Proyectos de RStudio y directorios de trabajo: una guía para principiantes

Escrito el 23 de enero de 2020
10 min de lectura

aprendizaje-r viñetas

Introducción

Esta publicación proporciona una introducción básica sobre cómo usar RStudio Projects y estructurar sus directorios de trabajo, ¡lo cual vale la pena leer si todavía está usando `setwd()` para configurar sus directorios!

Aunque el directorio de trabajo de R es un tema bastante básico y bastante bien cubierto, sentí que todavía valdría la pena compartir mi propio enfoque de estructurar directorios de trabajo, ya que claramente puede haber múltiples formas sensatas y válidas de estructurar un directorio de trabajo. La estructura del directorio del proyecto que se cubre en esta publicación es la que uso yo mismo día a día, y la que considero la más adecuada para el tipo de trabajo de análisis con el que normalmente me ocupo, es decir, conjuntos de datos cargados en la memoria y guardados en El directorio de trabajo en sí.

Si recién está comenzando en R, mi consejo personal es que el uso de proyectos de RStudio y la estructuración de directorios de trabajo son "imprescindibles". El uso de proyectos de RStudio elimina gran parte de la molestia y la confusión de la etapa inicial de leer y exportar datos. Configurar un directorio de trabajo correctamente también ayuda a desarrollar buenos hábitos que conducen a un análisis reproducible. Es una de las partes no relacionadas con el código de la programación de R que creo que es extremadamente útil saber, y posiblemente para un alumno, ¡incluso una mayor prioridad que aprender a usar GitHub! 1

¿Qué es un proyecto RStudio y por qué?

Cuando comencé a usar R hace varios años, se usaba el libro de texto y el enfoque principal para configurar directorios de trabajo `setwd()`, que toma una ruta de archivo *absoluta* como entrada y luego la establece como el directorio de trabajo actual del proceso de R. Luego, utiliza `getwd()` para averiguar cuál es el directorio de trabajo actual y verifica que su directorio de trabajo esté configurado correctamente.

El problema con este enfoque es que, dado que se `setwd()` basa en una ruta de archivo *absoluta* , esto hace que los enlaces se rompan muy fácilmente y es muy difícil compartir su análisis con otros. Una simple acción de mover todo el directorio a una subcarpeta diferente o a una unidad diferente romperá los enlaces y su script no se ejecutará. Como señala Jenny Bryan , el `setwd()` enfoque hace que sea prácticamente imposible para cualquier otra persona que no sea el autor original del script, en su computadora, hacer que las rutas de archivo funcionen:

> *La posibilidad de que el `setwd()` comando tenga el efecto deseado (hacer que las rutas de archivo funcionen) para cualquier persona además de su autor es 0%. También es poco probable que funcione para el autor dentro de uno o dos años o para computadoras a partir de ahora. El proyecto no es autónomo y portátil. Para recrear y tal vez extender esta trama, el destinatario afortunado deberá editar manualmente uno o más caminos para reflejar dónde ha aterrizado el proyecto en su máquina. Cuando haces esto por 73ª vez en 2 días, mientras marcas una tarea, comienzas a fantasear con encender la computadora del perpetrador.*

(Mira este enlace para la publicación original del blog)

Al principio era escéptico sobre el movimiento aparentemente radical de abandonar por `setwd()` completo a los ortodoxos, pero desde que probé el flujo de trabajo del proyecto, nunca pensé en volver a usar rutas de archivos absolutas. ¡Así que estoy totalmente con Jenny Bryan en este caso! 2

Fácil referencia de ruta de archivo con proyectos de RStudio

Los proyectos de RStudio resuelven el problema de las rutas de archivos 'frágiles' al hacer que las rutas de archivos sean *relativas* . El archivo de proyecto RStudio es un archivo que se encuentra en el directorio raíz, con la extensión .Rproj. Cuando su sesión de RStudio se ejecuta a través del archivo de proyecto (.Rproj), el directorio de trabajo actual apunta a la carpeta raíz donde se guarda ese archivo .Rproj.

Aquí hay un ejemplo: supongamos que mi directorio de trabajo es una carpeta llamada *SurveyAnalysis1* . En lugar de enumerar la ruta completa del archivo absoluto, *C:/Users/Martin/Documents/Analysis/SurveyAnalysis1/Data/Data1.xlsx* , simplemente puedo referir el mismo archivo de Excel en el nivel de directorio cuando uso proyectos, es decir,

simplemente consulte archivo por *Data* / *Data1.xlsx* . La idea es que si un día decido mover toda mi carpeta / directorio *SurveyAnalysis1* a otra ubicación, o tal vez abrirlo en una computadora diferente, todas las rutas de archivos especificadas en mis scripts R seguirán funcionando siempre que comience la sesión abriendo el archivo .Rproj.

Este archivo .Rproj se puede crear yendo a **Archivo> Nuevo proyecto ...** en RStudio, que luego se asocia con la carpeta o directorio especificado. La mentalidad debería ser que el directorio (la carpeta completa y sus subcarpetas y contenido) es independiente y portátil, lo que en otras palabras significa que no debe leer ni escribir datos en archivos *externos*El directorio. Todo lo relacionado con ese análisis o proyecto solo debe ocurrir dentro de ese directorio, excepto en los casos en que su análisis requiera interactuar con una fuente de Internet, por ejemplo, raspado web, llamadas a API. Al abrir un proyecto existente, primero debe abrir el archivo .Rproj y solo posteriormente abrir los scripts R (extensiones con .R) de la sesión RStudio, en lugar de ir directamente a los scripts R para abrirlos. Puede pensar en abrir el archivo .Rproj como un paso de 'inicialización' para la sesión de RStudio, lo que garantiza que todo lo que ejecute desde esta sesión pueda encontrar las rutas de archivo adecuadas dentro de ese directorio. RStudio tiene una documentación más detallada sobre los proyectos de RStudio que vale la pena consultar, que tiene más información sobre los archivos .RData y .Rhistory. Capítulo 8(*Flujo de trabajo: proyectos*) de *R for Data Science* también ofrece una guía de "inicio rápido" sobre cómo usar los proyectos de RStudio.

Estructurando su directorio de trabajo 🔨

Además de usar proyectos de RStudio, también es una buena práctica estructurar su directorio de una manera que ayude a cualquier persona con la que esté colaborando, o una versión futura de usted que intente reproducir algunos análisis, para navegar el análisis fácilmente. Recomiendo lo siguiente como una configuración básica de directorio 'de inicio':

Estructura basica

En su directorio de trabajo, tendrá lo siguiente:

- **Datos** : esta es la subcarpeta donde guardo los archivos que necesito leer en R para hacer mi análisis o visualización. Estos pueden ser desde archivos SPSS (* .sav), archivos Excel / CSV, archivos .FST o .RDS. La idea clave es que estos son **archivos de datos de origen** , y en ningún momento R debe guardar o editar estos archivos para garantizar la reproducibilidad. El razonamiento es que el análisis reproducible no es realmente posible si el archivo de datos fuente sigue siendo modificado por el análisis (piense en el análisis en hojas de cálculo). Si necesita cambiar el archivo de datos de origen, cree una nueva versión y asegúrese de que el nuevo nombre de archivo refleje adecuadamente ese cambio.
- **Script** : aquí es donde guardo mis scripts R y archivos RMarkdown (archivos con la extensión .R *y* .Rmd).
  - **Análisis** : todos mis scripts de análisis R principales se guardan aquí, lo que creo que es bueno para la mayoría de los propósitos si tiene múltiples scripts que realizan diferentes tareas guardadas aquí. Personalmente no tengo un proyecto por análisis, ya que esto podría salirse de control cuando tenga más de 20 análisis diferentes que me gustaría realizar en un solo conjunto de datos. Mi regla general (en realidad bastante simple) para decidir si separar un análisis es imaginar si alguien completamente nuevo en el proyecto podría navegar y descubrir qué está pasando con este directorio. Como nota al margen, ¡los nombres de archivos bien pensados y sensibles ayudan mucho!
  - **Funciones** : es opcional si tiene sus funciones personalizadas guardadas en una subcarpeta separada. Personalmente, me parece conveniente porque si quiero reutilizar una función que recuerdo haber escrito en un proyecto en particular, puedo examinar rápidamente todas las funciones que he escrito para ese proyecto. Guardar funciones por separado acompaña un flujo de trabajo donde se utilizan `source()` para leer funciones en el 'script de análisis principal', en lugar de tenerlo junto con el análisis principal.
  - **Archivos RMarkdown: los archivos** RMarkdown son un caso especial, ya que funcionan de manera ligeramente diferente a los archivos .R en términos de rutas de archivo, es decir, se comportan como mini proyectos propios, donde el directorio de trabajo predeterminado es donde se guarda el archivo Rmd. Para guardar archivos RMarkdown en esta configuración, se recomienda que use el paquete {here} y su flujo de trabajo. Alternativamente, puede ejecutar `knitr::opts_knit$set(root.dir = "../")` en su fragmento de configuración para que el directorio de trabajo se establezca en el directorio raíz en lugar de en otra subcarpeta donde se guarda el archivo RMarkdown (menos ideal que usar {aquí}). En mi otra publicación, discutí brevemente una estructura de directorio para combinar múltiples archivos RMarkdown en un solo documento RMarkdown largo] (https://martinctc.github.io/blog/first-world-problems-very-long-rmarkdown-documents/)
- **Salida** : guarde todas sus salidas aquí, incluidos gráficos, HTML y exportaciones de datos.
  - Tener esta carpeta de Salida ayuda a otros a identificar qué archivos son **salidas** del código, a diferencia de los archivos de origen que se usaron para producir el análisis.
  - Lo que ha configurado como subcarpetas no importa demasiado, siempre que sean sensatas. Puede decidir configurar las subcarpetas para que se alineen con el análisis en lugar del tipo de exportación de archivos.

- La `timed_fn()` función de mi paquete surveytoolbox (disponible en GitHub) ayuda a crear marcas de tiempo para los nombres de archivo, que utilizo a menudo para asegurarme de no perder el trabajo cuando repito el análisis.

Esta 'plantilla' de estructura de directorio debería proporcionar un buen punto de partida para organizar proyectos si el flujo de trabajo de un proyecto es nuevo para usted. Sin embargo, si bien la coherencia es excelente, los diferentes proyectos tendrán diferentes necesidades y, por lo tanto, siempre se debe pensar en lo que se necesita y lo que sucederá al configurar la estructura del directorio de trabajo, y adaptarse adecuadamente.

Lectura adicional 🕮

Para leer más, Chris Von Csefalvay tiene este gran artículo sobre la estructuración de proyectos de R , que proporciona una guía más detallada sobre lo que debe considerar al estructurar sus proyectos de R. Recomienda también tener un archivo README disponible para cada proyecto (guardado en el directorio raíz), lo cual es particularmente importante para proyectos con más complejidad.

Como de costumbre, los comentarios, comentarios y preguntas son bienvenidos. Si te gusta esta publicación, consulta mis otras publicaciones en https://martinctc.github.io/blog/ .

---

1. Los repositorios de GitHub están estructurados como directorios de trabajo, por lo tanto, tendría sentido aprender cómo estructurar un directorio de trabajo antes de aprender a usar GitHub. ↩

2. *El libro R de Garrett Grolemund* y Hadley Wickham *para ciencia de datos* hace una recomendación similar en el capítulo 8.3 . ↩

Navegar por tema

viñetas tidyverse aprendizaje-r rdatatable sobresalir rmarkdown entrevistas revisiones de paquetes fuente abierta

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS

Name

郭小宏 • 4 days ago

It really helps me.
Thanks a lot.

1 • Reply • Share ›

Martin Chan Mod 郭小宏 • 4 days ago

Very glad to hear that, thanks! :)

• Reply • Share ›

Anthony Schmidt • 5 days ago

Is there a way within a R Project to set a default output sub-directory?

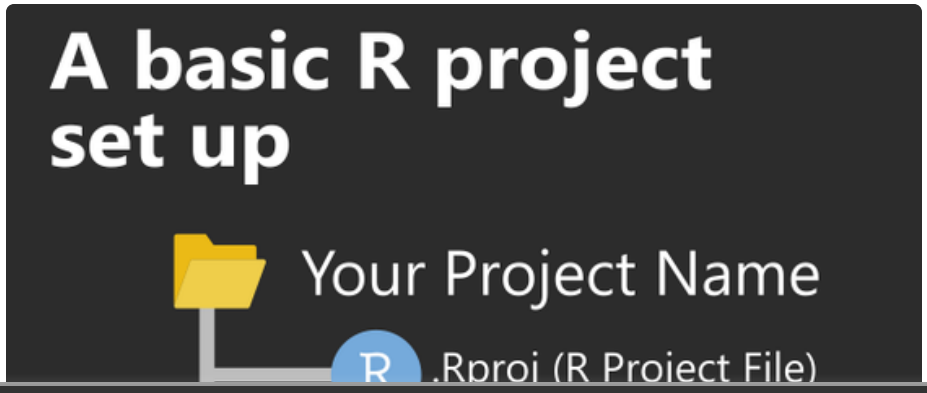
1 • Reply • Share ›

Martin Chan Mod Anthony Schmidt • 4 days ago

Not that I know of, but certainly it would be a very good feature to have or add!

Mara Averick
@dataandme

🗂️ I'm always intrigued by proj setups…
"RStudio Projects and Working Directories: A Beginner's Guide"
👨‍💻 Martin Chan buff.ly/38vwee8 #rstats



see more

• Reply • Share ›

Paulovic • 3 days ago

Thanks! Really useful for beginners like me :)

• Reply • Share ›

infominer • 3 days ago

How did you create the image showing folder structure ?

Martin Chan Mod infominer • 3 days ago

I didn’t actually use R to create this image - I’ve put this together using Inkscape (an open source vector graphics editor) and some source SVG files.

• Reply • Share ›