Evaluating the Suitability of Lithic Illustrations in Morphometric Analyses

Christian Steven Hoggard, Thomas Birch, Cory Marie Stade, Katrien Janin and Felix Riede

Abstract

Illustrations of lithic artefacts are an abundant source of morphological and technological information for those interested in our human past. As a typical part of archaeological reports and publications, lithic drawings are - or have to be - trusted as faithful reproductions of the selected artefacts. Despite the considerable epistemic work lithic illustrations (and illustrators) are expected to do, usually little information is available regarding the illustrator's technical skill; thus, it remains unknown whether drawings produced by illustrators of differing technical skill are comparable or produce images of equal analytical potential to other media, e.g. photographs. The issue of lithic illustration accuracy is brought to the fore by the recent emergence of geometric morphometric approaches as innovative and powerful ways of describing and analysing complex shapes, as lithic illustrations provide one of the key sources for such analyses. Motivated by these issues, we present an experiment investigating the degree of error observed in illustrations of differing technical illustrative skill. Analyses suggest that lithic illustrations produced by individuals with a variety of experience in drawing lithics create, in the majority of instances, equally faithful representations (in outline shape) of chipped stone artefacts. With error observed in a small number of instances, archaeologists are still urged to be critical of an illustration's source prior to lineal and geometric morphometric methodologies. Despite this, archaeologists can be confident in their exactitude and we remain strong advocates in favour of lithic illustrations as a readily available legacy resource for morphometric analyses.

Introduction

This is a R Markdown documented associated with the article *Evaluating the Suitability of Lithic Illustrations in Morphometric Analyses*, original research for *The Journal of the Lithic Studies Society* (http://www.lithics.org/). This Markdown document details the analytical procedure used throughout the article, from data importing and the packages used, to the univariate and multivariate framework underpinning the article's findings.

Data files

For this markdown, a number of different files are necessary:
1. The three raw landmark files (for handaxe, tanged point and elongated artefacts), these are: **elongated.tps**, **handaxe.tps** and **tanged.tps**
2. The three data frames for the respective landmark files: **elongated.csv**, **handaxe.csv** and **tanged.csv**
3. The raw measurement length and width data for all artefacts: **measurement_data.csv**
4) The files necessary to investigate landmark digitisation error: **digitisation_error.tps** and **digitisation_error.csv**
5) The file necessary to investigate measurement error: **measurement_error.csv**
6) The table of links for all landmark configurations: **curveslide.csv**

A copy of all files can also be found on GitHub and the Open Science Framework (OSF).
GitHub: https://github.com/CSHoggard/-Lithic_Illustrations
OSF: https://osf.io/xtghn/

Stage 1: Package installation and data extraction

For this article, ten packages are required:
1. **psych v.1.8.12** https://cran.r-project.org/web/packages/psych/index.html
2. **geomorph v.3.1.2** https://cran.r-project.org/web/packages/geomorph/index.html
3. **tidyverse v.1.2.1** https://cran.r-project.org/web/packages/tidyverse/index.html
4. **vegan v.2.5-4** https://cran.r-project.org/web/packages/vegan/index.html
5. **MASS v.7.3-51.4** https://cran.r-project.org/web/packages/MASS/index.html
6. **cowplot v.0.9.3** https://cran.r-project.org/web/packages/cowplot/index.html
7. **ggpubr v.0.9.3** https://cran.r-project.org/web/packages/ggpubr/index.html
8. **rio v.0.5.16** https://cran.r-project.org/web/packages/rio/index.html
9. **LaMBDA v.0.1.0.9000** https://rdrr.io/github/akiopteryx/lambda/
10. **devtools v.2.3.1** https://cran.r-project.org/web/packages/devtools/index.html

The latest versions of each package can be installed and sourced through the following code:

```
if(!require("psych")) install.packages('psych', repos='http://cran.us.r-project.org')
if(!require("geomorph")) install.packages('geomorph', repos='http://cran.us.r-project.org')
if(!require("tidyverse")) install.packages('tidyverse', repos='http://cran.us.r-project.org')
if(!require("vegan")) install.packages('vegan', repos='http://cran.us.r-project.org')
if(!require("MASS")) install.packages('MASS', repos='http://cran.us.r-project.org')
if(!require("cowplot")) install.packages('cowplot', repos='http://cran.us.r-project.org')
if(!require("ggpubr")) install.packages('ggpubr', repos='http://cran.us.r-project.org')
if(!require("rio")) install.packages('rio', repos='http://cran.us.r-project.org')
```

The `LaMBDA` package (LandMark-Based Data Assessment) is not on CRAN and requires downloading from GitHub (via the `devtools` package):

```
install.packages("devtools")
devtools::install_github("akiopteryx/lambda")
```

Should specific functions become outdated then please refer to the package versions as cited above. Once downloaded, activate all packages:

```
library(psych) ### load the listed package
library(geomorph) ### load the listed package
library(tidyverse) ### load the listed package
library(vegan) ### load the listed package
library(MASS) ### load the listed package
library(cowplot) ### load the listed package
library(ggpubr) ### load the listed package
library(LaMBDA) ### load the listed package
library(rio) ### load the listed package
```

Stage 2: Bringing all files into the R Environment

Given the large number of files required for the individual analyses (shape analysis, metric analysis, digitisation error and measurement error), and to prevent issues of working directories, all files are brought into the R Environment as .rds objects; these are objects which were imported into R and stored on GitHub (and downloaded when required). Using the `rio::import()` function allows all objects to be brought into the workspace.

```
landmarks_elongated <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/landmarks_e
landmarks_handaxe <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/landmarks_han
landmarks_tanged <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/landmarks_tang
shape_data_elongated <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/shape_data
shape_data_tanged <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/shape_data_ta
```

```
shape_data_handaxe <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/shape_data_l
metric_data <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/metric_data.rds")
digitisation_error_landmarks <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/di
digitisation_error_landmarks_data <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/mast
digitisation_error_metrics <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/dig
shape_data_sliders <- import("https://github.com/CSHoggard/-Lithic_Illustrations/raw/master/shape_data_s
```

Stage 3: Sources of Error and Investigating Landmark Count

Stage 3A: Digitisation and Landmark Error

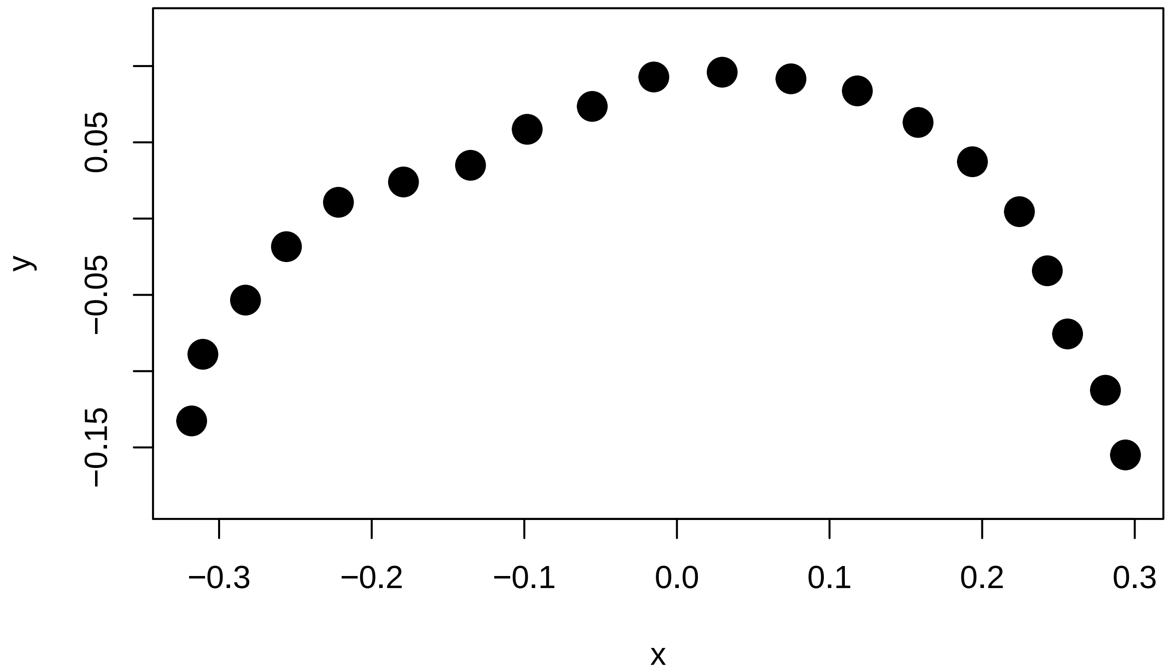
A Procrustes ANOVA (see in-text) was performed to calculate error associated with the digitisation of landmarks.

```
gpa_digi_error <- gpagen(digitisation_error_landmarks, Proj = TRUE, curves =  shape_data_sliders, ProcD
gpa_digi_error  ### calls the GPA landmark configuration
```

```
##
## Call:
## gpagen(A = digitisation_error_landmarks, curves = shape_data_sliders,
##     surfaces = NULL, ProcD = TRUE, Proj = TRUE, print.progress = FALSE)
##
##
##
## Generalized Procrustes Analysis
## with Partial Procrustes Superimposition
##
## 2 fixed landmarks
## 18 semilandmarks (sliders)
## 2-dimensional landmarks
## 3 GPA iterations to converge
## Minimized squared Procrustes Distance used
##
##
## Consensus (mean) Configuration
##
##              X           Y
## 1  -0.31796682 -0.132644965
## 2  -0.31060449 -0.088929377
## 3  -0.28269701 -0.053436920
## 4  -0.25585407 -0.018405380
## 5  -0.22179239  0.010680904
## 6  -0.17917516  0.024001159
## 7  -0.13525110  0.034933759
## 8  -0.09809871  0.058525436
## 9  -0.05547201  0.073540152
## 10 -0.01513393  0.092829862
## 11  0.02965721  0.095950162
## 12  0.07472658  0.091629691
## 13  0.11822149  0.083726851
## 14  0.15801543  0.063045388
## 15  0.19363531  0.037275511
## 16  0.22441752  0.004551561
## 17  0.24271449 -0.034206469
## 18  0.25597780 -0.075621222
```

```
## 19  0.28077551 -0.112520244
## 20  0.29390492 -0.154928393
```

```
plot(gpa_digi_error) ### visualisation of the landmark configuration
```




```
gpa_digi_error_df <- geomorph.data.frame(gpa_digi_error, attempt = digitisation_error_landmarks_data$Att
gpaprocD <- procD.lm(coords ~ attempt, data = gpa_digi_error_df, print.progress = FALSE) ### ANOVA (coo
summary(gpaprocD) ### summary
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df         SS         MS   Rsq      F       Z Pr(>F)
## attempt    1 1.1246e-06 1.1246e-06 0.086 0.2823 -2.2192 0.9915
## Residuals  3 1.1952e-05 3.9841e-06 0.914
## Total      4 1.3077e-05
##
## Call: procD.lm(f1 = coords ~ attempt, data = gpa_digi_error_df, print.progress = FALSE)
```

```
gpaprocD$aov.table$SS[1]/gpaprocD$aov.table$SS[3]*100 ### error expressed as a percentage (8.599812%)
```

```
## [1] 8.599812
```

Stage 3B: Measurement Error

To calculate error associated with lineal measurements fractional uncertainity (standard error divided by the mean) was calculated:

```
head(digitisation_error_metrics)
```

```
##                   Attempt Scale_Factor Length_mm Width_mm
## T_Image_1_Copy_1       1     0.026200     82.15    22.51
## T_Image_1_Copy_2       2     0.026390     82.77    22.61
## T_Image_1_Copy_3       3     0.026080     81.80    22.29
## T_Image_1_Copy_4       4     0.026140     82.01    22.30
## T_Image_1_Copy_5       5     0.026110     81.98    22.35
## T_Image_1_Copy_6       6     0.026316     82.55    22.26
```

```
statsl  <- describe(digitisation_error_metrics$Length_mm) ### descriptive statistics
statsw  <- describe(digitisation_error_metrics$Width_mm) ### descriptive statistics
statssf <- describe(digitisation_error_metrics$Scale_Factor) ### descriptive statistics

(statsl$se/statsl$mean)   * 100 ### fractional uncertainity (length)
```

```
## [1] 0.1188221
```

```
(statsw$se/statsw$mean)   * 100 ### fractional uncertainity (width)
```

```
## [1] 0.2275143
```

```
(statssf$se/statssf$mean) * 100 ### fractional uncertainity (scale factor: as calibrated through tpsDig
```

```
## [1] 0.1223673
```

Stage 3C: Investigating Landmark Count

The `LaMBDA::LaSEC()` function is used to assess the fidelity of morphological characterisation. This function helps the user to identify under- and over-sampling of landmarks, the robustness of the characterisation and determine how many landmarks can be removed without compromising the necessary shape information. A two-dimensional array of each landmark file was used, with 500 iterations, to determine if twenty landmarks were suitable. 500 iterations were used for the analysis, here 10 iterations are exemplified (for ease and speed):

```
lasec(two.d.array(landmarks_elongated), 2, iter = 10, show.progress = F) ### may take some time
```

```
## $fit
##       [,1] [,2]       [,3]      [,4]      [,5]      [,6]      [,7]      [,8]
##  [1,]   NA   NA 0.16353116 0.2532542 0.3203179 0.3921320 0.4496996 0.8641523
##  [2,]   NA   NA 0.52824807 0.6014112 0.6811728 0.7021068 0.7223433 0.8103473
##  [3,]   NA   NA 0.33084838 0.4658694 0.6749621 0.7175439 0.7704487 0.8053725
##  [4,]   NA   NA 0.10452644 0.4400741 0.6158463 0.6742656 0.7702470 0.7889963
##  [5,]   NA   NA 0.08285789 0.2776336 0.6209322 0.6558858 0.7371672 0.7439683
##  [6,]   NA   NA 0.14387346 0.5701167 0.6745579 0.6741968 0.7714671 0.8127097
##  [7,]   NA   NA 0.38129853 0.5247893 0.6657151 0.7078289 0.8160774 0.8225324
##  [8,]   NA   NA 0.25844652 0.3773386 0.6950926 0.8182842 0.8504605 0.8818444
##  [9,]   NA   NA 0.13512499 0.1713878 0.2673203 0.4473745 0.6795436 0.7641056
## [10,]   NA   NA 0.41662444 0.6528608 0.6720454 0.7376002 0.8230296 0.8497633
##            [,9]     [,10]     [,11]     [,12]     [,13]     [,14]     [,15]
##  [1,] 0.8763064 0.8745613 0.9003915 0.9221413 0.9310493 0.9403371 0.9490036
##  [2,] 0.8482248 0.8743404 0.8886915 0.9155578 0.9518321 0.9649648 0.9694788
##  [3,] 0.8274601 0.9214266 0.9345165 0.9397551 0.9469293 0.9563559 0.9616411
```

```
##  [4,] 0.8821818 0.8987399 0.9112453 0.9193910 0.9313445 0.9406548 0.9486728
##  [5,] 0.8589064 0.8688101 0.8740344 0.8844424 0.8879074 0.9196396 0.9376273
##  [6,] 0.8288089 0.8527485 0.8966499 0.9036842 0.9114113 0.9155344 0.9373540
##  [7,] 0.8626579 0.8757108 0.8871334 0.9145820 0.9273358 0.9342071 0.9385756
##  [8,] 0.8918670 0.9013592 0.9225814 0.9433432 0.9557917 0.9663552 0.9718803
##  [9,] 0.7730955 0.7750793 0.7866916 0.7896447 0.8028147 0.8012668 0.8124653
## [10,] 0.8624582 0.9100098 0.9217635 0.9275385 0.9462556 0.9567803 0.9615444
##           [,16]     [,17]     [,18]     [,19] [,20]
##  [1,] 0.9745360 0.9865534 0.9910919 0.9951146     1
##  [2,] 0.9757659 0.9824092 0.9885669 0.9939606     1
##  [3,] 0.9661377 0.9836512 0.9885583 0.9939523     1
##  [4,] 0.9525787 0.9808720 0.9832653 0.9890343     1
##  [5,] 0.9498876 0.9594395 0.9647885 0.9928333     1
##  [6,] 0.9572932 0.9695654 0.9775017 0.9939523     1
##  [7,] 0.9454061 0.9512646 0.9837778 0.9939523     1
##  [8,] 0.9770720 0.9841295 0.9886862 0.9963338     1
##  [9,] 0.9666450 0.9772675 0.9892422 0.9950913     1
## [10,] 0.9677554 0.9820975 0.9876514 0.9939523     1
##
## $median.fit
##  [1]        NA        NA 0.2109888 0.4529717 0.6688803 0.6881862 0.7703478
##  [8] 0.8115285 0.8606823 0.8751360 0.8985207 0.9174744 0.9311969 0.9404959
## [15] 0.9488382 0.9663914 0.9814848 0.9881049 0.9939523 1.0000000
##
## $maxfit.landmark
## < table of extent 0 >
##
## $minfit.landmark
## < table of extent 0 >
##
## $fit.cs
##  [1]        NA        NA 0.9990298 0.9994840 0.9994428 0.9994783 0.9996251
##  [8] 0.9997835 0.9998566 0.9998498 0.9998973 0.9999261 0.9999295 0.9999589
## [15] 0.9999697 0.9999791 0.9999839 0.9999936 0.9999967 1.0000000
```

```
lasec(two.d.array(landmarks_tanged), 2, iter = 10, show.progress = F) ### may take some time
```

```
## $fit
##       [,1] [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]
##  [1,]   NA   NA 0.3637104 0.4970388 0.5437679 0.6313296 0.6665750 0.6941520
##  [2,]   NA   NA 0.4702792 0.7561133 0.8677004 0.8637114 0.8670633 0.9310894
##  [3,]   NA   NA 0.2949477 0.5681083 0.7685849 0.7926670 0.7898533 0.7949672
##  [4,]   NA   NA 0.5210015 0.7368207 0.8773437 0.8796057 0.8908005 0.9004354
##  [5,]   NA   NA 0.3858198 0.4722266 0.5711067 0.6838602 0.7448244 0.9003974
##  [6,]   NA   NA 0.4746495 0.6695535 0.7769632 0.9077250 0.9104285 0.9253586
##  [7,]   NA   NA 0.1195304 0.3216357 0.5695952 0.5737160 0.8653222 0.8863542
##  [8,]   NA   NA 0.3083874 0.4017041 0.7292814 0.7406238 0.7159066 0.7578050
##  [9,]   NA   NA 0.3605516 0.4532269 0.8254409 0.8966036 0.9085586 0.8945971
## [10,]   NA   NA 0.5357263 0.6840842 0.8430015 0.8498234 0.8805263 0.8943378
##            [,9]     [,10]     [,11]     [,12]     [,13]     [,14]     [,15]
##  [1,] 0.9220333 0.9255628 0.9404104 0.9657278 0.9662211 0.9672965 0.9742326
##  [2,] 0.9391179 0.9434230 0.9576137 0.9620358 0.9691823 0.9687496 0.9880111
##  [3,] 0.9511377 0.9443155 0.9482071 0.9637054 0.9679705 0.9681118 0.9660157
##  [4,] 0.8984567 0.9433475 0.9488213 0.9566490 0.9602328 0.9710564 0.9821300
##  [5,] 0.9117537 0.9157032 0.9127381 0.9204037 0.9687928 0.9808239 0.9825749
```

```
##  [6,] 0.9253685 0.9485701 0.9630692 0.9654016 0.9787967 0.9808377 0.9833579
##  [7,] 0.9536132 0.9672120 0.9687005 0.9697125 0.9756404 0.9800621 0.9825824
##  [8,] 0.7580566 0.8838261 0.8911148 0.8809086 0.9841480 0.9859973 0.9901149
##  [9,] 0.9095510 0.9196096 0.9232251 0.9646382 0.9663521 0.9720790 0.9702083
## [10,] 0.9027873 0.9064047 0.9074050 0.9417934 0.9442489 0.9445892 0.9447509
##           [,16]     [,17]     [,18]     [,19] [,20]
##  [1,] 0.9761445 0.9745929 0.9891150 0.9928224     1
##  [2,] 0.9896031 0.9910047 0.9939657 0.9976160     1
##  [3,] 0.9759641 0.9741435 0.9780492 0.9985716     1
##  [4,] 0.9880010 0.9935069 0.9961252 0.9984673     1
##  [5,] 0.9892960 0.9906517 0.9920892 0.9940233     1
##  [6,] 0.9838510 0.9850888 0.9936976 0.9983303     1
##  [7,] 0.9838759 0.9852637 0.9966887 0.9984673     1
##  [8,] 0.9922575 0.9925156 0.9972689 0.9990520     1
##  [9,] 0.9773046 0.9808418 0.9961766 0.9972152     1
## [10,] 0.9478968 0.9721460 0.9932081 0.9952842     1
##
## $median.fit
##  [1]        NA        NA 0.3747651 0.5325736 0.7727740 0.8212452 0.8661928
##  [8] 0.8944674 0.9168935 0.9344551 0.9443088 0.9628706 0.9683817 0.9715677
## [15] 0.9823524 0.9838634 0.9851763 0.9938316 0.9979732 1.0000000
##
## $maxfit.landmark
## < table of extent 0 >
##
## $minfit.landmark
## < table of extent 0 >
##
## $fit.cs
##  [1]        NA        NA 0.9992925 0.9996987 0.9997926 0.9998255 0.9998709
##  [8] 0.9999137 0.9999194 0.9999317 0.9999483 0.9999431 0.9999561 0.9999608
## [15] 0.9999665 0.9999642 0.9999812 0.9999915 0.9999978 1.0000000
```

```r
lasec(two.d.array(landmarks_handaxe), 2, iter = 10, show.progress = F) ### may take some time
```

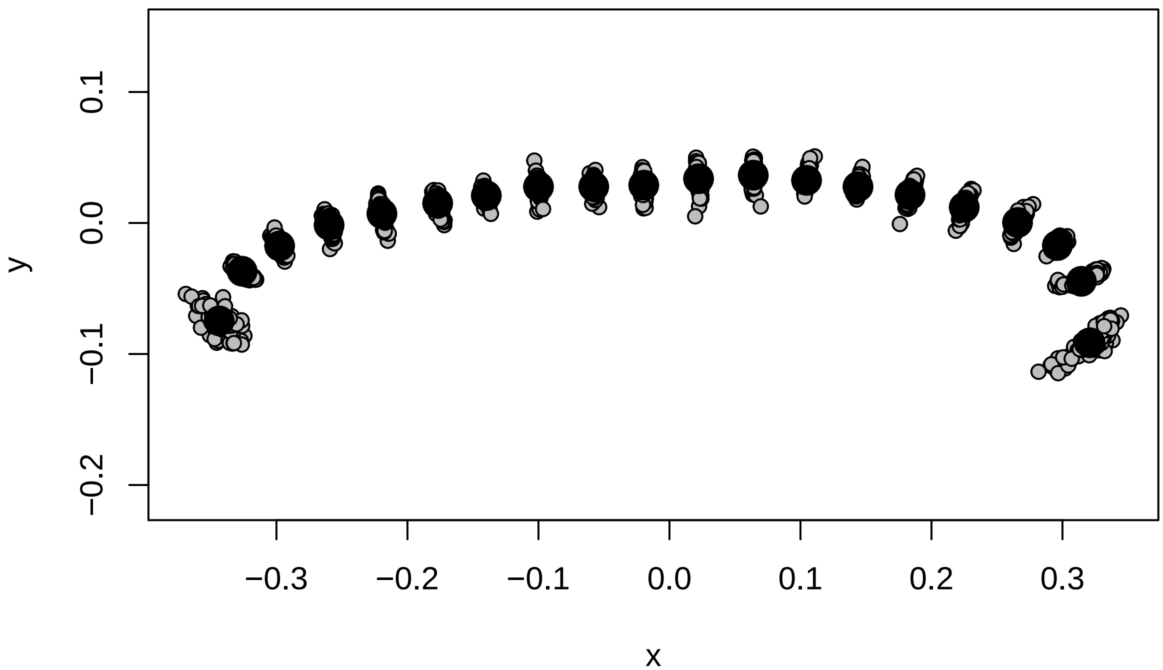
```
## $fit
##       [,1] [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]
##  [1,]   NA   NA 0.6855587 0.7103135 0.8025947 0.8509098 0.8573438 0.8689123
##  [2,]   NA   NA 0.2804016 0.8955476 0.9131987 0.9283516 0.9324729 0.9475356
##  [3,]   NA   NA 0.6258260 0.7414667 0.7410309 0.7382395 0.7155373 0.7004799
##  [4,]   NA   NA 0.7452771 0.8061092 0.7799164 0.7790021 0.9467000 0.9558506
##  [5,]   NA   NA 0.8797704 0.8899131 0.9148558 0.9219254 0.9347888 0.9474790
##  [6,]   NA   NA 0.8511077 0.8853058 0.8991234 0.9391811 0.9462415 0.9484214
##  [7,]   NA   NA 0.6772855 0.7809577 0.7668213 0.8819412 0.8759503 0.8559443
##  [8,]   NA   NA 0.2700441 0.7646333 0.7668143 0.8426508 0.8425407 0.9442350
##  [9,]   NA   NA 0.5435225 0.8424814 0.8477936 0.8448803 0.8435495 0.9111778
## [10,]   NA   NA 0.8242519 0.8852478 0.9024199 0.9312038 0.9341052 0.9432530
##            [,9]     [,10]     [,11]     [,12]     [,13]     [,14]     [,15]
##  [1,] 0.8839044 0.8994479 0.9253058 0.9623100 0.9763859 0.9816885 0.9818936
##  [2,] 0.9501617 0.9534634 0.9634309 0.9688812 0.9755900 0.9783093 0.9796748
##  [3,] 0.6906477 0.9723285 0.9793204 0.9845823 0.9856407 0.9910867 0.9920987
##  [4,] 0.9738044 0.9811761 0.9848224 0.9861031 0.9889872 0.9895479 0.9921942
##  [5,] 0.9568907 0.9707670 0.9728024 0.9740955 0.9773217 0.9806615 0.9819932
##  [6,] 0.9620719 0.9668667 0.9700201 0.9749305 0.9774886 0.9833139 0.9861461
##  [7,] 0.8559095 0.9487660 0.9682635 0.9713208 0.9755099 0.9761948 0.9899015
```

```
##  [8,] 0.9665686 0.9652620 0.9654490 0.9849790 0.9864889 0.9918007 0.9924210
##  [9,] 0.9095742 0.9351353 0.9323753 0.9339304 0.9715195 0.9761736 0.9809275
## [10,] 0.9616632 0.9726396 0.9796293 0.9823730 0.9854570 0.9861743 0.9873440
##           [,16]     [,17]     [,18]     [,19] [,20]
##  [1,] 0.9823700 0.9871082 0.9930286 0.9945479     1
##  [2,] 0.9881773 0.9942921 0.9971337 0.9984116     1
##  [3,] 0.9939808 0.9947631 0.9972426 0.9983804     1
##  [4,] 0.9936505 0.9963348 0.9983437 0.9992688     1
##  [5,] 0.9862378 0.9871127 0.9921415 0.9983021     1
##  [6,] 0.9870036 0.9900438 0.9957442 0.9986945     1
##  [7,] 0.9913917 0.9926340 0.9941598 0.9990658     1
##  [8,] 0.9944044 0.9953030 0.9970115 0.9985557     1
##  [9,] 0.9866891 0.9904447 0.9918592 0.9989569     1
## [10,] 0.9892900 0.9909397 0.9928622 0.9983804     1
##
## $median.fit
##  [1]        NA        NA 0.6814221 0.8242953 0.8251942 0.8664255 0.9042116
##  [8] 0.9437440 0.9535262 0.9660643 0.9691418 0.9745130 0.9774051 0.9825012
## [15] 0.9867451 0.9887337 0.9917869 0.9949520 0.9984836 1.0000000
##
## $maxfit.landmark
## < table of extent 0 >
##
## $minfit.landmark
## < table of extent 0 >
##
## $fit.cs
##  [1]        NA        NA 0.9987699 0.9987841 0.9991353 0.9994381 0.9994530
##  [8] 0.9995515 0.9996893 0.9997335 0.9997917 0.9998494 0.9998864 0.9998809
## [15] 0.9999144 0.9999383 0.9999785 0.9999666 0.9999938 1.0000000
```

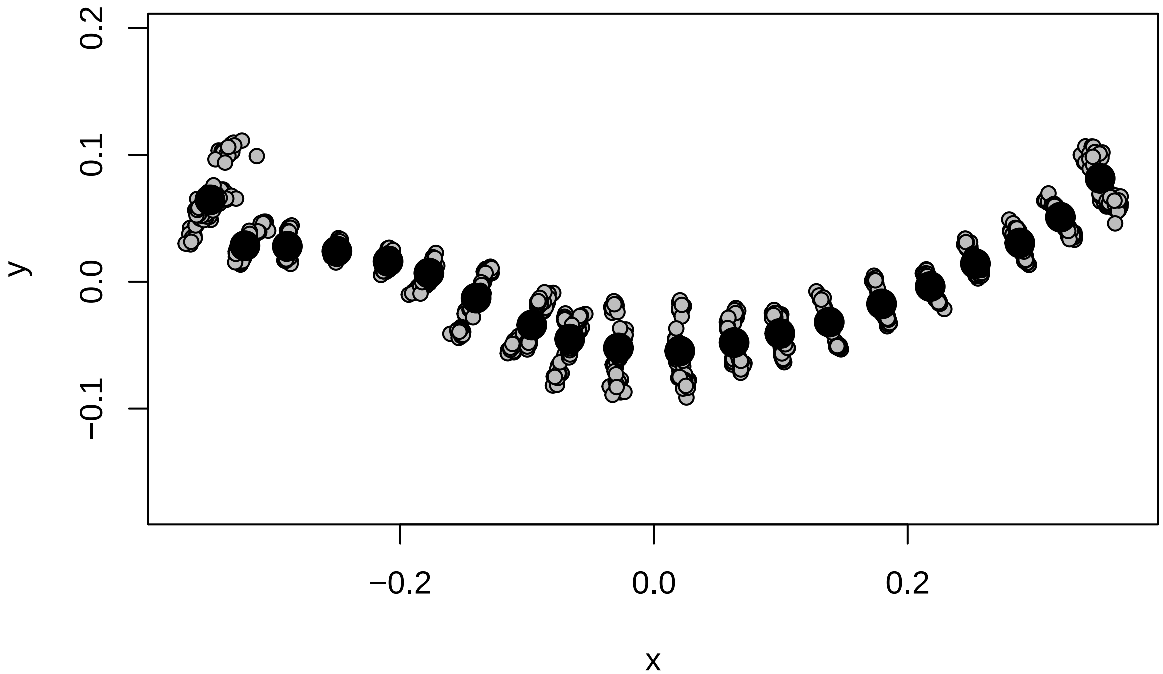
Stage 4: Generalised Procrustes Analysis (GPA)

With all errors calculated and deemed insignificant, we can now transform each artefact class through a GPA using the `geomorph::gpagen()` function as above:

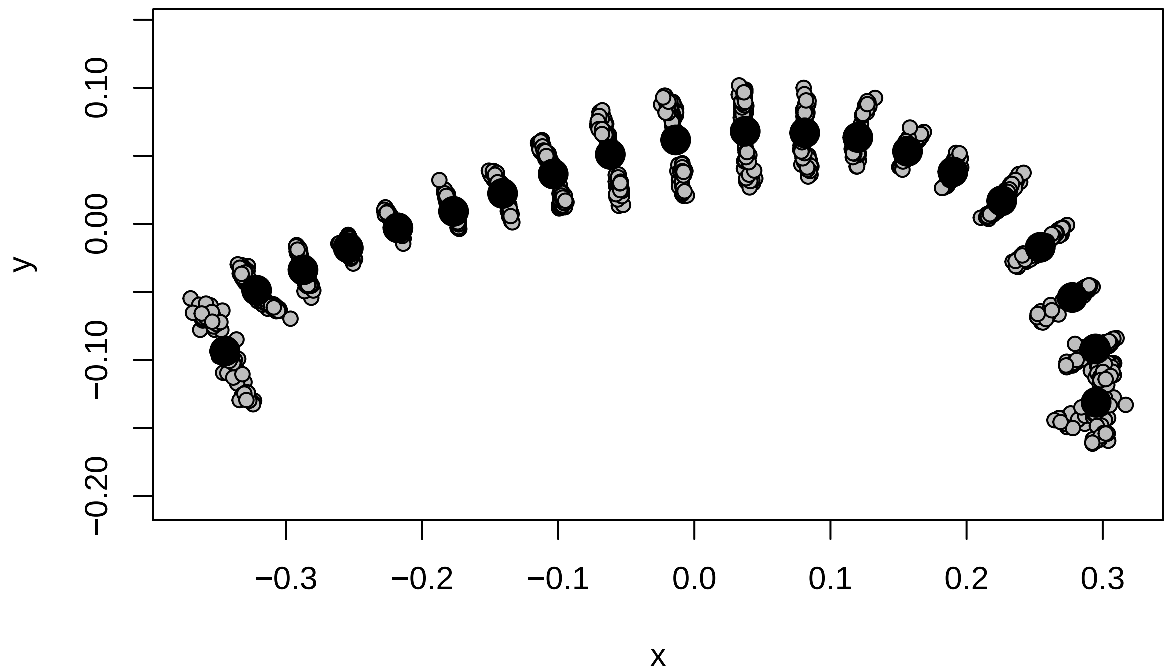
```r
gpa_elongated <- gpagen(landmarks_elongated, Proj = TRUE, ProcD = TRUE, curves = shape_data_sliders, su
plot(gpa_elongated) ### plots the procrustes coordinates for all elongated artefacts
```




```
gpa_tanged <- gpagen(landmarks_tanged, Proj = TRUE, ProcD = TRUE, curves = shape_data_sliders, surfaces
plot(gpa_tanged) ### plots the procrustes coordinates for all tanged artefacts
```




```
gpa_handaxe <- gpagen(landmarks_handaxe, Proj = TRUE, ProcD = TRUE, curves = shape_data_sliders, surfac
plot(gpa_handaxe) ### plots the procrustes coordinates for all handaxe artefacts
```



Stage 5: Exploratory Analysis through Principal Component Analysis (PCA)

Individual PCAs for all artefact classes were created to visualise differences in illustration skill between and within the different artefacts. PCAs were made in geomorph, through the `geomorph:gm.prcomp()` function, before being exported and modified (for better visualisation) in the tidyverse Note: specific shape differences can be calculated through the created files. The visualisations can then be exported through the `tidyverse:ggsave()` function.

```
pca_elongated <- gm.prcomp(gpa_elongated$coords) ### pca (geomorph)
pca_elongated ### pca summary
```

```
##
## Ordination type: Principal Component Analysis
## Centering and projection: OLS
## Number of observations 90
## Number of vectors 40
##
## Importance of Components:
##                               Comp1        Comp2        Comp3        Comp4
## Eigenvalues            0.0006240892 0.0003309824 0.0001584863 5.961311e-05
## Proportion of Variance 0.4581554039 0.2429803127 0.1163477493 4.376308e-02
## Cumulative Proportion  0.4581554039 0.7011357166 0.8174834660 8.612466e-01
##                               Comp5        Comp6        Comp7        Comp8
## Eigenvalues            3.529451e-05 3.177163e-05 0.0000203628 1.880991e-05
## Proportion of Variance 2.591035e-02 2.332414e-02 0.0149487086 1.380870e-02
## Cumulative Proportion  8.871569e-01 9.104810e-01 0.9254297512 9.392385e-01
##                               Comp9       Comp10       Comp11       Comp12
```

```
## Eigenvalues            1.340489e-05 1.198961e-05 1.008453e-05 8.642198e-06
## Proportion of Variance 9.840777e-03 8.801791e-03 7.403236e-03 6.344397e-03
## Cumulative Proportion  9.490792e-01 9.578810e-01 9.652843e-01 9.716287e-01
##                              Comp13       Comp14       Comp15       Comp16
## Eigenvalues            6.693848e-06 5.406835e-06 4.978051e-06 4.684685e-06
## Proportion of Variance 4.914078e-03 3.969257e-03 3.654479e-03 3.439114e-03
## Cumulative Proportion  9.765427e-01 9.805120e-01 9.841665e-01 9.876056e-01
##                              Comp17       Comp18       Comp19       Comp20
## Eigenvalues            3.897801e-06 3.361388e-06 2.190264e-06 1.677687e-06
## Proportion of Variance 2.861448e-03 2.467657e-03 1.607913e-03 1.231621e-03
## Cumulative Proportion  9.904670e-01 9.929347e-01 9.945426e-01 9.957742e-01
##                              Comp21       Comp22       Comp23       Comp24
## Eigenvalues            1.463887e-06 8.279955e-07 7.209098e-07 5.838376e-07
## Proportion of Variance 1.074666e-03 6.078468e-04 5.292332e-04 4.286060e-04
## Cumulative Proportion  9.968489e-01 9.974567e-01 9.979860e-01 9.984146e-01
##                              Comp25       Comp26       Comp27       Comp28
## Eigenvalues            3.725275e-07 3.456928e-07 2.633273e-07 2.420822e-07
## Proportion of Variance 2.734793e-04 2.537795e-04 1.933134e-04 1.777170e-04
## Cumulative Proportion  9.986881e-01 9.989418e-01 9.991351e-01 9.993129e-01
##                              Comp29       Comp30       Comp31       Comp32
## Eigenvalues            2.363967e-07 1.846166e-07 1.492681e-07 1.177400e-07
## Proportion of Variance 1.735432e-04 1.355305e-04 1.095805e-04 8.643509e-05
## Cumulative Proportion  9.994864e-01 9.996219e-01 9.997315e-01 9.998180e-01
##                              Comp33       Comp34       Comp35       Comp36
## Eigenvalues            8.787721e-08 7.107291e-08 4.752191e-08 2.573367e-08
## Proportion of Variance 6.451228e-05 5.217594e-05 3.488671e-05 1.889157e-05
## Cumulative Proportion  9.998825e-01 9.999346e-01 9.999695e-01 9.999884e-01
##                              Comp37       Comp38       Comp39       Comp40
## Eigenvalues            1.454692e-08 1.228972e-09 1.832541e-22 5.967109e-34
## Proportion of Variance 1.067916e-05 9.022114e-07 1.345302e-19 4.380565e-31
## Cumulative Proportion  9.999991e-01 1.000000e+00 1.000000e+00 1.000000e+00
```

```r
elongated_ds <- cbind(shape_data_elongated, pca_elongated$x) ### tidyverse compatible format

pca_tanged <- gm.prcomp(gpa_tanged$coords) ### pca (geomorph)
pca_tanged ### pca summary
```

```
##
## Ordination type: Principal Component Analysis
## Centering and projection: OLS
## Number of observations 90
## Number of vectors 40
##
## Importance of Components:
##                              Comp1       Comp2        Comp3       Comp4
## Eigenvalues            0.002108332 0.001407887 0.0002833669 0.000161115
## Proportion of Variance 0.508049684 0.339261820 0.0682835983 0.038824262
## Cumulative Proportion  0.508049684 0.847311504 0.9155951019 0.954419364
##                               Comp5        Comp6        Comp7        Comp8
## Eigenvalues            4.604155e-05 2.857256e-05 1.833576e-05 1.477163e-05
## Proportion of Variance 1.109474e-02 6.885197e-03 4.418412e-03 3.559556e-03
## Cumulative Proportion  9.655141e-01 9.723993e-01 9.768177e-01 9.803773e-01
##                               Comp9       Comp10       Comp11       Comp12
## Eigenvalues            1.262627e-05 9.103601e-06 8.270341e-06 7.226064e-06
## Proportion of Variance 3.042582e-03 2.193716e-03 1.992924e-03 1.741282e-03
```

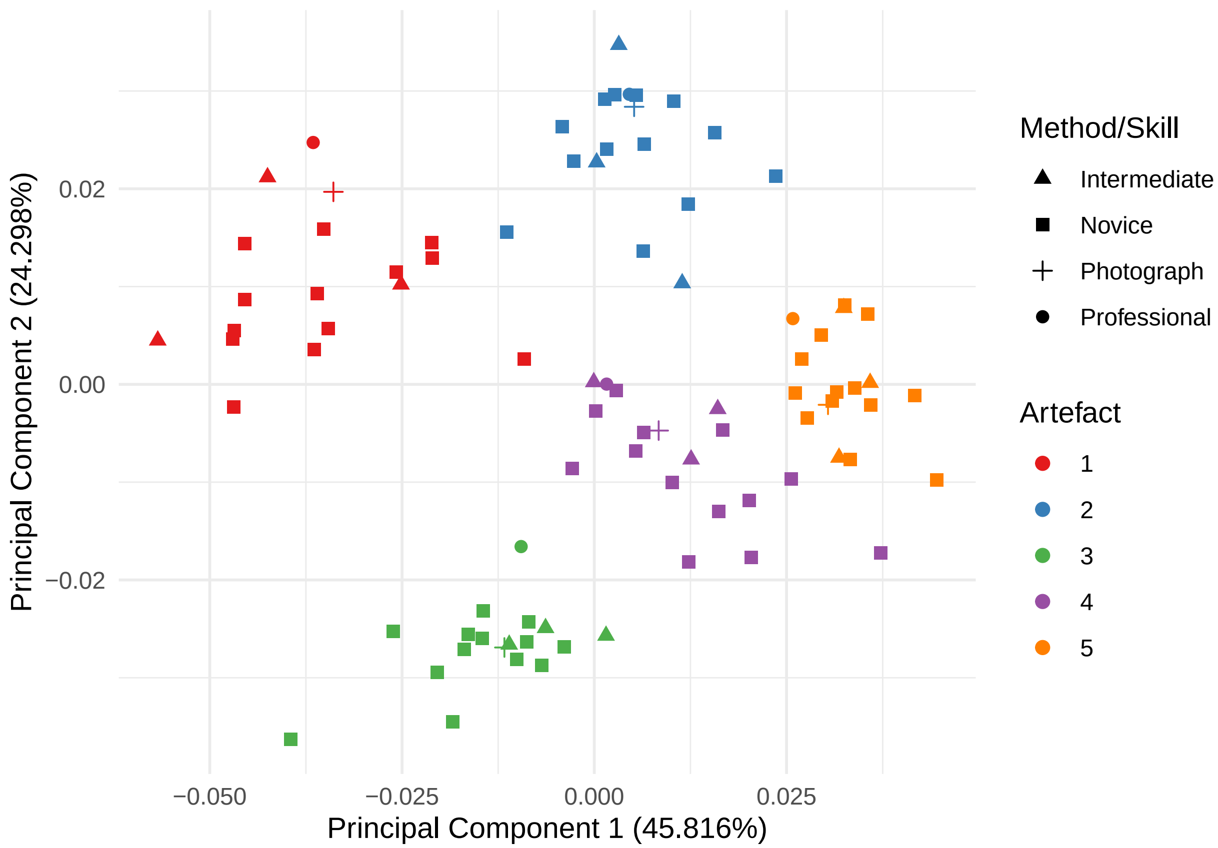
```
## Cumulative Proportion  9.834199e-01 9.856136e-01 9.876065e-01 9.893478e-01
##                              Comp13       Comp14       Comp15       Comp16
## Eigenvalues            6.433424e-06 0.0000059361 4.986095e-06 4.260570e-06
## Proportion of Variance 1.550278e-03 0.0014304362 1.201511e-03 1.026680e-03
## Cumulative Proportion  9.908981e-01 0.9923284887 9.935300e-01 9.945567e-01
##                              Comp17       Comp18       Comp19       Comp20
## Eigenvalues            3.425980e-06 2.990619e-06 2.843070e-06 2.461052e-06
## Proportion of Variance 8.255666e-04 7.206566e-04 6.851013e-04 5.930455e-04
## Cumulative Proportion  9.953822e-01 9.961029e-01 9.967880e-01 9.973810e-01
##                              Comp21       Comp22       Comp23       Comp24
## Eigenvalues            2.244288e-06 1.821357e-06 1.559110e-06 1.216749e-06
## Proportion of Variance 5.408114e-04 4.388967e-04 3.757024e-04 2.932029e-04
## Cumulative Proportion  9.979219e-01 9.983608e-01 9.987365e-01 9.990297e-01
##                              Comp25       Comp26       Comp27       Comp28
## Eigenvalues            7.597572e-07 5.893029e-07 5.709725e-07 4.840509e-07
## Proportion of Variance 1.830805e-04 1.420057e-04 1.375886e-04 1.166429e-04
## Cumulative Proportion  9.992127e-01 9.993547e-01 9.994923e-01 9.996090e-01
##                              Comp29       Comp30       Comp31       Comp32
## Eigenvalues            4.310409e-07 2.515842e-07 2.356631e-07 2.089888e-07
## Proportion of Variance 1.038689e-04 6.062485e-05 5.678829e-05 5.036052e-05
## Cumulative Proportion  9.997128e-01 9.997735e-01 9.998303e-01 9.998806e-01
##                              Comp33       Comp34       Comp35       Comp36
## Eigenvalues            1.713893e-07 1.193250e-07 1.144246e-07 5.925712e-08
## Proportion of Variance 4.130009e-05 2.875404e-05 2.757317e-05 1.427933e-05
## Cumulative Proportion  9.999219e-01 9.999507e-01 9.999783e-01 9.999925e-01
##                              Comp37       Comp38       Comp39      Comp40
## Eigenvalues            2.710922e-08 3.891138e-09 8.741952e-23 1.53472e-33
## Proportion of Variance 6.532573e-06 9.376569e-07 2.106569e-20 3.69825e-31
## Cumulative Proportion  9.999991e-01 1.000000e+00 1.000000e+00 1.00000e+00
```

```
tanged_ds <- cbind(shape_data_tanged, pca_tanged$x) ### tidyverse compatible format

pca_handaxe <- gm.prcomp(gpa_handaxe$coords) ### pca (geomorph)
pca_handaxe ### pca summary
```

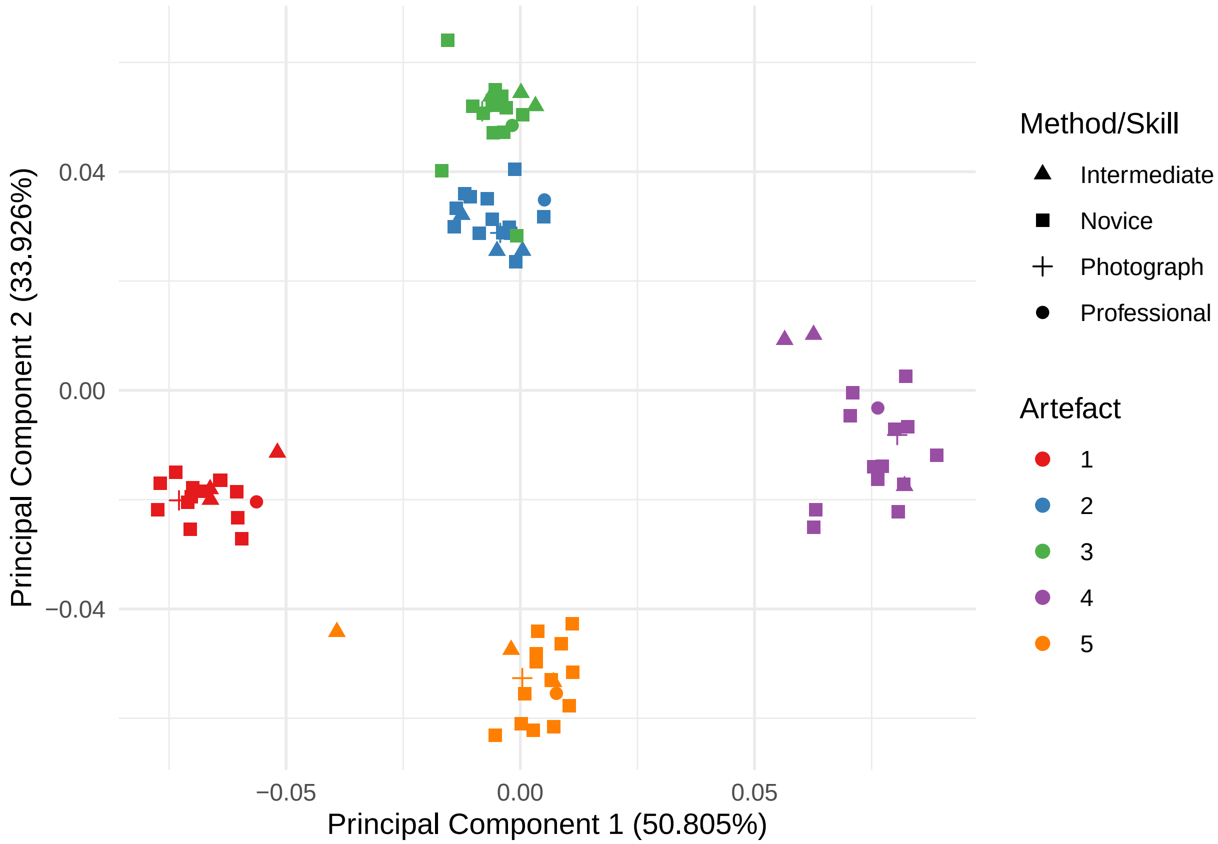
```
##
## Ordination type: Principal Component Analysis
## Centering and projection: OLS
## Number of observations 90
## Number of vectors 40
##
## Importance of Components:
##                              Comp1        Comp2        Comp3        Comp4
## Eigenvalues            0.004147443 0.0003415703 0.0001480715 5.776983e-05
## Proportion of Variance 0.848362755 0.0698684628 0.0302881402 1.181686e-02
## Cumulative Proportion  0.848362755 0.9182312181 0.9485193583 9.603362e-01
##                               Comp5        Comp6        Comp7        Comp8
## Eigenvalues            3.702773e-05 2.686763e-05 1.884701e-05 1.644421e-05
## Proportion of Variance 7.574051e-03 5.495794e-03 3.855170e-03 3.363675e-03
## Cumulative Proportion  9.679103e-01 9.734061e-01 9.772612e-01 9.806249e-01
##                               Comp9       Comp10       Comp11       Comp12
## Eigenvalues            1.401099e-05 1.294287e-05 1.019683e-05 8.733280e-06
## Proportion of Variance 2.865959e-03 2.647474e-03 2.085770e-03 1.786399e-03
## Cumulative Proportion  9.834909e-01 9.861383e-01 9.882241e-01 9.900105e-01
##                              Comp13       Comp14       Comp15       Comp16
```

```
## Eigenvalues            7.370640e-06 6.467041e-06 4.826731e-06 4.408248e-06
## Proportion of Variance 1.507670e-03 1.322838e-03 9.873115e-04 9.017105e-04
## Cumulative Proportion  9.915182e-01 9.928410e-01 9.938283e-01 9.947300e-01
##                              Comp17       Comp18       Comp19       Comp20
## Eigenvalues            3.801114e-06 3.306690e-06 3.009149e-06 2.486183e-06
## Proportion of Variance 7.775208e-04 6.763861e-04 6.155238e-04 5.085508e-04
## Cumulative Proportion  9.955076e-01 9.961840e-01 9.967995e-01 9.973080e-01
##                              Comp21       Comp22       Comp23       Comp24
## Eigenvalues            2.276660e-06 1.958038e-06 1.744844e-06 1.488234e-06
## Proportion of Variance 4.656925e-04 4.005181e-04 3.569091e-04 3.044195e-04
## Cumulative Proportion  9.977737e-01 9.981742e-01 9.985311e-01 9.988356e-01
##                              Comp25       Comp26       Comp27       Comp28
## Eigenvalues            1.367614e-06 8.485558e-07 8.095846e-07 7.015244e-07
## Proportion of Variance 2.797465e-04 1.735727e-04 1.656012e-04 1.434974e-04
## Cumulative Proportion  9.991153e-01 9.992889e-01 9.994545e-01 9.995980e-01
##                              Comp29       Comp30       Comp31       Comp32
## Eigenvalues            5.635145e-07 3.907648e-07 2.811766e-07 2.106388e-07
## Proportion of Variance 1.152673e-04 7.993123e-05 5.751488e-05 4.308632e-05
## Cumulative Proportion  9.997133e-01 9.997932e-01 9.998507e-01 9.998938e-01
##                              Comp33       Comp34       Comp35       Comp36
## Eigenvalues            1.589855e-07 1.475755e-07 1.146792e-07 7.430645e-08
## Proportion of Variance 3.252060e-05 3.018668e-05 2.345772e-05 1.519944e-05
## Cumulative Proportion  9.999263e-01 9.999565e-01 9.999799e-01 9.999951e-01
##                              Comp37       Comp38       Comp39       Comp40
## Eigenvalues            1.945077e-08 4.274790e-09 2.907959e-21 9.892121e-34
## Proportion of Variance 3.978670e-06 8.744115e-07 5.948252e-19 2.023441e-31
## Cumulative Proportion  9.999991e-01 1.000000e+00 1.000000e+00 1.000000e+00
```

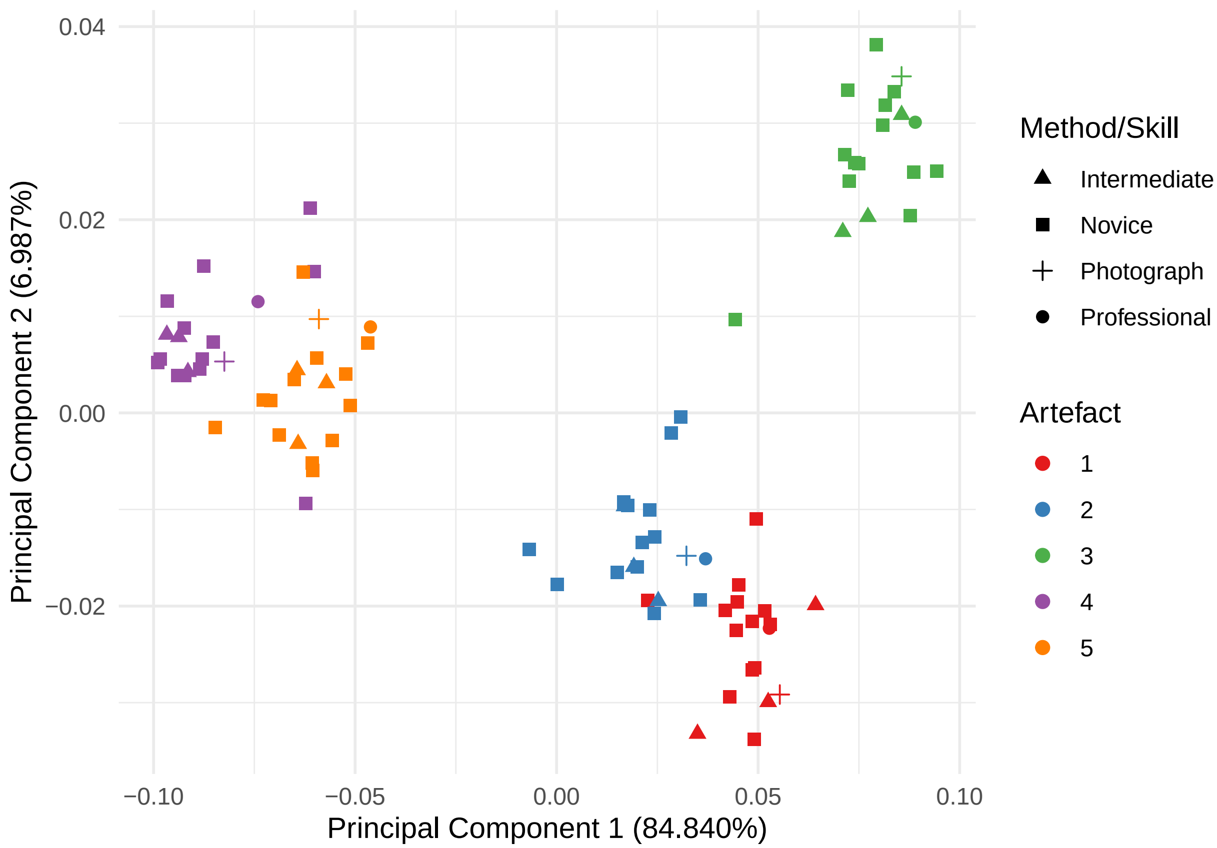
```
handaxe_ds <- cbind(shape_data_handaxe, pca_handaxe$x) ### tidyverse compatible format

ggplot(elongated_ds) +
  geom_point(aes(x = Comp1, y = Comp2, colour = Artefact, shape = Class), size = 2) +
  labs(x = "Principal Component 1 (45.816%)", y = "Principal Component 2 (24.298%)", shape = "Method/Ski
  scale_color_manual(values=c("#e41a1c","#377eb8","#4daf4a","#984ea3","#ff7f00")) +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() ### figure 5 creation
```




```r
ggplot(tanged_ds) +
  geom_point(aes(x = Comp1, y = Comp2, colour = Artefact, shape = Class), size = 2) +
  labs(x = "Principal Component 1 (50.805%)", y = "Principal Component 2 (33.926%)", shape = "Method/Sk
  scale_color_manual(values=c("#e41a1c","#377eb8","#4daf4a","#984ea3","#ff7f00")) +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() ### figure 6 creation
```




```
ggplot(handaxe_ds) +
  geom_point(aes(x = Comp1, y = Comp2, colour = Artefact, shape = Class), size = 2) +
  labs(x = "Principal Component 1 (84.840%)", y = "Principal Component 2 (6.987%)", shape = "Method/Skil
  scale_color_manual(values=c("#e41a1c","#377eb8","#4daf4a","#984ea3","#ff7f00")) +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() ### figure 7 creation
```



Stage 6: Further analysis (MANOVA and Discriminant Analysis)

Each individual artefact (and the collections of drawings for each artefact) was then analysed to examine whether differences in illustrator skill were observed. This was first done through a Procrustes ANOVA, providing a statistical framework for detecting any such difference. Discriminant analyses were then conducted for each artefact to test for between-group variance. The code for each artefact follows this order:

1) The Procrustes coordinates from an artefact are extracted from the dataframe.
2) A geomorph-specific data-frame is created through the `geomorph::geomorph.data.frame()` function, with the Procrustes coordinates and the class factor.
3) A Procrustes ANOVA is then performed through the `geomorph::procD.lm()` function, with the results summarised through the `base::summary()` argument.
4) A new data frame is generated to produce the discriminant analysis (through the `MASS::lda()` `function`).
5) The subsequent data is visualised through the `tidyverse` package.
6) This is repeated for each artefact within each artefact class.

```
elongated1 <- gpa_elongated$coords[, , 1:18]
df_elongated1 <- geomorph.data.frame(shape = elongated1, class = shape_data_elongated$Class[1:18], arte
E1 <- procD.lm(shape ~ class, data = df_elongated1, print.progress = FALSE)
summary(E1)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F       Z Pr(>F)
## class      3 0.0016387 0.00054623 0.17824 1.0122 0.18333  0.398
## Residuals 14 0.0075548 0.00053963 0.82176
## Total     17 0.0091935
##
## Call: procD.lm(f1 = shape ~ class, data = df_elongated1, print.progress = FALSE)
```

```
elongated1pcs <- as.data.frame(pca_elongated$x[1:18, 1:10])
elongated1class <- shape_data_elongated$Class[1:18]
elongated1pcs <- cbind(elongated1pcs, elongated1class)
elongated1pcs <- rename(elongated1pcs, Class = elongated1class)
elongated1lda <- lda(Class ~ ., data = elongated1pcs)
elongated1ldapredict <- predict(elongated1lda)
elongated1ldaplot <- cbind(elongated1ldapredict$x[, 1:3], elongated1pcs)
elongated1ldaplotggplot <- ggplot(elongated1ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        axis.title.x = element_text(size = 6),
        axis.title.y = element_text(size = 6),
        legend.position = "none")

elongated2 <- gpa_elongated$coords[, , 19:36]
df_elongated2 <- geomorph.data.frame(shape = elongated2, class = shape_data_elongated$Class[19:36], art
E2 <- procD.lm(shape ~ class, data = df_elongated2, print.progress = FALSE)
summary(E2)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F       Z Pr(>F)
## class      3 0.0007434 0.00024780 0.08883 0.4549 -1.8498  0.976
## Residuals 14 0.0076257 0.00054469 0.91117
## Total     17 0.0083691
##
## Call: procD.lm(f1 = shape ~ class, data = df_elongated2, print.progress = FALSE)
```

```
elongated2pcs <- as.data.frame(pca_elongated$x[19:36, 1:10])
elongated2class <- shape_data_elongated$Class[19:36]
```

```
elongated2pcs <- cbind(elongated2pcs, elongated2class)
elongated2pcs <- rename(elongated2pcs, Class = elongated2class)
elongated2lda <- lda(Class ~ ., elongated2pcs)
elongated2ldapredict <- predict(elongated2lda)
elongated2ldaplot <- cbind(elongated2ldapredict$x[, 1:3], elongated2pcs)
elongated2ldaplotggplot <- ggplot(elongated2ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

elongated3 <- gpa_elongated$coords[, , 37:54]
df_elongated3 <- geomorph.data.frame(shape = elongated3, class = shape_data_elongated$Class[37:54], art
E3 <- procD.lm(shape ~ class, data = df_elongated3, print.progress = FALSE)
summary(E3)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F        Z Pr(>F)
## class      3 0.0007131 0.00023771 0.14157 0.7696 -0.28451   0.57
## Residuals 14 0.0043243 0.00030888 0.85843
## Total     17 0.0050375
##
## Call: procD.lm(f1 = shape ~ class, data = df_elongated3, print.progress = FALSE)
```

```
elongated3pcs <- as.data.frame(pca_elongated$x[37:54, 1:10])
elongated3class <- shape_data_elongated$Class[37:54]
elongated3pcs <- cbind(elongated3pcs, elongated3class)
elongated3pcs <- rename(elongated3pcs, Class = elongated3class)
elongated3lda <- lda(Class ~ ., elongated3pcs)
elongated3ldapredict <- predict(elongated3lda)
elongated3ldaplot <- cbind(elongated3ldapredict$x[, 1:3], elongated3pcs)
elongated3ldaplotggplot <- ggplot(elongated3ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))
```

```r
elongated4 <- gpa_elongated$coords[, , 55:72]
df_elongated4 <- geomorph.data.frame(shape = elongated4, class = shape_data_elongated$Class[55:72], art
E4 <- procD.lm(shape ~ class, data = df_elongated4, print.progress = FALSE)
summary(E4)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F       Z Pr(>F)
## class      3 0.0007477 0.00024923 0.10896 0.5707 -0.7742  0.793
## Residuals 14 0.0061144 0.00043675 0.89104
## Total     17 0.0068621
##
## Call: procD.lm(f1 = shape ~ class, data = df_elongated4, print.progress = FALSE)
```

```r
elongated4pcs <- as.data.frame(pca_elongated$x[55:72, 1:10])
elongated4class <- shape_data_elongated$Class[55:72]
elongated4pcs <- cbind(elongated4pcs, elongated4class)
elongated4pcs <- rename(elongated4pcs, Class = elongated4class)
elongated4lda <- lda(Class ~ ., elongated4pcs)
elongated4ldapredict <- predict(elongated4lda)
elongated4ldaplot <- cbind(elongated4ldapredict$x[, 1:3], elongated4pcs)
elongated4ldaplotggplot <- ggplot(elongated4ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

elongated5 <- gpa_elongated$coords[, , 73:90]
df_elongated5 <- geomorph.data.frame(shape = elongated5, class = shape_data_elongated$Class[73:90], art
E5 <- procD.lm(shape ~ class, data = df_elongated5, print.progress = FALSE)
summary(E5)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS    Rsq      F       Z Pr(>F)
## class      3 0.0005381 0.00017936 0.0878 0.4492 -1.2749  0.905
## Residuals 14 0.0055901 0.00039929 0.9122
```

```
## Total     17 0.0061282
##
## Call: procD.lm(f1 = shape ~ class, data = df_elongated5, print.progress = FALSE)
```

```
elongated5pcs <- as.data.frame(pca_elongated$x[73:90, 1:10])
elongated5class <- shape_data_elongated$Class[73:90]
elongated5pcs <- cbind(elongated5pcs, elongated5class)
elongated5pcs <- rename(elongated5pcs, Class = elongated5class)
elongated5lda <- lda(Class ~ ., elongated5pcs)
elongated5ldapredict <- predict(elongated5lda)
elongated5ldaplot <- cbind(elongated5ldapredict$x[, 1:3], elongated5pcs)
elongated5ldaplotggplot <- ggplot(elongated5ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

tanged1 <- gpa_tanged$coords[, , 1:18]
df_tanged1 <- geomorph.data.frame(shape = tanged1, class = shape_data_tanged$Class[1:18], artefact = sha
T1 <- procD.lm(shape ~ class, data = df_tanged1, print.progress = FALSE)
summary(T1)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS    Rsq      F        Z Pr(>F)
## class      3 0.0006374 0.00021247 0.1445 0.7882 -0.41375  0.659
## Residuals 14 0.0037737 0.00026955 0.8555
## Total     17 0.0044111
##
## Call: procD.lm(f1 = shape ~ class, data = df_tanged1, print.progress = FALSE)
```

```
tanged1pcs <- as.data.frame(pca_tanged$x[1:18, 1:10])
tanged1class <- shape_data_tanged$Class[1:18]
tanged1pcs <- cbind(tanged1pcs, tanged1class)
tanged1pcs <- rename(tanged1pcs, Class = tanged1class)
tanged1lda <- lda(Class ~ ., tanged1pcs)
tanged1ldapredict <- predict(tanged1lda)
tanged1ldaplot <- cbind(tanged1ldapredict$x[, 1:3], tanged1pcs)
tanged1ldaplotggplot <- ggplot(tanged1ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
```

```
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

tanged2 <- gpa_tanged$coords[, , 19:36]
df_tanged2 <- geomorph.data.frame(shape = tanged2, class = shape_data_tanged$Class[19:36], artefact = sl
T2 <- procD.lm(shape ~ class, data = df_tanged2, print.progress = FALSE)
summary(T2)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F       Z Pr(>F)
## class      3 0.0005912 0.00019707 0.17446 0.9862 0.12861  0.418
## Residuals 14 0.0027976 0.00019983 0.82554
## Total     17 0.0033888
##
## Call: procD.lm(f1 = shape ~ class, data = df_tanged2, print.progress = FALSE)
```

```
tanged2pcs <- as.data.frame(pca_tanged$x[19:36, 1:10])
tanged2class <- shape_data_tanged$Class[19:36]
tanged2pcs <- cbind(tanged2pcs, tanged2class)
tanged2pcs <- rename(tanged2pcs, Class = tanged2class)
tanged2lda <- lda(Class ~ ., tanged2pcs)
tanged2ldapredict <- predict(tanged2lda)
tanged2ldaplot <- cbind(tanged2ldapredict$x[, 1:3], tanged2pcs)
tanged2ldaplotggplot <- ggplot(tanged2ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

tanged3 <- gpa_tanged$coords[, , 37:54]
df_tanged3 <- geomorph.data.frame(shape = tanged3, class = shape_data_tanged$Class[37:54], artefact = sl
T3 <- procD.lm(shape ~ class, data = df_tanged3, print.progress = FALSE)
summary(T3)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
```

```
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F        Z Pr(>F)
## class      3 0.0004659 0.00015530 0.13042 0.6999 -0.47251  0.649
## Residuals 14 0.0031064 0.00022189 0.86958
## Total     17 0.0035724
##
## Call: procD.lm(f1 = shape ~ class, data = df_tanged3, print.progress = FALSE)
```

```
tanged3pcs <- as.data.frame(pca_tanged$x[37:54, 1:10])
tanged3class <- shape_data_tanged$Class[37:54]
tanged3pcs <- cbind(tanged3pcs, tanged3class)
tanged3pcs <- rename(tanged3pcs, Class = tanged3class)
tanged3lda <- lda(Class ~ ., tanged3pcs)
tanged3ldapredict <- predict(tanged3lda)
tanged3ldaplot <- cbind(tanged3ldapredict$x[, 1:3], tanged3pcs)
tanged3ldaplotggplot <- ggplot(tanged3ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

tanged4 <- gpa_tanged$coords[, , 55:72]
df_tanged4 <- geomorph.data.frame(shape = tanged4, class = shape_data_tanged$Class[55:72], artefact = sh
T4 <- procD.lm(shape ~ class, data = df_tanged4, print.progress = FALSE)
summary(T4)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F       Z Pr(>F)
## class      3 0.0020486 0.00068286 0.21029 1.2427 0.64326  0.242
## Residuals 14 0.0076931 0.00054951 0.78971
## Total     17 0.0097417
##
## Call: procD.lm(f1 = shape ~ class, data = df_tanged4, print.progress = FALSE)
```

```
tanged4pcs <- as.data.frame(pca_tanged$x[55:72, 1:10])
tanged4class <- shape_data_tanged$Class[55:72]
tanged4pcs <- cbind(tanged4pcs, tanged4class)
tanged4pcs <- rename(tanged4pcs, Class = tanged4class)
tanged4lda <- lda(Class ~ ., tanged4pcs)
tanged4ldapredict <- predict(tanged4lda)
tanged4ldaplot <- cbind(tanged4ldapredict$x[, 1:3], tanged4pcs)
tanged4ldaplotggplot <- ggplot(tanged4ldaplot, aes(LD1, LD2)) +
```

```
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

tanged5 <- gpa_tanged$coords[, , 73:90]
df_tanged5 <- geomorph.data.frame(shape = tanged5, class = shape_data_tanged$Class[73:90], artefact = sl
T5 <- procD.lm(shape ~ class, data = df_tanged5, print.progress = FALSE)
summary(T5)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F       Z Pr(>F)
## class      3 0.0014793 0.00049310 0.19693 1.1444 0.44278  0.237
## Residuals 14 0.0060323 0.00043088 0.80307
## Total     17 0.0075116
##
## Call: procD.lm(f1 = shape ~ class, data = df_tanged5, print.progress = FALSE)
```

```
tanged5pcs <- as.data.frame(pca_tanged$x[73:90, 1:10])
tanged5class <- shape_data_tanged$Class[73:90]
tanged5pcs <- cbind(tanged5pcs, tanged5class)
tanged5pcs <- rename(tanged5pcs, Class = tanged5class)
tanged5lda <- lda(Class ~ ., tanged5pcs)
tanged5ldapredict <- predict(tanged5lda)
tanged5ldaplot <- cbind(tanged5ldapredict$x[, 1:3], tanged5pcs)
tanged5ldaplotggplot <- ggplot(tanged5ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

handaxe1 <- gpa_handaxe$coords[, , 1:18]
df_handaxe1 <- geomorph.data.frame(shape = handaxe1, class = shape_data_handaxe$Class[1:18], artefact =
H1 <- procD.lm(shape ~ class, data = df_handaxe1, print.progress = FALSE)
summary(H1)
```

```
##
```

```
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F        Z Pr(>F)
## class      3 0.0012319 0.00041064 0.14025 0.7613 -0.42438  0.604
## Residuals 14 0.0075520 0.00053943 0.85975
## Total     17 0.0087839
##
## Call: procD.lm(f1 = shape ~ class, data = df_handaxe1, print.progress = FALSE)
```

```
handaxe1pcs <- as.data.frame(pca_handaxe$x[1:18, 1:10])
handaxe1class <- shape_data_handaxe$Class[1:18]
handaxe1pcs <- cbind(handaxe1pcs, handaxe1class)
handaxe1pcs <- rename(handaxe1pcs, Class = handaxe1class)
handaxe1lda <- lda(Class ~ ., handaxe1pcs)
handaxe1ldapredict <- predict(handaxe1lda)
handaxe1ldaplot <- cbind(handaxe1ldapredict$x[, 1:3], handaxe1pcs)
handaxe1ldaplotggplot <- ggplot(handaxe1ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

handaxe2 <- gpa_handaxe$coords[, , 19:36]
df_handaxe2 <- geomorph.data.frame(shape = handaxe2, class = shape_data_handaxe$Class[19:36], artefact =
H2 <- procD.lm(shape ~ class, data = df_handaxe2, print.progress = FALSE)
summary(H2)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F       Z Pr(>F)
## class      3 0.0007207 0.00024023 0.11133 0.5846 -1.0482  0.843
## Residuals 14 0.0057525 0.00041090 0.88867
## Total     17 0.0064732
##
## Call: procD.lm(f1 = shape ~ class, data = df_handaxe2, print.progress = FALSE)
```

```
handaxe2pcs <- as.data.frame(pca_handaxe$x[19:36, 1:10])
handaxe2class <- shape_data_handaxe$Class[19:36]
handaxe2pcs <- cbind(handaxe2pcs, handaxe2class)
```

```
handaxe2pcs <- rename(handaxe2pcs, Class = handaxe2class)
handaxe2lda <- lda(Class ~ ., handaxe2pcs)
handaxe2ldapredict <- predict(handaxe2lda)
handaxe2ldaplot <- cbind(handaxe2ldapredict$x[, 1:3], handaxe2pcs)
handaxe2ldaplotggplot <- ggplot(handaxe2ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

handaxe3 <- gpa_handaxe$coords[, , 37:54]
df_handaxe3 <- geomorph.data.frame(shape = handaxe3, class = shape_data_handaxe$Class[37:54], artefact =
H3 <- procD.lm(shape ~ class, data = df_handaxe3, print.progress = FALSE)
summary(H3)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq     F        Z Pr(>F)
## class      3 0.0010882 0.00036274 0.11811 0.625 -0.64478   0.73
## Residuals 14 0.0081253 0.00058038 0.88189
## Total     17 0.0092135
##
## Call: procD.lm(f1 = shape ~ class, data = df_handaxe3, print.progress = FALSE)
```

```
handaxe3pcs <- as.data.frame(pca_handaxe$x[37:54, 1:10])
handaxe3class <- shape_data_handaxe$Class[37:54]
handaxe3pcs <- cbind(handaxe3pcs, handaxe3class)
handaxe3pcs <- rename(handaxe3pcs, Class = handaxe3class)
handaxe3lda <- lda(Class ~ ., handaxe3pcs)
handaxe3ldapredict <- predict(handaxe3lda)
handaxe3ldaplot <- cbind(handaxe3ldapredict$x[, 1:3], handaxe3pcs)
handaxe3ldaplotggplot <- ggplot(handaxe3ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

handaxe4 <- gpa_handaxe$coords[, , 55:72]
```

```r
df_handaxe4 <- geomorph.data.frame(shape = handaxe4, class = shape_data_handaxe$Class[55:72], artefact =
H4 <- procD.lm(shape ~ class, data = df_handaxe4, print.progress = FALSE)
summary(H4)
```

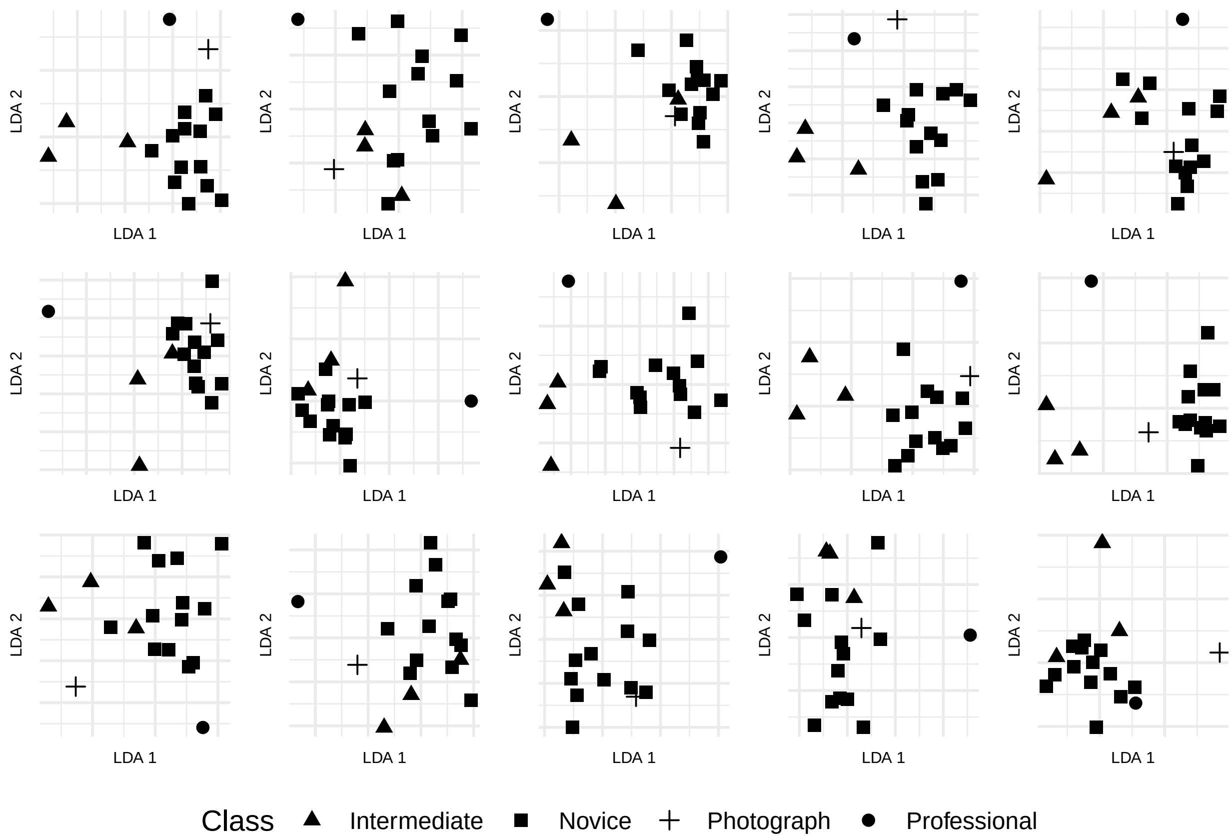
```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq      F       Z Pr(>F)
## class      3 0.0006189 0.00020630 0.08025 0.4072 -1.4387  0.938
## Residuals 14 0.0070934 0.00050667 0.91975
## Total     17 0.0077123
##
## Call: procD.lm(f1 = shape ~ class, data = df_handaxe4, print.progress = FALSE)
```

```r
handaxe4pcs <- as.data.frame(pca_handaxe$x[55:72, 1:10])
handaxe4class <- shape_data_handaxe$Class[55:72]
handaxe4pcs <- cbind(handaxe4pcs, handaxe4class)
handaxe4pcs <- rename(handaxe4pcs, Class = handaxe4class)
handaxe4lda <- lda(Class ~ ., handaxe4pcs)
handaxe4ldapredict <- predict(handaxe4lda)
handaxe4ldaplot <- cbind(handaxe4ldapredict$x[, 1:3], handaxe4pcs)
handaxe4ldaplotggplot <- ggplot(handaxe4ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

handaxe5 <- gpa_handaxe$coords[, , 73:90]
df_handaxe5 <- geomorph.data.frame(shape = handaxe5, class = shape_data_handaxe$Class[73:90], artefact =
H5 <- procD.lm(shape ~ class, data = df_handaxe5, print.progress = FALSE)
summary(H5)
```

```
##
## Analysis of Variance, using Residual Randomization
## Permutation procedure: Randomization of null model residuals
## Number of permutations: 1000
## Estimation method: Ordinary Least Squares
## Sums of Squares and Cross-products: Type I
## Effect sizes (Z) based on F distributions
##
##           Df        SS         MS     Rsq     F        Z Pr(>F)
## class      3 0.0007086 0.00023620 0.11527 0.608 -0.93807  0.822
## Residuals 14 0.0054387 0.00038848 0.88473
## Total     17 0.0061473
```

```
##
## Call: procD.lm(f1 = shape ~ class, data = df_handaxe5, print.progress = FALSE)
```

```
handaxe5pcs <- as.data.frame(pca_handaxe$x[73:90, 1:10])
handaxe5class <- shape_data_handaxe$Class[73:90]
handaxe5pcs <- cbind(handaxe5pcs, handaxe5class)
handaxe5pcs <- rename(handaxe5pcs, Class = handaxe5class)
handaxe5lda <- lda(Class ~ ., handaxe5pcs)
handaxe5ldapredict <- predict(handaxe5lda)
handaxe5ldaplot <- cbind(handaxe5ldapredict$x[, 1:3], handaxe5pcs)
handaxe5ldaplotggplot <- ggplot(handaxe5ldaplot, aes(LD1, LD2)) +
  geom_point(aes(shape = Class), size = 2) +
  labs(x = "LDA 1", y = "LDA 2") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none",
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 6))

lda.figure <-  plot_grid(elongated1ldaplotggplot,
                         elongated2ldaplotggplot,
                         elongated3ldaplotggplot,
                         elongated4ldaplotggplot,
                         elongated5ldaplotggplot,
                         tanged1ldaplotggplot,
                         tanged2ldaplotggplot,
                         tanged3ldaplotggplot,
                         tanged4ldaplotggplot,
                         tanged5ldaplotggplot,
                         handaxe1ldaplotggplot,
                         handaxe2ldaplotggplot,
                         handaxe3ldaplotggplot,
                         handaxe4ldaplotggplot,
                         handaxe5ldaplotggplot,
                         ncol = 5, align = 'v')

lda.figure.legend <- get_legend(elongated1ldaplotggplot +
                                  guides(color = guide_legend(nrow = 1)) +
                                  theme(legend.position = "bottom"))

plot_grid(lda.figure, lda.figure.legend, ncol = 1, rel_heights = c(1, .1))
```

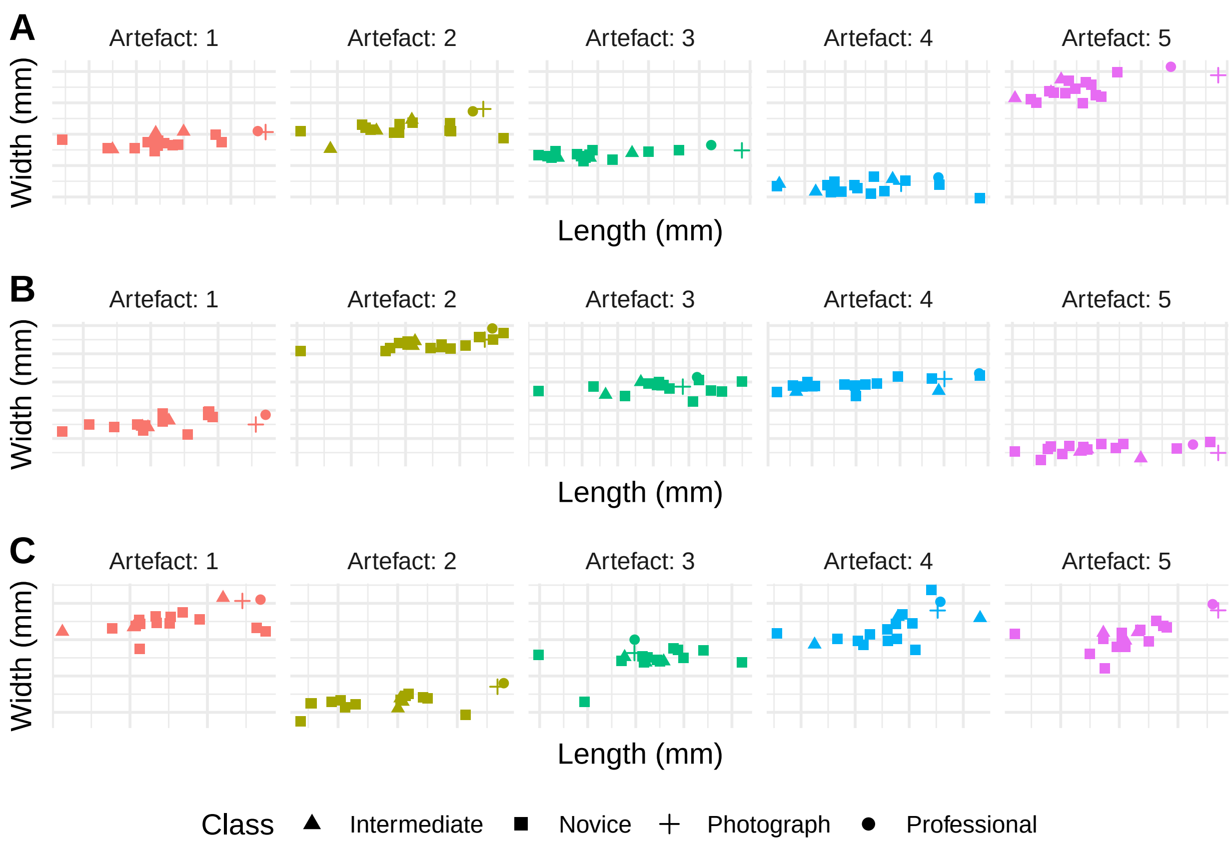


Stage 7: Measurement Data (Exploratory Framework)

To first examine the metric data, bivariate plots for all three artefact groups were generated in ggplot:

```
metric_data_elongated <- metric_data[which(metric_data$Type=="Elongated"),]
metric_data_handaxe   <- metric_data[which(metric_data$Type=="Handaxe"),]
metric_data_tanged    <- metric_data[which(metric_data$Type=="Tanged"),]

figure_7a <- ggplot(metric_data_elongated, aes(Length_mm, Width_mm, colour = Artefact, shape = Class)) +
  geom_point() +
  facet_grid(cols = vars(Artefact), scales = "free", labeller=label_both) +
  labs(x = "Length (mm)", y = "Width (mm)") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none")

figure_7b <- ggplot(metric_data_tanged, aes(Length_mm, Width_mm, colour = Artefact, shape = Class)) +
  geom_point() +
  facet_grid(cols = vars(Artefact), scales = "free", labeller=label_both) +
  labs(x = "Length (mm)", y = "Width (mm)") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
```

```
        legend.position = "none")

figure_7c <- ggplot(metric_data_handaxe, aes(Length_mm, Width_mm, colour = Artefact, shape = Class)) +
  geom_point() +
  facet_grid(cols = vars(Artefact), scales = "free", labeller=label_both) +
  labs(x = "Length (mm)", y = "Width (mm)") +
  scale_shape_manual(values=c(17,15,3,16)) +
  theme_minimal() +
  theme(axis.text.x = element_blank(),
        axis.text.y = element_blank(),
        legend.position = "none")

figure_7 <- plot_grid(figure_7a,
                      figure_7b,
                      figure_7c,
                      labels= "AUTO",
                      ncol = 1,
                      align = 'v')

plot_grid(figure_7, lda.figure.legend, ncol = 1, rel_heights = c(1, .1))
```



Stage 8: Measurement Data (Analytical Framework)

```
metric_data_elongated <- arrange(metric_data_elongated, Artefact, Class)
```

```r
metric_data_elongated[1:18,]  %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 4 x 7
##   Class        count mean     sd       min    max    cv
##   <chr>        <chr> <chr>    <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     110.6033 1.516652 108.99 112.00 1.371253
## 2 Novice       13    110.8454 1.739443 106.87 113.60 1.569251
## 3 Photograph   1     115.4700 NA       115.47 115.47 NA
## 4 Professional 1     115.1400 NA       115.14 115.14 NA
```

```r
metric_data_elongated[1:18,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.725882
```

```r
metric_data_elongated[19:36,] %>%
  group_by(Class) %>%
  summarise(count = n(), mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 4 x 7
##   Class        count mean     sd       min    max    cv
##   <chr>        <chr> <chr>    <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     126.6633 1.915446 124.67 128.49 1.512234
## 2 Novice       13    128.3285 2.477559 123.27 132.80 1.930639
## 3 Photograph   1     131.8400 NA       131.84 131.84 NA
## 4 Professional 1     131.3500 NA       131.35 131.35 NA
```

```r
metric_data_elongated[19:36,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.932418
```

```r
metric_data_elongated[37:54,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 4 x 7
##   Class        count mean     sd       min    max    cv
##   <chr>        <chr> <chr>    <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     123.8233 1.464559 122.43 125.35 1.182781
## 2 Novice       13    123.6092 1.578076 121.67 127.20 1.276665
## 3 Photograph   1     129.6800 NA       129.68 129.68 NA
## 4 Professional 1     128.4700 NA       128.47 128.47 NA
```

```r
metric_data_elongated[37:54,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.511663
```

```r
metric_data_elongated[55:72,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 4 x 7
##   Class        count mean     sd       min   max   cv
##   <chr>        <chr> <chr>    <chr>    <chr> <chr> <chr>
## 1 Intermediate 3     95.60333 1.429452 94.37 97.17 1.495191
## 2 Novice       13    96.54923 1.292810 94.31 99.33 1.339016
## 3 Photograph   1     97.38000 NA       97.38 97.38 NA
## 4 Professional 1     98.30000 NA       98.30 98.30 NA
```

```r
metric_data_elongated[55:72,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count      cv
## 1    17 1.41499
```

```r
metric_data_elongated[73:90,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 4 x 7
##   Class        count mean   sd       min    max    cv
##   <chr>        <chr> <chr>  <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     181.76 1.408226 180.17 182.85 0.7747721
## 2 Novice       13    183.56 1.501927 181.08 186.11 0.8182210
## 3 Photograph   1     191.97 NA       191.97 191.97 NA
## 4 Professional 1     189.22 NA       189.22 189.22 NA
```

```r
metric_data_elongated[73:90,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.161859
```

```r
shapiro.test(metric_data_elongated$Length_mm[2:4])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[2:4]
## W = 0.994, p-value = 0.8519
```

```r
shapiro.test(metric_data_elongated$Width_mm[2:4])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[2:4]
## W = 0.8097, p-value = 0.1379
```

```r
shapiro.test(metric_data_elongated$Length_mm[5:17])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[5:17]
## W = 0.93829, p-value = 0.4352
```

```r
shapiro.test(metric_data_elongated$Width_mm[5:17])
```

```
##
##  Shapiro-Wilk normality test
##
```

```
## data:  metric_data_elongated$Width_mm[5:17]
## W = 0.95188, p-value = 0.6271
```

```r
shapiro.test(metric_data_elongated$Length_mm[20:22])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[20:22]
## W = 0.86266, p-value = 0.2749
```

```r
shapiro.test(metric_data_elongated$Width_mm[20:22])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[20:22]
## W = 0.99576, p-value = 0.8755
```

```r
shapiro.test(metric_data_elongated$Length_mm[23:35])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[23:35]
## W = 0.9602, p-value = 0.7568
```

```r
shapiro.test(metric_data_elongated$Width_mm[23:35])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[23:35]
## W = 0.88869, p-value = 0.0936
```

```r
shapiro.test(metric_data_elongated$Length_mm[38:40])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[38:40]
## W = 0.87597, p-value = 0.3127
```

```r
shapiro.test(metric_data_elongated$Width_mm[38:40])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[38:40]
## W = 0.93589, p-value = 0.5111
```

```r
shapiro.test(metric_data_elongated$Length_mm[41:53])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[41:53]
## W = 0.86234, p-value = 0.04136
```

```r
shapiro.test(metric_data_elongated$Width_mm[41:53])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[41:53]
## W = 0.92754, p-value = 0.3164
```

```r
shapiro.test(metric_data_elongated$Length_mm[56:58])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[56:58]
## W = 0.91854, p-value = 0.4472
```

```r
shapiro.test(metric_data_elongated$Width_mm[56:58])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[56:58]
## W = 0.90007, p-value = 0.3857
```

```r
shapiro.test(metric_data_elongated$Length_mm[59:71])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[59:71]
## W = 0.97937, p-value = 0.9763
```

```r
shapiro.test(metric_data_elongated$Width_mm[59:71])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[59:71]
## W = 0.98079, p-value = 0.9831
```

```r
shapiro.test(metric_data_elongated$Length_mm[74:76])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[74:76]
## W = 0.80888, p-value = 0.1359
```

```r
shapiro.test(metric_data_elongated$Width_mm[74:76])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[74:76]
## W = 0.93099, p-value = 0.4923
```

```r
shapiro.test(metric_data_elongated$Length_mm[77:89])
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Length_mm[77:89]
## W = 0.83103, p-value = 0.01632
```

```r
shapiro.test(metric_data_elongated$Width_mm[77:89])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_elongated$Width_mm[77:89]
## W = 0.95961, p-value = 0.7476
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_elongated[1:18,]))
```

```
##           Df Pillai approx F num Df den Df  Pr(>F)
## Class      3 0.7104   2.5707      6     28 0.04128 *
## Residuals 14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_elongated[1:18,]))
```

```
##  Response Length_mm :
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Class        3 36.277  12.092  4.1384 0.02699 *
## Residuals   14 40.908   2.922
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response Width_mm :
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Class        3 30.542 10.1808  3.4272 0.04676 *
## Residuals   14 41.588  2.9706
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_elongated[19:36,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.59119   1.9583      6     28  0.106
## Residuals 14
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_elongated[19:36,]))
```

```
##  Response Length_mm :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Class        3 29.647  9.8824  1.7081  0.211
## Residuals   14 80.997  5.7855
##
##  Response Width_mm :
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Class        3 74.511 24.8370  5.0069 0.01451 *
## Residuals   14 69.448  4.9606
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_elongated[37:54,]))
```

```
##           Df  Pillai approx F num Df den Df  Pr(>F)
## Class      3 0.78278   3.0011      6     28 0.02159 *
## Residuals 14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_elongated[37:54,]))
```

```
##  Response Length_mm :
##             Df Sum Sq Mean Sq F value   Pr(>F)
## Class        3 53.177  17.726  7.2617 0.003573 **
## Residuals   14 34.174   2.441
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response Width_mm :
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Class        3 11.829  3.9430  3.2225 0.05519 .
## Residuals   14 17.130  1.2236
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_elongated[55:72,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.37886   1.0906      6     28 0.3922
## Residuals 14
```

```
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_elongated[55:72,]))
```

```
##  Response Length_mm :
##             Df  Sum Sq Mean Sq F value Pr(>F)
## Class        3  6.4359  2.1453   1.244 0.3312
## Residuals   14 24.1430  1.7245
##
##  Response Width_mm :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Class        3 10.653  3.5511  0.9591 0.4392
## Residuals   14 51.837  3.7026
```

```
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_elongated[73:90,]))
```

```
##           Df  Pillai approx F num Df den Df   Pr(>F)
## Class      3 0.96814   4.3785      6     28 0.003066 **
## Residuals 14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_elongated[73:90,]))
```

```
##  Response Length_mm :
##             Df  Sum Sq Mean Sq F value    Pr(>F)
## Class        3 108.308  36.103  16.286 7.665e-05 ***
## Residuals   14  31.036   2.217
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response Width_mm :
##             Df  Sum Sq Mean Sq F value  Pr(>F)
## Class        3  74.424 24.8082   2.875 0.07372 .
## Residuals   14 120.803  8.6288
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
metric_data_handaxe <- arrange(metric_data_handaxe, Artefact, Class)

metric_data_handaxe[1:18,]  %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 4 x 7
##   Class        count mean     sd       min    max    cv
##   <chr>        <chr> <chr>    <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     115.6233 5.201167 110.63 121.01 4.498371
## 2 Novice       13    117.6562 2.969151 113.84 123.76 2.523583
## 3 Photograph   1     122.2600 NA       122.26 122.26 NA
## 4 Professional 1     123.4300 NA       123.43 123.43 NA
```

```r
metric_data_handaxe[1:18,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 3.047414
```

```r
metric_data_handaxe[19:36,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
```

```
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 4 x 7
##   Class        count mean     sd       min    max    cv
##   <chr>        <chr> <chr>    <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     102.0900 0.080000 102.01 102.17 0.07836223
## 2 Novice       13    101.1138 1.780503 98.74  104.27 1.76088978
## 3 Photograph   1     105.3500 NA       105.35 105.35 NA
## 4 Professional 1     105.5600 NA       105.56 105.56 NA
```

```
metric_data_handaxe[19:36,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.866549
```

```
metric_data_handaxe[37:54,] %>%
  group_by(Class) %>%
  summarise(count = n(), mean = mean(Length_mm, na.rm = TRUE), sd = sd(Length_mm, na.rm = TRUE), min = 
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 4 x 7
##   Class        count mean     sd       min    max    cv
##   <chr>        <chr> <chr>    <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     102.9600 1.015923 101.92 103.95 0.9867164
## 2 Novice       13    103.3415 2.663196 97.45  108.02 2.5770820
## 3 Photograph   1     102.4300 NA       102.43 102.43 NA
## 4 Professional 1     102.4400 NA       102.44 102.44 NA
```

```
metric_data_handaxe[37:54,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(), cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 2.274329
```

```
metric_data_handaxe[55:72,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 4 x 7
##   Class        count mean     sd       min    max    cv
```

```
##   <chr>        <chr> <chr>    <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     188.9167 7.650054 181.25 196.55 4.049433
## 2 Novice       13    187.2654 3.720118 177.76 192.04 1.986549
## 3 Photograph   1     192.6300 NA       192.63 192.63 NA
## 4 Professional 1     192.8800 NA       192.88 192.88 NA
```

```
metric_data_handaxe[55:72,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 2.366245
```

```
metric_data_handaxe[73:90,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 4 x 7
##   Class        count mean     sd       min    max    cv
##   <chr>        <chr> <chr>    <chr>    <chr>  <chr>  <chr>
## 1 Intermediate 3     180.1900 1.185411 178.91 181.25 0.6578674
## 2 Novice       13    180.1177 2.722349 172.88 183.23 1.5114281
## 3 Photograph   1     186.7500 NA       186.75 186.75 NA
## 4 Professional 1     186.3700 NA       186.37 186.37 NA
```

```
metric_data_handaxe[73:90,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.569395
```

```
shapiro.test(metric_data_handaxe$Length_mm[2:4])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[2:4]
## W = 0.85647, p-value = 0.2579
```

```
shapiro.test(metric_data_handaxe$Width_mm[2:4])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[2:4]
```

```
## W = 0.97381, p-value = 0.6896
```

```r
shapiro.test(metric_data_handaxe$Length_mm[5:17])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[5:17]
## W = 0.85528, p-value = 0.0334
```

```r
shapiro.test(metric_data_handaxe$Width_mm[5:17])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[5:17]
## W = 0.94153, p-value = 0.4771
```

```r
shapiro.test(metric_data_handaxe$Length_mm[20:22])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[20:22]
## W = 0.80959, p-value = 0.1376
```

```r
shapiro.test(metric_data_handaxe$Width_mm[20:22])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[20:22]
## W = 0.75, p-value < 2.2e-16
```

```r
shapiro.test(metric_data_handaxe$Length_mm[23:35])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[23:35]
## W = 0.92891, p-value = 0.3298
```

```r
shapiro.test(metric_data_handaxe$Width_mm[23:35])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[23:35]
## W = 0.95638, p-value = 0.6972
```

```r
shapiro.test(metric_data_handaxe$Length_mm[38:40])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[38:40]
## W = 0.99997, p-value = 0.9892
```

```r
shapiro.test(metric_data_handaxe$Width_mm[38:40])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[38:40]
## W = 0.98821, p-value = 0.7922
```

```r
shapiro.test(metric_data_handaxe$Length_mm[41:53])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[41:53]
## W = 0.94426, p-value = 0.5144
```

```r
shapiro.test(metric_data_handaxe$Width_mm[41:53])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[41:53]
## W = 0.64516, p-value = 0.0001654
```

```r
shapiro.test(metric_data_handaxe$Length_mm[56:58])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[56:58]
## W = 0.99874, p-value = 0.9323
```

```r
shapiro.test(metric_data_handaxe$Width_mm[56:58])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[56:58]
## W = 0.83824, p-value = 0.2095
```

```r
shapiro.test(metric_data_handaxe$Length_mm[59:71])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[59:71]
## W = 0.923, p-value = 0.2753
```

```r
shapiro.test(metric_data_handaxe$Width_mm[59:71])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[59:71]
## W = 0.87222, p-value = 0.05601
```

```r
shapiro.test(metric_data_handaxe$Length_mm[74:76])
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[74:76]
## W = 0.92031, p-value = 0.4534
```

```r
shapiro.test(metric_data_handaxe$Width_mm[74:76])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[74:76]
## W = 0.80075, p-value = 0.1163
```

```r
shapiro.test(metric_data_handaxe$Length_mm[77:89])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Length_mm[77:89]
## W = 0.94055, p-value = 0.464
```

```r
shapiro.test(metric_data_handaxe$Width_mm[77:89])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_handaxe$Width_mm[77:89]
## W = 0.98557, p-value = 0.9964
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_handaxe[1:18,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.54763   1.7596      6     28 0.1441
## Residuals 14
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_handaxe[1:18,]))
```

```
##  Response Length_mm :
##             Df  Sum Sq Mean Sq F value Pr(>F)
## Class        3  65.912  21.971  1.9237 0.1722
## Residuals   14 159.895  11.421
##
##  Response Width_mm :
##             Df  Sum Sq Mean Sq F value  Pr(>F)
## Class        3  76.939 25.6464  2.8436 0.07571 .
## Residuals   14 126.264  9.0189
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_handaxe[19:36,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.59356   1.9695      6     28 0.1042
## Residuals 14
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_handaxe[19:36,]))
```

```
##  Response Length_mm :
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Class        3 33.082 11.0275  4.0569 0.02869 *
## Residuals   14 38.055  2.7182
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response Width_mm :
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Class        3 45.094 15.0313  3.6851 0.03813 *
## Residuals   14 57.105  4.0789
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
adonis(cbind(Length_mm, Width_mm) ~ Class,
       data = metric_data_handaxe[37:54,])
```

```
##
## Call:
## adonis(formula = cbind(Length_mm, Width_mm) ~ Class, data = metric_data_handaxe[37:54,      ])
##
## Permutation: free
## Number of permutations: 999
##
## Terms added sequentially (first to last)
##
##           Df SumsOfSqs    MeanSqs F.Model      R2 Pr(>F)
## Class      3 0.0003423 0.00011411 0.49079 0.09516  0.592
## Residuals 14 0.0032550 0.00023250         0.90484
## Total     17 0.0035973                    1.00000
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_handaxe[55:72,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.27216  0.73507      6     28 0.6256
## Residuals 14
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_handaxe[55:72,]))
```

```
##  Response Length_mm :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Class        3  54.38  18.127  0.8963 0.4674
## Residuals   14 283.12  20.223
##
##  Response Width_mm :
##             Df  Sum Sq Mean Sq F value Pr(>F)
## Class        3  84.734  28.245  1.5192 0.2531
## Residuals   14 260.287  18.592
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_handaxe[73:90,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.57182   1.8685      6     28 0.1218
## Residuals 14
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_handaxe[73:90,]))
```

```
##  Response Length_mm :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Class        3 73.558 24.5195  3.7416 0.0365 *
## Residuals   14 91.745  6.5532
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response Width_mm :
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Class        3 135.10  45.034  3.9909 0.03015 *
## Residuals   14 157.98  11.284
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
metric_data_tanged <- arrange(metric_data_tanged, Artefact, Class)

metric_data_tanged[1:18,]  %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 4 x 7
##   Class        count mean     sd        min   max   cv
##   <chr>        <chr> <chr>    <chr>     <chr> <chr> <chr>
## 1 Intermediate 3     84.07333 0.4119871 83.76 84.54 0.490033
## 2 Novice       13    84.19154 1.4370679 81.38 85.84 1.706903
## 3 Photograph   1     87.12000 NA        87.12 87.12 NA
## 4 Professional 1     87.41000 NA        87.41 87.41 NA
```

```r
metric_data_tanged[1:18,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.754211
```

```r
metric_data_tanged[19:36,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
```

```r
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 4 x 7
##   Class        count mean     sd        min    max    cv
##   <chr>        <chr> <chr>    <chr>     <chr>  <chr>  <chr>
## 1 Intermediate 3     105.6333 0.5614564 105.27 106.28 0.5315145
## 2 Novice       13    105.8038 2.0141356 101.13 108.62 1.9036507
## 3 Photograph   1     107.9200 NA        107.92 107.92 NA
## 4 Professional 1     108.2000 NA        108.20 108.20 NA
```

```r
metric_data_tanged[19:36,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(), cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.749417
```

```r
metric_data_tanged[37:54,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 4 x 7
##   Class        count mean     sd        min   max   cv
##   <chr>        <chr> <chr>    <chr>     <chr> <chr> <chr>
## 1 Intermediate 3     87.53000 0.8166395 86.66 88.28 0.9329824
## 2 Novice       13    88.27692 1.5441901 84.78 90.49 1.7492568
## 3 Photograph   1     88.83000 NA        88.83 88.83 NA
## 4 Professional 1     89.23000 NA        89.23 89.23 NA
```

```r
metric_data_tanged[37:54,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 1.614196
```

```r
metric_data_tanged[55:72,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
```

```
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 4 x 7
##   Class        count mean     sd       min   max   cv
##   <chr>        <chr> <chr>    <chr>    <chr> <chr> <chr>
## 1 Intermediate 3     54.83000 1.783003 53.64 56.88 3.251875
## 2 Novice       13    54.95231 1.324425 53.20 57.82 2.410136
## 3 Photograph   1     57.01000 NA       57.01 57.01 NA
## 4 Professional 1     57.80000 NA       57.80 57.80 NA
```

```
metric_data_tanged[55:72,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 2.691978
```

```
metric_data_tanged[73:90,] %>%
  group_by(Class) %>%
  summarise(count = n(),
            mean = mean(Length_mm, na.rm = TRUE),
            sd = sd(Length_mm, na.rm = TRUE),
            min = min(Length_mm, na.rm = TRUE),
            max = max(Length_mm, na.rm = TRUE),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 4 x 7
##   Class        count mean     sd        min   max   cv
##   <chr>        <chr> <chr>    <chr>     <chr> <chr> <chr>
## 1 Intermediate 3     60.11000 0.7754354 59.58 61.00 1.290027
## 2 Novice       13    59.83846 1.2881631 58.06 62.62 2.152734
## 3 Photograph   1     62.81000 NA        62.81 62.81 NA
## 4 Professional 1     62.22000 NA        62.22 62.22 NA
```

```
metric_data_tanged[73:90,]  %>%
  filter(Class=="Novice" | Class == "Intermediate" | Class == "Professional") %>%
  summarise(count = n(),
            cv = sd(Length_mm, na.rm = TRUE)/mean(Length_mm, na.rm = TRUE)*100) %>%
  dplyr::mutate_if(is.numeric, format, 1)
```

```
##   count       cv
## 1    17 2.140211
```

```
shapiro.test(metric_data_tanged$Length_mm[2:4])
```

```
##
##  Shapiro-Wilk normality test
##
```

```
## data:  metric_data_tanged$Length_mm[2:4]
## W = 0.95243, p-value = 0.5801
```

```r
shapiro.test(metric_data_tanged$Width_mm[2:4])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[2:4]
## W = 0.99986, p-value = 0.9773
```

```r
shapiro.test(metric_data_tanged$Length_mm[5:17])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[5:17]
## W = 0.96154, p-value = 0.7772
```

```r
shapiro.test(metric_data_tanged$Width_mm[5:17])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[5:17]
## W = 0.94549, p-value = 0.5318
```

```r
shapiro.test(metric_data_tanged$Length_mm[20:22])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[20:22]
## W = 0.99957, p-value = 0.9604
```

```r
shapiro.test(metric_data_tanged$Width_mm[20:22])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[20:22]
## W = 0.85465, p-value = 0.253
```

```r
shapiro.test(metric_data_tanged$Length_mm[23:35])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[23:35]
## W = 0.93546, p-value = 0.4009
```

```r
shapiro.test(metric_data_tanged$Width_mm[23:35])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[23:35]
## W = 0.95349, p-value = 0.652
```

```r
shapiro.test(metric_data_tanged$Length_mm[38:40])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[38:40]
## W = 0.93746, p-value = 0.5173
```

```r
shapiro.test(metric_data_tanged$Width_mm[38:40])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[38:40]
## W = 0.99882, p-value = 0.9344
```

```r
shapiro.test(metric_data_tanged$Length_mm[41:53])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[41:53]
## W = 0.946, p-value = 0.5391
```

```r
shapiro.test(metric_data_tanged$Width_mm[41:53])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[41:53]
## W = 0.91872, p-value = 0.2411
```

```r
shapiro.test(metric_data_tanged$Length_mm[56:58])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[56:58]
## W = 0.99389, p-value = 0.8506
```

```r
shapiro.test(metric_data_tanged$Width_mm[56:58])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[56:58]
## W = 0.83275, p-value = 0.1953
```

```r
shapiro.test(metric_data_tanged$Length_mm[59:71])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[59:71]
## W = 0.93757, p-value = 0.4262
```

```r
shapiro.test(metric_data_tanged$Width_mm[59:71])
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[59:71]
## W = 0.94463, p-value = 0.5195
```

```
shapiro.test(metric_data_tanged$Length_mm[74:76])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[74:76]
## W = 0.99856, p-value = 0.9275
```

```
shapiro.test(metric_data_tanged$Width_mm[74:76])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[74:76]
## W = 0.98702, p-value = 0.7819
```

```
shapiro.test(metric_data_tanged$Length_mm[77:89])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Length_mm[77:89]
## W = 0.92621, p-value = 0.3038
```

```
shapiro.test(metric_data_tanged$Width_mm[77:89])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  metric_data_tanged$Width_mm[77:89]
## W = 0.9138, p-value = 0.2066
```

```
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_tanged[1:18,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.56874   1.8544      6     28 0.1245
## Residuals 14
```

```
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_tanged[1:18,]))
```

```
##  Response Length_mm :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Class        3 17.112  5.7041  3.1789 0.0572 .
## Residuals   14 25.121  1.7944
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response Width_mm :
##             Df  Sum Sq Mean Sq F value Pr(>F)
## Class        3  2.5154 0.83848  0.4928  0.693
## Residuals   14 23.8228 1.70163
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_tanged[19:36,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.48288   1.4853      6     28 0.2193
## Residuals 14
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_tanged[19:36,]))
```

```
##  Response Length_mm :
##              Df Sum Sq Mean Sq F value Pr(>F)
## Class         3  9.418  3.1392  0.8913 0.4698
## Residuals    14 49.311  3.5222
##
##  Response Width_mm :
##              Df  Sum Sq Mean Sq F value  Pr(>F)
## Class         3  8.0821 2.69404  3.1918 0.05659 .
## Residuals    14 11.8169 0.84406
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_tanged[37:54,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.23153  0.61096      6     28 0.7195
## Residuals 14
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_tanged[37:54,]))
```

```
##  Response Length_mm :
##              Df Sum Sq Mean Sq F value Pr(>F)
## Class         3  2.858 0.95265  0.4453 0.7244
## Residuals    14 29.948 2.13915
##
##  Response Width_mm :
##              Df  Sum Sq Mean Sq F value Pr(>F)
## Class         3  3.4126  1.1376  0.9378 0.4486
## Residuals    14 16.9828  1.2131
```

```r
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_tanged[55:72,]))
```

```
##           Df  Pillai approx F num Df den Df Pr(>F)
## Class      3 0.52873   1.6771      6     28 0.1636
## Residuals 14
```

```r
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_tanged[55:72,]))
```

```
##  Response Length_mm :
##              Df Sum Sq Mean Sq F value Pr(>F)
## Class         3 11.244  3.7480  1.9145 0.1736
## Residuals    14 27.407  1.9577
##
```

```
##  Response Width_mm :
##             Df  Sum Sq Mean Sq F value  Pr(>F)
## Class        3  7.6851  2.5617  2.8893 0.07283 .
## Residuals   14 12.4124  0.8866
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(manova(cbind(Length_mm, Width_mm) ~ Class,
               data = metric_data_tanged[73:90,]))
```

```
##           Df  Pillai approx F num Df den Df  Pr(>F)
## Class      3 0.66976   2.3496      6     28 0.05791 .
## Residuals 14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary.aov(manova(cbind(Length_mm, Width_mm) ~ Class,
                   data = metric_data_tanged[73:90,]))
```

```
##  Response Length_mm :
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Class        3 12.610  4.2032  2.7869 0.07947 .
## Residuals   14 21.115  1.5082
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response Width_mm :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Class        3 2.8109 0.93696  1.3714 0.2922
## Residuals   14 9.5652 0.68323
```

Concluding Remarks

For any queries, or for more information on any aspect of this markdown, please contact C.S.Hoggard@soton.ac.uk.

Acknowledgements

We thank Moesgaard Museum for loaning the artefacts used throughout this study. We would also like to thank all illustrators for their time during this experiment. CSH and FR thank the Independent Research Fund Denmark for grant #610700059B and FR also gratefully acknowledges funding the European Research Council (grant agreement 817564 under the Horizon 2020 research and innovation programme). We also thank John McNabb for comments on earlier drafts of this paper.