Archiving Reproducible Research with R and Dataverse

by Thomas J. Leeper[1]

Abstract Reproducible research and data archiving are increasingly important issues in research involving statistical analyses of quantitative data. This article introduces the **dvn** package, which allows R users to publicly archive datasets, analysis files, codebooks, and associated metadata in Dataverse Network online repositories, an open-source data archiving project sponsored by Harvard University. In this article I review the importance of data archiving in the context of reproducible research, introduces the Dataverse Network, explain the implementation of the **dvn** package, and provide example code for archiving and releasing data using the package.

Data archiving and reproducible research

Reproducible research involves the careful, annotated preservation of data, analysis code, and associated files, such that statistical procedures, output, and published results can be directly and fully replicated (see, for example, King, 1995; Boyer, 2003). R provides an increasingly large set of tools for engaging in reproducible research practices. Perhaps most prominently is **knitr** (Xie, 2013), which expands upon Sweave (Leisch, 2002) to help R users produce fully reproducible research reports for a variety of markup languages. Gandrud (2013b) and others have advocated for the widespread use of reproducible research tools, but a major shortcoming of the existing set of tools is a means of easily archiving reproducible research files for use by others. While it is easy to privately archive files for reproducing one's own work later on, making study files publicly available is not something readily supported by R or other existing tools.

A reproducible research workflow is most valuable when it allows others to replicate results and reproduce published results precisely. Thus complete records of all data and analyses need to be publicly and persistently available via a stable, freely available archive. Gandrud (2013a,b) has advocated for the use of Dropbox and GitHub for fulfilling this archiving role. Both sites are supported, to varying extents, by existing R packages: **rDrop** (Ram, 2012) for Dropbox and **rgithub** (Scheidegger, 2013) for GitHub. However, Dropbox is not designed for persistent archiving, depends on assumptions about the future availability of a proprietary, closed-source web service, and offers no platform for finding archived data (thus introducing a further dependence on the publication of a Dropbox link somewhere else on the web). GitHub, which integrates with the git version-control system (which many R users may already invoke as part of their existing workflow), is helpful for supporting an ongoing (and possibly collaborative) reproducible research workflow but is not designed to be a persistent data archive.

Thus researchers remain in need of a service dedicated to the persistent preservation of data with sophisticated metadata support to allow others to easily find, understand, and use archived files. One such option is The Dataverse Network, a free-to-use, open-source data archive sponsored and developed by the Institute for Quantitative Social Science at Harvard University (Altman et al., 2001; King, 2007).[2] The next section introduces The Dataverse Network and the remainder of the article describes how to utilize the archive directly from R via the **dvn** package.

The Dataverse Network

The Dataverse Network is a server application that hosts collections of studies, called dataverses.[3] As an open-source application, The Dataverse Network has many instances (i.e., hosted by different institutions) and each implementation, e.g., the Harvard IQSS Dataverse Network, hosts many dataverses controlled by individuals, institutions, and academic journals. Though there are few The Dataverse Network instances (about 10 worldwide), each of these — especially those hosted by Harvard University and the Odum Institute at the University of North Carolina–Chapel Hill — hosts thousands of individual dataverses containing a collective hundreds of thousands of study listings. Consequently, The Dataverse Network is an increasingly prominent repository of data and associated files and is the default archive for many journals that currently require public archiving of replication files as a condition of publication. To use The Dataverse Network, one needs to create an account. To

[2]Another alternative is `http://figshare.com/`, which is already supported by **rfigshare**.

[3]Hereafter, the small-d "dataverse" refers to a collection of studies stored with The Dataverse Network and "The Dataverse Netowrk" refers to the general web application.

release studies, one also needs to create a personal dataverse collection associated with the account. Accounts are tied to specific The Dataverse Network instances, so in this article we will focus for simplicity sake on the The Harvard Dataverse Network.

Each study archived in a dataverse contains, at a minimum, an identifying title, but may additionally include an array of metadata and files. This means The Dataverse Network can be used to store not just a free-standing data file, but associated files in almost any format and one might include files such as data, codebooks, analysis replication files, statistical packages, questionnaires, experimental materials, and so forth. Data uploaded to a dataverse is converted to a generic tabular format and can be subsequently downloaded into one of several common formats (e.g., Stata, S-Plus, R, and tab-delimited).

Once created, a study is given a global identifier in the form of either a Handle,[4] or DataCite DOI,[5] digital object identifiers that uniquely and globally identify the study. Studies are version controlled when new files are added or when metadata is modified and previous versions remain persistently accessible. Furthermore, when data files are modified, the study identifier is updated with additional version information in the form of Universal Numeric Fingerprint (UNF) (Altman and King, 2007; Altman, 2008). The UNF is a format-independent hash of a data file, which provides a unique identifier for a dataset without revealing the content of that dataset, such that any change to the underlying data yields a new UNF. This means that data stored in a dataverse can be cited (e.g., in scholarly publications) not only by their handle but also by their specific version using the UNF.[6] Reproducible research efforts can thus be enhanced by use not only of the same named dataset but the exact same version of that dataset, a major advantage over storing data in other types of archive.[7] Via a UNF, data users can thus check a dataset they possess against one cited in a research publication or stored in a dataverse.[8]

Archiving data and files using The Dataverse Network also means that those files can be made available to others. The Dataverse Network settings offer fine-grained control over access to archived studies, meaning a study collection can be regulated anywhere from completely publicly accessible, to accessible only to registered users, to accessible only to whitelisted users. Public data then becomes easily searchable by others based on any of the metadata associated with the study files, a major advantage over archiving on any other website.[9] The supported metadata elements include almost all imaginable aspects of a data collection, include those provided by the Dublin Core standard,[10] such as creator, subject, description, date, type, rights,[11] citation, etc. Additional metadata elements are also supported and Dataverse additionally supplies metadata in the Data Documentation Initiative (DDI) format,[12] a comprehensive metadata schema for quantitative data, especially for the social sciences.

The dvn package

The **dvn** package provides a lightweight wrapper for two Application Programming Interfaces (APIs) provided by The Dataverse Network. The first API — called the Data Sharing API — is essentially a search utility for locating existing studies within a Dataverse Network instance using simple HTTP GET requests. The second API — called the Data Deposit API — is a bit more complicated. It is built on the SWORD (Simple Web-service Offering Repository Deposit) protocol,[13] an open-source standard for the deposit of digital records.[14] This API relies on HTTP requests to GET, PUT, POST, and DELETE resources on a Dataverse server. Access to both APIs is implemented through **RCurl** (Temple Lang, 2012a) and responses are parsed with **XML** (Temple Lang, 2012b). From the user perspective, **dvn** masks the distinction between the two APIs, offering functions both for conducting data searches and for archiving data, code, and associated metadata.

[4]`http://hdl.net/`

[5]`http://www.datacite.org/`

[6]The version-controlled Handle thus serves as a complete data citation, referencing a specific version of a specific data file (see `http://thedata.org/citation/standard`). A description of (current) Version 5 of the UNF algorithm is available at `http://thedata.org/book/unf-version-5-0`.

[7]Of course, a version-controlled repository (e.g., on GitHub), would also create a hash for objects.

[8]Functions necessary to calculate a UNF signature on a dataframe are provided by **UNF** (Leeper, 2013).

[9]The need for metadata in the reproducible research workflow has previously been identified by Gentleman and Temple Lang (2004).

[10]`http://dublincore.org/documents/dcmi-terms/`

[11]The Dataverse Network supports any user-defined licensing terms for data use. For a discussion of issues related to data licensing, see Stodden (2009).

[12]`http://www.ddialliance.org/Specification/`

[13]`http://swordapp.org/`

[14]As an example, the SWORD standard is also used by Open Journal Systems for managing articles through the review and publication process and for making them publicly available.

Thus **dvn** adds to an increasingly large number of packages that connect R to web-based services,[15] such as those built by the ROpenSci project.[16] Perhaps the most proximate existing R package to **dvn** is **rfigshare** (Boettiger et al., 2013), which allows R users to persistently store files on http://figshare.com/. **rdryad** (Chamberlain et al., 2013) and **OAIHarvester** (Hornik, 2013) — which can be used to harvest metadata records from http://datadryad.org/ or, in the latter case, any public repository that supports the OAI-PMH standard[17] — share commonalities with The Dataverse Network Data Sharing API but neither allows users to upload their own files.

dvn will remain under active development for the foreseeable future and will track any modifications made to The Dataverse Network APIs. Bug reports, code pull requests, and other suggestions are welcome on the **dvn** homepage at https://github.com/rOpenSci/dvn. The next two sections describe functionality of the package in its current form.

Data and metadata search with dvn

The search functionality of **dvn** can be used to easily locate archived data in any public Dataverse Network. **dvn** defaults to providing access to The Harvard Dataverse Network (located at https://thedata.harvard.edu/dvn/), but this can be changed in each function call or globally using options:

```
# use Odum Institute Dataverse
options(dvn = 'https://arc.irss.unc.edu/dvn/')
# use Harvard IQSS Dataverse
options(dvn = 'https://thedata.harvard.edu/dvn/')
```

for any valid Dataverse Network.[18] With a Dataverse Network selected, we can easily search that network for studies using any metadata field. `dvSearchField` provides a list of available metadata search fields. These can be used to search for studies by various authors or on various topics using `dvSearch` (which accepts a boolean logic):

```
dvSearch(list(authorName = "leeper"))
dvSearch(list(title = "Denmark", title = "Sweden"), boolean = "OR")
dvSearch("Puppies")
```

Calling `dvSearch` with a single character argument searches all metadata fields simultaneously using a boolean OR logic. The result is (presently) a one-column dataframe listing `objectId` values, which represent the global identifier for each study.[19] Here's an example using our final search from above:

```
> dvSearch('puppies')
2 search results returned

          objectId
1 hdl:1902.1/21521
2 hdl:1902.1/21522
```

An `objectId` can then be used to retrieve detailed metadata available for a study. Metadata is available in two formats: Dublic Core (DC) and Data Documentation Initiative (DDI).[20] DC is an almost universally adopted, minimalist metadata schema widely used in information science. DDI is a more sophisticated format designed initially for social science data applications that can include considerably more metadata information.[21] By default, Dataverse and **dvn** return the more comprehensive DDI format. To request DC, one needs to specify it using the `format.type` argument:

```
dvMetadata("hdl:1902.1/21964") # return DDI (by default)
dvMetadata("hdl:1902.1/21964", format.type="oai_dc") # return DC
```

Currently `dvMetadata` returns metadata as a character string with S3 class "`dvMetadata`" because both DC and DDI output can be quite extensive and are better viewed in a text editor than in the

[15] http://cran.r-project.org/web/views/WebTechnologies.html

[16] http://ropensci.org/

[17] http://www.openarchives.org/pmh/

[18] See http://thedata.org/book/dataverse-networks-around-world for a complete listing of public Dataverse Networks. Harvard also hosts a demo server for experimenting with The Dataverse Network: http://dvn-demo.iq.harvard.edu/dvn/.

[19] Dataverse has used both Handles and DOIs as global identifiers, thus the generic and forward-compatible `objectId` terminology.

[20] For a given study, it is possible to check the availability of different metadata formats using `dvMetadataFormats`.

[21] A crosswalk mapping metadata elements from DC, DDI, and the Dataverse web interface is available at http://thedata.harvard.edu/guides/dataverse-api-main.html#data-deposit-api.

R console. For similar reasons, **dvn** does not fully parse the DDI or DC XML. Instead, the wrapper function dvExtractFileIds extracts relevant information from the DDI formatted metadata.[22] Here's an example:

```
files <- dvExtractFileIds(dvMetadata("hdl:1902.1/21964"))
> files[, c('fileName', 'fileId')]
                         fileName  fileId
1    study2-replication-analysis.r 2341713
2    study1-replication-analysis.r 2341709
3                      coefpaste.r 2341888
4                     expResults.r 2341889
5            Study 2 Codebook.docx 2341712
6 study2-data-final-2012-06-08.csv 2341711
7 study1-data-final-2012-06-08.csv 2341710
8             Study 2 Webpages.zip 2341714
9             Study 1 Webpages.zip 2341715
```

With a fileId returned by dvExtractFileIds, one can see the formats available for a file (e.g., data can be downloaded in a number of formats), using dvDownloadInfo:

```
> dvDownloadInfo(files$fileId[1])
File Name:       study2-replication-analysis.r
File ID:         2341713
File Type:       text/plain; charset=US-ASCII
File Size:       15699
Authentication:  anonymous
Direct Access?   false

    Terms of Use apply.

Access Services:
  serviceName            serviceArgs     contentType
1  termsofuse        TermsOfUse=true      text/plain
2   bundleTOU package=WithTermsOfUse application/zip
                                                 serviceDesc
1 Terms of Use/Access Restrictions associated with the data file
2              Data File and the Terms of Use in a Zip archive
```

If allowed by Terms of Use, dvDownload can be used to download a file directly into memory. Otherwise the output of dvDownload will advise the user to download the data through a browser using the URI returned by dvExtractFileIds:

```
> dvDownload(files$fileId[1])
Error in dvDownload(files$fileId[1]) :
    Terms of Use apply.

Data cannot be accessed directly...
try browsing URI from dvExtractFileIds(dvMetadata())

> files$URI[1]
[1] http://thedata.harvard.edu/dvn/dv/leeper/FileDownload/study2-
replication-analysis.r?fileId=2341713
```

Depositing data with dvn

While searching for extant data files is a helpful feature to have within R, the **dvn** wrappers for the Data Deposit API offer much more useful tools for archiving a reproducible research project. The basic workflow involves functions to:

1. create a study using appropriate metadata (dvCreateStudy)
2. add one or more files to that study (dvAddFile)
3. release the study to the public (dvReleaseStudy)

[22]Another wrapper extracts Terms of Use from DDI metadata and opens those terms of use as a temporary HTML file: dvTermsOfUse(dvMetadata(objectid)).

Use of the Data Deposit functions requires authentication, which is currently provided by HTTP basic authentication, relying on the username and password associated with a Dataverse Network user account. The username and password can be included atomically in each function call or stored globally (to conserve typing and allow the sharing of code without credentials embedded in each function call):

```
options(dvn.user = "username")
options(dvn.pwd = "password")
```

As a simple check of authentication, one can call `dvServiceDoc()` with no arguments, which (if authentication is successful) returns a brief statement confirming the Dataverse implementation and user account in use along with a list of available dataverse collections, or otherwise reports an HTTP 403 (Forbidden) code.

To create a study, we simply need to call `dvCreateStudy` with some associated metadata in Dublin Core format. We can either build a metadata file manually or use `dvBuildMetadata` to build that XML for us. The only required field is "title." The built metadata can then be supplied to `dvCreateStudy`, additionally specifying a named dataverse to which the user has access:[23]

```
m <- dvBuildMetadata(title = "My Study")
s <- dvCreateStudy("mydataverse", m)
```

If successful, the response is a simple summary of the created study. Included in the response is the new study's `objectId`, which uniquely identifies the study. We can use either the `objectId` or the return value of `dvCreateStudy` (an object of class `"dvStudyAtom"`) to add files and release the study.

We can add files in a number of ways and in a number of formats. For example, we can add one or more files simply by specifying their filenames in `dvAddFile`. The filename is used to list the file in the study. We can also send multiple files in a compressed zip directory, which The Dataverse Network application will automatically unpack.[24] Another option is to directly send a named dataframe directly from memory:

```
# add files and release study using `objectid`
dvAddFile(s, "mydata.zip")
# or add multiple files:
dvAddFile(s, c("file1.csv", "file2.txt"))
# or add R dataframes as files:
mydf <- data.frame(x = 1:10, y = 11:20)
dvAddFile(s, dataframe = "mydf")
```

Adding files is non-destructive, so we can add files all at once in a single .zip directory or we can add them sequentially. The Dataverse Network does not overwrite an existing file. Attempting to do so will result in an error and `dvAddFile` will return `NULL`. If a file fails to upload successfully, `dvAddFile` will return an error, and if a data file uploads successfully but then fails to decompress or is in an unrecognizable format, you will receive an email to the account affiliated with your dataverse informing you of the failure.

With metadata and files added, we can release the study using `dvReleaseStudy`. Before doing so, it may be reasonable to check the study using `dvStudyStatement` to see a short summary of study contents:

```
> dvStudyStatement(s)
Study author:  Unknown
Study title:   My Study
ObjectId:      doi:10.7910/DVN/ILAJO
Study URI:     https://thedata.harvard.edu/dvn/api/data-deposit/v1/swordv2/edit/study/
               doi:10.7910/DVN/ILAJO
Last updated:  2013-12-02T09:56:07.298Z
Status:        DRAFT
Locked?        false
Files:
  src
1 https://thedata.harvard.edu/dvn/api/data-deposit/v1/swordv2/edit-media/file/2349641/
  mydf.tab
  type                         updated                  fileId
1 text/tab-separated-values    2013-12-02T09:56:16.227Z 2349641
```

[23] Access can be checked by examining the output of `dvServiceDoc`.

[24] An optional `category` argument additionally allows you to group files into categories, such as "data" and "code."

The above example output shows a variety of information, including the lack of a study author, the state of the study as `DRAFT` (meaning the current version is unreleased), and a list of the included files (we see the `mydf` object uploaded above). If satisfied, we can release the study:

```
dvReleaseStudy(s)
```

When a study is released, it becomes immediately accessible via its persistent Handle or DOI. If changes are made to the dataverse (either metadata or files), these remain in draft form until the dataverse is released again. Each release is persistently available unless the entire study collection is deleted.[25]

Before releasing a study, we can delete it from The Dataverse Network server using `dvDeleteStudy`. Once a study is released, it cannot be deleted, but access to it can be revoked. Thus, calling `dvDeleteStudy` only deletes studies that are unreleased. Calling `dvDeleteStudy` on a released study revokes public access, but a persistent page will remain at the study URL noting that a study was previously released. Similarly, it is possible to delete files using `dvDeleteFile` with an argument specifying the `fileId` (which we can extract from `dvStudyStatement`):

```
> d <- dvStudyStatement(s)
> d$files$fileId
[1] "2349642" "2349641"
> dvDeleteFile("2349642")
Operation appears to have succeeded.
[1] ""
```

If a file deletion is successful, **dvn** will report a success message and `dvDeleteFile` will return an empty character string.[26]

We can also modify a study's metadata using `dvEditStudy`, supplied with a metadata listing (e.g., as returned by `dvBuildMetadata`).[27]

```
> m2 <- dvBuildMetadata(title = 'An Actual Title', creator = 'Me', date = '2013')
> dvEditStudy(s, m2)
Citation:      Me, 2013, "An Actual Title", http://dx.doi.org/10.7910/DVN/ILAJO V1
ObjectId:      doi:10.7910/DVN/ILAJO
Study URI:     https://thedata.harvard.edu/dvn/api/data-deposit/v1/swordv2/edit/study/
               doi:10.7910/DVN/ILAJO
Generated by:  http://www.swordapp.org/ 2.0
```

The response reflects the updated metadata. If changes are made to a study (files are added or deleted, or metadata is modified), those changes will be visible via **dvn** as a `DRAFT` study version, but will not be public. To make them public, simply release the study again with `dvReleaseStudy`. Both study versions will remain accessible via The Dataverse Network web browser interface.

Once we've created one or more studies (regardless of whether they are released), we can view them using `dvUserStudies` for a named dataverse or from a `dvServiceDoc` response:

```
dvUserStudies("mydataverse")
# or:
dvUserStudies(dvServiceDoc())
```

Thus while we only need three core functions (`dvCreateStudy`, `dvAddFile`, and `dvReleaseStudy`) to complete a reproducible research workflow, the other functions supplied by **dvn** allow a great degree of control over the appearance and contents of a study.

Conclusion

As Gandrud (2013b) has demonstrated, R is a powerful tool in the reproducible research workflow. Yet a missing element of that workflow to-date has been the easy ability to store research files in a persistent, public data repository designed for reproducible research. The Dataverse Network is a free, open-source service explicitly dedicated to permanently storing data, with implementations hosted by

[25]Note, however, that The Dataverse Network APIs do not currently offer access to previous study versions, though they are accessible via the web interface.

[26]Note, however, that attempting to delete a file that does not exist returns no error message from the SWORD server.

[27]Note, however, that this function completely overwrites all metadata in a study, effectively deleting the current metadata and repopulating it with the supplied metadata.

a variety of institutions (mostly research universities) offering additional choice over whom to trust with the persistent storage of study records. By automatically providing study records a unique URI, data stored using The Dataverse Network has a version-controlled and persistent identifier that can be cited by subsequent users of the data, continuing the reproducible workflow process beyond the phase of data production and initial analysis. With the popularity of The Dataverse Network increasing, it is a logical choice for archiving one's reproducible data and analysis files. **dvn** thus provides R users with the tools necessary to quickly and easily integrate data archiving into the reproducible research workflow.

Thomas J. Leeper
Department of Political Science and Government
Aarhus University
Denmark
thosjleeper@gmail.com

Bibliography

M. Altman. A fingerprint method for scientific data verification. In T. Sobh, editor, *Advances in Computer and Information Sciences and Engineering*, chapter 57, pages 311—-316. Springer Netherlands, Netherlands, 2008. ISBN 978-1-4020-8740-0. doi: 10.1007/978-1-4020-8741-7_57. URL http://link.springer.com/chapter/10.1007/978-1-4020-8741-7_57. [p2]

M. Altman and G. King. A proposed standard for the scholarly citation of quantitative data. *D-Lib Magazine*, 13(3/4):1, 2007. doi: doi:10.1045/march2007-altman. URL http://www.dlib.org/dlib/march07/altman/03altman.html. [p2]

M. Altman, L. Andreev, M. Diggory, G. King, A. Sone, S. Verba, D. L. Kiskis, and M. Krot. A digital library for the dissemination and replication of quantitative social science research: The virtual data center. *Social Science Computer Review*, 19(4):458–470, 2001. [p1]

C. Boettiger, S. Chamberlain, K. Ram, and E. Hart. *rfigshare: an R interface to figshare.com*, 2013. URL http://cran.fhcrc.org/web/packages/rfigshare/index.html. R package version 0.2-6. [p3]

M. A. Boyer. Symposium on replication in international studies research. *International Studies Perspecives*, 4:72–107, 2003. [p1]

S. Chamberlain, C. Boettiger, and K. Ram. *rdryad: Interface to the API for Dryad*, 2013. URL http://cran.r-project.org/web/packages/rdryad/index.html. R package version 0.1.1. [p3]

C. Gandrud. Github: A tool for social data development and verification in the cloud. *The Political Methodologist*, 20(2):7–16, 2013a. URL http://polmeth.wustl.edu/methodologist/tpm_v20_n2.pdf. [p1]

C. Gandrud. *Reproducible Research with R and RStudio*. Chapman and Hall/CRC, Boca Raton, FL, 2013b. ISBN 9781466572843. [p1, 6]

R. Gentleman and D. Temple Lang. Bioconductor project statistical analyses and reproducible statistical analyses and reproducible. Unpublished paper, 2004. [p2]

K. Hornik. *OAIHarvester: Harvest Metadata Using OAI-PMH v2.0*, 2013. URL http://cran.r-project.org/web/packages/OAIHarvester/index.html. R package version 0.1-6. [p3]

G. King. Replication, replication. *PS: Political Science & Politics*, 28(3):444–451, Sept. 1995. [p1]

G. King. An introduction to the Dataverse Network as an infrastructure for data sharing. *Sociological Methods & Research*, 36(2):173–199, Nov. 2007. ISSN 0049-1241. doi: 10.1177/0049124107306660. URL http://smr.sagepub.com/cgi/doi/10.1177/0049124107306660. [p1]

T. J. Leeper. *UNF: Tools for creating universal numeric fingerprints for data*. Aarhus University, Aarhus, Denmark, 2013. URL https://github.com/leeper/UNF. R package version 0.1. [p2]

F. Leisch. Sweave: Dynamic generation of statistical reports using literate data analysis. *Compstat 2002 - Proceedings in Computational Statistics*, (69):575—-580, 2002. URL http://www.stat.uni-muenchen.de/~leisch/Sweave/. [p1]

K. Ram. *rDrop: Programmatic interface to Dropbox*, 2012. URL https://github.com/karthik/rDrop. R package version 0.3-0. [p1]

C. Scheidegger. *rgithub: R bindings for the Github API*, 2013. URL https://github.com/cscheid/rgithub. R package version 0.9.5. [p1]

V. Stodden. Enabling reproducible research: Licensing for scientific innovation. *International Journal of Communications Law & Policy*, 13:1–25, 2009. [p2]

D. Temple Lang. *RCurl: General network (HTTP/FTP/...) client interface for R*, 2012a. URL http://cran.r-project.org/web/packages/RCurl/index.html. [p2]

D. Temple Lang. *XML: Tools for parsing and generating XML within R and S-Plus*, 2012b. URL http://cran.r-project.org/package=XML. [p2]

Y. Xie. knitr: A general-purpose package for dynamic report generation in r, 2013. URL http://cran.r-project.org/web/packages/knitr/index.html. [p1]