

R と非線形回帰分析

1. 非線形回帰分析とは

線形モデルは被説明変数を説明変数の線形関数で表現するモデルである。本稿では、線形モデル以外のモデルを非線形モデルと呼ぶことにする。

非線形回帰分析には、説明変数や被説明変数に何らかの変換を施し、線形関係に置き換え線形回帰分析を行う方法、非線形の関数を当てはめる方法、加法モデル、樹木モデル、ニューラルネットワーク法など多数の方法が提案されている。

2. 非線形回帰モデルと関数 nls

非線形回帰分析の方法の最も基本的な方法は、非線形の関数式を用いてデータを当てはめる方法である。

Rには、自由に関数式を指定することができる非線形回帰分析の関数 `nls` がある。関数 `nls` の簡単な書き方を次に示す。

```
nls(formula,data,start,trace)
```

引数 `formula` の書き式は、関数 `lm` や `glm` と若干異なる。関数 `lm` と `glm` では非説明変数と説明変数のみを指定すればよかったが、関数 `nls` では被説明関数と説明変数との関係式を具体的に書く必要がある。

例えば、データに関数 $y = \frac{a}{1 + be^{cx}}$ を当てはめる非線形回帰分析の場合は `formula` の書き式は次のように関係式を書く。

```
y~ a/(1+b*exp(c*x))
```

この `formula` 中の `a`、`b`、`c` が求める回帰係数(パラメータ)である。

関数 `nls` では、線形回帰分析の場合と同じく、被説明変数の実測値と予測値(あるいは推測値)との差を最小とする最小2乗法で係数(パラメータ)を求めている。

関数 `nls` では、パラメータの名前とその初期値を指定する必要がある。初期値の指定は、引数 `start` で設定した方がよい。上記の `formula` に対応する `start` の記述例を次に示す。

```
nls(y~ a/(1+b*exp(c*x)),start=c(a=1,b=1,c=-1))
```

初期値は、データ解析者の勘と経験に頼るのが殆どである。

`trace` はパラメータの計算の過程を返すか否かを指定する引数である。計算過程の結果を返す場合は、`TRUE`(あるいは `T`)を指定する。デフォルトは `FALSE` になっている。

非線形回帰分析は、線形回帰分析より計算が難しいため、係数パラメータを求める際に、数値解がうまく求められず、プログラム実行が失敗する 경우가しばしばある。その際には、モデルを換えて試行錯誤しなければならない。

ここでは一般化線モデルを説明する際に用いた日本におけるカラーテレビの普及率(先月の号を参照)の例を用いて非線形回帰関数 nls の使用方法を説明する。

```
>年度<-c(1966:1984)
>普及率<-c(0.003,0.016,0.054,0.139,0.263,0.423,0.611,0.758,0.859,0.903,0.937,0.954,0.978,0.978,
0.982,0.985,0.989,0.988,0.992)
```

このような普及率、成長率のデータは、通常ロジスティック関数が多く用いられている。ここでは、次に示すロジスティック関数を用いてみよう。

$$y = \frac{a}{1 + be^{cx}}$$

上記に説明した手順で、次のように指定して実行するとエラーメッセージが返される。

```
>fm<-nls(普及率 ~ a/(1+b*exp(c*年度)),start=c(a=1,b=1,c=-1),trace = TRUE)
```

これは、関係式の中の exp(年度)が非常に大きな値になってしまったのが原因である。そこで、説明変数 1966 ~ 1984 を 1 ~ 19 に置き換えて実行してみよう。

```
>fm<-nls(普及率~ a/(1+b*exp(c*1:19)), start=c(a=1,b=1,c=-1),trace = TRUE)
```

```
3.905671 :          1  1 -1
2.387674 :          0.9824052  0.4300442 -0.1029666
1.743185 :          0.8872618  0.8264732 -0.2623701
0.7740847 :          0.9841109  2.3123040 -0.2310466
0.5578214 :          0.9271411  7.5149748 -0.5270324
0.09229084 :          1.0001338 17.1134434 -0.4350728
0.06874581 :          0.9606817 40.3886048 -0.6606493
0.01653944 :          0.9826601 75.9510234 -0.7160221
0.003486704 :          0.9806949 110.6878664 -0.7509771
0.001959816 :          0.9804580 123.8500804 -0.7565368
0.001949770 :          0.9806034 123.8048180 -0.7553703
0.001949752 :          0.9806268 123.6621023 -0.7551686
0.001949752 :          0.9806279 123.6609507 -0.7551648
```

このようにパラメータの計算過程が返され、パラメータ a、b、c の最終推定値はそれぞれ 0.9807489、123.54058867、-0.7549528 となっている。関数 summary で回帰の要約が返される。

```
>summary(fm)
```

```
Formula: 普及率 ~ a/(1 + b * exp(c * 1:19))
```

```
Parameters:
```

```
Estimate Std. Error t value Pr(>|t|)
```

```
a 0.98063 0.00384 255.401 < 2e-16 ***
```

```
b 123.66095 13.56739 9.115 9.82e-08 ***
```

```

c -0.75516    0.01742 -43.347 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.01104 on 16 degrees of freedom
Correlation of Parameter Estimates:
      a      b
b -0.2950
c  0.3967 -0.9709

```

予測値は関数 `lm` の場合と同じく関数 `fitted` で返される。

次のコマンドで、実測値と非線形回帰関数 `nls` による予測値をグラフで示す。図 1 から、予測値は実測値に非常によく近似していることが分かる。

```

>plot(年度,普及率,cex=2)
>lines(年度,普及率)
>lines(年度,fitted(fm),col=2,lty=2,lwd=2)
>legend(1975,0.5,c("実測値","予測値"), col=1:2,lty=1:2,lwd=2)

```

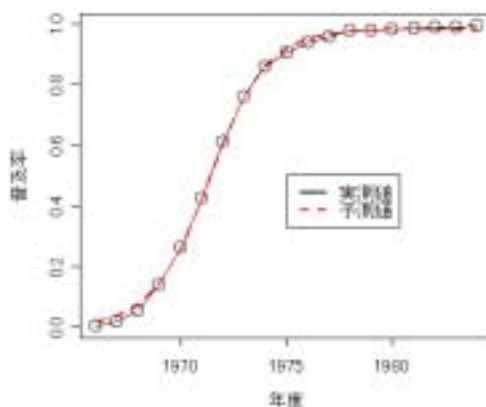


図 1 テレビの普及率の実測値と予測値

残差は関数 `resid` で返され、関数 `plot` で予測値対標準化された残差の散布図が作成される。

Rには頻繁に使用されている関数式が用意されているものもある。関数 `nls` では、その関数式をリンクして、非線形回帰分析を行うことができる。用意されている関数をリンクして回帰分析を行う際には、その関数の具体的な関数式を記述する必要がなく、関数のオブジェクトの名前を引数として用いる。例えば、ロジスティック関数のオブジェクトの名前は **SSlogis**、ワイブール関数は **SSweibull** となっている。

カラーテレビの普及率のデータをロジスティック関数 `SSlogis` を用いる場合は、次のような書き式で非線形回帰分析ができる。

```

>fm1<-nls(普及率~SSlogis(1:19, Asym, xmid, scal ),list( Asym = 1, xmid = 0, scal = 1 ),trace = TRUE)

```

上記の具体的な関係式を記述して求めた予測値とロジスティック関数 `SSlogis` を用いて求め

た予測値を次に示す。異なる両方法で得られた予測値は非常によく近似している。

```
> data.frame(fitted(fm),fitted(fm1))
  fitted.fm  fitted.fm1
1  0.01660369 0.01660370
2  0.03466288 0.03466289
< 中略 >
18 0.98059694 0.98059694
19 0.98067744 0.98067745
```

関数 nls による非線形回帰分析の説明のため、人工データを用いた例を次に示す。まず次のような人工データを作成し、そのデータの散布図を図 2 に示す。

```
>x<-seq(-5, 5,0.1)
>y<-10*x^3+100*norm(x,0,1)
>plot(x,y)
```

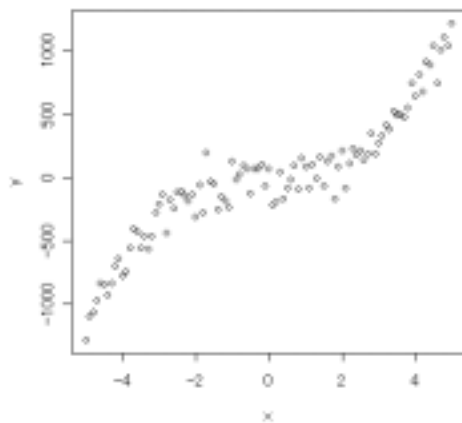


図 2 3 次多項式の人口データ

データの散布図から縦軸 y と横軸 x の関係は 3 次多項式の関係であることが分かる。

3 次多項式の一般式は $y = a + bx + cx^2 + dx^3$

である。この式を用いた nls は次のように記述する。

```
>fm3=nls(y~a+b*x+c*x^2+d*x^3,start=c(a=1,b=1,c=1,d=1),trace=T)
20490809 : 1 1 1 1
1089805 : -20.576319 24.692240 1.206357 8.678613
> summary(fm3)
Formula: y ~ a + b * x + c * x^2 + d * x^3
Parameters:
  Estimate Std. Error t value Pr(>|t|)
a -20.5763 15.8218 -1.301 0.19651
b 24.6922 9.0471 2.729 0.00754 **
c 1.2064 1.3875 0.869 0.38674
```

```

d 8.6786 0.5421 16.011 < 2e-16 ***
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 106 on 97 degrees of freedom
Correlation of Parameter Estimates:
      a      b      c
b 8.778e-09
c -7.454e-01 -3.082e-09
d -2.804e-08 -9.166e-01 2.545e-08

```

返された結果に基づいた多項式回帰モデルを次に示す。

読者が同じ方法で上記の操作を行っても本稿と同じ結果が得られないのは、間違いではなく、人工データを作成する際に正規乱数を生成して用いたためである。

$$y = -20.576 + 24.692x + 1.206x^2 + 8.679x^3$$

統計モデルを作成する際には、パラメータの数を少なくするのが良い。そこで、簡潔な 3 次式 $y = a + bx^3$ を用いて同じデータについて非線形回帰分析を行ってみよう。

```

>(fm4=nls(y~a+b*x^3,start=c(a=1,b=1),trace=T))
20715546 : 1 1
1181990 : -10.32228 10.03462
Nonlinear regression model
 model: y ~ a + b * x^3
 data: parent.frame()
      a      b
-10.32228 10.03462
residual sum-of-squares: 1181990

```

返された結果のモデルは $y = -10.322 + 10.035x^3$ となる。この異なる 2 つの多項式モデルによる AIC 値を次に示す。

```

> AIC(fm3)
[1] 1232.551
> AIC(fm4)[1]
[1] 1236.752

```

AIC の値には大きな差がないことから、両モデルによる当てはめの良さには明らかな差がないことが分かる。

視覚的に考察するため、両モデルによる予測値を図 2 に加えるコマンドとその結果を図 3 に示す。

```

>plot(x,y)
>lines(x,fitted(fm3),lty=1,col=2)

```

```
>lines(x,fitted(fm4),lty=2,col=4,lwd=2)
>legend(1,-700,c("一般式","簡潔式"), lty=1:2,col=c(2,4),lwd=2)
```

図 3 から分かるように、両モデルの予測値には差がほとんど見られない。

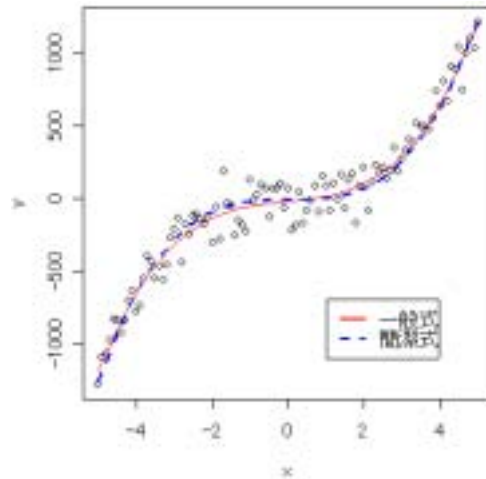


図 3 人工データの実値と両モデルの予測値

初期値は大まかなものであるので、多少異なってもプログラムは作動する。初期値の設定に左右されるのは計算量である。

このような多項式による回帰分析は、線形回帰関数 `lm` を次のように記述することで等価の回帰モデルを求めることができる。

```
lm(y~poly(x,3))
```

上記の書き式の中の `poly(x,3)` は、3 次多項式(Polynomial)のリンクに関する記述である。

3. その他の非線形回帰モデル

(1) 平滑化

関数 `lm`、`glm`、`nls` が対応できない、より複雑なデータにおける回帰モデルの説明のため、次の人工データを用いる。

```
>x1=seq(-10,10,0.1)
>y1=50*sin(x1)+x1^2+10*rnorm(length(x1),0,1)
>plot(x1,y1)
```

上記のコマンドで生成された `x1` を横軸、`y1` を縦軸とした散布図を図 4 に示す。データ `y1` を被説明変数、`x1` を説明変数とした非線形回帰モデルを考える場合、散布図から思い付くのは多項式回帰モデルであろう。

次の 3 種類の多項式回帰モデルのコマンドとその回帰モデルによる予測値を散布図に加えるコマンドを示す。

```

>lm.p2<-fitted(lm(y1~poly(x1,2)))
>lines(x1,lm.p2,lty=4,col=4,lwd=2)
#2 次多項式モデルの予測値
>lm.p5<-fitted(lm(y1~poly(x1,5)))
>lines(x1,lm.p5,lty=2,col=2,lwd=2)
#5 次多項式モデルの予測値
>lm.p8<-fitted(lm(y1~poly(x1,8)))
>lines(x1,lm.p8)
#8 次多項式モデルの予測値
>legend(-2.5,140,c("poly(x1,2)","poly(x1,5)","poly(x1,8)"),col=c(4,2,1),lty=c(4,2,1),lwd=2)
#凡例の表示

```

図 4 から分かるように、このデータにおいては、多項式によるモデルの当てはめの精度が良いとは言いがたい。

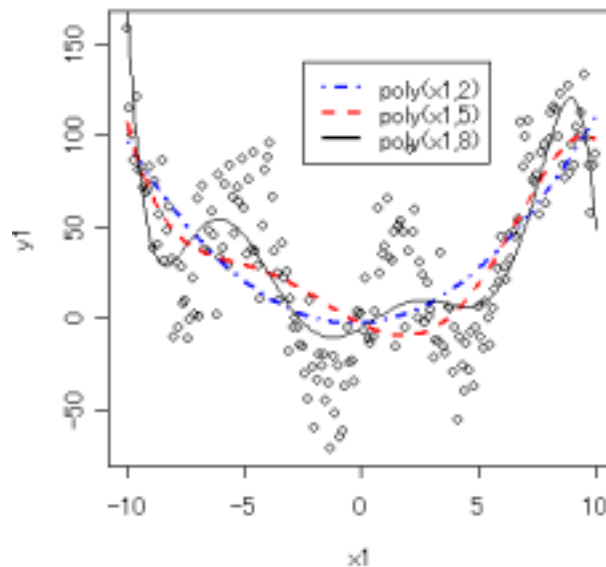


図 4 3 種類の多項式回帰モデルの予測値

このように複雑に変化する被説明変数を推定する方法として**平滑化回帰**の方法が提案されている。平滑化回帰を詳しく説明する紙面がないので、ここでは R で提供している関数 **smooth.spline**、**ksmooth**、**supsmu**、**lowess** を用いた平滑化された回帰曲線を図 5 に示す。

```

>par(mfrow=c(2,2),oma=c(2,2,2,2),mar=c(2,2,2,2))
#1 つの画面に 4 コマのグラフを配置
>plot(x1,y1,main="関数 smooth.spline による平滑結果")
>lines(smooth.spline(x1,y1),col=2,lwd=2)
#smooth.spline による平滑化
>plot(x1,y1,main="関数 ksmooth による平滑結果")
>lines(ksmooth(x1,y1),col=4,lwd=2)
#ksmooth による平滑化
>plot(x1,y1,main="関数 supsmu による平滑結果")
>lines(supsmu(x1,y1),col=3,lwd=2)
#supsmu による平滑化
>plot(x1,y1,main="関数 lowess による平滑結果")

```

```
>lines(lowess(x1,y1,f=0.1),col=5,lwd=2)
#lowess による平滑化
```

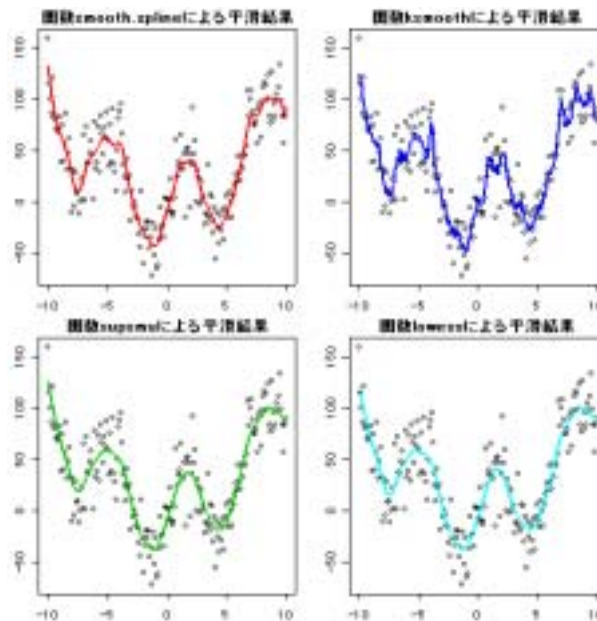


図 5 平滑化の例

パッケージ `fields` 中の関数 `sreg` を用いると平滑化スプライン(smoothing spline)回帰を行うことができる。

(2) 加法モデル

加法回帰モデル(additive regression model)は、次のような数式で表されるモデルである。

$$y = a + f_1(x_1) + f_2(x_2) + \dots + f_i(x_i) + \dots + f_n(x_n) + \varepsilon$$

式の中の y は被説明変数、 x_i は説明変数、 f_i は説明変数の変換を行う関数、 ε は残差である。

線形回帰モデル、`nls` による非線形回帰モデルは加法モデルの特殊なケースと考えられる。 f_i が全て線形関数の場合は、線形回帰モデルとなる。

一般化線形モデルを加法モデル化したとき、**一般化加法モデル**(GAM: generalized additive model)と呼ぶ。

R のパッケージ `mgcv` に一般化加法モデルの関数 `gam` がある。関数 `gam` では、それぞれの説明変数を、平滑化関数をリンクして用いることもできる。

データ `airquality` を用いてその使用法を例示する。ここでは、用いる全ての説明変数について平滑化スプラインをリンクすることにする。平滑化スプライン `smooth.spline` の略記号は `s` である。次にその書き式を示す。加法回帰モデルの要約は `summary` で返される。

```
> library(mgcv); data(airquality)
> airq.gam <- gam(Ozone ~ s(Solar.R) + s(Wind) + s(Temp), data = airquality)
> summary(airq.gam)
Family: gaussian
Link function: identity
```

Formula:

Ozone ~ s(Solar.R) + s(Wind) + s(Temp)

Parametric coefficients:

	Estimate	std. err.	t ratio	Pr(> t)
(Intercept)	42.099	1.663	25.32	< 2.22e-16

Approximate significance of smooth terms:

	edf	chi.sq	p-value
s(Solar.R)	2.76	13.7	0.0038243
s(Wind)	2.91	50.559	5.3379e-09
s(Temp)	3.833	55.901	3.4155e-09

R-sq.(adj) = 0.723 Deviance explained = 74.7%

GCV score = 338.9 Scale est. = 306.83 n = 111

説明変数を平滑化スプライン関数にリンクした加法モデルの自由度調整済みの重相関係数は 0.73 である。

同じの説明変数を用いた関数 `lm` による線形回帰モデルの自由度調整済みの重相関係数は 0.6059 である(先月号を参照)ので、上記の加法モデルが線形回帰モデルより当てはめが良いと言えよう。

(3) その他の方法

上記の説明以外にも多くの回帰方法が提案されている。その全てを詳細に説明する余裕がないので、**局所回帰モデル**、**射影追跡回帰モデル**、**樹木モデル**、**ニューラルネットワークモデル** について簡潔に紹介する。

局所回帰モデル(local regression model)は、局所的にデータに回帰関数、あるいは回帰曲面を当てはめる 1 つの方法である。平滑化関数 `lowess` は局所多項式を用いている。R には局所回帰関数 `loess` がパッケージ `stats` の中に含まれている。関数 `loess` は局所多項式による回帰モデルである。

射影追跡回帰モデル(projection pursuit regression model)は、スタンフォード大学の Friedman らによって 1980 年代に提案された、射影した説明変数を加法モデルに適用する方法である。R のパッケージ `MASS` に射影追跡回帰関数 `ppr` がある。

樹木モデル(tree-based model)は、社会科学、機械学習の分野で提案された決定木(decision trees)モデルである。統計学で決定木に対して注目し始めたのは、1980 年代からである。R には樹木モデルのパッケージ `tree`、`rpart`、`mvpart` がある。

ニューラルネットワークモデル(neural networks mode)は人工知能の分野で発展してきたニューロンをモデル化したものであり、多くの方法が提案されているが、回帰モデルに用いるのは階層型ニューラルネットワークである。R には、階層型ニューラルネットワークのパッケージ `nnet` がある。

これらの回帰モデルに関する参考書として下記の参考文献がある。

参考文献

- [1] S-PLUS によるデータ解析、伊藤幹夫・その他 訳、シュプリンガーフェアラーク東京。
- [2] S と統計モデル - データ科学の新しい波 - 、柴田里程 訳、共立出版株式会社。
- [3] 平滑化とノンパラメトリック回帰への招待、竹澤邦夫・大森宏 訳、(財)農林統計協会。
- [4] みんなのためのノンパラメトリック回帰(上)(下)、竹澤邦夫 著、吉岡書店。