Possible Testing Methodologies for GSoC'21 Project: Improvements to nls()

Arkajyoti Bhattacharjee:
Indian Institute of Technology, Kanpur

28 June, 2021

The purpose of this document is to collect notes on possible testing method(s) that can be used in the `Google Summer of Code`, 2021 project - **Improvements to nls()** under the umbrella of `The R Project for Statistical Computing`.

After a few initial searches on the internet and suggestions from my mentors - Dr. John and and Dr. Heather, I found the following testing packages available in R:

- testthat
- RUnit
- testthis
- unitizer
- tinytest

Other packages that may be useful are:

- covr
- covrpage

They seem to be mostly associated with test coverage and reports.

R itself has an internal testing facility in "R CMD CHECK" - so the above options are some of the external testing facilities in R.

For the purpose of experimentation, I made a trivial package called `linearModel`. It has been created in **RStudio** from `File > New Project > New Directory > R Package > Package name(linearModel)`. This creates 3 files- .Rbuildignore, NAMESPACE, DESCRIPTION, 2 folders- R and man and 1 linearModel.Rproj. Note that no *tests* directory has been created yet. Writing a few R files inside the R directory, I do `Build > Clean and Rebuild`. Result - `* DONE (linearModel)`

R CMD check

On doing `Build > Check Package`, I get the following output:

```
==> devtools::check(document = FALSE)

-- Building ------------------------------------------------------- linearModel --
Setting env vars:
* CFLAGS    : -Wall -pedantic -fdiagnostics-color=always
* CXXFLAGS  : -Wall -pedantic -fdiagnostics-color=always
* CXX11FLAGS: -Wall -pedantic -fdiagnostics-color=always
--------------------------------------------------------------------------------
v  checking for file 'C:\Users\ARKAJYOTI\Desktop\linearModel/DESCRIPTION' (593ms)
-  preparing 'linearModel':
```

```
v  checking DESCRIPTION meta-information ...
-  checking for LF line-endings in source and make files and shell scripts
-  checking for empty or unneeded directories
   Omitted 'LazyData' from DESCRIPTION
-  building 'linearModel_0.1.0.tar.gz'

-- Checking -------------------------------------------------------- linearModel --
Setting env vars:
* _R_CHECK_CRAN_INCOMING_REMOTE_: FALSE
* _R_CHECK_CRAN_INCOMING_       : FALSE
* _R_CHECK_FORCE_SUGGESTS_      : FALSE
* NOT_CRAN                      : true
-- R CMD check -------------------------------------------------------------------
-  using log directory 'C:/Users/ARKAJYOTI/Desktop/linearModel.Rcheck' (647ms)
-  using R version 4.1.0 (2021-05-18)
-  using platform: x86_64-w64-mingw32 (64-bit)
-  using session charset: ISO8859-1
-  using options '--no-manual --as-cran'
v  checking for file 'linearModel/DESCRIPTION'
-  checking extension type ... Package
-  this is package 'linearModel' version '0.1.0'
-  package encoding: UTF-8
v  checking package namespace information
v  checking package dependencies (7.2s)
v  checking if this is a source package ...
v  checking if there is a namespace
v  checking for executable files (483ms)
v  checking for hidden files and directories ...
W  checking for portable file names
   Found the following files with non-portable file names:
     R/fitted values.R
     R/predicted vs actual plots.R
   These are not fully portable file names.
   See section 'Package structure' in the 'Writing R Extensions' manual.
v  checking serialization versions
v  checking whether package 'linearModel' can be installed (6.5s)
v  checking installed package size ...
v  checking package directory
v  checking for future file timestamps (729ms)
W  checking DESCRIPTION meta-information (814ms)
   Non-standard license specification:
     What license is it under?
   Standardizable: FALSE
v  checking top-level files
v  checking for left-over files ...
v  checking index information
v  checking package subdirectories ...
v  checking R files for non-ASCII characters ...
v  checking R files for syntax errors ...
v  checking whether the package can be loaded (566ms)
v  checking whether the package can be loaded with stated dependencies ...
v  checking whether the package can be unloaded cleanly (356ms)
v  checking whether the namespace can be loaded with stated dependencies ...
```

```
v  checking whether the namespace can be unloaded cleanly (563ms)
v  checking loading without being on the library search path (666ms)
W  checking dependencies in R code (453ms)
   'library' or 'require' call not declared from: 'MASS'
   'library' or 'require' call to 'MASS' in package code.
     Please use :: or requireNamespace() instead.
     See section 'Suggested packages' in the 'Writing R Extensions' manual.
v  checking S3 generic/method consistency (892ms)
v  checking replacement functions (368ms)
v  checking foreign function calls (364ms)
N  checking R code for possible problems (5.7s)
   coeff: no visible global function definition for 'ginv'
   Undefined global functions or variables:
     ginv
v  checking Rd files (445ms)
v  checking Rd metadata ...
v  checking Rd line widths ...
v  checking Rd cross-references (360ms)
W  checking for missing documentation entries (437ms)
   Undocumented code objects:
     'coeff' 'fit.vals' 'plt' 'resid' 'sumary'
   All user-level objects in a package should have documentation entries.
   See chapter 'Writing R documentation files' in the 'Writing R
   Extensions' manual.
W  checking for code/documentation mismatches (452ms)
   Functions or methods with usage in documentation object 'hello' but not in code:
     'hello'

v  checking Rd \usage sections (2s)
v  checking Rd contents ...
v  checking for unstated dependencies in examples (343ms)
E  checking examples (1.2s)
   Running examples in 'linearModel-Ex.R' failed
   The error most likely occurred in:

   > base::assign(".ptime", proc.time(), pos = "CheckExEnv")
   > ### Name: hello
   > ### Title: Hello, World!
   > ### Aliases: hello
   >
   > ### ** Examples
   >
   > hello()
   Error in hello() : could not find function "hello"
   Execution halted
v  checking for non-standard things in the check directory ...
v  checking for detritus in the temp directory ...

   See
     'C:/Users/ARKAJYOTI/Desktop/linearModel.Rcheck/00check.log'
   for details.

-- R CMD check results ---------------------------------- linearModel 0.1.0 ----
```

```
Duration: 34.5s

> checking examples ... ERROR
  Running examples in 'linearModel-Ex.R' failed
  The error most likely occurred in:

  > base::assign(".ptime", proc.time(), pos = "CheckExEnv")
  > ### Name: hello
  > ### Title: Hello, World!
  > ### Aliases: hello
  >
  > ### ** Examples
  >
  > hello()
  Error in hello() : could not find function "hello"
  Execution halted

> checking for portable file names ... WARNING
  Found the following files with non-portable file names:
    R/fitted values.R
    R/predicted vs actual plots.R
  These are not fully portable file names.
  See section 'Package structure' in the 'Writing R Extensions' manual.

> checking DESCRIPTION meta-information ... WARNING
  Non-standard license specification:
    What license is it under?
  Standardizable: FALSE

> checking dependencies in R code ... WARNING
  'library' or 'require' call not declared from: 'MASS'
  'library' or 'require' call to 'MASS' in package code.
    Please use :: or requireNamespace() instead.
    See section 'Suggested packages' in the 'Writing R Extensions' manual.

> checking for missing documentation entries ... WARNING
  Undocumented code objects:
    'coeff' 'fit.vals' 'plt' 'resid' 'sumary'
  All user-level objects in a package should have documentation entries.
  See chapter 'Writing R documentation files' in the 'Writing R
  Extensions' manual.

> checking for code/documentation mismatches ... WARNING
  Functions or methods with usage in documentation object 'hello' but not in code:
    'hello'

> checking R code for possible problems ... NOTE
  coeff: no visible global function definition for 'ginv'
  Undefined global functions or variables:
    ginv

1 error x | 5 warnings x | 1 note x
Error: R CMD check found ERRORs
```

```
Execution halted

Exited with status 1.
```

R CMD check does a pretty extensive job it seems. The errors are mostly due to the default files that should be edited. Upon editing and adding small stuffs, I ran R CMD check again. The output is -

```
==> devtools::check(document = FALSE)

-- Building ---------------------------------------- linearModel --
Setting env vars:
* CFLAGS    : -Wall -pedantic -fdiagnostics-color=always
* CXXFLAGS  : -Wall -pedantic -fdiagnostics-color=always
* CXX11FLAGS: -Wall -pedantic -fdiagnostics-color=always
-------------------------------------------------------------------
v  checking for file 'C:\Users\ARKAJYOTI\Desktop\linearModel/DESCRIPTION' (653ms)
-  preparing 'linearModel': (408ms)
v  checking DESCRIPTION meta-information ...
-  checking for LF line-endings in source and make files and shell scripts
-  checking for empty or unneeded directories
   Omitted 'LazyData' from DESCRIPTION
-  building 'linearModel_0.1.0.tar.gz'

-- Checking ---------------------------------------- linearModel --
Setting env vars:
* _R_CHECK_CRAN_INCOMING_REMOTE_: FALSE
* _R_CHECK_CRAN_INCOMING_       : FALSE
* _R_CHECK_FORCE_SUGGESTS_      : FALSE
* NOT_CRAN                      : true
-- R CMD check -----------------------------------------------------------------
-  using log directory 'C:/Users/ARKAJYOTI/Desktop/linearModel.Rcheck' (586ms)
-  using R version 4.1.0 (2021-05-18)
-  using platform: x86_64-w64-mingw32 (64-bit)
-  using session charset: ISO8859-1
-  using options '--no-manual --as-cran'
v  checking for file 'linearModel/DESCRIPTION'
-  checking extension type ... Package
-  this is package 'linearModel' version '0.1.0'
-  package encoding: UTF-8
v  checking package namespace information
v  checking package dependencies (5.7s)
v  checking if this is a source package ...
v  checking if there is a namespace
v  checking for executable files (607ms)
v  checking for hidden files and directories ...
v  checking for portable file names ...
v  checking serialization versions ...
v  checking whether package 'linearModel' can be installed (6s)
v  checking installed package size ...
v  checking package directory
v  checking for future file timestamps (618ms)
v  checking DESCRIPTION meta-information (739ms)
v  checking top-level files ...
v  checking for left-over files
```

```
v  checking index information
v  checking package subdirectories ...
v  checking R files for non-ASCII characters ...
v  checking R files for syntax errors ...
v  checking whether the package can be loaded (570ms)
v  checking whether the package can be loaded with stated dependencies ...
v  checking whether the package can be unloaded cleanly (346ms)
v  checking whether the namespace can be loaded with stated dependencies ...
v  checking whether the namespace can be unloaded cleanly (569ms)
v  checking loading without being on the library search path (691ms)
v  checking dependencies in R code (460ms)
v  checking S3 generic/method consistency (1.1s)
v  checking replacement functions (373ms)
v  checking foreign function calls (340ms)
v  checking R code for possible problems (5.6s)
v  checking Rd files (579ms)
v  checking Rd metadata ...
v  checking Rd line widths ...
v  checking Rd cross-references (346ms)
v  checking for missing documentation entries (472ms)
v  checking for code/documentation mismatches (1.4s)
v  checking Rd \usage sections (1.5s)
v  checking Rd contents (348ms)
v  checking for unstated dependencies in examples (455ms)
v  checking examples (1.7s)
v  checking for non-standard things in the check directory
v  checking for detritus in the temp directory


-- R CMD check results ---------------------------------- linearModel 0.1.0 ----
Duration: 34.2s

0 errors v | 0 warnings v | 0 notes v

R CMD check succeeded
```

So, what I think is that R CMD check simply checks whether the functions are working fine or not. It does not check if the function does what we intend it to do??.(Yes) This is where we need external testing packages?(Yes) Note that a `tests` directory has yet not been created.

I create a `tests` directory and run R CMD check and `Build>Test Package`. The former has the same results while the latter outputs-

```
==> devtools::test()

i Loading linearModel
i Testing linearModel
Error: No test files found
Backtrace:
    x
 1. \-devtools::test()
 2.   \-testthat::test_local(...)
 3.     \-testthat::test_dir(...)
Warning message:
In normalizePath(path.expand(path), winslash, mustWork) :
  path[1]="C:/Users/ARKAJYOTI/Desktop/linearModel/tests/testthat": The system cannot find the file speci
Execution halted
```

I need to create “test” files.

We can create the `tests` directory using the following command also-

```
> usethis::use_testthat()
### OUTPUT--->

 Setting active project to 'C:/Users/ARKAJYOTI/Desktop/linearModel'
 Adding 'testthat' to Suggests field in DESCRIPTION
 Setting Config/testthat/edition field in DESCRIPTION to '3'
 Creating 'tests/testthat/'
 Writing 'tests/testthat.R'
 Call `use_test()` to initialize a basic test file and open it for  editing.
```

Note that this creates a `testthat` directory inside the `tests` directory. I use `usethis::use_test("filename")` to create test files of the function files. It will be created in the `tests\testthat` directory.

```
>usethis::use_test("5-summary")

### OUTPUT--->

  Writing 'tests/testthat/test-5-summary.R'
* Modify 'tests/testthat/test-5-summary.R'
```

Similarly, we create test files for all the R files and write tests in it.

Now, we run `Build>Check Package` again the output is-

```
==> devtools::check(document = FALSE)

-- Building ------------------------------------------------- linearModel --
Setting env vars:
* CFLAGS    : -Wall -pedantic -fdiagnostics-color=always
* CXXFLAGS  : -Wall -pedantic -fdiagnostics-color=always
* CXX11FLAGS: -Wall -pedantic -fdiagnostics-color=always
---------------------------------------------------------------------------
v  checking for file 'C:\Users\ARKAJYOTI\Desktop\linearModel/DESCRIPTION' (373ms)
-  preparing 'linearModel': (365ms)
v  checking DESCRIPTION meta-information ...
-  checking for LF line-endings in source and make files and shell scripts
-  checking for empty or unneeded directories
   Omitted 'LazyData' from DESCRIPTION
-  building 'linearModel_0.1.0.tar.gz'

-- Checking ------------------------------------------------- linearModel --
Setting env vars:
* _R_CHECK_CRAN_INCOMING_REMOTE_: FALSE
* _R_CHECK_CRAN_INCOMING_       : FALSE
* _R_CHECK_FORCE_SUGGESTS_      : FALSE
* NOT_CRAN                      : true
-- R CMD check ------------------------------------------------------------
-  using log directory 'C:/Users/ARKAJYOTI/Desktop/linearModel.Rcheck' (433ms)
-  using R version 4.1.0 (2021-05-18)
-  using platform: x86_64-w64-mingw32 (64-bit)
-  using session charset: ISO8859-1
-  using options '--no-manual --as-cran'
v  checking for file 'linearModel/DESCRIPTION'
-  checking extension type ... Package
-  this is package 'linearModel' version '0.1.0'
-  package encoding: UTF-8
v  checking package namespace information
v  checking package dependencies (6.4s)
v  checking if this is a source package ...
v  checking if there is a namespace
v  checking for executable files (596ms)
v  checking for hidden files and directories ...
v  checking for portable file names ...
v  checking serialization versions ...
v  checking whether package 'linearModel' can be installed (4.6s)
v  checking installed package size ...
v  checking package directory (411ms)
v  checking for future file timestamps (32.5s)
v  checking DESCRIPTION meta-information (369ms)
N  checking top-level files ...
   Non-standard files/directories found at top level:
     'TestsDoc.Rmd' 'TestsDoc.pdf'
v  checking for left-over files
v  checking index information
v  checking package subdirectories ...
v  checking R files for non-ASCII characters ...
v  checking R files for syntax errors ...
v  checking whether the package can be loaded (457ms)
v  checking whether the package can be loaded with stated dependencies ...
v  checking whether the package can be unloaded cleanly ...
v  checking whether the namespace can be loaded with stated dependencies ...
v  checking whether the namespace can be unloaded cleanly (347ms)
v  checking loading without being on the library search path (507ms)
v  checking dependencies in R code ...
v  checking S3 generic/method consistency (663ms)
v  checking replacement functions ...
v  checking foreign function calls ...
v  checking R code for possible problems (3.5s)
v  checking Rd files (351ms)
v  checking Rd metadata ...
v  checking Rd line widths ...
v  checking Rd cross-references ...
v  checking for missing documentation entries ...
v  checking for code/documentation mismatches (802ms)
v  checking Rd \usage sections (898ms)
v  checking Rd contents ...
v  checking for unstated dependencies in examples ...
v  checking examples (1.1s)
v  checking for unstated dependencies in 'tests' ...
-  checking tests ...
E  Running 'testthat.R' (1.3s)
   Running the tests in 'tests/testthat.R' failed.
   Last 13 lines of output:
     `names(expected)` is a character vector ('1', '2', '3', '4', '5', ...)
     -- Failure (test-predicted_vs_actual_plots.R:2:3): plt function works ----------
     plt(cars$speed, cars$dist) does not invisibly
     -- Failure (test-resids.R:2:3): residual function works! -----------------------
     resid(cars$speed, cars$dist) (`actual`) not equal to residuals(lm(cars$dist ~ cars$speed)) (`expec

     `dim(actual)` is an integer vector (50, 1)
     `dim(expected)` is absent

     `names(actual)` is absent
     `names(expected)` is a character vector ('1', '2', '3', '4', '5', ...)

     [ FAIL 4 | WARN 0 | SKIP 0 | PASS 4 ]
     Error: Test failures
     Execution halted
v  checking for non-standard things in the check directory
v  checking for detritus in the temp directory ...

   See
     'C:/Users/ARKAJYOTI/Desktop/linearModel.Rcheck/00check.log'
   for details.

-- R CMD check results ---------------------------------- linearModel 0.1.0 ----
Duration: 59.6s

> checking tests ...
  See below...

> checking top-level files ... NOTE
  Non-standard files/directories found at top level:
    'TestsDoc.Rmd' 'TestsDoc.pdf'

-- Test failures ------------------------------------------------- testthat ----

> library(testthat)
> library(linearModel)

Attaching package: 'linearModel'

The following object is masked from 'package:stats':

    resid

>
> test_check("linearModel")
== Failed tests ================================================================
-- Failure (test-beta_coeffs.R:2:5): coeff function works! ---------------------
coeff(cars$speed, cars$dist) (`actual`) not equal to coef(lm(cars$dist ~ cars$speed)) (`expected`).

`dim(actual)` is an integer vector (2, 1)
`dim(expected)` is absent

`names(actual)` is absent
```

```
`names(expected)` is a character vector ('(Intercept)', 'cars$speed')
-- Failure (test-fitted_values.R:2:5): residual function works! -----------------
fit.vals(cars$speed, cars$dist) (`actual`) not equal to fitted(lm(cars$dist ~ cars$speed)) (`expected`)

`dim(actual)` is an integer vector (50, 1)
`dim(expected)` is absent

`names(actual)` is absent
`names(expected)` is a character vector ('1', '2', '3', '4', '5', ...)
-- Failure (test-predicted_vs_actual_plots.R:2:3): plt function works ----------
plt(cars$speed, cars$dist) does not invisibly
-- Failure (test-resids.R:2:3): residual function works! -----------------------
resid(cars$speed, cars$dist) (`actual`) not equal to residuals(lm(cars$dist ~ cars$speed)) (`expected`)

`dim(actual)` is an integer vector (50, 1)
`dim(expected)` is absent

`names(actual)` is absent
`names(expected)` is a character vector ('1', '2', '3', '4', '5', ...)

[ FAIL 4 | WARN 0 | SKIP 0 | PASS 4 ]
Error: Test failures
Execution halted

1 error x | 0 warnings v | 1 note x
Error: R CMD check found ERRORs
Execution halted

Exited with status 1.
```

We get an error as 4 tests fail. `R CMD check` does and extensive job here, but it is in association with `testthat`.

Note that when we to `R CMD check` it triggers `testthat`. So, we explore `testthat` next.

testthat

We use 'Build>Test Package' or use `devtools::test()`. THe output is-

```
==> devtools::test()

i Loading linearModel
i Testing linearModel
v |  OK F W S | Context
v |   1       | 5-summary [0.1 s]
x |   1 1     | beta_coeffs [0.1 s]
---------------------------------------------------------------------
Failure (test-beta_coeffs.R:2:5): coeff function works!
coeff(cars$speed, cars$dist) (`actual`) not equal to coef(lm(cars$dist ~ cars$speed)) (`expected`).

`dim(actual)` is an integer vector (2, 1)
`dim(expected)` is absent

`names(actual)` is absent
`names(expected)` is a character vector ('(Intercept)', 'cars$speed')
```

```
--------------------------------------------------------------------
x |   1 1     | fitted_values
--------------------------------------------------------------------
Failure (test-fitted_values.R:2:5): residual function works!
fit.vals(cars$speed, cars$dist) (`actual`) not equal to fitted(lm(cars$dist ~ cars$speed)) (`expected`)

`dim(actual)` is an integer vector (50, 1)
`dim(expected)` is absent

`names(actual)` is absent
`names(expected)` is a character vector ('1', '2', '3', '4', '5', ...)
--------------------------------------------------------------------
x |   0 1     | predicted_vs_actual_plots [0.1 s]
--------------------------------------------------------------------
Failure (test-predicted_vs_actual_plots.R:2:3): plt function works
plt(cars$speed, cars$dist) does not invisibly
--------------------------------------------------------------------
x |   1 1     | resids
--------------------------------------------------------------------
Failure (test-resids.R:2:3): residual function works!
resid(cars$speed, cars$dist) (`actual`) not equal to residuals(lm(cars$dist ~ cars$speed)) (`expected`)

`dim(actual)` is an integer vector (50, 1)
`dim(expected)` is absent

`names(actual)` is absent
`names(expected)` is a character vector ('1', '2', '3', '4', '5', ...)
--------------------------------------------------------------------

== Results =========================================================
Duration: 0.5 s

[ FAIL 4 | WARN 0 | SKIP 0 | PASS 4 ]
```

`testthat` has lot of inbuilt functions that help in creating tests. Also, `R CMD check` just prints the last 13 lines of the result. `testthat` has `ListReporter`??how to use??

unitizer(2021)

```
Description: Simplifies regression tests by comparing objects produced by test code
with earlier versions of those same objects. If objects are unchanged the tests pass,
otherwise execution stops with error details. If in interactive mode, tests can be reviewed
through the provided interactive environment.
```

Based on the description and the functions included in the documentation, this package doesn't seem to be useful for the testing of the project.

tinytest (2020)

Based on the manual - "Provides a lightweight (zero-dependency) and easy to use unit testing framework. Main features: install tests with the package. Test results are treated as data that can be stored and manipulated. Test files are R scripts interspersed with test commands, that can be programmed over. Fully automated build-install-test sequence for packages. Skip tests when not run locally (e.g. on CRAN). Flexible and configurable output printing. Compare computed output with output stored with the package. Run tests in parallel. Extensible by other packages. Report side effects.", `tinytest` seems appealing.

It also has expectation functions similar to `testthat`. Note that `expect_vector` and `expect_output` are not there in tiny_test and testin individual test files is easy using `run_test_file("test-file.R")`. Running the entire tests directory is yet to be done.

Useful Links

[1] Unit Tests in R

[2] PACKAGES FOR TESTING YOUR R PACKAGE

[3] testthat: Unit Testing for R

[4] https://debruine.github.io/tutorials/packages.html

[5] https://kbroman.org/pkg_primer/pages/tests.html

[6] testthat: Unit Tesing for R

[7] Not Explored (automated testing)

[8] Not Explored (Github repo)

[9] https://towardsdatascience.com/unit-testing-in-r-68ab9cc8d211

[10] https://debruine.github.io/tutorials/packages.html