Lithic analysis of both squares

Ben Marwick

November 10, 2015

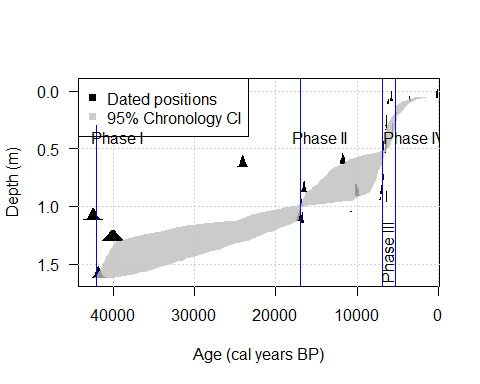
# Results for square A and square B combined

Here we combine stone artefacts from both squares.

# Chronology of the excavated deposit

The code chunk below reproduces Figure 2, the depth-age distribution for radiocarbon dates from Jerimalai square B.

sqB\_dates <- read.csv("data/Jerimalai\_dates\_Square\_B.csv", as.is = TRUE)  
sqA\_dates <- read.csv("data/Jerimalai\_dates\_Square\_A.csv", as.is = TRUE)  
sqA\_dates$sq <- "A"  
sqB\_dates$sq <- "B"  
both\_sqs\_dates <- rbind(sqB\_dates, sqA\_dates)  
  
# we have to put in order to let the calibration work...  
both\_sqs\_dates <- both\_sqs\_dates %>%  
 arrange(depth\_bs)   
  
 # ages <- BchronCalibrate(ages = dates$age,  
 # ageSds = dates$error,  
 # positions = dates$depth\_bs,   
 # calCurves = rep("intcal13",   
 # length(dates$age)))  
# show tables of calibrated age ranges for each date  
# summary(ages)  
  
# plot(ages, xlab='Age (cal years BP)', withPositions = TRUE)  
  
  
ages\_predict = Bchronology(ages = both\_sqs\_dates$age,  
 ageSds = both\_sqs\_dates$error,  
 positions = both\_sqs\_dates$depth\_bs,   
 positionThicknesses = 0.01,  
 ids = both\_sqs\_dates$lab\_code,  
 calCurves = rep("intcal13", length(both\_sqs\_dates$age)))  
  
# save plot  
png("figures/fig\_2-Jeremalai-dates.png",   
 width = 200,   
 height = 120,   
 units = "mm", res = 100)  
  
bchron\_plot <- function(){  
plot(ages\_predict,  
 main="",  
 xlab='Age (cal years BP)',  
 ylab='Depth (m)',  
 las=1 ) #,   
 #asp=0.6)  
  
# add phases  
  
phases <- data.frame(phase = 1:4,  
 start = c(42, 17, 6.9, 5.3),  
 end = c(35, 9, 5.5, 0 ))  
  
phase\_lines <- (phases$start \* 1000)  
line\_height <- c(1.7, 0.3, -0.1, 0.4)  
  
lines(rep(phase\_lines[1],2), line\_height[1:2], col = "blue")  
lines(rep(phase\_lines[2],2), line\_height[c(1,3)], col = "blue")  
lines(rep(phase\_lines[3],2), line\_height[c(1,3)], col = "blue")  
lines(rep(phase\_lines[4],2), line\_height[c(1,3)], col = "blue")  
  
text(phase\_lines[1] - 2500, line\_height[4], labels = "Phase I")  
text(phase\_lines[2] - 2500, line\_height[4], labels = "Phase II")  
text(phase\_lines[3] - 600, line\_height[4]+1, labels = "Phase III", srt = 90)  
text(phase\_lines[4] - 2500, line\_height[4] , labels = "Phase IV")  
}  
bchron\_plot()  
# end saving plot  
dev.off()  
  
# combine all the calibrated dates into a single plot (using results = 'hide' to hide the progress bar)  
# ages\_densities <- BchronDensity(ages = dates$age,  
# ageSds = dates$error,  
#   
# calCurves = rep("intcal13", length(dates$age)))  
  
# plot(ages\_densities, xlab='Age (cal years BP)', withPositions = TRUE)  
  
# and show the plot when this document is knited  
bchron\_plot()



# various dadta  
core\_types <- read.csv("data/Jerimalai\_cores\_techno\_metrics.csv")  
# retouch indices  
retouch\_indices <- read.csv("data/Jerimalai\_retouch\_indices.csv")  
# sediment volumes  
vols <- read.csv("data/Artefact densities with soil volumes Sq B.csv", skip = 1)  
  
  
# read in sq B data  
sqB\_all <- read.csv("data/Jerimalai\_All\_Artefacts\_Square\_B.csv")  
sqB\_depths <- read.csv("data/Jerimalai\_spit\_depths\_Square\_B.csv")  
  
  
# read in sq A data  
sqA\_all <- read.csv("data/Jerimalai\_All\_Artefacts\_Square\_A.csv") %>%   
 filter(Square == "A")  
sqA\_depths <- read.csv("data/Jerimalai\_spit\_depths\_Square\_A.csv")   
  
# merge A and B  
names\_both <- intersect(names(sqA\_all), names(sqB\_all))  
# select only columns in both data sets  
sqA\_all\_s <- sqA\_all %>% select\_(.dots = names\_both)  
sqB\_all\_s <- sqB\_all %>% select\_(.dots = names\_both)  
  
# combine  
both\_sqs\_all <- rbind(sqA\_all\_s, sqB\_all\_s)  
  
# subset flakes  
both\_sqs\_flakes <- both\_sqs\_all %>%  
 filter(Artclas == "Flake")

# Results: Raw materials at both squares

The code chunk below computes the percentages of the two major cortex types for chert flakes, and displays the result in a table.

cortex\_type <- as.data.frame(table(both\_sqs\_flakes$Cortype))[-1,]  
cortex\_type$prop <- round(prop.table(cortex\_type$Freq),2)  
names(cortex\_type) <- c("Cortex\_type", "Frequency", "Proportion")  
kable(cortex\_type, caption = "Cortex type among chert flakes")

Cortex type among chert flakes

|  |  |  |  |
| --- | --- | --- | --- |
|  | Cortex\_type | Frequency | Proportion |
| 2 | Angul | 315 | 0.57 |
| 3 | Round | 236 | 0.43 |

The chunk above also generates the values that are found in this sentence in the manuscript: "The chert artefacts have a combination of rounded cortex (43%) and angular cortex (57%.")

The code chunk below subsets the stone artefact data so that we only include flakes without any missing mass data. We then assign each flake to a depositional phase based on the spit it was recovered from. There is no visual output from this chunk but the values are stored and used in the following chunks.

# we have to do this separately for sq A and sq B.  
## sq A  
  
# omit rows with blanks or NAs  
sqA\_all <- both\_sqs\_all %>%  
 filter(Square == "A") %>%  
 filter(Weight != "") %>%  
 filter(!is.na(Weight))  
  
# put depths on lithic data  
sqA\_all$depth <- sqA\_depths$Depth.bs..m[match(sqA\_all$Spit,sqA\_depths$Spit.no)]  
  
# omit rows with blanks or NAs... again  
sqA\_all <- sqA\_all[!(sqA\_all$depth == "" | is.na(sqA\_all$depth)), ]  
  
# make phases for sq A  
sqA\_makephases <- function(x) {  
   
 ifelse(x >= 1 & x <= 5, 4,  
 ifelse(x >= 6 & x <= 25, 3,  
 ifelse(x >= 26 & x <= 38, 2,  
 ifelse(x >= 39 & x <= 46, 1, NA))))  
}  
  
sqA\_all$phase <- sqA\_makephases(sqA\_all$Spit)  
  
# check if any NA  
check <- unique(sqA\_all$phase)  
# omit NA  
sqA\_all <- sqA\_all %>% filter(!is.na(phase))  
  
## sq B  
  
# omit rows with blanks or NAs  
sqB\_all<- both\_sqs\_all %>%  
 filter(Square == "B") %>%  
 filter(Weight != "") %>%  
 filter(!is.na(Weight))  
  
# put depths on lithic data  
sqB\_all$depth <- sqB\_depths$Depth.bs..m[match(sqB\_all$Spit,sqB\_depths$Spit.no)]  
  
# omit rows with blanks or NAs... again  
sqB\_all <- sqB\_all[!(sqB\_all$depth == "" | is.na(sqB\_all$depth)), ]  
  
# make phases for sq B  
sqB\_makephases <- function(x) {  
 ifelse(x > 3 & x <= 20, 4,  
 ifelse(x >= 21 & x <= 39, 3,  
 ifelse(x >= 40 & x <= 48, 2,  
 ifelse(x >= 49 & x <= 69, 1, NA))))  
  
}  
  
sqB\_all$phase <- sqB\_makephases(sqB\_all$Spit)  
  
# check if any NA  
check <- unique(sqB\_all$phase)  
# omit NA  
sqB\_all <- sqB\_all %>% filter(!is.na(phase))  
  
######  
  
# combine both squares again  
both\_sqs\_all\_with\_phases <- rbind(sqA\_all, sqB\_all)

The code chunk below generates a table that summarises the frequencies of the main raw materials in each depositional phase. A raw material is considered dominant here if there are more than ten peices in a phase.

# get all flakes  
flakes <- both\_sqs\_all\_with\_phases %>% filter(Artclas == "Flake")  
# raw material  
raw <- dcast(flakes, Material ~ phase)   
rownames(raw) <- raw[,1]  
# get rid of rows with no value  
raw <- raw[rownames(raw) != "",]  
# remove col of NA  
raw <- raw[,colnames(raw) != "NA"]  
raw <- raw[,-1]  
# subset dominant raw materials  
dom <- raw[rowSums(raw) > 10,]  
colnames(dom) <- c("phase 1", "phase 2", "phase 3", "phase 4")  
kable(dom, caption = "Frequencies of dominant raw materials by depositional phase in both squares")

Frequencies of dominant raw materials by depositional phase in both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | phase 1 | phase 2 | phase 3 | phase 4 |
| Chert | 677 | 1088 | 1789 | 818 |
| Obsidian | 0 | 5 | 1 | 5 |
| Quartz | 0 | 1 | 9 | 7 |
| Quartzite | 2 | 2 | 15 | 11 |
| Silcrete | 9 | 11 | 13 | 13 |
| Unknown | 7 | 7 | 2 | 2 |
| Volcanic | 33 | 30 | 73 | 89 |

The code chunk below uses the Bayesian Poisson exponential ANOVA to compute the probabilities of any credible interactions between raw material frequencies and depositional phase. The goal is to determine if there are any significant changes in the use of raw materials for stone artefacts over time.

data <- melt(as.matrix(dom), varnames=c("raw\_material", "phase"), value.name="Freq")  
myDataFrame = data  
yName=names(myDataFrame)[3] # Freq  
x1Name=names(myDataFrame)[2] # phase   
x2Name=names(myDataFrame)[1] # raw material  
x1contrasts = list(   
 list( c("phase 1") , c("phase 2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("phase 2") , c("phase 3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("phase 3") , c("phase 4") , compVal=0.0 , ROPE=c(-0.1,0.1))   
 )  
numSavedSteps = 12000 # MCMC parameters  
thinSteps = 10 # MCMC parameters  
  
mcmcCoda = genMCMC\_cont\_table( datFrm=myDataFrame ,   
 yName=yName , x1Name=x1Name , x2Name=x2Name ,  
 numSavedSteps=numSavedSteps , thinSteps=thinSteps ,   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 22:53:16 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 346  
  
WARNING: Unused variable(s) in data table:  
ySum  
  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 1000  
-------------------------------------------------| 1000  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 2000  
-------------------------------------------------| 2000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . . . . . . . Failed to set trace monitor for delta  
Node not found  
. Updating 40000  
-------------------------------------------------| 40000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

# Get summary statistics of chain:  
summaryInfo = smryMCMC\_cont\_table( mcmcCoda ,   
 datFrm=myDataFrame , x1Name=x1Name , x2Name=x2Name ,  
 x1contrasts=x1contrasts   
 )  
  
# # Display posterior information (not easy to read unless using interactively):  
# plotMCMC\_cont\_table( mcmcCoda ,   
# datFrm=myDataFrame , yName=yName , x1Name=x1Name ,   
# x2Name=x2Name ,  
# x1contrasts=x1contrasts )  
  
# subset summaryInfo to show HDI interval for interactions  
HDI\_intervals\_for\_interactions <- summaryInfo[c((nrow(summaryInfo)-length(x1contrasts)+1):nrow(summaryInfo)), c("HDIlow", "HDIhigh")]  
kable(round(HDI\_intervals\_for\_interactions,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and raw material frequencies in both squares.")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and raw material frequencies in both squares.

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| phase 1.v.phase 2 | -0.818 | 0.075 |
| phase 2.v.phase 3 | -0.860 | -0.106 |
| phase 3.v.phase 4 | -0.273 | 0.421 |

The code chunk above provides psoterior distributions to evaluate the credibility of differences in raw material frequences over time. The posterior distributions for the phase interactions are close to zero, indicating a small effect. Among the contrast of the phases, all HDIs exclude zero except for phase 2 v. phase 3. This indicates that there are credibly different frequencies of raw material between each phase except for 2 and 3. This is likely are result of small changes in the low frequencies of quartz, quartzite and silcrete in the earlier phases, as shown in the phase by raw material table above.

The code chunk below computes a NHST equivalent to the above Bayesian test, in this case a chi-square test. We also compute Cramer's V, a measure of association for nominal variables that ranges from 0 (no association between the variables) to 1 (strong association between the variables).

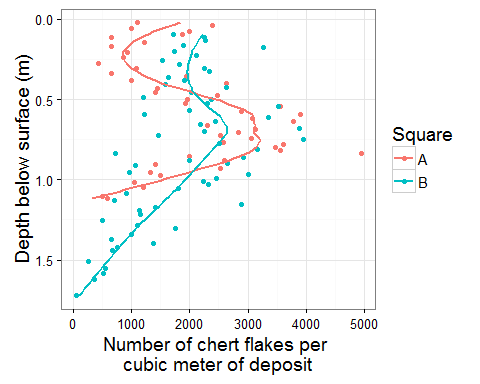
# here is the frequentist equivalent   
raw\_material\_by\_phase\_nhst <- chisq.test(dom)  
cramers\_V <- assocstats(as.matrix(dom))

The code chunk above returns a chi-squared value of 108.2111456 and a p-value of 6.770139610^{-15}, indicating a signficant difference in raw material frequencies by phase. However, the Cramer's V value of 0.087428 indicates that the effect size is extremely small. We interpret this result to mean that there is no substantial significance in the differences in raw materials frequencies by phase.

# Results: Discard rates

The code chunk below generates the figure that shows discard rates of chert artefacts over time at Jerimalai in both squares. Each point is an excavation unit. The blue line is a locally weighted regression line (span = 0.4) to aid in visualising the trend of increased discard in the upper part of the deposit.

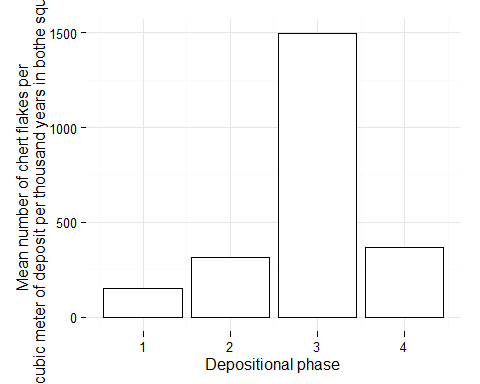
# discard rates  
discard <- aggregate(Weight ~ depth + Spit + Square, flakes, length)  
  
# get sediment volumes from CSV file  
sqA\_vols <- vols %>%  
 plyr::summarize(spit = X.3, vol = Soil.1)  
  
sqB\_vols <- vols %>%  
 plyr::summarize(spit = X, vol = Soil)  
  
sqA\_discard <- discard %>% filter(Square == "A")  
sqA\_discard$sedvol <- sqA\_vols$vol[match(sqA\_discard$Spit, sqA\_vols$spit)]  
sqA\_discard$thick <- c(0.018, diff(sqA\_discard$depth))  
sqA\_discard$kgsed <- with(sqA\_discard, Weight / sedvol)  
sqA\_discard$cubmet <- with(sqA\_discard, Weight / thick)  
  
sqB\_discard <- discard %>% filter(Square == "B")  
sqB\_discard$sedvol <- sqB\_vols$vol[match(sqB\_discard$Spit, sqB\_vols$spit)]  
sqB\_discard$thick <- c(0.018, diff(sqB\_discard$depth))  
sqB\_discard$kgsed <- with(sqB\_discard, Weight / sedvol)  
sqB\_discard$cubmet <- with(sqB\_discard, Weight / thick)  
  
# combine again  
both\_sqs\_discard <- rbind(sqA\_discard, sqB\_discard)  
  
# remove spit that is far too thin...  
both\_sqs\_discard <- both\_sqs\_discard %>% filter(thick > 0.01)  
  
  
# Plot  
ggplot(both\_sqs\_discard, (aes(depth, cubmet, colour = Square))) +  
 geom\_point() +  
 stat\_smooth(span = 0.5, se = FALSE) +  
 xlab("Depth below surface (m)") +  
 ylab("Number of chert flakes per \ncubic meter of deposit") +  
 theme(axis.text=element\_text(size=10)) +  
 coord\_flip() +  
 scale\_x\_reverse()



# save plot  
jhe\_90mm\_ggsave("figures/fig\_3\_Jeremalai-flake-discard.png")

The code chunk below the generates the figure showing discard rates of chert artefacts per depositional phase at Jerimalai square B. This aggregates the individual excavation units.

# plot artefacts/cubic meter/1000 years by phase, get phase number for each spit  
# this is the most sensible option  
both\_sqs\_discard$phase <- ifelse(both\_sqs\_discard$Square == "A",   
 sqA\_makephases(both\_sqs\_discard$Spit),   
 sqB\_makephases(both\_sqs\_discard$Spit))  
# this is ok for both squares  
phases <- data.frame(phase = 1:4,  
 start = c(42, 17, 6.9, 5.3),  
 end = c(35, 9, 5.5, 0 ))  
phases$duration <- with(phases, start - end)  
  
discard\_agg <- aggregate(cubmet ~ phase, both\_sqs\_discard, mean)  
discard\_agg$cubmetperkyr <- discard\_agg$cubmet / phases$duration  
ggplot(discard\_agg, (aes(phase, cubmetperkyr))) +  
 geom\_bar(stat="identity", colour = "black", fill = "white") +  
 theme\_minimal() +  
 xlab("Depositional phase") +  
 ylab("Mean number of chert flakes per \ncubic meter of deposit per thousand years in bothe squares")



# save plot  
jhe\_90mm\_ggsave("figures/fig\_4\_Jeremalai-flake-discard-phase-m3.png", height = 90)

# Results: Artefact taphonomy

The code chunk below computes a Bayesian Poisson exponential ANOVA to investigate differences in flake breakage classes over time.

allchert <- both\_sqs\_all[both\_sqs\_all$Material == 'Chert', ]  
# assign phases  
allchert$phase <- ifelse(allchert$Square == "A",   
 sqA\_makephases(allchert$Spit),   
 sqB\_makephases(allchert$Spit))  
  
  
# make Artclass that is long and transv breaks  
allchert$Artclas <- ifelse(allchert$Breaks == "",   
 as.character(allchert$Artclas),   
 paste(allchert$Artclas, allchert$Breaks, sep = "-"))  
taph <- data.frame(table(allchert$Artclas))  
# use regex to get broken flakes -b-   
broken <- allchert[grep("-b", allchert$Artclas), ]  
# get counts of broken to complete per phase  
# flake to -b-  
breaks <- dcast(allchert, Artclas ~ phase)[-1,]  
breaks <- breaks[breaks$Artclas =="Flake" | grepl("-b-", breaks$Artclas), ]  
allchert$Artclas <- tolower(allchert$Artclas)  
allchert$breakt <- "" # create variable to fill  
allchert$breakt[grep("trans", allchert$Artclas)] <- "trans"  
allchert$breakt[grep("long", allchert$Artclas)] <- "long"  
# per depositional phase  
breakt <- dcast(allchert, breakt ~ phase)[-1,]  
# add complete flake counts  
breakt <- rbind( breakt , setNames( breaks[1, ] , names( breakt ) ) )  
# shift rownames out and delete them  
rownames(breakt) <- breakt[,1]  
breakt <- breakt[,-1]  
# exclude artefacts not assigned to a phase  
breakt <- breakt[, -which(names(breakt) == "NA")]  
# do bayesian contingency table test  
colnames(breakt) <- c("phase 1", "phase 2", "phase 3", "phase 4")  
data <- melt(as.matrix(breakt), varnames=c("breakt", "phase"), value.name="Freq")  
  
myDataFrame = data  
yName=names(myDataFrame)[3] # Freq   
x1Name=names(myDataFrame)[2] # phase  
x2Name=names(myDataFrame)[1] # break type  
x1contrasts = list(   
 list( c("phase 1") , c("phase 2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("phase 2") , c("phase 3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("phase 3") , c("phase 4") , compVal=0.0 , ROPE=c(-0.1,0.1))   
 #list( c("phase 4") , c("phase 5") , compVal=0.0 , ROPE=c(-0.1,0.1))  
 )  
numSavedSteps = 12000 # MCMC parameters  
thinSteps = 10 # MCMC parameters  
  
mcmcCoda = genMCMC\_cont\_table( datFrm=myDataFrame ,   
 yName=yName , x1Name=x1Name , x2Name=x2Name ,  
 numSavedSteps=numSavedSteps , thinSteps=thinSteps ,   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 22:53:50 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 178  
  
WARNING: Unused variable(s) in data table:  
ySum  
  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 1000  
-------------------------------------------------| 1000  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 2000  
-------------------------------------------------| 2000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . . . . . . . Failed to set trace monitor for delta  
Node not found  
. Updating 40000  
-------------------------------------------------| 40000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

# Get summary statistics of chain:  
summaryInfo = smryMCMC\_cont\_table( mcmcCoda ,   
 datFrm=myDataFrame , x1Name=x1Name , x2Name=x2Name ,  
 x1contrasts=x1contrasts   
 )  
  
# # Display posterior information (not easy to read unless using interactively):  
# plotMCMC\_cont\_table( mcmcCoda ,   
# datFrm=myDataFrame , yName=yName , x1Name=x1Name ,   
# x2Name=x2Name ,  
# x1contrasts=x1contrasts )  
  
# table of raw counts  
kable(breakt, caption = "Table of frequencies of each class of breakage by phase, in both squares")

Table of frequencies of each class of breakage by phase, in both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | phase 1 | phase 2 | phase 3 | phase 4 |
| long | 85 | 66 | 125 | 61 |
| trans | 192 | 256 | 437 | 136 |
| Flake | 676 | 1081 | 1769 | 810 |

# subset summaryInfo to show HDI interval for interactions  
HDI\_intervals\_for\_interactions <- summaryInfo[c((nrow(summaryInfo)-length(x1contrasts)+1):nrow(summaryInfo)), c("HDIlow", "HDIhigh")]  
kable(round(HDI\_intervals\_for\_interactions,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and flake breakage classes, in both squares.")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and flake breakage classes, in both squares.

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| phase 1.v.phase 2 | -0.324 | -0.071 |
| phase 2.v.phase 3 | -0.661 | -0.439 |
| phase 3.v.phase 4 | 0.767 | 1.003 |

The above code chunk returns results that there are credible difference but small differences in the frequences of flake breakage types by phase.

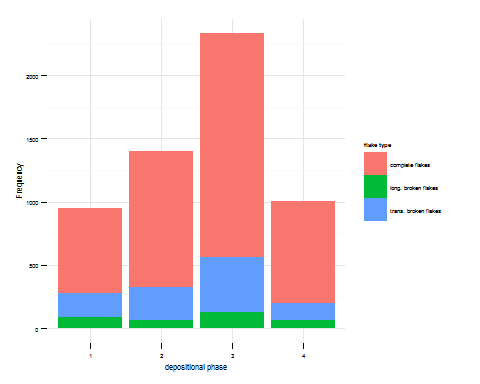
The code chunk below computes the frequentist equivalent, the chi-square test and the Cramer's V for effect size.

artefact\_taphonomy\_nhst <- assocstats(as.matrix(breakt))

The code chunk above returns a chi-squared value of 39.8021634 and a p-value of 4.981619710^{-7}, indicating a signficant difference in frequencies of breakage classes by phase. However, the Cramer's V value of 0.0591193 indicates that the effect size is extremely small. We interpret this result to mean that although the test result is statistically significant, there is no substantial significance in the differences in frequencies of breakage types by phase.

The code chunk below draws a plot of frequencies of complete flakes, transversely broken flakes and longitudinally broken flakes made from chert in each depositional unit at Jerimalai square B.

data$phase <- gsub("[[:alpha:]]\*", "", data$phase)  
# recode break type for pretty legend  
data$breakt\_ <- with(data, ifelse(breakt == 'long', 'long. broken flakes',  
 ifelse(breakt == "trans", "trans. broken flakes",  
 ifelse(breakt == "Flake", "complete flakes", NA))))  
ggplot(data, aes(phase, Freq, fill = breakt\_)) +  
 ylab("Frequency") +  
 geom\_bar(stat = "identity") +  
 scale\_fill\_discrete(name="flake type") +  
 xlab("depositional phase") +  
 theme\_minimal((base\_size = 6))



jhe\_90mm\_ggsave("figures/fig\_5\_Jeremalai-flake-broken-phase.png", height = 90/2)

The code chunk below summarises the frequencies of heat-treated flakes in both squares at Jerimalai

check <- sum(flakes$Heat, na.rm = TRUE) / nrow(flakes)  
heat <- aggregate(Heat ~ phase, flakes, length)  
total <- aggregate(Spit ~ phase, flakes, length)  
heat$Not\_heat <- total$Spit - heat$Heat  
# show proportions that are heat-treated  
max\_heat <- max(heat$Heat / total$Spit)  
min\_heat <- min(heat$Heat / total$Spit)  
heat <- t(heat)  
# do bayesian contingency table test  
colnames(heat) <- c("phase 1", "phase 2", "phase 3", "phase 4")  
heat <- heat[-1,]

Between 9% and 17% of chert artefacts in each depositional phase show signs of having been heated, such as crenation, potlid scars or surface crazing.

The code chunk below computes a Bayesian Poisson exponential ANOVA to investigate differences in heat treatment by phase.

data <- melt(as.matrix(heat), varnames=c("heat", "phase"), value.name="Freq")  
myDataFrame = data  
yName=names(myDataFrame)[3] # Freq   
x1Name=names(myDataFrame)[2] # phase  
x2Name=names(myDataFrame)[1] # heat  
x1contrasts = list(   
 list( c("phase 1") , c("phase 2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("phase 2") , c("phase 3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("phase 3") , c("phase 4") , compVal=0.0 , ROPE=c(-0.1,0.1))   
   
 )  
numSavedSteps = 12000 # MCMC parameters  
thinSteps = 10 # MCMC parameters  
  
mcmcCoda = genMCMC\_cont\_table( datFrm=myDataFrame ,   
 yName=yName , x1Name=x1Name , x2Name=x2Name ,  
 numSavedSteps=numSavedSteps , thinSteps=thinSteps ,   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 22:54:06 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 136  
  
WARNING: Unused variable(s) in data table:  
ySum  
  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 1000  
-------------------------------------------------| 1000  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 2000  
-------------------------------------------------| 2000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . . . . . . . Failed to set trace monitor for delta  
Node not found  
. Updating 40000  
-------------------------------------------------| 40000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

# Get summary statistics of chain:  
summaryInfo = smryMCMC\_cont\_table( mcmcCoda ,   
 datFrm=myDataFrame , x1Name=x1Name , x2Name=x2Name ,  
 x1contrasts=x1contrasts   
 )  
  
# # # Display posterior information (not easy to read unless using interactively):  
# plotMCMC\_cont\_table( mcmcCoda ,   
# datFrm=myDataFrame , yName=yName , x1Name=x1Name ,   
# x2Name=x2Name ,  
# x1contrasts=x1contrasts )  
  
# table of heated artefact counts  
kable(heat, caption = "Table of frequencies of heat treatment by phase at both squares")

Table of frequencies of heat treatment by phase at both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | phase 1 | phase 2 | phase 3 | phase 4 |
| Heat | 109 | 191 | 232 | 86 |
| Not\_heat | 619 | 953 | 1675 | 860 |

# subset summaryInfo to show HDI interval for interactions  
HDI\_intervals\_for\_interactions <- summaryInfo[c((nrow(summaryInfo)-length(x1contrasts)+1):nrow(summaryInfo)), c("HDIlow", "HDIhigh")]  
kable(round(HDI\_intervals\_for\_interactions,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and flake heat treatment at both squares.")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and flake heat treatment at both squares.

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| phase 1.v.phase 2 | -0.627 | -0.373 |
| phase 2.v.phase 3 | -0.491 | -0.284 |
| phase 3.v.phase 4 | 0.685 | 0.942 |

The code chunk above returned results that indicate credible differences in the frequency of heat treatment between each phase except for phase one and two. The phase three to four transition is particularly different from the others, but this corresponds to only a 9% change increase in frequency of heated peices. all the HDIs are close to zero, indicating a small effect size.

The code chunk below computes a frequentist chi-square and Cramer's V test for the heat alteration data.

chert\_artefacts\_heat\_nhst <- assocstats(as.matrix(heat))

The code chunk above returns a chi-squared value of 30.0930868 and a p-value of 1.319214410^{-6}, indicating a signficant difference in frequencies of breakage classes by phase. However, the Cramer's V value of 0.0798054 indicates that the effect size is extremely small. We interpret this result to mean that although the test result is statistically significant, there is no substantial significance in the differences in frequencies of heat alteration by phase.

# Results: Metric and technological characteristics of cores and unretouched flakes

The code chunk below produces the table "summarizing the attributes of chert complete flakes from Jerimalai. Each cell contains mean ± standard deviation unless otherwise indicated. The table produced here has been rearranged by hand for the paper.

# the table has been rearranged by hand for the paper  
metrics <- flakes %>%   
  
 group\_by(phase) %>%  
  
 summarise(median(Length, na.rm = TRUE),   
 median(Width, na.rm = TRUE),   
 median(Thick, na.rm = TRUE),  
 median(Weight, na.rm = TRUE),  
 median(Length, na.rm = TRUE),  
 median(Platwid, na.rm = TRUE),  
 median(Platthic, na.rm = TRUE),  
 median(NoDS, na.rm = TRUE),  
 median(Cortex, na.rm = TRUE),  
 IQR(Length, na.rm = TRUE),   
 IQR(Width, na.rm = TRUE),   
 IQR(Thick, na.rm = TRUE),  
 IQR(Weight, na.rm = TRUE),  
 IQR(Length, na.rm = TRUE),  
 IQR(Platwid, na.rm = TRUE),  
 IQR(Platthic, na.rm = TRUE),  
 IQR(NoDS, na.rm = TRUE),  
 IQR(Cortex, na.rm = TRUE),  
 n = length(Weight))  
  
# get overhang removal data also  
ohr <- filter(flakes, Overhang == "Yes") %>%  
 group\_by(phase) %>%  
 summarise(OHR\_n = length(Overhang))  
# get percentages of OHR per phase  
ohr$OHR\_perc <- ohr$OHR\_n/metrics$n \* 100  
  
# combine  
metrics <- cbind(metrics, ohr[,c("OHR\_n", "OHR\_perc")])  
metrics <- t(round(metrics,2))  
# save as CSV to tidy up, there is no simple way to make the table  
# that appears in the paper  
write.csv(metrics, "figures/table\_6\_flake\_metrics.csv")  
# show a slightly untidy version  
metrics <- data.frame(metrics)  
names(metrics) <- c("phase 1", "phase 2", "phase 3", "phase 4")  
kable(metrics[-1,], caption = "Summary of attributes of chert complete flakes from Jerimalai both squares. Each cell contains median ± interquartile range unless otherwise indicated. ")

Summary of attributes of chert complete flakes from Jerimalai both squares. Each cell contains median ± interquartile range unless otherwise indicated.

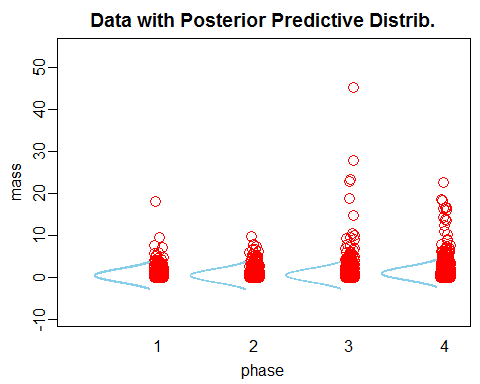
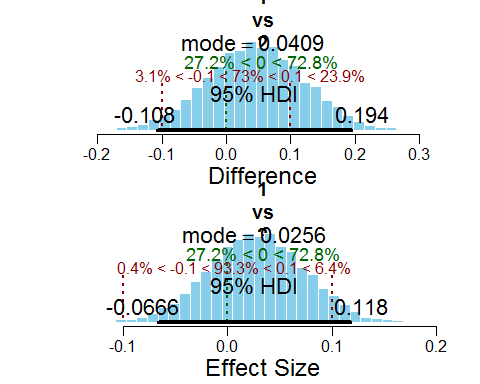
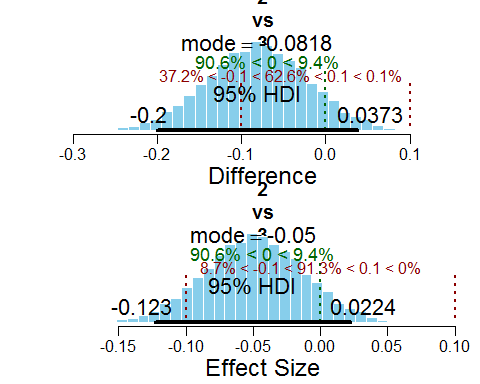
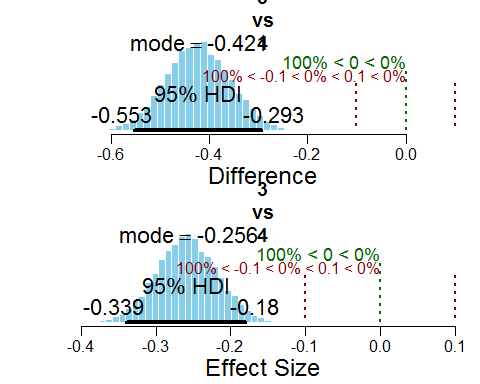
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | phase 1 | phase 2 | phase 3 | phase 4 |
| median(Length, na.rm = TRUE) | 8.44 | 8.71 | 9.04 | 10.55 |
| median(Width, na.rm = TRUE) | 7.92 | 8.33 | 8.45 | 9.62 |
| median(Thick, na.rm = TRUE) | 1.97 | 1.95 | 2.05 | 2.54 |
| median(Weight, na.rm = TRUE) | 0.16 | 0.16 | 0.17 | 0.31 |
| median(Platwid, na.rm = TRUE) | 5.77 | 5.58 | 5.95 | 7.24 |
| median(Platthic, na.rm = TRUE) | 1.94 | 1.76 | 1.95 | 2.54 |
| median(NoDS, na.rm = TRUE) | 4.00 | 4.00 | 4.00 | 4.00 |
| median(Cortex, na.rm = TRUE) | 0.00 | 0.00 | 0.00 | 0.00 |
| IQR(Length, na.rm = TRUE) | 6.97 | 6.97 | 6.16 | 8.56 |
| IQR(Width, na.rm = TRUE) | 5.87 | 6.32 | 6.20 | 7.54 |
| IQR(Thick, na.rm = TRUE) | 2.30 | 2.15 | 2.03 | 2.85 |
| IQR(Weight, na.rm = TRUE) | 0.46 | 0.50 | 0.44 | 0.85 |
| IQR(Platwid, na.rm = TRUE) | 4.91 | 4.62 | 5.23 | 5.68 |
| IQR(Platthic, na.rm = TRUE) | 2.08 | 1.77 | 1.81 | 2.71 |
| IQR(NoDS, na.rm = TRUE) | 2.00 | 3.00 | 2.00 | 2.00 |
| IQR(Cortex, na.rm = TRUE) | 0.00 | 0.00 | 0.00 | 0.00 |
| n | 728.00 | 1144.00 | 1907.00 | 946.00 |
| OHR\_n | 407.00 | 633.00 | 1136.00 | 553.00 |
| OHR\_perc | 55.91 | 55.33 | 59.57 | 58.46 |

The code chunk below uses a Bayesian one-way ANOVA to investigate differences in chert flake mass by depositional phase.

myDataFrame <- data.frame(phase = flakes$phase, mass = flakes$Weight)  
# remove flakes with no phase  
myDataFrame <- myDataFrame[!is.na(myDataFrame$phase),]  
yName = names(myDataFrame)[2] # mass  
xName = names(myDataFrame)[1] # phase  
contrasts = list(  
 list( c("1") , c("2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("2") , c("3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("3") , c("4") , compVal=0.0 , ROPE=c(-0.1,0.1)) )  
# Generate the MCMC chain:  
 mcmcCoda\_ANOVA = genMCMC\_ANOVA(datFrm=myDataFrame , yName=yName , xName=xName ,  
 numSavedSteps=11000 , thinSteps=10   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 22:54:18 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 9488  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 500  
-------------------------------------------------| 500  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 1000  
-------------------------------------------------| 1000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . Updating 36670  
-------------------------------------------------| 36650  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
\* 100%  
. . . . Updating 0  
. Deleting model  
.   
Following the progress of chain 3 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 22:54:19 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 9488  
. Reading parameter file inits3.txt  
. Initializing model  
. Adapting 500  
-------------------------------------------------| 500  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 1000  
-------------------------------------------------| 1000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . Updating 36670  
-------------------------------------------------| 36650  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

# Get summary statistics of chain:  
 summaryInfo\_ANOVA = smryMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
 datFrm=myDataFrame , xName=xName ,  
 contrasts=contrasts   
 )  
   
# Display posterior information (not easy to read unless using interactively):  
 plotMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
 datFrm=myDataFrame , yName=yName , xName=xName ,   
 contrasts=contrasts )

# subset summaryInfo to show HDI interval for interactions  
 HDI\_intervals\_for\_interactions <- summaryInfo\_ANOVA[c((nrow(summaryInfo\_ANOVA)-length(contrasts)+1):nrow(summaryInfo\_ANOVA)), c("HDIlow", "HDIhigh")]  
  
kable(round(HDI\_intervals\_for\_interactions,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and flake mass from both squares.")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and flake mass from both squares.

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| 1.v.2 | -0.108 | 0.194 |
| 2.v.3 | -0.200 | 0.037 |
| 3.v.4 | -0.553 | -0.293 |

The code chunk above produced results that indicate that only phase four and five have a credibly different distributions of flake mass. In the code chunk below we repeat the same investigation using a frequentist ANOVA.

# ANOVA with Tukey's HSD  
fit <- aov(Weight ~ as.factor(phase), flakes)  
fit\_summary <- summary(fit)  
tuk <- TukeyHSD(fit)  
# plot(tuk, las = 2, cex = 0.1)  
kable(round(tuk$`as.factor(phase)`,3), caption = "Tukey's Honest Significant Difference for phase by phase comparisons of flake mass both squares.")

Tukey's Honest Significant Difference for phase by phase comparisons of flake mass both squares.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | diff | lwr | upr | p adj |
| 2-1 | -0.046 | -0.245 | 0.153 | 0.934 |
| 3-1 | 0.038 | -0.145 | 0.221 | 0.951 |
| 4-1 | 0.473 | 0.266 | 0.680 | 0.000 |
| 3-2 | 0.084 | -0.073 | 0.241 | 0.516 |
| 4-2 | 0.519 | 0.334 | 0.704 | 0.000 |
| 4-3 | 0.435 | 0.268 | 0.602 | 0.000 |

In the code chunk above we obtain the results of a frequentist ANOVA which returns a statistically significant result: F = 21.2322309, df = 3, p = 1.171483310^{-13}. The Tukey's Honest Significant Difference test shows confidence intervals on the differences between the means artefact masses of a phase. When the interval includes zero, the difference is considered not significant. In this case we see that phase five differs from phase four, equivalent to what we see in the Bayesian HDI intervals.

The code chunk below produces a summary table of metric attributes of chert complete cores from Jerimalai square B.

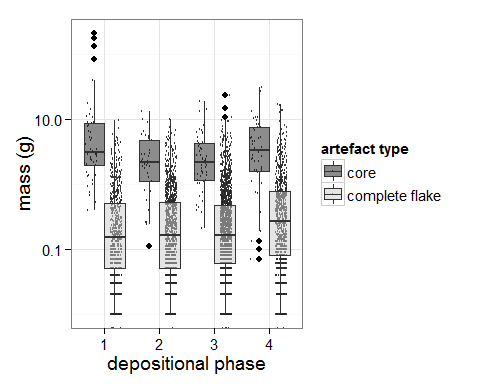
cores\_mass <- allchert[allchert$Artclas == "core", ]  
core\_metrics <- cores\_mass %>%   
 group\_by(phase) %>%   
 filter(phase %in% 1:4) %>%  
 summarise(median(Weight, na.rm = TRUE),   
 median(Length, na.rm = TRUE),   
 median(Width, na.rm = TRUE),   
 median(Thick, na.rm = TRUE),  
 median(as.numeric(cores\_mass$NoDS), na.rm = TRUE),  
 median(as.numeric(cores\_mass$Cortex), na.rm = TRUE),  
 median(as.numeric(cores\_mass$Corerot), na.rm = TRUE),  
 IQR(Weight, na.rm = TRUE),   
 IQR(Length, na.rm = TRUE),   
 IQR(Width, na.rm = TRUE),   
 IQR(Thick, na.rm = TRUE),  
 IQR(as.numeric(cores\_mass$NoDS), na.rm = TRUE),  
 IQR(as.numeric(cores\_mass$Cortex), na.rm = TRUE),  
 IQR(as.numeric(cores\_mass$Corerot), na.rm = TRUE),  
 n = length(Weight))  
core\_metrics\_t <- t(round(core\_metrics,2))  
colnames(core\_metrics\_t) <- c("phase 1", "phase 2", "phase 3", "phase 4" )  
kable(core\_metrics\_t, caption = "Summary of chert core metrics for both squares")

Summary of chert core metrics for both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | phase 1 | phase 2 | phase 3 | phase 4 |
| phase | 1.00 | 2.00 | 3.00 | 4.00 |
| median(Weight, na.rm = TRUE) | 3.05 | 2.12 | 2.17 | 3.29 |
| median(Length, na.rm = TRUE) | 22.63 | 18.72 | 17.36 | 18.38 |
| median(Width, na.rm = TRUE) | 14.80 | 14.78 | 13.44 | 14.26 |
| median(Thick, na.rm = TRUE) | 9.70 | 7.33 | 8.52 | 9.54 |
| median(as.numeric(cores\_mass$NoDS), n... | 11.00 | 11.00 | 11.00 | 11.00 |
| median(as.numeric(cores\_mass$Cortex),... | 0.00 | 0.00 | 0.00 | 0.00 |
| median(as.numeric(cores\_mass$Corerot)... | 3.00 | 3.00 | 3.00 | 3.00 |
| IQR(Weight, na.rm = TRUE) | 6.51 | 3.57 | 3.03 | 5.80 |
| IQR(Length, na.rm = TRUE) | 13.43 | 7.61 | 7.91 | 12.66 |
| IQR(Width, na.rm = TRUE) | 9.48 | 7.53 | 6.24 | 8.51 |
| IQR(Thick, na.rm = TRUE) | 7.20 | 5.61 | 4.50 | 4.09 |
| IQR(as.numeric(cores\_mass$NoDS), na.r... | 6.00 | 6.00 | 6.00 | 6.00 |
| IQR(as.numeric(cores\_mass$Cortex), na... | 10.00 | 10.00 | 10.00 | 10.00 |
| IQR(as.numeric(cores\_mass$Corerot), n... | 1.00 | 1.00 | 1.00 | 1.00 |
| n | 50.00 | 37.00 | 56.00 | 46.00 |

The code chunk below creates a figure that shows the distribution of core and complete flake mass by depositional phases

# plot flake and core mass by phase in one box plot  
flakes\_cores\_weight <- allchert %>%  
 filter(Artclas %in% c("flake", "core")) %>%  
 filter(phase %in% 1:4) %>%  
 select(Artclas, Weight, phase)  
   
  
ggplot(flakes\_cores\_weight, aes(fill = Artclas, as.factor(phase), Weight)) +  
 geom\_point(aes(colour = Artclas), size = 0.5, alpha = 0.9, shape = 1,   
 position=position\_jitterdodge(dodge.width=0.9)) +  
 geom\_boxplot(alpha = 0.5) +  
 scale\_y\_log10() +  
 #theme\_minimal(base\_size = 4) +  
 xlab('depositional phase') +  
 ylab("mass (g)") +  
 scale\_fill\_manual(name="artefact type",  
 labels=c("core", "complete flake"),   
 values = c("grey10", "grey80")) +  
 scale\_colour\_manual(name="artefact type",  
 labels=c("core", "complete flake"),   
 values = c("grey10", "grey15"))



# save plot  
jhe\_190mm\_ggsave("figures/fig\_6\_Jeremalai-flake-core-mass-phase.png",   
 height = (190/2))

The code chunk below uses a Bayesian one-way ANOVA to investigate differences in chert core mass by depositional phase.

myDataFrame <- data.frame(phase = cores\_mass$phase, mass = cores\_mass$Weight)  
# remove flakes with no phase  
myDataFrame <- myDataFrame[!is.na(myDataFrame$phase),]  
yName = names(myDataFrame)[2] # mass  
xName = names(myDataFrame)[1] # phase  
contrasts = list(  
 list( c("1") , c("2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("2") , c("3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("3") , c("4") , compVal=0.0 , ROPE=c(-0.1,0.1)) )  
# Generate the MCMC chain:  
 mcmcCoda\_ANOVA = genMCMC\_ANOVA(datFrm=myDataFrame , yName=yName , xName=xName ,  
 numSavedSteps=11000 , thinSteps=10   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 22:57:44 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 416  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 500  
-------------------------------------------------| 500  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 1000  
-------------------------------------------------| 1000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . Updating 36670  
-------------------------------------------------| 36650  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

# Get summary statistics of chain:  
 summaryInfo\_ANOVA = smryMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
 datFrm=myDataFrame , xName=xName ,  
 contrasts=contrasts   
 )  
   
# # Display posterior information (not easy to read unless using interactively):  
# plotMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
# datFrm=myDataFrame , yName=yName , xName=xName ,   
# contrasts=contrasts )  
#   
# subset summaryInfo to show HDI interval for interactions  
 HDI\_intervals\_for\_interactions <- summaryInfo\_ANOVA[c((nrow(summaryInfo\_ANOVA)-length(contrasts)+1):nrow(summaryInfo\_ANOVA)), c("HDIlow", "HDIhigh")]  
kable(round(HDI\_intervals\_for\_interactions,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and core mass, both squares.")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and core mass, both squares.

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| 1.v.2 | 1.034 | 20.519 |
| 2.v.3 | -8.236 | 8.658 |
| 3.v.4 | -9.676 | 6.440 |

The code chunk above produced results that indicate that the HDIs for all phases include zero, which we interpret as no credible difference. In the code chunk below we repeat the same investigation using a frequentist ANOVA.

# ANOVA with Tukey's HSD  
fit <- aov(mass ~ as.factor(phase), myDataFrame)  
fit\_summary <- summary(fit)  
tuk <- TukeyHSD(fit)  
# plot(tuk, las = 2, cex = 0.1)  
kable(round(tuk$`as.factor(phase)`,3), caption = "Tukey's Honest Significant Difference for phase by phase comparisons of core mass, both squares.")

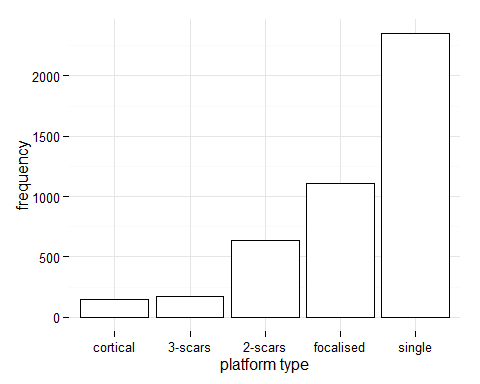
Tukey's Honest Significant Difference for phase by phase comparisons of core mass, both squares.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | diff | lwr | upr | p adj |
| 2-1 | -13.586 | -26.021 | -1.150 | 0.026 |
| 3-1 | -13.420 | -24.577 | -2.263 | 0.011 |
| 4-1 | -11.279 | -22.995 | 0.436 | 0.064 |
| 3-2 | 0.166 | -11.983 | 12.315 | 1.000 |
| 4-2 | 2.307 | -10.356 | 14.970 | 0.965 |
| 4-3 | 2.141 | -9.270 | 13.552 | 0.962 |

In the code chunk above we obtain the results of a frequentist ANOVA which returns a statistically significant result: F = 4.1790192, df = 3, p = 0.006845. The Tukey's Honest Significant Difference test shows confidence intervals on the differences between the means artefact masses of a phase. In this case we see that only the comparison of phase four to phase two has a significant difference in core mass. Consequtive phases show no signifance difference, which we interpret as evidence of overall no substantial change in core mass.

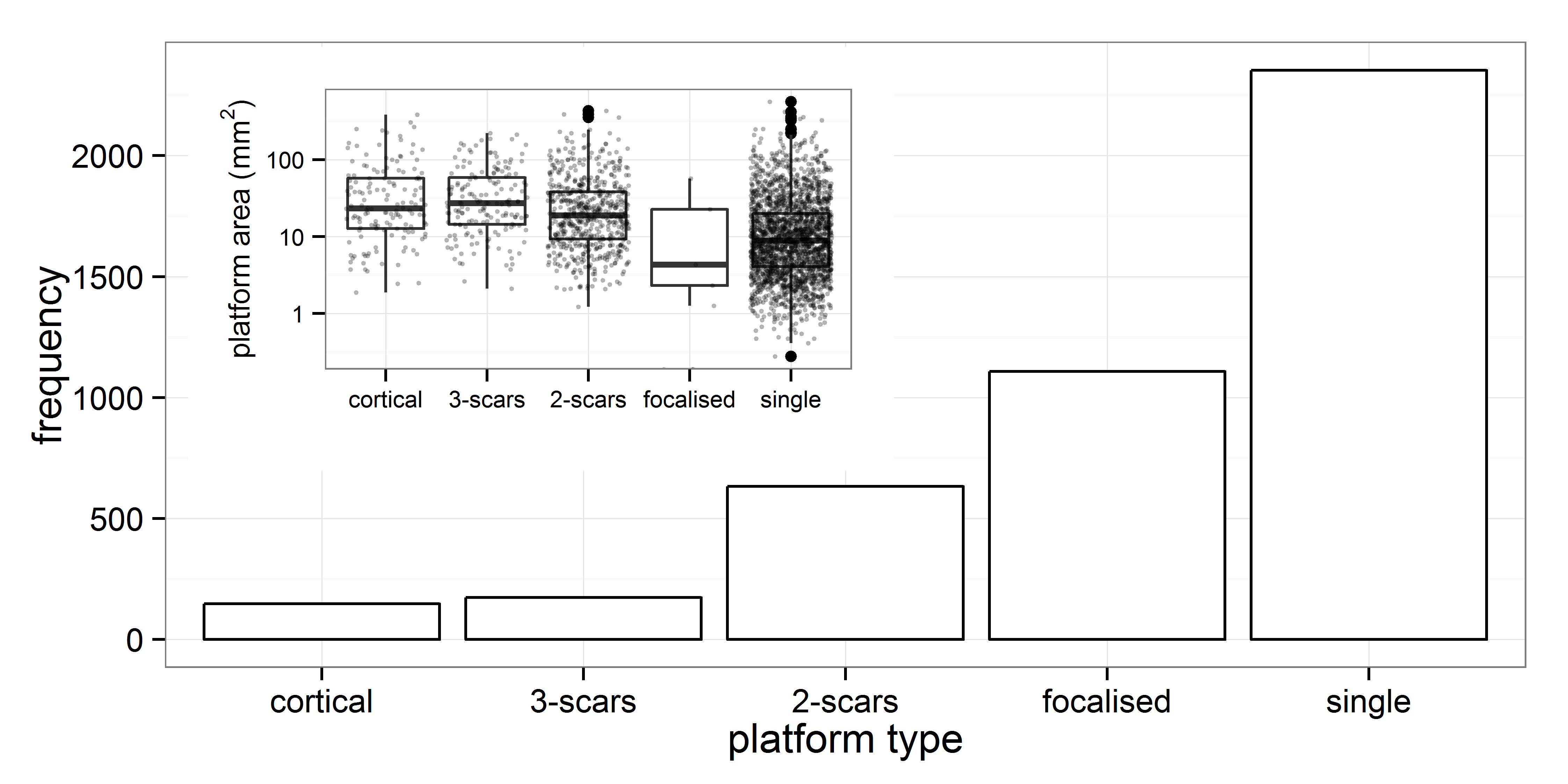
The code chunk below produces the figures that illustrate the frequency of flake platform categories for chert complete flakes at Jerimalai square B.

# flake platform  
plat <- dcast(flakes, Plat ~ depth)   
rownames(plat) <- plat[,1]  
# get rid of rows with no plat  
plat <- plat[rownames(plat) != "",]  
plat <- plat[,-1]  
plat\_freqs <- data.frame(plat\_types = rownames(plat), Freq = rowSums(plat))  
plat\_freqs <- plat\_freqs[plat\_freqs$Freq > 90,]  
# what is 'coll'? Dropping it  
plat\_freqs <- plat\_freqs[plat\_freqs$plat\_types != 'Coll', ]  
plat\_freqs$plat\_types <- c('2-scars', '3-scars', 'cortical', 'focalised', 'single')  
# plot freq of platform types  
main <- ggplot(plat\_freqs, aes(reorder(plat\_types, Freq), Freq)) +   
 geom\_bar(stat="identity", fill = "white", colour = "black") +  
 # theme\_minimal() +  
 xlab("platform type") +  
 ylab("frequency") +  
 # remove grid lines for subplot  
 theme\_update(panel.background = element\_blank(),  
 panel.grid.major = element\_blank(),  
 panel.grid.minor = element\_blank())  
main + theme\_minimal()



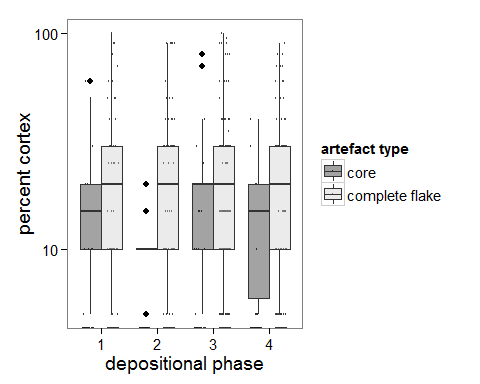
# raw materials by spit  
# compute proportions per layer (col props)  
all\_tab <- data.frame()  
for(i in seq(ncol(plat))){  
 for(j in seq(nrow(plat))){  
 all\_tab[j,i] <- plat[j,i]/colSums(plat)[i]  
 }  
}  
# check   
check <- colSums(all\_tab) # should == 1  
colnames(all\_tab) <- colnames(plat)   
check <- rowSums(all\_tab) # should be various  
all\_tab$plat\_type <- rownames(plat)   
# get rid of raw materials that are not very abundant  
all\_tab <- all\_tab[which(rowSums(all\_tab[,1:ncol(all\_tab)-1]) > 1.5) , ]  
# get rid of NA column  
all\_tab <- all\_tab[,names(all\_tab) != 'NA']  
# plot  
all\_tab\_m <- melt(all\_tab, id.var = 'plat\_type')  
# by phase  
plat <- dcast(flakes, Plat ~ phase)  
rownames(plat) <- plat[,1]  
# get rid of rows with no plat  
plat <- plat[rownames(plat) != "",]  
plat <- plat[,-1]  
# raw materials by site  
# compute proportions per phase (col props)  
all\_tab <- data.frame()  
for(i in seq(ncol(plat))){  
 for(j in seq(nrow(plat))){  
 all\_tab[j,i] <- plat[j,i]/colSums(plat)[i]  
 }  
}  
all\_tab$plat\_type <- rownames(plat)   
# get rid of raw materials that are not very abundant  
all\_tab <- all\_tab[which(rowSums(all\_tab[,1:ncol(all\_tab)-1]) > 0.15) , ]  
# get rid of NA column  
all\_tab <- all\_tab[,names(all\_tab) != 'NA']  
# plot distibution of platform sizes for each type  
flakes$Platarea <- with(flakes, (Platthic \* Platwid))  
plat\_area\_type <- flakes[flakes$Plat %in% c("Single", "Focal", "2-scars", "Cort", '3-scars'),]  
  
# make names a bit more readable  
plat\_area\_type$Plat <- ifelse(plat\_area\_type$Plat == 'Cort', 'cortical',  
 ifelse(plat\_area\_type$Plat == 'Focal', 'focalised',   
 ifelse(plat\_area\_type$Plat == 'Single', 'single', as.character(plat\_area\_type$Plat))))   
  
# put types in same order as frequency plot  
plat\_area\_type$Plat <- factor(plat\_area\_type$Plat,   
 levels = c('cortical', '3-scars','2-scars', 'focalised', 'single'), ordered = TRUE)  
   
sub <- ggplot(plat\_area\_type, aes(Plat, Platarea)) +  
 geom\_boxplot() +  
 geom\_jitter(size = 0.5, alpha = 0.3, shape = 1) +  
 scale\_y\_log10() +  
 ylab(as.expression(bquote('platform area (' \* mm^{2} \* ")" ))) +  
 xlab("") +  
 theme\_minimal()  
  
# plot freq of plat type and platform area together in one plot  
   
vp <- viewport(width = 0.45, height = 0.54,   
 x = 0.57, y = 0.4,   
 just = c("right", "bottom"))  
  
# combine plots, print and save (wont show in console)  
 png("figures/fig\_7\_Jeremalai-platform-area-by-plat-type.png",   
 units = "mm", w = 190, h = 190/2, res = 600)  
 print(main)  
 print(sub + theme\_bw(base\_size = 10), vp = vp)  
 dev.off()

png   
 2



The code chunk below creates a plot showing the distributions of core and flake cortex by depositional phase at both squares of Jerimalai.

flakes\_cores\_cortex <- allchert %>%  
 filter(Artclas %in% c("flake", "core")) %>%  
 filter(phase %in% 1:4) %>%  
 select(Artclas, Cortex, phase)  
  
ggplot(flakes\_cores\_cortex, aes( fill = Artclas, as.factor(phase), Cortex)) +  
 geom\_point(aes(colour = Artclas), size = 0.5, alpha = 0.9, shape = 1,   
 position=position\_jitterdodge(dodge.width=0.9)) +  
 geom\_boxplot(alpha = 0.4) +  
 scale\_y\_log10() +  
 xlab('depositional phase') +  
 ylab("percent cortex") +  
 scale\_fill\_manual(name="artefact type",  
 labels=c("core", "complete flake"),   
 values = c("grey10", "grey80")) +  
 scale\_colour\_manual(name="artefact type",  
 labels=c("core", "complete flake"),   
 values = c("grey10", "grey20"))



# save plot  
jhe\_190mm\_ggsave("figures/fig\_8\_Jeremalai-flake-core-cortex-phase.png", height = 190/2)

The code chunk below uses a Bayesian one-way ANOVA to investigate differences in chert flake cortex by depositional phase.

myDataFrame <- data.frame(phase = flakes$phase, cortex = flakes$Cortex)  
# remove flakes with no phase  
myDataFrame <- myDataFrame[!is.na(myDataFrame$phase),]  
myDataFrame <- myDataFrame[!is.na(myDataFrame$cortex),]  
# replace zeros with ones, since mode can't be zero here  
# myDataFrame$cortex <- ifelse(myDataFrame$cortex == 0, 1, myDataFrame$cortex)  
yName = names(myDataFrame)[2] # cortex  
xName = names(myDataFrame)[1] # phase  
contrasts = list(  
 list( c("1") , c("2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("2") , c("3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("3") , c("4") , compVal=0.0 , ROPE=c(-0.1,0.1)) )  
# Generate the MCMC chain:  
 mcmcCoda\_ANOVA = genMCMC\_ANOVA(datFrm=myDataFrame , yName=yName , xName=xName , numSavedSteps=11000 , thinSteps=10   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 22:58:10 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 9484  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 500  
-------------------------------------------------| 500  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 1000  
-------------------------------------------------| 1000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . Updating 36670  
-------------------------------------------------| 36650  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

summaryInfo\_ANOVA = smryMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
 datFrm=myDataFrame , xName=xName ,  
 contrasts=contrasts   
 )  
 # # Display posterior information (not easy to read unless using interactively):  
# plotMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
# datFrm=myDataFrame , yName=yName , xName=xName ,   
# contrasts=contrasts )  
#   
# subset summaryInfo to show HDI interval for interactions  
HDI\_intervals\_for\_interactions\_f <- summaryInfo\_ANOVA[c((nrow(summaryInfo\_ANOVA)-length(contrasts)+1):nrow(summaryInfo\_ANOVA)), c("HDIlow", "HDIhigh")]  
kable(round(HDI\_intervals\_for\_interactions\_f,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and flake cortex at both squares")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and flake cortex at both squares

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| 1.v.2 | -0.413 | 1.714 |
| 2.v.3 | -1.111 | 0.588 |
| 3.v.4 | -2.348 | -0.368 |

The code chunk above produced results that indicate that the HDIs for phases 3 to 4 exclude zero, which we interpret as credible difference for this transition. In the code chunk below we repeat the same investigation using a frequentist ANOVA.

# ANOVA with Tukey's HSD  
fit <- aov(cortex ~ as.factor(phase), myDataFrame)  
fit\_summary <- summary(fit)  
tuk <- TukeyHSD(fit)  
# plot(tuk, las = 2, cex = 0.1)  
kable(round(tuk$`as.factor(phase)`,3), caption = "Tukey's Honest Significant Difference for phase by phase comparisons of flake cortex, both squares")

Tukey's Honest Significant Difference for phase by phase comparisons of flake cortex, both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | diff | lwr | upr | p adj |
| 2-1 | -0.772 | -2.253 | 0.710 | 0.538 |
| 3-1 | -0.433 | -1.794 | 0.929 | 0.847 |
| 4-1 | 1.154 | -0.387 | 2.694 | 0.218 |
| 3-2 | 0.339 | -0.830 | 1.508 | 0.879 |
| 4-2 | 1.925 | 0.552 | 3.298 | 0.002 |
| 4-3 | 1.586 | 0.344 | 2.829 | 0.006 |

In the code chunk above we obtain the results of a frequentist ANOVA which returns a statistically nonsignificant result: F = 4.9984974, df = 3, p = 0.0018396. The Tukey's Honest Significant Difference test shows confidence intervals on the differences between the means artefact masses of a phase. We see the phase 3-4 interaction showing a significant difference.

In the two code chunks below we repeat the statistical tests above for core cortex.

myDataFrame <- data.frame(phase = cores\_mass$phase, cortex = cores\_mass$Cortex)  
myDataFrame <- myDataFrame[complete.cases(myDataFrame),] # omit NAs  
yName = names(myDataFrame)[2] # cortex  
xName = names(myDataFrame)[1] # phase  
contrasts = list(  
 list( c("1") , c("2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("2") , c("3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("3") , c("4") , compVal=0.0 , ROPE=c(-0.1,0.1))   
 )  
# Generate the MCMC chain:  
 mcmcCoda\_ANOVA = genMCMC\_ANOVA(datFrm=myDataFrame , yName=yName , xName=xName ,  
 numSavedSteps=11000 , thinSteps=10   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 23:01:54 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 412  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 500  
-------------------------------------------------| 500  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 1000  
-------------------------------------------------| 1000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . Updating 36670  
-------------------------------------------------| 36650  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

# Get summary statistics of chain:  
 summaryInfo\_ANOVA = smryMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
 datFrm=myDataFrame , xName=xName ,  
 contrasts=contrasts   
 )  
 # # Display posterior information (not easy to read unless using interactively):  
# plotMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
# datFrm=myDataFrame , yName=yName , xName=xName ,   
# contrasts=contrasts )  
#   
# subset summaryInfo to show HDI interval for interactions  
 HDI\_intervals\_for\_interactions\_c <- summaryInfo\_ANOVA[c((nrow(summaryInfo\_ANOVA)-length(contrasts)+1):nrow(summaryInfo\_ANOVA)), c("HDIlow", "HDIhigh")]  
kable(round(HDI\_intervals\_for\_interactions\_c,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and core cortex at both squares")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and core cortex at both squares

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| 1.v.2 | -1.902 | 6.755 |
| 2.v.3 | -8.598 | 0.690 |
| 3.v.4 | -0.802 | 7.872 |

# combine core and flake HDIs into one table  
HDIs\_for\_cores\_and\_flakes <- round(cbind(HDI\_intervals\_for\_interactions\_c, HDI\_intervals\_for\_interactions\_f),3)  
kable(HDIs\_for\_cores\_and\_flakes, caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and core and flake cortex, both squares")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and core and flake cortex, both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | HDIlow | HDIhigh | HDIlow | HDIhigh |
| 1.v.2 | -1.902 | 6.755 | -0.413 | 1.714 |
| 2.v.3 | -8.598 | 0.690 | -1.111 | 0.588 |
| 3.v.4 | -0.802 | 7.872 | -2.348 | -0.368 |

The code chunk above produced results that indicate that the HDIs for all phases include zero, which we interpret as no credible difference. In the code chunk below we repeat the same investigation using a frequentist ANOVA.

# ANOVA with Tukey's HSD  
fit <- aov(cortex ~ as.factor(phase), myDataFrame)  
fit\_summary <- summary(fit)  
tuk <- TukeyHSD(fit)  
# plot(tuk, las = 2, cex = 0.1)  
kable(round(tuk$`as.factor(phase)`,3), caption = "Tukey's Honest Significant Difference for phase by phase comparisons of flake cortex, both squares")

Tukey's Honest Significant Difference for phase by phase comparisons of flake cortex, both squares

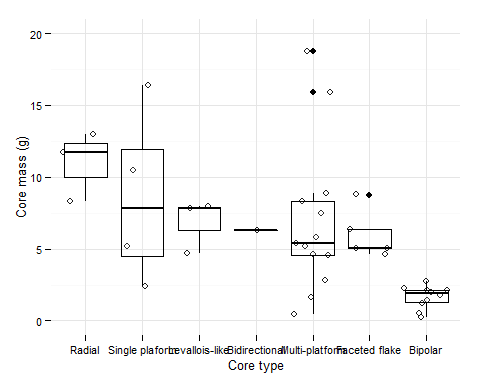
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | diff | lwr | upr | p adj |
| 2-1 | -3.262 | -9.944 | 3.420 | 0.586 |
| 3-1 | 2.382 | -3.613 | 8.377 | 0.732 |
| 4-1 | -2.350 | -8.719 | 4.019 | 0.774 |
| 3-2 | 5.644 | -0.883 | 12.172 | 0.116 |
| 4-2 | 0.912 | -5.961 | 7.785 | 0.986 |
| 4-3 | -4.732 | -10.939 | 1.475 | 0.201 |

In the code chunk above we obtain the results of a frequentist ANOVA which returns a statistically nonsignificant result: F = 2.1491321, df = 3, p = 0.0956167. The Tukey's Honest Significant Difference test shows confidence intervals on the differences between the means artefact masses of a phase. None of the interactions show a significant difference. We interpret this as evidence of overall no change in core cortex.

# Results: Core technology

This code draws a plot of differences in the mass of cores recovered from Jerimalai by core type. There are 42 cores in the assemblage.

both\_sqs\_core\_types <- core\_types   
both\_sqs\_core\_types$Type\_long <- with(both\_sqs\_core\_types,   
 ifelse(Type == "SPC", "Single plaform",  
 ifelse(Type == "RC", "Radial",  
 ifelse(Type == "BDC", "Bidirectional",  
 ifelse(Type == "BiC", "Bipolar",  
 ifelse(Type == "MPC", "Multi-platform",  
 ifelse(Type == "LLC", "Levallois-like",  
 ifelse(Type == "FFC", "Faceted flake", NA))))))) )  
# plot  
ggplot(both\_sqs\_core\_types, aes(reorder(Type\_long, -Mass, FUN=median), Mass)) +  
 geom\_jitter(alpha = 0.9, shape = 1) +   
 geom\_boxplot(alpha = 0.1, fill = "white", colour = "black") +  
 ylim(0,20) +  
 xlab("Core type") +  
 ylab("Core mass (g)") +  
 theme\_minimal(base\_size = 10)



# save  
jhe\_190mm\_ggsave("figures/fig\_9\_Jeremalai-core-by-type.png", height = 190/1.6)

The code chunk below computes the amount of cortex for each core type.

core\_cortex <- aggregate( X..Cortex ~ Type, data = core\_types, mean)  
# what is the average amount of cortex for each core type?  
core\_cortex\_means <- arrange(core\_cortex, -X..Cortex)  
names(core\_cortex\_means) <- c("Type", "Cortex percentage")  
core\_cortex\_means[,2] <- round(core\_cortex\_means[,2],1)

Consistent with overall size, single platform cores retain the most cortex on average (20%), followed by radial cores (10%), multiplatform cores (6%) and faceted flake cores (4%). Levallois-like cores and bipolar cores exhibit the least cortex (<4%).

The code chunk below computes the numbers of flake scars by each core type.

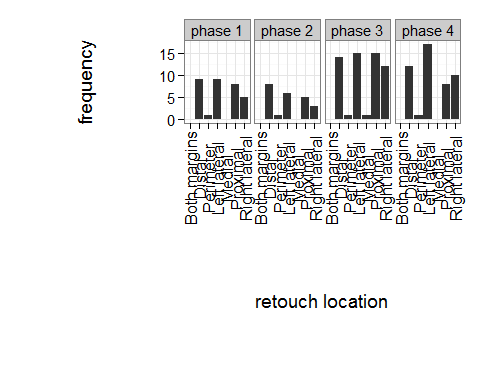
core\_scars <- core\_types %>%  
 group\_by(Type) %>%  
 summarize(means = round(mean(Number.of.Scars, na.rm = TRUE),0),  
 sds = round(sd(Number.of.Scars, na.rm = TRUE)),0) %>%  
 arrange(means)

Levallois-like cores exhibit almost twice the number of flake scars on average as other cores in the assemblage (mean = 27±9 scars versus 15±5 on average). Single platform cores have the least scars on average (10±3).

# Results: Retouched artefacts

The code chunk below draws a plot showing locations of retouch on chert flakes by depositional unit in both squares of Jerimalai.

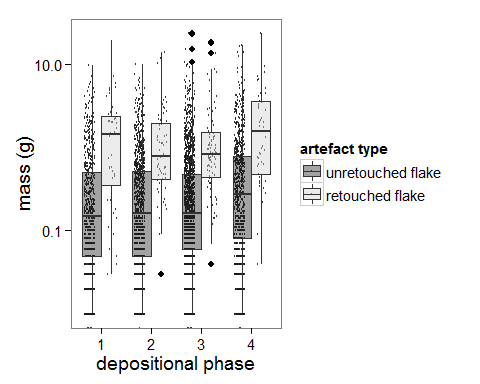
# get sq B retouch  
rt\_flakes <- read.csv("data/JB\_Chert\_Flakes\_and\_Retouch.csv")  
rt\_flakes$phase <- ifelse(rt\_flakes$Square == "A",   
 sqA\_makephases(rt\_flakes$Spit),   
 sqB\_makephases(rt\_flakes$Spit))  
  
rt\_flakes <- rt\_flakes[!is.na(rt\_flakes$phase), ]  
# frequency of flakes with retouch per phase  
rt <- flakes[flakes$Rtch == "Yes" , ] # sq A retouch also here  
rt\_ <- rt\_flakes[rt\_flakes$Rtch == "Yes" , ]  
common\_names <- intersect(names(rt\_), names(rt))   
rt\_ <- rt\_ %>% select\_(.dots = common\_names)  
rt <- rt %>% select\_(.dots = common\_names)  
rt <- rbind(rt , rt\_)  
# what is this for?  
rt\_tab <- t(data.frame(rt = aggregate(Retype ~ phase, rt,   
 function(x) rt\_count = length(x)),   
 flakes = aggregate(Retype ~ phase, flakes,   
 function(x) fk\_count = length(x))[,2] ))  
  
# retouch locations  
rt\_loc <- data.frame(phase = rt$phase, rt\_loc = rt$Retloc)  
rt\_loc <- rt\_loc[!(is.na(rt\_loc$phase)), ]  
rt\_sum <- as.data.frame.matrix(table(rt\_loc))[,-1]  
rt\_sum$phase <- paste0("phase ", row.names(rt\_sum))  
rt\_sum <- rt\_sum[,-which(names(rt\_sum) == "Bothlats")] # not in both squares  
colnames(rt\_sum) <- c("Both margins", "Distal", "Perimeter", "Left lateral", "Medial", "Proximal", "Right lateral", "phase")  
rt\_sum\_m <- melt(rt\_sum)  
# plot  
ggplot(rt\_sum\_m, aes(variable, value)) +  
 geom\_bar(stat="identity") +  
 theme(axis.text.x = element\_text(angle=90, vjust=0.5, size=12)) +  
 ylab("frequency") +  
 xlab("retouch location") +  
 facet\_wrap(~phase, ncol = 5)



# save plot  
jhe\_190mm\_ggsave("figures/fig\_15\_Jeremalai-flake-retouched-flake-location-phase.png", height = 190/1.6)

The code chunk below draws a plot of the distribution of flake lengths for retouched and unretouched flakes by depositional unit for bohth squares at Jerimalai.

flakes\_retouch\_size <- allchert %>%  
 filter(Artclas %in% c("flake", "retf")) %>%  
 filter(phase %in% 1:4) %>%  
 select(Artclas, Weight, Length, phase)  
  
ggplot(flakes\_retouch\_size, aes( fill = Artclas, as.factor(phase), Weight)) +  
 geom\_point(aes(colour = Artclas), size = 0.5, alpha = 0.9, shape = 1,   
 position=position\_jitterdodge(dodge.width=0.9)) +   
 geom\_boxplot(alpha = 0.4) +  
 scale\_y\_log10() +  
 xlab('depositional phase') +  
 ylab("mass (g)") +  
 scale\_fill\_manual(name="artefact type",  
 labels=c("unretouched flake", "retouched flake"),   
 values = c("grey10", "grey80")) +  
 scale\_colour\_manual(name="artefact type",  
 labels=c("unretouched flake", "retouched flake"),   
 values = c("grey10", "grey15"))



# save plot  
jhe\_190mm\_ggsave("figures/fig\_16\_Jeremalai-flake-retouchedflake-mass-phase.png", height = 190/1.6)

The code chunk below computes the tests for credible difference in retouch flake frequency by phase.

rt\_tab <- data.frame(rt\_tab)  
colnames(rt\_tab) <- c("phase 1", "phase 2", "phase 3", "phase 4")  
rt\_tab <- rt\_tab[-1,]  
myDataFrame <- melt(as.matrix(rt\_tab), varnames=c("retouch", "phase"), value.name="Freq")  
myDataFrame <- myDataFrame[complete.cases(myDataFrame),] # omit NAs  
yName=names(myDataFrame)[3] # Freq   
x1Name=names(myDataFrame)[2] # phase  
x2Name=names(myDataFrame)[1] # retouch  
x1contrasts = list(   
 list( c("phase 1") , c("phase 2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("phase 2") , c("phase 3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("phase 3") , c("phase 4") , compVal=0.0 , ROPE=c(-0.1,0.1))   
 )  
numSavedSteps = 12000 # MCMC parameters  
thinSteps = 10 # MCMC parameters  
  
mcmcCoda = genMCMC\_cont\_table( datFrm=myDataFrame ,   
 yName=yName , x1Name=x1Name , x2Name=x2Name ,  
 numSavedSteps=numSavedSteps , thinSteps=thinSteps ,   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 23:02:16 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 136  
  
WARNING: Unused variable(s) in data table:  
ySum  
  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 1000  
-------------------------------------------------| 1000  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 2000  
-------------------------------------------------| 2000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . . . . . . . Failed to set trace monitor for delta  
Node not found  
. Updating 40000  
-------------------------------------------------| 40000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

# Get summary statistics of chain:  
summaryInfo = smryMCMC\_cont\_table( mcmcCoda ,   
 datFrm=myDataFrame , x1Name=x1Name , x2Name=x2Name ,  
 x1contrasts=x1contrasts  
 )  
  
# # # Display posterior information (not easy to read unless using interactively):  
# plotMCMC\_cont\_table( mcmcCoda ,   
# datFrm=myDataFrame , yName=yName , x1Name=x1Name ,   
# x2Name=x2Name ,  
# x1contrasts=x1contrasts )  
  
# table of retouch artefact counts  
kable(rt\_tab, caption = "Table of frequencies of retouched flakes by phase, both squares")

Table of frequencies of retouched flakes by phase, both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | phase 1 | phase 2 | phase 3 | phase 4 |
| rt.Retype | 32 | 24 | 61 | 54 |
| flakes | 728 | 1144 | 1907 | 946 |

# subset summaryInfo to show HDI interval for interactions  
 HDI\_intervals\_for\_interactions\_fr <- summaryInfo\_ANOVA[c((nrow(summaryInfo\_ANOVA)-length(contrasts)+1):nrow(summaryInfo\_ANOVA)), c("HDIlow", "HDIhigh")]  
kable(round(HDI\_intervals\_for\_interactions\_fr,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and retouched flake frequency, both squares.")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and retouched flake frequency, both squares.

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| 1.v.2 | -1.902 | 6.755 |
| 2.v.3 | -8.598 | 0.690 |
| 3.v.4 | -0.802 | 7.872 |

The results of the code chunk above indicates that there are no credible differences in the frequencies of retouched artefacts. A look at the raw counts shows that from phase three to four there is a decrease in the proportion of retouched pieces, but that the overall count is still low.

chert\_artefacts\_retouch\_nhst <- assocstats(as.matrix(rt\_tab))

The code chunk above returns a chi-squared value of 20.021136 and a p-value of 1.6803910^{-4}, indicating a nonsignficant difference in frequencies of breakage classes by phase. The Cramer's V value of 0.0639475 indicates that the effect size is extremely small. We interpret this result to mean that there is no substantial significance in the differences in frequencies of retouched flakes by phase.

The code chunk below investigates changes in the length of the retouched margin of artefacts over time.

rt\_len <- data.frame(phase = rt$phase, rt\_len = rt$Retlen)  
# omit NAs  
rt\_len <- rt\_len[!(is.na(rt\_len$rt\_len)), ]  
# get mean lengths  
mean\_lengths <- aggregate(rt\_len ~ phase, rt\_len, mean)  
# do bayesian ANOVA  
myDataFrame <- rt\_len  
myDataFrame <- myDataFrame[complete.cases(myDataFrame),] # omit NAs  
yName = names(myDataFrame)[2] # retouch length  
xName = names(myDataFrame)[1] # phase  
contrasts = list(  
 list( c("1") , c("2") , compVal=0.0 , ROPE=c(-0.1,0.1) ) ,  
 list( c("2") , c("3") , compVal=0.0 , ROPE=c(-0.1,0.1)) ,  
 list( c("3") , c("4") , compVal=0.0 , ROPE=c(-0.1,0.1)) )  
# Generate the MCMC chain:  
 mcmcCoda\_ANOVA = genMCMC\_ANOVA(datFrm=myDataFrame , yName=yName , xName=xName ,  
 numSavedSteps=11000 , thinSteps=10   
 )

Calling 3 simulations using the parallel method...  
Following the progress of chain 1 (the program will wait for all  
chains to finish before continuing):  
Welcome to JAGS 3.4.0 on Tue Nov 10 23:02:29 2015  
JAGS is free software and comes with ABSOLUTELY NO WARRANTY  
Loading module: basemod: ok  
Loading module: bugs: ok  
. . Reading data file data.txt  
. Compiling model graph  
 Resolving undeclared variables  
 Allocating nodes  
 Graph Size: 216  
. Reading parameter file inits1.txt  
. Initializing model  
. Adapting 500  
-------------------------------------------------| 500  
++++++++++++++++++++++++++++++++++++++++++++++++++ 100%  
Adaptation successful  
. Updating 1000  
-------------------------------------------------| 1000  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
. . . . . . Updating 36670  
-------------------------------------------------| 36650  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100%  
\* 100%  
. . . . Updating 0  
. Deleting model  
.   
All chains have finished  
Simulation complete. Reading coda files...  
Coda files loaded successfully  
Finished running the simulation

# Get summary statistics of chain:  
 summaryInfo\_ANOVA = smryMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
 datFrm=myDataFrame , xName=xName ,  
 contrasts=contrasts   
 )  
 # # Display posterior information (not easy to read unless using interactively):  
# plotMCMC\_ANOVA( mcmcCoda\_ANOVA ,   
# datFrm=myDataFrame , yName=yName , xName=xName ,   
# contrasts=contrasts )  
#   
# subset summaryInfo to show HDI interval for interactions  
 HDI\_intervals\_for\_interactions\_l <- summaryInfo\_ANOVA[c((nrow(summaryInfo\_ANOVA)-length(contrasts)+1):nrow(summaryInfo\_ANOVA)), c("HDIlow", "HDIhigh")]  
kable(round(HDI\_intervals\_for\_interactions\_l ,3), caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and retouch length, both squares")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and retouch length, both squares

|  |  |  |
| --- | --- | --- |
|  | HDIlow | HDIhigh |
| 1.v.2 | -2.624 | 6.737 |
| 2.v.3 | -8.481 | 2.211 |
| 3.v.4 | -3.785 | 6.370 |

# combine retouch freq and length into one table  
HDIs\_flake\_retouch\_freq\_and\_length <- round(cbind(HDI\_intervals\_for\_interactions\_fr, HDI\_intervals\_for\_interactions\_l),3)  
kable(HDIs\_flake\_retouch\_freq\_and\_length, caption = "Highest density intervals (95%) for the posterior distributions of the interactions between phases and retouch frequency and length, both squares")

Highest density intervals (95%) for the posterior distributions of the interactions between phases and retouch frequency and length, both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | HDIlow | HDIhigh | HDIlow | HDIhigh |
| 1.v.2 | -1.902 | 6.755 | -2.624 | 6.737 |
| 2.v.3 | -8.598 | 0.690 | -8.481 | 2.211 |
| 3.v.4 | -0.802 | 7.872 | -3.785 | 6.370 |

The results of the code chunk above show that the HDIs for all interactions include zero. This indicates no credible difference in retouch length by phase.

The code chunk below repeats this analysis using a frequentist test.

# ANOVA with Tukey's HSD  
fit <- aov(rt\_len ~ as.factor(phase), myDataFrame)  
fit\_summary <- summary(fit)  
tuk <- TukeyHSD(fit)  
# plot(tuk, las = 2, cex = 0.1)  
kable(round(tuk$`as.factor(phase)`,3), caption = "Tukey's Honest Significant Difference for phase by phase comparisons of flake retouch length, both squares")

Tukey's Honest Significant Difference for phase by phase comparisons of flake retouch length, both squares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | diff | lwr | upr | p adj |
| 2-1 | -2.956 | -11.028 | 5.115 | 0.772 |
| 3-1 | 2.099 | -6.090 | 10.289 | 0.907 |
| 4-1 | 0.263 | -8.195 | 8.721 | 1.000 |
| 3-2 | 5.056 | -3.808 | 13.919 | 0.445 |
| 4-2 | 3.219 | -5.893 | 12.332 | 0.791 |
| 4-3 | -1.837 | -11.053 | 7.380 | 0.954 |

In the code chunk above we obtain the results of a frequentist ANOVA which returns a statistically nonsignificant result: F = 0.768798, df = 3, p = 0.5146413. The Tukey's Honest Significant Difference test shows confidence intervals on the differences between the means artefact retouch length of a phase. None of the interactions show a significant difference. We interpret this as evidence of overall no change in flake retouch length.

In the code chunk below we compute a Bayesian t-test to investigate differences in the mass of retouched and non-retouched flakes.

# Bayesian t-test of the mass of all retouched vs all unretouched flakes  
# flakes\_retouch\_size\_t\_test <- BESTmcmc(flakes\_retouch\_size[flakes\_retouch\_size$Artclas == "flake", ]$Weight, flakes\_retouch\_size[flakes\_retouch\_size$Artclas == "retf", ]$Weight)

In the code chunk below we perform a frequentist t-test on flake mass of retouched and non-retouched flakes.

retouch\_nonretouch\_flake\_mass\_nhst <- t.test(flakes\_retouch\_size[flakes\_retouch\_size$Artclas == "flake", ]$Weight, flakes\_retouch\_size[flakes\_retouch\_size$Artclas == "retf", ]$Weight)

The code chunk above returns the result of the frequentist t-test as follows: t = -7.6366818, df = 268.9616914 and p = 3.911675810^{-13}.

The code chunk below makes a table the summarises retouch indices for retouched pieces recovered from Jerimalai. GIUR = Geometric Index of Unifacial Retouch, II = Index of Invasiveness, % = percent of perimeter with retouch

retouch\_indices[is.na(retouch\_indices)] <- 0  
retouch\_indices$GIUR <- with(retouch\_indices, t1/T1 + t2/T2 + t3/T3)/3  
retouch\_indices$perimeter\_perc <- with(retouch\_indices, length/perimeter \* 100)  
retouch\_indices$II <- with(retouch\_indices, ((X0.5 \* 0.5) + (X1 \* 1))/16)  
# get mean and standard deviation for each index  
retouch\_indices\_subset <- retouch\_indices %>% select(GIUR, perimeter\_perc, II)   
# sweep over the columns to compute mean and standard deviation  
retouch\_indices\_means <- data.frame(t(round(apply(retouch\_indices\_subset, 2, mean, na.rm = TRUE),2)))  
retouch\_indices\_sds <- data.frame(t(round(apply(retouch\_indices\_subset, 2, sd, na.rm = TRUE),2)))  
# make table  
retouch\_table <- retouch\_indices %>%  
 select(Square, Spit, Type, GIUR, II, perimeter\_perc)  
# do some rounding  
retouch\_table[,4:6] <- apply(retouch\_table[,4:6], 2, round, 2)  
# have a look  
retouch\_table <- arrange(retouch\_table, Spit)  
# assign phases  
retouch\_table$Phase <- ifelse(retouch\_table$Square == "A",   
 sqA\_makephases(retouch\_table$Spit),   
 sqB\_makephases(retouch\_table$Spit))  
  
# remove NA  
retouch\_table <- retouch\_table[!is.na(retouch\_table$Phase), ]  
# reorder cols  
retouch\_table <- retouch\_table[, c(1,2,7,3:6)]  
# write the table to a csv file so we can put it in the word doc  
write.csv(retouch\_table, file = 'retouch\_table.csv', row.names = FALSE)  
kable(retouch\_table, caption = "Summary of retouch indices for retouched pieces recovered from both squares of Jerimalai. GIUR = Geometric Index of Unifacial Retouch, II = Index of Invasiveness, % = percent of perimeter with retouch")

Summary of retouch indices for retouched pieces recovered from both squares of Jerimalai. GIUR = Geometric Index of Unifacial Retouch, II = Index of Invasiveness, % = percent of perimeter with retouch

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Square | Spit | Phase | Type | GIUR | II | perimeter\_perc |
| 3 | B | 4 | 4 | Double side and end (steep edged) | 0.83 | 0.28 | 50.00 |
| 4 | B | 4 | 4 | Double Side | 0.85 | 0.34 | 33.04 |
| 5 | B | 5 | 4 | Side Ventral | NaN | 0.06 | 11.94 |
| 6 | B | 10 | 4 | Side | 0.37 | 0.09 | 18.66 |
| 7 | B | 10 | 4 | End | 0.46 | 0.03 | 11.80 |
| 8 | B | 12 | 4 | Drill? | 0.55 | 0.19 | 43.10 |
| 9 | B | 14 | 4 | Notch | 0.62 | 0.09 | 20.00 |
| 10 | B | 19 | 4 | Denticulate | 0.19 | 0.16 | 27.68 |
| 11 | B | 22 | 3 | Notch (Ventral) | NaN | 0.09 | 40.69 |
| 12 | B | 25 | 3 | Bifacial End | 0.37 | 0.09 | 17.36 |
| 13 | B | 34 | 3 | Side and End (Alternating bifacial) | 0.38 | 0.12 | 28.23 |
| 14 | B | 38 | 3 | Notch (Ventral) | NaN | 0.06 | 14.59 |
| 15 | B | 39 | 3 | Concave Bifacial side and end | 0.80 | 0.56 | 60.15 |
| 16 | B | 40 | 2 | Side | 0.26 | 0.09 | 21.31 |
| 17 | A | 43 | 1 | Bec | 0.52 | 0.16 | 29.51 |

The retouch intensity can be summarised with the following metrics: GIUR = 0.51 +/- 0.21 perimeter = 29.48 +/- 14.09% II = 0.16 +/- 0.13

# Results: Technological types

This code makes the table "Summary of counts and classes. Counts refers to the count of spits in each depositional phase containing a given class."

## read in the techno types for both squares  
summaryt\_props <- read.csv("data/Jerimalai\_technological\_counts\_both\_squares\_percentages.csv", skip = 1)  
  
names(summaryt\_props) <- c("category", "type", "phase I", "phase II", "phase III", "phase IV")  
  
kable(summaryt\_props, caption = "Summary of proportions and classes. Proportions refers to the proportion of spits in each depositional phase containing a given class")

Summary of proportions and classes. Proportions refers to the proportion of spits in each depositional phase containing a given class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| category | type | phase I | phase II | phase III | phase IV |
| Retouch | Distal | 23.81 | 12.50 | 10.26 | 40 |
|  | Lateral | 19.05 | 18.75 | 10.26 | 32 |
|  | Notched | 14.29 | 0.00 | 7.69 | 24 |
|  | Lateral and Distal | 9.52 | 6.25 | 0.00 | 4 |
|  | Double Side and End | 9.52 | 12.50 | 2.56 | 20 |
|  | Denticulated | 0.00 | 3.13 | 2.56 | 4 |
|  | Notched Lateral and/or Distal | 0.00 | 6.25 | 5.13 | 0 |
|  | Retouched Pointed Projections (Piercer/Drill-like) | 4.76 | 0.00 | 0.00 | 8 |
|  | Old Ventral Surface on DorsalPlatform | 52.38 | 9.38 | 30.77 | 32 |
| Flake Features | Platform Faceting | 71.43 | 34.38 | 74.36 | 76 |
|  | Flake With Old Ventral on Dorsal | 19.05 | 12.50 | 23.08 | 32 |
|  | Truncated Flake | 4.76 | 9.38 | 2.56 | 0 |
|  | Flake With Gloss | 0.00 | 0.00 | 2.56 | 4 |
| Ground | Striated Haematite | 0.00 | 0.00 | 5.13 | 4 |
| Technological | Microblade | 23.81 | 6.25 | 20.51 | 8 |
| Types | Levallois-like Flake | 9.52 | 0.00 | 5.13 | 0 |
|  | Burin Spall | 14.29 | 12.50 | 2.56 | 16 |
|  | Pointed Flake with Facetted Platform | 14.29 | 0.00 | 0.00 | 0 |
|  | Bipolar Flake | 9.52 | 21.88 | 7.69 | 16 |
|  | Éclat Débordant | 0.00 | 9.38 | 5.13 | 0 |
|  | Cortical Flake | 4.76 | 0.00 | 2.56 | 12 |
|  | Redirecting Flake | 33.33 | 21.88 | 17.95 | 44 |
|  | Chopper Anvil | 0.00 | 3.13 | 0.00 | 4 |
| Cores | Discoidal/Semi Discoidal | 4.76 | 6.25 | 0.00 | 8 |
|  | Faceted Radial Levallois-like | 14.29 | 3.13 | 2.56 | 12 |
|  | Truncated Faceted | 14.29 | 3.13 | 0.00 | 4 |
|  | Bipolar | 9.52 | 9.38 | 7.69 | 16 |
|  | Bidirectional | 4.76 | 3.13 | 2.56 | 8 |
|  | Single Platform | 9.52 | 0.00 | 2.56 | 16 |
|  | Multiplatform | 23.81 | 12.50 | 10.26 | 16 |

# figure  
technological\_types\_table <- read.csv("data/Jerimalai\_technological\_counts\_both\_squares.csv")  
  
# spits per phase, both squares  
spits\_per\_phase <- both\_sqs\_all\_with\_phases %>%   
 group\_by(phase) %>%   
 summarize(n\_spits=n\_distinct(Spit))  
  
perc\_spits\_techno\_class <- technological\_types\_table %>%  
 group\_by(class) %>%  
 filter(class %in% c("Retouch", "Bipolar", "Flake cores", "Levallois", "multiplatform")) %>%  
 summarize(I = sum(Phase.I.total.count, na.rm = TRUE) ,  
 II = sum(Phase.II.total.count, na.rm = TRUE),  
 III = sum(Phase.III.total.count, na.rm = TRUE),  
 IV = sum(Phase.IV.total.count, na.rm = TRUE),  
 counts = n())  
  
# divide counts of spits for each class by number of spits per phase  
  
tmp <- perc\_spits\_techno\_class[,-1]  
for(i in seq\_along(perc\_spits\_techno\_class$counts)){  
   
 tmp[,i] <- perc\_spits\_techno\_class[,-1][ ,i] / perc\_spits\_techno\_class$counts[i]  
   
}  
  
apply(tmp[,1:4], 1, function(i) i \* spits\_per\_phase$n)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| I | 54.00000 | 243.00000 | 67.50000 | 162.00000 | 229.5000 |
| II | 73.33333 | 58.66667 | 29.33333 | 80.66667 | 139.3333 |
| III | 68.00000 | 238.00000 | 56.66667 | 124.66667 | 170.0000 |
| IV | 80.00000 | 170.00000 | 30.00000 | 150.00000 | 330.0000 |

# Colophon

This report was generated on 2016-01-18 22:01:12 using the following computational environment and dependences:

# which R packages and versions?  
sessionInfo()

R version 3.2.3 (2015-12-10)  
Platform: x86\_64-w64-mingw32/x64 (64-bit)  
Running under: Windows 7 x64 (build 7601) Service Pack 1  
  
locale:  
[1] LC\_COLLATE=English\_United States.1252   
[2] LC\_CTYPE=English\_United States.1252   
[3] LC\_MONETARY=English\_United States.1252  
[4] LC\_NUMERIC=C   
[5] LC\_TIME=English\_United States.1252   
  
attached base packages:  
[1] grid stats graphics grDevices utils datasets methods   
[8] base   
  
other attached packages:  
 [1] JerimalaiStoneArtefacts\_0.0.0.9000 Bchron\_4.1.2   
 [3] inline\_0.3.14 dependencies\_0.0-1   
 [5] git2r\_0.13.1 data.table\_1.9.6   
 [7] xtable\_1.8-0 lattice\_0.20-33   
 [9] BEST\_0.4.0 runjags\_2.0.2-8   
[11] rjags\_4-5 coda\_0.18-1   
[13] vcd\_1.4-1 dplyr\_0.4.3   
[15] plyr\_1.8.3 reshape2\_1.4.1   
[17] ggplot2\_2.0.0 rmarkdown\_0.9.3   
[19] printr\_0.0.4 knitr\_1.12   
  
loaded via a namespace (and not attached):  
 [1] Rcpp\_0.12.3 highr\_0.5.1 formatR\_1.2.1 tools\_3.2.3   
 [5] mclust\_5.1 digest\_0.6.9 jsonlite\_0.9.19 evaluate\_0.8   
 [9] gtable\_0.1.2 DBI\_0.3.1 jagsUI\_1.3.7 yaml\_2.1.13   
[13] parallel\_3.2.3 stringr\_1.0.0 lmtest\_0.9-34 ellipse\_0.3-8   
[17] hdrcde\_3.1 R6\_2.1.1 magrittr\_1.5 codetools\_0.2-14  
[21] scales\_0.3.0 htmltools\_0.3 MASS\_7.3-45 assertthat\_0.1   
[25] colorspace\_1.2-6 labeling\_0.3 stringi\_1.0-1 lazyeval\_0.1.10   
[29] munsell\_0.4.2 chron\_2.3-47 zoo\_1.7-12

# what other pieces of software?  
needs <- needs()  
c(needs$depends$SystemRequirements[needs$depends$SystemRequirements != "NULL"], needs$imports$SystemRequirements[needs$imports$SystemRequirements != "NULL"])

[1] "zlib headers and library. OpenSSL headers and\nlibrary. LibSSH2 (optional on non-Windows) to enable the SSH\ntransport."  
[2] "JAGS (http://mcmc-jags.sourceforge.net)"   
[3] "JAGS 4.x.y"   
[4] "pandoc (>= 1.12.3) -\nhttp://johnmacfarlane.net/pandoc"   
[5] "JAGS (http://mcmc-jags.sourceforge.net)"   
[6] "ICU4C (>= 50, optional)"

# what commit is this file at?  
repo <- repository(path = "../")  
last\_commit <- commits(repo)[[1]]

The current git commit of this file is c94f3a4b9e29b9117f96f8b8137d212f95190792, which is on the master branch and was made by Ben Marwick on 2015-11-25 12:06:02. The current commit message is "before JAS submit".