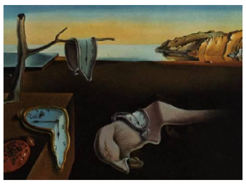

Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. doi: 10.18637/jss.v000.i00

Reproducible, flexible and high throughput data extraction from primary literature: The metaDigitise R package

Joel L. Pick
University of New South Wales

Shinichi Nakagawa
University of New South Wales

Daniel W.A. Noble
University of New South Wales

Abstract

Keywords: meta-analysis, comparative analysis, data extraction, R, reproducibility, figures, images, summary statistics.

1. Introduction

In many different contexts, researchers need to make use of data presented in primary literature. Most notably, this includes meta-analysis, which is becoming increasingly common in many research fields. Meta-analysis uses effect size estimates and their sampling variance, taken from many studies, to understand whether particular effects are common across studies and to explain variation among these effects (Nakagawa et al. 2017). Meta-analysis therefore relies foremost on data extracted from primary literature and more specifically statistics that have been reported in the text or tables of research papers. These summary statistics are, however, frequently presented in figures and so need to be manually extracted using digitising programs. Although there are several existing tools to perform tasks like this (EXAMPLES), these tools are not designed this specific purpose (i.e. a meta-analysis). Specifically, they do not differentiate between common plot types that are use to present data, meaning that they require a large amount of downstream data manipulation. For example, many stand alone programs only provide the raw data or values from plots. This means that these data need to be imported into other programs to calculate correlations (if scatter plots) or errors on means (from mean error plots) need to be back calculated by hand to convert them to a common type (e.g., standard errors to standard deviations). Digitising programs also do not easily allow for different grouping of points to be made, and frequently do not integrate metadata (such as variable names, group names and sample sizes) at the time of data extraction. Additionally, existing stand-alone software does not allow easy import into commonly used statistical software (such as R), or try to optimize the research pipelines to facilitate editing and reproducibility. These are major issues because extracting from figures can be an incredibly time-consuming process. Furthermore, although meta-analysis is an important tool in consolidating the data from many studies, relying on transparency of those studies, many of the processes involved in data extraction are opaque and difficult to reproduce, making extending studies problematic. Having a tool that facilitates reproducibility in meta-analyses will increase transparency and go a long way to resolving the reproducibility crises we are seeing in many fields (REF).

Here, we present an interactive R package, **metaDigitise**, which is designed for large scale data extraction from figures, specifically catering to the the needs of meta-analysts. To this end, we provide tools specific to data extraction from common plot types (mean and error plots, box plots, scatter plots and histograms, see Figure 1). **metaDigitise** operates within the R environment making data extraction, analysis and export more streamlined. It also provides users with options to conduct the necessary calculations on raw data immediately after extraction so that comparable summary statistics can be obtained quickly. Summaries will condense multiple figures into data frames or lists (depending on the type of figures) and these objects can easily be exported from R, or if using the raw data, analysed in any way the user desires. Conveniently, when needing to process many figures at different times **metaDigitise** will only import figures not already completed within a directory. This makes it easy to add new figures at any time. **metaDigitise** has also been built for reproducibility in mind. It has functions that allow users to redraw their digitisations on figures, make corrections and access the raw calibration data which is written automatically for each figure that is digitised into a special folder within the directory. This makes sharing figure digitisation and reproducing the work of others simple and easy, and allows meta-analysts to update meta-analyses more easily.

2. Directory Structure, Image Processing and Reproducibility

The **metaDigitise** package is designed to be flexible, yet simple to use. There is one main function in the package, `metaDigitise()`, which interactively takes the user through the process of extracting data from figures. `metaDigitise()` was created with the idea that the user would likely have multiple images to extract from. It therefore operates in the same way whether the user has one or multiple images. `metaDigitise()` is designed to work on a directory containing images of figures copied from primary literature, in .png, .jpg, .tiff, .pdf format. This directory is sepcified to `metaDigitise()` through the `dir` argument. The user is free to set their own broad directory structure (e.g. one directory for all images or one directory for each paper extracted from). We would recommend having all files for one project in a single directory with an informative and unambiguous naming scheme for images to make it easy to identify the paper and figure the data come from. This cuts out the need to change directories constantly. For example the directory structure could look like:

```
* Main project directory
    + FiguresToExtract
        + Paper1_Figure1_trait1.png
        + Paper1_Figure2_trait2.png
        + Paper1_Figure3_trait3.png
        + Paper2_Figure1_trait1.png
        + Paper2_Figure2_trait2.png
        + Paper2_Figure3_trait3.png
```

It is important to for the user to think about their directory structure early on in this process (also more generally in the context of their entire project), especially if they plan to share the

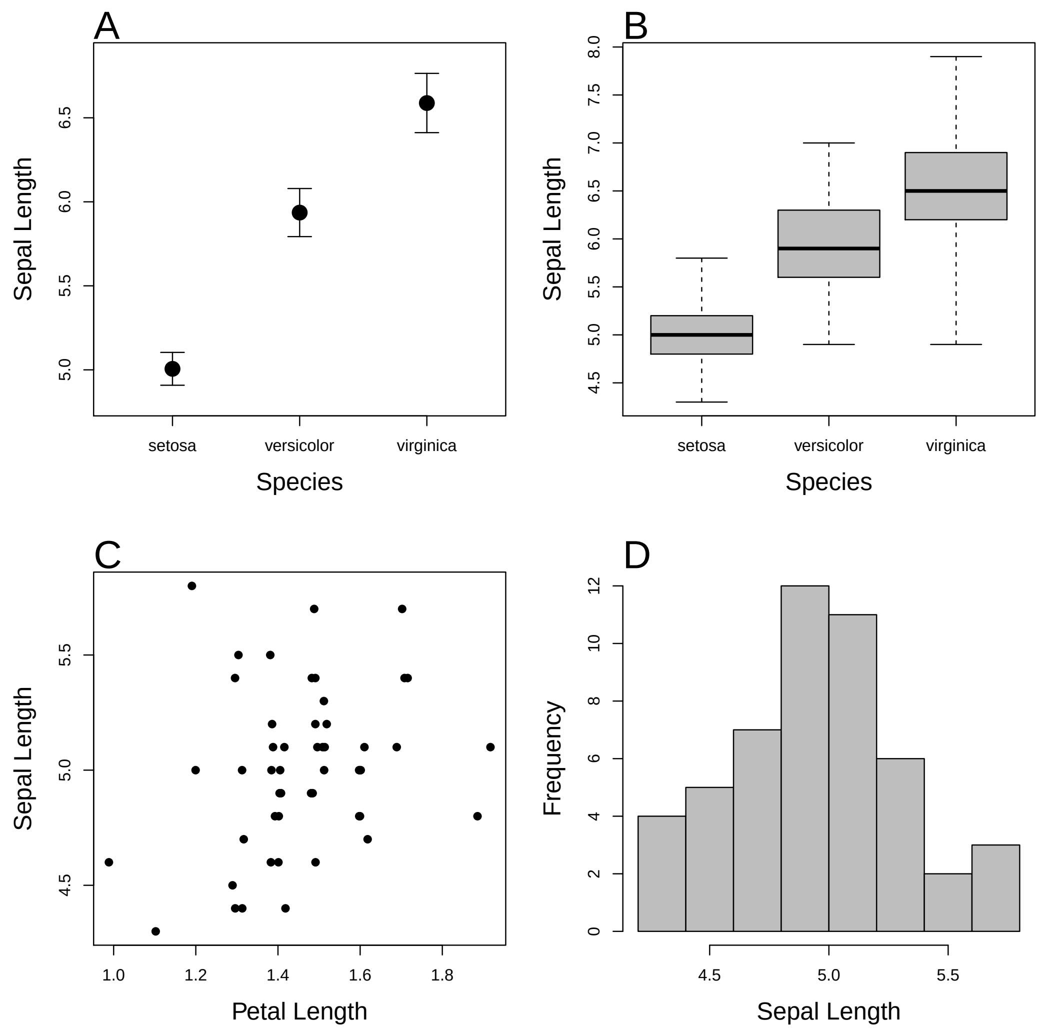

Figure 1: Four plot types that **metaDigitise** is designed to extract data from: A) mean and error plot, B) box plot, C) scatter plot and D) histogram. Data is taken from the iris dataset in R. A and B are plotted with the whole dataset, C and D are just the data for the species setosa.

extractions with collaborators or when publishing the project.

When `metaDigitise()` is run, it recognizes all the images in a directory and automatically imports them one by one, allowing the user to click and enter relevant information about a figure as they go. This expedites digitising figures by preventing users from having to constantly change directories and / or open new images. The data from a completed image is automatically saved in an .RDS file to a `caldat` directory, that is created within the parent directory when first executing the `metaDigitise()` function. These files enable re-plotting and editing of images at a later point (see below), but do not need to be directly accessed by the user.

A particularly powerful and flexible aspect of `metaDigitise()` is its ability to identify images that have been previously digitised and only import images that have not been digitised in subsequent calls of the function. This means both that all figures do not need to be extracted at one time and that new figures can be added as the project develops. After each image is extracted, the user is asked whether they wish to continue or quit the extraction process. Upon rerunning `metaDigitise()`, previously digitised figures are simply ignored during processing, but their data re-integrated within the final output after new files are completed.

After completing all images, or upon quitting, the processed data (in a form specified by the user) is then returned. From all plot types, `metaDigitise()` summarises the data from a figure as a mean, standard deviation and sample size, for each identified group within the plot (should multiple groups exist). These are the summary statistics needed to create many of the relevant effect sizes and sampling error for a meta-analysis. In the case of scatter plots, `metaDigitise()` also returns the correlation coefficient between the points within each identified group.

3. Diverse Plot Types

metaDigitise recognises four main types of plot; Mean and error plots, box plots, scatter plots and histograms, shown in Figure 1. Each of these can be processed together and integrated into a single output. Alternatively, users can keep like figures together and process them separately.

In order to correctly extract data from figures `metaDigitise()` always requires the user to calibrate the axes in the figure. To do this, the user is required to click on two known points on the axis in question, and then enter the value of those points in the figure . Using this information, `metaDigitise()` then calculates the value of any clicked points in terms of the figure axes. In the case of mean and error plots and box plots, it calibrates only the y axis (assuming the x axis is redundant). For scatter plots and histograms both axes are calibrated.

3.1. Mean and error plots

`metaDigitise()` prompts the user to enter groups names and allows the user to enter sample sizes (n), which are used in downstream processing. The user is then prompted to click on an error bar followed by the mean. Error bars above or below the mean can be clicked - sometimes one is clearer than the other. `metaDigitise()` assumes that the error bars are symmetrical. Where the user has clicked the error is displayed in a different colour to the mean (Figure 2A). The user can subsequently add more groups, edit groups or remove groups. Finally the user is asked what type of error was used in the figure: standard deviation (SD,

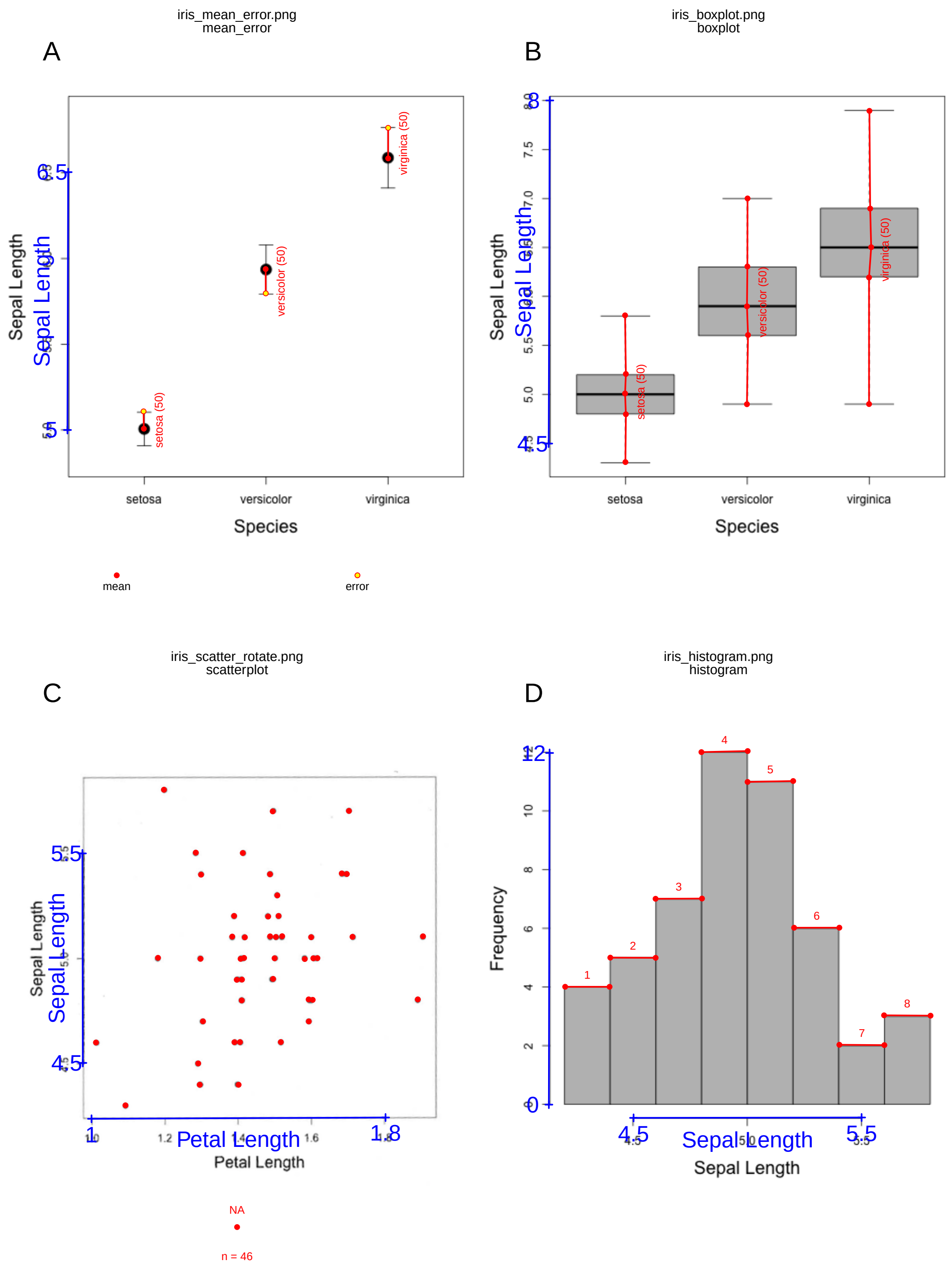

Figure 2: Demonstration of data extraction from different plot types

σ), standard error (SE) or 95% confidence intervals (CI95). Standard deviation is calculated from standard error as

$$\sigma = SE * \sqrt{n} \tag{1}$$

and from 95% confidence intervals as

$$\sigma = \frac{CI}{1.96} * \sqrt{n} \tag{2}$$

If the user does not enter a sample size at the time of data extraction (if for example the information is not readily available) the SD is not calculated. This can be entered at a later time, however (see below).

3.2. Box plots

As with mean and error plots, `metaDigitise()` prompts the user to enter groups names and allows the user to enter sample sizes (n), which are used in downstream processing. The user is then prompted to click on the maximum (b), upper quartile (q_3), median (m), lower quartile (q_1) and minimum (a). `metaDigitise()` will check that the maximum is greater than the minimum, and return an warning if that is not the case. The user can subsequently add, edit or remove groups. From the extracted data, the mean (μ) and SD are calculated as

$$\mu = \frac{a + 2q_1 + 2m + 2q_3 + b}{8} \tag{3}$$

$$\sigma = \frac{b-a}{4\Phi^{-1}(\frac{n-0.375}{n+0.25})} + \frac{q_3 - q_1}{4\Phi^{-1}(\frac{0.75n-0.125}{n+0.25})} \tag{4}$$

, where $\Phi^{-1}(z)$ is the upper zth percentile of the standard normal distribution, following Wan, Wang, Liu, and Tong (2014). As with mean and error plots, if the user does not enter a sample size at the time of data extraction the SD is not calculated.

3.3. Scatter plots

`metaDigitise()` prompts the user to enter groups names and then to click on points. Points added by mistake can be deleted. The user can subsequently add groups, edit groups (add or remove points) or delete groups. Different groups are plotted in different colours and shapes, with a legend at the bottom of the figure (Figure 2C). Mean, SD and n are calculated from the clicked points, for each group. In the case that the sample size from the clicked points does not match a known sample size (e.g. if there are overlaid points), the user can enter a alternate sample size.

Histograms

metaDigitise prompts the user to enter groups names and then to click on the top corners of each bar. bars can subsequently be deleted. **MetaDigitise** then calculates the midpoint and the frequency for each bar.

4. Extracting Data From Plots

We'll now demonstrate how `metaDigitise()` works using figures generated from the well known iris data set. Users can install the **metaDigitise** package from GitHub as follows:

```
R> install.packages("devtools")
R> devtools::install_github("daniel1noble/metaDigitise")
R> library(metaDigitise)
```

Assume that we would like to extract summary statistics from studies measuring sepal length or width in iris species for a fictitious project. There are a few studies that only present these data in figures. As we read papers found from a systematic search we can add in figures with relevant data to a "FiguresToExtract" folder as follows

```
*FiguresToExtract
    + 001_Anderson_1935_Fig1.png
```

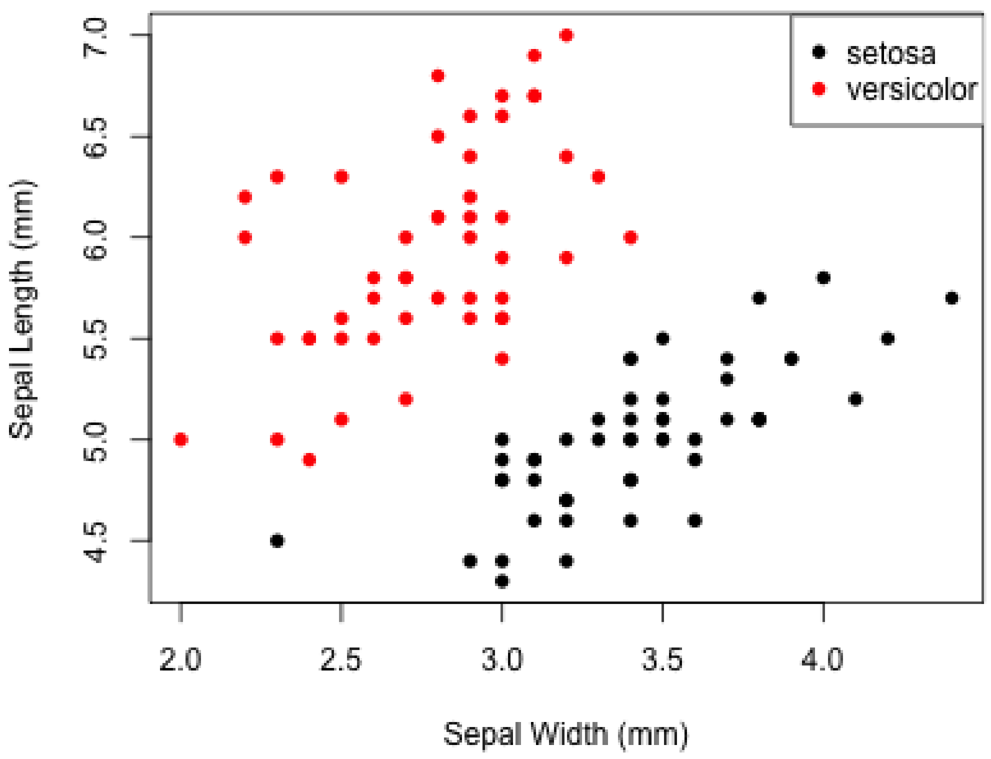

Figure 3: Example scatterplot (001_Anderson_1935_Fig1.png) of sepal length and width for two species of iris (setosa and versicolor)

Here, the naming of the files placed in the folder will contain the paper number, first author and the figure number to keep data uniquely associated with figures. We can use `metaDigitise()` to begin processing the image from the first paper. Running `metaDigitise()` brings up a series of prompts for the user using a main menu that provides access to a number of its features:

```
R> digitised_data <- metaDigitise("~/FiguresToExtract", summary = TRUE)
```

```
        Do you want to...
1: Process new images
2: Import existing data
3: Edit existing data
Selection:
```

The user simply enters in the numeric value that corresponds to what they would like to do. In this case we want to "Process new images".

As soon as `metaDigitise()` is executed, it will bring the user to the main menu (see "Re-importing, Editing and Plotting Previously Digitised data" above). When the user selects option one ("Process new images"), we are asked whether we have different types of plot(s) in the folder. This question is most relevant when there are lots of different figures in the folder because it will then ask us for the type of figure as they are cycled through.

```
Are all plot types the same? (diff/same)
```

`metaDigitise()` then asks the user whether the figure needs to be rotated or flipped. This can be needed when box plots and mean and error plots are not orientated correctly. In some cases, older papers can give slightly off angled images which can be corrected by rotating. So, in this prompt the user has three options: `f` for "Flip", `r` for "rotate" or `c` for "continue.

```
mean_error and boxplots should be vertically orientated

       _
       |
  I.E. o    NOT  |-o-|
       |
       -

If they are not then chose flip to correct this.

If figures are wonky, chose rotate.

Otherwise chose continue

Flip, rotate or continue f/r/c

c
```

After this, `metaDigitise()` will ask the user to specify the plot type. Depending on the figure, the user can specify that it is a figure containing the mean and error (`m`), a box plot (`b`), a scatter plot (`s`) or a histogram (`h`). If the user has specified `diff` instead of `same` in response to the question about whether the plot types are the same or different, this question will pop up for each plot, but will only be asked once if plots are all the same.

```
Please specify the plot_type as either: mean and error, box plot,
 scatter plot or histogram m/b/s/h:
```

After selecting the figure type a new set of prompts will come up that will ask the user first what the y and x-axis variables are. This is useful as users can keep track of the different variables across figures and papers. Here, the user can just add this information in to the R console. Once complete, details on how to calibrate the x and y-axis appear, so that the relevant statistics / data can be correctly calculated. When working with a plot of mean and standard errors, the x-axis is rather useless in terms of calibration so **metaDigitise** just asks the user to calibrate the y-axis.

```
What is the y variable?

R> Sepal Length (mm)

What is the x variable?

R> Sepal Width (mm)

On the Figure, click IN ORDER:
      y1, y2 , x1, x2


    Step 1 ----> Click on y1
  |
  |
  |
  |
  y1
  |_________________________
  ....

    Step 3 ----> Click on x1
  |
  |
  |
  |
  |
  |_____x1__________________

  ....
```

The user can just follow the instructions on screen step-by-step (instructions above have been truncated by '...' to simplify), and in the order specified. Before moving on, the user is forced to check whether or not the calibration has been set up correctly. If 'n' is chosen because something needs to be fixed then the user can re-calibrate.

```
What is the value of y1 ?

R> 4.5

What is the value of y2 ?

R> 7

What is the value of x1 ?

R> 2

What is the value of x2 ?

R> 4

Re-calibrate? (y/n)
```

Often, plots might contain multiple groups that the meta-analyst wants to extract from. `metaDigitise()` handles this nicely by prompting the user to enter the group first, followed by the digitisation of this groups data. After digitising the first group, and having exited (i.e., hit 'esc' from plot window), `metaDigitise()` will ask the user whether they would like to add another group. Users can continually add groups (`a`), or simply continue to the next plot (`c` - if another plot exists). The number of groups are not really limited and users can just keep adding in groups to accommodate the different numbers that may be presented across figures (although it can get complicated with too many).

```
Follow instructions below, to exit point adding or removing:

 - Windows: right click on the plot area and choose 'Stop'!

 - X11: hit any mouse button other than the left one.

 - quartz/OS X: hit ESC

Group identifier:

R> setosa

Click on points you want to add.
If you want to remove a point, or are finished with a
group, exit (see above), then follow prompts

Add points, delete points or continue? a/d/c

c
```

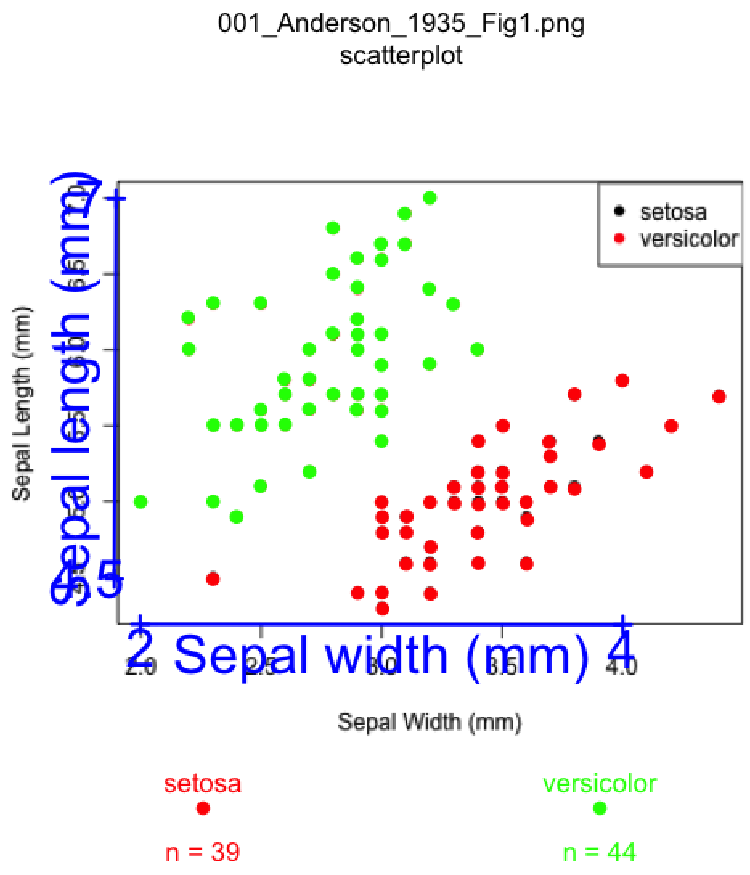

Figure 4: Digitisation of sepal length and width for two species of iris (setosa and versicolor). Names of the variables and calibration (in blue) are plotted alongside the digitised points (green = versicolor; red = setosa). The sample sizes for each group are provided on the lower part of the plot. All figures are clearly labeled at the top to remind users of the filename and plot type. This reduces errors throughout the digitisation process.

Once we are done digitising all the groups our plot will look something like Figure 4

When completed **metaDigitise** will write the digitised data to the caldat directory, such that our new directory structure is as follows

```
*figs_to_extract
    + caldat
        + 001_Anderson_1935_Fig1
    + 001_Anderson_1935_Fig1.png
```

Users can access the **metaDigitise** object created (001_Anderson_1935_Fig1) at any time using the metaDigitse function. In the R console, the summarised data for the digitised figure can be printed on screen or even written to a .csv file:

```
R> digitised_data

 filename    group_id         variable       mean       sd  n          r  plot_type
1 001_Anderson_1935_Fig1.png     setosa  Sepal Width (mm) 3.420332 0.4034259 39 0.7524909
```

```
2 001_Anderson_1935_Fig1.png     setosa Sepal Length (mm) 4.998267 0.3831294 39 0.7524909
3 001_Anderson_1935_Fig1.png versicolor  Sepal Width (mm) 2.766857 0.3234723 44 0.5157938
4 001_Anderson_1935_Fig1.png versicolor Sepal Length (mm) 5.936769 0.5292535 44 0.5157938
```

The mean for each of the two variables, along with the two species, are provided. Since this is a scatterplot, the user also gets the Person's correlation coefficient between sepal length and width for each species. These match reasonably well with the actual means of sepal length and width for each of the species in the full 'iris' dataset:

```
     Species meanSL meanSW
1     setosa  5.006  3.428
2 versicolor  5.936  2.770
```

One thing anyone with a familiarity with the iris dataset will notice is that the sample sizes for each of these species (which are n = 50 each) are quite a bit lower. This is an example of some of the challenges when extracting data from scatter plots, often data points will overlap with each other making it impossible (without having the real data) to know whether this is a problem. However, a meta-analyst will probably realise that the sample sizes here conflict with what is reported in the paper. Hence, **metaDigitise** also provides the user with options to input the sample sizes directly (see Editing section above), even for scatter plots and histograms where, strictly speaking, this should not be necessary. Nonetheless, it is important to recognise the impact that overlapping points can have on summary statistics, particularly its effects on standard deviation (SD) and standard error (SE). Here, the mean point estimates are nearly bang on, but the SD's are slightly over-estimated:

```
     Species    meanSL    meanSW
1     setosa 0.3524897 0.3790644
2 versicolor 0.5161711 0.3137983
```

4.1. Adding new figures

Users can add additional figures as new papers with relevant information are found. Each figure should be in its own file with unique naming, even if a single paper has multiple figures for extraction. For example, another paper on different populations (and one new species) of iris contained two additional figures where important data could be extracted. These figures can simply be named accordingly and added directly to the same extraction folder:

```
*figs_to_extract
        + caldat
                  + 001_Anderson_1935_Fig1
    + 001_Anderson_1935_Fig1.png
    + 002_Doe_2013_Fig1.png
    + 002_Doe_2013_Fig3.png
```

We have already processed one figure (001_Anderson_1935_Fig1.png) and we can tell because it has digitised data (caldat/001_Anderson_1935_Fig1), but now we have our two new figures that have not yet been digitised. This example will nicely demonstrate how users can easily pick up from where they left off and how all previous data gets re-integrated. It will also demonstrate how different plot types are handled. All we have to do to begin, is again, provide the directory where all the figures are located:

```
R> setwd("~/figs_to_extract")
R> digitised_data <- metaDigitise(".", summary = TRUE)
```

The user gets the same set of prompts and simply chooses option one to digitise the new figures. This will permit users to digitise new figures, and will integrate previously completed digitisations along with newly digitised data together at the end of the session, or when the user decides to quit. This time, 001_Anderson_1935_Fig1.png, is ignored and the new plots cycle on screen. First for 002_Doe_2013_Fig1.png and then 002_Doe_2013_Fig3.png. Since there are a few new figures, we answer the first question in the R console as "diff":

```
Are all plot types the same? (diff/same)

R> diff

**** NEW PLOT ****

mean_error and boxplots should be vertically orientated

       _
       |
  I.E. o    NOT  |-o-|
       |
       _

If they are not then chose flip to correct this.

If figures are wonky, chose rotate.

Otherwise chose continue

Flip, rotate or continue f/r/c

R> c

Please specify the plot_type as either: mean and error, box plot,
scatter plot or histogram m/b/s/h:

R> m
```

Here, we specify the new plot type as `m` for 002_Doe_2013_Fig1.png because we have a plot of the mean and error of sepal length for each of the three species. We're then prompted a bit differently from our scatter plot as the x-axis is not needed for calibration:

```
What is the variable?

R> Sepal length

On the Figure, click IN ORDER:
      y1, y2


    Step 1 ----> Click on y1
  |
  |
  |
  |
  y1
  |_________________________


    Step 2 ----> Click on y2
  |
  y2
  |
  |
  |
  |_________________________

What is the value of y1 ?

R> 5

What is the value of y2 ?

R> 6.5

Re-calibrate? (y/n)

R> n

Enter sample sizes? y/n

R> y

Group identifier:

R> setosa

Group sample size:
```

```
R> 50

Click on Error Bar, followed by the Mean

Add group, Delete group or Finish plot? a/d/f

R> a
```

Again, `metaDigitise()` will simply guide the user through digitising each of these figures describing to them exactly what needs to be done. At any point if mistakes are made the user can choose relevant options to edit or correct things. This can continues for each plot so long as the user would like to continue after completing a single plot:

```
Do you want continue: 1 plots out of 2 plots remaining (y/n)

R> y
```

This continues until users have completed all non-digitised figures in the folder, at which point `metaDigitise()` concatenates the new data with previously digitised data in the object:

```
data
                filename    group_id          variable     mean        sd  n         r
1 001_Anderson_1935_Fig1.png      setosa  Sepal Width (mm) 3.420332 0.4034259 39 0.7524909
2 001_Anderson_1935_Fig1.png      setosa Sepal Length (mm) 4.998267 0.3831294 39 0.7524909
3 001_Anderson_1935_Fig1.png versicolor  Sepal Width (mm) 2.766857 0.3234723 44 0.5157938
4 001_Anderson_1935_Fig1.png versicolor Sepal Length (mm) 5.936769 0.5292535 44 0.5157938
5      002_Doe_2013_Fig1.png      setosa      Sepal length 5.000336 0.7828656 50        NA
6      002_Doe_2013_Fig1.png versicolor      Sepal length 5.931340 1.0466177 50        NA
7      002_Doe_2013_Fig1.png viriginica      Sepal length 6.588705 1.2608173 50        NA
8      003_Doe_2013_Fig3.png        <NA>      Sepal length 4.948472 0.3624212 50        NA
```

5. Re-importing, Editing and Plotting Previously Digitised data

A particularly useful feature of **metaDigitise** is its ability to re-import, edit and re-plot previously digitised figures. We can do this from the initial options from `metaDigitise()`

```
R> metaDigitise("./FiguresToExtract")

    Do you want to...
1: Process new images
2: Import existing data
3: Edit existing data
Selection:
```

If the user chooses "Import existing data", they have the option of either 1) importing data from all digitised images or 2) importing data from particular image that has been digitised. If 2, then a list of files are provided to the user that they can select. Editing existing data allows users to easily re-plot or edit information or digitisations that have previously be done for any plot. This is accomplished by guiding the user through a new set of options:

```
Choose how you want to edit files:
1: Cycle through images
2: Choose specific file to edit
3: Enter previously omitted sample sizes
Selection:
```

If the user is unsure about the name of the specific figure they need to edit or simply want to just check the digitisations of figures they can choose "Cycle through images", which will bring up each figure, one by one, overlaying the calibrations, group names (if they exist), sample sizes (if they were entered) and the selected points. The user will then be given the choice to edit individual images. Alternatively, choosing option 2, will bring up a list of the completed files in the folder and the specific file can be chosen, at which point it will be replotted. Either of these options will cycle through a number of questions asking the user what they would like to edit:

```
Edit rotation? If yes, then the whole extraction will be redone (y/n)

R> n

Change plot type? If yes, then the whole extraction will be redone (y/n)

R> n

Variable entered as:

R> Sepal length

Rename Variables (y/n)

R> n

Edit calibration? (y/n)

R> n

Re-extract data (y/n)

R> y

Change group identifier? (y/n)
```

```
R> n

Add group, Delete group or Finish plot? a/d/f

R> d

1: setosa
2: versicolor
3: viriginica
Selection:

R> 2

Add group, Delete group or Finish plot? a/d/f

R> a
```

A whole host of information can be edited from the rotation, plot type, the variable name that was provided, the calibration and even the digitisation of groups. When editing the `metaDigitise` object is re-written to the caldat folder and the edits are immediately integrated into the existing object once complete.

6. Additional Features

6.1. Figure Rotation and Adjustment

Figure may have been extracted from old publications, for example from scanned images, and so are not perfectly orientated on the image. This will make the calibration of the points in the figure from the image problematic. `metaDigitise()` allows users to rotate the image. By clicking two points on the x-axis, metaDigitse calculates the angle needed to rotate the image so the x-axis is horizontal, and rotates it. (Figure 5A,B)

Furthermore, in some figures mean and error, boxplots or histograms may be presented with horizontal bars. `metaDigitise()` assumes that the bars are vertical, but allows the user to flip the image so that the bars are vertical (Figure 5C,D).

6.2. Obtaining Raw Data

While `metaDigitise()` provides users with the summary statistics by default, for all plot types, in many cases the user may actually be interested in obtaining the raw digitised data from scatter plots. This is very easy to do my changing the default `summary` argument from TRUE to FALSE in `metaDigitise()`. Instead of providing the user with summary statistics it will return a list containing four slots for each of the figure types (mean error, box plot, histogram and scatter plots). An example of a data object returned from digitising figures is as follows:

```
>R str(data)
```

```
List of 3
 $ mean_error :List of 1
```

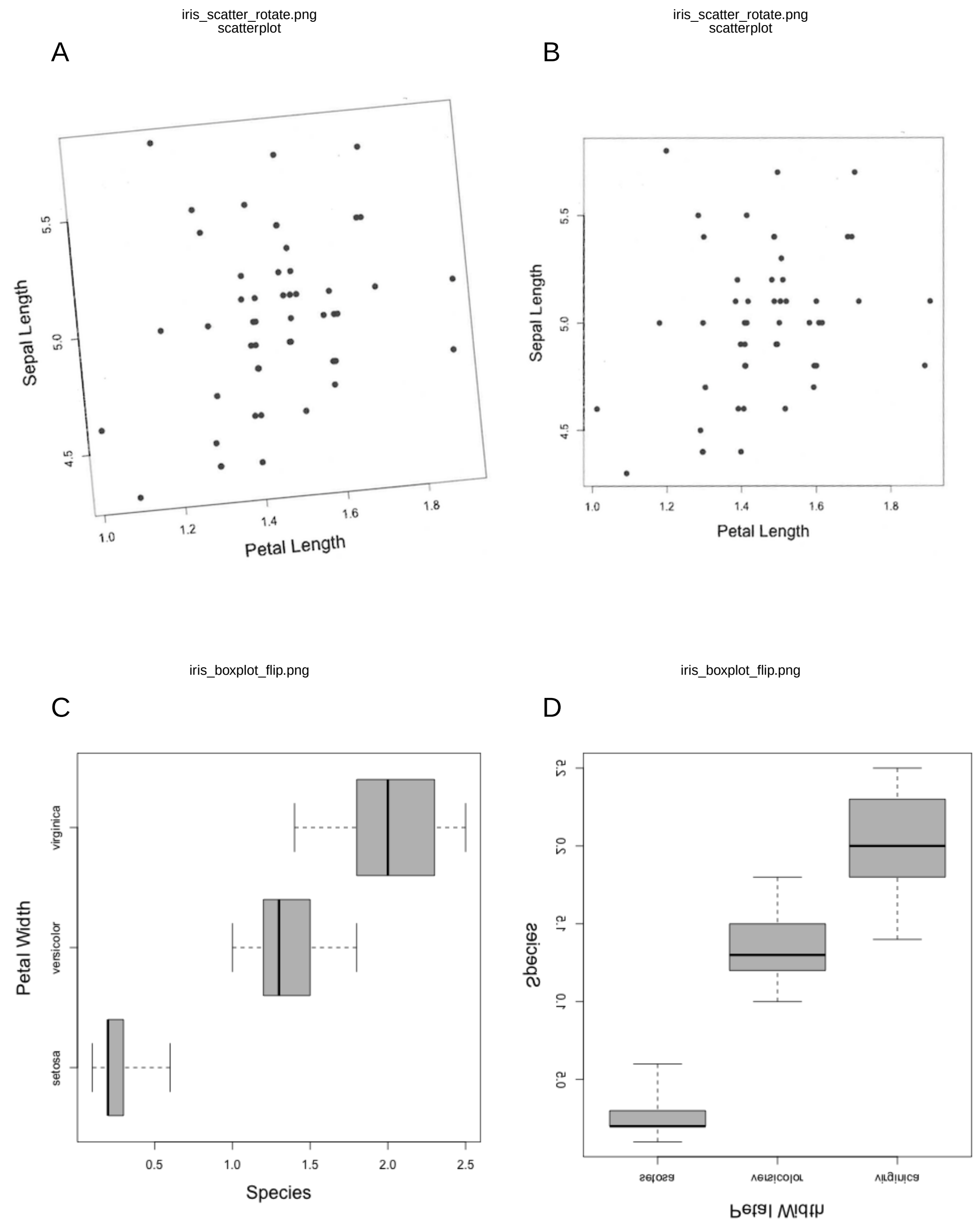

Figure 5: Figure rotation. A) and B) show how non-aligned imaged can be realigned through user defined rotation. C) and D) show show figures can be re-orientated so as to aid data input.

```
  ..$ 002_Doe_2013_Fig1.png:'data.frame': 3 obs. of  5 variables:
  .. ..$ id      : Factor w/ 3 levels "setosa","versicolor",..: 1 2 3
  .. ..$ mean    : num [1:3] 5 5.93 6.59
  .. ..$ error   : num [1:3] 0.111 0.148 0.178
  .. ..$ n       : num [1:3] 50 50 50
  .. ..$ variable: chr [1:3] "Sepal length" "Sepal length" "Sepal length"
 $ hist       :List of 1
  ..$ 003_Doe_2013_Fig3.png:'data.frame': 8 obs. of  3 variables:
  .. ..$ midpoints: num [1:8] 4.3 4.5 4.7 4.9 5.1 ...
  .. ..$ frequency: num [1:8] 4 5 7 12 11 6 2 3
  .. ..$ variable : chr [1:8] "Sepal length" "Sepal length" "Sepal length" "Sepal length"
 $ scatterplot:List of 1
  ..$ 001_Anderson_1935_Fig1.png:'data.frame':  83 obs. of  8 variables:
  .. ..$ id        : Factor w/ 2 levels "setosa","versicolor": 1 1 1 1 1 1 1 1 1 1 ...
  .. ..$ x         : num [1:83] 2.3 2.9 3 3 3 ...
  .. ..$ y         : num [1:83] 4.5 4.4 4.41 4.3 4.8 ...
  .. ..$ group     : num [1:83] 1 1 1 1 1 1 1 1 1 1 ...
  .. ..$ col       : Factor w/ 2 levels "red","green": 1 1 1 1 1 1 1 1 1 1 ...
  .. ..$ pch       : num [1:83] 19 19 19 19 19 19 19 19 19 19 ...
  .. ..$ y_variable: chr [1:83] "Sepal length (mm)" "Sepal length (mm)" "Sepal length (mm)
  .. ..$ x_variable: chr [1:83] "Sepal width (mm)" "Sepal width (mm)" "Sepal width (mm)" "
```

Here, the user can easily access the list of raw scatter plot data by simply extracting the scatter plot slot:

```
>R scatterplot <- data$scatterplot
```

7. Testing

8. Conclusions and Future Enhancements

metaDigitise is available through GitHub. WHile it is incredibly flexible already, our future plans include building in abilities to deal with log-transformed axes, including arguments for calculating standard deviations from 95% confidence intervals and standard errors using the t-distribution to correct the t-value for small sample sizes (currently assumed t-values are 1.96 as is normal), dealing with asymmetric error bars and the possibility of zooming in plots such that greater accuracy can be achieved when digitising. We hope to also provide options for assessing inter-observer reliability in the future.

Acknowledgments

We thanks the i-deel group at UNSW for useful feedback, and ...

References

Wan X, Wang W, Liu J, Tong T (2014). "Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range." *BMC medical research methodology*, **14**(1), 135. ISSN 1471-2288. `doi:10.1186/1471-2288-14-135`. `arXiv:1407.8038v3`, URL `http://www.ncbi.nlm.nih.gov/pubmed/25524443`.

Affiliation:

Joel L. Pick, Shinichi Nakagawa, Daniel W.A. Noble
Ecology and Evolution Research Center
and
School of Biological, Earth and Environmental Sciences
University of New South Wales
b
b
E-mail: `joel.l.pick@gmail.com`
URL: `https://eeecon.uibk.ac.at/~zeileis/`