Create your own R package

Aimin Yan

Basic structure of an R package

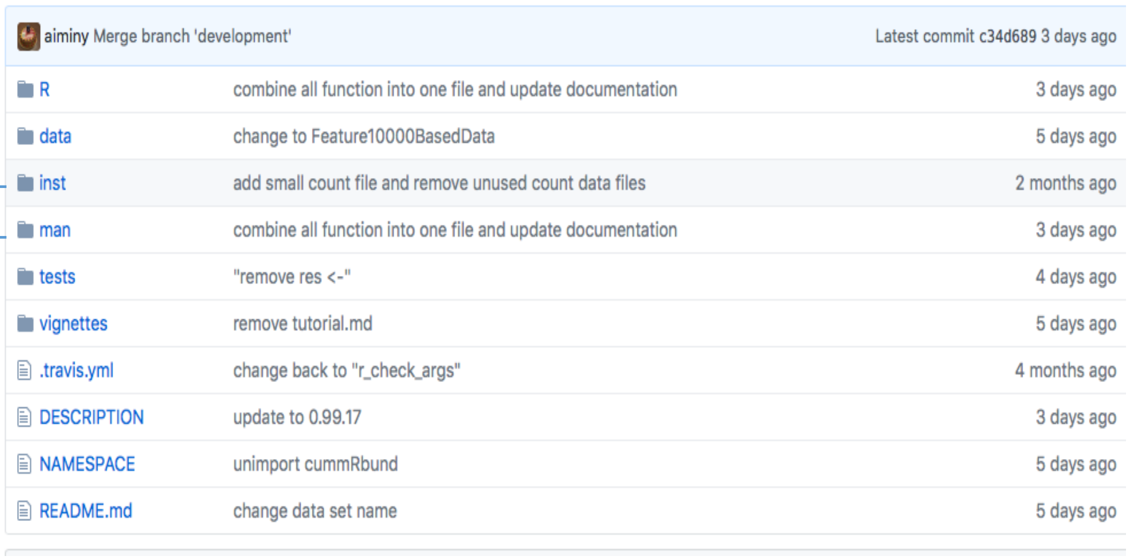
R code

External things

Doc

Test your product to make sure it can handle different situations

metadata



aiminy Merge branch 'development' Latest commit c34d689 3 days ago

R	combine all function into one file and update documentation	3 days ago
data	change to Feature10000BasedData	5 days ago
inst	add small count file and remove unused count data files	2 months ago
man	combine all function into one file and update documentation	3 days ago
tests	"remove res <-"	4 days ago
vignettes	remove tutorial.md	5 days ago
.travis.yml	change back to "r_check_args"	4 months ago
DESCRIPTION	update to 0.99.17	3 days ago
NAMESPACE	unimport cummRbund	5 days ago
README.md	change data set name	5 days ago

Use Rstudio to make an R package

- Open Rstudio
- Make a R file, Ex: test.R
- File -> New project -> New Directory -> R package
- Give package name, Ex:testR
- Add test.R
- Select directory where you want to put your testR, package name is usually your directory name
- Create project-> clean and rebuild
- Basic structure of your package is done, next you put your content in your package

Keep truck of R package building using Git

- Go to github
- Create a new git reposipotry, Ex: testR
- Go to shell:
  - cd testR (go to testR directory)
  - git remote add origin https://github.com/SCCC-BBC/testR.git
  - git add R
  - git commit -m "add first R”
  - git push -u origin master
- Go to Rstudio, you will find buttons of pull and push are already highlighted. Your R package git truck setting is done. From now on, you can keep truck of your development on rstudio, do not need worry to go to shell

Version control of R package

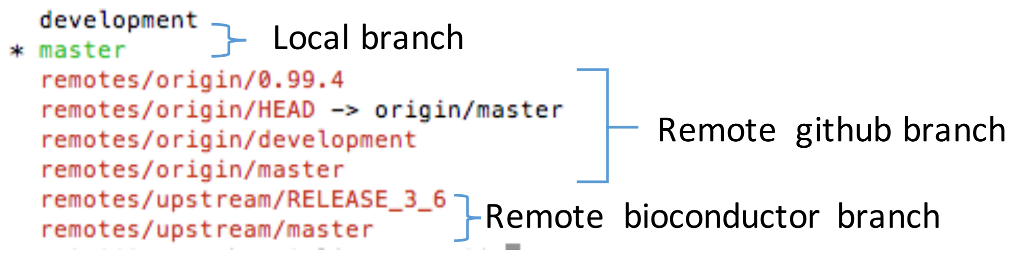
- Usually at least you need to have two versions for your R package
- Release version is used for users to use
- Development version to let you put new functions in this package
- Test in development version, and when it is ready, you can merge new updates to release version

Merge between two branches

- #create development branch
  - git checkout -b development
- #make changes, commit
  - git push origin -u development
- #merge the changes in development to master
  - # Merge all changes in development to master
  - # change to master , then do the following on the shell
    - git branch –a # check which branch you are in
    - git checkout master # if you are not in master branch, change to master branch
    - git merge --no-ff development
  - # Merge the changes in some files in development to master
  - git checkout development path/file1 path/file2,…

Maintain an R package that has been released

- Your R package has been released, you have updates(Bug,new function,users requets,…). You need to change it:
- Go to your local directory of your package
- git remote add upstream git@git.bioconductor.org:packages/PathwaySplice.git
- git fetch –all
- git branch -a # check branch



```
  development
* master
  remotes/origin/0.99.4
  remotes/origin/HEAD -> origin/master
  remotes/origin/development
  remotes/origin/master
  remotes/upstream/RELEASE_3_6
  remotes/upstream/master
```

Maintain an R package that has been released-continued--

- git checkout master
- git merge upstream/master
- Change to local development branch, update version number as the master branch, make any updates here
- git checkout master
- git merge --no-ff development
- git push origin master # update on github
- git push upstream master # update on bioconductor

Maintain an R package that has been released-continued--

- git checkout RELEASE_3_6 # create the local release branch
- git merge upstream/RELEASE_3_6 # merge upstream to local
- git push upstream RELEASE_3_6 # update bioconductor RELEASE_3_6 branch
- git push origin RELEASE_3_6 # create a github RELEASE_3_6 branch

Next step

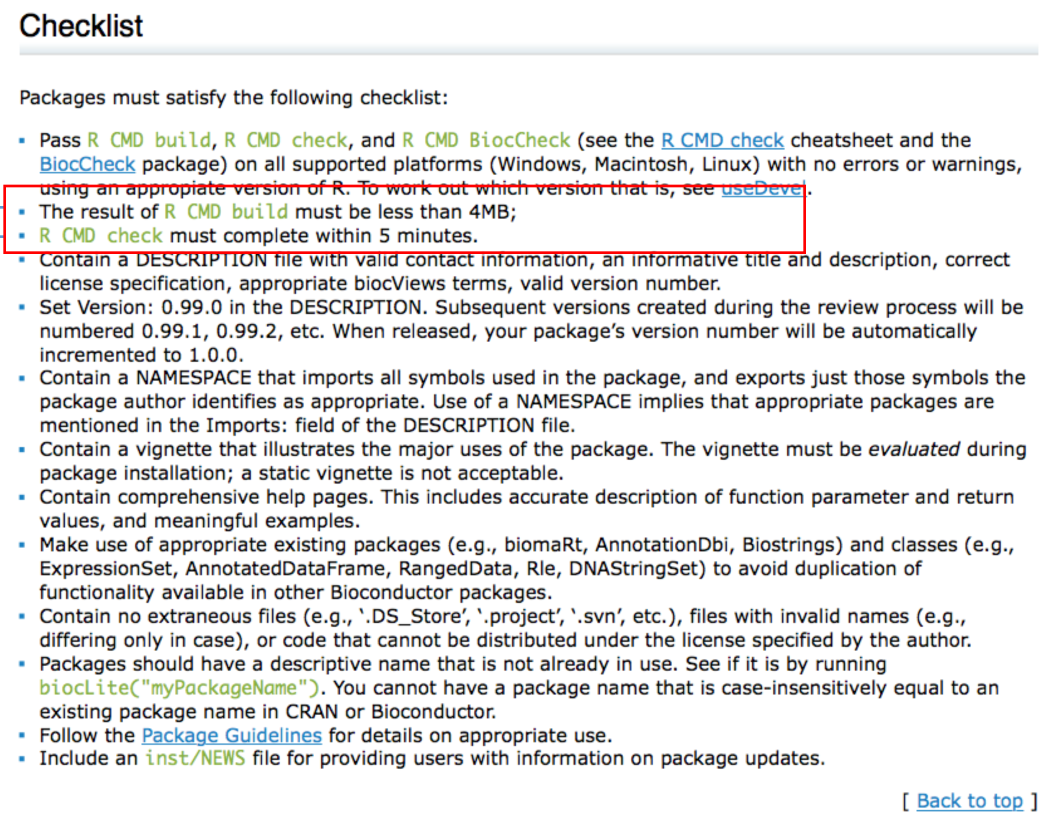
- Make your R functions and documentations
- Your documentation is created automatically
- Testing,…, submission,testing,...,

Trust levels of R package

Git CRAN Biocconductor

Checklist before submission

So your example test should be suitable



Checklist

Packages must satisfy the following checklist:

- Pass `R CMD build`, `R CMD check`, and `R CMD BiocCheck` (see the R CMD check cheatsheet and the BiocCheck package) on all supported platforms (Windows, Macintosh, Linux) with no errors or warnings, using an appropriate version of R. To work out which version that is, see useDevel.
- The result of `R CMD build` must be less than 4MB;
- `R CMD check` must complete within 5 minutes.
- Contain a DESCRIPTION file with valid contact information, an informative title and description, correct license specification, appropriate biocViews terms, valid version number.
- Set Version: 0.99.0 in the DESCRIPTION. Subsequent versions created during the review process will be numbered 0.99.1, 0.99.2, etc. When released, your package's version number will be automatically incremented to 1.0.0.
- Contain a NAMESPACE that imports all symbols used in the package, and exports just those symbols the package author identifies as appropriate. Use of a NAMESPACE implies that appropriate packages are mentioned in the Imports: field of the DESCRIPTION file.
- Contain a vignette that illustrates the major uses of the package. The vignette must be *evaluated* during package installation; a static vignette is not acceptable.
- Contain comprehensive help pages. This includes accurate description of function parameter and return values, and meaningful examples.
- Make use of appropriate existing packages (e.g., biomaRt, AnnotationDbi, Biostrings) and classes (e.g., ExpressionSet, AnnotatedDataFrame, RangedData, Rle, DNAStringSet) to avoid duplication of functionality available in other Bioconductor packages.
- Contain no extraneous files (e.g., '.DS_Store', '.project', '.svn', etc.), files with invalid names (e.g., differing only in case), or code that cannot be distributed under the license specified by the author.
- Packages should have a descriptive name that is not already in use. See if it is by running `biocLite("myPackageName")`. You cannot have a package name that is case-insensitively equal to an existing package name in CRAN or Bioconductor.
- Follow the Package Guidelines for details on appropriate use.
- Include an `inst/NEWS` file for providing users with information on package updates.

[Back to top]

Next, just do it…,

Install and use your package

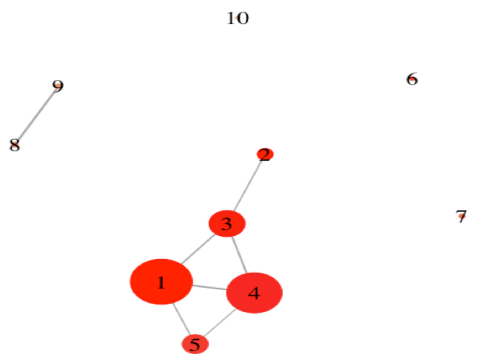
Several R Packages I am working on:

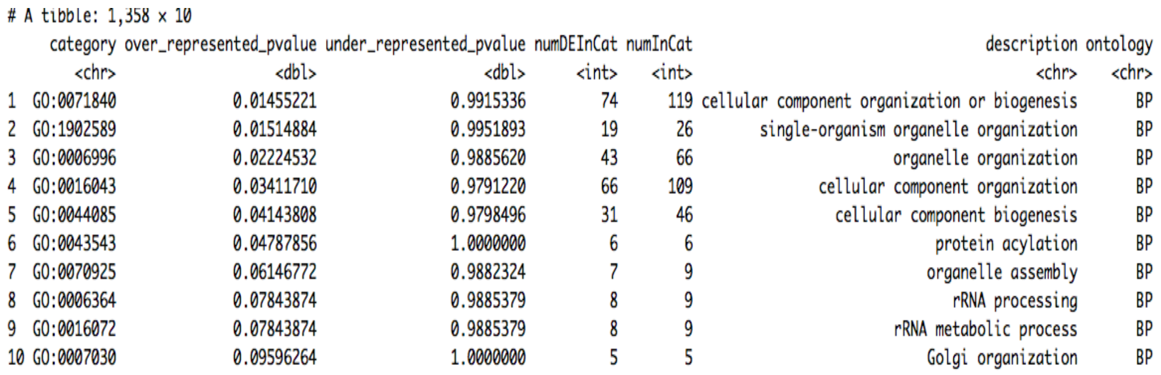
1. flowR
2. Sophia
3. Exome
4. ChipSeq
5. DoGs
6. PathwaySplice
7. RNGS
8. SraSubmitter

Use one package as an example

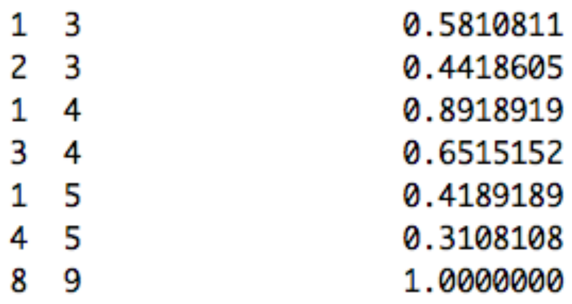
```
R -e 'library(devtools);install_github("aiminy/SRA-data-submission")'

library(SraSubmitter)
# Ex: you put your bam files in the following directory
input.bam.file.path <- "/media/H_driver/2016/Submission2SRA/Guoyan_Nimer_mouse_RNA-Seq_BAM/"
# you can run
callSubmitter(input.bam.file.path)
```

Enrichment map from PathwaySplice





```
# A tibble: 1,358 x 10
     category over_represented_pvalue under_represented_pvalue numDEInCat numInCat                                  description ontology
        <chr>                   <dbl>                    <dbl>      <int>    <int>                                        <chr>    <chr>
1  GO:0071840             0.01455221                0.9915336         74      119 cellular component organization or biogenesis       BP
2  GO:1902589             0.01514884                0.9951893         19       26        single-organism organelle organization       BP
3  GO:0006996             0.02224532                0.9885620         43       66                       organelle organization       BP
4  GO:0016043             0.03411710                0.9791220         66      109              cellular component organization       BP
5  GO:0044085             0.04143808                0.9798496         31       46                cellular component biogenesis       BP
6  GO:0043543             0.04787856                1.0000000          6        6                            protein acylation       BP
7  GO:0070925             0.06146772                0.9882324          7        9                           organelle assembly       BP
8  GO:0006364             0.07843874                0.9885379          8        9                              rRNA processing       BP
9  GO:0016072             0.07843874                0.9885379          8        9                       rRNA metabolic process       BP
10 GO:0007030             0.09596264                1.0000000          5        5                           Golgi organization       BP
```



```
1  3         0.5810811
2  3         0.4418605
1  4         0.8918919
3  4         0.6515152
1  5         0.4189189
4  5         0.3108108
8  9         1.0000000
```

Vertex color corresponds to over represented pvalue
Vertex size corresponds to numDEinCat
Edge width corresponds to Jaccard similarity coefficient
(0: there are no overlapping genes between two gene sets.
1: indicates two gene sets are identical)