実習 –作図編–

飯島勇人

2008 年 4 月 14 日

目次

1 **重ねた図** 1
- 1.1 作業手順 . . . 2
- 1.2 怠惰の極み〜for による全自動化〜 . . . 3

2 **連続図** 4
- 2.1 図の数とレイアウト . . . 4
- 2.2 図の間の間隔の調整 . . . 5
- 2.3 複数の図に対するラベル . . . 5
- 2.4 その他 . . . 6
- 2.5 y 軸の変数だけが図間で異なる複数の図 . . . 6
- 2.6 x 軸も y 軸も同じ変数である複数の図 . . . 6

3 **曲線の描画** 7
- 3.1 成長予測式 . . . 7

4 **平面図** 8
- 4.1 平面を発生させる . . . 8
- 4.2 点の描画 . . . 9
- 4.3 線の描画 . . . 10
- 4.4 四角の描画 . . . 10
- 4.5 グリッドの挿入 . . . 10

1 重ねた図

図を重ねる（というか、同一図内で複数のカテゴリーを、色やシンボルの形を変えて表示させる）には、高水準作図で作図し、シンボルの書き分けを指定することで可能になります。

今回は、「個体重–幹重」という関係を、syori 間で比べる図を作ってみましょう。

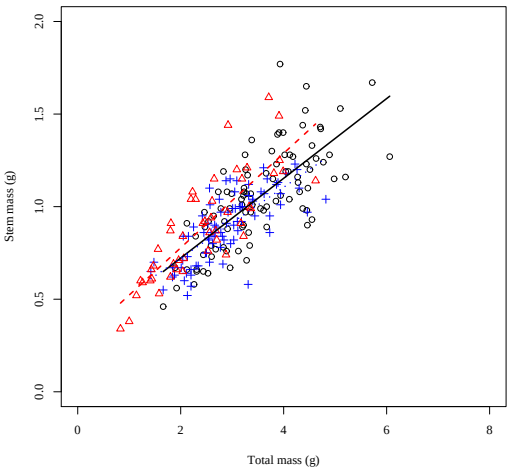

図 1　カテゴリー別の散布図

1.1　作業手順

1. x 軸と y 軸の値の範囲を決定しておく
2. 高水準作図を 1 つ描き、その引数で、シンボルの指定をする

1.1.1　値の範囲

いちいち各パラメータについて値の範囲を調べるのは面倒なので、`summary()` を使って楽しちゃいましょう。

```
> summary(d)
```

そして、x 軸と y 軸になる列に関し、最大値と最小値から、値の範囲を決めればいいわけです。

1.1.2　図の描画

```
> plot(miki ~ omosa, xlim=c(0, 7), ylim=c(0, 2),
+      pch=as.numeric(d$syori),
+      col=as.numeric(d$syori), d)
```

pch はシンボルの形を指定する低水準作図命令ですが、ここに、カテゴリーを指定する列を、`as.numeric()` で**数字として**投入することで、データをわざわざカテゴリーごとに分けなくてもシンボルごとに描画できます。

ちなみに、色を自分で指定したければ、このカテゴリーの場合はこの色、という指定を事前にしておけば可能です。今回は futsuu を黒、ijime を赤、zeitaku を青と指定します。

```
> iro <- c("black", "red", "blue") #カテゴリーのアルファベット順に指定します
> plot(miki ~ omosa, xlim=c(0, 7), ylim=c(0, 2),
+      pch=as.numeric(d$syori)-1,
+      col=iro[as.numeric(d$syori)], d)
```

iro[*] とすると、iro の*番目に入れた色が使われます。as.numeric() で各カテゴリーを数字化して指定しているわけですね。

1.1.3 回帰直線

せっかくカテゴリーごとに点を描いたので、カテゴリーごとに回帰直線を引いてみましょう。

```
> test <- split(d, d$syori)
> lm1 <- lm(miki ~ omosa, test[[1]])
> co <- lm1$coefficients
> curve(co[1] + co[2] * x, add=TRUE, from = min(test[[1]]$omosa),
+ to = max(test[[1]]$omosa))
> lm2 <- lm(miki ~ omosa, test[[2]])
> co <- lm2$coefficients
> curve(co[1] + co[2] * x, lty=2, col="red", add=TRUE,
+ from = min(test[[2]]$omosa), to = max(test[[2]]$omosa))
> lm3 <- lm(miki ~ omosa, test[[3]])
> co <- lm3$coefficients
> curve(co[1] + co[2] * x, lty=3, col="blue", add=TRUE,
+ from = min(test[[3]]$omosa), to = max(test[[3]]$omosa))
```

1.2 怠惰の極み〜for による全自動化〜

ちなみに、繰り返し命令（for()）を使うと、以上のプロセスが一度にできます。

```
> d <- read.csv("data.csv")
> test <- split(d, d$syori)
> iro <- c("black", "red", "blue")
> plot(miki ~ omosa, xlim=c(0, 8), ylim=c(0, 2),
+      xlab="Total mass (g)", ylab="Stem mass (g)",
+      pch=as.numeric(d$syori), col=iro[as.numeric(d$syori)], d)
```

```
> for (i in 1:3) {
+       reg <- lm(miki ~ omosa, test[[i]])
+       co <- reg$coefficients
+       curve(co[1] + co[2] * x, lty=i, col=iro[i], lwd=2, add=TRUE,
+       from = min(test[[i]]$omosa), to = max(test[[i]]$omosa))
+       }
```

2 連続図

よく、

1. x 軸は全部同じだけど、y 軸だけが異なる図が複数有り、それをきれいに並べて表示したい
2. x 軸も y 軸も同じだけど、（何でもいいんですけど）年度とか種とかが違う複数の図を書きたい

というケースがあると思います。ゑくせるでも頑張ればできます。ですが、ゑくせるで作業したことがある人は分かると思いますが、2 つ以上の図をきれいに並べるためには、マウスで微妙な大きさをあわせなければならないので、とても時間がかかります。やってられません。R なら、フォーマットを整えた図を作るのも簡単です。

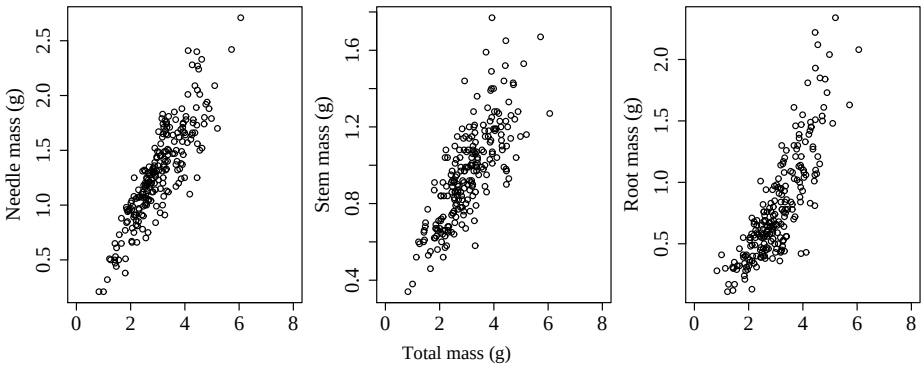

図 2　個別の散布図

2.1 図の数とレイアウト

まず、図を縦に何個、横に何個（何 × 何）で配置するかを決めましょう。mfrow=c(,) を使います。実際に使うときは、

```
> par(mfrow=c(縦に並べる図の数，横に並べる図の数))
```

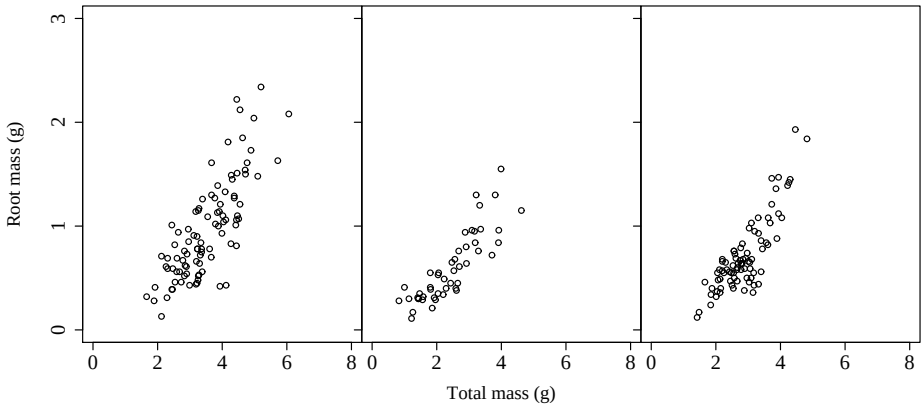

図 3　個別の散布図 2

というようにします。

2.2　図の間の間隔の調整

複数図を作る場合、図の間の間隔を調整する必要があります。

1. x 軸は全部同じだけど、y 軸だけが異なる図が複数有り、それをきれいに並べて表示したい → **縦方向は隙間なし、横方向はラベルが入るように少しあける**
2. x 軸も y 軸も同じだけど、年度とか種とかが違う複数の図を書きたい → **縦方向も横方向も隙間なし**（隙間を入れることもできますが）。

図の間の間隔を調整するには、`par(mar=c( , , , ))` を使います。`mar()` には、左から順に、図の下、左、上、右に、どれほどの隙間をとるか設定します。

2.3　複数の図に対するラベル

さらに、複数の図を描いた場合、x 軸や y 軸のラベルを、それぞれの図につけるのではなく、複数の図全体に対してラベルをつけたいことがあると思います。このためには、

- 複数の図の外に、x 軸と y 軸のラベル用のスペースを作る。
- 複数の図の x 軸と y 軸を描く。

というようにします。なぜこんなことをする必要があるかというと、例えば、描く図の数が 2 つの場合、各図の x 軸にラベルを設定すると、複数の図の真ん中にラベルを描くことができません。

まず、複数の図全体に関して、図の外の余白を設定します。これには、`oma()` を使います。`mar()` との違いは、

- mar(): 各図に関して、図の外の余白を設定する。
- oma(): **複数の図群に関して、**図群の外の余白を設定する。

ということです。

次に、複数の図を描いた後で、x 軸と y 軸を描きます。

```
> mtext(1, line=4, outer=T, text="x 軸のラベル")
> mtext(2, line=4, outer=T, text="y 軸のラベル")
```

mtext() の最初の数字は 1 が図の下（x 軸）、2 が図の左（y 軸）にラベルを描くことを、line の数字は、図からどれだけラベルを離して描くか、outer は、図の外にラベルを描くことを認めるか（T は TRUE）を示しています。

2.4 その他

複数の図を並べるときに、x 軸や y 軸のうち、図に共通な部分は、軸の値の間隔もそろえるのが一般的です。xlim=c(,) や ylim=c(,) を使ってそろえておきましょう。

2.5 y 軸の変数だけが図間で異なる複数の図

全データに関し、個体重に対して左から葉重、幹重、根重の関係の図を描いて見ましょう。

```
> d <- read.csv("data.csv")
> par(mfrow=c(1, 3))
> par(mar=c(0, 5, 0, 0), oma=c(5, 0, 1, 1), ps=20)
> summary(d) #値の範囲を決めるため
> plot(ha ~ omosa, xlim=c(0, 8), xlab="", ylab="Needle mass (g)", d)
> plot(miki ~ omosa, xlim=c(0, 8), xlab="", ylab="Stem mass (g)", d)
> plot(ne ~ omosa, xlim=c(0, 8), xlab="", ylab="Root mass (g)", d)
> mtext(1, line=3, outer=T, text="Total mass (g)", cex=0.6)
```

ちなみに、mtext は設定してあるフォントサイズよりも大きめになっているので、cex=0.6 することで、サイズをあわせることができます。

2.6 x 軸も y 軸も同じ変数である複数の図

syori ごとに、「個体重–根重」を描いてみます。

```
> d <- read.csv("data.csv")
> test <- split(d, d$syori)
> par(mfrow=c(1, 3), mar=c(0, 0, 0, 0), oma=c(6, 6, 1, 1), ps=20)
```

```
> summary(d) #値の範囲を決めるため
> for (i in 1:3) {
+       plot(ne ~ omosa, xlim=c(0, 8), ylim=c(0, 3),
+       xlab="", ylab="", yaxt="n", test[[i]])
+       if (i == 1) { axis(2, at=0:3) }
+       else { axis(2, at=0:3, label=FALSE) }
+       }
> mtext(1, line=3, outer=T, text="Total mass (g)", cex=0.6)
> mtext(2, line=4, outer=T, text="Root mass (g)", cex=0.6)
```

3 曲線の描画

ゑくせるでは、すでに決まっている種類の直線しか引くことができませんが、R では、式の形が決まっていれば、どんな直線、曲線も描くことができます。

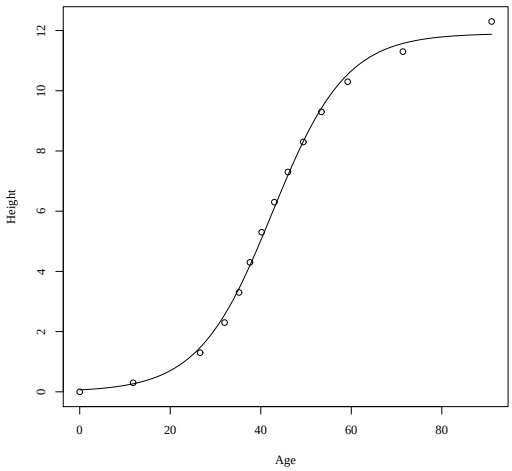

図 4　樹齢–樹高曲線（ロジスティック回帰）

3.1 成長予測式

樹齢と樹高に関する以下のようなデータセットを図にしてみます。

```
> Height <- c(0, 0.3:12.3)
> Age <- c(0, 11.8, 26.6, 32, 35.2, 37.6, 40.2, 43, 46, 49.4, 53.4, 59.2, 71.4, 91)
> d <- data.frame(Height, Age)
```

```
> plot(Height ~ Age, d)
```

樹齢と樹高の関係は、古くから成長予測式と呼ばれる式で近似できるとされてきました。今回は、ロジスティック曲線を使って近似し、その結果を図に描いてみます。

```
> result <- nls(Height ~ a/(1+b*exp(-c*Age)), start=c(a=10, b=100, c=0.5), d)
#さりげなく非線形回帰してます＾＾；)。これでパラメータ推定をしてるわけですね。
> x <- 0:91
> lines(x, 11.9/(1+186.4*exp(-0.12*x)))
```

curve() を使ってもいいのですが、今回は lines() を使ってみます。x には x 軸のデータ範囲を、そして予測式をその隣に書くことで、描画することができます。predict() を使う場合は、

```
> y <- predict(result, list(Age=x))
> lines(x, y)
```

4 平面図

広域のフィールド調査を行っていると、平面上にデータを描画したいケースが結構あると思います。ゑくせるでは点を平面状に描画するぐらいが関の山ですが、R を使えば、もう少し複雑な図が描けます。

今回は、以下のようなデータフレームを使います。せっかくなので、R で作ってみましょう。ヒグマの行動データです。GPS で各個体に関して 20 点の位置データが落ちています。

```
> ID <- rep(c("A1", "A2", "B1", "B2", "C1"), 20)
> Age <- rep(c("Adult", "Child", "Adult", "Child", "Adult"), 20)
> X <- rnorm(100, 50, 20)
> Y <- rnorm(100, 50, 20)
> Mass <- rep(c(200, 70, 150, 50, 170), 20)
> d <- data.frame(ID, Age, X, Y, Mass)
```

4.1 平面を発生させる

plot() と box() を使います。

- plot() で、座標の範囲を決めます。
- box() で、外枠を描きます。

例えば、

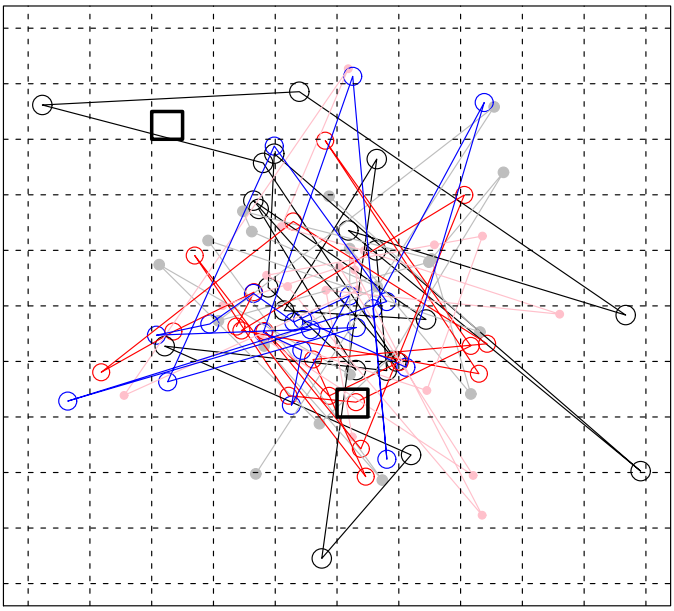

図 5　平面図

```
> plot(1, xlim=c(0, 100), ylim=c(0, 100), type="n", ann=F, axes=F) #x、y 座標の
値の範囲を決定
> box() #外枠を描画
```

とすることで、x 軸と y 軸それぞれ 0 から 100 までの範囲の箱を描きます。

4.2　点の描画

points() を使います。各個体のデータを落としてみましょう。

```
> iro <- c("black", "gray", "red", "pink", "blue")
> points(d$X, d$Y, col=iro, cex=log(d$Mass)-3, pch=rep(c(1,16,1,16,1), 20))
```

ちなみに、シンボルの拡大率を示す cex に、個体サイズをあてがうことで、シンボルのサイズに個体サイズを反映させることができます。また、シンボルの形そのものは、pch で変更できます。

4.3 線の描画

lines() を使います。各個体の移動の軌跡を、線で表現してみましょう。

```
> d2 <- split(d, d$ID)
> for (i in 1:5) {
+       lines(d2[[i]]$X, d2[[i]]$Y, col=iro[i])
+       }
```

4.4 四角の描画

rect() を使います。rect(, , ,) では、描く四角の左下の x 座標, y 座標, 右上の x 座標, y 座標という順で数字を入力します。例えば、冬眠穴が発見された場所を四角で囲むとすると

```
> rect(50, 30, 55, 35, lwd=3)
> rect(20, 80, 25, 85, lwd=3)
```

というようにします。

4.5 グリッドの挿入

いくつかありますが、ここでは abline を使ってみます。abline は直線を描く関数ですが、引数に h と v を指定することで、規則的なグリッドを描くことができます。

```
> abline(h=0:10*10, v=0:10*10, lty=2) #10 ごとに格子を発生させる。
```

h は horizontal、v は vertical です。