Uso de ImageJ y R para la digitalización de datos de ubicación de árboles en inventarios forestales

Convertir la ubicación de los árboles de coordenadas en píxeles a coordenadas locales

José Ulises Jiménez S., Xiomara Quintanar y Julissa Guevara

20 de mayo de 2022

Índice

Introducción

Este tutorial muestra el uso de código R para leer los mapas de cuadrantes (formularios croquis) en inventarios de árboles. Primero, se presenta la digitalización de píxeles, de una imagen escaneada del croquis dibujado en campo, con el programa `ImageJ` (Schneider et al., 2012); y segundo, la conversión de los píxeles digitalizados en coordenadas dentro de un cuadrante o lo que se conoce como coordenadas locales. Aquí, se mostrará paso a paso una técnica computacional para obtener los datos de ubicación de los árboles inventariados en cuadrantes de 20 m x 20 m utilizando código R[1] (R Core Team, 2021).

Este material educativo se basa en el tutorial *Converting ImageJ output to local plot coordinates using R*, que pueden encontrar pulsando aquí: **enlace** (Condit, 2011). Con este manual queremos comunicar y transferir la capacidad adquirida a través de los cursos talleres impartidos por el Dr. Richard Condit. En el 2013, el Dr. Condit lideraba el Centro de Ciencias de los Bosques Tropicales[2] (CTFS, por sus siglas en inglés) y nos invitó a participar de los cursos sobre el uso de R que se impartían en el Centro Tupper del STRI (*Smithsonian Tropical Research Institute*). También, nos visitó en dos ocasiones en la Universidad Tecnológica de Panamá (UTP) sede Tocumen para mostrarnos el uso de algunas funciones del `paquete CTFS R` para la estimación de la biomasa sobre el suelo a partir de datos de inventarios forestales.

Aprovechamos para agradecer al Dr. Condit, Rolando Pérez, Salomón Aguilar, Suzanne Lao y a todo el equipo de la CTFS del Instituto Smithsonian de Investigaciones Tropicales. Muchas gracias por mostrarnos

[1] es un entorno de software libre y lenguaje de programación interpretado, comúnmente utilizado para la computación estadística y gráfica.

[2] Actualmente, se conoce como ForestGEO, Observatorio Forestal Mundial de la Tierra. ForestGEO es una red global de sitios de investigación forestal y científicos dedicados al estudio de la función y diversidad de los bosques tropicales y templados.

la metodología y los protocolos de la CTFS para la adquisición de datos en inventarios forestales. Esto nos ayudó en los trabajos que realizábamos sobre la primera parcela permanente de muestreo de una hectárea establecida por el Centro de Investigaciones Hidráulicas e Hidrotécnicas de la Universidad Tecnológica de Panamá, ubicada en Cerro Pelado, Gamboa, Provincia de Colón. Esta experiencia nos permitió aprender sobre la entrada de los datos, la creación de la base de datos y la aplicación de las funciones básicas del paquete CTFS R para los análisis de estimación de la biomasa sobre el suelo y carbono.

El objetivo de este manual es transferir esos conocimientos adquiridos y aportar a la formación de los estudiantes de la carrera de Ingeniería Forestal de la UTP mostrando una técnica para el tratamiento de datos de ubicación de árboles en inventarios forestales. Comencemos.

Parcelas de muestreo

En el contexto de la estadística, en los estudios vegetacionales la unidad básica de muestreo se asocia comúnmente a un área determinada, por ejemplo: parcelas (rectangulares incluye cuadradas). A veces, la unidad es suficientemente grande para dividirla en cuadrantes y subcuadrantes. Los cuadrantes son la principal unidad organizativa de la parcela. Cada árbol marcado se cartografía en relación con la distancia a las líneas de las columnas y las filas adyacentes en el cuadrante usando el formulario croquis. De un conjunto de unidades de muestreo se puede obtener: la densidad de árboles, el área basal, el número de especies, la biomasa; todas expresadas por unidad de área de la superficie del suelo. Los árboles individuales son las unidades de medición dentro de la parcela y forman el conjunto de observaciones. Generalmente, en los inventarios de árboles se toman las medidas de: diámetro normal, altura total, localización dentro del cuadrante; para cada individuo a partir de un tamaño de diámetro determinado. También, se les identifica taxonómicamente. De esta información se obtienen las características por especie de: abundancia, frecuencia y dominancia. Sobre el tema de diseño de la parcela y el muestreo sugerimos revisar Dallmeier (1992), Klein y Morales (2002), Alder y Synnott (1992), Camacho Calvo (2000) y Condit (2008).

Instalación del programa ImageJ

Para instalar ImageJ visite la página https://imagej.nih.gov/ij/download.html. Descargue la carpeta .zip para `Windows`. El programa es portátil y solo tiene que descomprimir la carpeta.

Adicionalmente, se necesita un complemento llamado *pointpicker* que está modificado especialmente para esta tarea. El complemento *pointpicker* modificado, está en archivo comprimido, se descomprime y se copia en la carpeta `plugins` dentro de la carpeta *ImageJ*. Luego se abre a aplicación *ImageJ*, se hace clic en *Help* del menú principal y se hace clic en *Refresh Menus*. El complemento *pointpicker* modificado nos fue proporcionado por el Dr. Condit.

Digitalizar los croquis

Los formularios (croquis) se escanean como archivos de imagen .jpg. Para comenzar se abre la aplicación `ImageJ` y se va al menú principal, se hace clic en *File* y clic en *Open* y se escoge para abrir la imagen de donde va a extraer las coordenadas de ubicación de los árboles en píxeles. Se sugieren utilizar nombres para los croquis de las parcelas que denota el número del cuadrante. Los nombres *map_0001.jpg* y *quadrat_0001.jpg* son ejemplos que siguen la convención de nomenclatura estándar sobre cómo debe guardarse el archivo de texto para ser utilizado por la función `fullplot.imageJ()`.

Una vez abierta la imagen `.jpg` en *ImageJ*, hay que seleccionar la opción de menú *PointPicker* de la siguiente manera:

`Plugins → pointpicker → PointPicker`

Se hace clic en *Añadir cruces* en el menú de *pointpicker* (la primera opción con una punta de plumilla y el signo más) y con el ratón sobre la imagen se hace clic para añadir puntos a las cuatro esquinas del mapa y también, para añadir puntos sobre el centro mismo de todos los tallos dibujados y etiquetados en el croquis.

Main tree inventory: quadrat map

Plot: _____ **Quadrat:** _____ **Start date:** _____ **End date:** _____

Field crew: _____

Supervisor: _____ Data Entered by: _____ Date: _____

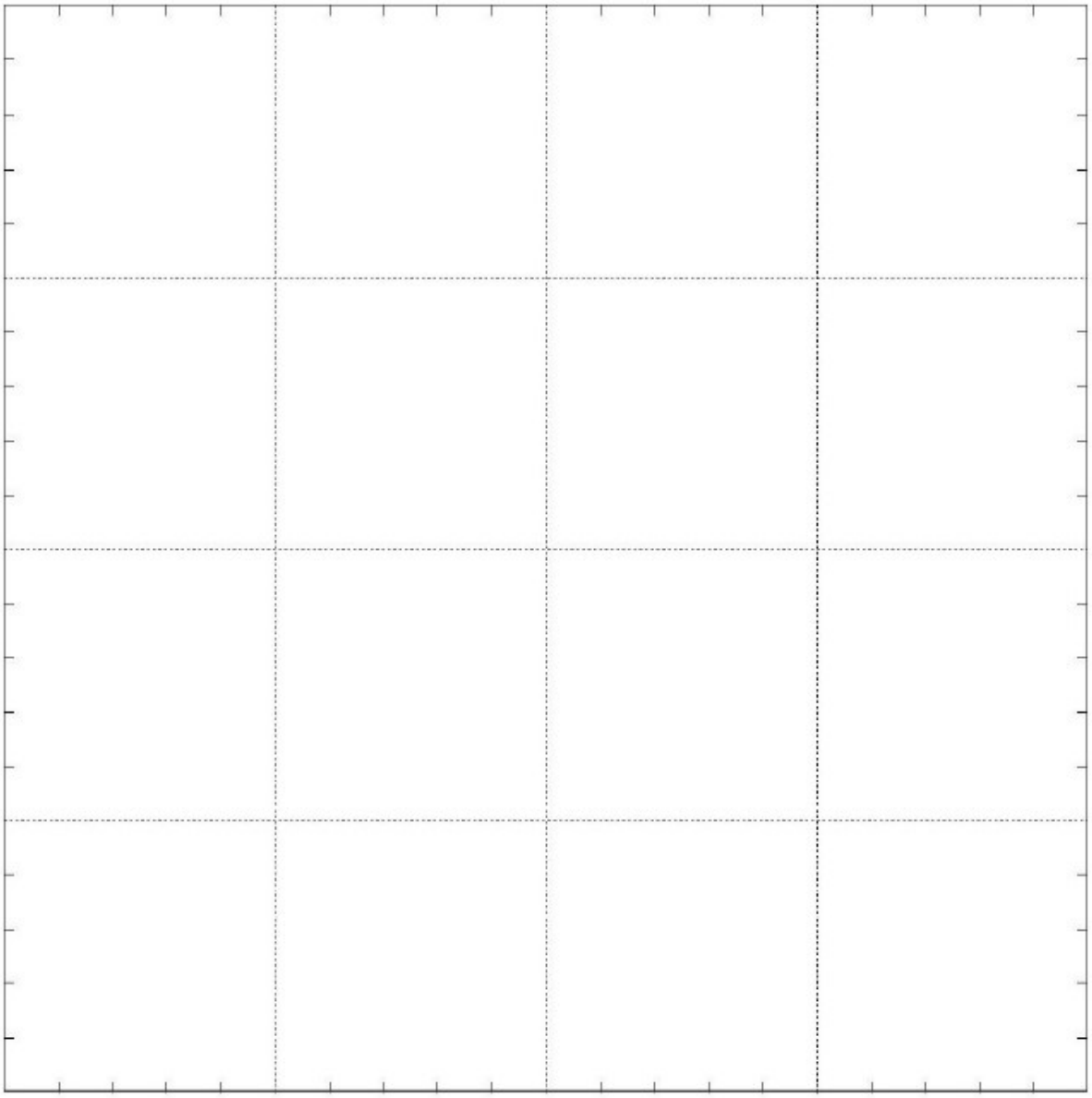

Figura 1. Formulario croquis. Extraído de Condit (2008).

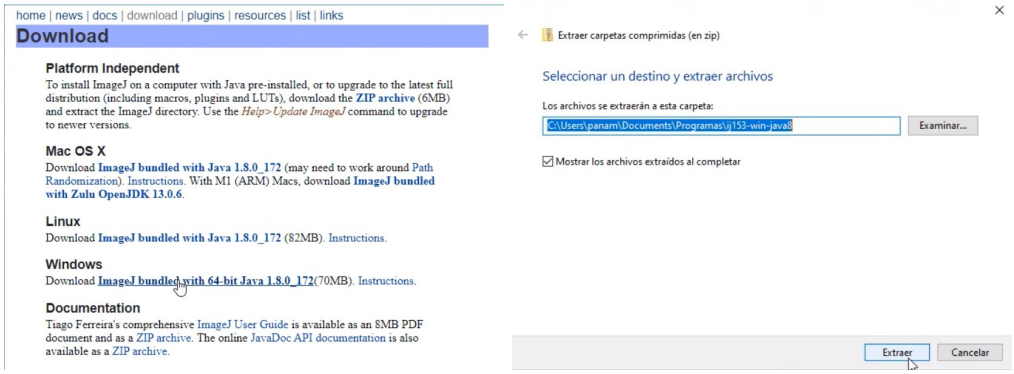

Figura 2. Instalación de ImageJ.

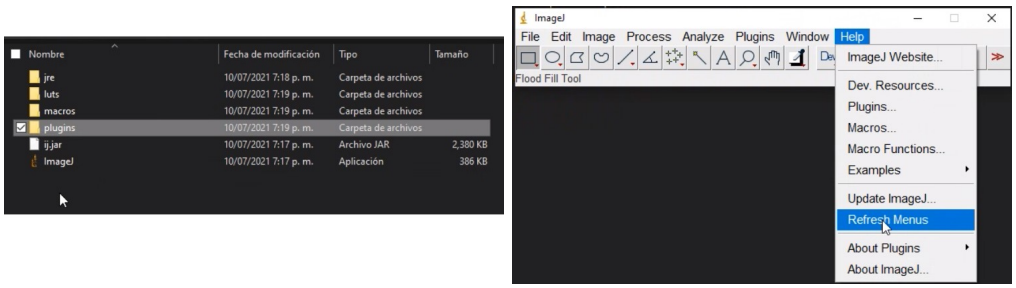

Figura 3. Instalación de pointpicker.

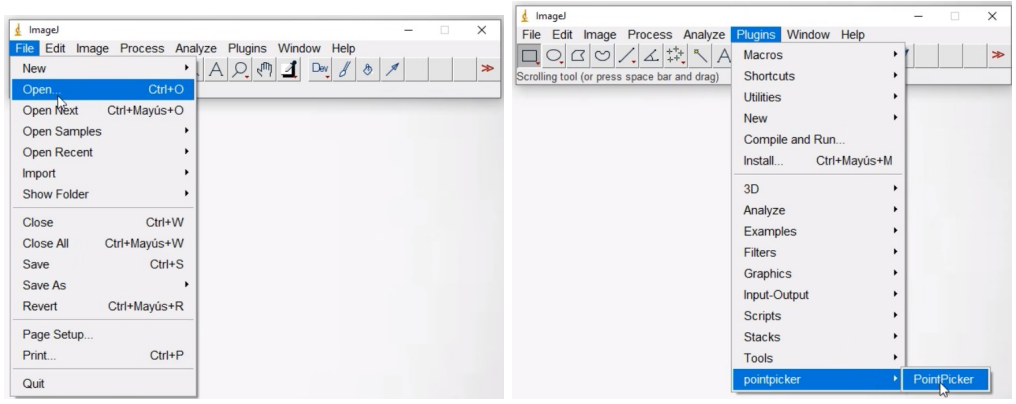

Figura 4. Abrir la aplicación y escoger imagen.

Antes de añadir los puntos (cruces), se recomienda examinar la copia original en papel del mapa dibujado a mano junto con la hoja de datos correspondiente para asegurarse de que todos los tallos están realmente ubicados en el cuadrante, y que todos los números de etiqueta concuerden.

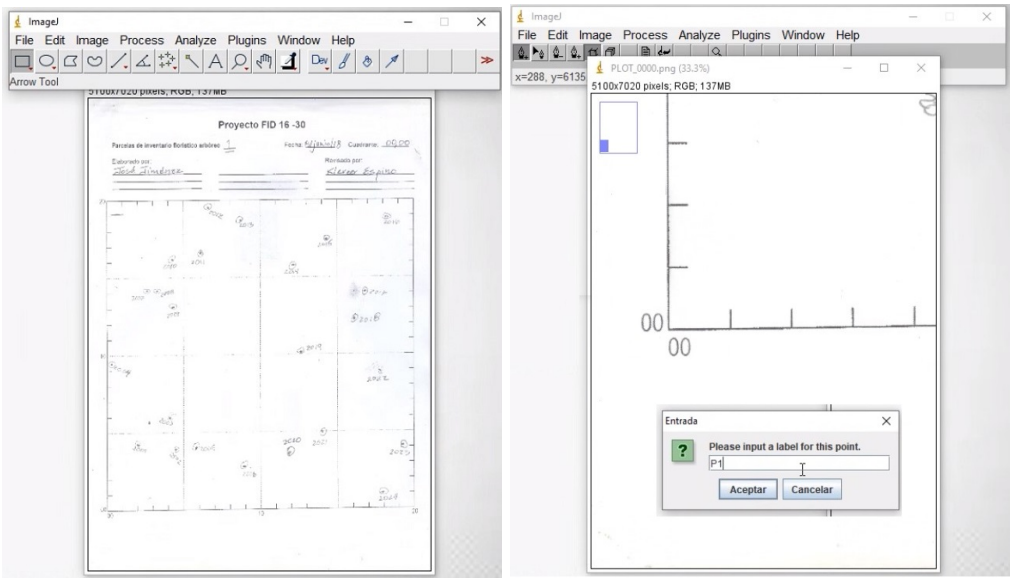

Figura 5. Abrir la aplicación y escoger imagen.

Al añadir cruces, las cuatro esquinas del croquis del cuadrante deben hacerse primero en el siguiente orden y etiquetarse como sigue:

1. *p1* = inferior izquierda
2. *p2* = superior izquierda
3. *p3* = superior derecha
4. *p4* = inferior derecha

Se sugiere que se añadan cruces utilizando el número de etiqueta o placa de cuatro o seis dígitos (por ejemplo, 1001) a todos los tallos etiquetados en orden secuencial y que las etiquetas fuera de secuencia se hagan al final. Otro enfoque a considerar sería terminar un subcuadrante a la vez, asegurándose de digitalizar todos los tallos de ese subcuadrante. Compruebe la lista de resultados una vez que se hayan añadido todas las cruces a la imagen del mapa.

La función *Lupa* puede utilizarse para acercarse a las zonas del mapa en las que los puntos de los tallos o los números de etiqueta parecen pequeños en la ampliación original, o en las que muchos puntos están agrupados. Los valores cruzados añadidos pueden editarse mediante la función *Editar etiquetas* (el cuarto ícono de izquierda a derecha, con una punta de plumilla y un signo más), y las cruces pueden eliminarse con el botón *Eliminar cruces* (el tercer ícono de izquierda a derecha, con una punta de plumilla y un signo menos).

Una vez que se hayan añadido todas las cruces, compruebe la imagen del croquis para asegurarse de que cada tallo dibujado tiene la marca de cruz añadida. Si alguna de las cruces añadidas aparece descentrada con respecto al punto dibujado, las cruces pueden ajustarse utilizando el botón *Mover cruces* (el segundo ícono de izquierda a derecha, con una punta de plumilla y una flecha apuntando hacia arriba).

A continuación, se debe revisar la lista de resultados para asegurarse de que las cuatro esquinas, que todos los tallos marcados tienen cruces asociadas a cada uno y que todas las cruces añadidas están etiquetadas correctamente. Se accede a la lista de resultados haciendo clic en *Exportar/Importar lista de puntos* y luego en la opción *Mostrar*.

Cuando se haya verificado que todo este correcto para el croquis de cuadrantes ahora con los datos digitalizados, es necesario guardarlo en un directorio adecuado como un archivo con extensión txt o csv que contenga la lista de puntos y las coordenadas x/y correspondientes, por ejemplo:

```
"C:\Rwork\croquis"
```

Se sugiere usar un nombre adecuado para los archivos, los cuales deben identificarse con un prefijo preferiblemente corto, por ejemplo:

```
"quadrat_0001.txt"
```

El archivo de extensión txt o csv puede guardarse haciendo clic en el botón *Exportar/Importar lista de puntos* y luego en la opción *Guardar como*.

Una vez finalizada la digitalización del cuadrante, se debe cerrar la imagen para abrir la siguiente imagen de otro cuadrante. Puede hacerlo, dando clic en *Exit PointPicker*, luego haga clic en la opción *Done* para salir de la función *PointPicker* que eliminará todas las cruces añadidas de la imagen del croquis y volverá a la configuración original de `ImageJ`. A continuación, cierre la imagen del cuadrante y ahora, puede abrirse la siguiente imagen. Otra opción sería simplemente cerrar la aplicación `ImageJ` por completo y luego volverla abrir para restablecerla completamente para la siguiente imagen de croquis.

Nombre para los archivos y carpetas

Los archivos `ImageJ` de entrada deben nombrarse de una manera muy específica. Todos los archivos deben estar almacenados en una carpeta y a esta carpeta general se le puede dar cualquier nombre. Debe proporcionarse la ruta completa de esta carpeta como un parámetro en la función. Todos los archivos de mapas pueden guardarse en la misma carpeta, dividirse en subcarpetas de columna o incluso en subcarpetas de cuadrantes. En el sistema *Windows*, la ruta completa sería algo como, por ejemplo:

```
"C:/Rpractice/mapas"
```

En los siguientes ejemplos, se asignó esta carpeta a una variable llamada `mapfolder`. Debe asignar la ruta apropiada a la carpeta para su computadora. Le sugerimos que guarde los archivos utilizando la estructura de carpetas que se detalla a continuación, lo que facilita la búsqueda de un cuadrante en particular. No es absolutamente necesario usar esta estructura, pero sí debe usar la siguiente convención de nomenclatura de archivos:

1. Tiene que usar el mismo prefijo para todos los archivos de cuadrantes resultantes.
2. Todos deben usar el mismo delimitador (comas o delimitado por tabulaciones).
3. Los nombres de los cuadrantes deben tener 4 dígitos (es decir, rellenar con ceros para columnas o filas de un solo dígito).

Dentro de la carpeta `mapas`, le sugerimos que cree una subcarpeta por cuadrante y que a cada una se le llame `cuadrante_####`, por ejemplo `cuadrante_0001`, `cuadrante_0002`, etc.

Dentro de la subcarpeta cuadrante, debe haber un archivo `ImageJ` llamado `mapa_####.txt`, por ejemplo, `mapa_0001.txt`, que corresponde al mapa del cuadrante. El número (generalmente 4 dígitos) es el número de cuadrante, que coincide con el número de la carpeta del cuadrante en la que se encuentra.

Ejecutando R

Descargue y obtenga el paquete CTFS R. El programa de conversión se llama `fullplot.imageJ`. Como prueba, ejecútelo primero con una sola carpeta configurando la ruta a esa carpeta.

Los otros argumentos son:

1. **Colrange** especifica el rango de las columnas y el rango de filas, según los dos primeros dígitos y los dos últimos dígitos respectivamente, de nuestra convención de denominación de cuadrantes.
2. **Gridsize** es el tamaño de cada mapa individual.

3. Cuando `include.subdir = F`, solo busca archivos de mapas en la carpeta especificada en el parámetro de ruta. Si `include.subdir = T`, también buscará todas las subcarpetas en la carpeta especificada en el parámetro de ruta.
4. Cuatro argumentos nombran las esquinas del mapa; deben coincidir con las etiquetas en la salida de `ImageJ` para las cuatro esquinas de cada mapa. Deben ser idénticos para cada mapa, de lo contrario el programa no puede funcionar.
5. Las etiquetas de las esquinas son insensibles a las mayúsculas, por lo que las etiquetas `p1` y `P1` se considerarán idénticas.
6. El prefijo usado para nombrar los archivos del mapa debe ser consistente. Puede ser por ejemplo: `mapa_` o simplemente `q_`.
7. Los sufijos del subcuadro deben ser consistentes y nombrados en el sentido de las agujas del reloj desde el subcuadro inferior izquierdo (No usamos esta opción en el ejemplo).
8. La configuración de `outfile = NULL` significa que los datos `NO` se escribirán en un archivo, solo se devolverán como el objeto R llamado `coords.col##`.

En el ejemplo a continuación, se establece una carpeta de asignación de variables al nombre del directorio donde se encuentra una de las carpetas de columnas. Tendrá que establecer esto en la carpeta de su computadora.

Comentarios adicionales

```
# Instrucciones para registrar las coordenadas locales
# Después de obtener los archivos .txt de ImageJ
# Debo revisar que delimita (comas o tabs) las columnas en el .txt,
# Primero se debe cargar el paquete de la CTFS.
# Hay varias formas de cargarlo.
# Se utiliza la función attach() indicando
# la ubicación del fichero.
# attach("C:/R/MANGLAR/CTFSRPackage.rdata")
# o También puedes dirigir la ubicación
# de la carpeta de trabajo
# y luego cargas el paquete con el comando
# attach("CTFSRPackage.rdata").
```

Ejemplo de códigos

```
setwd("C:/Rpractice/mapas")
getwd()
attach("CTFSRPackage.rdata")
ls(2)
search()
fullplot.imageJ
```

Más comentarios

```
# Para convertir los archivos txt
# imageJ txt de pixeles a coordenadas x,y
# en R hay que hacer lo siguiente:
# Cargar el paquete CTFSRPackage.rdata
# Correr el comando fullplot.imageJ()
# Escoger los parámetros correctos
#  para fullplot.imageJ()
#  de acuerdo a la ruta especificada (path="C:/R... "),
#  el tamaño de los cuadrantes
```

```
#  y cómo llamaron los archivos
#  text de ImageJ.
#  Por ejemplo:
#  Primero creo un objeto que indique
#  la ruta en donde están los archivos
#  "C:/Rpractice/mapas"
#  creo un objeto para la salida de
#  resultados (es un dataframe)
#  Para obtener las coordenadas de los áboles
#  de las parcelas de 20 m por 20 m.
```

Continuamos,

```
coord_plot1=fullplot.imageJ(
  path=mapas,
  outfile = NULL,
  include.subdir = F,
  delim = "\t",
  corners=c("p1","p2","p3","p4"),
  colrange=c(00,05),
  rowrange=c(00,05),
  prefix='cuadrante_',
  suffix="",
  gridsize=c(20,20)
  )
```

luego de correr los códigos, debes obtener `Finished calculating coordinates for quadrat...` etc.

Revisar los resultados

```
coord_plot1
head(coord_plot1)
dim(coord_plot1)
range(coord_plot1$lx)
range(coord_plot1$ly)
```

Si todo está bien, generamos los resultados en un archivo con extensión `.txt`. Especificamos el nombre de este archivo en el argumento `outfile`.

```
coord_plot1=fullplot.imageJ(
  path=mapas,
  outfile='Converted.txt',
  include.subdir = F,
  delim = "\t",
  corners=c("p1","p2","p3","p4"),
  colrange=c(00,05),
  rowrange=c(00,05),
  prefix='cuadrante_',
  suffix="",
  gridsize=c(20,20)
  )
```

Esto pondrá todas las coordenadas convertidas con sus números de placa y los nombres de los cuadrantes correspondientes en un archivo llamado `Converted.txt`. El archivo será salvado en la ruta especificada con la extensión .txt, con el prefijo `cuadrante_`.

Ejemplo práctico 1

Este ejemplo es para un solo cuadrante de 20 m x 20 m. Los píxeles fueron digitalizados y están en el archivo `PLOT_0000.txt`. Debe ajustar la ruta a la carpeta de trabajo propia.

```
ruta <- "E:/Rpractice/extra1"
setwd(ruta)
getwd()
```

```
## [1] "E:/Rpractice/extra1"
```

Cargamos el paquete de la CTFS.

```
# Cargar el paquete de la CTFS
attach("CTFSRPackage.rdata")
```

Observe como ajustamos los argumentos. Algunos valores predeterminados no los ajustamos, por ejemplo: delim = ",".

```
# Utilizar la función fullplot.imageJ
fullplot.imageJ(path = ruta, outfile = NULL,
    gridsize = c(20,20), prefix = "PLOT_", include.subdir = F,
    corners = c("P1", "P2", "P3", "P4"))
```

```
## Finished calculating coordinates for quadrat  0000
```

```
##     tag         lx         ly quadrat
## 5  2001  2.0330251  4.2372156    0000
## 6  2002  4.5074795  3.9551340    0000
## 7  2003  4.0922754  5.9677490    0000
## 8  2004  0.2942232  9.4559414    0000
## 9  2005  5.8151294  4.0193895    0000
## 10 2006  8.9428388  2.7752870    0000
## 11 2007  2.6405903 14.1810578    0000
## 12 2008  3.2969640 14.1474910    0000
## 13 2009  4.3448497 13.0929131    0000
## 14 2010  4.2994124 16.1690212    0000
## 15 2011  6.1388782 16.6312861    0000
## 16 2012  6.6187073 19.4948262    0000
## 17 2013  8.7180788 18.6907673    0000
## 18 2019 12.6505536 10.1571976    0000
## 19 2014 12.1419329 15.8044366    0000
## 20 2015 14.3528335 17.4174418    0000
## 21 2016 18.2799185 18.8751165    0000
## 22 2017 16.8633363 14.0788008    0000
## 23 2018 16.1224461 12.3097149    0000
## 24 2020 11.8966334  3.5903748    0000
## 25 2021 14.0346006  4.9187991    0000
## 26 2022 17.7911669  8.8420506    0000
## 27 2023 19.2796093  4.0789654    0000
## 28 2024 18.0098646  0.9932375    0000
```

Como vemos que todo resultó bien, esta vez corremos los mismos pero ajustando el argumento `outfile` para que nos genere un archivo .txt de salida. Lo llamamos `Converted.txt`.

```
fullplot.imageJ(path = ruta, outfile = "Converted.txt",
    gridsize = c(20,20), prefix = "PLOT_", include.subdir = F,
    corners = c("P1", "P2", "P3", "P4"))
```

```
## Finished calculating coordinates for quadrat  0000
```

```
##     tag         lx         ly quadrat
## 5  2001  2.0330251  4.2372156    0000
## 6  2002  4.5074795  3.9551340    0000
## 7  2003  4.0922754  5.9677490    0000
## 8  2004  0.2942232  9.4559414    0000
## 9  2005  5.8151294  4.0193895    0000
## 10 2006  8.9428388  2.7752870    0000
## 11 2007  2.6405903 14.1810578    0000
## 12 2008  3.2969640 14.1474910    0000
## 13 2009  4.3448497 13.0929131    0000
## 14 2010  4.2994124 16.1690212    0000
## 15 2011  6.1388782 16.6312861    0000
## 16 2012  6.6187073 19.4948262    0000
## 17 2013  8.7180788 18.6907673    0000
## 18 2019 12.6505536 10.1571976    0000
## 19 2014 12.1419329 15.8044366    0000
## 20 2015 14.3528335 17.4174418    0000
## 21 2016 18.2799185 18.8751165    0000
## 22 2017 16.8633363 14.0788008    0000
## 23 2018 16.1224461 12.3097149    0000
## 24 2020 11.8966334  3.5903748    0000
## 25 2021 14.0346006  4.9187991    0000
## 26 2022 17.7911669  8.8420506    0000
## 27 2023 19.2796093  4.0789654    0000
## 28 2024 18.0098646  0.9932375    0000
```

Ahora, llamamos los datos del archivo `Converted.txt` y creamos un objeto `dataframe` que llamamos `cnvrtd`.

```
cnvrtd <- read.table(file = "Converted.txt", header = TRUE,
    sep = "\t", dec = ".")
```

Las variables de este `dataframe` son: tag, lx, ly y quadrat.

El archivo mangrove.csv tiene los otros datos del inventario para la parcela. llamamos los datos del archivo `mangrove.csv` y creamos un objeto `dataframe` que llamamos `manglar`.

```
manglar <- read.csv(file = "mangrove.csv")
```

Las variables de este `dataframe` son: plot, tag, stem, latin y dbh.

Fusionamos ambos `dataframes` usando la variable tag y creamos el objeto `dfplot`.

```
dfplot <- merge(x = manglar, y = cnvrtd,
    by.x = "tag", by.y = "tag")
```

Luego, limpiamos el `dataframe` eliminando una de las columnas repetidas (plot y quadrat, eliminamos plot que esta en la posición 2 entre las columnas). Salvamos la base de datos como un archivo `.rdata`.

```
dfplot <- dfplot[ ,-2]
save(dfplot, file = "Manglar.rdata")
```

```
list.files(pattern = "rdata")
```

```
## [1] "CTFSRPackage.rdata" "Manglar.rdata"      "Manglar2.rdata"
```

También, podemos guardarlo como un archivo `.txt`.

```
sink("manglar2.txt")
dfplot
```

```
##     tag stem  latin dbh         lx         ly quadrat
## 1  2001    1 avicbi 113  2.0330251  4.2372156       0
## 2  2002    1 avicbi 226  4.5074795  3.9551340       0
## 3  2003    1 avicbi 359  4.0922754  5.9677490       0
## 4  2004    1 avicbi 116  0.2942232  9.4559414       0
## 5  2005    1 avicge 266  5.8151294  4.0193895       0
## 6  2006    1 avicbi 105  8.9428388  2.7752870       0
## 7  2007    1 avicbi 115  2.6405903 14.1810578       0
## 8  2008    1 avicge 189  3.2969640 14.1474910       0
## 9  2009    1 avicge 125  4.3448497 13.0929131       0
## 10 2010    1 avicge 267  4.2994124 16.1690212       0
## 11 2011    1 avicbi 153  6.1388782 16.6312861       0
## 12 2012    1 avicbi 100  6.6187073 19.4948262       0
## 13 2013    1 avicbi 107  8.7180788 18.6907673       0
## 14 2014    1 avicbi 129 12.1419329 15.8044366       0
## 15 2015    1 avicge 217 14.3528335 17.4174418       0
## 16 2016    1 avicbi 116 18.2799185 18.8751165       0
## 17 2017    1 avicge 163 16.8633363 14.0788008       0
## 18 2018    1 avicbi 180 16.1224461 12.3097149       0
## 19 2019    1 avicbi 221 12.6505536 10.1571976       0
## 20 2020    1 avicbi 197 11.8966334  3.5903748       0
## 21 2021    1 avicbi 258 14.0346006  4.9187991       0
## 22 2022    1 avicbi 142 17.7911669  8.8420506       0
## 23 2023    1 avicbi 246 19.2796093  4.0789654       0
## 24 2024    1 avicbi 165 18.0098646  0.9932375       0
```

```
sink()
```

```
list.files(pattern = "txt")
```

```
## [1] "Converted.txt" "manglar2.txt"  "PLOT_0000.txt"
```

Por último, podemos iniciar una nueva sesión y llamar a nuestra base de datos. El objeto `dfplot` aparecerá en nuestro `Global Environment`.

```
# dfplot
load("Manglar.rdata")
```

Y podemos hacer los análisis que necesitemos, por ejemplo: hacer un gráfico con `ggplot2` que muestre como se distribuyen los tallos en el hábitat, por especie y dap. El paquete ggplot2 (versión 3.3.6) es un sistema organizado de visualización de datos (Wickham, 2016).

```
library(ggplot2)
```

Este es el código para obtener la gráfica. En sentido general es una gráfica de dispersión de puntos, donde los ejes de "x" y "y" son las variables lx y ly, respectivamente. Las especies se distinguen por el color, usando la variable `latin` y el tamaño del punto es proporcional al diámetro del tallo a la altura del pecho, usando la variable `dbh`.

```
p1 <- ggplot(dfplot) +
 aes(x = lx, y = ly, colour = latin, size = dbh) +
 geom_point(shape = "circle plus") +
 scale_color_manual(values = c(avicbi = "#B20D0D", avicge = "#1A44E4")) +
 theme_linedraw()
```

```
ggExtra::ggMarginal(
  p = p1,
  type = 'boxplot',
  margins = 'x',
  size = 2.5,
  groupColour = TRUE,
  fill = 'transparent'
)
```

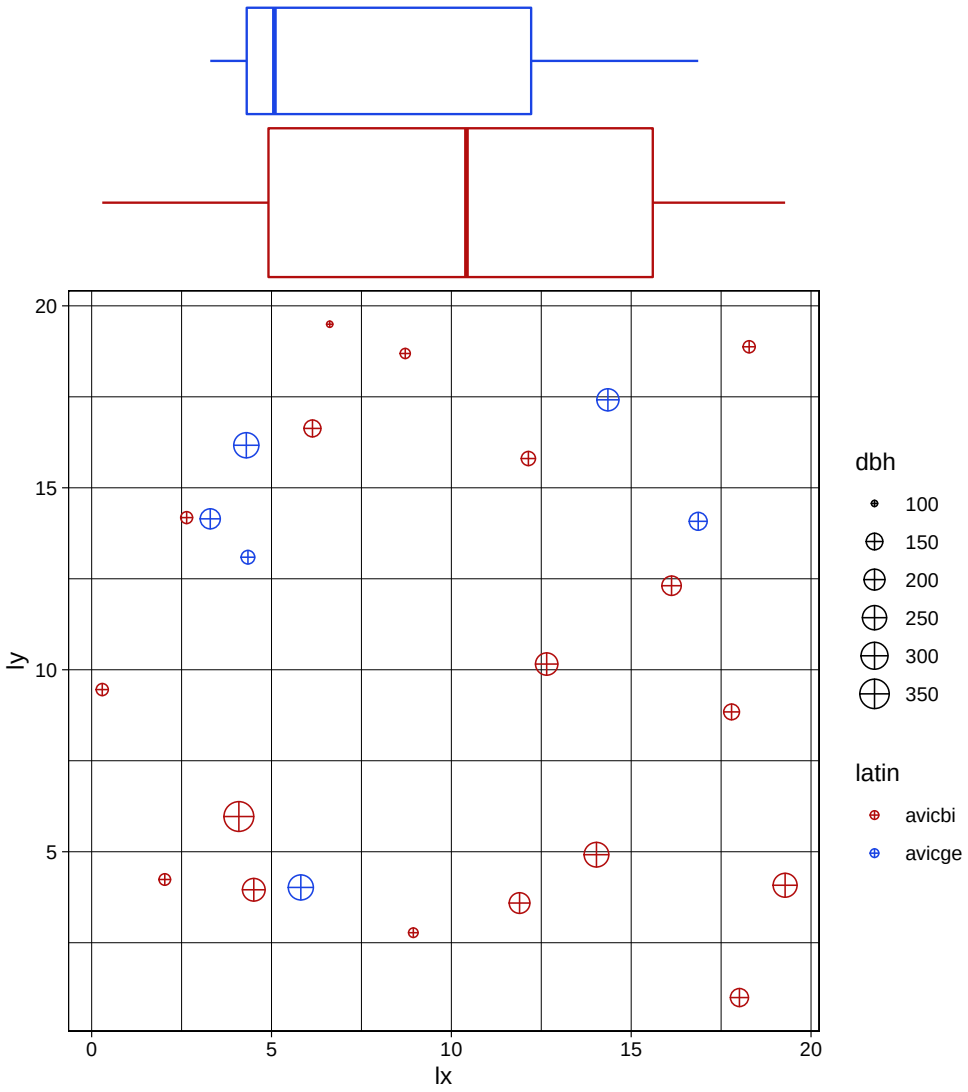

Figura 6. Ubicación de los árboles en el cuadrante.

Ejemplo práctico 2

En este ejemplo usamos cuatro cuadrantes de una parcela de 40 m x 40 m. Cada una de las salidas `txt` digitalizadas en `ImageJ` se colocó en su propia subcarpeta con el mismo nombre, en una ruta específica. Las columnas en los archivos `txt` están separadas por tabulaciones. Observe los cambios en los argumentos de la función `fullplot.imageJ`.

```
fullplot.imageJ(
 path = "E:/Rpractice/R",
 include.subdir = T,
 delim = "\t",
 gridsize = c(20,20),
 prefix = "quadrat_",
 colrange = c(0,2),
 rowrange = c(0,2),
 corners = c("p1","p2","p3","p4"),
    )
```

```
## Finished calculating coordinates for quadrat  0000
## Finished calculating coordinates for quadrat  0001
## Finished calculating coordinates for quadrat  0100
## Finished calculating coordinates for quadrat  0101
```

```
##      tag         lx         ly quadrat
## 5   2001  1.9890863  4.2445796    0000
## 6   2002  4.4680440  3.9097015    0000
## 7   2003  4.0939179  5.9358051    0000
## 8   2005  5.8068039  4.0211245    0000
## 9   2006  8.9253155  2.7632293    0000
## 10  2004  0.2981796  9.4532640    0000
## 11  2007  2.6361502 14.2061206    0000
## 12  2008  3.2662544 14.1333465    0000
## 13  2009  4.3266159 13.0905255    0000
## 14  2010  4.2575534 16.1553103    0000
## 15  2011  6.1240868 16.6161364    0000
## 16  2012  6.6330187 19.5452571    0000
## 17  2013  8.7186889 18.6480220    0000
## 18  2014 12.1538430 15.7496836    0000
## 19  2015 14.3318755 17.4312444    0000
## 20  2016 18.2434229 18.8510343    0000
## 21  2017 16.8048469 14.0565925    0000
## 22  2018 16.1322633 12.2828649    0000
## 23  2019 12.6753758 10.1666395    0000
## 24  2022 17.7985821  8.8801150    0000
## 25  2020 11.8805619  3.6041574    0000
## 26  2021 14.0202939  4.9671202    0000
## 27  2023 19.2830734  4.0912429    0000
## 28  2024 17.9578091  0.9895237    0000
## 51  2025  0.9533438  1.0463022    0001
## 61  2032  9.4165889  4.6285765    0001
## 71  2031  8.2924238  7.1432709    0001
## 81  2026  1.5358735  8.6877315    0001
## 91  2030  5.4155282  9.5291625    0001
## 101 2027  0.8377662 10.3219738    0001
## 111 2028  3.2990767 11.1410877    0001
## 121 2029  4.1662740 10.4610787    0001
```

```
## 131 2038  8.4252429 12.2049865    0001
## 141 2037  8.1550482 14.4228149    0001
## 151 2033  1.5899736 16.0487548    0001
## 161 2034  4.0003326 19.8127674    0001
## 171 2035  5.3498530 19.8719328    0001
## 181 2036  8.6795695 18.5837145    0001
## 191 2039 12.3924735 19.2427733    0001
## 201 2040 13.4180048 18.2676946    0001
## 211 2041 10.9081185 17.1569559    0001
## 221 2042 15.4574984 16.4613211    0001
## 231 2043 15.3082556 14.0208865    0001
## 241 2044 15.2487765 12.4410185    0001
## 251 2045 12.6204311 10.2225746    0001
## 261 2046 18.9692959  8.6516304    0001
## 271 2047 13.4038975  7.7888635    0001
## 281 2048 16.9645184  4.8780717    0001
## 29  2090 11.5537378  1.6087305    0001
## 30  2049 15.9818668  0.9133563    0001
## 52  2050  1.6972053  1.6652438    0100
## 62  2051  4.8633207  3.1403327    0100
## 72  2052  5.8175768  3.9692980    0100
## 82  2053  2.2280371  8.4516467    0100
## 92  2054  1.7952025  9.7647907    0100
## 102 2055  8.8934936 11.2967844    0100
## 112 2056  8.6422032 12.3200189    0100
## 122 2057  6.7248380 18.3875912    0100
## 132 2058 12.8820427 19.7746887    0100
## 142 2059 14.8684638 19.8162629    0100
## 152 2060 17.4291208 17.3381047    0100
## 162 2061 15.0074099 12.7325507    0100
## 172 2062 19.8611770 12.0375320    0100
## 182 2063 10.4379438  9.6214068    0100
## 192 2064 13.2855459  8.2331402    0100
## 202 2065 13.4068938  5.1479913    0100
## 212 2066 14.7146862  4.7036147    0100
## 222 2067 13.4698317  1.5675839    0100
## 232 2068 11.3984285  1.7798363    0100
## 242 2069 18.8260089  1.6406401    0100
## 252 2070 17.6219280  1.9267082    0100
## 53  2071  0.9643473  4.6610452    0101
## 63  2072  1.0329735  6.1865944    0101
## 73  2073  9.0856152  3.5585103    0101
## 83  2074  7.9867432  7.6946132    0101
## 93  2075  2.4052798 10.4667976    0101
## 103 2076  5.3806261 12.2242198    0101
## 113 2077  7.6311138 11.3061476    0101
## 123 2078  9.0700845 16.8298514    0101
## 133 2079  4.5074471 15.3873139    0101
## 143 2080  2.3874884 17.4894270    0101
## 153 2081 13.3355059 18.7149226    0101
## 163 2082 17.3243334 19.3081218    0101
## 173 2083 11.9387672 15.9445064    0101
## 183 2084 12.3705848  9.6774419    0101
## 193 2085 14.7104914  8.5471816    0101
```

```
## 203 2086 10.5688650  5.8715599    0101
## 213 2087 19.2688800  8.4318897    0101
## 223 2088 15.5065614  3.1985878    0101
## 233 2089 18.1310077  2.8973211    0101
```

Como no se ajustó el argumento `outfile` se generó un archivo `.txt` de salida con el nombre `plotLxLy`, que es el valor predeterminado para este argumento dentro de la función. Lo mostramos a continuación:

```
list.files(path = "E:/Rpractice/R", pattern = "txt")
```

```
## [1] "plotLxLy.txt"
```

Cargamos los datos con las coordenadas locales. Creamos un objeto llamado `cnvrtd2`.

```
cnvrtd2 <- read.table(file = "E:/Rpractice/R/plotLxLy.txt",
    header = TRUE, sep = "", dec = ".")
```

También, cargamos los datos con la información de las especies y los dap. Creamos un objeto llamado `datos2`.

```
datos2 <- read.csv(file = "E:/Rpractice/R/data_manglar1.csv",
    header = TRUE, sep = ";")
```

Como la parcela es de 40 m x 40 m se deben convertir las coordenadas locales a coordenadas generales. Creamos una función para esto y la llamamos `convertir`. Se puede editar para otros tamaños de parcelas.

```
convertir <- function(object = NULL, lx = NULL, ly = NULL, quadrat=NULL)
{
  if (!is.null(object)) {
    lx = object$lx
    ly = object$ly
    quadrat=object$quadrat
    }
  if (quadrat==0000) {gx=lx}
  if (quadrat==0000) {gy=ly}
  if (quadrat==0001) {gx=lx}
  if (quadrat==0001) {gy=ly+20}
  if (quadrat==0100) {gx=lx+20}
  if (quadrat==0100) {gy=ly}
  if (quadrat==0101) {gx=lx+20}
  if (quadrat==0101) {gy=ly+20}
  return(data.frame(gx,gy))
}
```

Con la siguiente línea de código aplicamos la función `convertir` para cada fila en el objeto `cnvrtd2`.

```
library (plyr)
df1 <- adply(split(cnvrtd2,cnvrtd2$tag),.margins = 1,.fun = convertir)
```

Revisamos.

```
head(df1)
```

```
##     X1        gx       gy
## 1 2001 1.9890863 4.244580
## 2 2002 4.4680440 3.909701
## 3 2003 4.0939179 5.935805
## 4 2004 0.2981796 9.453264
## 5 2005 5.8068039 4.021124
## 6 2006 8.9253155 2.763229
```

Revisamos, más.

```
tail(df1)
```

```
##      X1       gx       gy
## 85 2085 34.71049 28.54718
## 86 2086 30.56886 25.87156
## 87 2087 39.26888 28.43189
## 88 2088 35.50656 23.19859
## 89 2089 38.13101 22.89732
## 90 2090 11.55374 21.60873
```

Fusionamos las bases de datos.

```
dfplot2 <- merge(x = datos2, y = df1, by.x ="tag" , by.y ="X1")
```

Revisamos.

```
head(dfplot2)
```

```
##    tag cuadrat stem  latin dbh        gx       gy
## 1 2001       0    1 avicbi 113 1.9890863 4.244580
## 2 2002       0    1 avicbi 226 4.4680440 3.909701
## 3 2003       0    1 avicbi 359 4.0939179 5.935805
## 4 2004       0    1 avicbi 116 0.2981796 9.453264
## 5 2005       0    1 avicge 266 5.8068039 4.021124
## 6 2006       0    1 avicbi 105 8.9253155 2.763229
```

Revisamos, más.

```
tail(dfplot2)
```

```
##      tag cuadrat stem  latin dbh       gx       gy
## 100 2087     101    2 avicbi 112 39.26888 28.43189
## 101 2087     101    3 avicbi  64 39.26888 28.43189
## 102 2087     101    4 avicbi  41 39.26888 28.43189
## 103 2088     101    1 avicbi 131 35.50656 23.19859
## 104 2089     101    1 avicbi 105 38.13101 22.89732
## 105 2090       1    1 avicbi 171 11.55374 21.60873
```

Salvamos nuestra base de datos para analizarlos cuando queramos.

```
save(dfplot2, file = "Manglar2.rdata")
```

```
list.files(pattern = "rdata")
```

```
## [1] "CTFSRPackage.rdata" "Manglar.rdata"      "Manglar2.rdata"
```

Por último, podemos iniciar una nueva sesión y llamar a nuestra base de datos. El objeto `dfplot2` aparecerá en nuestro `Global Environment`.

```
load("Manglar2.rdata")
```

Hacemos un gráfico.

```
title <-  "Parcela de muestreo"
lbls <-  dfplot2$tag
```

```
p  <- qplot(gx, gy,
data = dfplot2, colour = latin, size = dbh, xlim = c(0, 40),
ylim = c(0, 40)) +
```

```
geom_text(aes(label = lbls),
size = 2, hjust = -0.50, vjust = -0.50) +
geom_point(alpha = 0.05) +
scale_color_brewer(palette = "Set1")
```

```
bp <- p + theme_bw () + theme(legend.position="bottom")+ ggtitle(title)
```

```
bp + scale_y_continuous(breaks=seq(0, 40, 10)) + scale_x_continuous(breaks=seq(0, 40, 10))
```

```
## Scale for 'y' is already present. Adding another scale for 'y', which will
## replace the existing scale.
```

```
## Scale for 'x' is already present. Adding another scale for 'x', which will
## replace the existing scale.
```

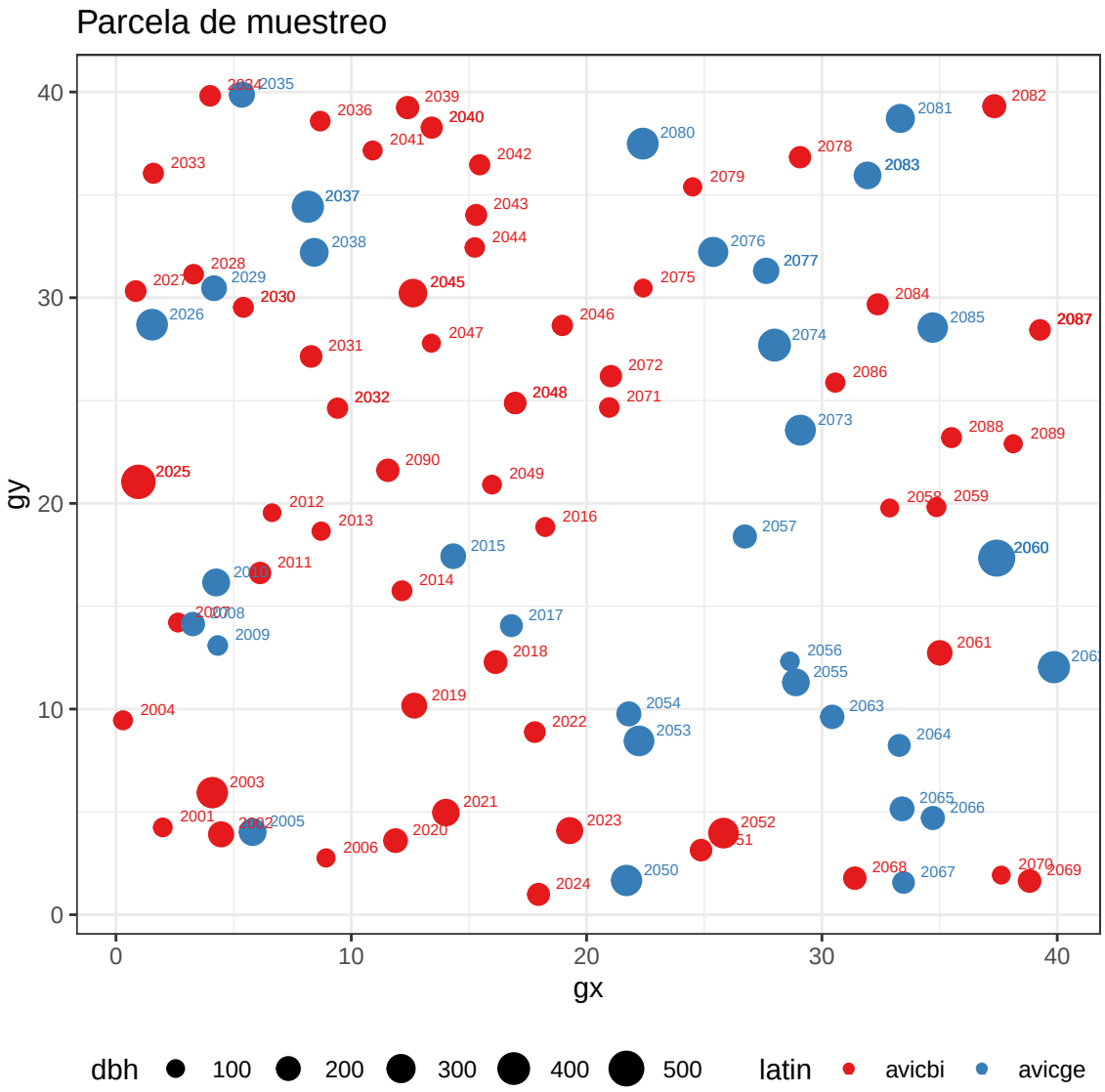

Figura 7. Ubicación de los árboles en la parcela.

Información sobre la sesión

Este documento y los códigos fueron escritos usando *R Markdown*[3] (Allaire et al., 2021; Xie et al., 2018, 2020), LaTeX[4] (LaTeX Project Team, 2006), *R*[5] (R Core Team, 2021) y *RStudio*[6] (RStudio Team, 2021).

```
sessioninfo::session_info(info = "platform")
```

```
## - Session info -----------------------------------------------------------------
##  setting  value
##  version  R version 4.2.0 (2022-04-22 ucrt)
##  os       Windows 10 x64 (build 19042)
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United States.utf8
##  ctype    English_United States.utf8
##  tz       America/Bogota
##  date     2022-05-20
##  pandoc   2.17.1.1 @ C:/Program Files/RStudio/bin/quarto/bin/ (via rmarkdown)
##
## --------------------------------------------------------------------------------
```

[3] R Markdown es un formato de sintaxis simple que permite una fácil creación de documentos, presentaciones dinámicas y informes de R.

[4] LaTeX es un sistema de preparación de documentos utilizado para la comunicación y publicación de documentos científicos de alta calidad.

[5] R es un entorno y lenguaje de programación con un enfoque al análisis estadístico.

[6] RStudio es un entorno de desarrollo integrado para el lenguaje de programación R.

Referencias

Alder, D. y Synnott, T. J. (1992). *Permanent sample plot techniques for mixed tropical forest* (p. 124). Oxford Forestry Institute, University of Oxford. Oxford, Inglaterra.

Allaire, J., Xie, Y., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W. y Iannone, R. (2021). *Rmarkdown: Dynamic documents for r*. https://github.com/rstudio/rmarkdown

Camacho Calvo, M. (2000). *Parcelas permanentes de muestreo en bosque natural tropical: Guía para el establecimiento y medición* (p. 52). Manual Técnico No. 52. Unidad de Manejo de Bosques Naturales, CATIE. Turrialba, Costa Rica.

Condit, R. (2008). *Methods for estimating aboveground biomass of forest and replacement vegetation in the tropics* (pp. 1–73). Center for Tropical Forest Science. Research Manual.

Condit, R. (2011). *Converting ImageJ output to local plot coordinates using R*. ForestGEO. http://ctfs.si.edu/ctfsdev/CTFSRPackageNew/index.php/web/tutorials/imageJTutorial/index.html

Dallmeier, F. (Ed.). (1992). *Long-term monitoring of biological diversity in tropical forest areas* (p. 72). Educational, Scientific and Cultural Organization. Paris, Francia.

Klein, C. y Morales, D. (2002). Consideraciones metodológicas al establecer parcelas permanentes de observación en bosque natural o plantaciones forestales. *Revista Forestal Centroamericana*, *39-40*.

LaTeX Project Team. (2006). *The LaTeX Project*. http://www.latex-project.org

R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

RStudio Team. (2021). *RStudio: Integrated development environment for R*. RStudio, PBC. http://www.rstudio.com/

Schneider, C. A., Rasband, W. S. y Eliceiri, K. W. (2012). NIH image to ImageJ: 25 years of image analysis. *Nature Methods*, *9*(7), 671–675. https://doi.org/doi:10.1038/nmeth.2089

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org

Xie, Y., Allaire, J. J. y Grolemund, G. (2018). *R markdown: The definitive guide*. Chapman; Hall/CRC. https://bookdown.org/yihui/rmarkdown

Xie, Y., Dervieux, C. y Riederer, E. (2020). *R Markdown cookbook*. Chapman; Hall/CRC. https://bookdown.org/yihui/rmarkdown-cookbook