択肢iが選択されたとき1，それ以外で0

つまり，最大化したい対数尤度関数は

$$LL_n(\beta_1, \beta_2, \dots \beta_K) = \sum_{n=1}^{N} \sum_{i} y_{in} log P_n(i)$$

Logit Modelのパラメータ推定

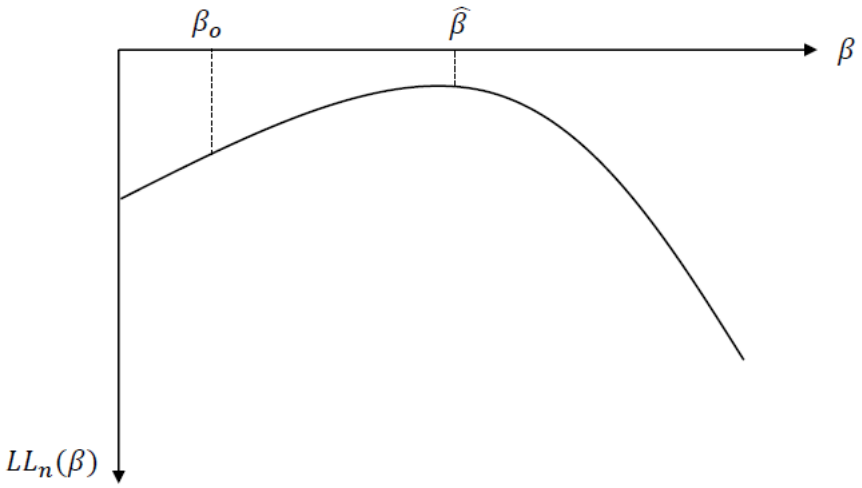
最尤推定法

βを動かして，最大尤度を与える$\hat{\beta}$を探せばいい

→ 各パラメータβ_kについての一階の偏微分が0（第一条件）

$$\frac{\partial LL}{\partial \widehat{\beta_k}} = 0, for\ k = 1, \dots, K$$

最大尤度では，すべてのパラメータについて一階の偏微分が0



尤度関数が大域的に凸なら，解$\hat{\beta}$は一意な解となる．

この時，尤度関数のHessianは半負定値行列．

＝すべての固有値が0か負

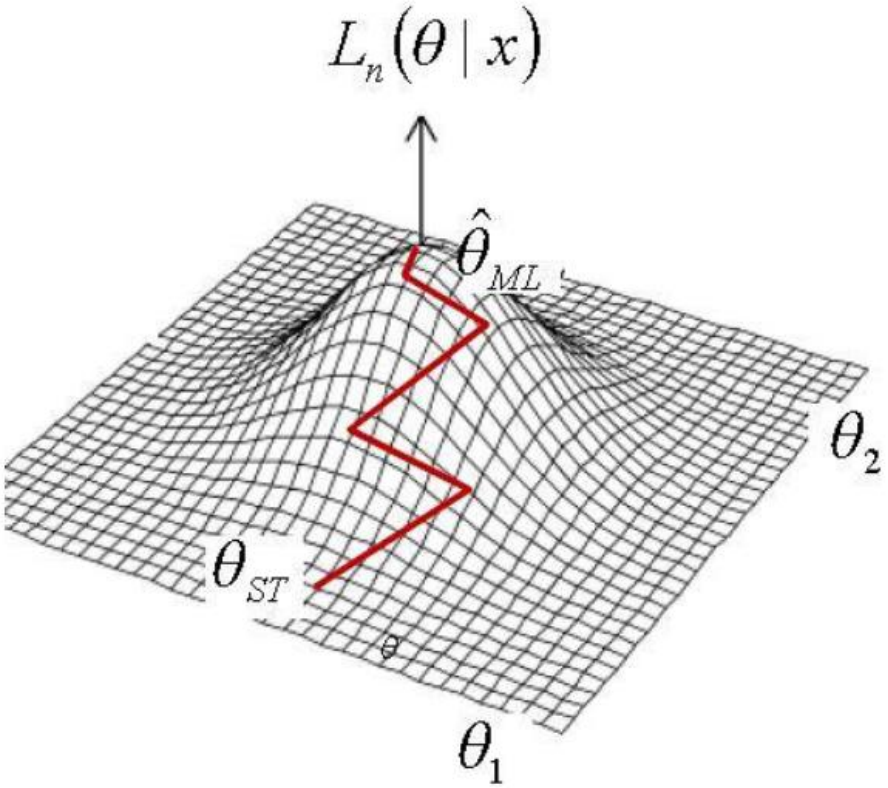
Logit Modelのパラメータ推定

最尤推定法

対数尤度関数の最大化問題 ＝ 連続最適化問題

→ 反復的な解法が用いられる

- 直線探索法
- 信頼領域法



詳細は略

過去のゼミ
http://bin.t.u-tokyo.ac.jp/summercamp2018/

MNLのパラメータ推定(R)

R導入

- Rは統計解析に特化したプログラミング言語
- 無償＋オープンソースのソフトウェア
- Tipsがネットに豊富

R-Tips

http://cse.naro.affrc.go.jp/takezawa/r-tips/r.html

Rの基本操作はここにだいたい書いてある

RjpWiki

http://www.okadajp.org/RWiki/

Rについてのwiki．みんなで編集できる

特徴

- ベクトル，行列演算が簡単
- そこそこ速い（らしい）
- 可視化もできる
- 初心者に優しい（でも他人の書いたコードが読みずらい）

R導入

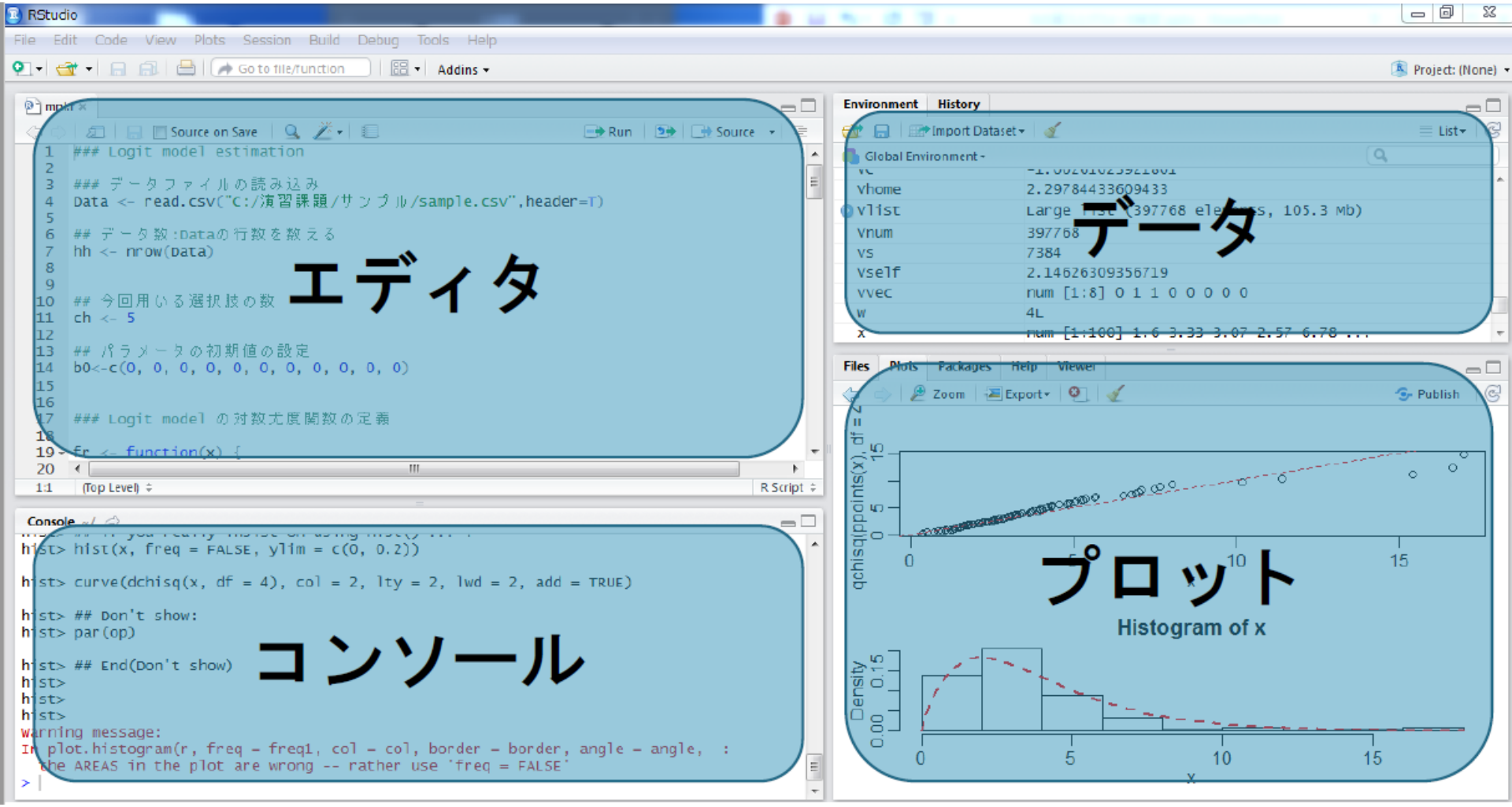
インストールするもの

■ R本体
http://cse.naro.affrc.go.jp/takezawa/r-tips/r/01.html
https://qiita.com/FukuharaYohei/items/8e0ddd0af11132031355
windowsじゃない人も調べれば出てきます

■ R studio
Rの総合開発環境（IDE）．他にもあるがこれがおすすめ.
http://memorandum2015.sakura.ne.jp/index_rstudio.html
https://qiita.com/FukuharaYohei/items/3468bd2a6b2f07b8963e

使ってみる

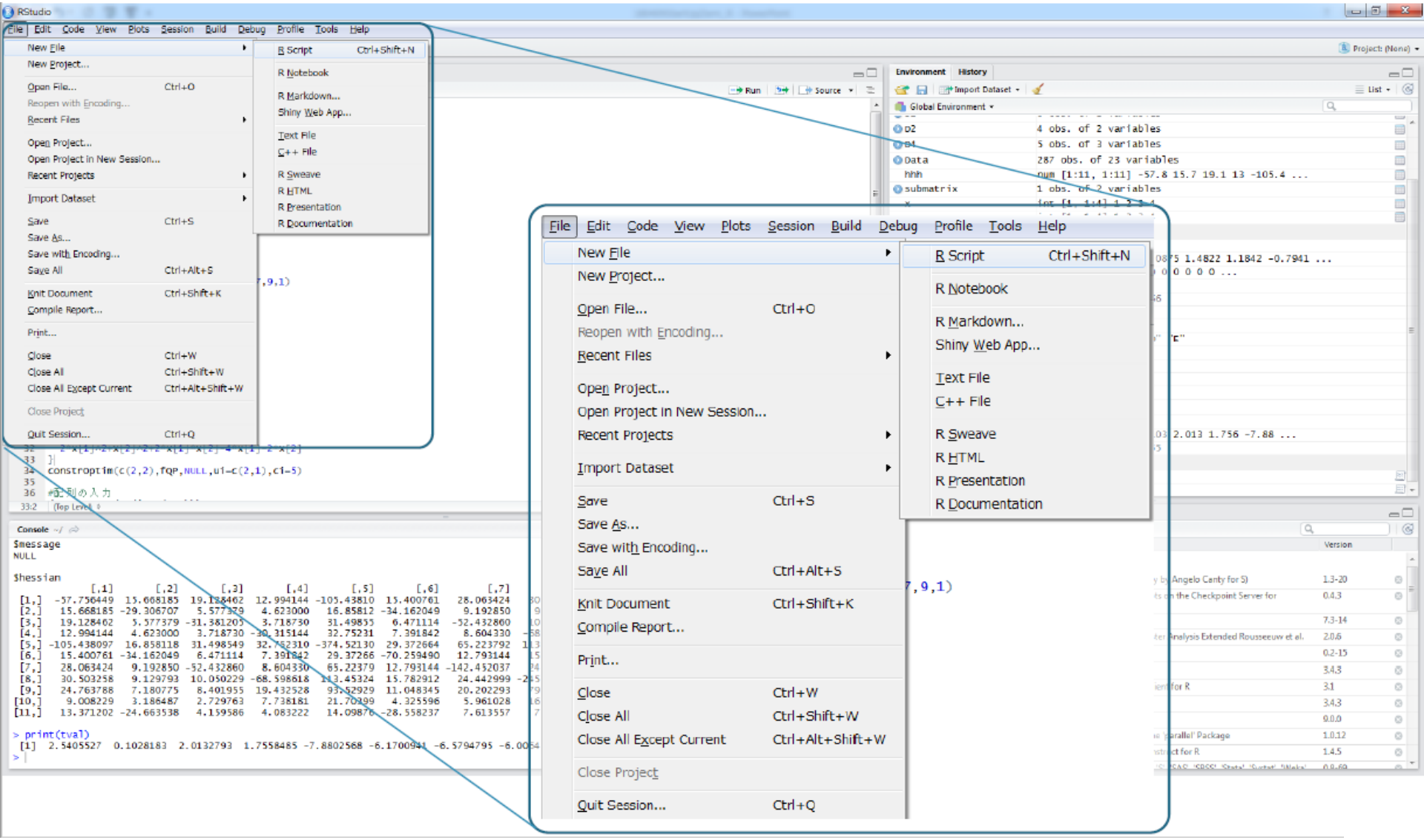
R studioの画面



エディタにプログラムを書いて実行→それ以外の部分に実行結果が出る

使ってみる

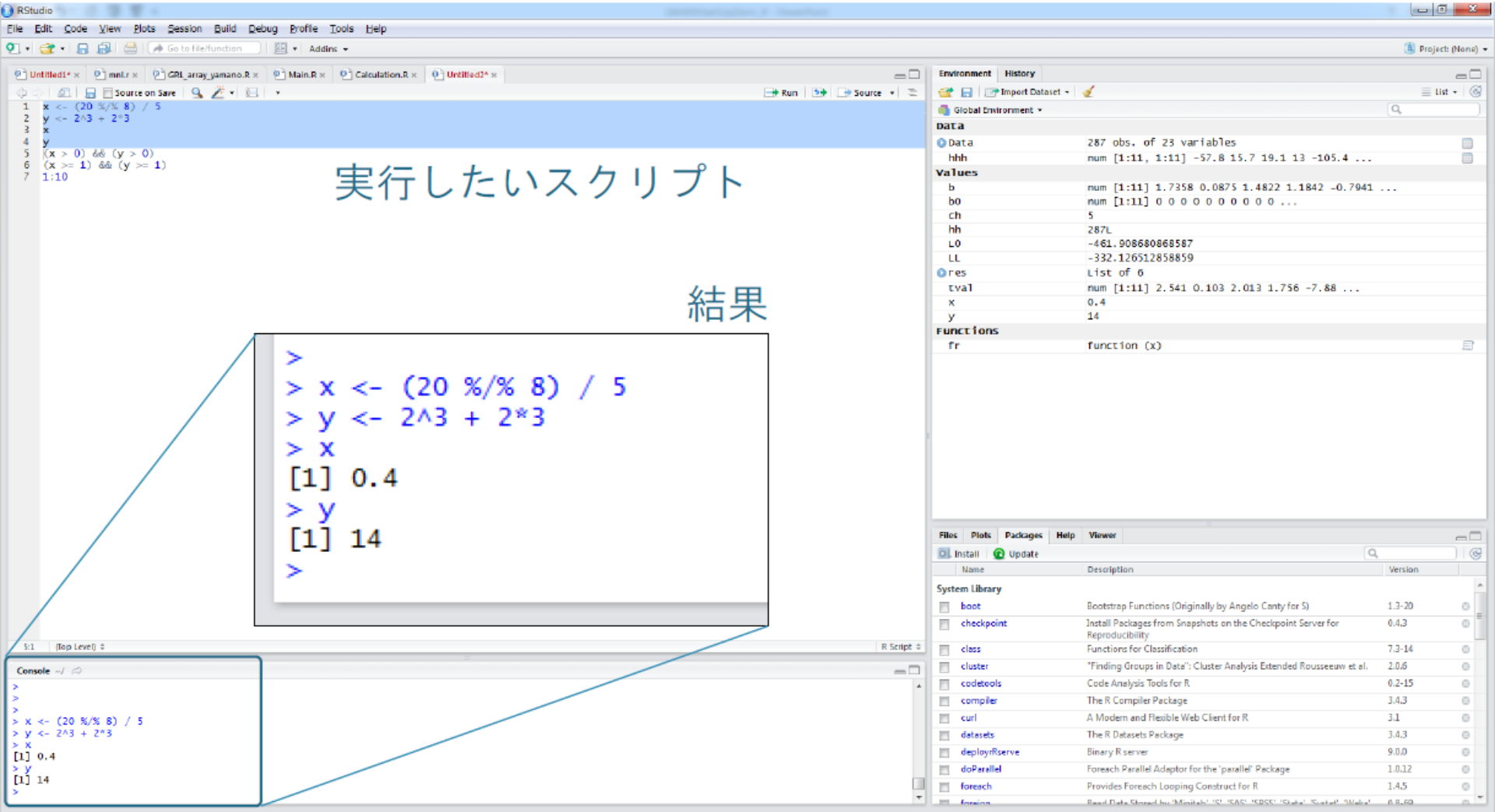
R studioを起動　→　新規ファイル作成（R Script）　→　スクリプトを書いて実行



使ってみる

R studioを起動　→　新規ファイル作成（R Script）　→　スクリプトを書いて実行

実行したいスクリプトを選択してRun



基本事項

- 演算子
- 変数とオブジェクト
- 関数
- 繰り返し文

ここで全部わかります！
任せました！（信頼）

R-Tips
http://cse.naro.affrc.go.jp/takezawa/r-tips/r.html

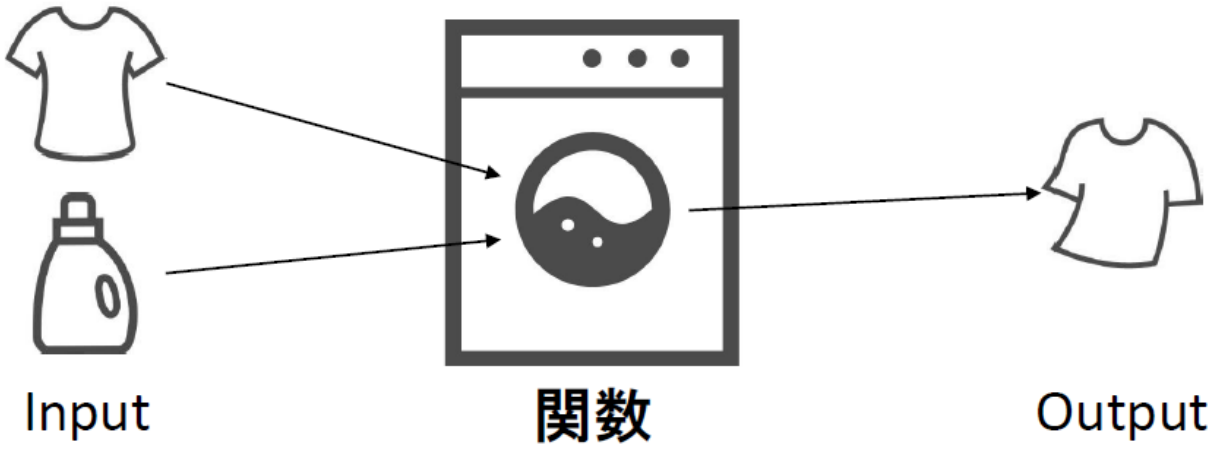
注意事項

- 全角と半角，大文字と小文字は区別されます
- 誤字，余計な空白もNG
- コードを書いてると，バージョン管理が結構大事なことに気づいてくる

大体これのせいでフォルダがぐちゃぐちゃになる…

→ 世の人々は「Git」で管理しているらしい

関数



定義

```
(関数名) <- function (引数) {
                処理
                return (戻り値)
            }
```

例：絶対値を求める

```
10 abs <- function(a,b){
11   if(a>=b){
12     return(a-b)
13   }else{
14     return(b-a)
15   }
16 }
```

←aとbを入れたら

←そのa, bに
この処理をします

```
>
> x <- abs(3,6)
> print(x)
[1] 3
>
```

←引数に代入

←戻り値GET

MNLモデルのパラメータ推定

データの読み込み

```
1  ### Multinomial Logit model estimation
2
3  ### データファイルの読み込み
4  Data <- read.csv("C:/ensyu2011all.csv",header=T)
5  ## データ数:Data の行数を数える
6  hh <- nrow(Data)
7
```

コメントアウト

データのパスを指定

データの行数（＝サンプル数）を変数hhに代入

Data

トリップID	ユーザーコード	目的コード	目的	出発日付	到着日付	トリップ時間_秒	出発地コード	出発地	出発地ユーザー施設名	出発地施設属性コード
15521	ec003	410	食事	2007/2/19 18:03	2007/2/19 18:41	2314	10578	松山河川国道事務所	松山河川国道事務所	120
15556	ec003	200	帰宅	2007/2/19 21:46	2007/2/19 22:14	1648	10581	なが坂	なが坂	190
15595	ec003	100	出勤・登校	2007/2/20 7:51	2007/2/20 8:19	1635	10579	自宅	自宅	110
15880	ec003	0	--	2007/2/20 21:03	2007/2/20 21:03	0				
15882	ec003	100	出勤・登校	2007/2/21 7:46	2007/2/21 8:11	1459	10579	自宅	自宅	110
15944	ec003	310	業務	2007/2/21 11:21	2007/2/21 12:57	5770	10578	松山河川国道事務所	松山河川国道事務所	120
16008	ec003	300	帰社・帰校	2007/2/21 15:58	2007/2/21 17:04	3958	10580	今治商工会議所	今治商工会議所	270
16068	ec003	200	帰宅	2007/2/21 19:19	2007/2/21 19:40	1258	10578	松山河川国道事務所	松山河川国道事務所	120
16135	ec003	100	出勤・登校	2007/2/22 7:42	2007/2/22 8:02	1176	10579	自宅	自宅	110
16199	ec003	310	業務	2007/2/22 11:29	2007/2/22 12:21	3098	10578	松山河川国道事務所	松山河川国道事務所	120
16290	ec003	300	帰社・帰校	2007/2/22 16:27	2007/2/22 18:00	5573	12072	西条国道維持出張所	西条出張所	130
16406	ec003	100	出勤・登校	2007/2/23 7:38	2007/2/23 8:02	1455	10579	自宅	自宅	110
16492	ec003	310	業務	2007/2/23 12:10	2007/2/23 12:36	1576	10578	松山河川国道事務所	松山河川国道事務所	120
16565	ec003	300	帰社・帰校	2007/2/23 16:45	2007/2/23 17:10	1537	12073	愛媛大学工学部	愛大	130
16655	ec003	200	帰宅	2007/2/23 20:47	2007/2/23 21:11	1466	10578	松山河川国道事務所	松山河川国道事務所	120
16982	ec003	400	買い物	2007/2/24 19:01	2007/2/24 20:56	6916	10579	自宅	自宅	110
17131	ec003	499	その他私用	2007/2/25 11:34	2007/2/25 11:46	719	10579	自宅	自宅	110
17147	ec003	400	買い物	2007/2/25 12:04	2007/2/25 12:15	640	12071	実家	実家	140
17172	ec003	200	帰宅	2007/2/25 13:54	2007/2/25 14:01	393	12075	いよ立花駅	立花駅	280
17227	ec003	410	食事	2007/2/25 16:06	2007/2/25 17:47	6051	12071	実家	実家	140
17305	ec003	200	帰宅	2007/2/25 20:41	2007/2/25 20:50	530	12071	実家	実家	140
17346	ec003	100	出勤・登校	2007/2/26 7:40	2007/2/26 8:11	1856	10579	自宅	自宅	110

MNLモデルのパラメータ推定

パラメータの初期値の設定

```
 8  ## パラメータの初期値の設定
 9  b0 <- numeric(6)
10
```

0が6個並んだベクトルをb0に代入

- 効用確定項の式にパラメータがn個あったらn個の初期値が必要
- 初期値はだいたい0にすることが多い

$$V_{car} = ASC_{car} + \beta_{time} TravelTime + \beta_{cost}(\frac{Cost}{Income}) + \gamma_{worker} Worker$$

$$V_{car} = \quad 0 \quad + \beta_{time} TravelTime + \beta_{cost}(\frac{Cost}{Income}) + 0$$

MNLモデルのパラメータ推定

パラメータの宣言

```
11  ##### Logit model の対数尤度関数の定義 #####
12
13  fr <- function(x) {
14    ### パラメータの宣言:
15    ## 定数項
16    b1 <- x[1]
17    b2 <- x[2]
18    b3 <- x[3]
19    b4 <- x[4]
20
21    ## 目的地までの所要時間
22    d1 <- x[5]
23
24    ## 料金
25    f1 <- x[6]
26
27    ## 対数尤度のための変数を宣言
28    LL = 0
29
```

xというパラメータ（ベクトル）を入れたらその時の対数尤度を計算する関数（fr）をつくる

xの各要素がどのパラメータなのか宣言

対数尤度の初期値を宣言

MNLモデルのパラメータ推定

効用確定項Vの計算

```
30    ### 今回用いる目的地は以下の 5 つ.
31    ## 鉄道(train)
32    ## バス(bus)
33    ## 自動車(car)
34    ## 自転車(bike)
35    ## 徒歩(walk)
36
37    ## 効用の計算:説明変数にしたい列を入れる.
38                                    # 時間                          # 料金
      # 定数項
39    train <- Data$代替手段生成可否 train * exp( d1*Data$総所要時間 train/100
      + f1*Data$費用 train/100 + b1*matrix(1,nrow =hh,ncol=1))
40    bus   <- Data$代替手段生成可否 bus   * exp( d1*Data$総所要時間 bus/100
      + f1*Data$費用 bus/100 + b2*matrix(1,nrow =hh,ncol=1))
41    car   <- Data$代替手段生成可否 car   * exp( d1*Data$所要時間 car/100
      + b3*matrix(1,nrow =hh,ncol=1))
42    bike  <- Data$代替手段生成可否 bike  * exp( d1*Data$所要時間 bike/100
      + b4*matrix(1,nrow =hh,ncol=1))
43    walk  <- Data$代替手段生成可否 walk  * exp( d1*Data$所要時間 walk/100)
44
```

0 or 1の値

$$\frac{\exp(\mu V_{in})}{\sum_{j \in C} \exp(\mu V_{jn})}$$

ここを書いていってる

色んなテクが詰まってる

MNLモデルのパラメータ推定

選択確率の計算

```
45    ### 選択確率の計算
46    ## 分母となる，各々の exp(V) の和をつくる
47    deno <- (car + train + bus + bike + walk)
48
49    ## それぞれ計算する
50    Ptrain <- Data$代替手段生成可否 train*(train / deno)
51    Pbus   <- Data$代替手段生成可否 bus  *(bus   / deno)
52    Pcar   <- Data$代替手段生成可否 car  *(car   / deno)
53    Pbike  <- Data$代替手段生成可否 bike *(bike  / deno)
54    Pwalk  <- Data$代替手段生成可否 walk *(walk  / deno)
55
```

$$\frac{\exp(\mu V_{in})}{\sum_{j \in C} \exp(\mu V_{jn})}$$

ここ

それぞれの選択肢を選択する確率

MNLモデルのパラメータ推定

選択確率の補正

```
56    ## 選択確率が 0 になってしまった場合に起こる問題の回避
57    Ptrain   <- (Ptrain!=0) * Ptrain + (Ptrain ==0)
58    Pbus     <- (Pbus!=0)   * Pbus   + (Pbus   ==0)
59    Pcar     <- (Pcar!=0)   * Pcar   + (Pcar   ==0)
60    Pbike    <- (Pbike!=0)  * Pbike  + (Pbike  ==0)
61    Pwalk    <- (Pwalk!=0)  * Pwalk  + (Pwalk  ==0)
62
```

鉄道の選択確率が0 → 鉄道を選択することの尤度も0

→ 対数尤度 $\ln 0 = \infty$ となりエラーという問題を回避するために

選択確率が0じゃなければそのまま，0ならば1にするという処理を施している

Tips：論理式

MNLモデルのパラメータ推定

選択結果

```
63  ## 選択結果
64  Ctrain   <- Data$代表交通手段 == "鉄道"
65  Cbus     <- Data$代表交通手段 == "バス"
66  Ccar.    <- Data$代表交通手段 == "自動車"
67  Cbike    <- Data$代表交通手段 == "自転車"
68  Cwalk    <- Data$代表交通手段 == "徒歩"
69
```

選択結果が鉄道の行は1，それ以外は0を出力し，Ctrainに代入

MNLモデルのパラメータ推定

対数尤度の計算

```
70    ## 対数尤度の計算
71    LL <- colSums(Ctrain*log(Ptrain) + Cbus*log(Pbus) +
72                  Ccar  *log(Pcar)   + Cbike  *log(Pbike) +Cwalk *log(Pwalk))
73
74  }
75
```

$$LL_n(\beta_1, \beta_2, \ldots \beta_K) = \sum_{n=1}^{N} \sum_{i} y_{in} log P_n(i)$$

これを計算してLLに代入

ここでようやく
「パラメータxを入れたら，実際の行動データDataを参照して対数尤度LLを計算する関数fr」が完成

MNLモデルのパラメータ推定

対数尤度関数の最大化

```
76  ##### 対数尤度関数 fr の最大化#####
77
78  ##パラメータ値の最適化
79  res <- optim(b0,fr, method = "Nelder-Mead", hessian = TRUE, control=list(fnscale=-1))
80
```

最適化関数optim （Rがあらかじめ用意してくれている関数）

optim(par, fn, gr =NULL, method = “~~~~”, lower, upper, control = list(), hessian)

b0というパラメータを初期値として，frを最大化するようにパラメータを動かしながら反復して探索.
その時の探索方法は”~~~~”.
ヘッセ行列を返すように指示して，最小化ではなく最大化.

という命令をしている.

対数尤度を最大化したときの結果のもろもろを変数resに入れている.

MNLモデルのパラメータ推定

結果の表示のための準備

```
81  ## パラメータ推定値，ヘッセ行列
82  b   <- res$par
83  hhh <- res$hessian
84
```

対数尤度を最大化するパラメータ
その時のヘシアン

```
85  ## t 値の計算
86  tval <- b/sqrt(-diag(solve(hhh)))
87
```

t値を計算

```
88  ## 初期尤度
89  L0 <- fr(b0)
90  ## 最終尤度
91  LL <- res$value
92
```

尤度比を出すために初期尤度と最終尤度を計算（後述）

MNLモデルのパラメータ推定

結果の表示

```
93  ##### 結果の出力 #####
94  print(res)
95  ## 初期尤度
96  print(L0)
97  ## 最終尤度
98  print(LL)
99  ##ρ^2 値
100 print((L0-LL)/L0)
101 ## 修正済ρ^2 値
102 print((L0-(LL-length(b)))/L0)
103 ##パラメータ推定値
104 print(b)
105 ## t 値
106 print(tval)
107
```

結果を全部print!!

MNLモデルのパラメータ推定

結果の意味

```
> print(res)
$par
[1]  0.5166185024 -1.7122114083 -1.5555058043 -1.3701750852 -0.1111461187  0.0001876913

$value
[1] -1265.759

$counts
function gradient
     104       22

$convergence
[1] 0

$message
NULL

$hessian
             [,1]         [,2]         [,3]       [,4]          [,5]          [,6]
[1,]   -173.841516     7.556166   119.703557   27.11163 -2.452542e+03    -62600.711
[2,]      7.556166   -34.187062     9.432767    7.43628  4.993405e+01     -5191.759
[3,]    119.703557     9.432767  -225.822769   53.23063  2.626840e+03     54186.559
[4,]     27.111629     7.436279    53.230633 -146.45477  6.721227e+02      7664.358
[5,]  -2452.541643    49.934052  2626.840222  672.12266 -8.258336e+04  -1094200.328
[6,] -62600.710523 -5191.758620 54186.559083 7664.35787 -1.094200e+06 -36048315.893
```

最大尤度を与えるパラメータ

最大尤度

計算の繰り返し回数，1階偏微分を行った回数

収束したか否か

エラーが出たらここで言われる

ヘッセ行列

MNLモデルのパラメータ推定

結果の意味

```
> ## 初期尤度
> print(L0)
[1] -2109.710
> ## 最終尤度
> print(LL)
[1] -1265.759
> ##ρ^2 値
> print((L0-LL)/L0)
[1] 0.4000319
> ## 修正済ρ^2 値
> print((L0-(LL-length(b)))/L0)
[1] 0.3971879
> ##パラメータ推定値
> print(b)
[1]  0.5166185024 -1.7122114083 -1.5555058043 -1.3701750852 -0.1111461187  0.0001876913
> ## t 値
> print(tval)
[1]   3.4565899  -8.6617975 -13.2299589 -12.4937770 -20.4124961   0.6119783
>
>
```

初期尤度から最終尤度までどれだけ尤度が上がったか

↑からサンプル数の影響を排除した値

推定されたパラメータ

t値

- パラメータの正負が直観に合うかなどのチェックをここでする
- パラメータの値そのものは**直接の意味はない**
- 重要なのは他のパラメータに比べてどれだけ差があるか（弾性，限界効果）

MNLモデルのパラメータ推定

結果の書き方

説明変数	パラメータ	t値
所要時間[分/10]	-0.762	-7.41 **
費用[円/100]	-0.044	-0.98
乗車外時間[分/10]	-1.049	-8.07 **
女性ダミー	1.811	5.53 **
定数項（鉄道）	3.168	7.46 **
定数項（自動車）	0.868	2.08 *
定数項（自転車）	0.046	0.13
定数項（徒歩）	2.037	4.81 **
サンプル数	400	
初期尤度	-564.18	
最終尤度	-344.08	
尤度比	0.390	
修正済み尤度比	0.376	

*5%有意　**1%有意

- 列は線で区切らない
- 有意なパラメータに*
- 単位を書く

MNLモデルのパラメータ推定

うまく回らないときは

- データセットの不備（欠損，誤字，数字と文字の混在）
- パラメータの設定（個数，宣言）
- ファイルのパス
- 関数の式
  - 括弧の閉じ忘れ
  - 説明変数の数
  - 列名の指定

etc...

デバッグに慣れよう

一行ずつ実行してみることがコツ

課題

■ **MNLによる交通機関選択モデルのパラメータ推定を行う**

- 前回の横浜tripデータにクリーニングをかけて，交通機関ごとのLOSを加えたものを渡します
- サンプルコードも渡します
- 何と少し書き換えてRunするだけで回ります

■ **MNLによる目的地選択モデルのパラメータ推定を行う**

- 松山の買い物トリップのデータを抽出したものを渡します
- 交通機関選択モデルのコードを書き替えてパラメータ推定してみてください

■ **なんでもいいので自分で効用関数を設定してパラメータ推定を行う**

ここまででパラメータ推定のイメージはつかめるはず

政策シミュレーション

パラメータ推定の結果から，ある政策を実施した際の起こる変化を予測し，その効果について述べてください

例）横浜臨海部で自動運転タクシーを導入したい．料金をいくらにすればどれくらいの利用者が確保できるだろうか？

ヒント

1. 交通機関選択モデルのパラメータを推定する
2. 自動運転タクシーを導入した際のLOSについて考える
3. ある距離帯の移動における自動運転タクシーの選択確率を計算する

I.I.A特性の緩和

- I.I.A特性を考慮したモデルにすることで，モデルの当てはまりがよくなる場合がある
- NLモデル，CNLモデルなどについてもサンプルコードを渡します
- これらのモデルの詳細については過去のゼミによくまとまっているので参考にしてください（wikiで「NLモデル」とかで検索！）

選択肢集合，LOS

■ 選択肢集合の設定は難しい

- 買い物トリップのデータに対して，どれくらいの行き先が候補にあって，その場所が選ばれているのかは厳密には分からない
- 経路選択をするときにどの経路が選択肢に考えられているかもわからない

■ LOSは普通，行動データにはない

- 行動データは「選択の結果」のみについての情報しかないため，代替選択肢についての情報は自分で作る必要がある

ここをどうするかを考えるのも実際の推定ではポイントになります