Rで統計解析入門

(番外篇) データハンドリングと作図 Tips

本日のメニュー

1. データハンドリング
2. 作図 Tips

準備：データの型

- R には「データの型」という概念があり，「数値」「文字」「日付」「因子（カテゴリ）」などを区別する ⇒ 日付の処理例は次頁

```
> height <- c(158,162,177,173,166)    # 数値型
> group  <- c("A","A","B","C","C")    # 文字型
> group  <- as.factor(group)          # 関数 as.factor で
>                                     # 因子型(カテゴリ)に変換
> groupc <- as.character(group)       # 文字型に変換
> date   <- as.Date("111111", format="%y%m%d") # 日付型に変換
```

- 外部ファイルを R に読み込むと「数値」は「数値型」，「文字」は「因子型（カテゴリ）」に自動変換される

  ⇒ 「文字」を「文字型」としたい場合は要変換！

準備：日付データのハンドリング例

```
> as.Date("2012/01/26", format="%Y/%m/%d") # 文字列を日付に変換
[1] "2012-01-26"
> x <- as.Date("2012/01/26", format="%Y/%m/%d") -
+       as.Date("111111", format="%y%m%d") # 日数の差を計算
> as.numeric(x)                           # 結果を数値に変換
[1] 76
```

命令	機能
%A，%a	曜日の英語名（小文字は略記）
%B，%b	月の英語名（小文字は略記）
%d	日（01-31）
%m	月（01-12）
%Y，%y	西暦（大文字：4 桁表示，小文字：2 桁表示）

場面設定と使用するデータの概要

- 糖尿病予備軍の患者さんに A または B の糖尿病予防薬を投与し，投与終了日における HbA1c（6.5%以上であれば糖尿病）を測定し，治療効果を確認する

- データフレーム「demo」⇒ C:/demo.txt にデータあり
  - ID：患者さんを表す番号
  - AGE：年齢（歳），数値
  - GENDER：性別（1：男性，2：女性），数値
  - DATE：薬剤の投与開始日，文字列

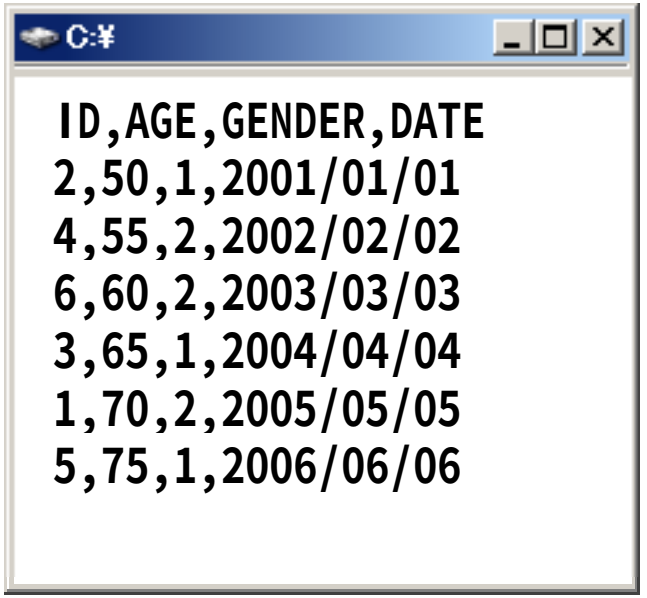
- データフレーム「hba1c」⇒ C:/hba1c.txt にデータあり
  - ID：患者さんを表す番号
  - GROUP：投与される薬剤の種類，文字（A，B）
  - HBA1C：薬剤の投与終了日に測定したHbA1c（%），数値
  - DATE：薬剤の投与終了日，文字列

データフレーム「demo」の準備

- 患者さんの背景データ「demo.txt」を読み込み

  ⇒ 列名があり，データ間がコンマで区切られている

```
> demo <- read.table("C:/demo.txt",
+                    header=T, sep=",")
> demo

  ID AGE GENDER       DATE
1  2  50      1 2001/01/01
2  4  55      2 2002/02/02
3  6  60      2 2003/03/03
4  3  65      1 2004/04/04
5  1  70      2 2005/05/05
6  5  75      1 2006/06/06
```



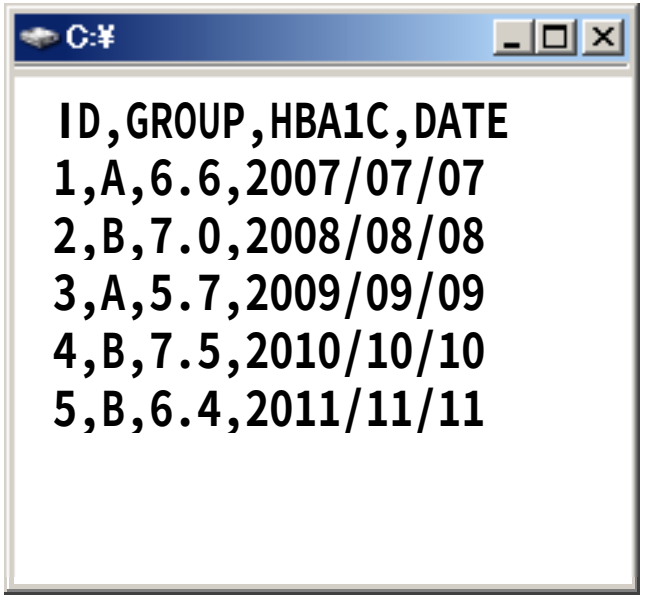
demo.txt

データフレーム「hba1c」の準備

- 患者さんの臨床検査データ「hba1c.txt」を読み込み

  ⇒ 列名があり，データ間がコンマで区切られている

```
> hba1c <- read.table("C:/hba1c.txt",
+                     header=T, sep=",")
> hba1c

  ID GROUP HBA1C       DATE
1  1     A   6.6 2007/07/07
2  2     B   7.0 2008/08/08
3  3     A   5.7 2009/09/09
4  4     B   7.5 2010/10/10
5  5     B   6.4 2011/11/11
```



hba1c.txt

作成したいデータのイメージ（目標）

	ID	GROUP	AGE	AGE_CT	GENDER	HBA1C	EVENT	DAY
1	1	A	70	>=65	Female	6.6	1	793
2	2	B	50	<65	Male	7.0	1	2776
3	3	A	65	>=65	Male	5.7	0	1984
4	4	B	55	<65	Female	7.5	1	3172
5	5	B	75	>=65	Male	6.4	0	1984

① 「demo」を ID が小さい順に並べ替え（ソート）
② 年齢（AGE）が「65歳未満」「65歳以上」を表す変数 AGE_CT を作成
③ 性別の変数を因子型（カテゴリ）に変換
④ HbA1c が 6.5 未満（0）か 6.5 以上（1）かを表す変数 EVENT を作成
⑤ 「demo」と「hba1c」をくっつけ，両方に存在するレコードのみ残す
⑥ DAY（投与終了日－投与開始日）を作成
⑦ 変数を上記に絞り，変数の順番も上記に従う

【道具】データへのアクセス方法

データフレーム x に対する命令	機能
x$列名，x["列名"]，x[["列名"]]	指定した列データを表示
x[2], x[[2]]	2 番目の列データを表示
x[3, 2], x[[3, 2]]	3 行 2 列目のデータを表示
x[[3,"列名"]], x[[3,"列名"]]	指定した列の 3 行目のデータを表示
x[c(1, 2)]	1 列目と 2 列目のデータを表示
x[c(3, 4),]	3 行目と 4 行目のデータを表示
x[,c(T,F,T)]	論理ベクトル c(T,F,T) が TRUE となっている列を表示
x[GENDER==2,]	性別が 2（女性）である行を表示
x[,GENDER==2 & AGE>60]	性別が 2（女性）かつ年齢が 60歳より大きい行を表示
subset(x, gender==2 & age>60)	x[,GENDER==2 & AGE>60] と同様の機能

【道具】データの加工・抽出

データフレーム x に対する命令	機能
head(x，n=a)	先頭から a 行だけ抽出する
tail(x，n=b)	末尾から b 行だけ抽出する
na.omit(x)	NA を含む行を削除する
transform(x，y=ベクトル)	データフレーム x に新たな列 y を追加する
subset(x，条件式)	条件式に合う行のみを抽出する
subset(x，条件式，ベクトル)	ベクトルで指定した列に対し，条件式に合う行のみを抽出する
reshape(x, ...)	データフレーム x を横展開／縦展開する
apply(x[,範囲], 1, 関数)	データフレーム x の指定した範囲について，各行ごとに関数を適用する （各列ごと： apply(x[,範囲], 2, 関数) とする）

【道具】データの結合など

データフレーム x に対する命令	機能
ncol(x)	x の列数（変数の数）を求める
nrow(x)	x の行数（データ数）を求める
names(x)	x の列名を表示する
rbind(x,y)	x と y を縦に並べて結合する
cbind(x,y)	x と y を横に並べて結合する
data.frame(x,y)	x と y を横に並べて結合する
merge(x,y)	x と y をくっつける（マージする） ※ all=T を指定するとデータを全て残す all=T を指定しなければデータの共通部分が結果として返される

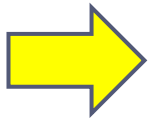
※ たまに関数 attach() や detach() を使用している資料が見受けられるが、
使わない方が良い（実際にデータ解析を行う場合ではまず使わないです）

① データフレーム「demo」の処理

- ID が小さい順に並べ替え（ソート）を行う
- そのままだと行番号がバラバラ

```
> sortlist <- order(demo$ID)          # 順番を取得
> demo     <- demo[sortlist,]         # 整列
```

	ID	AGE	GENDER	DATE
1	2	50	1	2001/01/01
2	4	55	2	2002/02/02
3	6	60	2	2003/03/03
4	3	65	1	2004/04/04
5	1	70	2	2005/05/05
6	5	75	1	2006/06/06



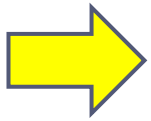
	ID	AGE	GENDER	DATE
5	1	70	2	2005/05/05
1	2	50	1	2001/01/01
4	3	65	1	2004/04/04
2	4	55	2	2002/02/02
6	5	75	1	2006/06/06
3	6	60	2	2003/03/03

① データフレーム「demo」の処理

- 行番号がバラバラなので，行番号を整形する

```
> sortlist <- order(demo$ID)          # 順番を取得
> demo     <- demo[sortlist,]         # 整列
> rownames(demo) <- c(1:nrow(demo)) # 行番号の整形
```

	ID	AGE	GENDER	DATE
5	1	70	2	2005/05/05
1	2	50	1	2001/01/01
4	3	65	1	2004/04/04
2	4	55	2	2002/02/02
6	5	75	1	2006/06/06
3	6	60	2	2003/03/03



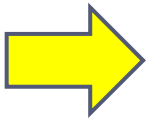
	ID	AGE	GENDER	DATE
1	1	70	2	2005/05/05
2	2	50	1	2001/01/01
3	3	65	1	2004/04/04
4	4	55	2	2002/02/02
5	5	75	1	2006/06/06
6	6	60	2	2003/03/03

② データフレーム「demo」の処理

- 年齢（AGE）が「65歳未満」「65歳以上」を表す変数 AGE_CT を作成

```
> tmp  <- ifelse(demo$AGE<65, "<65", ">=65") # 変数 tmp に退避
> demo <- transform(demo, AGE_CT=tmp)        # demo に変数追加
```

ID	AGE	GENDER	DATE
1	70	2	2005/05/05
2	50	1	2001/01/01
3	65	1	2004/04/04
4	55	2	2002/02/02
5	75	1	2006/06/06
6	60	2	2003/03/03



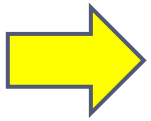
ID	AGE	GENDER	DATE	AGE_CT
1	70	2	2005/05/05	>=65
2	50	1	2001/01/01	<65
3	65	1	2004/04/04	>=65
4	55	2	2002/02/02	<65
5	75	1	2006/06/06	>=65
6	60	2	2003/03/03	<65

③ データフレーム「demo」の処理

- 性別の変数を因子型に変換
  ⇒ 1 を Male（男性），2 を Female（女性）なるカテゴリとして扱う

```
> demo$GENDER <- factor(demo$GENDER, levels=c(1,2),
+                       labels=c("Male","Female"))
> demo$GENDER <- ordered(demo$GENDER)    # 順序関係をつける場合
```

ID	AGE	GENDER	DATE	AGE_CT
1	70	2	2005/05/05	>=65
2	50	1	2001/01/01	<65
3	65	1	2004/04/04	>=65
4	55	2	2002/02/02	<65
5	75	1	2006/06/06	>=65
6	60	2	2003/03/03	<65



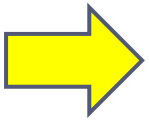
ID	AGE	GENDER	DATE	AGE_CT
1	70	Female	2005/05/05	>=65
2	50	Male	2001/01/01	<65
3	65	Male	2004/04/04	>=65
4	55	Female	2002/02/02	<65
5	75	Male	2006/06/06	>=65
6	60	Female	2003/03/03	<65

④ データフレーム「hba1c」の処理

- HbA1c が 6.5 未満（0）か 6.5 以上（1）かを表す変数 EVENT を追加
  ⇒ 1：イベント発生（糖尿病発症），0：イベントなし

```
> hba1c$EVENT <- ifelse(hba1c$HBA1C<6.5, 0, 1)
```

ID	GROUP	HBA1C	DATE
1	A	6.6	2007/07/07
2	B	7.0	2008/08/08
3	A	5.7	2009/09/09
4	B	7.5	2010/10/10
5	B	6.4	2011/11/11



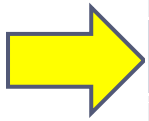
ID	GROUP	HBA1C	DATE	EVENT
1	A	6.6	2007/07/07	1
2	B	7.0	2008/08/08	1
3	A	5.7	2009/09/09	0
4	B	7.5	2010/10/10	1
5	B	6.4	2011/11/11	0

○「demo」と「hba1c」をくっつける前の処理

- 「hba1c」の中の変数 DATE を DATE_END に変更する
  ⇒「demo」の中の変数 DATE とゴッチャにならないように

```
> hba1c <- transform(hba1c, DATE_END=DATE) # DATE_END にコピー
> hba1c$DATE <- NULL                       # DATE を削除
```

ID	GROUP	HBA1C	DATE	EVENT
1	A	6.6	2007/07/07	1
2	B	7.0	2008/08/08	1
3	A	5.7	2009/09/09	0
4	B	7.5	2010/10/10	1
5	B	6.4	2011/11/11	0



ID	GROUP	HBA1C	EVENT	DATE_END
1	A	6.6	1	2007/07/07
2	B	7.0	1	2008/08/08
3	A	5.7	0	2009/09/09
4	B	7.5	1	2010/10/10
5	B	6.4	0	2011/11/11

⑤「demo」と「hba1c」をくっつける

- 「demo」と「hba1c」をくっつけ（マージ），両方に存在するレコードのみ残す ⇒ 出来あがったデータセットを「full」とする

```
> full <- merge(demo, hba1c, by="ID", all=F, sort=T)
```

ID	AGE	GENDER	DATE	AGE_CT	GROUP	HBA1C	EVENT	DATE_END
1	70	Female	2005/05/05	>=65	A	6.6	1	2007/07/07
2	50	Male	2001/01/01	<65	B	7.0	1	2008/08/08
3	65	Male	2004/04/04	>=65	A	5.7	0	2009/09/09
4	55	Female	2002/02/02	<65	B	7.5	1	2010/10/10
5	75	Male	2006/06/06	>=65	B	6.4	0	2011/11/11
~~6~~	~~60~~	~~Female~~	~~2003/03/03~~	~~<65~~				

⑥ データフレーム「full」の処理

- DAY（投与終了日ー投与開始日）を作成する

```
> day  <- as.Date(full$DATE_END, format="%Y/%m/%d") -
+           as.Date(full$DATE, format="%Y/%m/%d") # 日数の差
> day  <- as.numeric(day)                         # 数値に変換
> full <- transform(full, DAY=day)                # 変数を追加
```

ID	AGE	GENDER	DATE	AGE_CT	GROUP	HBA1C	EVENT	DATE_END	DAY
1	70	Female	2005/05/05	>=65	A	6.6	1	2007/07/07	793
2	50	Male	2001/01/01	<65	B	7.0	1	2008/08/08	2776
3	65	Male	2004/04/04	>=65	A	5.7	0	2009/09/09	1984
4	55	Female	2002/02/02	<65	B	7.5	1	2010/10/10	3172
5	75	Male	2006/06/06	>=65	B	6.4	0	2011/11/11	1984

⑦ データフレーム「full」の処理

- 変数を「ID,GROUP,AGE,AGE_CT,GENDER,HBA1C,EVENT,DAY」に絞り，変数の順番も整える（３つの方法あり） ⇒ 完成っ！

```
> full <- full[,c(1,6,2,5,3,7,8,10)]
> full <- full[,c("ID","GROUP","AGE","AGE_CT","GENDER","HBA1C","EVENT","DAY")]
> full <- subset(full, select=c(ID,GROUP,AGE,AGE_CT,GENDER,HBA1C,EVENT,DAY))
```

ID	GROUP	AGE	AGE_CT	GENDER	HBA1C	EVENT	DAY
1	A	70	>=65	Female	6.6	1	793
2	B	50	<65	Male	7.0	1	2776
3	A	65	>=65	Male	5.7	0	1984
4	B	55	<65	Female	7.5	1	3172
5	B	75	>=65	Male	6.4	0	1984

解析用データが出来あがった後は・・・

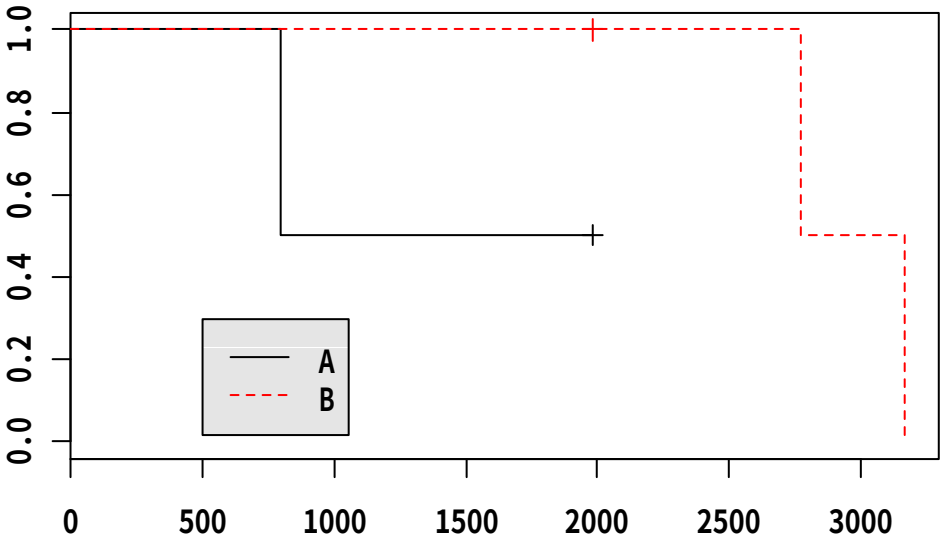
- 年齢（AGE_CT）と性別（GENDER）のクロス集計

```
> result <- table(full$AGE_CT, full$GENDER)
> addmargins(result, 1:2)
      Male Female Sum
  <65    1      1   2
  >=65   2      1   3
  Sum    3      2   5

> prop.table(result, 1)
           Male    Female
  <65  0.5000000 0.5000000
  >=65 0.6666667 0.3333333
```

解析用データが出来あがった後は・・・

- 各薬剤のイベント発現率（カプラン・マイヤー法による推定）

```
> library(survival)
> result <- survfit(Surv(DAY,EVENT) ~ GROUP, data=full)
> plot(result, lty=1:2, col=1:2)
> legend(500,0.3,c("A","B"),col=1:2,lty=1:2,ncol=1,bg='gray90')
```



解析用データが出来あがった後は・・・

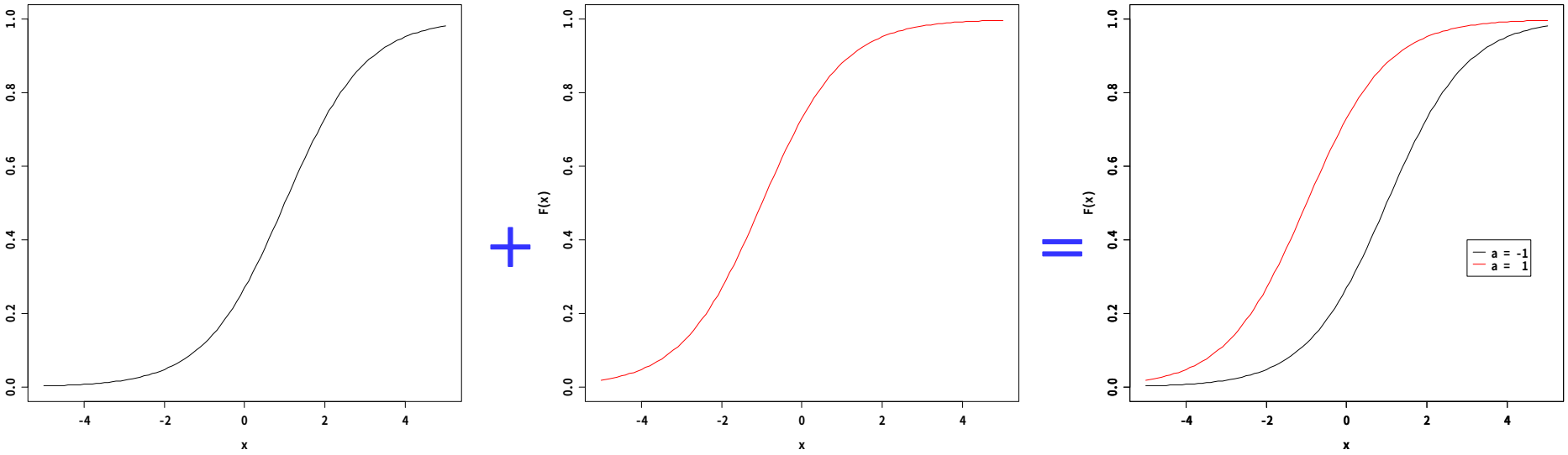
- Cox 回帰：イベント発生までの期間 = 薬剤

```
> coxph(Surv(DAY,EVENT) ~ GROUP, data=full)

Call:
coxph(formula = Surv(DAY, EVENT) ~ GROUP, data = full)


        coef exp(coef) se(coef)         z p
GROUPB -21.7  3.82e-10    41772 -0.000519 1

Likelihood ratio test=1.83  on 1 df, p=0.176  n= 5,
number of events= 3
```

本日のメニュー

1. データハンドリング
2. 作図 Tips
   - 重ねた図の描き方
   - R の画像をパワーポイントに貼る
   - 関数を用いて図を保存する

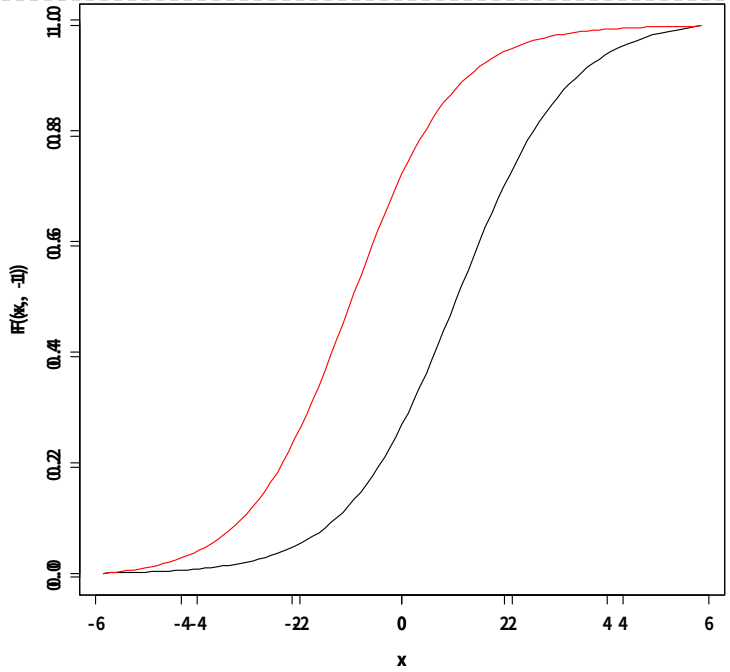
重ねた図の描き方



■ ２枚の図を１枚に重ねて表示することを考える

```
F <- function(x, a) { 1/(1+exp(-a-x)) }   # 作図する関数
curve(F(x,-1), col=1, xlim=c(-5,5),  ylim=c(-0,1), ylab="")
par(new=T)
curve(F(x, 1), col=2, xlim=c(-5,5),  ylim=c(-0,1), ylab="F(x)")
legend(3, 0.4, c("a = -1","a =  1"), lty=1, col=1:2)
```

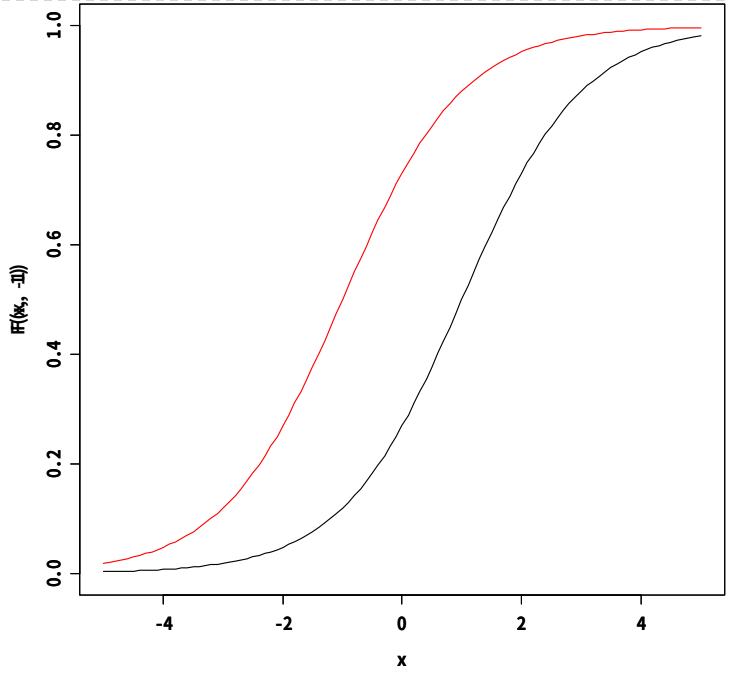
重ねた図の描き方：失敗例①



- 関数 par()で２枚の図を１枚に重ねて表示出来るが，
  そのままでは x 軸と y 軸がバラバラ…

```
curve(F(x,-1), col=1)
par(new=T)   # 前の図を残したまま次の図を描く，という命令
curve(F(x, 1), col=2)
```

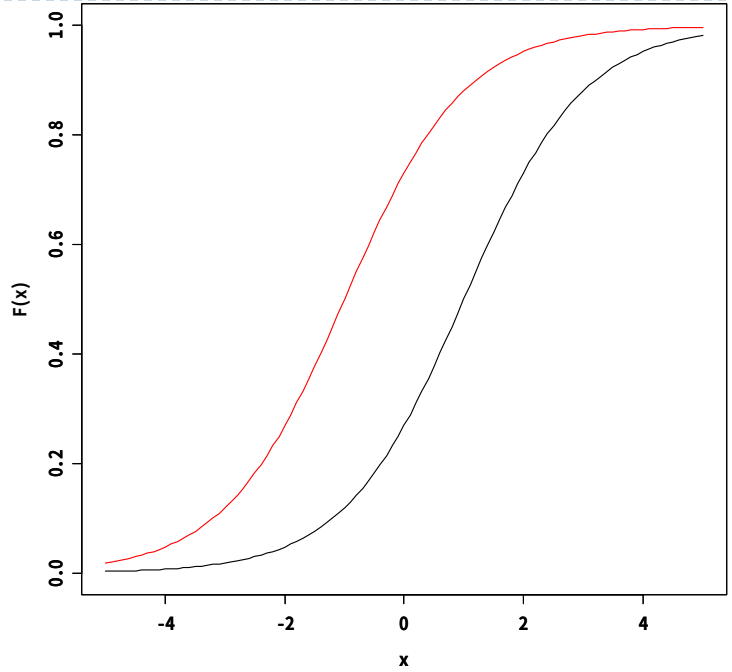
重ねた図の描き方：失敗例②



- 引数 xlim と ylim で２枚の図の座標を合わせればよいが，今度は y 軸ラベルがおかしい…

```
curve(F(x,-1), col=1, xlim=c(-5,5), ylim=c(-0,1))
par(new=T)
curve(F(x, 1), col=2, xlim=c(-5,5), ylim=c(-0,1))
```

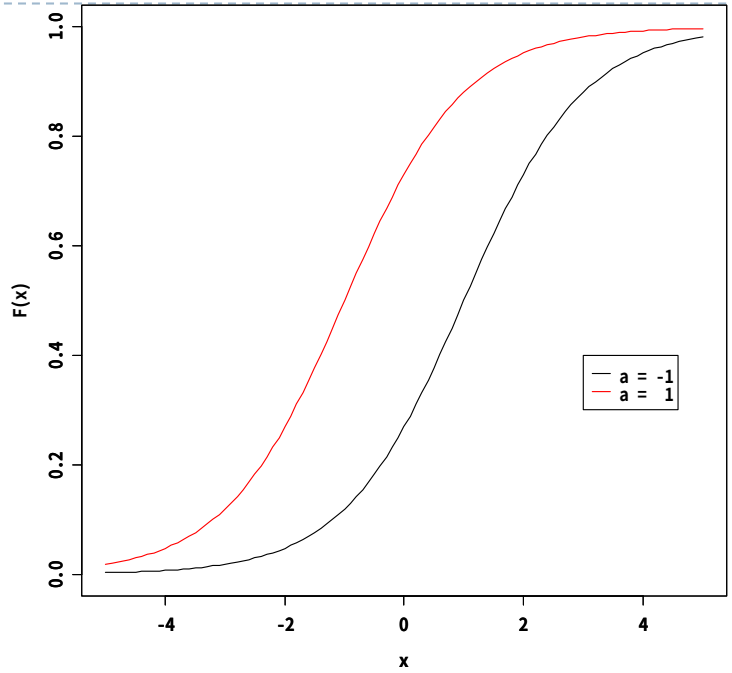
重ねた図の描き方：成功例①



- １つめの引数 ylab を " "（空白）にして，
  ２つめの引数 ylab で y 軸ラベルを指定する

```
curve(F(x,-1), col=1, xlim=c(-5,5),  ylim=c(-0,1), ylab="")
par(new=T)
curve(F(x, 1), col=2, xlim=c(-5,5),  ylim=c(-0,1), ylab="F(x)")
```

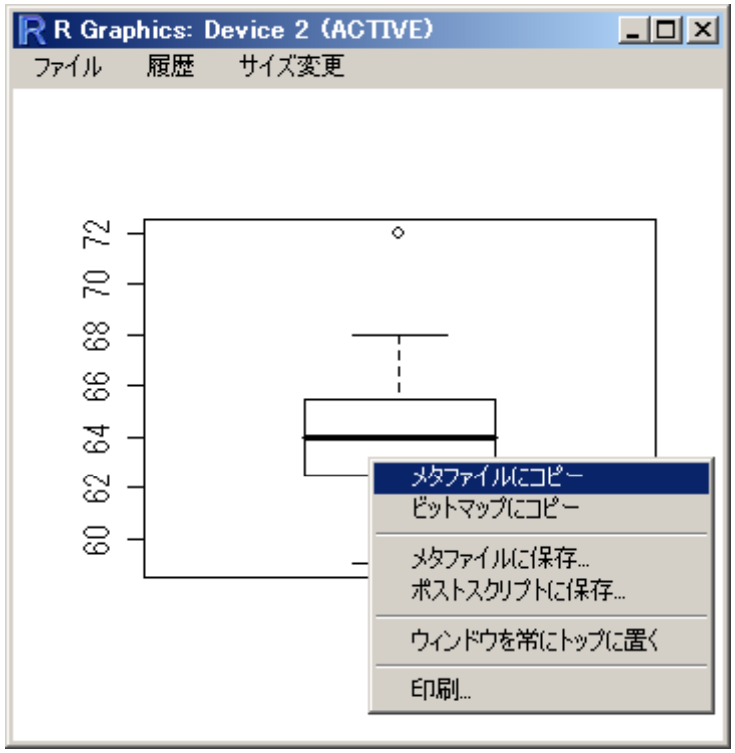
重ねた図の描き方：成功例②



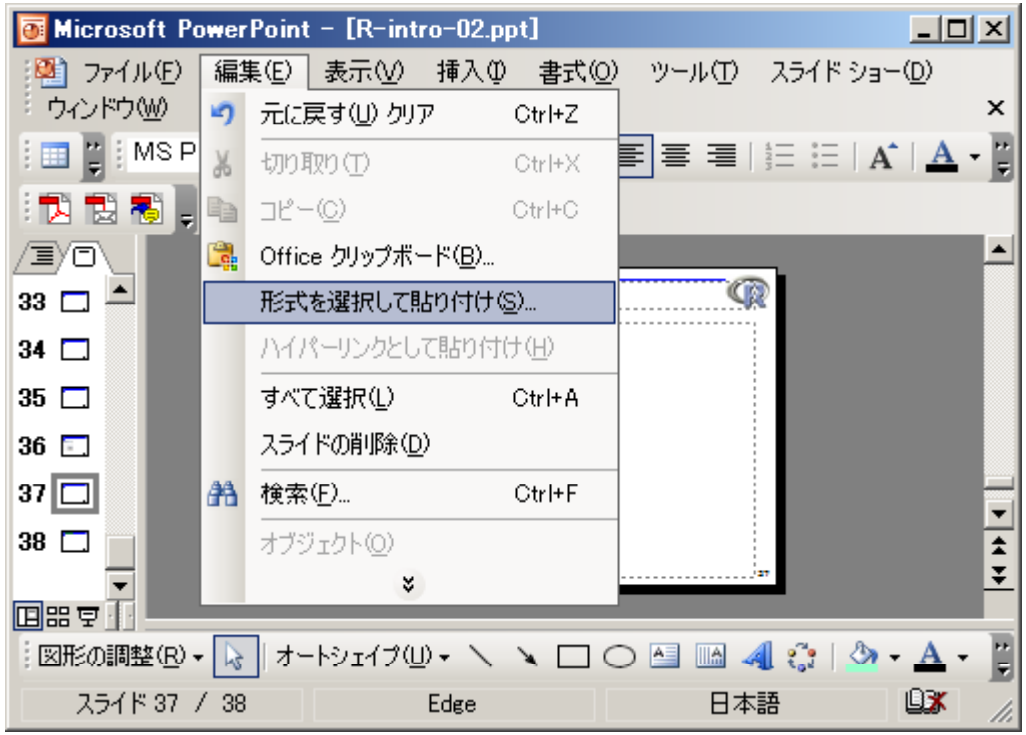
- 凡例をつけて完成！

```
curve(F(x,-1), col=1, xlim=c(-5,5),  ylim=c(-0,1), ylab="")
par(new=T)
curve(F(x, 1), col=2, xlim=c(-5,5),  ylim=c(-0,1), ylab="F(x)")
legend(3, 0.4, c("a = -1","a =  1"), lty=1, col=1:2)
```

R の画像をパワーポイントに貼る

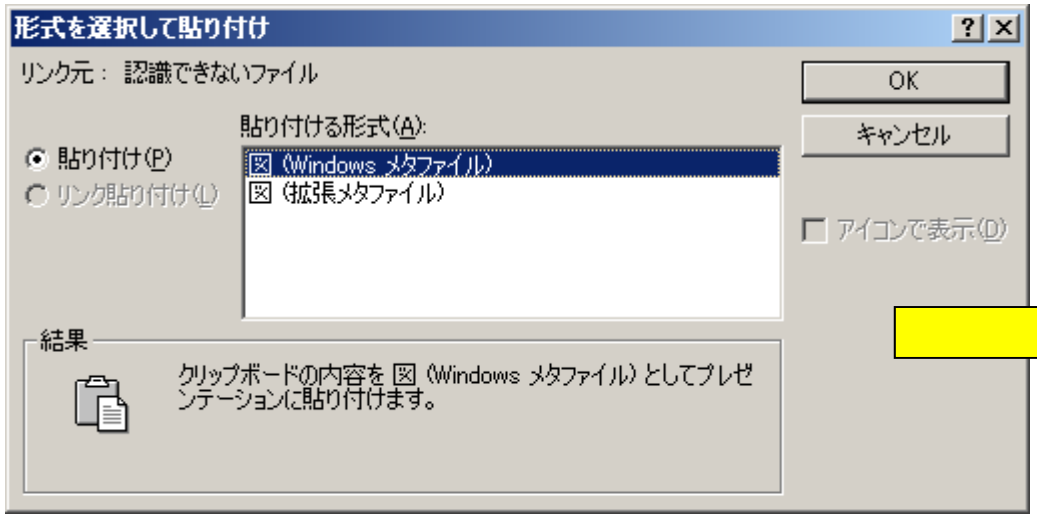


デバイスウインドウを右クリックした後「メタファイルにコピー」を選択

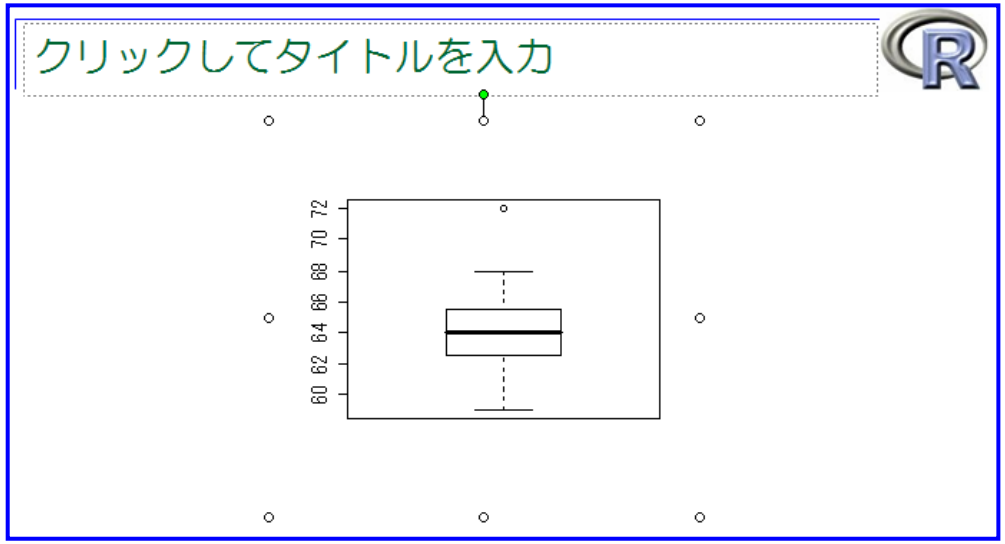


パワーポイント上で［編集］→［形式を選択して貼り付け］

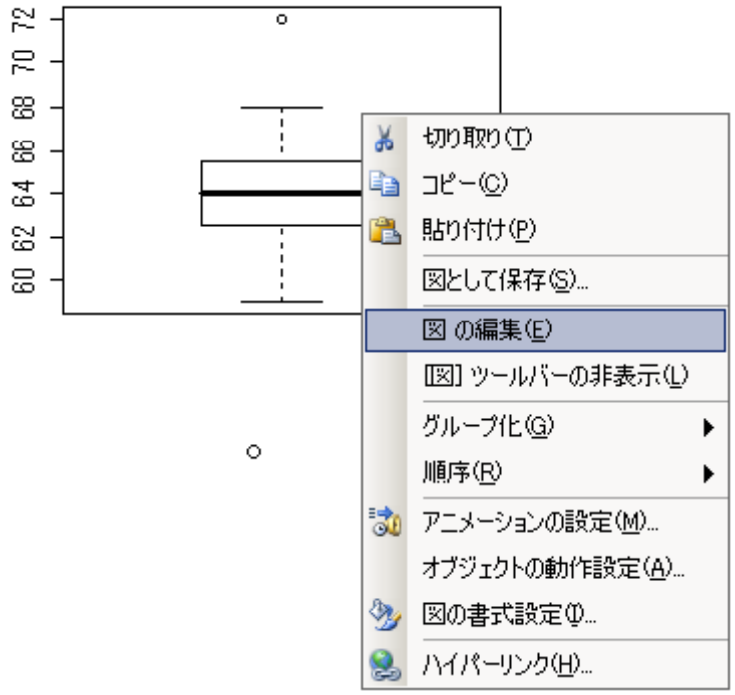
R の画像をパワーポイントに貼る



[OK] をクリックするとめでたく図が貼られる

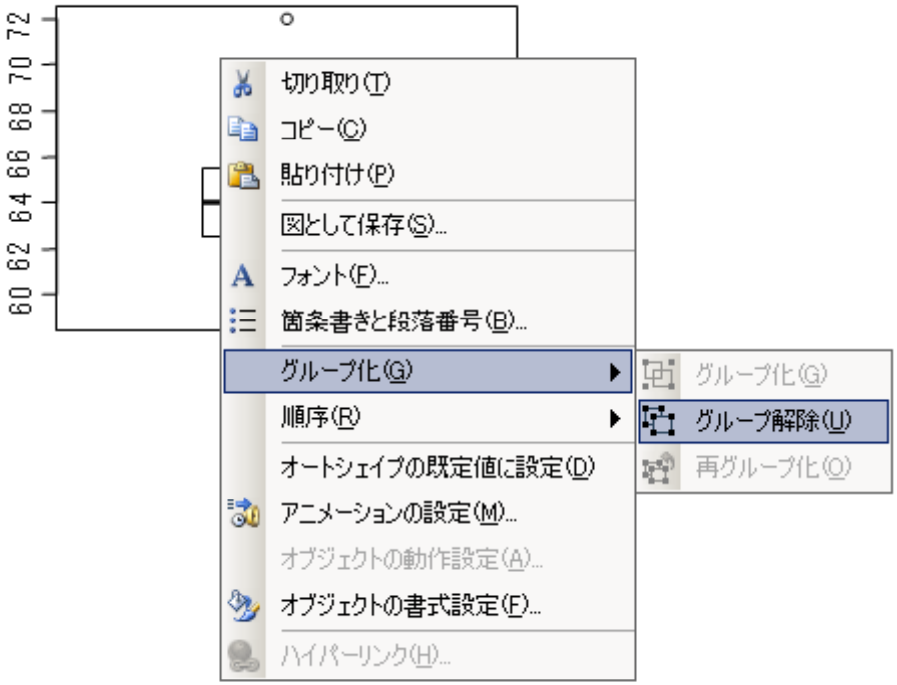


R の画像をパワーポイントに貼る

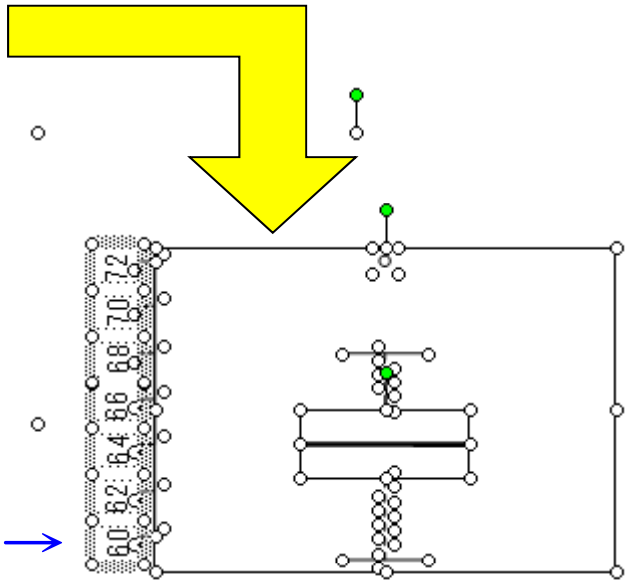


画像を右クリックした後
「図の編集」を選択
（その後 [はい] を選択）

R の画像をパワーポイントに貼る



再度，画像を右クリックした後
「グループ化」「グループ解除」
を選択（必要ない場合もあり）



図が編集できるようになる →

関数を用いて図を保存する

- 図を描いた後，関数 dev.copy() で図を保存
  ⇒ 関数は次頁で紹介する関数（作図デバイス）が使用可

```
> plot(1:10)                          # 図を描いた後
> dev.copy(pdf, "Scatter.pdf")        # pdf に書き出し
> dev.off()                           # ファイルを閉じる
```

- 図を eps ファイルに保存する場合は関数 dev.copy2eps() を使用する

```
> plot(1:10)                                  # 図を描いた後
> dev.copy2eps(file="finename.eps")           # pdf に書き出し
> dev.off()                                   # ファイルを閉じる
```

作図デバイスの種類

- パソコンの画面に表示するためのデバイス（装置）
- 画像ファイルに保存するためのデバイス（装置）
  - **bmp()**：ビットマップ形式
  - **jpeg()**：JPEG 形式（3段階で品質が選択出来る）
  - **pdf()**：ADOBE PDF 形式
  - **pictex()**：LaTeX の画像形式
  - **png()**：PNG 形式
  - **postscript()**：ADOBE PostScript 形式
    ⇒ 関数 dev.copy2eps()でEPSファイルへの保存も出来る
  - **win.metafile()**：windows meta file
    ⇒ emf, wmf 形式：パワーポイント上で編集することが出来る形式

本日のメニュー

1. データハンドリング
2. 作図 Tips
   - 重ねた図の描き方
   - R の画像をパワーポイントに貼る
   - 関数を用いて図を保存する

参考文献

- R データ自由自在
  （ Phil Spector 著，石田 基広 他翻訳，シュプリンガージャパン ）
- The R Tips 第 2 版（オーム社）

Rで統計解析入門

終