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

Supplementary Materials

Joel L. Pick, Shinichi Nakagawa & Daniel W.A. Noble

June 19, 2018

Contents

S1 Tutorial

S1.1 Extracting Data From Plots

We can demonstrate how `metaDigitise()` works using figures generated from the well known iris data set. **metaDigitise** can installed from GitHub as follows:

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

Assume you would like to extract descriptive 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 you read papers found from a systematic search, you add figures with relevant data to a "FiguresToExtract" folder as follows

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

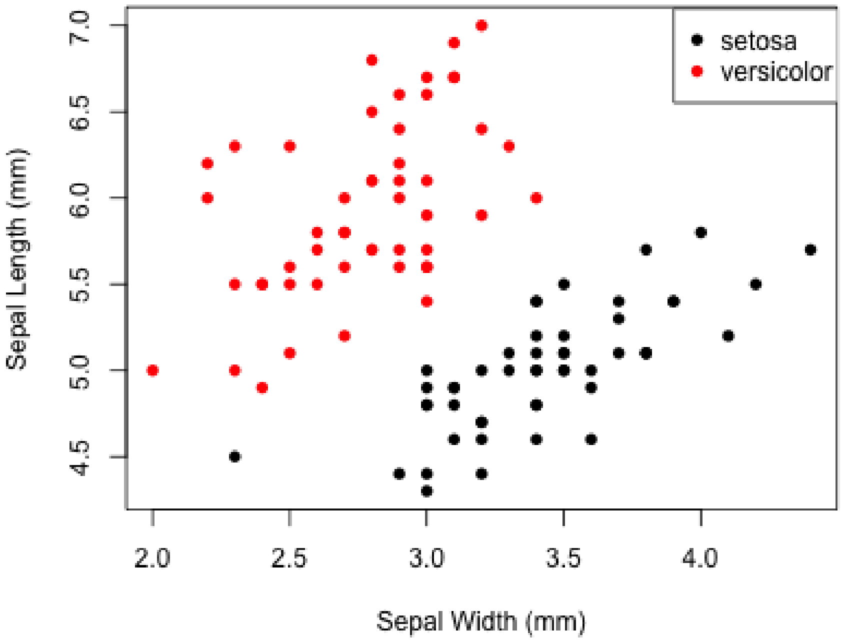

Figure S1: Example scatterplot (001_Anderson_1935_Fig1.png) of sepal length and width for two species of iris (setosa and versicolor)

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. At first there is one figure in the folder, shown in Figure S1. Running `metaDigitise()` brings up a series of prompts using a main menu that provides access to a number of its features (”...” here represents the user’s path to the project directory):

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

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

You then simply enter in the numeric value that corresponds to what you would like to do. In this case we want to ”Process new images”. You are then asked whether there are 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 for the type of figure as they are cycled through.

```
Are all plot types Different or the Same? (d/s)
```

`metaDigitise()` then asks 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. 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)
```

```
R> c
```

After this, `metaDigitise()` will ask you to specify the plot type. Depending on the figure, you 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 you has specified `d` instead of `s` 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:

 m: Mean and error
 b: Box plot
 s: Scatter plot
 h: Histogram
```

```
R> s
```

After selecting the figure type a new set of prompts will come up that will ask 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, you 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 errors, the x-axis is rather useless in terms of calibration so `metaDigitise()` just asks you 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 known value on y axis - y1
  |
  |
  |
  |
  y1
  |_________________________
  ....

    Step 3 ----> Click on known value on x axis - x1
  |
  |
  |
  |
  |
  |_____x1__________________

  ....
```

You 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, you are forced to check whether or not the calibration has been set up correctly. If `n` is chosen because something needs to be fixed then you 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)

R> n
```

Often, plots might contain multiple groups that a meta-analyst wants to extract from. `metaDigitise()` handles this nicely by prompting you to enter the group first, followed by digitisation of this groups data. After digitising the first group, `metaDigitise()` will ask whether you would like to add another group. You can continually add groups (`a`), delete groups (`d`), edit groups (`e`) or finish a plot and continue to the next one (`f -` if another plot exists). The number of groups are not really limited and you 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).

```
If there are multiple groups, enter unique group identifiers (otherwise press enter)
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 by clicking on red box in bottom left corner, then follow prompts
```

To finish selecting points, you can exit by clicking on the red button that appears when extracting points. You are then asked if you want to add or delete points from that group.

```
Add or Delete points to this group, or Continue? (a/d/c)
```

```
R> c
```

Once we are done digitising all the groups our plot will look something like Figure S2.

When completed `metaDigitise()` will write the digitised data as a `metaDigitise` object to a RDS file in the caldat directory, such that our new directory structure is as follows

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

You 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  error error_type  n     r    sd  plot_type
001_Anderson_1935_Fig1.png      setosa  Sepal width (mm)   3.42  0.40  sd        39  0.75  0.40  scatterplot
001_Anderson_1935_Fig1.png      setosa  Sepal length (mm)  5.00  0.38  sd        39  0.75  0.38  scatterplot
001_Anderson_1935_Fig1.png  versicolor  Sepal width (mm)   2.77  0.32  sd        44  0.52  0.32  scatterplot
001_Anderson_1935_Fig1.png  versicolor  Sepal length (mm)  5.95  0.53  sd        44  0.52  0.53  scatterplot
```

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

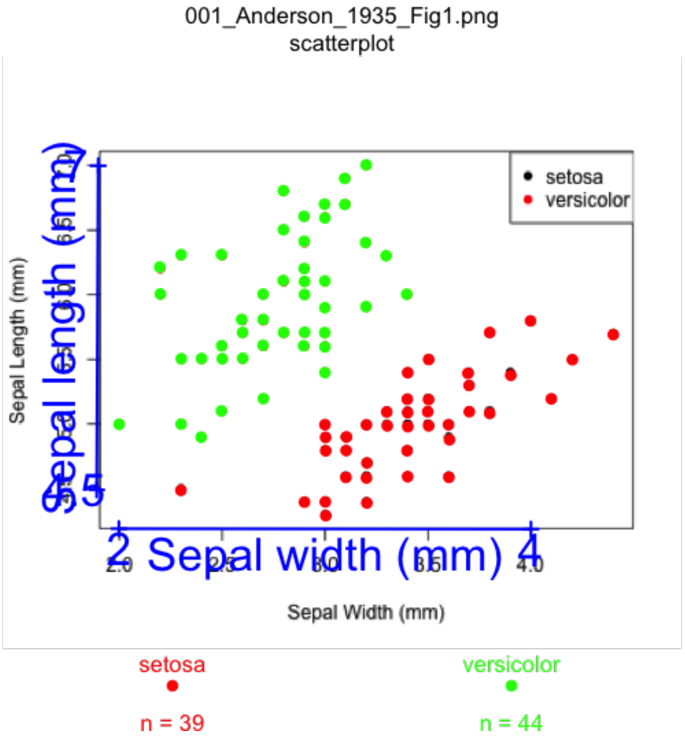

Figure S2: Digitisation of sepal length and width for two species of iris (setosa and versicolor). Names of the variables and calibration (in blue) are plotted alongside the digitised points (green = versicolor; red = setosa). The sample sizes for each group are provided on the lower part of the plot. All figures are clearly labelled at the top to remind users of the filename and plot type. This reduces errors throughout the digitisation process.

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 you with options to input the sample sizes directly, 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 sampling variance. Here, the mean point estimates are nearly exactly the same as the true values, but the SD's are slightly over-estimated:

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

S1.2 Adding new figures

You can add additional figures to the directory 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:

```
*FiguresToExtract/
    + 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). We can tell this because the caldat folder has digitised data in it (caldat/001_Anderson_1935_Fig1). Now there are two new figures that have not yet been digitised. This example will also nicely demonstrate how you can easily pick up from where you left off and how 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> digitised_data <- metaDigitise("~/FiguresToExtract", summary = TRUE)
```

You get the same set of prompts and simply chooses option one. 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 you decides to quit. This time, 001_Anderson_1935_Fig1.png is ignored and the new plots cycle on screen; first 002_Doe_2013_Fig1.png and then 002_Doe_2013_Fig3.png. Since there are a few different figure types, you answer the first question in the R console as "d":

```
Are all plot types Different or the Same? (d/s)
```

```
R> d

**** 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:

 m: Mean and error
 b: Box plot
 s: Scatter plot
 h: Histogram

R> m
```

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

```
What is the y 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

Do you know sample sizes? (y/n)

R> y

If there are multiple groups, enter unique group identifiers (otherwise press enter)
Group identifier:

R> setosa

Group sample size:

R> 50

Click on Error Bar, followed by the Mean
```

```
Add group, Edit Group, Delete group or Finish plot? (a/e/d/f)
```

```
R> a
```

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

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

```
R> y
```

This continues until you 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  error error_type n    r    sd   plot_type
001_Anderson_1935_Fig1.png      setosa  Sepal width (mm)  3.42  0.40  sd        39  0.75  0.40  scatterplot
001_Anderson_1935_Fig1.png      setosa  Sepal length (mm) 5.00  0.38  sd        39  0.75  0.38  scatterplot
001_Anderson_1935_Fig1.png  versicolor  Sepal width (mm)  2.77  0.32  sd        44  0.52  0.32  scatterplot
001_Anderson_1935_Fig1.png  versicolor  Sepal length (mm) 5.95  0.53  sd        44  0.52  0.53  scatterplot
     002_Doe_2013_Fig1.png      setosa  Sepal length      5.00  0.11  se        50  NA    0.78  mean_error
     002_Doe_2013_Fig1.png  viriginica  Sepal length      6.59  0.18  se        50  NA    1.26  mean_error
     002_Doe_2013_Fig1.png  versicolor  Sepal length      5.94  0.14  se        50  NA    1.01  mean_error
     003_Doe_2013_Fig3.png      catana  Sepal length      4.95  0.36  sd        50  NA    0.36  histogram
```

S1.3 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> digitised_data <- metaDigitise(".../FiguresToExtract")
```

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

If we choose ”Import existing data”, you have the option of either 1) importing data from all digitised images or 2) importing data from a single image that has been digitised. If

2, then a list of files are provided for you to select. Editing existing data allows you to easily re-plot or edit information or digitisations that have previously be done for any plot. This is accomplished by guiding you 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 you is unsure about the name of the specific figure you 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 you 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:
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 including the rotation, plot type, the variable name(s) that were 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.

S1.4 Obtaining Calibrated Data

While metaDigitise() provides you with the summary statistics by default, for all plot types, in many cases the user may actually be interested in obtaining the calibrated data from scatter plots (i.e. all calibrated points). This is very easy to do by changing the default summary argument from TRUE to FALSE in metaDigitise(). Instead of providing you 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
  ..$ 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" ...
 $ 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 ...
  .. ..$ 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 ...
  .. ..$ 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)"  ...
  .. ..$ x_variable: chr [1:83] "Sepal width (mm)" "Sepal width (mm)"    ...
```

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

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

Additionally there is another function, `getExtracted()`, that imports all digitised data from a directory, as either summarised or calibrated data. For example,

```
>R getExtracted(".../FiguresToExtract", summary=TRUE)
```

```
                  filename    group_id         variable   mean  error error_type n    r    sd   plot_type
001_Anderson_1935_Fig1.png      setosa  Sepal width (mm)  3.42  0.40  sd        39  0.75  0.40  scatterplot
001_Anderson_1935_Fig1.png      setosa  Sepal length (mm) 5.00  0.38  sd        39  0.75  0.38  scatterplot
001_Anderson_1935_Fig1.png  versicolor  Sepal width (mm)  2.77  0.32  sd        44  0.52  0.32  scatterplot
001_Anderson_1935_Fig1.png  versicolor  Sepal length (mm) 5.95  0.53  sd        44  0.52  0.53  scatterplot
     002_Doe_2013_Fig1.png      setosa  Sepal length      5.00  0.11  se        50  NA    0.78  mean_error
     002_Doe_2013_Fig1.png  viriginica  Sepal length      6.59  0.18  se        50  NA    1.26  mean_error
     002_Doe_2013_Fig1.png  versicolor  Sepal length      5.94  0.14  se        50  NA    1.01  mean_error
     003_Doe_2013_Fig3.png      catana  Sepal length      4.95  0.36  sd        50  NA    0.36  histogram
```

This function is useful in scripts the extracted data simply needs to be imported, and where the interactive element of `metaDigitise()` becomes problematic.

S1.5 Adding sample sizes to previous Digitisations

In many cases important information, such as sample sizes, may not be readily available or clear when digitising figures. In these circumstances you would have answered 'no' to the question about whether you have sample sizes or not while digitising. To expedite finding and adding in these sample sizes to do the necessary calculations (if for example a figure presented 95% CI's or standard errors), `metaDigitise()` has s specific edit option that allows users to enter in previously omitted sample sizes. It works by first identifying the missing sample sizes in the digitised output, re-plotting the relevant figure and then prompting you to enter the sample sizes for the relevant groups in the figure, one by one. As an example, assume that we were missing sample sizes for two groups in 002_Doe_2013_Fig1.png:

```
            filename    group_id         variable   mean  error error_type n    r    sd   plot_type
002_Doe_2013_Fig1.png      setosa  Sepal length      5.00  0.11  se        NA  NA    NA  mean_error
002_Doe_2013_Fig1.png  viriginica  Sepal length      6.59  0.18  se        NA  NA    NA  mean_error
```

Here, we can see that we are missing the sample sizes for setosa and viriginica, and as a result, sd is not calculated because `metaDigitise()` needs this information to make the calculation. If, for examnple, you found this information after contacting the authors for clarification then you can add these in as follows:

```
R> digitised_data <- metaDigitise(".../FiguresToExtract")

Do you want to...

1: Process new images
2: Import existing data
3: Edit existing data

Selection:

R> 3

Choose how you want to edit files:

1: Cycle through images
2: Choose specific file to edit
3: Enter previously omitted sample sizes

Selection:

>R 3
```

`metaDigitise()` will replot the figure after this and list, only the groups missing data, for which you can then update the data. This is then re-integrated back into the data automatically and the sd calculated.

```
Group " setosa ": Enter sample size

R> 50
```

```
Group " viriginica ": Enter sample size

R> 50
```

S2 Derivation of mean, standard deviation and sample size from different plot types

S2.1 Mean/Error Plots

The standard deviation is calculated depending on the type of error presented. The user can choose from standard deviation (SD, σ), standard error (SE) or 95% confidence intervals (CI95). Standard deviation is calculated from standard error as

$$\sigma = SE\sqrt{n} \tag{S1}$$

and from 95% confidence intervals as

$$\sigma = \frac{CI}{1.96}\sqrt{n} \tag{S2}$$

S2.2 Box Plots

The mean (μ) and SD are calculated sing the maximum (b), upper quartile (q_3), median (m), lower quartile (q_1) and minimum (a) as

$$\mu = \frac{(n+3)(a+b)+2(n-1)(q_1+m+q_3)}{8n} \tag{S3}$$

following Bland (2015) and

$$\sigma = \frac{b-a}{4\Phi^{-1}(\frac{n-0.375}{n+0.25})} + \frac{q_3-q_1}{4\Phi^{-1}(\frac{0.75n-0.125}{n+0.25})} \tag{S4}$$

where $\Phi^{-1}(z)$ is the upper zth percentile of the standard normal distribution, following Wan et al. (2014).

S2.3 Histograms

For each bar, the user click two point (the top of the bar). Using these points, a midpoint (m; mean x coordinates) and a frequency (f; mean y coordinates, rounded to the nearest integer) is calculated for each bar. The sample size, mean and SD are calculated as:

$$n = \sum_{i=1}^{n} f_i \tag{S5}$$

$$\mu = \frac{\sum_{i=1}^{n} m_i f_i}{n} \tag{S6}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (m_i f_i - \mu f_i)^2}{n-1}} \tag{S7}$$

S3 Software Validation: Details of Simulations

S3.1 Inter-observer variability in digitisations

In order to evaluate the consistency of digitisation using **metaDigitise** between users, we simulated a dataset of two variables with two groups ($n = 10$ within groups). Each variable was plotted twice for each plot type (figures were modified slightly to give users a sense that they were digitising new data) generating a total of 14 figures. Fourteen independent digitisers (including the authors) were provided with a directory with all 14 figures in a randomised order. Digitisers ran **metaDigistise** on their own computers, across different operating systems (including Mac, Windows and Linux). Digitisers varied in their level of experience, from people with experience of meta-analyses to those without any science background. We asked users to digitise all 14 figures and collected the mean, standard deviation and correlation coefficient (r, for scatter plots) generated by `metaDigitise()` for every plot digitised ($n = 28$ per digitiser per metric, $n = 4$ for r).

As a measure of bias, we calculated the percentage differences from the true summary statistics as

$$\frac{\theta - \hat{\theta}}{\hat{\theta}} \tag{S8}$$

where θ is the estimate and $\hat{\theta}$ is the true value. The deviation from the true value of r was not further standardised, as it is already on a standardised scale. We also took the absolute values of these standardised differences as a measure of precision. The resulting data was used to assess between- and within- user variability (i.e., the intra-class correlation coefficient - ICC). This was done using linear mixed effect models with user identity as a random effect using **lme4** (Bates et al., 2015) in R. Standardised mean, standard deviation and correlation coefficients were used as response variables in separate models. Sampling variance for ICC estimates was generated based on 1000 parametric bootstraps of the model and the significance was tested using likelihood ratio tests, using **rptR** (Stoffel, Nakagawa & Schielzeth, 2017).

S3.2 Testing the accuracy of digitisations

To test how accurate **metaDigitise** is at matching clicked points to their true values, we generated four random scatterplots, each with 20 data points, and digitised these with `metaDigitise()`. This was done by one digitiser (J.L.P.), as there is no detectable between user variation.

S4 Comparison with existing software

Function	metaDigitise	GraphClick[1]	DataThief[2]	DigitizeIt[3]	WebPlotDigitizer[4]	metagear[5]	digitize[6]
Scatterplots	✓	✓	✓	✓	✓	✓[7]	✓
Mean/error plots	✓	✓	✓	×	×	✓[7]	×
Boxplots	✓	×	×	×	×	×	×
Histograms	✓	×	×	×	✓[7]	×	×
Graph rotation[8]	✓	✓	✓	✓	✓	×	×
Groups	✓	✓	×	✓	✓	×	×
Entry of metadata	✓	×	×	×	×	×	×
Summarising data	✓	×	×	×	×	×	×
Multiple image processing	✓	×	×	×	×	×	×
Reproducable[9]	✓	✓	✓	×	✓	✓	×
Automated point detection	×	✓	×	✓	✓	✓	×
Line extraction	×	✓	✓	✓	✓	×	×
Zoom	×	✓	✓	✓	✓	×	×
Log axis	✓	✓	✓	✓	✓	×	×
Dates	×	×	✓	×	✓	×	×
Asymmetric error bars	×	×	✓	×	×	×	×
Freeware	✓[10]	✓[11]	✓[11]	×[11]	✓[11]	✓[10]	✓[10]

[1] Arizona-Software (2008) [2] Tummers (2006) [3] Bormann (2012) [4] Rohatgi (2017) [5] Lajeunesse (2016) [6] Poisot (2011)

[7] Only automated, no manual extraction.

[8] Or handles rotated graphs.

[9] Allows saving, re-plotting and editing of data extraction.

[10] R package.

[11] Standalone software.

Table S1: Comparison of functionality between different digitisation softwares.

References

Arizona-Software (2008) *GraphClick Software, Version 3.0.*

Bates, D., Maechler, M., Bolker, B. & Walker, S. (2015) Fitting Linear Mixed-Effects Models Using lme4. *Journal of Statistical Software*, **67**, 1–48.

Bland, M. (2015) Estimating Mean and Standard Deviation from the Sample Size, Three Quartiles, Minimum, and Maximum. *International Journal of Statistics in Medical Research*, **4**, 57–64.

Bormann, I. (2012) *DigitizeIt Software, Version 2.0.* Braunschweig, Germany.

Lajeunesse, M.J. (2016) Facilitating systematic reviews, data extraction, and meta-analysis with the metagear package for R. *Methods in Ecology and Evolution*, **7**, 323–330.

Poisot, T. (2011) The digitize package: extracting numerical data from scatterplots. *The R Journal*, **3**, 25–26.

Rohatgi, A. (2017) *WebPlotDigitizer Software, Version 4.0.* Austin, Texas, USA.

Stoffel, M.A., Nakagawa, S. & Schielzeth, H. (2017) rptR: repeatability estimation and variance decomposition by generalized linear mixed-effects models. *Methods in Ecology and Evolution*, **8**, 1639–1644.

Tummers, B. (2006) *DataThief Software, Version 3.0.*

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**, 135.