Understaning nlsModel in the base R nls() function

Arkajyoti Bhattacharjee, Indian Institute of Technology, Kanpur
John C. Nash, University of Ottawa, Canada

05/06/2021

nlsModel()

- created script `tnlsModel.R` as first step to create and produce an "m" object
- Current understanding (JN): nlsModel (and nlsModel.plinear too probably) creates an R object that it labels as class "nlsModel." This object contains functions that are called from nls.c::nls_iter to run the interation and estimate the parameters in the model. There seem to be some extraneous functions, and we can hopefully learn enough to remove the extras.
- The current structure is to particularize the functions in "m" so the (essentially) external nls.c code acts on these. As a first goal, and part of learning how things work, we will want to replace the nls.c::nls_iter with all-R equivalent.

A script to examine the output of nlsModel()

We will use the Hobbs weed infestation problem (Nash (1979), page 120) again. ?? AB: we should build a set of test problems that are easy to try out. Let us discuss the examples that are in the help for nls. (?nls in R will show them.)

```r
# Data for Hobbs problem
ydat  <-  c(5.308, 7.24, 9.638, 12.866, 17.069, 23.192, 31.443,
            38.558, 50.156, 62.948, 75.995, 91.972) # for testing
tdat  <-  seq_along(ydat) # for testing

# A simple starting vector -- must have named parameters for nlxb, nls, wrapnlsr.
start1  <-  c(b1=1, b2=1, b3=1)
eunsc  <-   y ~ b1/(1+b2*exp(-b3*tt))
str(eunsc)
```

```
## Class 'formula'  language y ~ b1/(1 + b2 * exp(-b3 * tt))
##   ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
```

```r
# Can we convert a string form of this "model" to a formula
ceunsc <- " y ~ b1/(1+b2*exp(-b3*tt))"
str(ceunsc)
```

```
##  chr " y ~ b1/(1+b2*exp(-b3*tt))"
```

```r
weeddata1  <-  data.frame(y=ydat, tt=tdat)

## Now ready to try things out.
library(nlsalt) # ?? needed because base R does not export nlsModel()
```

```
## Registered S3 methods overwritten by 'nlsalt':
##   method             from
##   anova.nls          stats
##   coef.nls           stats
##   confint.nls        stats
##   deviance.nls       stats
##   df.residual.nls    stats
##   fitted.nls         stats
##   formula.nls        stats
##   logLik.nls         stats
##   nobs.nls           stats
##   plot.profile.nls   stats
##   predict.nls        stats
##   print.nls          stats
##   print.summary.nls stats
##   profile.nls        stats
##   residuals.nls      stats
##   summary.nls        stats
##   vcov.nls           stats
##   weights.nls        stats
```

```
##
## Attaching package: 'nlsalt'
```

```
## The following objects are masked from 'package:stats':
##
##     asOneSidedFormula, getInitial, nlminb, nls, nls.control,
##     NLSstAsymptotic, NLSstClosestX, NLSstLfAsymptote, NLSstRtAsymptote,
##     numericDeriv, selfStart, setNames, sortedXyData
```

```
nmod1<-nlsModel(form=eunsc, data=weeddata1, start=start1, wts=NULL, upper=NULL, scaleOffset = 0, nDcent
```

```
## numericDeriv-Alt
```

```
str(nmod1)
```

```
## List of 16
##  $ resid     :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 356 15 356 30 22 37 1004 1004
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ fitted    :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 357 16 357 29 23 36 1005 1005
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ formula   :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 358 17 358 31 24 38 1006 1006
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ deviance  :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 359 18 359 31 25 38 1007 1007
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ lhs       :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 360 13 360 26 20 33 1008 1008
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ gradient  :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 361 18 361 57 25 64 1009 1009
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ conv      :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 362 14 362 34 21 41 1010 1010
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ incr      :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 363 14 363 42 21 49 1011 1011
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ setVarying:function (vary = rep_len(TRUE, np))
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 364 20 387 7 27 14 1012 1035
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ setPars   :function (newPars)
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 388 17 395 7 24 14 1036 1043
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ getPars   :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 396 17 396 36 24 43 1044 1044
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ getAllPars:function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 397 20 397 39 27 46 1045 1045
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ getEnv    :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 398 16 398 29 23 36 1046 1046
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ trace     :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 399 15 405 7 22 14 1047 1053
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ Rmat      :function ()
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 406 14 406 32 21 39 1054 1054
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  $ predict   :function (newdata = list(), qr = FALSE)
##   ..- attr(*, "srcref")= 'srcref' int [1:8] 407 17 408 56 24 56 1055 1056
##   .. ..- attr(*, "srcfile")=Classes 'srcfilealias', 'srcfile' <environment: 0x5611bd437298>
##  - attr(*, "class")= chr "nlsModel"
```

```
ls.str(nmod1)
```

```
## conv : function ()
## deviance : function ()
## fitted : function ()
## formula : function ()
## getAllPars : function ()
## getEnv : function ()
## getPars : function ()
## gradient : function ()
## incr : function ()
## lhs : function ()
## predict : function (newdata = list(), qr = FALSE)
## resid : function ()
## Rmat : function ()
## setPars : function (newPars)
## setVarying : function (vary = rep_len(TRUE, np))
## trace : function ()
```

```
print(nmod1)
```

```
## $resid
## function() resid
## <bytecode: 0x5611bddf01b8>
## <environment: 0x5611bde30740>
##
## $fitted
## function() rhs
## <bytecode: 0x5611bddef1c0>
## <environment: 0x5611bde30740>
##
## $formula
## function() form
## <bytecode: 0x5611bddf1ff8>
## <environment: 0x5611bde30740>
##
## $deviance
## function() dev
## <bytecode: 0x5611bddf6d50>
## <environment: 0x5611bde30740>
##
## $lhs
## function() lhs
## <bytecode: 0x5611bddf5d58>
## <environment: 0x5611bde30740>
##
## $gradient
## function() .swts * attr(rhs, "gradient")
## <bytecode: 0x5611bddf8b58>
## <environment: 0x5611bde30740>
##
## $conv
## function() convCrit()
## <bytecode: 0x5611bddf7fc0>
## <environment: 0x5611bde30740>
##
## $incr
## function() qr.coef(QR, resid)
## <bytecode: 0x5611bddf7540>
## <environment: 0x5611bde30740>
##
## $setVarying
## function(vary = rep_len(TRUE, np)) {
##                  np <- length(useParams)
##       useParams <<- useP <-
##                      if(is.character(vary)) {
##                          temp <- logical(np)
##                          temp[unlist(ind[vary])] <- TRUE
##                          temp
##                      } else if(is.logical(vary) && length(vary) != np)
##                          stop("setVarying : 'vary' length must match length of parameters")
##                      else
##                          vary # envir = thisEnv
##       gradCall[[length(gradCall) - 1L]] <<- useP
##       if(all(useP)) {
##           setPars <<- setPars.noVarying
##           getPars <<- getPars.noVarying
##           getRHS  <<-  getRHS.noVarying
##             npar    <<- length(useP)
##         } else {
##             setPars <<- setPars.varying
##             getPars <<- getPars.varying
##             getRHS  <<-  getRHS.varying
##             npar    <<- sum(useP)
##         }
##         }
## <bytecode: 0x5611bddf9cb0>
## <environment: 0x5611bde30740>
##
## $setPars
## function(newPars) {
##         setPars(newPars)
##         resid <<- .swts * (lhs - (rhs <<- getRHS())) # envir = thisEnv {2 x}
##         dev   <<- sum(resid^2) # envir = thisEnv
##         if(length(gr <- attr(rhs, "gradient")) == 1L) gr <- c(gr)
##         QR <<- qr(.swts * gr) # envir = thisEnv
##         (QR$rank < min(dim(QR$qr))) # to catch the singular gradient matrix
##         }
## <bytecode: 0x5611bde193f0>
## <environment: 0x5611bde30740>
##
## $getPars
## function() getPars()
## <bytecode: 0x5611bde211f8>
## <environment: 0x5611bde30740>
##
## $getAllPars
## function() getPars()
## <bytecode: 0x5611bde20778>
## <environment: 0x5611bde30740>
##
## $getEnv
## function() env
## <bytecode: 0x5611bde1fcf8>
## <environment: 0x5611bde30740>
##
## $trace
## function() {
##         d <- getOption("digits")
##         cat(sprintf("%-*s (%.2e): par = (%s)\n", d+4L+2L*(scaleOffset > 0),
##                 formatC(dev, digits=d, flag="#"),
##                 convCrit(),
##                 paste(vapply(getPars(), format, ""), collapse=" ")))
##         }
## <bytecode: 0x5611bde24970>
## <environment: 0x5611bde30740>
##
## $Rmat
## function() qr.R(QR)
## <bytecode: 0x5611bde2ce08>
## <environment: 0x5611bde30740>
##
```

```
## $predict
## function(newdata = list(), qr = FALSE)
##                  eval(form[[3L]], as.list(newdata), env)
## <bytecode: 0x5611bde2bd30>
## <environment: 0x5611bde30740>
##
## attr(,"class")
## [1] "nlsModel"
```

Nash, John C. 1979. *Compact Numerical Methods for Computers : Linear Algebra and Function Minimisation.* Book. Hilger: Bristol.