Reproducible research `works_with_R`

by Toby Dylan Hocking

Introduction

Q1. What does the following R code mean?

```
install.packages("changepoint")
```

A. "Download the **most recent version** of the changepoint package, and install it ."

Q2. What does the following R code mean?

```
library(changepoint)
```

A. "Load the **currently installed version** of the changepoint package."

But for reproducible research we often want to do something different:

"Load the version of the changepoint package that was used to carry out the original analysis **several years ago**."

"Load the same package **versions that my collaborator used** on her computer."

A real problem

Q3. Will my R script give the same results if I run it using a different version of the changepoint package?

A. Not necessarily! CRAN does not enforce consistency between package versions. So any **two versions of the same package are neither guaranteed to yield identical nor consistent results**.

Example 1.

```
> library(changepoint)
> x <- 1:10
> fit <- cpt.mean(x)
> cpts(fit)
```

With changepoint_1.0.4 I got

```
cpt
  5
```

With changepoint_0.8 I got

```
cpt
  5  10
```

Example 2.
The ggplot2 package introduced several backwards-incompatible changes from version 1.0.1 to version 2.0 in 2015.

```
> ggplot()+geom_point(
+   aes(Petal.Length, Sepal.Length),
+   data=iris)+
+  facet_grid(Species~.,
+   labeller=function(var, val){
+    paste(var, val)
+   })
Error in paste(var, val) (from #1) :
argument "val" is missing, with no
default
>
```

Proposal: write versions in R code

For example instead of `library(changepoint)` we can write

```
works_with_R("3.2.3", changepoint="0.8")
```

which means "the following R code should run and produce the expected results if you install R-3.2.3, then install and load version 0.8 of the changepoint package."
And

```
works_with_R("3.2.2",
"hadley/scales@2c3edf45de56d617444dc38e47e0404173817886",
"tdhock/ggplot2@a8b06ddb680acdcdbd927773b1011c562134e4d2",
"tdhock/animint@6b1c9e588b03f632cd39cdec9bbcfa730db9e889")
```

which means "the following R code works with these specific package versions from GitHub."

Implementation Details

How should works_with_R be distributed?
- In a package? Chicken and egg problem, what version of the `works_with_R` package should be required?
- In `~/.Rprofile` -- user-specific implementations.

What actions should the works_with_R function carry out?
- "**Stop** with an error if the written package versions do not match the installed package versions. Then let the user decide whether to (1) run the script with the incorrect versions, or (2) take time to install the old versions." -- easy to implement, not so user-friendly.
- "**Warn** if the written CRAN package versions do not match the installed package versions. If any installed GitHub packages do not match the written versions, then install them using devtools." -- the current version that I am using, as defined in my `https://github.com/tdhock/dotfiles/blob/master/.Rprofile`
- "**Install** and load the written versions of each package." -- more user-friendly, but harder to implement. How to programmatically install old versions of R and CRAN packages?

Some ideas:
- `repmis::InstallOldPackages` provides a programmatic method to install old packages from CRAN.
- `library(installr)` can install R itself (only on windows).

Comparison with other solutions

- Python's `pip` and `easy_install` command line tools allow installation of any versions of packages on the Python Package Index (PyPI).
- Ruby's `rvm` can install any version of Ruby and any gem.
- Bioconductor uses a different CRAN-like repository for each of its releases. Should be OK if you only use BioC packages.
- `library(checkpoint)` can install packages as if you had a CRAN time machine that can go back to 2014-09-17. Should be OK if you only use CRAN packages.
- `library(packrat)` writes a project-specific `packrat.lock` file that contains versions. Same idea as `works_with_R`, but less explicit. Works with GitHub packages!

Limitations

Backwards-incompatible changes in external C libraries used by R, e.g. Perl-Compatible Regular Expressions (PCRE). Should we also write down versions for these libraries via `extSoftVersion()`? (new in R-3.2.0, Apr 2015)

Conclusions

- It is possible to do reproducible research in R, if you write down the version numbers of R and all of the packages that you used.
- `works_with_R` declarations are a simple, readable, and executable method that can be used to facilitate reproducible research.
- The R community needs to develop tools for installing old versions of R and CRAN packages.