Dynamic Documents with R and knitr - Reproducible Research

Pedro J. Vargas B.
Maestría en Ciencias de la Información y las Comunicaciones
Ingeniero de sistemas
Universidad Distrital
Bogotá
Email: pjvargasb@correo.udistrital.edu.co

***Resumen*—The chapter notes that the results of scientific research must be reproducible to be trustworthy. It is unlikely to sustain a finding of credibility as a result of an isolated, as when a researcher specify only laboratory can produce results in the conditions a specific day, and nobody else can produce the same results.**

***Keywords*—R, Reproducible Research, Dynamic Documents, código.**

I. Introducción

En el capítulo se menciona que la investigación Reproducible (RR por sus siglas en inglés – Reproducible Research) es un posible subproducto de la dinámica documentos, pero los documentos dinámicos no garantizan absolutamente RR. El hecho de que sea posible la generar un documento o informe dinámico, es debido a que no existe intervención de personas durante su generación, de ser así sería muy probable que se cometieran constantes errores en los resultados. Como es relativamente fácil de preparar el mismo software y entorno de hardware, podemos pensar que contamos con los elementos necesarios para reproducir los resultados. Sin embargo, el significado de reproducibilidad puede estar más allá de reproducir un resultado específico o un informe en particular. Como un ejemplo trivial, uno podría haber hecho un Monte Carlo de simulación con una cierta semilla aleatoria y tener una buena estimación de un parámetro, pero el resultado será en realidad debido a una semilla aleatoria ”suerte”. Aunque podemos reproducir estrictamente la estimación, en realidad no es reproducible en el sentido general. Existen problemas similares en la optimización algoritmos, por ejemplo, diferentes valores iniciales pueden dar lugar a diferentes raíces de la misma ecuación.

En todo caso la generación de dinámica de informes/documentos sigue siendo un paso importante hacia la investigación reproducible.

II. Literatura

Igualmente en el capítulo se dice que el término Investigación Reproducible fue propuesto por primera vez por Jon Claerbout en la Universidad de Stanford (Fomel y Claerbout , 2009) .La idea es que el producto final de la investigación no es sólo el papel en sí , sino que también es el entorno computacional completo utilizado para producir los resultados en el documento, tal como el código y los datos necesarios para la reproducción de los resultados y basándose en la investigación.

Otro personaje del que se hace mención en el documento es Buckheit y Donoho (1995) quien indicó la esencia de la beca de un artículo así: Un artículo sobre la ciencia computacional en una publicación científica no es la beca en sí, es simplemente la publicidad de la beca. La beca real es el entorno de desarrollo de software completo y el conjunto completo de instrucciones que generaron las figuras. Afortunadamente, las revistas se han estado moviendo en esa dirección también. Por ejemplo, Peng (2009) proporcionan instrucciones detalladas a los autores de los criterios de reproducibilidad y la forma de presentar los materiales para reproducir el artículo en la revista Bioestadística. A nivel técnico, RR suele estar relacionada con la programación literaria (Knuth, 1984), un paradigma concebido por Donald Knuth para integrar código de computadora con la documentación del software en un solo documento. Sin embargo, las primeras implementaciones como WEB (Knuth, 1983) y noweb (Ramsey, 1994) no eran directamente adecuadas para el análisis de datos y generación de informes. Hay otras herramientas en este camino de la generación de documentación, como roxygen2 (Wickham et al., 2015), que es una implementación de R Doxygen (van Heesch, 2008). Sweave (Leisch, 2002) fue una de las primeras implementaciones para tratar los documentos dinámicos en R (Ihaka y Gentleman, 1996; R Core Team, 2015). Todavía hay una serie de problemas que no fueron resueltos por las herramientas existentes; por ejemplo, Sweave está estrechamente ligada al látex y difícil de extender. El paquete knitr (Xie, 2015b) se basa en las ideas de las herramientas anteriores con un framework rediseñado, lo que permite un control sencillo y fino de muchos aspectos de un informe.

Una visión general de la programación ilustrada/literaria aplicada a análisis estadístico se puede encontrar en Rossini (2002). Gentleman y Temple Lang (2004) introdujo los conceptos generales de los documentos de programación literaria para el análisis estadístico, con una discusión de la arquitectura de software. Gentleman (2005) es un ejemplo práctico basado en Gentleman y Temple Lang (2004), usando un paquete de R GolubRR para distribuir el análisis reproducible. Baggerly et al. (2004) puso de manifiesto varios

problemas que pueden surgir con la práctica habitual de la publicación de resultados de análisis de datos, lo que puede dar lugar a falsos descubrimientos debido a la falta de detalles sobre la reproducibilidad. (Incluso con conjuntos de datos suministrados). En lugar de separar los resultados de la computación, podemos poner todo en un solo documento (llamado un compendio de Gentleman y Temple Lang (2004)), incluyendo el código de ordenador y narrativas. Cuando compilamos este documento, se ejecutará el código de computadora, que nos da los resultados directamente.

III. Buenas y malas prácticas

En el capítulo se menciona que la clave a tener en cuenta para la investigación reproducible es que otras personas puedan ser capaces de reproducir nuestros resultados, por lo tanto, se debe hacer todo lo posible para hacer que la computación sea portable. Se discuten algunas buenas prácticas para el RR a continuación y se explica por qué puede ser malo no seguirlas.

Gestionar todos los archivos de origen en el mismo directorio y utilizar rutas relativas siempre que sea posible: rutas absolutas pueden romper reproducibilidad, por ejemplo, un archivo de datos como C: /Users/john/foo.csv o /home/joe/foo.csv puede existir solamente en una computadora, y otras personas pueden no ser capaces de leerlo ya que es probable que sea diferente la ruta absoluta en su disco duro. Si mantenemos todo bajo el mismo directorio, se puede leer un archivo de datos con read.csv ('foo.csv') (si está bajo el directorio actual de trabajo) o read.csv ('../data/ foo.csv ') (subir un nivel y encontrar el archivo en el directorio /data); cuando difundimos los resultados, podemos hacer un archivo de todo el directorio (por ejemplo, como un paquete zip).

No cambiar el directorio de trabajo después de que la ejecución ha comenzado: setwd() es la función en R para establecer el directorio de trabajo, y no es raro ver setwd ('C: /paht/to/some/dir') en el código de usuario, que es malo porque no sólo es una ruta absoluta, sino que también tiene un efecto global sobre el resto del documento de origen. En ese caso, hay que tener en cuenta que todas las rutas relativas pueden necesitar ajustes desde que el directorio raíz haya cambiado, y el software puede escribir el resultado en un lugar inesperado (por ejemplo, se espera que las cifras que se genere en el directorio ./figures/, pero son realmente escritos en ./data/figures/ en lugar si nosotros setwd ('./data/')). Si tenemos que establecer el directorio de trabajo absoluto, es mejor hacerlo al comienzo mismo de una sesión de R; la mayor parte de los editores siguen esta regla, y el directorio de trabajo se establece en el directorio del documento de origen antes de que knitr compile documentos. Si es inevitable, o hace que sea mucho más conveniente para usted para escribir código después de establecer un directorio de trabajo diferente, se debería restaurar el directorio más adelante; por ejemplo.,

Compilar los documentos en una sesión de R limpia: objetos R existentes en la sesión actual de R pueden çontaminar"los resultados en la salida. Está bien si escribimos un informe mediante la acumulación de fragmentos de código uno por uno y se ejecuta de forma interactiva para comprobar los resultados, pero al final nos debería compilar un informe en el modo por lotes con una nueva sesión de R por lo que todos los resultados son recién generados a partir del código.

Evitar los comandos que requieren la interacción humana: la intervención humana puede ser muy impredecible; por ejemplo, no se sabe con certeza qué archivo el usuario elegiría si apareciera un cuadro de diálogo que le pidiera seleccionar un archivo de datos. En lugar de utilizar funciones como file.choose () para introducir un archivo a read.table (), se debe escribir el nombre del archivo de forma explícita; por ejemplo, read.table ('a-específica-archivo.txt').

Evitar las variables de entorno para el análisis de datos: mientras que las variables de entorno a menudo se utilizan en gran medida en la programación para fines de configuración, es poco aconsejable utilizarlos en el análisis de datos, ya que requieren instrucciones adicionales para configurar por los usuarios, y los seres humanos simplemente se olvidan de hacer esto . Si hay alguna opción para configurar, es mejor hacerlo dentro del documento fuente.

Adjuntar sessioninfo() (o devtools::sesióninfo()) y las instrucciones de cómo compilar el documento: la información de la sesión hace a un lector consciente del entorno de software, tales como la versión de R, el sistema operativo, y paquetes utilizados. A veces no es tan simple como llamar a una sola función para compilar un documento, y tenemos que dejar claro cómo compilar si se requieren pasos adicionales; pero es mejor para proporcionar las instrucciones en la forma de una secuencia de comandos del ordenador; por ejemplo, un script de shell, un Makefile, o un archivo por lotes.

Estas prácticas no están necesariamente se restringen al lenguaje de R, aunque utilizamos R para los ejemplos, las mismas reglas se aplican también a otros entornos de computación.

Se debe tener en cuenta que las herramientas de programación literaria a menudo requieren que los usuarios recopilen los documentos en el modo por lotes, y es bueno para la investigación reproducible, pero el modo por lotes puede ser engorroso para el análisis exploratorio de datos. Cuando no hemos decidido qué poner en el documento final, es posible que tengamos que interactuar con los datos y el código frecuentemente, y no vale la pena la compilación de todo el documento cada vez que se actualice el código. Este problema puede ser resuelto por un editor capaz como RStudio y Emacs / ESS, que se introducen en el Capítulo 4. En estos editores, se puede interactuar con el código y explorar los datos libremente (por ejemplo, enviar o escribir código R en una sesión R asociada), y una vez que se termine el trabajo de codificación, se puede compilar todo el documento en el modo por lotes para asegurarse de que todo el código funciona en una sesión de R limpia.

IV. Barreras

Se mencionan en el documento algunas barreras prácticas que existen a pesar de todas las ventajas de la RR, y se menciona una lista no exhaustiva de ellas:

Los datos pueden ser enormes: por ejemplo, es común

que la física de alta energía y datos de secuenciación de próxima generación en la biología puedan producir decenas de terabytes de datos, y no es trivial archivar los datos con los reportes y distribuirlos.

Confidencialidad de los datos: se puede estar prohibido para liberar los datos en bruto con el informe, sobre todo cuando se está involucrado con sujetos humanos debido a los problemas de confidencialidad.

Versión del software y configuración: un informe puede ser generado con una versión antigua de un paquete de software que ya no está disponible, o con un paquete de software que compila de manera diferente en diferentes sistemas operativos.

La competencia: uno puede optar por no liberar el código o datos con el informe debido al hecho de que los competidores potenciales pueden conseguir fácilmente todo de forma gratuita, mientras que los autores originales han invertido una gran cantidad de dinero y esfuerzo.

En el capítulo se menciona que desde luego, no se debe esperar que todos los informes en el mundo estén a disposición del público y estrictamente reproducibles, pero es mejor compartir código aún mediocre, dañado o datasets problemáticos o defectuosos que no compartir nada en absoluto. En vez de persuadir a la gente a realizar RR por las políticas, se puede tratar de crear herramientas que hagan más fácil RR que cortar y pegar, y knitr es una buena alternativa. El éxito de RPubs (http://rpubs.com) es evidencia de que una herramienta fácil puede promover rápidamente RR, ya que los usuarios disfrutan de su uso. Los lectores pueden encontrar cientos de informes aportados por los usuarios en el sitio web RPubs. Es bastante común ver las tareas de estudiantes y ejercicios allí, y una vez que los estudiantes son entrenados de esta manera, se puede esperar que la investigación científica más reproducible en el futuro.

V. Conclusiones

La programación literaria permite que los usuarios puedan modificar y probar con mayor facilidad las hipótesis de los proyectos con las modificaciones que consideren en sus proyectos; se resalta la importancia y las ventajas de utilizar la programación literaria para el desarrollo de informes, papers y otra documentación utilizando LaTeX; la programación literaria permite compartir los resultados de investigación para ser verificados con otros valores en sus variables.