Brandon

Wilfrid S. Kendall

26 January 2017

1 Introduction: important functions

Loading libraries:

```
library(AS.preprocessing)
library(igraph)
library(AS.angles)
library(AS.circular)
library(png)
library(ggplot2)
```

Definitions and functions used in quantogram construction:

```
seed <- 1
set.seed(seed)
smooth.sd <- 2.5
simulated.replicates <- 499
freq <- seq(from=0.01, to=3, length=1000)

cluster <- function(a, b) cos(2 * pi * a / b) # Used only in quantocalc below!
quantocalc <- function (f, x) apply(outer(x, 1/f, cluster), 2, sum)*sqrt(2/length(x))
quantogram <- function (q, f=seq(from=0.01, to=3, length=1000), ...)
  plot(f, q,
       xlab=expression(paste("Frequency" ~ tau ~ "(" ~ "metre"^{-1} ~ ")")),
       ylab=expression(paste("Quantogram score ", phi(tau))),
       type="l", ...)

maxfreq.previous <- 0.21
quantum.previous<- 1 / maxfreq.previous
```

2 Preparation of Brandon image

We examine posthole data from Brandon phase 2.2, bearing in mind the quantum `quantum.previous`=4.7619048 (`maxfreq.previous` = 0.21) found in or implied by previous investigations such as Kendall [2013], Huggins [1991]. Selecting building corners *by hand* (using a layered map in `GIMP`), we export the layer containing corner locations, read in the posthole map from the exported layer, and generate a structure which enumerates these corner locations (Figure 1). Numbered locations of corners are given in Figure 2. Finally, note that this map must be scaled, which is accomplished by noting that a specific pair of locations 66 and 67 delineate the scale of the map. This enables `scale.distance` to be computed, the scaling factor required to generate distances in metres.

```
Brandon.image <- readPNG("Brandon.png")
n1 <- nrow(Brandon.image)
n2 <- ncol(Brandon.image)
Brandon.data <- matrix(1 - Brandon.image, nrow=n1, ncol=n2)
r <- raster(Brandon.data, xmn = 0, ymn = 0, xmx = 1, ymx = n1/n2)
```

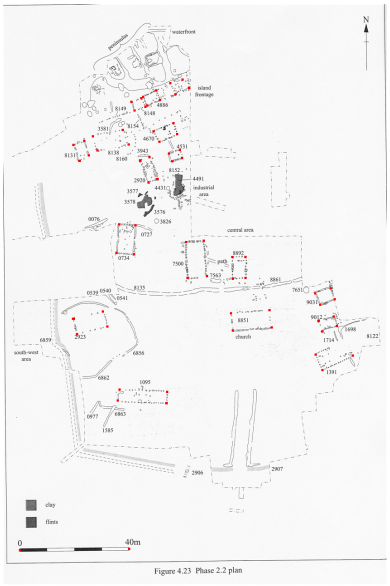

Figure 1: Brandon, with building corners and scale limits selected by hand.

```
threshold <- 0.1
r0 <- r <= threshold
cc <- clump(r0, dir = 4)
clump.types <- cbind(unique(getValues(cc)), 0)
clump.types <- clump.types[!is.na(clump.types[,1]),]
brandon <- list(features = cc, feature.types = clump.types)
get.postholes(brandon, plot=F)
save.features(brandon, "Brandon-final")
write.csv(centres, "Brandon-posthole-centres.csv", row.names=F)

# Computation of scale.
# Check choice of points 66, 67 from following figure:
# They should be the pair of points at the bottom of the figure.
pt.dist <- function (i, j) sqrt((centres[i, 1]-centres[j, 1])**2 +
                                (centres[i, 2]-centres[j, 2])**2)
scale.distance <- 40 / pt.dist(67, 66)
centres <- scale.distance * centres

qplot(data=centres, xm, ym, xlim=c(0, 130), ylim=c(0, 190), asp=190/130) +
  geom_text(label=row.names(centres), hjust=1.75, vjust=0, size=2) +
  coord_fixed() +
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
          axis.text.y=element_blank(),axis.ticks=element_blank(),
          axis.title.x=element_blank(),
          axis.title.y=element_blank())
```

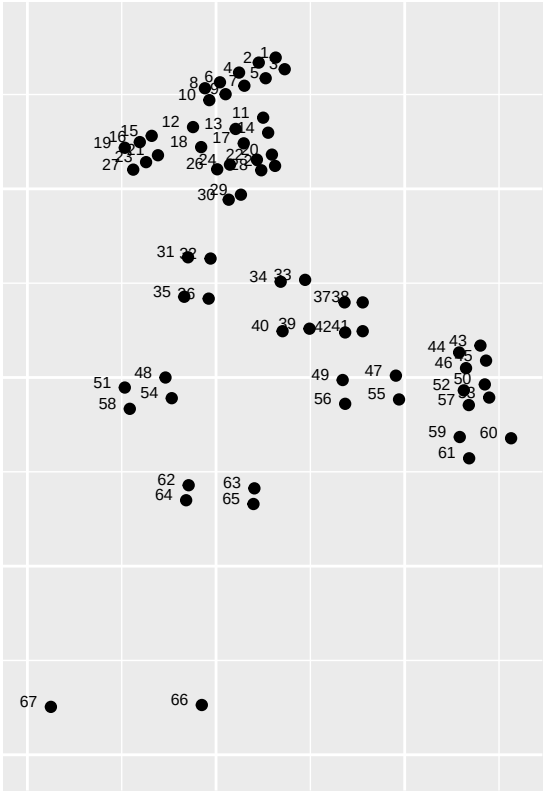

Figure 2: Brandon phase 2, numbered corner locations: points 66, 67 indicate scale of 40m.

3 Statistical analyses

3.1 First attempt: dealing with each building in isolation

Our initial analysis deals only with the dimensions of each building: following the approach indicated in Huggins et al. [1982], we do not consider the geographical relationship of each building to all the others. The next stage is rather clumsy: compute representative length and width for each building. Points 66 and 67 define the scale of 40 metres. Note that one building is defined by just three points 60, 61, 59: this has one corner missing, and therefore interpretation is difficult. It has to be treated in a different way: in this first attempt it is simply omitted.

```
capture <- function (x) c((pt.dist(x[1], x[2]) + pt.dist(x[3], x[4]))/2,
                          (pt.dist(x[1], x[4]) + pt.dist(x[3], x[2]))/2)
mm0 <- list(
            c( 1,  2,  5,  3),
            c( 4,  6,  9,  7),
            c( 6,  8, 10,  9),
            c(12, 15, 21, 18),
            c(16, 19, 27, 23),
            c(11, 13, 17, 14),
            c(24, 26, 30, 29),
            c(20, 22, 28, 25),
            c(32, 31, 35, 36),
            c(48, 51, 58, 54),
            c(33, 34, 40, 39),
```

```
            c(38, 37, 42, 41),
            c(47, 49, 56, 55),
            c(43, 44, 46, 45),
            c(50, 52, 57, 53),
            c(60, 61, 59),
            c(63, 62, 64, 65))
mm <- mm0[unlist(lapply(mm0, function(x) length(x)==4))]
Brandon.dist <- matrix(unlist(lapply(mm, capture)), nrow=length(mm), byrow=T)
```

Compute segments to indicate lengths measured: see Figure 3.

```
segs <- data.frame(matrix(unlist(mm), nrow=length(mm), byrow=T))
segs$x <- centres[segs$X1,1]
segs$y <- centres[segs$X1,2]
segs$xend <- centres[segs$X2,1]
segs$yend <- centres[segs$X2,2]
c <- centres
qplot(data=centres, xm, ym) +
  geom_text(label=row.names(centres), hjust=1.75, vjust=0, size=2) +
  geom_segment(data=segs, aes(x=c[segs$X1,1], y=c[segs$X1,2],
                              xend=c[segs$X2,1], yend=c[segs$X2,2]), colour="grey") +
  geom_segment(data=segs, aes(x=c[segs$X3,1], y=c[segs$X3,2],
                              xend=c[segs$X4,1], yend=c[segs$X4,2]), colour="grey") +
  geom_segment(data=segs, aes(x=c[segs$X1,1], y=c[segs$X1,2],
                              xend=c[segs$X4,1], yend=c[segs$X4,2]), colour="pink") +
  geom_segment(data=segs, aes(x=c[segs$X3,1], y=c[segs$X3,2],
                              xend=c[segs$X2,1], yend=c[segs$X2,2]), colour="pink") +
  coord_fixed()
```

We compute the quantogram and simulation upper envelopes. Figure 4 presents the quantogram with simulation envelope: it appears that there is no particularly good evidence for a quantum in these data alone. (The dashed blue vertical line indicates the quanta obtained in previous investigations.)

```
bd <- as.vector(Brandon.dist)
Brandon.quantos <- quantocalc(freq, bd)

result <- c()

for (i in seq(simulated.replicates)) {
  baseperturbation <- rnorm(1, 0, smooth.sd)
  ysim <- abs(bd + rnorm(bd, 0, smooth.sd) - baseperturbation)
  result <- rbind(result, quantocalc(freq, ysim))
  }
upper <- apply(result, 2, sort)[dim(result)[1]-4,]

quantogram(Brandon.quantos)
lines(freq, upper, type="l", lty="dashed", col="dark red")
abline(v=maxfreq.previous, lty="dashed", col="dark blue")
```

3.2 Second attempt: considering buildings taken together

The previous approach neglects the possibility of alignment of the buildings themselves on a wide-spread grid. Inspection of Figure 1 makes it plain, we need a different more global approach if we are to succeed. In

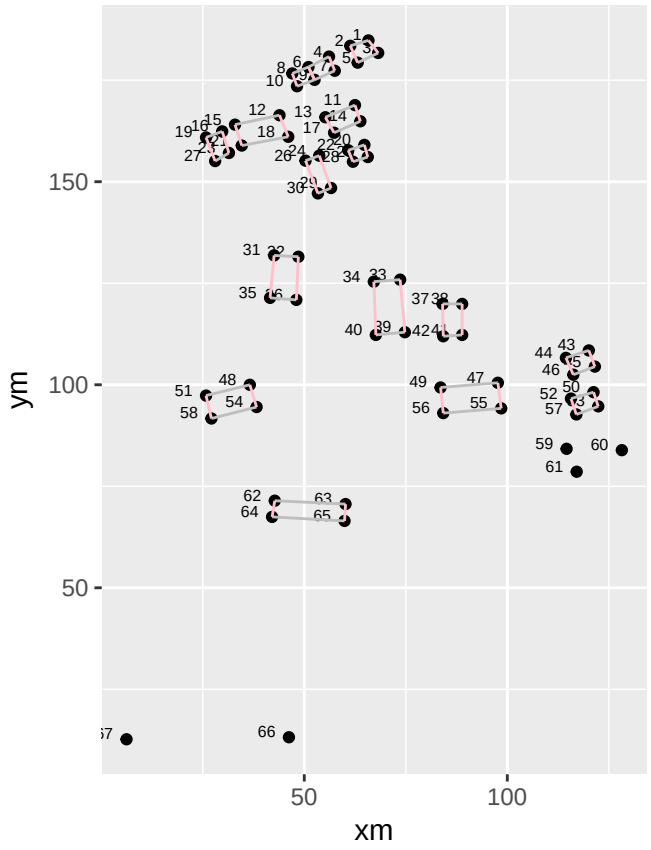

Figure 3: Brandon phase 2, measured lengths for complete buildings.

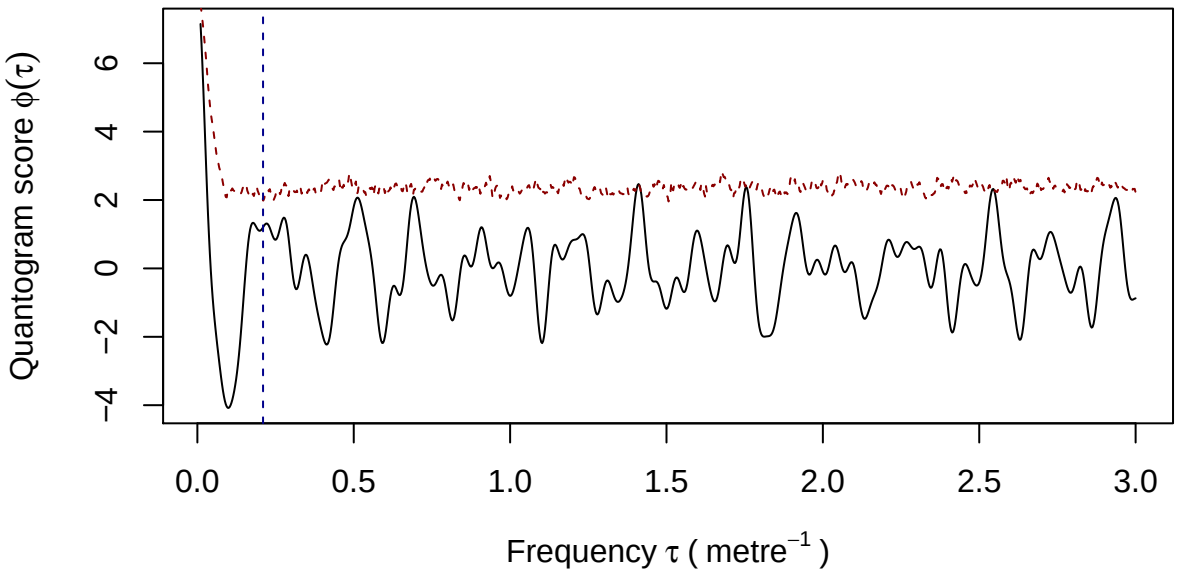

Figure 4: Quantogram with simulation upper envelope for Brandon data, considered locally (building by building). Vertical dashed line corresponds to quanta determined in previous analyses.

particular we need to separate out the buildings according to their alignment on at least two different though overlapping grids.

From now on, we include the partial building with just three corners.

To begin with, we convert edges into grid-orientations, working with the sequences of vertices in the list `mm0`. Note that the list of 3 vertices requires different treatment: we cannot reliably impute the position of the fourth vertex so can only extract two edges hence two orientations, as opposed to four when dealing with four edges forming a complete rectangle. Since we are concerned with *grid* orientations, we represent orientations modulo $\pi/2$ (in radians) and then generate a computationally convenient transformation by multiplying them by 4 to obtain `circular` data. We then obtain initial graphical evidence of bimodality (hence, 2 grid systems): Figure 5 presents a histogram of grid orientations,
rotated by 45 degrees in order clearly to portray the grouping of grid orientations into two different albeit overlapping clusters.

```
c.g <- function (x)
  ((pi + atan2(unlist(x[2]), unlist(x[1]))) %% (pi / 2))
convert.to.grids <- function (ll)
{
  x <- centres[ll, ] ;
  if (length(ll)==4)
    c(c.g(x[2,]-x[1,]), c.g(x[3,]-x[2,]), c.g(x[4,]-x[3,]), c.g(x[1,]-x[4,]))
  else if (length(ll)==3)
    c(c.g(x[2,]-x[1,]), c.g(x[3,]-x[2,]))
}
orientations <- lapply(lapply(mm0, convert.to.grids), unname)
s.orientations <- circular(4 * unlist(orientations))
```

```
qplot((180/pi) * ((pi/4 + unlist(orientations)) %% (pi/2)),
      xlim=c(0,90),
      xlab="Orientation in degrees (45 degrees rotation)",
      binwidth=3)
```

We quantify this notion of bimodality by fitting a mixture of two von Mises distributions to the individual wall orientations, scaled by 4 to fit to the conventional range of angular data in radians, using the EM method. Note that results of the EM fit do depend on initialization of the random number generator seed (`set.seed` above). For some seed values we end up with different mean orientations for the two mixtures, and thus different assignations of clusters. We have chosen a seed which results in just one of the buildings not being fitted, because that building has sides attributed to different grids. Other seeds end up with three buildings not being fitted, for the same reason. In separate calculations, the EM algorithm was repeated from independent random starting points over 200 iterations. Reported maximum log-likelihoods can be summarized by the `fivenum` of `-74.49932 -62.55545 -62.55532 -60.77147 -60.77147` with only 2 log-likelihoods evaluating to less than -63. The two lowest values corresponded to `mu` vectors of `c(1.182078, 6.039959)` and `c(1.061790, 1.062784)`. The remainder delivered essentially two different solutions: `c(0.55, 1.45)` at log-likelihood -60.77147 and `c(0.36, 1.48)` at log-likelihood -62.55. Our chosen seed picks the most likely of these modes.

```
mixture <- EM.vonmises(s.orientations, 2)
mixture$mu
```

```
## Circular Data:
## Type = angles
## Units = radians
## Template = none
## Modulo = asis
## Zero = 0
## Rotation = counter
## [1] 1.44719522 0.05505553
```

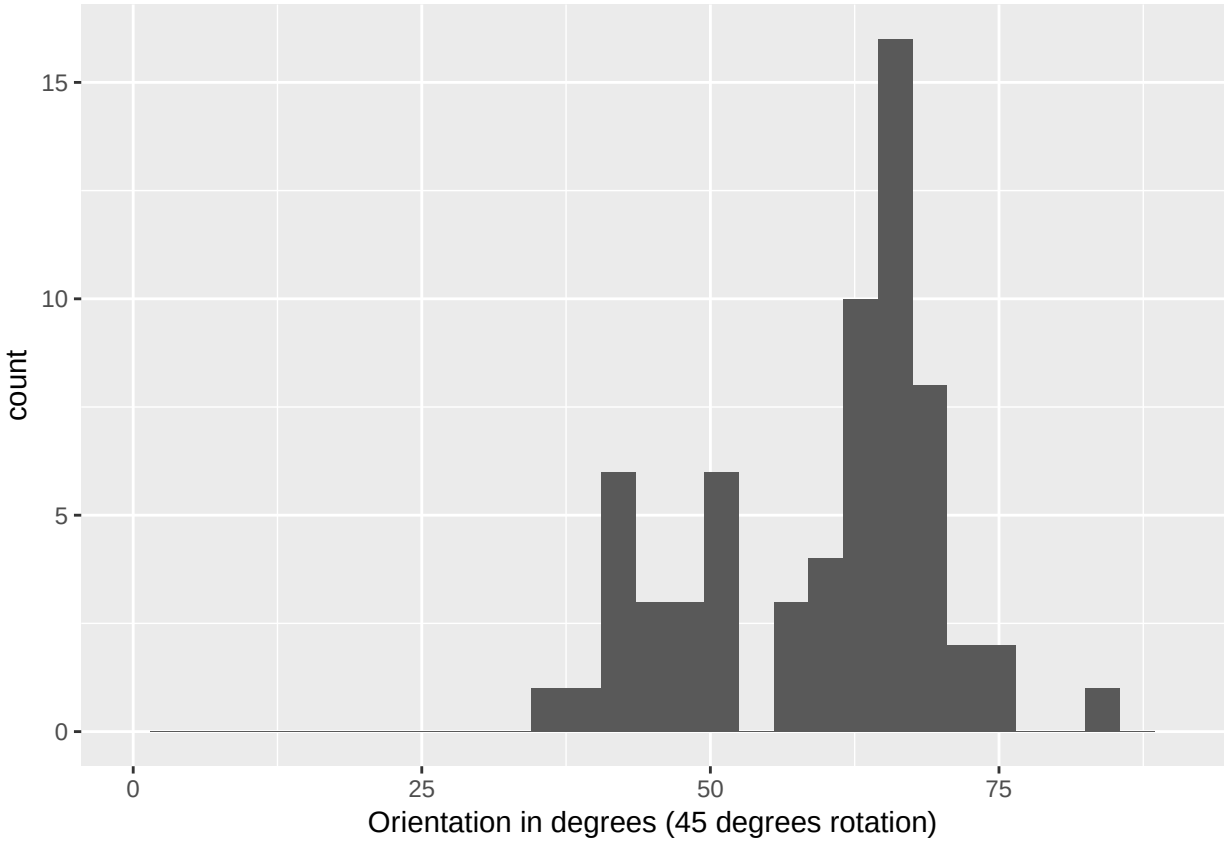

Figure 5: Histogram of grid orientations for building sides: note there is a clear suggestion of two different overlapping clusters.

```
mixture[5]
```

```
## $log.lh
## [1] -60.77147
```

We present the two orientations fitted by the mixture, measured in degrees of arc.

```
plot.EM.vonmises(s.orientations, mixture)
```

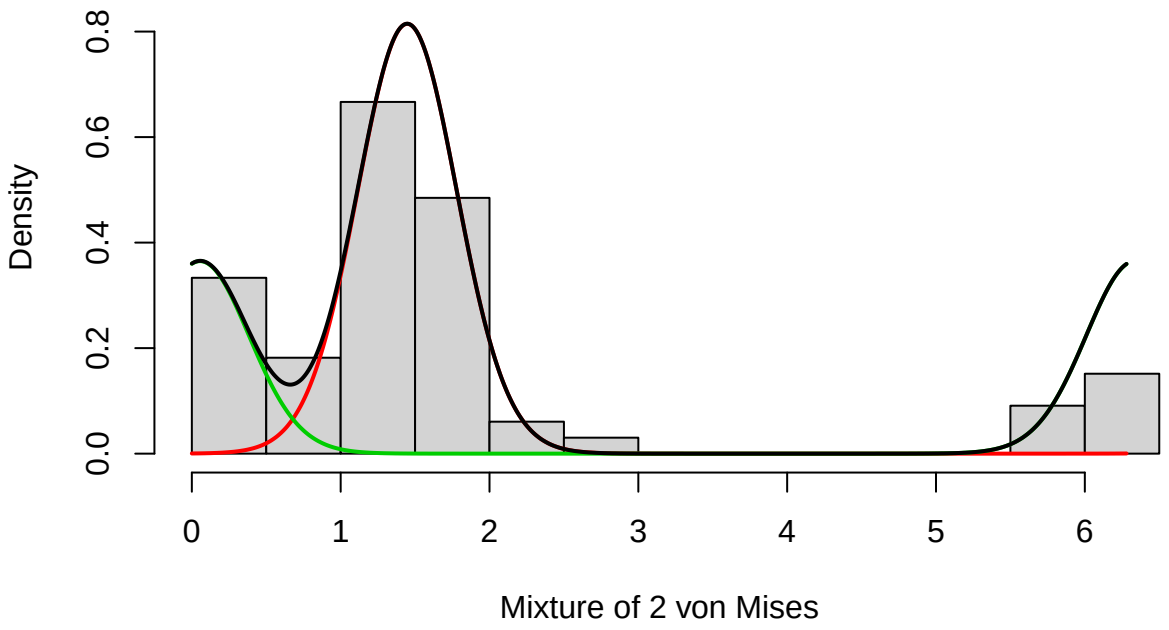

Figure 6: Fitted mixture of two von Mises together with individual orientation data (scaled by 4).

```
cat("Modes in degrees ", as.numeric((mixture$mu)) * (45/pi))
```

```
## Modes in degrees  20.72954 0.7886123
```

Consider now the goodness-of-fit, using residual plots. Fit seems moderately good.

```
residuals <- mvM.PP(s.orientations, mixture$mu[1], mixture$kappa[1],
                    mixture$mu[2], mixture$kappa[2],
                    mixture$alpha[1])
```

```
mean(residuals^2); sd(residuals^2)
```

```
## [1] 0.001011531
```

```
## [1] 0.001194659
```

The two fitted orientations are located (in radians) at 0.3617988, 0.0137639. We now assign observations according to which of the two mixture components evaluates to a higher probability density.

```
cluster1.test <- function (xo)
  dvonmises(circular(4 * xo), circular(mixture$mu[1]), mixture$kappa[1], T) >
  dvonmises(circular(4 * xo), circular(mixture$mu[2]), mixture$kappa[2], T)
cluster2.test <- function (xo)
  dvonmises(circular(4 * xo), circular(mixture$mu[1]), mixture$kappa[1], T) <=
  dvonmises(circular(4 * xo), circular(mixture$mu[2]), mixture$kappa[2], T)
c1 <- unlist(lapply(lapply(orientations, cluster1.test), all))
```

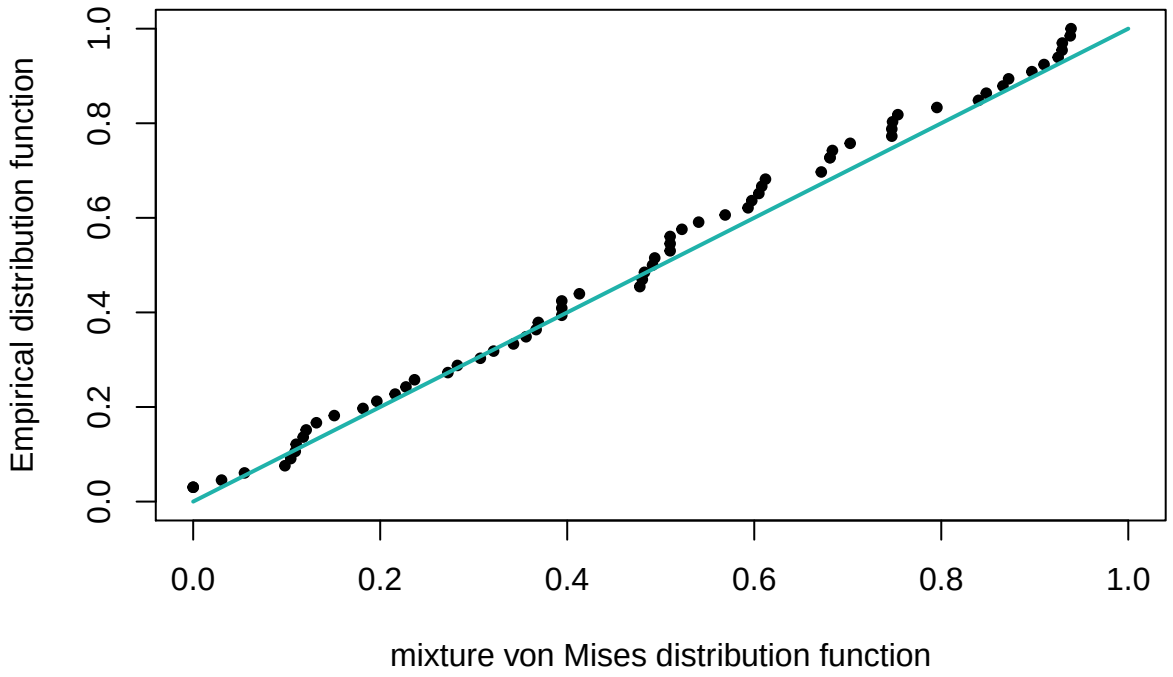

Figure 7: Residual plot for fit to orientation data of a mixture of two von Mises distributions.

```
c2 <- unlist(lapply(lapply(orientations, cluster2.test), all))
```

Buildings in cluster `c1`, with mean orientation 0.3617988, are numbers 1, 2, 3, 5, 6, 7, 8, 10, 14, 15, 16, and buildings in cluster `c2`, with mean orientation 0.0137639, are numbers 9, 11, 12, 13, 17 (numbers as in list `mm` above, including the building with only three vertices recorded). Just one building is omitted because its edges are assigned to different clusters.

```
count(!c1 & !c2)
```

```
##       x freq
## 1 FALSE   16
## 2  TRUE    1
```

We use different colours and shapes to plot the two different clusters and the omitted building (Figure 8).

```
qplot(data=centres[unlist(mm0[!c1 & !c2]),], xm, ym,
      xlim=c(0, 130), ylim=c(40, 190), asp=150/130, colour=I("red")) +
  geom_point(data=centres[unlist(mm0[c2]),], shape=1, colour="dark blue") +
  geom_point(data=centres[unlist(mm0[c1]),], shape=2, colour="dark green") +
  coord_fixed() +
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
          axis.text.y=element_blank(),axis.ticks=element_blank(),
          axis.title.x=element_blank(),
          axis.title.y=element_blank())
```

3.2.1 Quantograms

We can now compute four separate sequences of resolved measurements, essentially `x` and `y` measurements for each grid. First, note that `unlist(mm0[c1])` and `unlist(mm0[c1])` form the lists of vertices involved in each of the two clusters. Note that (at present!) there are occasionally repetitions due to shared edges. We

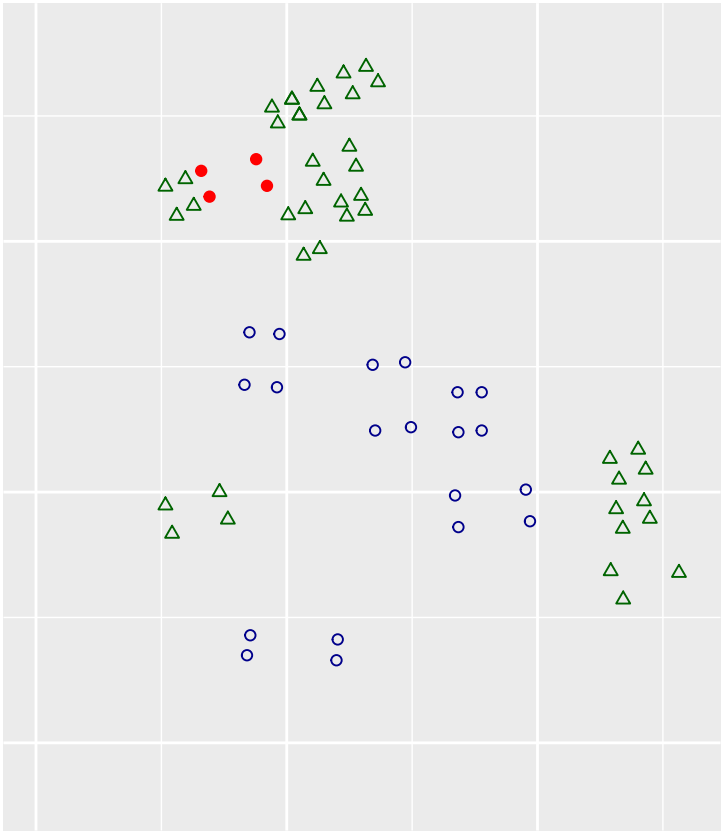

Figure 8: The two different clusters of buildings (open blue circles and green triangles) and the omitted building (solid red circles).

also need the angles corresponding to the two grid systems. We can then compute properly scaled resolutions against axes.

```
c1.pts <- as.matrix(centres[unlist(mm0[c1]),])
c2.pts <- as.matrix(centres[unlist(mm0[c2]),])
theta1 <- as.numeric(mixture$mu[1])/4
theta2 <- as.numeric(mixture$mu[2])/4
locs1x <- sort(c1.pts %*% c(cos(theta1), sin(theta1)))
locs1y <- sort(c1.pts %*% c(cos(theta1 + pi/2), sin(theta1 + pi/2)))
locs2x <- sort(c2.pts %*% c(cos(theta2), sin(theta2)))
locs2y <- sort(c2.pts %*% c(cos(theta2 + pi/2), sin(theta2 + pi/2)))
```

We compute all possible differences for each of these cases, and concatenate the results, scale, and plot the resulting quantogram of the compendium data together with local maximum near left of quantogram. We `focus` on the range from 3.0 metres up to 6.5 metres.

```
freqbounds <- c(1/6.5, 1/3.0)
focus <- (freqbounds[1] < freq & freq < freqbounds[2])

all.differences <- function (z) outer(z, z, '-')[lower.tri(mat.or.vec(length(z), length(z)))]
l1x <- all.differences(locs1x)
l1y <- all.differences(locs1y)
l2x <- all.differences(locs2x)
l2y <- all.differences(locs2y)
data <- c(l1x, l1y, l2x, l2y)
```

```
result <- c()
result1 <- c()
result2 <- c()
result1x <- c()
result1y <- c()
result2x <- c()
result2y <- c()

for (i in seq(simulated.replicates)) {
  sim1x <- all.differences(rnorm(locs1x, locs1x, smooth.sd))
  sim1y <- all.differences(rnorm(locs1y, locs1y, smooth.sd))
  sim2x <- all.differences(rnorm(locs2x, locs2x, smooth.sd))
  sim2y <- all.differences(rnorm(locs2y, locs2y, smooth.sd))
  result <- rbind(result, quantocalc(freq, c(sim1x, sim1y, sim2x, sim2y)))
  result1 <- rbind(result1, quantocalc(freq, c(sim1x, sim2x)))
  result2 <- rbind(result2, quantocalc(freq, c(sim1x, sim2x)))
  result1x <- rbind(result1x, quantocalc(freq, sim1x))
  result1y <- rbind(result1y, quantocalc(freq, sim1y))
  result2x <- rbind(result2x, quantocalc(freq, sim2x))
  result2y <- rbind(result2y, quantocalc(freq, sim2y))
}
upper <- apply(result, 2, sort)[dim(result)[1]-4,]
upper1 <- apply(result1, 2, sort)[dim(result1)[1]-4,]
upper2 <- apply(result2, 2, sort)[dim(result2)[1]-4,]
upper1x <- apply(result1x, 2, sort)[dim(result1x)[1]-4,]
upper1y <- apply(result1y, 2, sort)[dim(result1y)[1]-4,]
upper2x <- apply(result2x, 2, sort)[dim(result2x)[1]-4,]
upper2y <- apply(result2y, 2, sort)[dim(result2y)[1]-4,]
```

```
q <- quantocalc(freq, data)

maxfreq <- freq[focus][q[focus]==max(q[focus])]
quantum <- 1/maxfreq

quantogram(q)
lines(freq, upper, type="l", lty="dashed", col="dark red")

abline(v=maxfreq, col="pink")
abline(v=maxfreq.previous, col="blue", lty="dashed", lwd=2)
abline(v=freqbounds, col="green", lty=3, lwd=1)
lines(freq, q)
```

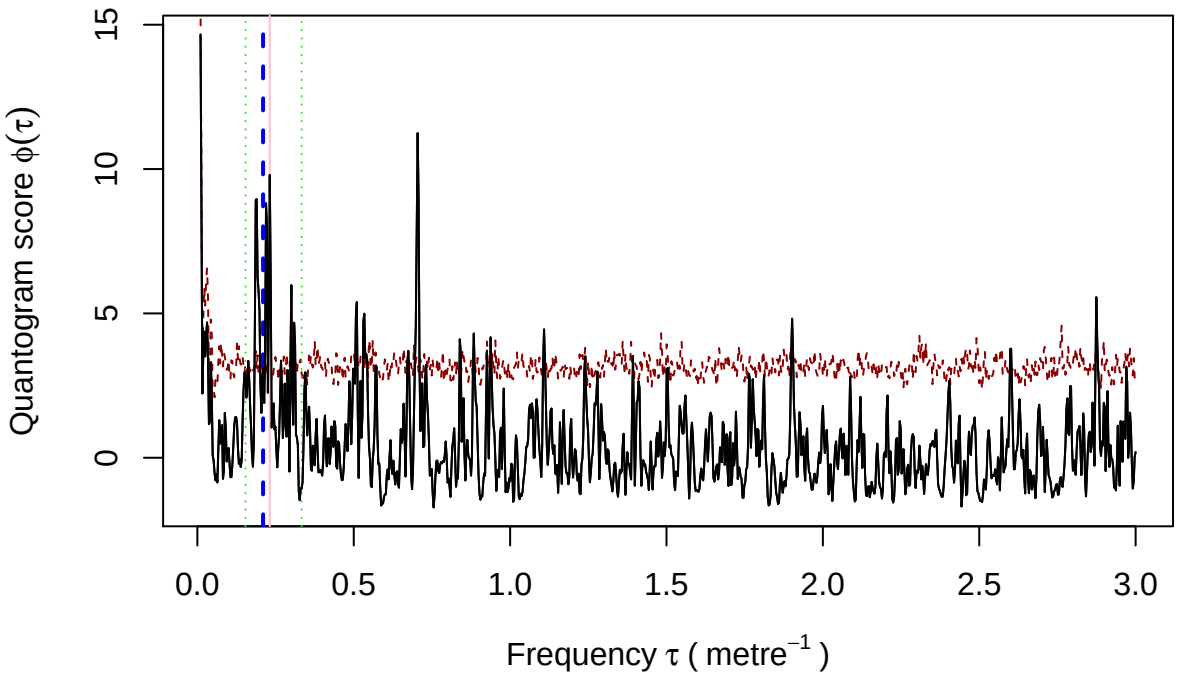

Figure 9: Quantogram using compendium of all distances, with 1% simulation upper envelope as irregular red dotted curve. Solid vertical pink line indicates frequency yielding highest level within region of interest (delimited by vertical dotted green lines). Dashed vertical blue line indicates frequency yielding highest level for fit to church foundations.

```
cat("Module is estimated as ", quantum)
```

```
## Module is estimated as  4.32
```

The larger peak to the right of the chosen range is at order of 1/3 of the frequency for the estimated quantum: we do not consider this because it is out of the range of relevant possibilities based on other quantogram measurements. Perhaps it arises from design ratios for the buildings. The previously quantum was `quantum.previous`=4.7619048: as can be seen from the quantogram, one could also argue for this being significant. We chose in the following to work with the current fit of 4.32, but one could opt otherwise.

```
maxfreq.secondary <-
  freq[freq>freqbounds[2]][q[freq>freqbounds[2]]==max(q[freq>freqbounds[2]])]
c(maxfreq, maxfreq.secondary)
```

```
## [1] 0.2314815 0.7043744
```

```
maxfreq.secondary / maxfreq
```

```
## [1] 3.042897
```

```
data.large <- data[data > 5]
q.large <- quantocalc(freq, data.large)

maxfreq.large <- freq[focus][q.large[focus]==max(q.large[focus])]
quantum.large <- 1/maxfreq.large

quantogram(q.large)
abline(v=maxfreq.large, col="pink")
abline(v=maxfreq.secondary,col="pink", lty=2)
abline(v=freqbounds, col="green", lty=3, lwd=2)
lines(freq, q.large)
```

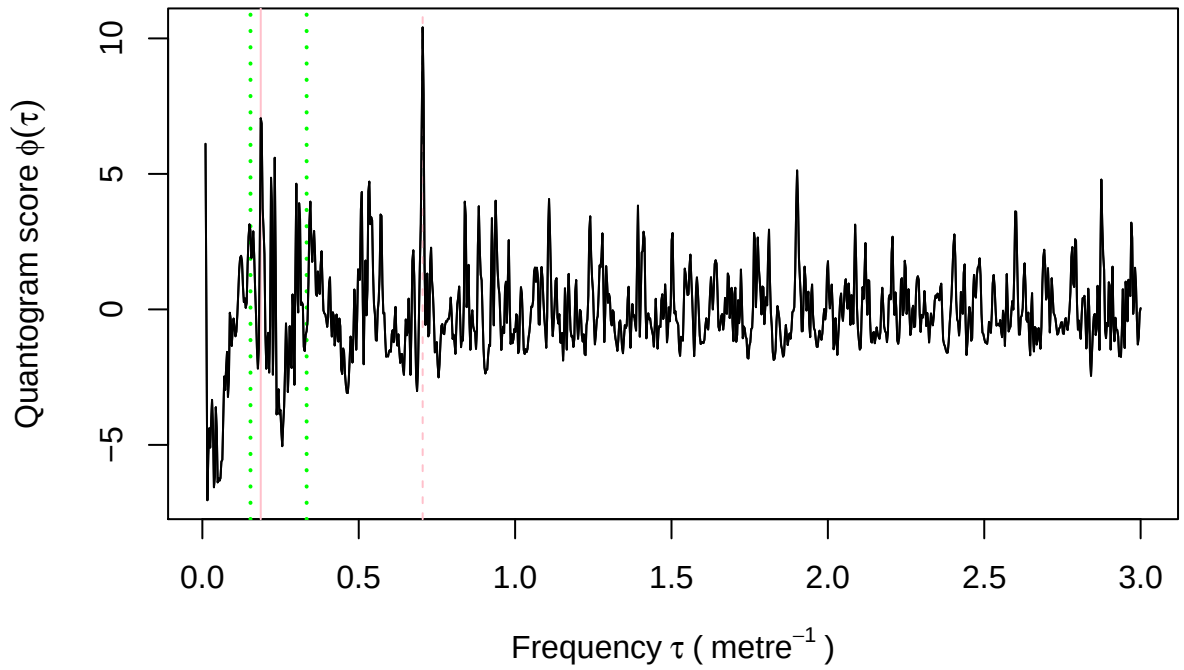

Figure 10: Quantogram using only large distances. Solid vertical red line indicates frequency yielding highest level within region of interest (delimited by vertical dotted green lines). This produces a quantum larger than that from the compendium fit.

```
c(quantum, 1/maxfreq.large)
```

```
## [1] 4.320000 5.359442
```

Consider the quantogram obtained by considering only the large distances (Figure 10): we obtain a larger quantum of 5.3594421. We will however persist with the compendium quantum 4.32. The large secondary peak persists, suggesting that this phenomenon arises as a genuine feature of the dataset. (We omit the simulation envelopes here for the sake of computational convenience: selecting large distances from the set of all possible simulated distances would be somewhat tedious.)

Consider the four lines of measurement separately.

```
par(mfrow=c(2,2))
```

```
q1x <- quantocalc(freq, l1x)
maxfreq1x <- freq[focus][q1x[focus]==max(q1x[focus])]
quantogram(q1x)
lines(freq, upper1x, type="l", lty="dashed", col="dark red")
abline(v=maxfreq1x, col="pink")
abline(v=freqbounds, col="green", lty=3, lwd=2)
lines(freq, q1x)

q1y <- quantocalc(freq, l1y)
maxfreq1y <- freq[focus][q1y[focus]==max(q1y[focus])]
quantogram(q1y)
lines(freq, upper1y, type="l", lty="dashed", col="dark red")
abline(v=maxfreq1y, col="pink")
abline(v=freqbounds, col="green", lty=3, lwd=2)
lines(freq, q1y)

q2x <- quantocalc(freq, l2x)
maxfreq2x <- freq[focus][q2x[focus]==max(q2x[focus])]
quantogram(q2x)
lines(freq, upper2x, type="l", lty="dashed", col="dark red")
abline(v=maxfreq2x, col="pink")
abline(v=freqbounds, col="green", lty=3, lwd=2)
lines(freq, q2x)

q2y <- quantocalc(freq, l2y)
maxfreq2y <- freq[focus][q2y[focus]==max(q2y[focus])]
quantogram(q2y)
lines(freq, upper2y, type="l", lty="dashed", col="dark red")
abline(v=maxfreq2y, col="pink")
abline(v=freqbounds, col="green", lty=3, lwd=2)
lines(freq, q2y)
```

```
1/maxfreq1x
```

```
## [1] 5.27483
```

```
1/maxfreq1y
```

```
## [1] 4.32
```

```
1/maxfreq2x
```

```
## [1] 5.113375
```

```
1/maxfreq2y
```

```
## [1] 6.383387
```

Alternatively consider the two different grids.

```
par(mfrow=c(1,2))
q1 <- quantocalc(freq, c(l1x, l1y))
maxfreq1 <- freq[focus][q1[focus]==max(q1[focus])]
quantogram(q1)
lines(freq, upper1, type="l", lty="dashed", col="dark red")
abline(v=maxfreq1, col="pink")
abline(v=freqbounds, col="green", lty=3, lwd=2)
```

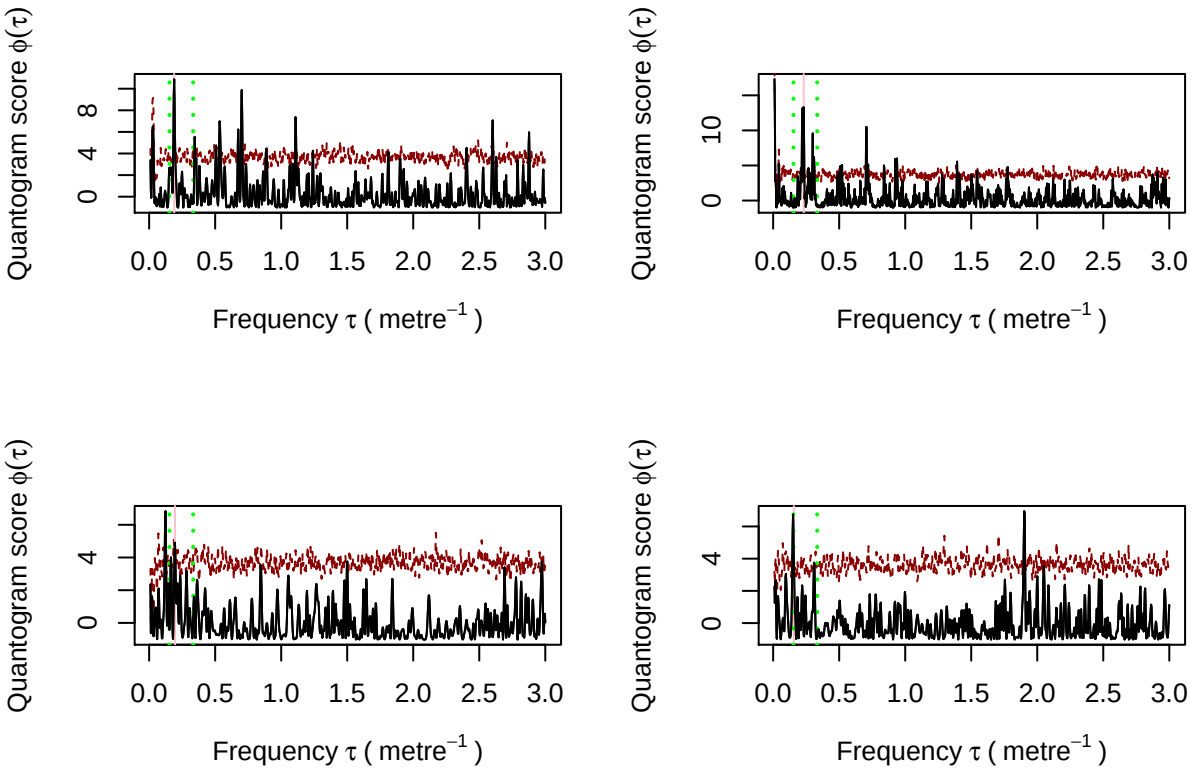

Figure 11: Quantograms for each of the four lines of measurement, cluster 1 at top and cluster 2 at bottom, with so-called x-axis to left and so-called y-axis to right, with 1%-significance level simulation upper envelopes as irregular red dotted curves. Solid vertical red lines indicate frequencies yielding highest level within region of interest (delimited by vertical dotted green lines).

```
lines(freq, q1)
quantum1 <- 1/maxfreq1
quantum1
```

```
## [1] 4.32
```

```
q2 <- quantocalc(freq, c(l2x, l2y))
maxfreq2 <- freq[focus][q2[focus]==max(q2[focus])]
quantogram(q2)
lines(freq, upper2, type="l", lty="dashed", col="dark red")
abline(v=maxfreq2, col="pink")
abline(v=freqbounds, col="green", lty=3, lwd=2)
lines(freq, q2)
```

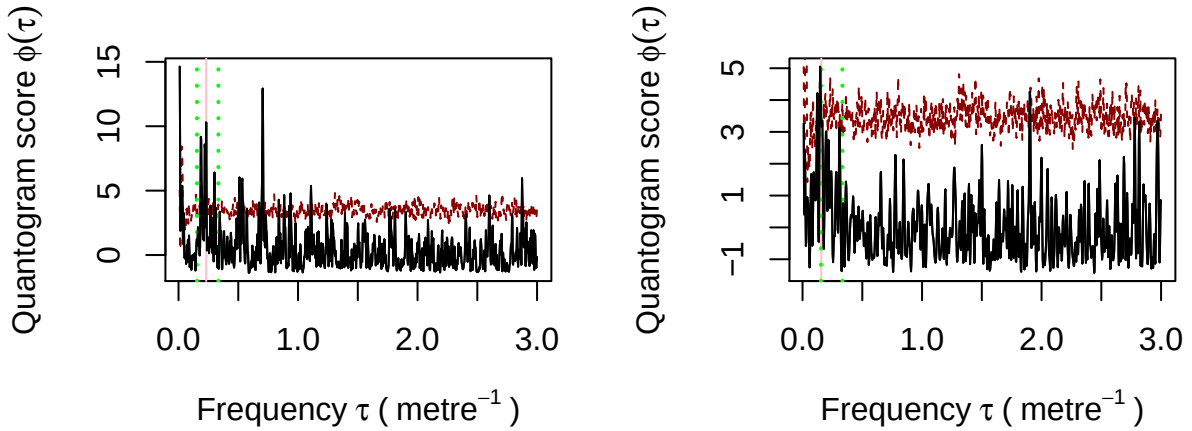

Figure 12: Quantograms for each of the two different grids, cluster 1 to left and cluster 2 to right, with 1%-significance level simulation upper envelopes as irregular red dotted curves. Solid vertical red lines indicate frequencies yielding highest level within region of interest (delimited by vertical dotted green lines).

```
quantum2 <- 1/maxfreq2
quantum2
```

```
## [1] 6.383387
```

The dataset corresponding to the second cluster contains significantly fewer buildings (5 versus 11), thus it is not surprising that the quantogram analysis in this case suggests that evidence for a quantum is less marked. Note that the quantum which is fitted here is rather large, at 6.3833866, whereas the quantum for the first cluster remains numerically the same at 4.32. In the following we shall lead with consideration of grids corresponding to the quantum 4.32 arising from the single fit to all data, though we shall also investigate the grid arising from use of the larger quantum 6.3833866 for the second cluster.

3.2.2 Estimate of "Construction error"

Proceeding as in Kendall [2013], we can compute the underlying "construction error" by a method-of-moments argument. For now, we work with the quantogram of the compendium data.

```
builders.sigma <- quantum * sqrt(log(sqrt(2 * length(data)) / max(q[focus]))
                                 / (2 * pi^2))
builders.sigma
```

```
## [1] 1.343591
```

3.2.3 Detecting the grids

We can detect the grids as follows: for each line of measurement `locs1x`, `locs1y`, `locs2x`, `locs2y`, reduce the measurements modulo the compendium quantum '*quantum*' = 4.32, scale to circular data over the range $[0, 2\pi)$, fit a von Mises distribution, and scale back to the range $(0, 4.32)$. This provides a set of 4 displacements defining the grids ("vertical" and "horizontal" displacements for each grid). Note that the fitted mode `mu` can also be viewed as a Fréchet mean of circular data; the von Mises procedure also equips the fitted mode with statistical information but we might choose simply to regard the statistical fit as a mechanism for arriving at a reasonable displacement of the grid.

```
vm1x <- mle.vonmises(circular((2 * pi / quantum) * (locs1x %% quantum)))
disp1x <- (quantum / (2 * pi)) * as.numeric(vm1x[2])

vm1y <- mle.vonmises(circular((2 * pi / quantum) * (locs1y %% quantum)))
disp1y <- (quantum / (2 * pi)) * as.numeric(vm1y[2])

vm2x <- mle.vonmises(circular((2 * pi / quantum) * (locs2x %% quantum)))
disp2x <- (quantum / (2 * pi)) * as.numeric(vm2x[2])

vm2y <- mle.vonmises(circular((2 * pi / quantum) * (locs2y %% quantum)))
disp2y <- (quantum / (2 * pi)) * as.numeric(vm2y[2])
```

Goodness of fit: for cluster 1, the x coordinate seems fine, the y coordinate fails two of the four tests. For cluster 2, tests are passed. However it should be noted that the "peaked-ness" of the y coordinate fit for cluster 1 is considerably higher (`vm1y$kappa`, versus `vm1x$kappa` for the corresponding x coordinate, and `vm2x$kappa` and `vm2y$kappa` for the second cluster): statistical details of the von Mises distribution mean that this allows more room for the fit to be poor. We supplement the significance levels with QQ-plots. The relative sparsity of data for cluster 2 (second row) leads to larger fluctuations from the ideal diagonal line. Note that the so-called y coordinate of the first cluster (top row, right) leads to significantly more deviation from the ideal diagonal line, compared with the x coordinate (top row, left).

```
c1resx <- vM.GoF(circular((2 * pi / quantum) * (locs1x %% quantum)), vm1x$mu, vm1x$kappa)
```

```
##
##        Watson's Test for the von Mises Distribution
##
## Test Statistic: 0.0398
## P-value > 0.10
##
##
##        Kuiper's Test of Uniformity
##
## Test Statistic:  0.9102
## P-value > 0.15
##
##
##        Watson's Test for Circular Uniformity
##
## Test Statistic: 0.0382
## P-value > 0.10
##
##
##        Rao's Spacing Test of Uniformity
##
## Test Statistic = 128.0163
```

```
## P-value > 0.10
##
##
##        Rayleigh Test of Uniformity
##        General Unimodal Alternative
##
## Test Statistic:  0.0249
## P-value:  0.9739
```

```
c1resy <- vM.GoF(circular((2 * pi / quantum) * (locs1y %% quantum)), vm1y$mu, vm1y$kappa)
```

```
##
##        Watson's Test for the von Mises Distribution
##
## Test Statistic: 0.13
## P-value < 0.01
##
##
##        Kuiper's Test of Uniformity
##
## Test Statistic:  1.7664
## 0.025 < P-value < 0.05
##
##
##        Watson's Test for Circular Uniformity
##
## Test Statistic: 0.1302
## P-value > 0.10
##
##
##        Rao's Spacing Test of Uniformity
##
## Test Statistic = 147.2715
## P-value > 0.10
##
##
##        Rayleigh Test of Uniformity
##        General Unimodal Alternative
##
## Test Statistic:  0.1361
## P-value:  0.4535
```

```
c2resx <- vM.GoF(circular((2 * pi / quantum) * (locs2x %% quantum)), vm2x$mu, vm2x$kappa)
```

```
##
##        Watson's Test for the von Mises Distribution
##
## Test Statistic: 0.0315
## P-value > 0.10
##
##
##        Kuiper's Test of Uniformity
##
## Test Statistic:  1.0264
## P-value > 0.15
```

```
##
##
##        Watson's Test for Circular Uniformity
##
## Test Statistic: 0.0278
## P-value > 0.10
##
##
##        Rao's Spacing Test of Uniformity
##
## Test Statistic = 145.0052
## P-value > 0.10
##
##
##        Rayleigh Test of Uniformity
##        General Unimodal Alternative
##
## Test Statistic:  0.0633
## P-value:  0.9248
```

```
c2resy <- vM.GoF(circular((2 * pi / quantum) * (locs2y %% quantum)), vm2y$mu, vm2y$kappa)
```

```
##
##        Watson's Test for the von Mises Distribution
##
## Test Statistic: 0.0368
## P-value > 0.10
##
##
##        Kuiper's Test of Uniformity
##
## Test Statistic:  0.9992
## P-value > 0.15
##
##
##        Watson's Test for Circular Uniformity
##
## Test Statistic: 0.0334
## P-value > 0.10
##
##
##        Rao's Spacing Test of Uniformity
##
## Test Statistic = 129.8573
## P-value > 0.10
##
##
##        Rayleigh Test of Uniformity
##        General Unimodal Alternative
##
## Test Statistic:  0.0289
## P-value:  0.9838
```

```
par(mfrow=c(2,2))
QQ.1x <- vM.QQ(circular((2 * pi / quantum) * (locs1x %% quantum)), vm1x$mu, vm1x$kappa)
```

```
QQ.1y <- vM.QQ(circular((2 * pi / quantum) * (locs1y %% quantum)), vm1y$mu, vm1y$kappa)
QQ.2x <- vM.QQ(circular((2 * pi / quantum) * (locs2x %% quantum)), vm2x$mu, vm2x$kappa)
QQ.2y <- vM.QQ(circular((2 * pi / quantum) * (locs2y %% quantum)), vm2y$mu, vm2y$kappa)
```

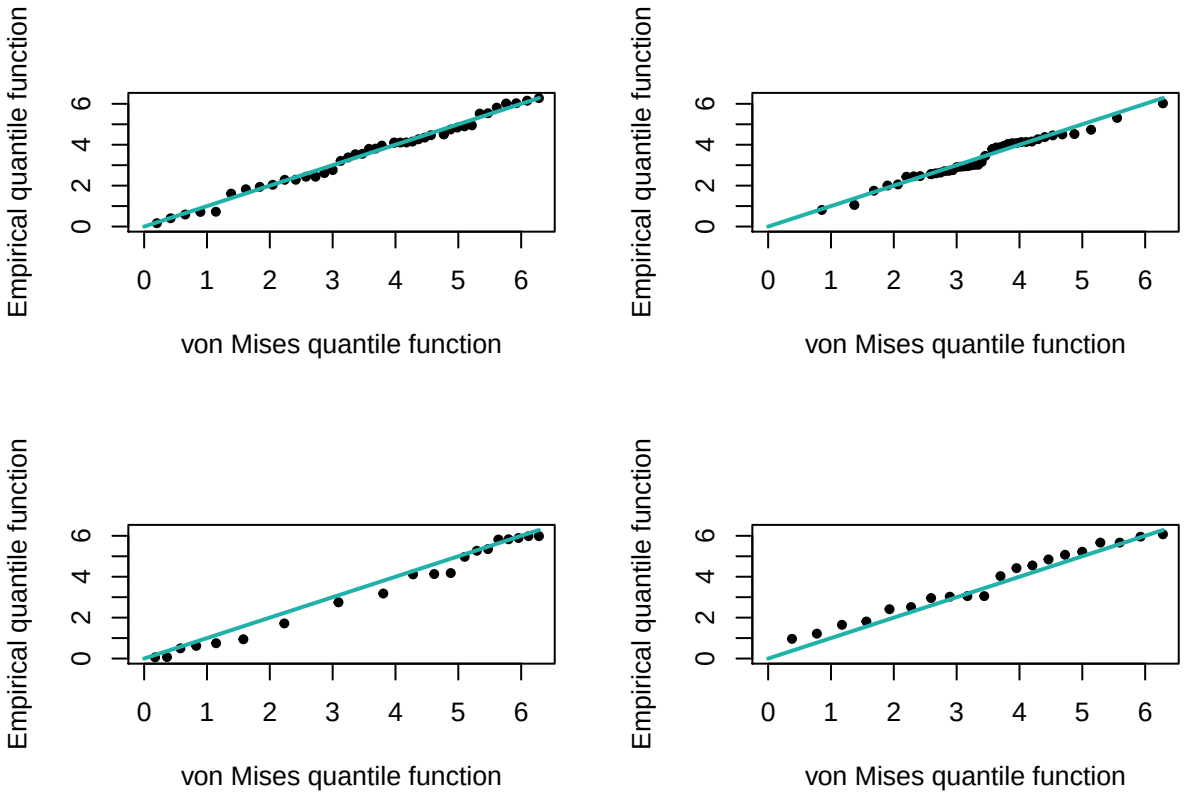

Figure 13: QQ-plots for each of the four lines of measurement, cluster 1 at top and cluster 2 at bottom, with so-called x-axis to left and so-called y-axis to right.

Bootstrap tests present a similar picture: the y coordinate for cluster 1 is a poorer fit.

```
vM.GoF.boot(circular((2 * pi / quantum) * (locs1x %% quantum)), show.progress=FALSE)
```

```
##   kuiper   watson      rao rayleigh
##   0.4891   0.3225   0.5502   0.4385
```

```
vM.GoF.boot(circular((2 * pi / quantum) * (locs1y %% quantum)), show.progress=FALSE)
```

```
##   kuiper   watson      rao rayleigh
##   0.0018   0.0093   0.0747   0.1028
```

```
vM.GoF.boot(circular((2 * pi / quantum) * (locs2x %% quantum)), show.progress=FALSE)
```

```
##   kuiper   watson      rao rayleigh
##   0.5247   0.5913   0.1174   0.4954
```

```
vM.GoF.boot(circular((2 * pi / quantum) * (locs2y %% quantum)), show.progress=FALSE)
```

```
##   kuiper   watson      rao rayleigh
##   0.4720   0.3596   0.3664   0.4287
```

We now use the modal displacements together with the fitted grid orientations and the `quantum` estimate to add fitted grid lines to the plot of building corners.

```
grid1x <- data.frame(
  intercept = (disp1y + quantum * (5:38)) / cos(theta1),
  slope = rep(tan(theta1), 38 - 5 + 1)
)

grid1y <- data.frame(
  intercept = (disp1x - quantum * (10:38)) / cos(theta1 + pi/2),
  slope = rep(tan(theta1 + pi/2), 38 - 10 + 1)
)

grid2x <- data.frame(
  intercept = (disp2y + quantum * (15:33)) / cos(theta2),
  slope = rep(tan(theta2), 33 - 15 + 1)
)

grid2y <- data.frame(
  intercept = (disp2x + quantum * (-25:-5)) / cos(theta2 + pi/2),
  slope = rep(tan(theta2 + pi/2), -5 + 25 + 1)
)

qplot(data=centres[unlist(mm0[!c1 & !c2]),], xm, ym,
      xlim=c(0, 130), ylim=c(40, 190), asp=150/130, colour=I("red")) +
  coord_fixed() +
  geom_abline(data=grid1x, aes(intercept=intercept, slope=slope), colour="light green") +
  geom_abline(data=grid1y, aes(intercept=intercept, slope=slope), colour="light green") +
  geom_abline(data=grid2x, aes(intercept=intercept, slope=slope), colour="light blue") +
  geom_abline(data=grid2y, aes(intercept=intercept, slope=slope), colour="light blue") +
  geom_point(data=centres[unlist(mm0[c2]),], shape=1, colour="dark blue") +
  geom_point(data=centres[unlist(mm0[c1]),], shape=2, colour="dark green") +
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
          axis.text.y=element_blank(),axis.ticks=element_blank(),
          axis.title.x=element_blank(),
          axis.title.y=element_blank())
```

The first cluster (green triangles) seems to fit well to its grid. The second cluster (open blue circles) has a less convincing fit to its grid. Perhaps this is because we are using the same `quantum`=4.32 to fit both grids. We try alternative fits using the quanta fitted separately to the two clusters, namely `quantum1`=4.32 and `quantum2`=6.3833866.

For cluster 1, no change is required:

```
qplot(data=centres[unlist(mm0[!c1 & !c2]),], xm, ym,
      xlim=c(0, 130), ylim=c(40, 190), asp=150/130, colour=I("red")) +
  coord_fixed() +
  geom_abline(data=grid1x, aes(intercept=intercept, slope=slope), colour="light green") +
  geom_abline(data=grid1y, aes(intercept=intercept, slope=slope), colour="light green") +
  geom_point(data=centres[unlist(mm0[c2]),], colour="dark blue") +
  geom_point(data=centres[unlist(mm0[c1]),], colour="dark green") +
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
          axis.text.y=element_blank(),axis.ticks=element_blank(),
          axis.title.x=element_blank(),
          axis.title.y=element_blank())
```

For cluster 2 we have to re-compute the displacements as we are using a new quantum `quantum2`=6.3833866. This passes statistical tests.

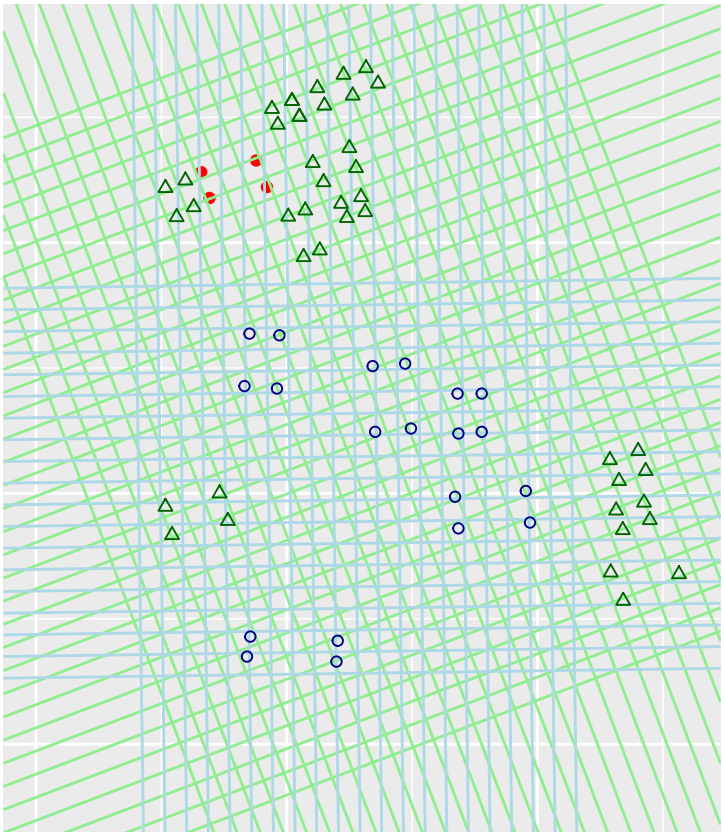

Figure 14: Fitted grids for the plot of building corners: cluster 1 uses green triangles and cluster 2 uses open blue circles.

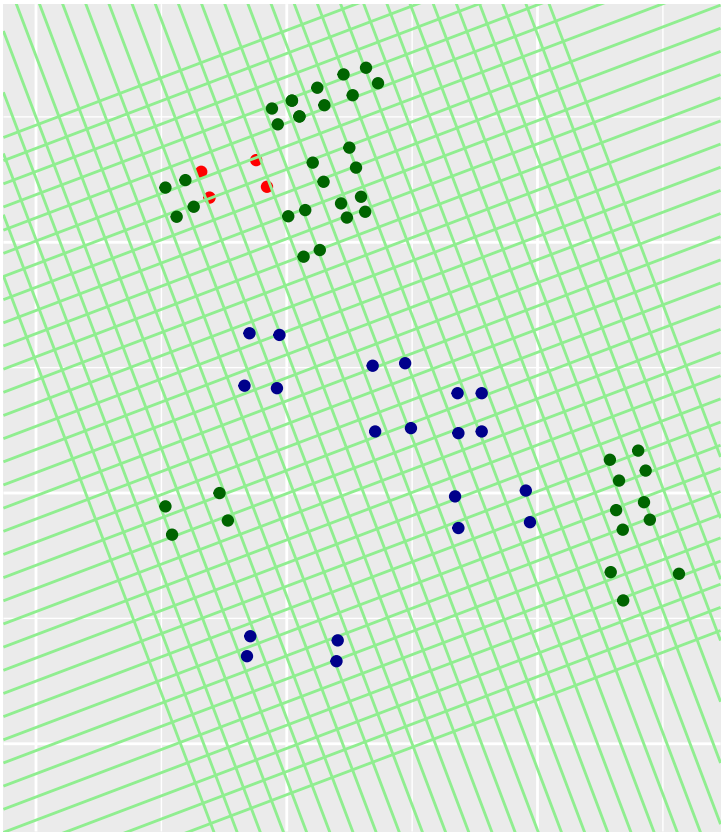

Figure 15: Fitted grids for the plot of building corners, working with cluster 1 (green) only.

```
vm2xa <- mle.vonmises(circular((2 * pi / quantum2) * (locs2x %% quantum2)))
vm2ya <- mle.vonmises(circular((2 * pi / quantum2) * (locs2y %% quantum2)))
c2resxa <- vM.GoF(circular((2 * pi / quantum2) * (locs2x %% quantum2)), vm2xa$mu, vm2xa$kappa)
```

```
##
##        Watson's Test for the von Mises Distribution
##
## Test Statistic: 0.0461
## P-value > 0.10
##
##
##        Kuiper's Test of Uniformity
##
## Test Statistic:  1.0144
## P-value > 0.15
##
##
##        Watson's Test for Circular Uniformity
##
## Test Statistic: 0.043
## P-value > 0.10
##
##
##        Rao's Spacing Test of Uniformity
##
## Test Statistic = 120.5348
## P-value > 0.10
##
##
##        Rayleigh Test of Uniformity
##        General Unimodal Alternative
##
## Test Statistic:  0.0751
## P-value:  0.8956
```

```
c2resya <- vM.GoF(circular((2 * pi / quantum2) * (locs2y %% quantum2)), vm2ya$mu, vm2ya$kappa)
```

```
##
##        Watson's Test for the von Mises Distribution
##
## Test Statistic: 0.0578
## P-value > 0.10
##
##
##        Kuiper's Test of Uniformity
##
## Test Statistic:  1.0603
## P-value > 0.15
##
##
##        Watson's Test for Circular Uniformity
##
## Test Statistic: 0.0551
## P-value > 0.10
```

```
##
##
##        Rao's Spacing Test of Uniformity
##
## Test Statistic = 132.5411
## P-value > 0.10
##
##
##        Rayleigh Test of Uniformity
##        General Unimodal Alternative
##
## Test Statistic:  0.1501
## P-value:  0.6429
```

```
disp2xa <- (quantum2 / (2 * pi)) * as.numeric(vm2xa[2])

disp2ya <- (quantum2 / (2 * pi)) * as.numeric(vm2ya[2])

grid2xa <- data.frame(
  intercept = (disp2ya + quantum2 * (10:23)) / cos(theta2),
  slope = rep(tan(theta2), 23 - 10 + 1)
)

grid2ya <- data.frame(
  intercept = (disp2xa + quantum2 * (-18:-5)) / cos(theta2 + pi/2),
  slope = rep(tan(theta2 + pi/2), -5 + 18 + 1)
)

qplot(data=centres[unlist(mm0[!c1 & !c2]),], xm, ym,
      xlim=c(0, 130), ylim=c(40, 190), asp=150/130, colour=I("red")) +
  coord_fixed() +
  geom_abline(data=grid2xa, aes(intercept=intercept, slope=slope), colour="light blue") +
  geom_abline(data=grid2ya, aes(intercept=intercept, slope=slope), colour="light blue") +
  geom_point(data=centres[unlist(mm0[c2]),], colour="dark blue") +
  geom_point(data=centres[unlist(mm0[c1]),], colour="dark green") +
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
          axis.text.y=element_blank(),axis.ticks=element_blank(),
          axis.title.x=element_blank(),
          axis.title.y=element_blank())
```

3.2.4 Using quantum from previous fit

We now repeat the grid exercise using the quantum from fits to foundations of large stone buildings, namely 4.7619048.

```
vm1xp <- mle.vonmises(circular((2 * pi / quantum.previous) * (locs1x %% quantum.previous)))
disp1xp <- (quantum.previous / (2 * pi)) * as.numeric(vm1xp[2])

vm1yp <- mle.vonmises(circular((2 * pi / quantum.previous) * (locs1y %% quantum.previous)))
disp1yp <- (quantum.previous / (2 * pi)) * as.numeric(vm1yp[2])

vm2xp <- mle.vonmises(circular((2 * pi / quantum.previous) * (locs2x %% quantum.previous)))
disp2xp <- (quantum.previous / (2 * pi)) * as.numeric(vm2xp[2])
```

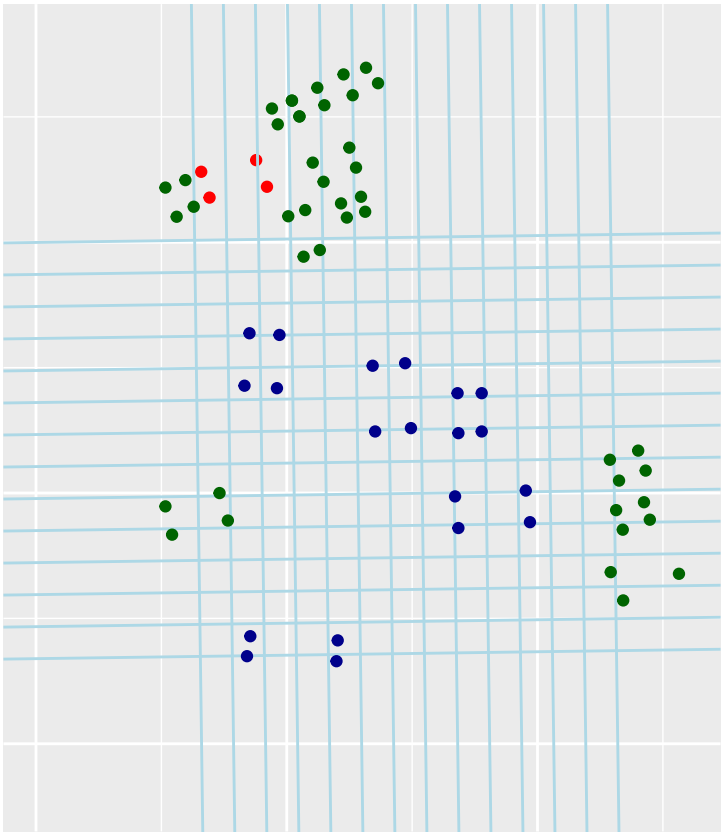

Figure 16: Fitted grids for the plot of building corners, working with cluster 2 (blue) only.

```
vm2yp <- mle.vonmises(circular((2 * pi / quantum.previous) * (locs2y %% quantum.previous)))
disp2yp <- (quantum.previous / (2 * pi)) * as.numeric(vm2yp[2])
```

We now use the modal displacements together with the fitted grid orientations and the `quantum.previous` estimate to add fitted grid lines to the plot of building corners.

```
grid1xp <- data.frame(
  intercept = (disp1yp + quantum.previous * (6:35)) / cos(theta1),
  slope = rep(tan(theta1), 35 - 6 + 1)
)

grid1yp <- data.frame(
  intercept = (disp1xp - quantum.previous * (10:35)) / cos(theta1 + pi/2),
  slope = rep(tan(theta1 + pi/2), 35 - 10 + 1)
)

grid2xp <- data.frame(
  intercept = (disp2yp + quantum.previous * (10:30)) / cos(theta2),
  slope = rep(tan(theta2), 30 - 10 + 1)
)

grid2yp <- data.frame(
  intercept = (disp2xp + quantum.previous * (-23:-5)) / cos(theta2 + pi/2),
  slope = rep(tan(theta2 + pi/2), -5 + 23 + 1)
)

qplot(data=centres[unlist(mm0[!c1 & !c2]),], xm, ym,
      xlim=c(0, 130), ylim=c(40, 190), asp=150/130, colour=I("red")) +
  coord_fixed() +
  geom_abline(data=grid1xp, aes(intercept=intercept, slope=slope), colour="light green") +
  geom_abline(data=grid1yp, aes(intercept=intercept, slope=slope), colour="light green") +
  geom_abline(data=grid2xp, aes(intercept=intercept, slope=slope), colour="light blue") +
  geom_abline(data=grid2yp, aes(intercept=intercept, slope=slope), colour="light blue") +
  geom_point(data=centres[unlist(mm0[c2]),], shape=1, colour="dark blue") +
  geom_point(data=centres[unlist(mm0[c1]),], shape=2, colour="dark green") +
  theme(axis.line=element_blank(),axis.text.x=element_blank(),
          axis.text.y=element_blank(),axis.ticks=element_blank(),
          axis.title.x=element_blank(),
          axis.title.y=element_blank())
```

3.2.5 Conclusion

We have not attempted to estimate the error in determination of the quantum in this exercise. Note from the quantograms that it is clear there are several contenders for the maximum frequency; considering the range of these, it is reasonable to take the view that there is sufficient information to argue for existence of a quantum and of a perpendicular grid, but not sufficient to determine the quantum to accuracy less than of order say 0.25m (consider difference between the compendium estimate 4.32 and the previous estimate 4.7619048).

It is evident from the grid-fitting illustrations that within this range of quanta we may expect fitted grids that match the data moderately well. While strict statistical significance is deficient for the y axis of cluster 1, this is likely to arise from deficiencies of the von Mises distribution. Visually the cluster 1 grid appears to provide a more satisfying fit than the cluster 2 grid.

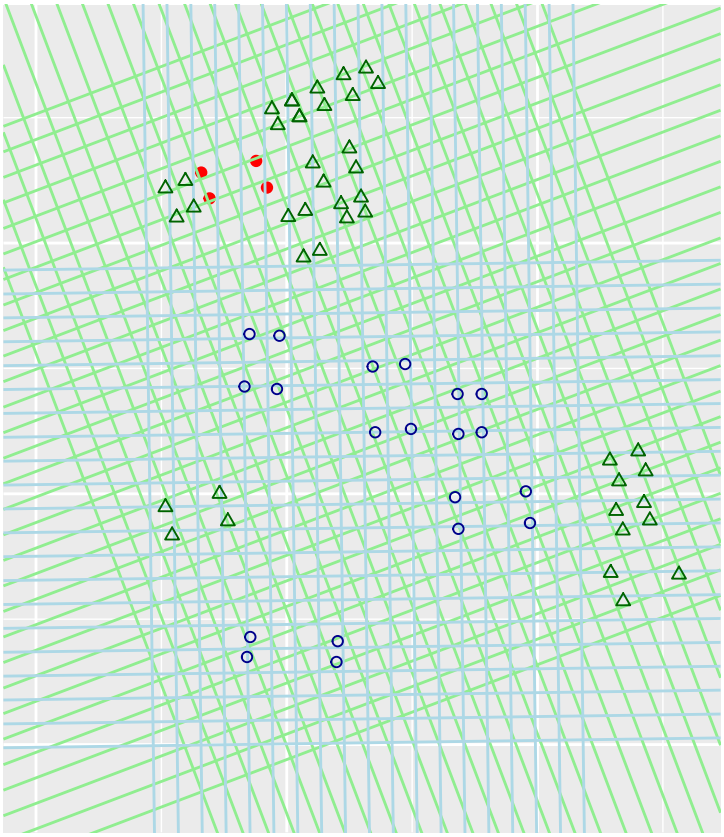

Figure 17: Fitted grids for the plot of building corners using previous quantum: cluster 1 uses green triangles and cluster 2 uses open blue circles.

References

Peter J Huggins. Anglo-Saxon Timber Building Measurements: Recent Results. *Medieval Archaeology*, 35: 6–28, 1991.

Peter J Huggins, Kirsty Rodwell, and Warwick J Rodwell. Anglo-Saxon and Scandinavian building measurements. In P J Drury, editor, *Structural Reconstruction: Approaches to the Interpretation of the Excavated Remains of Buildings*, page 65 pp. B A R British Series 110, 1982. ISBN 978-0-86054-193-6. URL http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Anglo-Saxon+and+Scandinavian+Building+Measurements#0.

Wilfrid Stephen Kendall. Modules for Anglo-Saxon constructions: Appendix to "Grid-Planning in Anglo-Saxon Settlements: the Short Perch and the Four-Perch Module" by John Blair. *Anglo-Saxon Studies in Archaeology and History*, 18:55–57, 2013.