Un primer ejemplo: procesamiento en paralelo en R y vinculación entre R y C

Carlos E. Rodríguez

8 de mayo de 2018

1. Introducción

R es una herramienta poderosa para realizar análisis estadísticos y también sumamente versátil como lenguaje de programación. Sin embargo, frecuentemente es criticado por su bajo desempeño para realizar procesos iterativos. Por ejemplo, cuando se necesita usar ciclos (e.g. `for` o `while`) en algún proceso puede ser muy lento. Existen varias maneras de solucionar este problema, por ejemplo:

1. **Vectorización**. En R prácticamente todo lo podemos ver como vectores y en lugar de hacer operaciones componente a componente del vector, las hacemos vector a vector. Por ejemplo, si `x` e `y` son vectores (de la misma longitud) para sumarlos, multiplicarlos o dividirlos simplemente hacemos `x + y`, `xy` e `x/y` (cuidando que no haya componentes en el vector `y` iguales a cero). Considerando vectores lógicos se pueden hacer cosas más interesantes.

2. **Uso de funciones eficientes de** R. Las funciones `apply, lappy, sapply, tapply` y `aggregate` son excelentes alternativas para evitar varios ciclos.

3. Mediante la librería `Rcpp` que ofrece una integración "fácil" entre R y C++

4. **Procesamiento en paralelo**. Varias librerías que ofrecen la oportunidad de realizar procesamiento en paralelo, siempre y cuando se tenga acceso a una máquina con varios procesadores. Algunas librerías son `doParallel`, `foreach`, `partools`, etc.

5. **Vinculación de** R **con** C. Cuando se tiene la habilidad de programar en C y se busca contar con una interfaz amigable de entrada y salida de resultados, esta es una buena opción. En este caso, lo que se hace es programar todo el algoritmo en C y simplemente se manda llamar este código desde R.

En estas notas daremos una breve introducción a las alternativas 4 y 5.

1.1. Estrategia

Lo más sencillo para aprender a realizar la vinculación entre R y C, así como implementar el procesamiento en paralelo, es trabajar un problema concreto y programarlo:

1. Completamente en R.

2. Implementando el procesamiento en paralelo.

3. Realizando la vinculación entre `R` y `C`.

Para finalmente comparar la eficiencia de cada versión tomando como referencia su tiempo de procesamiento.

2. Problema de juguete

En esta sección describiremos el problema que resolveremos mediante las tres estrategias descritas anteriormente.

2.1. Regresión Lineal Simple

Trabajaremos en el contexto de la Regresión Lineal Simple (RLS), en donde el modelo esta dado por

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i \text{ para, } i = 1, 2, \ldots, n. \tag{1}$$

Los supuestos básicos de la RLS son

- Existe una relación lineal entre x e y.
- $\epsilon_i \sim N(0, \sigma^2)$, para $i = 1, 2, \ldots, n$.
- Los errores son independientes entre sí.

Estos supuestos generalmente se "verifican" mediante diagnósticos gráficos.

Para describir a nuestra variable dependiente y, utilizando la información de x y el modelo de RLS, primero necesitamos estimar los coeficientes de la regresión β_0 y β_1. Lo anterior lo hacemos mediante la técnica conocida como mínimos cuadrados para obtener estimadores:

$$\begin{aligned} \widehat{\beta}_1 &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}, \\ \widehat{\beta}_0 &= \bar{y} - \widehat{\beta}_1 \bar{x}. \end{aligned}$$

Entonces, podemos calcular

- Los valores estimados $\widehat{y}_i = \widehat{\beta}_0 + \widehat{\beta}_1 x_i$, para $i = 1, \ldots, n$
- Los residuales $\widehat{\epsilon}_i = y_i - \widehat{y}_i$, para $i = 1, \ldots, n$
- El estimador de σ^2, dado por

$$\widehat{\sigma}^2 = \frac{\sum_{i=1}^n \widehat{\epsilon}_i^2}{n-2}$$

2.1.1. Predicción de nuevas observaciones

Cuando nuestro objetivo es predecir un nuevo valor y_*, que no tenemos, utilizando otro valor x_*, que sí tenemos, entonces calculamos

$$\widehat{y}_* = \widehat{\beta}_0 + \widehat{\beta}_1 x_*$$

Pero como buenos estadísticos no nos conformamos con la estimación puntual, lo que buscamos es un intervalo de confianza para tener una noción de la incertidumbre en nuestra predicción. Entonces, bajo los tres supuestos descritos anteriormente, es posible obtener

$$\widehat{y}_* \mp t_{1-\alpha/2,n-2}\ \widehat{\sigma}\ \sqrt{1 + \frac{1}{n} + \frac{(x_* - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}}$$

el intervalo de predicción del $(1-\alpha)\times 100\,\%$ de confianza para y_*.

3. Ejemplo

3.1. Todo usando R y asumiendo que se cumplen los supuestos de la RLS

3.2. Verificación de los supuestos

3.3. Re-muestreo Bootstrap para realizar predicción en RLS

En este caso eliminamos el supuesto de normalidad de los errores y por lo tanto no tenemos una expresión cerrada para la varianza de $\widehat{y}_*$. Lo que necesitamos estimar sería

$$\widehat{\epsilon}_* = y_* - \widehat{y}_*$$

el problema es que no conocemos y_*.

La estrategia es la siguiente:

$$\begin{aligned} \widehat{\epsilon}_* &= y_* - \widehat{y}_* && (2)\\ &= \beta_0 - \beta_1 x_* + \epsilon_* - (\widehat{\beta}_0 + \widehat{\beta}_1 x_*) && (3)\\ &= \beta_0 - \widehat{\beta}_0 + (\beta_1 - \widehat{\beta}_1) x_* + \epsilon_* && (4) \end{aligned}$$

Obviamente no conocemos a β_0, β_1 y ϵ_*, así que los re-muestreamos vía Bootstrap y sustituyendo obtenemos una muestra Bootstrap de $\widehat{\epsilon}_*$.

3.4. Programado en R

3.5. Programado en R implementando procesamiento en paralaleo

4. Alistando Windows para la vinculación entre R y C

Vincular R con C en computadoras con sistemas operativos iOS y LINUX es muy sencillo, ya que generalmente traen un compilador de C integrado (o es muy fácil de dscargar e instalar) y uno puede complilar el código fácilmente. En cambio, en Windows este no es el caso y es necesario realizar un procedimiento preliminar para poder hacer la vinculación.

Primero hay que descargar varios programas:

- **Rtools**. Este software nos permitirá compilar el código de C desde la consola de comandos de Windows (MS-DOS). Para descargarlo, hay que ir a la siguiente dirección:

  https://cran.r-project.org/bin/windows/Rtools/

  descargar e instalar la versión más reciente.

Opcionalmente se pueden descargar los paquetes

- **MikTeX**. Para poder escribir documentos en LaTeXdirectamente desde R.

  https://miktex.org/

  descargar e instalar la versión más reciente.

- **Atom**. Es un editor general para varios lenguajes de programación, nos servirá para poder programar en C. Si ya tienen algún editor de su agrado, este paso no será necesario. En el peor de los casos pueden utilizar el bloc de notas de Windows para hacer sus programas en C.

  https://atom.io/

  descargar e instalar la versión más reciente.

Una vez instalados los programas, es necesario verificar que en la instalación se haya creado una variable de entorno. Esto se hace de forma automática, pero es mejor verificar que todo este en orden.

1. Vayamos a Panel de control → Sistema → Configuración avanzada del sistema.
2. El proceso anterior abre la ventana "Propiedades del sistema", aquí hay que buscar la pestaña Opciones avanzadas dar clic en Variables de entorno.
3. En variables del sistema, hay que buscar y seleccionar la variable con nombre "Path" da click en editar y verificar que en toda la cadena de direcciones se encuentren las siguientes (la última es opcional, en caso de que se quieran generar documentos en LaTeXdesde R)

```
Path = C:\Rtools\bin;
       C:\Rtools\mingw_32\bin;
       C:\Program Files\R\R-3.5.0\bin\i386;
       C:\Program Files\MiKTeX 2.9\miktex\bin\
```

   En caso de que estén incluidas, no hay que hacer nada. Pero de no ser el caso, habrá que incluirlas verificando las rutas de cada programa. En caso de que se actualice R será necesario editar la variable "Path" para que coincida con la nueva ruta (lo mismo con MikTeX).

   En el ejemplo anterior el sistema operativo utilizado fue Windows 7 de 32 bits, en el caso de sistemas operativos de 64 bits será necesario considerar los cambios correspondientes.