[RFC] A case for freezing CRAN

Romain Francois-3 Mar 19, 2014; 8:13pm Re: [RFC] A case for freezing CRAN Reply | Threaded | More

126 posts

Weighting in. FWIW, I find the proposal conceptually quite interesting.

For package developers, it does not have to be a frustration to have to wait a new version of R to release their code. Anticipated frustration was my initial reaction. Thinking about this more, I think this could be changed into opportunity.

Since the pattern here is to use Rcpp as an example of something causing compatibility headaches, and I have some responsibility there, maybe I can comment on this. I would find it extremely valuable if there was only one unique version of Rcpp for a given released version of R.

Users would have to wait longer to have the new stuff, but one can argue that at least they get something that is more tested.

Would it be helpful for authors of package that have lots of dependency to start having stricter depends declarations in their DESCRIPTION files, e.g. :

Depends: R (== 3.1.0)

?

Romain

For example, personally I'm waiting for 3.1.0 for releasing Rcpp11 because I want to leverage some C++11 support that has been included in R. It has been frustrating to have to wait, but it does change the way I make changes to the codebase. Perhaps it is a good habit to take. And it does not need « more work » for others, just more discipline and self control from people implementing this pattern.

also, declaring a strict dependency requirement against a released version of R perhaps could resume the drama of « you were asked to test this against a very recent version of R-devel, and guess what a few hours ago I've just added a new test that makes your package non R CMD check worthy ». So less work for CRAN maintainers then.

Le 19 mars 2014 à 23:57, Hervé Pagès <[hidden email]> a écrit :

>
>
> On 03/19/2014 02:59 PM, Joshua Ulrich wrote:
>> On Wed, Mar 19, 2014 at 4:28 PM, Jeroen Ooms <[hidden email]> wrote:
>>> On Wed, Mar 19, 2014 at 11:50 AM, Joshua Ulrich <[hidden email]>
>>> wrote:
>>>>
>>>> The suggested solution is not described in the referenced article. It
>>>> was not suggested that it be the operating system's responsibility to
... [show rest of quote]

[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Gavin Simpson-3 Mar 19, 2014; 9:41pm Re: [RFC] A case for freezing CRAN Reply | Threaded | More

20 posts

In reply to this post by Carl Boettiger

"What am I overlooking?"

That this is already available and possible in R today, but perhaps not widely used. Developers do tend to only include a lower bound if they include any bounds at all on package dependencies.

As I mentioned elsewhere, R packages often aren't "built" against other R packages and often developers may have a range of versions being tested against, some of which may not be on CRAN yet.

Technically, all packages on CRAN would need to have a dependency cap on R-devel, but as that is a moving target until it is released I don't see in practice how enforcing an upper limit on the R dependency would work. The way CRAN works, you can't just set a dependency on R == 3.0.x say. (As far as I understand CRAN's policies.)

For packages it is quite trivial for the developers to manually add the required info for the upperbound, less so the lower bound, but you could just pick a known working version. An upper range on the dependencies could be stated as whatever version is current on CRAN. But then what happens? Unbeknownst to you, a few days after you release to CRAN your package foo with stated dependency on bar >= 1.2, bar <= 1.8, the developer of bar releases bar v 2.0 and your package no longer passes checks, CRAN gets in touch and you have to resubmit another version. This could be desirable in terms of helping contribute to reproducibility exercises, but incurs more effort on the CRAN maintainers and package maintainers. Now, this might be an issue because of the desire on CRAN's behalf to have some elements of human intervention in the submission process, but you either work with CRAN or do your own thing.

As Bioconductor have shown (for example) it is possible, if people want to put in time and effort and have a community buy into an ethos, to achieve staged releases etc.

G

On 19 March 2014 12:58, Carl Boettiger <[hidden email]> wrote:

> Dear list,
>
> I'm curious what people would think of a more modest proposal at this time:
>
> State the version of the dependencies used by the package authors when the
> package was built.
>
> Eventually CRAN could enforce such a statement be present in the
> description. We encourage users to declare the version of the packages they

... [show rest of quote]

--
Gavin Simpson, PhD

_____
[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Gavin Simpson-3 Mar 19, 2014; 9:54pm Re: [RFC] A case for freezing CRAN Reply | Threaded | More


20 posts

In reply to this post by Hervé Pagès

Given that R is (has) moved to a 12 month release cycle, I don't want to either i) wait a year to get new packages (or allow users to use new versions of my packages), or ii) have to run R-devel just to use new packages. (or be on R-testing for that matter).

People then will start finding ways around these limitations and then we're back to square one of having people use a set of R packages and R versions that could potentially be all over the place.

As a package developer, it is pretty easy to say I've tested my package works with these other packages and their versions, and set DESCRIPTION to reflect only those versions as allowed (or a range as a package matures and the maintainer has tested against more versions of the dependencies). CRAN may well not like this if your package no longer builds/checks on their system but then you have a choice to make; stick to your reproducibility guns & forsake CRAN in favour of something else (github, one's own repo), or relent and meet CRANs requirements.

On 19 March 2014 16:57, Hervé Pagès <[hidden email]> wrote:

>
>
> On 03/19/2014 02:59 PM, Joshua Ulrich wrote:
>>
>> On Wed, Mar 19, 2014 at 4:28 PM, Jeroen Ooms <[hidden email]>
>> wrote:
>>>
>>> On Wed, Mar 19, 2014 at 11:50 AM, Joshua Ulrich <[hidden email]>
>>> wrote:
... [show rest of quote]

--
Gavin Simpson, PhD

_____
[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Michael Weylandt Mar 19, 2014; 9:55pm Re: [RFC] A case for freezing CRAN Reply | Threaded | More


2453 posts

In reply to this post by Joshua Ulrich

On Mar 19, 2014, at 18:42, Joshua Ulrich <[hidden email]> wrote:

> On Wed, Mar 19, 2014 at 5:16 PM, Jeroen Ooms <[hidden email]> wrote:
>> On Wed, Mar 19, 2014 at 2:59 PM, Joshua Ulrich <[hidden email]> wrote:
>>>
>>> So implementation isn't a problem. The problem is that you need a way
>>> to force people not to be able to use different package versions than
>>> what existed at the time of each R release. I said this in my
>>> previous email, but you removed and did not address it: "However, you
>>> would need to find a way to actively _prevent_ people from installing

>>> newer versions of packages with the stable R releases." Frankly, I
... [show rest of quote]

Jeroen,

Reading this thread again, is it a fair summary of your position to say "reproducibility by default is more important than giving users access to the newest bug fixes and features by default?" It's certainly arguable, but I'm not sure I'm convinced: I'd imagine that the ratio of new work being done vs reproductions is rather high and the current setup optimizes for that already.

What I'm trying to figure out is why the standard "install the following list of package versions" isn't good enough in your eyes? Is it the lack of CRAN provided binaries or the fact that the user has to proactively set up their environment to replicate that of published results?

In your XML example, it seems the problem was that the reproducer didn't check that the same package versions as the reproducee and instead assumed that 'latest' would be the same. Annoying yes, but easy to solve.

Michael

Gavin Simpson-3 Mar 19, 2014; 10:17pm **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

20 posts

Michael,

I think the issue is that Jeroen wants to take that responsibility out of the hands of the person trying to reproduce a work. If it used R 3.0.x and packages A, B and C then it would be trivial to to install that version of R and then pull down the stable versions of A B and C for that version of R. At the moment, one might note the packages used and even their versions, but what about the versions of the packages that the used packages rely upon & so on? What if developers don't state know working versions of dependencies?

The problem is how the heck do you know which versions of packages are needed if developers don't record these dependencies in sufficient detail? The suggested solution is to freeze CRAN at intervals alongside R releases. Then you'd know what the stable versions were.

Or we could just get package developers to be more thorough in documenting dependencies. Or R CMD check could refuse to pass if a package is listed as a dependency but with no version qualifiers. Or have R CMD build add an upper bound (from the current, at build-time version of dependencies on CRAN) if the package developer didn't include and upper bound. Or... The first is unliekly to happen consistently, and no-one wants *more* checks and hoops to jump through :-)

To my mind it is incumbent upon those wanting reproducibility to build the tools to enable users to reproduce works. When you write a paper or release a tool, you will have tested it with a specific set of packages. It is relatively easy to work out what those versions are (there are tools in R for this). What is required is an automated way to record that info in an agreed upon way in an approved file/location, and have a tool that facilitates setting up a package library sufficient with which to reproduce a work. That approval doesn't need to come from CRAN or R Core - we can store anything in ./inst.

Reproducibility is a very important part of doing "science", but not everyone using CRAN is doing that. Why force everyone to march to the reproducibility drum? I would place the onus elsewhere to make this work.

Gavin
A scientist, very much interested in reproducibility of my work and others.

On 19 March 2014 19:55, Michael Weylandt <[hidden email]> wrote:

>
>
> On Mar 19, 2014, at 18:42, Joshua Ulrich <[hidden email]> wrote:
>
>> On Wed, Mar 19, 2014 at 5:16 PM, Jeroen Ooms <[hidden email]> wrote:
>>> On Wed, Mar 19, 2014 at 2:59 PM, Joshua Ulrich <[hidden email]> wrote:
>>>>
>>>> So implementation isn't a problem. The problem is that you need a way
>>>> to force people not to be able to use different package versions than
... [show rest of quote]

--
Gavin Simpson, PhD

_____
[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Michael Weylandt Mar 19, 2014; 10:44pm **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

2453 posts

On Mar 19, 2014, at 22:17, Gavin Simpson <[hidden email]> wrote:

> Michael,
>
> I think the issue is that Jeroen wants to take that responsibility out
> of the hands of the person trying to reproduce a work. If it used R
> 3.0.x and packages A, B and C then it would be trivial to to install
> that version of R and then pull down the stable versions of A B and C
> for that version of R. At the moment, one might note the packages used
> and even their versions, but what about the versions of the packages
> that the used packages rely upon & so on? What if developers don't
... [show rest of quote]

Doesn't sessionInfo() give all of this?

If you want to be very worried about every last bit, I suppose it should also include options(), compiler flags, compiler version, BLAS details, etc. (Good talk on the dregs of a floating point number and how hard it is to reproduce them across processors http://www.youtube.com/watch?v=GIlp4rubv8U)

>
> The problem is how the heck do you know which versions of packages are
> needed if developers don't record these dependencies in sufficient
> detail? The suggested solution is to freeze CRAN at intervals
> alongside R releases. Then you'd know what the stable versions were.

Only if you knew which R release was used.

>
> Or we could just get package developers to be more thorough in
> documenting dependencies. Or R CMD check could refuse to pass if a
> package is listed as a dependency but with no version qualifiers. Or
> have R CMD build add an upper bound (from the current, at build-time
> version of dependencies on CRAN) if the package developer didn't
> include and upper bound. Or... The first is unliekly to happen
> consistently, and no-one wants *more* checks and hoops to jump through
> :-)
... [show rest of quote]

But the tools already allow it with minimal effort. If the author can't even include session info, how can we be sure the version of R is known. If we can't know which version of R, can we ever change R at all? Etc to absurdity.

My (serious) point is that the tools are in place, but ramming them down folks' throats by intentionally keeping them on older versions by default is too much.

> When you write a paper
> or release a tool, you will have tested it with a specific set of

> packages. It is relatively easy to work out what those versions are
> (there are tools in R for this). What is required is an automated way
> to record that info in an agreed upon way in an approved
> file/location, and have a tool that facilitates setting up a package
> library sufficient with which to reproduce a work. That approval
> doesn't need to come from CRAN or R Core - we can store anything in
> ./inst.

I think the package version and published paper cases are different.

For the latter, the recipe is simple: if you want the same results, use the same software (as noted by sessionInfoPlus() or equiv)

For the former, I think you start straying into this NP complete problem:
http://people.debian.org/~dburrows/model.pdf

Yes, a good config can (and should be recorded) but isn't that exactly what sessionInfo() gives?

>
> Reproducibility is a very important part of doing "science", but not
> everyone using CRAN is doing that. Why force everyone to march to the
> reproducibility drum? I would place the onus elsewhere to make this
> work.
>

Agreed: reproducibility is the onus of the author, not the reader

> Gavin
> A scientist, very much interested in reproducibility of my work and others.

Michael
In finance, where we call it "Auditability" and care very much as well :-)

[[alternative HTML version deleted]]

_____
[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Jeroen Ooms. Mar 19, 2014; 10:45pm **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More


89 posts

In reply to this post by Michael Weylandt

On Wed, Mar 19, 2014 at 6:55 PM, Michael Weylandt
<[hidden email]> wrote:
> Reading this thread again, is it a fair summary of your position to say "reproducibility by default is more important than giving users access to the newest bug fixes and features by default?" It's certainly arguable, but I'm not sure I'm convinced: I'd imagine that the ratio of new work being done vs reproductions is rather high and the current setup optimizes for that already.

I think that separating development from released branches can give us both reliability/reproducibility (stable branch) as well as new features (unstable branch). The user gets to pick (and you can pick both!). The same is true for r-base: when using a 'released' version you get 'stable' base packages that are up to 12 months old. If you want to have the latest stuff you download a nightly build of r-devel. For regular users and reproducible research it is recommended to use the stable branch. However if you are a developer (e.g. package author) you might want to develop/test/check your work with the latest r-devel.

I think that extending the R release cycle to CRAN would result both in more stable released versions of R, as well as more freedom for package authors to implement rigorous change in the unstable branch. When writing a script that is part of a production pipeline, or sweave paper that should be reproducible 10 years from now, or a book on using R, you use stable version of R, which is guaranteed to behave the same over time. However when developing packages that should be compatible with the upcoming release of R, you use r-devel which has the latest versions of other CRAN and base packages.

> What I'm trying to figure out is why the standard "install the following list of package versions" isn't good enough in your eyes?

Almost nobody does this because it is cumbersome and impractical. We can do so much better than this. Note that in order to install old packages you also need to investigate which versions of dependencies of those packages were used. On win/osx, users need to manually build those packages which can be a pain. All in all it makes reproducible research difficult and expensive and error prone. At the end of the day most published results obtain with R just won't be reproducible.

Also I believe that keeping it simple is essential for solutions to be practical. If every script has to be run inside an environment with custom libraries, it takes away much of its power. Running a bash or python script in Linux is so easy and reliable that entire distributions are based on it. I don't understand why we make our lives so difficult in R.

In my estimation, a system where stable versions of R pull packages from a stable branch of CRAN will naturally resolve the majority of the reproducibility and reliability problems with R. And in contrast to what some people here are suggesting it does not introduce any limitations. If you want to get the latest stuff, you either grab a copy of r-devel, or just enable the testing branch and off you go. Debian 'testing' works in a similar way, see http://www.debian.org/devel/testing.

Michael Weylandt Mar 19, 2014; 11:16pm **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

2453 posts

On Mar 19, 2014, at 22:45, Jeroen Ooms <[hidden email]> wrote:

> On Wed, Mar 19, 2014 at 6:55 PM, Michael Weylandt
> <[hidden email]> wrote:
>> Reading this thread again, is it a fair summary of your position to say "reproducibility by default is more important than giving users access to the newest bug fixes and features by default?" It's certainly arguable, but I'm not sure I'm convinced: I'd imagine that the ratio of new work being done vs reproductions is rather high and the current setup optimizes for that already.
>
> I think that separating development from released branches can give us
> both reliability/reproducibility (stable branch) as well as new
... [show rest of quote]

I think where you are getting push back (e.g., Frank Harrell and Josh Ulrich) is from saying that 'stable' is the right branch for 'regular users.' And I tend to agree: I think most folks need features and bug fixes more than they need to reproduce a particular paper with no effort on their end.

>
> I think that extending the R release cycle to CRAN would result both
> in more stable released versions of R, as well as more freedom for
> package authors to implement rigorous change in the unstable branch.

Not sure what exactly you mean by this sentence.

> When writing a script that is part of a production pipeline, or sweave
> paper that should be reproducible 10 years from now, or a book on
> using R, you use stable version of R, which is guaranteed to behave
> the same over time.

Only if you never upgrade anything... But that's the case already, isn't it?

> However when developing packages that should be
> compatible with the upcoming release of R, you use r-devel which has
> the latest versions of other CRAN and base packages.
>
>
>> What I'm trying to figure out is why the standard "install the following list of package versions" isn't

good enough in your eyes?
>
> Almost nobody does this because it is cumbersome and impractical. We
... [show rest of quote]

So you want CRAN to host old binaries ad infinitum? I think that's entirely reasonable/doable if (big if) storage and network are free.

>
> Also I believe that keeping it simple is essential for solutions to be
> practical. If every script has to be run inside an environment with
> custom libraries, it takes away much of its power. Running a bash or
> python script in Linux is so easy and reliable that entire
> distributions are based on it. I don't understand why we make our
> lives so difficult in R.

Because for Debian style (stop the world on release) distro, there are no upgrades within a release. And that's only halfway reasonable because of Debian's shockingly good QA.

It's certainly not true for, e.g., Arch.

I've been looking at python incompatibilities across different RHEL versions lately. There's simply no way to get around explicit version pinning (either by release number or date, but when you have many moving pieces, picking a set of release numbers is much easier than finding a single day when they all happened to work together) if it has to work exactly as it used to.

>
> In my estimation, a system where stable versions of R pull packages
> from a stable branch of CRAN will naturally resolve the majority of
> the reproducibility and reliability problems with R.

And what everyone else is saying is "if you want to reproduce results made with old software, download and use the old software." Both can me made to work -- it's just a matter of pros and cons of different defaults.

> And in contrast
> to what some people here are suggesting it does not introduce any
> limitations. If you want to get the latest stuff, you either grab a
> copy of r-devel, or just enable the testing branch and off you go.
> Debian 'testing' works in a similar way, see
> http://www.debian.org/devel/testing.

Karl Millar Mar 20, 2014; 12:30am **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

8 posts

In reply to this post by Jeroen Ooms.

I think what you really want here is the ability to easily identify and sync to CRAN snapshots.

The easy way to do this is setup a CRAN mirror, but back it up with version control, so that it's easy to reproduce the exact state of CRAN at any given point in time. CRAN's not particularly large and doesn't churn a whole lot, so most version control systems should be able to handle that without difficulty.

Using svn, mod_dav_svn and (maybe) mod_rewrite, you could setup the server so that e.g.:
http://my.cran.mirror/repos/2013-01-01/
is a mirror of how CRAN looked at midnight 2013-01-01.

Users can then set their repository to that URL, and will have a stable snapshot to work with, and can have all their packages built with that snapshot if they like. For reproducibility purposes, all users need to do is to agree on the same date to use. For publication purposes, the date of the snapshot should be sufficient.

We'd need a version of update.packages() that force-syncs all the packages to the version in the repository, even if they're downgrades,

but otherwise it ought to be fairly straight-forward.

FWIW, we do something similar internally at Google. All the packages that a user has installed come from the same source control revision, where we know that all the package versions are mutually compatible. It saves a lot of headaches, and users can rollback to any previous point in time easily if they run into problems.

On Wed, Mar 19, 2014 at 7:45 PM, Jeroen Ooms <[hidden email]> wrote:

> On Wed, Mar 19, 2014 at 6:55 PM, Michael Weylandt
> <[hidden email]> wrote:
>> Reading this thread again, is it a fair summary of your position to say "reproducibility by default is more important than giving users access to the newest bug fixes and features by default?" It's certainly arguable, but I'm not sure I'm convinced: I'd imagine that the ratio of new work being done vs reproductions is rather high and the current setup optimizes for that already.
>
> I think that separating development from released branches can give us
> both reliability/reproducibility (stable branch) as well as new
... [show rest of quote]

David Winsemius Mar 20, 2014; 2:03am Re: [RFC] A case for freezing CRAN Reply | Threaded | More

9137 posts

In reply to this post by Jeroen Ooms.

On Mar 19, 2014, at 7:45 PM, Jeroen Ooms wrote:

> On Wed, Mar 19, 2014 at 6:55 PM, Michael Weylandt
> <[hidden email]> wrote:
>> Reading this thread again, is it a fair summary of your position to say "reproducibility by default is more important than giving users access to the newest bug fixes and features by default?" It's certainly arguable, but I'm not sure I'm convinced: I'd imagine that the ratio of new work being done vs reproductions is rather high and the current setup optimizes for that already.
>
> I think that separating development from released branches can give us
> both reliability/reproducibility (stable branch) as well as new
... [show rest of quote]

As I remember ... The example demonstrating the need for this was an XML package that cause an extract from a website where the headers were misinterpreted as data in one version of pkg:XML and not in another. That seems fairly unconvincing. Data cleaning and validation is a basic task of data analysis. It also seems excessive to assert that it is the responsibility of CRAN to maintain a synced binary archive that will be available in ten years. Bug fixes would be inhibited for years.... not unlike SAS and Excel. What next? Perhaps al bugs should be labeled as features? Surely this CRAN-of-the-future would be offering something that no other statistical package currently offers, nicht wahr?

Why not leave it to the authors to specify the packages which version numbers were used in their publications. The authors of the packages would get recognition and the dependencies would be recorded.

--
David.

>
>
>> What I'm trying to figure out is why the standard "install the following list of package versions" isn't good enough in your eyes?
>
> Almost nobody does this because it is cumbersome and impractical. We
> can do so much better than this. Note that in order to install old
> packages you also need to investigate which versions of dependencies
> of those packages were used. On win/osx, users need to manually build
... [show rest of quote]

David Winsemius
Alameda, CA, USA

[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Dan Tenenbaum Mar 20, 2014; 2:15am Re: [RFC] A case for freezing CRAN Reply | Threaded | More


96 posts

----- Original Message -----

> From: "David Winsemius" <[hidden email]>
> To: "Jeroen Ooms" <[hidden email]>
> Cc: "r-devel" <[hidden email]>
> Sent: Wednesday, March 19, 2014 11:03:32 PM
> Subject: Re: [Rd] [RFC] A case for freezing CRAN
>
>
> On Mar 19, 2014, at 7:45 PM, Jeroen Ooms wrote:
>
... [show rest of quote]

CRAN already does this, the bin/windows/contrib directory has subdirectories going back to 1.7, with packages dated October 2004. I don't see why it is burdensome to continue to archive these. It would be nice if source versions had a similar archive.

Dan

> Bug fixes would
> be inhibited for years.... not unlike SAS and Excel. What next?
> Perhaps al bugs should be labeled as features? Surely this
> CRAN-of-the-future would be offering something that no other
> statistical package currently offers, nicht wahr?
>
> Why not leave it to the authors to specify the packages which version
> numbers were used in their publications. The authors of the packages
> would get recognition and the dependencies would be recorded.
... [show rest of quote]

[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Rainer M Krug-3 Mar 20, 2014; 4:49am Re: [RFC] A case for freezing CRAN Reply | Threaded | More


68 posts

In reply to this post by Michael Weylandt

Michael Weylandt <[hidden email]> writes:

> On Mar 19, 2014, at 22:17, Gavin Simpson <[hidden email]> wrote:
>
>> Michael,
>>
>> I think the issue is that Jeroen wants to take that responsibility out
>> of the hands of the person trying to reproduce a work. If it used R
>> 3.0.x and packages A, B and C then it would be trivial to to install
>> that version of R and then pull down the stable versions of A B and C
>> for that version of R. At the moment, one might note the packages used
... [show rest of quote]
In principle yes - but this calls specifically for a package which is extracting the info and stores it into a human readable format, which can then be used to re-install (automatically) all the versions for (hopefully) reproducibility - because if there are external libraries included, you HAVE problems.

>
>>

>> The problem is how the heck do you know which versions of packages are
>> needed if developers don't record these dependencies in sufficient
>> detail? The suggested solution is to freeze CRAN at intervals
>> alongside R releases. Then you'd know what the stable versions were.
>
> Only if you knew which R release was used.

Well - that would be easier to specify in a paper then the version infos
of all packages needed - and which ones of the installed ones are
actually needed? OK - the ones specified in library() calls. But wait -
there are dependencies, imports, ... That is a lot of digging - I wpul;d
not know how to do this out of my head, except by digging through the
DESCRIPTION files of the packages...

>
>>
>> Or we could just get package developers to be more thorough in
>> documenting dependencies. Or R CMD check could refuse to pass if a
>> package is listed as a dependency but with no version qualifiers. Or
>> have R CMD build add an upper bound (from the current, at build-time
>> version of dependencies on CRAN) if the package developer didn't
>> include and upper bound. Or... The first is unliekly to happen
>> consistently, and no-one wants *more* checks and hoops to jump through
... [show rest of quote]
Dependencies, imports, package versions, ... not that straight forward I
would say.

>
> For the former, I think you start straying into this NP complete problem:
http://people.debian.org/~dburrows/model.pdf
>
> Yes, a good config can (and should be recorded) but isn't that exactly what sessionInfo() gives?
>
>>
>> Reproducibility is a very important part of doing "science", but not
>> everyone using CRAN is doing that. Why force everyone to march to the
... [show rest of quote]
Exactly - but also the authors of the software which is aimed at being
used in the context of reproducibility - the tools should be there to
make it easy!

My points are:

1) I think the snapshot idea of CRAN is a good idea which should be
followed
2) The snapshots should be incorporated at CRAN as I assume that CRAN
will be there longer then any third party repository.
3) the default for the user should *not* change, i.e. normal users will
always get the newest packages as it is now
4) If this can / will not be done because of workload, storage space,
... commands should be incorporated in a package (preferably which
becomes part of the core packages) to store snapshots of installed
package and R version information as a human readable text file, but
which can be parsed by a second command to re-create this setup.

Cheers, and thanks for this important discussion (could have been a GSoC
project?),

Rainer

>
>
>> Gavin
>> A scientist, very much interested in reproducibility of my work and others.
>
> Michael
> In finance, where we call it "Auditability" and care very much as well :-)
>
>
... [show rest of quote]
--
Rainer M. Krug

email: Rainer<at>krugs<dot>de
PGP: 0x0F52F982

[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

attachment0 (504 bytes) Download Attachment

Rainer M Krug-3 Mar 20, 2014; 5:04am Re: [RFC] A case for freezing CRAN Reply | Threaded | More


68 posts

In reply to this post by hadley wickham

Hadley Wickham <[hidden email]> writes:

>> What would be more useful in terms of reproducibility is the capability of
>> installing a specific version of a package from a repository using
>> install.packages(), which would require archiving older versions in a
>> coordinated fashion. I know CRAN archives old versions, but I am not aware
>> if we can programmatically query the repository about this.
>
> See devtools::install_version().
>
> The main caveat is that you also need to be able to build the package,
... [show rest of quote]
The compiling will always be the problem when using older source packages, whatever is done.

But for the dependencies: an automatic parsing of the dependencies (DEPENDS, IMPORTS, ...) would help a lot.

Together with a command which scans the installed package in the session and stores them in a parsable human readable format so that all packages (with the specified version) required can be installed with one command, and I think the problem would be much closer to be solved.

Rainer

>
> Hadley

--
Rainer M. Krug
email: Rainer<at>krugs<dot>de
PGP: 0x0F52F982

[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

attachment0 (504 bytes) Download Attachment

Duncan Murdoch-2 Mar 20, 2014; 6:52am Re: [RFC] A case for freezing CRAN Reply | Threaded | More


3037 posts

In reply to this post by Dan Tenenbaum

On 14-03-20 2:15 AM, Dan Tenenbaum wrote:

>
>
> ----- Original Message -----
>> From: "David Winsemius" <[hidden email]>
>> To: "Jeroen Ooms" <[hidden email]>
>> Cc: "r-devel" <[hidden email]>
>> Sent: Wednesday, March 19, 2014 11:03:32 PM
>> Subject: Re: [Rd] [RFC] A case for freezing CRAN
>>
... [show rest of quote]

The bin/windows/contrib directories are updated every day for active R versions. It's only when Uwe decides that a version is no longer worth active support that he stops doing updates, and it "freezes". A

consequence of this is that the snapshots preserved in those older directories are unlikely to match what someone who keeps up to date with R releases is using. Their purpose is to make sure that those older versions aren't completely useless, but they aren't what Jeroen was asking for.

Karl Millar's suggestion seems like an ideal solution to this problem. Any CRAN mirror could implement it. If someone sets this up and commits to maintaining it, I'd be happy to work on the necessary changes to the install.packages/update.packages code to allow people to use it from within R.

Duncan Murdoch

[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Roger Bivand Mar 20, 2014; 7:37am **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

367 posts

In reply to this post by Gavin Simpson-3

Gavin Simpson <ucfagls <at> gmail.com> writes:

>
...

>
>
> To my mind it is incumbent upon those wanting reproducibility to build
> the tools to enable users to reproduce works. When you write a paper
> or release a tool, you will have tested it with a specific set of
> packages. It is relatively easy to work out what those versions are
> (there are tools in R for this). What is required is an automated way
> to record that info in an agreed upon way in an approved
> file/location, and have a tool that facilitates setting up a package
... [show rest of quote]

Gavin,

Thanks for contributing useful insights. With reference to Jeroen's proposal and the discussion so far, I can see where the problem lies, but the proposed solutions are very invasive. What might offer a less invasive resolution is through a robust and predictable schema for sessionInfo() content, permitting ready parsing, so that (using Hadley's interjection) the reproducer could reconstruct the original execution environment at least as far as R and package versions are concerned.

In fact, I'd argue that the responsibility for securing reproducibility lies with the originating author or organisation, so that work where reproducibility is desired should include such a standardised record.

There is an additional problem not addressed directly in this thread but mentioned in some contributions, upstream of R. The further problem upstream is actually in the external dependencies and compilers, beyond that in hardware. So raising consciousness about the importance of being able to query version information to enable reproducibility is important.

Next, encapsulating the information permitting its parsing would perhaps enable the original execution environment to be reconstructed locally by installing external dependencies, then R, then packages from source, using the same versions of build train components if possible (and noting mismatches if not). Maybe ressurect StatDataML in addition to RData serialization of the version dependencies? Of course, current R and package versions may provide reproducibility, but if they don't, one would use the parseable record of the original development environment

>
> Reproducibility is a very important part of doing "science", but not
> everyone using CRAN is doing that. Why force everyone to march to the
> reproducibility drum? I would place the onus elsewhere to make this
> work.

Exactly.

Roger

>
> Gavin
> A scientist, very much interested in reproducibility of my work and others.
>
...
> >
> > _____
> > R-devel <at> r-project.org mailing list
> > https://stat.ethz.ch/mailman/listinfo/r-devel
>

_____
[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Roger Bivand
Department of Economics
NHH Norwegian School of Economics
Helleveien 30
N-5045 Bergen, Norway

S Ellison-2 Mar 20, 2014; 8:14am **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

301 posts

In reply to this post by Jeroen Ooms.

> If we could all agree on a particular set
> of cran packages to be used with a certain release of R, then it doesn't matter
> how the 'snapshotting' gets implemented.

This is pretty much the sticking point, though. I see no practical way of reaching that agreement without the kind of decision authority (and effort) that Linux distro maintainers put in to the internal consistency of each distribution.

CRAN doesn't try to do that; it's just a place to access packages offered by maintainers.

As a package maintainer, I think support for critical version dependencies in the imports or dependency lists is a good idea that individual package maintainers could relatively easily manage, but I think freezing CRAN as a whole or adopting single release cycles for CRAN would be thoroughly impractical.

S Ellison

****
This email and any attachments are confidential. Any use...{{dropped:8}}

_____
[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Jari Oksanen Mar 20, 2014; 9:43am **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

129 posts

On 20/03/2014, at 14:14 PM, S Ellison wrote:

>> If we could all agree on a particular set
>> of cran packages to be used with a certain release of R, then it doesn't matter
>> how the 'snapshotting' gets implemented.
>
> This is pretty much the sticking point, though. I see no practical way of reaching that agreement without the kind of decision authority (and effort) that Linux distro maintainers put in to the internal consistency of each distribution.
>
> CRAN doesn't try to do that; it's just a place to access packages offered by maintainers.
... [show rest of quote]

I have a feeling that this discussion has floated between two different arguments in favour of freezing: discontent with package authors who break their packages within R release cycle, and ability to reproduce old results. In the beginning the first argument was more prominent, but now the discussion has drifted to reproducing old results.

I cannot see how freezing CRAN would help with package authors who do not separate development and CRAN release branches but introduce broken code, or code that breaks other packages. Freezing a broken snapshot would only mean that the situation cannot be cured before next R release, and then new breakage could be introduced. Result would be dysfunctional CRAN. I think that quite a few of the package updates are bug fixes and minor enhancements. Further, I do think that these should be "backported" to previous versions of R: users of previous version of R should also benefit from bug fixes. This also is the current CRAN policy and I think this is a good policy. Personally, I try to keep my packages in such a condition that they will also work in previous versions of R so that people do not need to upgrade R to have bug fixes in packages.

The policy is the same with Linux maintainers: they do not just build a consistent release, but maintain the release by providing bug fixes. In Linux distributions, end of life equals freezing, or not providing new versions of software.

Another issue is reproducing old analyses. This is a valuable thing, and sessionInfo and ability to get certain versions of package certainly are steps forward. It looks that guaranteed reproduction is a hard task, though. For instance, R 2.14.2 is the oldest version of R that I can build out of the box in my Linux desktop. I have earlier built older, even much older, R versions, but something has happened in my OS that crashes the build process. To reproduce an old analysis, I also should install an older version of my OS, then build old R and then get the old versions of packages. It is nice if the last step is made easier.

Cheers, Jari Oksanen

[hidden email] mailing list
https://stat.ethz.ch/mailman/listinfo/r-devel

Hervé Pagès Mar 20, 2014; 3:14pm **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

452 posts

In reply to this post by Duncan Murdoch-2

On 03/20/2014 03:52 AM, Duncan Murdoch wrote:

> On 14-03-20 2:15 AM, Dan Tenenbaum wrote:
>>
>>
>> ----- Original Message -----
>>> From: "David Winsemius" <[hidden email]>
>>> To: "Jeroen Ooms" <[hidden email]>
>>> Cc: "r-devel" <[hidden email]>
>>> Sent: Wednesday, March 19, 2014 11:03:32 PM
>>> Subject: Re: [Rd] [RFC] A case for freezing CRAN
... [show rest of quote]

But it is almost completely useless from a reproducibility point of view to get random package versions. For example if some people try to use R-2.13.2 today to reproduce an analysis that was published 2 years ago, they'll get Matrix 1.0-4 on Windows, Matrix 1.0-3 on Mac, and Matrix 1.1-2-2 on Unix. And none of them of course is what was used by the authors of the paper (they used Matrix 1.0-1, which is what was current when they ran their analysis).

A big improvement from a reproducibility point of view would be to (a) have a clear cut for the freezes, (b) freeze the source packages as well as the binary packages, and (c) freeze the same versions of source and binaries. For example the freeze of bin/windows/contrib/x.y, bin/macosx/contrib/x.y and contrib/x.y could happen when the R-x.y series itself freezes (i.e. no more minor versions planned for this series).

Cheers,
H.

>
> Karl Millar's suggestion seems like an ideal solution to this problem.
> Any CRAN mirror could implement it. If someone sets this up and commits
> to maintaining it, I'd be happy to work on the necessary changes to the

> install.packages/update.packages code to allow people to use it from
> within R.
>
> Duncan Murdoch
>
... [show rest of quote]

--
Hervé Pagès

Program in Computational Biology
Division of Public Health Sciences
Fred Hutchinson Cancer Research Center
1100 Fairview Ave. N, M1-B514
P.O. Box 19024
Seattle, WA 98109-1024

E-mail: [hidden email]
Phone: (206) 667-5791
Fax: (206) 667-1319

Ted Byers Mar 20, 2014; 4:28pm **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

107 posts

On Thu, Mar 20, 2014 at 3:14 PM, Hervé Pagès <[hidden email]> wrote:

> On 03/20/2014 03:52 AM, Duncan Murdoch wrote:
>
>> On 14-03-20 2:15 AM, Dan Tenenbaum wrote:
>>
>>>
>>>
>>> ----- Original Message -----
>>>
>>> From: "David Winsemius" <[hidden email]>
... [show rest of quote]

Initially this discussion brought back nightmares of DLL hell on Windows. Those as ancient as I will remember that well. But now, the focus seems to be on reproducibility, but with what strikes me as a seriously flawed notion of what reproducibility means.

Herve Pages mentions the risk of irreproducibility across three minor revisions of version 1.0 of Matrix. My gut reaction would be that if the results are not reproducible across such minor revisions of one library, they are probably just so much BS. I am trained in mathematical ecology, with more than a couple decades of post-doc experience working with risk assessment in the private sector. When I need to do an analysis, I will repeat it myself in multiple products, as well as C++ or FORTRAN code I have hand-crafted myself (and when I wrote number crunching code myself, I would do so in multiple programming languages - C++, Java, FORTRAN, applying rigorous QA procedures to each program/library I developed). Back when I was a grad student, I would not even show the results to my supervisor, let alone try to publish them, unless the results were reproducible across ALL the tools I used. If there was a discrepancy, I would debug that before discussing them with anyone. Surely, it is the responsibility of the journals' editors and reviewers to apply a similar practice.

The concept of reproducibility used to this point in this discussion might be adequate from a programmers perspective (except in my lab), it is wholly inadequate from a scientist's perspective. I maintain that if you have the original data, and repeat the analysis using the latest version of R and the available, relevant packages, the original results are probably due to a bug either in the R script or in R or the packages used IF the results obtained using the latest versions of these are not consistent with the originally reported results. Therefore, of the concerns I see raised in this discussion, the principle one of concern is that of package developers who fail to pay sufficient attention to backwards compatibility: a new version ought not break any code that executes fine using previous

versions. That is not a trivial task, and may require contributors obtaining the assistance of a software engineer. I am sure anyone in this list who programs in C++ knows how the ANSI committees handle change management. Introduction of new features is something that is largely irrelevant for backwards compatibility (but there are exceptions), but features to be removed are handled by declaring them deprecated, and leaving them in that condition for years. That tells anyone using the language that they ought to plan to adapt their code to work when the deprecated feature is finally removed.

I am responsible for maintaining code (involving distributed computing) to which many companies integrate their systems, and I am careful to ensure that no change I make breaks their integration into my system, even though I often have to add new features. And I don't add features lightly, and have yet to remove features. When that eventually happens, the old feature will be deprecated, so that the other companies have plenty of time to adapt their integration code. I do not know whether CRAN ought to have any responsibility for this sort of change management, or if they have assumed some responsibility for some of it, but I would argue that the package developers have the primary responsibility for doing this right.

Just my $0.05 (the penny no longer exists in Canada)

Cheers

Ted
R.E. (Ted) Byers, Ph.D., Ed.D.

[[alternative HTML version deleted]]

Jeroen Ooms. Mar 20, 2014; 4:53pm **Re: [RFC] A case for freezing CRAN** Reply | Threaded | More

89 posts

On Thu, Mar 20, 2014 at 1:28 PM, Ted Byers <[hidden email]> wrote:
>
> Herve Pages mentions the risk of irreproducibility across three minor
> revisions of version 1.0 of Matrix. My gut reaction would be that if the
> results are not reproducible across such minor revisions of one library,
> they are probably just so much BS.
>

Perhaps this is just terminology, but what you refer to I would generally call 'replication'. Of course being able to replicate results with other data or other software is important to validate claims. But being able to reproduce how the original results were obtained is an important part of this process.

If someone is publishing results that I think are questionable and I cannot replicate them, I want to know exactly how those outcomes were obtained in the first place, so that I can 'debug' the problem. It's quite important to be able to trace back if incorrect results were a result of a bug, incompetence or fraud.

Let's take the example of the Reinhart and Rogoff case. The results obviously were not replicable, but without more information it was just the word of a grad students vs two Harvard professors. Only after reproducing the original analysis it was possible to point out the errors and proof that the original were incorrect.

[[alternative HTML version deleted]]

« Return to R devel | 229 views

1 2 3 4

Powered by Nabble

Edit this page