nlmrt-vignette

John C. Nash

March 26, 2012

Background

This vignette discusses the (at time of writing **experimental** R package `nlmrt`, that aims to provide computationally robust tools for nonlinear least squares problems. Note that R already has the `nls()` function to solve nonlinear least squares problems, and this function has a large repertoire of tools for such problems. However, it is specifically NOT indicated for problems where the residuals are small or zero. Furthermore, it frequently fails to find a solution if starting parameters are provided that are not close enough to a solution. The tools of `nlmrt` are very much intended to cope with both these issues.

`nlmrt` tools generally do not return the large nls-style object. However, we do provide a tool `wrapnls` that will run either `nlsmnq` for unconstrained problesm or `nlsmnqb` for bounds constrained problems followed by an appropriate call to `nls`.

1 An example problem

Let us try an example initially presented by [**?**] and developed by [**?**]. This is a model for the regrowth of pasture. We set up the computation by putting the data for the problem in a data frame, and specifying the formula for the model. This can be as a formula object, but I have found that saving it as a character string seems to give fewer difficulties. Note the " " that implies "is modeled by". There must be such an element in the formula for this package (and for `nls()`). We also specify two sets of starting parameters, that is, the `ones` which is a trivial (but possibly unsuitable) start with all parameters set to 1, and `huetstart` which was suggested in [**?**]. Finally we load the routines in the package `nlmrt`.

```
> options(width=60)
> pastured <- data.frame(
+ time=c(9, 14, 21, 28, 42, 57, 63, 70, 79),
+ yield= c(8.93, 10.8, 18.59, 22.33, 39.35,
+          56.11, 61.73, 64.62, 67.08))
> regmod<-"yield ~ t1 - t2*exp(-exp(t3+t4*log(time)))"
```

```
> ones<-c(t1=1, t2=1, t3=1, t4=1) # all ones start
> huetstart<-c(t1=70, t2=60, t3=0, t4=1)
> require(nlmrt)
```

Let us now call the routine `nlsmnqb` (even though we are not specifying bounds). We try both starts.

```
> anmrt<-nlsmnqb(regmod, start=ones, trace=FALSE, data=pastured)
'data.frame':        9 obs. of  2 variables:
 $ time : num  9 14 21 28 42 57 63 70 79
 $ yield: num  8.93 10.8 18.59 22.33 39.35 ...
NULL
$watch
[1] FALSE

$phi
[1] 1

$lamda
[1] 1e-04

$offset
[1] 100

$laminc
[1] 10

$lamdec
[1] 4

$femax
[1] 10000

$jemax
[1] 5000
> print(anmrt)
$resid
[1]  0.48069948  0.66930970 -2.28432650  0.84373846
[5]  0.73457526  0.06655466 -0.98580893 -0.02505846
[9]  0.50031634

$jacobian
      t1          t2        t3        t4
 [1,]  1 -0.98156716  1.126420  2.474999
 [2,]  1 -0.94819229  3.111329  8.210975
 [3,]  1 -0.86978356  7.484690 22.787306
```

```
 [4,]  1 -0.75843621 12.934908 43.101760
 [5,]  1 -0.48427212 21.659422 80.955765
 [6,]  1 -0.22338362 20.652294 83.498282
 [7,]  1 -0.14933159 17.515486 72.569018
 [8,]  1 -0.08690194 13.094925 55.633728
 [9,]  1 -0.03850206  7.735031 33.797814

$feval
[1] 76

$jeval
[1] 50

$coeffs
[1] 69.955179 61.681444 -9.208935  2.377819

$ssquares
[1] 8.375884


> anmrtx<-try(nlsmnqb(regmod, start=huetstart, trace=FALSE, data=pastured))
'data.frame':        9 obs. of  2 variables:
 $ time : num  9 14 21 28 42 57 63 70 79
 $ yield: num  8.93 10.8 18.59 22.33 39.35 ...
NULL
$watch
[1] FALSE

$phi
[1] 1

$lamda
[1] 1e-04

$offset
[1] 100

$laminc
[1] 10

$lamdec
[1] 4

$femax
[1] 10000
```

```
$jemax
[1] 5000
> print(strwrap(anmrtx))
 [1] "c(0.480699476110992, 0.669309701586503,"
 [2] "-2.28432650017661, 0.843738460841614,"
 [3] "0.734575256138093, 0.0665546618861583,"
 [4] "-0.985808933151056, -0.0250584603521418,"
 [5] "0.500316337120296)"
 [6] "c(1, 1, 1, 1, 1, 1, 1, 1, 1, -0.981567160420883,"
 [7] "-0.948192289406167, -0.869783557170751,"
 [8] "-0.758436212560273, -0.484272123696113,"
 [9] "-0.223383622127412, -0.149331587423979,"
[10] "-0.0869019449646661, -0.0385020596618461,"
[11] "1.12642043233262, 3.11132895498809, 7.48468988716119,"
[12] "12.9349083313689, 21.6594224095687, 20.652293670436,"
[13] "17.51548586967, 13.0949252904654, 7.73503096811733,"
[14] "2.47499865833493, 8.2109754835055, 22.7873063008638,"
[15] "43.1017598804902, 80.9557650898109, 83.4982821079476,"
[16] "72.56901775625, 55.6337277915341,"
[17] "61"
[18] "39"
[19] "c(69.9551789601637, 61.6814436396711,"
[20] "-9.20893535565824, 2.37781880027694)"
[21] "8.37588355893792"
```

Note that the standard `nls()` of R fails to find a solution from either start.

```
> cat("try regular\n")
try regular
> anls<-try(nls(regmod, start=ones, trace=TRUE, data=pastured))
> print(anls)
[1] "Error in nlsModel(formula, mf, start, wts) : \n  singular gradient matrix at initial
attr(,"class")
[1] "try-error"
attr(,"condition")
<simpleError in nlsModel(formula, mf, start, wts): singular gradient matrix at initial par


> anls<-try(nls(regmod, start=ones, trace=TRUE, data=pastured))
> print(anls)
[1] "Error in nlsModel(formula, mf, start, wts) : \n  singular gradient matrix at initial parameter estimates\n"
attr(,"class")
[1] "try-error"
attr(,"condition")
<simpleError in nlsModel(formula, mf, start, wts): singular gradient matrix at initial parameter estimates>
```

```
  > anls<-try(nls(regmod, start=ones, trace=TRUE, data=pastured))
  > print(strwrap(anls))
[1] "Error in nlsModel(formula, mf, start, wts) : singular"
[2] "gradient matrix at initial parameter estimates"
  > cat(strwrap(anls), sep='\n')
Error in nlsModel(formula, mf, start, wts) : singular
gradient matrix at initial parameter estimates
  > anlsx<-try(nls(regmod, start=huetstart, trace=TRUE, data=pastured))
13386.91 :  70 60  0  1
  > # capture.output(print(anlsx), file="anlsxout.txt")
  >
```

We were unable to install the INRA package `nls2` (there is a very different package by the same name on CRAN by Gabor Grothendieck).