９日目：項目のチェック（３）

今日は，度数分布表を作ったり，ヒストグラムを書いたりしてみましょう。なお，今回は項目のチェックが目的なので，あまり込みいった度数分布表などは作りませんので。あくまでも，ある項目で，１の回答が○名，２が△名…といったことを把握するレベルです。

ネットとかを検索しても，度数分布表の作り方は数多く紹介されています。そこでつかわれているのは table()です。

```
table(x[5])
table(x$年齢)
```

これだけの指示で簡単な度数分布表を作ってくれます。しかし，問題なのは欠損値をまったく無視してしまうところです。たとえば，b1 には欠損値がありますが，table(x$b1)で実行すると…

```
> table(x$b1)

1 2 3 4 5
2 6 4 4 3
```

このように出力されます。上の段がカテゴリ，下の段が度数を表しますので，１が２名，２が６名…「NA」は…？？わかりません…

欠損値も表に含めたいなら，以下のように exclude=NULL を入れる必要があります。

```
table(x$b1, exclude=NULL)
```

するとうまくいきます。

```
> table(x$b1, exclude=NULL)

   1    2    3    4    5 <NA>
   2    6    4    4    3    1
```

これで十分と言いたいところですが，欲張れば，%計算もやってほしい。ということで，これらの計算も一緒にやらせることはできないかと探したら，prettyR というパッケージにある，freq が使えるということでした。

そこで，まずは prettyR というパッケージをインストールしてください。

入手できたら，まずはそれを呼び出し，以下のように入力してください。

```
library(prettyR)
fr <- freq(x$b1)
```

```
print(fr, cum.pc=TRUE, show.total=TRUE)
```

```
> library(prettyR)
> fr <- freq(x$b1)
> print(fr, cum.pc=TRUE, show.total=TRUE)

Frequencies for x$b1
         2     3     4     5     1   NATotal
         6     4     4     3     2     1  20
%       30    20    20    15    10     5
cum%    30    50    70    85    95   100
%!NA 31.6 21.1 21.1 15.8 10.5
```

これでb1の，度数合計（Total），各カテゴリの度数と割合（NAを除いた%（%!NA）と，それを込みにした%（cum%），累積%（%）とともに作成してくれます。

ただし，このままだと，カテゴリを度数の多い順に勝手に並べ替えてしまいます（右図を見てください）。それを防ぐためには，2行目にdecr.order=FALSEを入れ，fr <-freq(x$b1, decr.order=FALSE)としておく必要があります。

なおこの2行の命令を読み解いてみると，freqで計算した結果を一度frという新しい入れ物に入れ，さらにprintでfrの出力調整をしているということになります。Rには，このように計算結果をいったん何かに入れ，結果表示コントロールを別に行って出力するという場合が少なくないようです。基本的に，これは結果を見やすくするためのようです。

ちなみに3行目のprintの部分をつけないと，Totalとcum%は出力されません。show.total=TRUEとcum.pc=TRUEが入っているところに注目すればすぐにピンとくるでしょう。

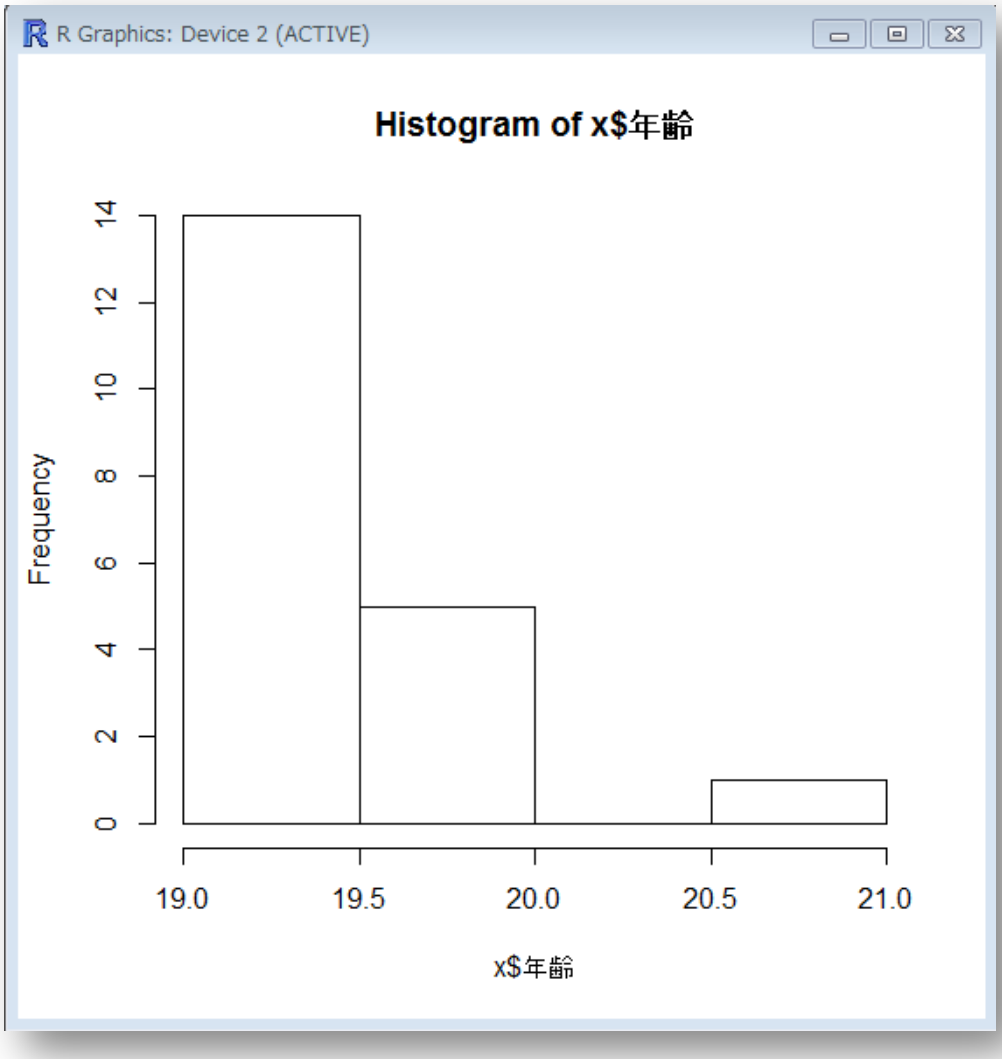
複数変数をまとめてやりたい時は，freq(x[2:5], decr.order=FALSE)といった指定でうまくいきます。またv3 <- c("性別","b2","b4","no")などと変数名をまとめておいて，freq(x[v3], decr.order=FALSE)で一気に計算することもできます。

では次に，ヒストグラムを作成してみます。繰り返しますが，ここでは項目チェックに使える簡単なものを目指します。

```
hist(x$年齢)
```

これだけを，実行してみてください。

これでどういう表示になるのか確認してください。



このように，Rで作図をしようとすると，図用の新しいウインドが開きます。通常のウインドと同様，縁にカーソルを合わせてドラッグすることで大きさを変えることができます(おそらく，デフォルトの大きさで特に問題ないかと思います)。

なお，いまは図表のタイトルが「Histogram of x$年齢」になっていますが，hist(x$年齢 , main="○○")と入力すれば，自由に図表のタイトルをつけることができます。

またこの「Device 2」というウインドですが，１枚の紙のようなイメージをもってもらえるとよいと思います。では，３回hist()を実行すると，どうなるでしょうか。ためしに，以下の命令を実行してみてください。

```
hist(x$b1)
hist(x$b2)
hist(x$b3)
```

「Device 2」のウインドには，最後のx$b3の結果だけが表示されていると思います。x$b1

やx$b2の結果はどこに行った？ということになるのですが，残念なことに上書きされてしまって見ることができません…。これはさすがに困りますね。

そこで，①windows()を使う，②par(mfrow=c(,))を使うという二通りの解決策を紹介します。

①windows()を使う

一度このコマンドをこのまま入力してみてください。すると，右のようなウインドが開くと思います。ここから分かるように，windows()というコマンドは，先に挙げた例にならうなら，白紙の紙を１枚用意するコマンドなのです。

そして，このコマンドを入力して新しいウインドを開いてからhist()を入力すれば，新しい方のウインドに図表が描かれます。

要は，「紙が１枚しかないせいで上書きするのだから，紙を増やそう」と考えるわけです。

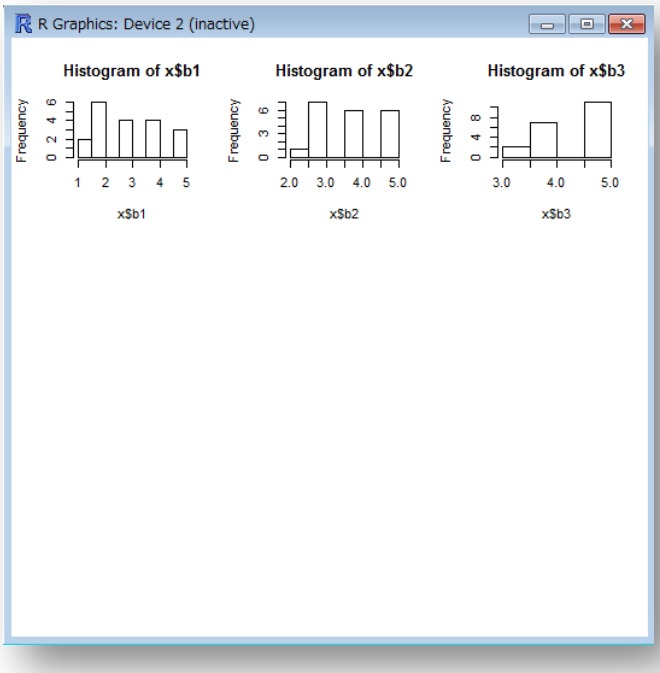
当然，２つ以上の図表を作るときは，その度ごとにwindows()と入力し，白紙のウインドを開く必要があります。

②par(mfrow=c(,))を使う

まず，括弧の中に数字を入れ，par(mfrow=c(3,3))と入力してみてください。もし既に描画ウインドが開いていれば何も反応は無く，描画ウインドが開いていなければ①のときと同じようなウインドが開くと思います。その状態でもう一度

```
hist(x$b1)
hist(x$b2)
hist(x$b3)
```

と入力してみてください。すると，右図のようになります。



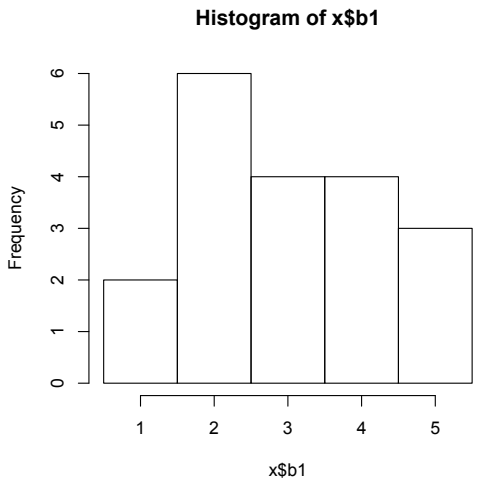
par(mfrow=c(,))を使うと，括弧の中の数字に応じて（今回は3×3）画面を分割し，図表を小さくして入れることができます。

ところが，この画面が図表でいっぱいになったときは，やはり上書きされてしまいます…。

また，出力された図表を右クリック→「メタファイルにコピー」もしくは「ビットマップにコピー」すれば，ワードやエクセルに貼り付けることができますし，また図表を保存することもできます（図表ウインドをアクティブにして，メニューバーから「ファイル」→「別名で保存」を選べば，保存形式を選ぶこともできます）

さて，`hist(x$年齢)`の部分ですが，変数さえ指定すれば（`x[,3]`という「,」付きなら列番号指定も可），適当にヒストグラムを作成してくれます。これに関するオプションの設定もたくさんあるので，自分の望むものを出力できるようにwebなどで調べてみましょう。

今回のように，項目単位での回答の分布を見るなら，ちょっと掟破りっぽいものでもいいのかなと思ったりします。b1からb5は1から5までの5件法なので，それぞれのカテゴリに何ケースあるのかがはっきりとわかる方がよいでしょう。それなら，`hist(x$b1, breaks=seq(0.5, 5.5, 1))`などと指定するのはどうかと思います。`breaks=seq`の後は，グラフの左端，右端，幅の3つの数値を入れます。ヒストグラムとしてどうか，ということは置いておきますが，1から5の選択肢の真上に棒の部分がのるため，私にとっては使いやすい（チェックしやすい）ものになります。



さて，あとはいくつかの変数をまとめて，一気にヒストグラムを作らせたいのですが，これがなかなか難しいようです。

いくつか試しましたが，ことごとくエラー…

```
> hist(x)
 以下にエラー hist.default(x) :  'x' は数値でなければなりません
> hist(x[2:4])
 以下にエラー hist.default(x[2:4]) :  'x' は数値でなければなりません
> v <- c("no","性別","年齢","b1","b2","b3","b4","b5")
> hist(x[v])
 以下にエラー hist.default(x[v]) :  'x' は数値でなければなりません
```

結局，このようなものに行き着きました…

```
for (a in 4:8)  { hist(x[,a]) }
```

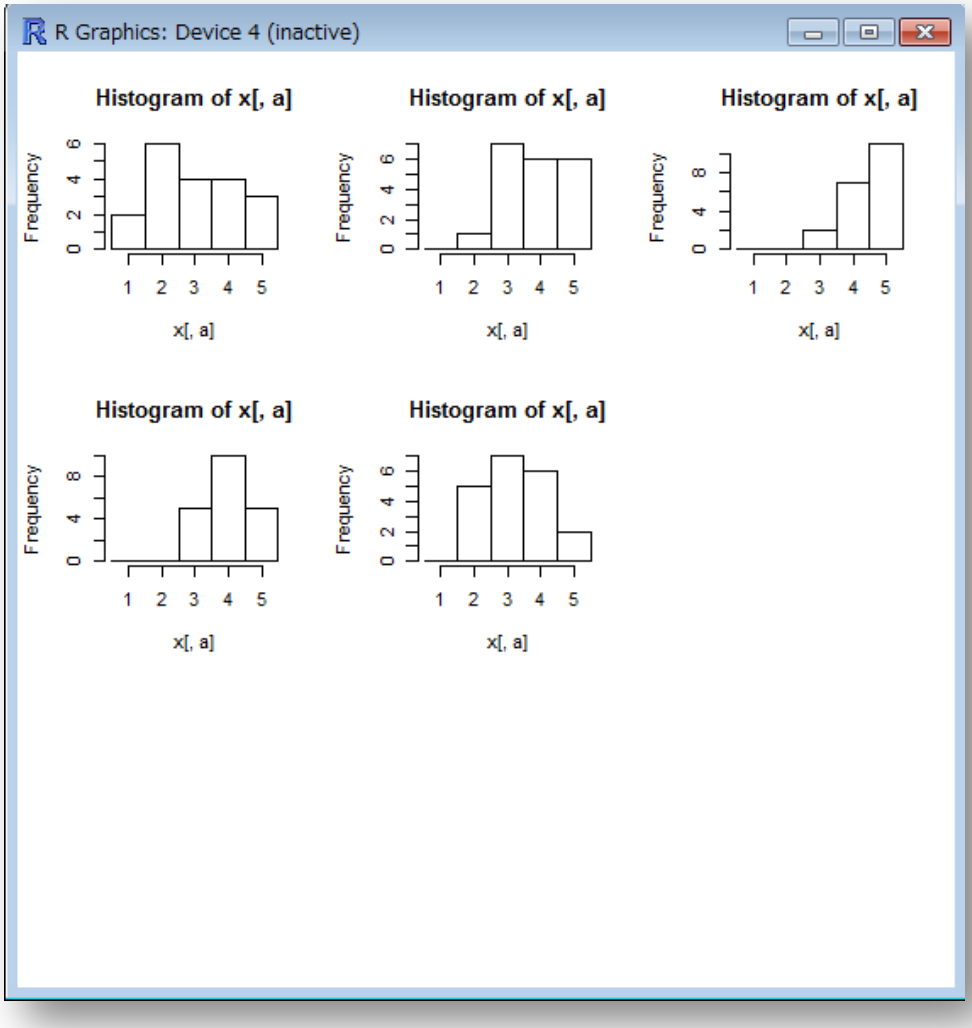
for (a in 4:8)は，aを４から８まで，１ずつ動かしなさいという命令です。そしてその命令を受けるのが{ }の中です。そこにhist(x[,a])を入れれば，aが先の命令を受けるので，hist(x[,4])， hist(x[,5])，… hist(x[,8])と順にやってくれるという仕組みです。

これを使って，b1からb5までのヒストグラムを１枚（３行３列）に収まるように出力したのが以下の例です。

```
par( mfrow=c(3,3))
for (a in 4:8)  { hist(x[,a], breaks=seq(0.5, 5.5, 1)) }
```

まあ，「これでもいいか…」くらいの出来です。問題はforを使うと次ページの結果のように変数名が表示されないこと。間違えないようにしないと…

これで９日目は終了です。明日は，相関係数を算出してみます。


R Graphics: Device 4 (inactive)
Histogram of x[, a]
Frequency
x[, a]
Histogram of x[, a]
Frequency
x[, a]
Histogram of x[, a]
Frequency
x[, a]
Histogram of x[, a]
Frequency
x[, a]
Histogram of x[, a]
Frequency
x[, a]