code_supplementary

Table of Contents

Intro

The file contains code for data analysis of the article. Stable isotope analysis in soil prospection reveals the type of historic land-use under contemporary temperate forests in Europe".

All data is available on the github repository.

Supplementary figure 1

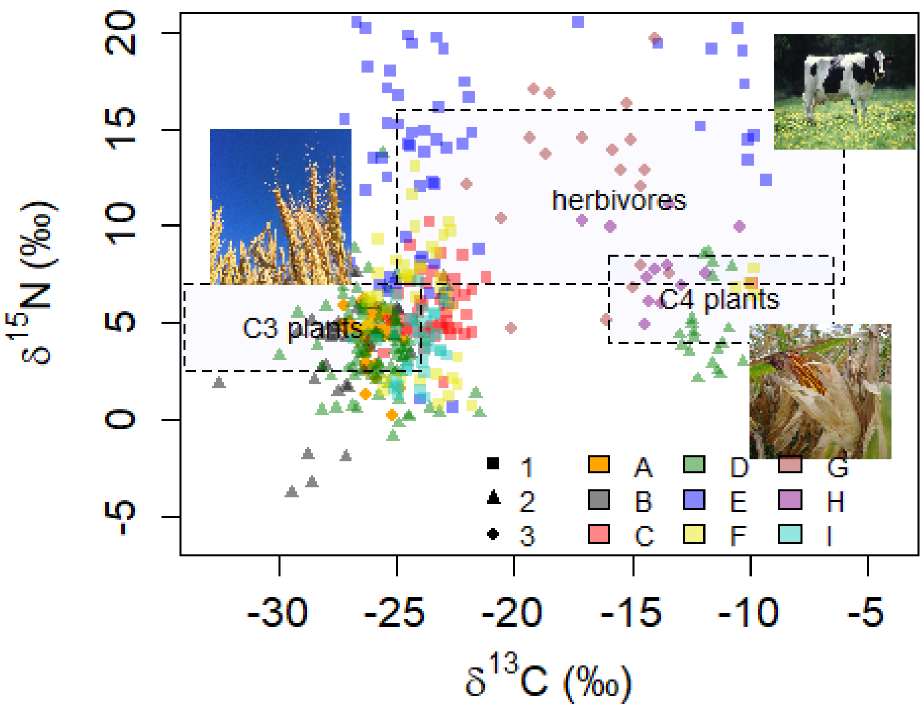
Data loading, all data available at https://github.com/Barilac/SR in S1.txt.

```
raw = read.table(file = "S1.txt", header=TRUE,
sep="\t", dec=".", check.names = F, stringsAsFactors = T)

data = raw[,c(2,3,9,11)]
#install.packages("cowplot")
#install.packages("gridGraphics")
#install.packages("jpeg")
library(cowplot)
library(gridGraphics)
library(jpeg)

# function to create ecosystem boundaries
recttext <- function(xl, yb, xr, yt, text, rectArgs = NULL, textArgs = NULL)
{
  center <- c(mean(c(xl, xr)), mean(c(yb, yt)))
  do.call('rect', c(list(xleft = xl, ybottom = yb, xright = xr, ytop = yt), r
ectArgs))
  do.call('text', c(list(x = center[1], y = center[2], labels = text), textAr
gs))
```

```
}

# Control
levels(data$Category2)

## [1] "A" "B" "C" "D" "E" "F" "G" "H" "I"

# Definition of colors
org <- rgb(255,165,0, max = 255, names = "myorange")
blck <- rgb(0,0,0, max = 255, alpha = 120, names = "myblack")
bl = rgb(0,0,255, max = 255, alpha = 120, names = "myblue")
yl = rgb(230,230,0, max = 255, alpha = 120, names = "myyellow")
rd = rgb(255,0,0, max = 255, alpha = 120, names = "myred")
gr = rgb(0,128,0, max = 255, alpha = 120, names = "mygreen")

c1 = rgb(0,255,0, max = 255, alpha = 120, names = "c1")
c2 = rgb(128,128,0, max = 255, alpha = 120,  names = "c2")
c3 = rgb(165,42,42, max = 255, alpha = 120,  names = "c3")
c4 = rgb(255,248,220, max = 255, alpha = 120,  names = "c4")
c5 = rgb(128,0,128, max = 255, alpha = 120, names = "c5")
c6 = rgb(33,199,188, max = 255, alpha = 120, names = "c6")

barva = c(org,blck, rd, gr,bl,yl,c3,c5,c6)
barva = barva[as.numeric(data$Category2)]

# Define shapes
shapes = c(15, 17, 16)
shapes <- shapes[as.numeric(unlist(data[4]))]

# adding images from wiki
#C3
c3_i = "https://upload.wikimedia.org/wikipedia/commons/6/6c/Melissa_Askew_201
5-08-08_%28Unsplash%29.jpg"
download.file(c3_i,'c3_i.jpg', mode = 'wb')
c3_i <- readJPEG("c3_i.jpg",native=TRUE)
#plot(0:1,0:1,type="n",ann=FALSE,axes=FALSE)
#rasterImage(c3_i,0,0,1,1)

#c4
c4_i = "https://upload.wikimedia.org/wikipedia/commons/6/6f/Klip_kukuruza_uzg
ojen_u_Me%C4%91imurju_%28Croatia%29.JPG"
download.file(c4_i,'c4_i.jpg', mode = 'wb')
c4_i <- readJPEG("c4_i.jpg",native=TRUE)
#plot(0:1,0:1,type="n",ann=FALSE,axes=FALSE)
#rasterImage(c4_i,0,0,1,1)

#herbivores
herb = "https://upload.wikimedia.org/wikipedia/commons/0/0c/Cow_female_black_
white.jpg"
download.file(herb,'herb.jpg', mode = 'wb')
```

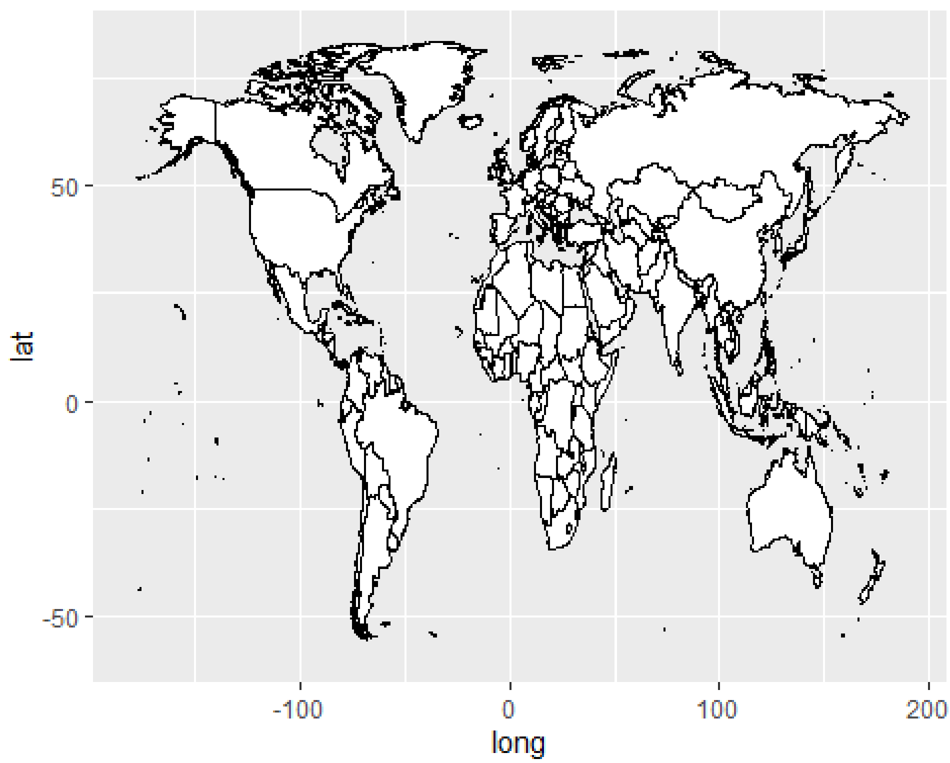
```
herb <- readJPEG("herb.jpg",native=TRUE)
#plot(0:1,0:1,type="n",ann=FALSE,axes=FALSE)
#rasterImage(herb,0,0,1,1

#Plotting Supplementary figure 1 - isotopes
par(mar= c(5,5,1,1))
plot(data$d15N ~ data$d13C, data=data, pch = shapes, col=barva, cex.lab=1.3,
cex.axis = 1.3,
     xlab = expression(paste(delta^13,"C"," (\u2030)")),
     ylab=expression(paste(delta^15,"N"," (\u2030)")), cex=0.8, xlim=c(-33,-4
), ylim=c(-6,20))

# text(data$d15N ~ data$d13C, label = data$Type, cex=0.5)


recttext(-34, 2.5, -24, 7, 'C3 plants',
         rectArgs = list(col =  rgb(0, 0, 255, max = 255, alpha = 4), lty = '
dashed'),
         textArgs = list(col = 'black', cex = .9))

# If you have data for carnivores, they can be inserted.

#recttext(-24, 15, -5, 20, 'carnivores',
#rectArgs = list(col =  rgb(0, 0, 255, max = 255, alpha = 10), lty = 'dashed'
),
#textArgs = list(col = 'black', cex = 0.5))


recttext(-25, 7, -6, 16, 'herbivores',
         rectArgs = list(col =  rgb(0, 0, 255, max = 255, alpha = 4), lty = '
dashed'),
         textArgs = list(col = 'black', cex = .9))

#recttext(-27, 4, -16, 8, 'CAM plants',
#         rectArgs = list(col =  rgb(0, 0, 255, max = 255, alpha = 4), lty =
'dashed'),
#        textArgs = list(col = 'black', cex = 1.4))

recttext(-16, 4, -6.5, 8.5, 'C4 plants',
         rectArgs = list(col =  rgb(0, 0, 255, max = 255, alpha = 4), lty = '
dashed'),
         textArgs = list(col = 'black', cex = .9))

#rastering images, xleft, ybottom, xright, ytop,
rasterImage(c3_i,-33,7,-27,15)
rasterImage(c4_i,-10,-2,-4,5)
rasterImage(herb,-9,14,-3,20)
```

```
legend(-18,-0.5, legend=levels(data$Category2), bty = "n",
       fill=c(org,blck, rd, gr,bl,yl,c3,c5,c6), cex=0.9, ncol=3)

legend(-22,-0.5, legend = c("1","2","3"), bty = "n",
       pch= c(15, 17, 16),  cex=0.9, ncol=1)
```



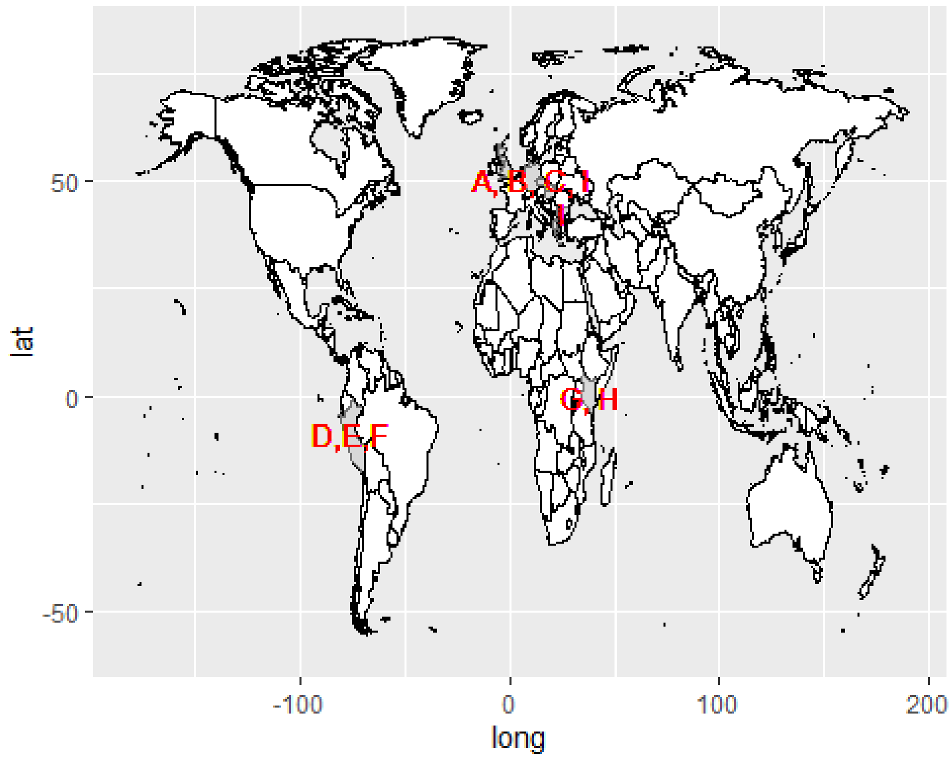

```
p1 <- recordPlot()

# Preparing map - Figure 1
library(ggplot2)

# Loading World data
world <- map_data("world")

# Remove the Antarctica region
world <- subset(world, region != "Antarctica")

# Remove the French Southern and Antarctic Lands region
world <- subset(world, region != "French Southern and Antarctic Lands")

# Select the countries you want to keep
countries <- subset(world, region %in% c("Czech Republic", "Peru","Slovakia",
"Bulgaria",
"Hungary", "Germany","Greece",
"USA","Japan", "Kenya", "UK","Denmark"))
```

```r
# Assign a code to the countries
cze <- subset(countries, region %in% c("Czech Republic"))
svk <- subset(countries, region %in% c("Slovakia"))
usa <- subset(countries, region %in% c("USA"))
ken <- subset(countries, region %in% c("Kenya"))
uk <- subset(countries, subregion %in% c("Great Britain"))
den <- subset(countries, subregion %in% c("Fyn"))
ger <- subset(countries, region %in% c("Germany"))
grc <- subset(countries, region %in% c("Greece"))
bul <- subset(countries, region %in% c("Bulgaria"))
hun <- subset(countries, region %in% c("Hungary"))
per <- subset(countries, region %in% c("Peru"))
bul <- subset(countries, region %in% c("Bulgaria"))
bul <- subset(countries, region %in% c("Bulgaria"))


map <- ggplot(data = world, aes(x = long, y = lat, group = group)) +
  geom_polygon(fill = "white", color = "black", size = 0.3)

map
```

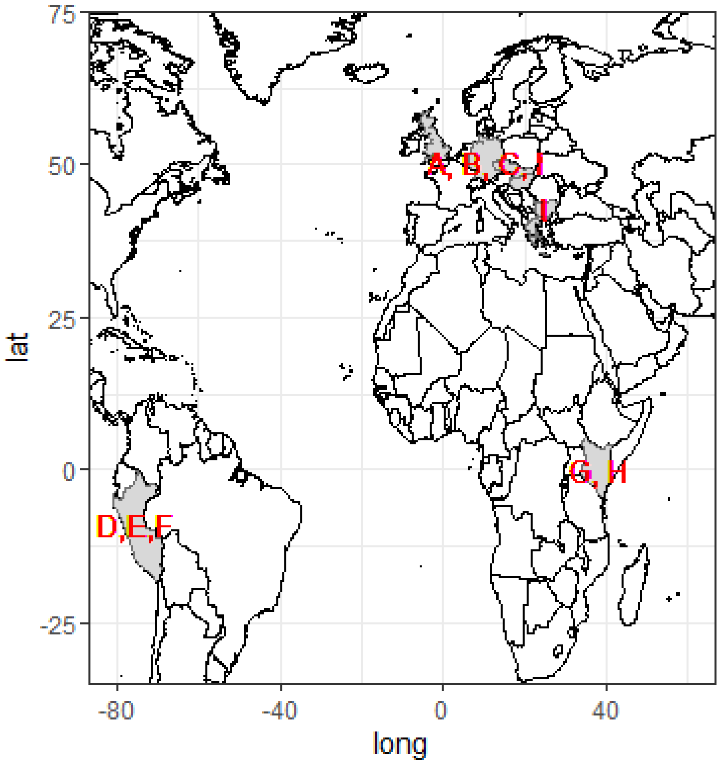



```r
finalmap <- map +
  geom_polygon(data = cze, fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = svk, fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = ken, fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = uk,  fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = den, fill = "grey", size = 0.3, alpha = 0.6) +
```

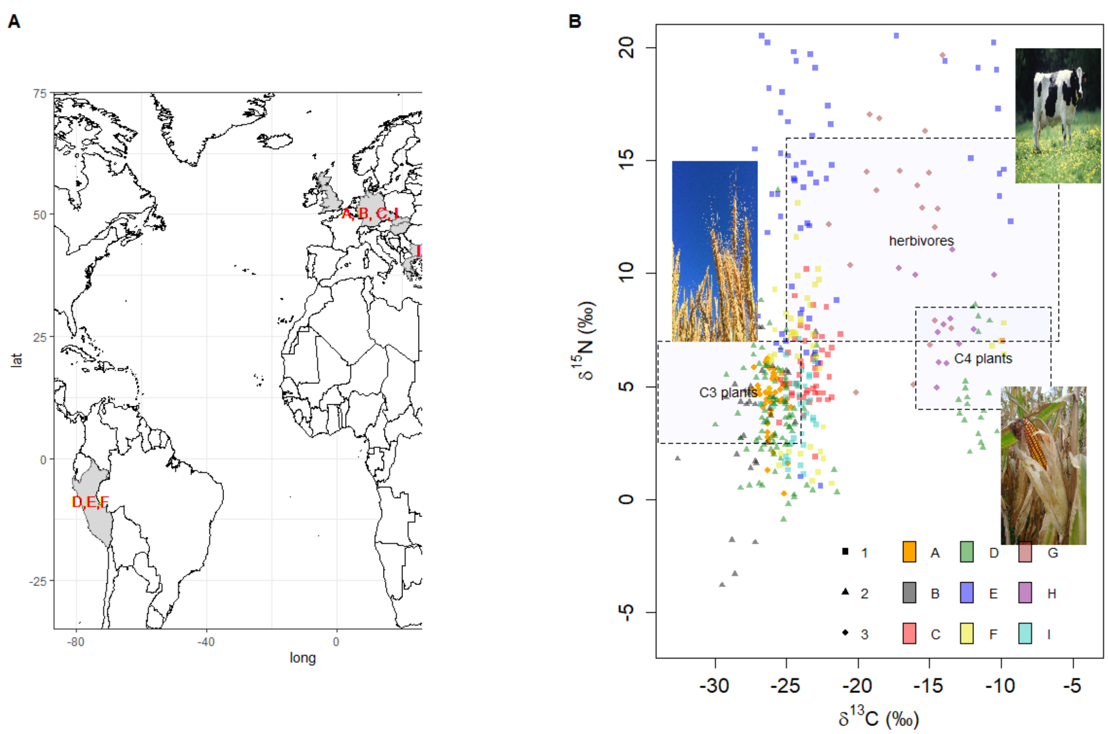
```r
  geom_polygon(data = den, fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = ger, fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = grc, fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = bul, fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = hun, fill = "grey", size = 0.3, alpha = 0.6) +
  geom_polygon(data = per, fill = "grey", size = 0.3, alpha = 0.6) +


  geom_text(aes(x = 10.6115861, y =  50.4462469,
                label = "A, B, C, I"),
            stat = "unique",
            size = 4, color = "red") +

  geom_text(aes(x = 25.3991836, y = 42.6694375,
                label = "I"),
            stat = "unique",
            size = 4, color = "red") +

# geom_text(aes(x = -0.3, y = 52.4720122,
#               label = "I"),
#           stat = "unique",
#           size = 4, color = "red") +

  geom_text(aes(x = -75.6091172, y =   -8.6853950,
                label = "D,E,F"),
            stat = "unique",
            size = 4, color = "red") +

  geom_text(aes(x =   38.3850236, y = 0.0176794,
                label = "G, H"),
            stat = "unique",
            size = 4, color = "red")

finalmap
```




```
p2=
finalmap +
  theme_bw() +
  # Choose the size of your map
  coord_fixed(ratio=1.5, xlim = c(-80,60), ylim = c(-30,70))

p2
```




```
#Plot both plots together
plot_grid(p2, p1, labels = "AUTO")
```



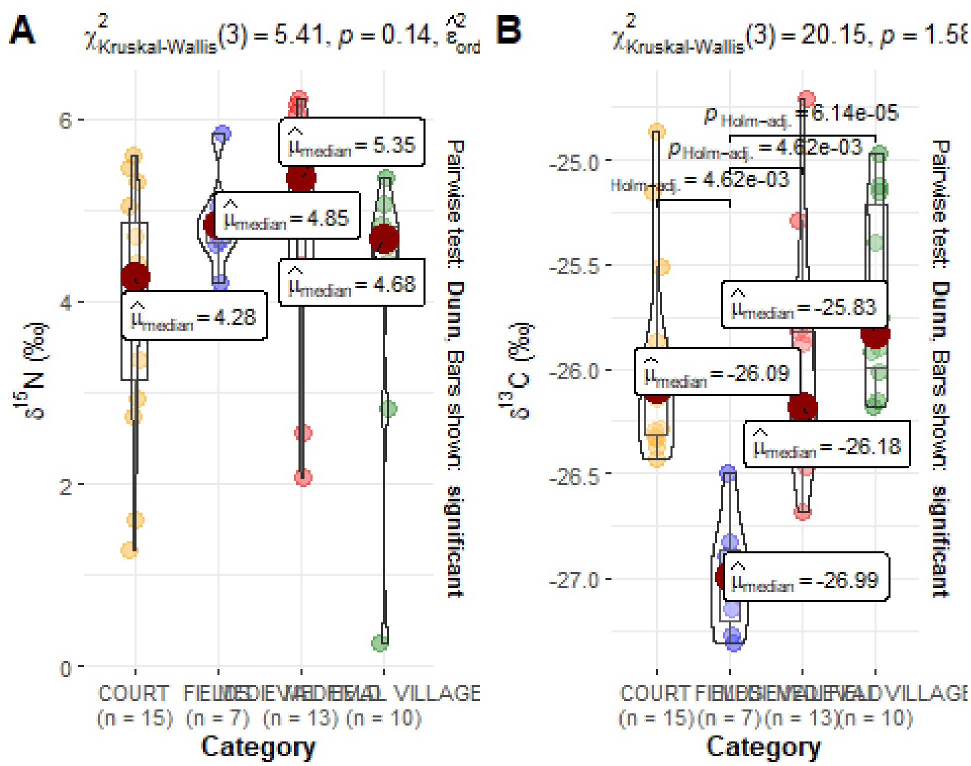
Supplementary figure 1. Isotopic binary diagram of site data (A) and supporting datasets (B-I) featured in this study. In the background: boundaries of ecosystems after Staddon

(79). Symbols show three types of data: 1) archaeobotanical samples, 2) modern plants, 3) soil samples (B). The map shows in grey the countries from where the datasets were obtained (A). Datasets and countries: A-C: Czechia and Slovakia, D-F: Peru, G-H: Kenya, I: Bulgaria, Denmark (Funen island), Germany, Greece, Hungary, United Kingdom.

Supplementary figures 4,5

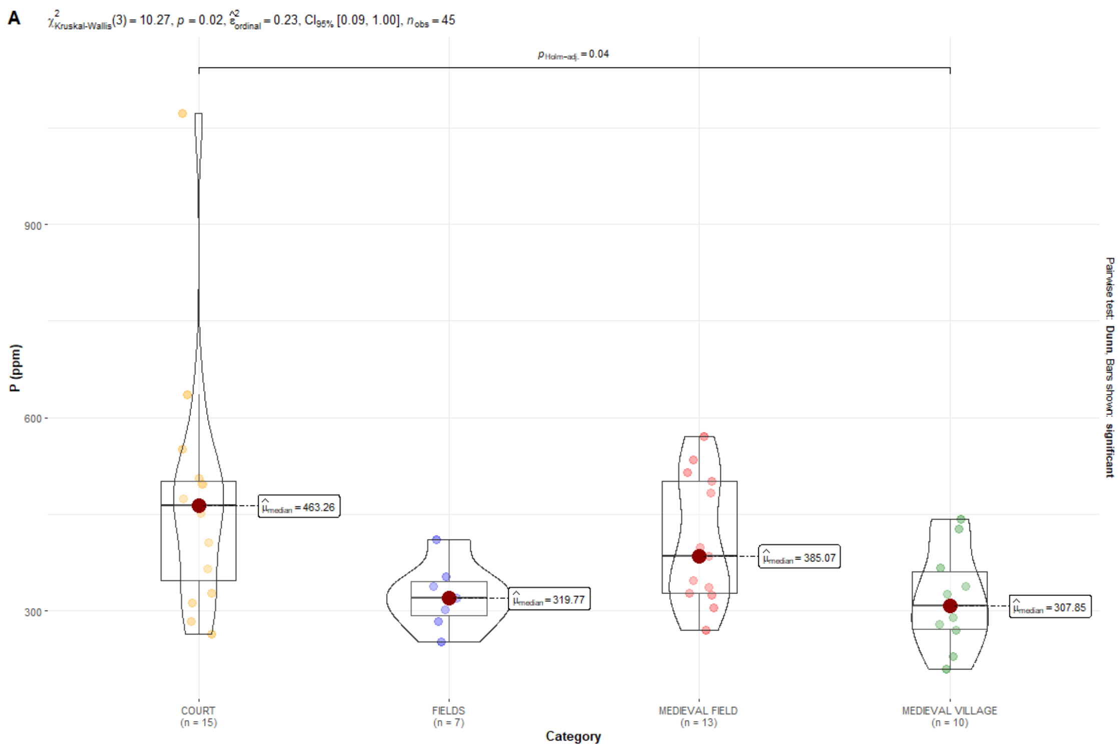
```
raw = read.table(file = "S1.txt", header=TRUE,
sep="\t", dec=".", check.names = F,
stringsAsFactors = T)
library(cowplot)
library(ggstatsplot)
newdata = raw[1:45,c(2,3,7,8,10)]
# Drop levels
levels(newdata$Category)

##  [1] ""                     "bitter vetch"         "COURT"
##  [4] "einkorn"              "einkorn wheat"        "emmer wheat"
##  [7] "FIELDS"               "free-threshing wheat" "grass pea"
## [10] "hulled barley"        "lentil"               "MEDIEVAL FIELD"
## [13] "MEDIEVAL VILLAGE"     "naked barley"         "pea"

levels(droplevels(newdata)$Category)

## [1] "COURT"            "FIELDS"           "MEDIEVAL FIELD"   "MEDIEVAL VIL
LAGE"

newdata$Category <- as.factor(as.character(newdata$Category))

# Definition of colors

org <- rgb(255,165,0, max = 255, alpha = 160, names = "myorange")
bl = rgb(0,0,255, max = 255, alpha = 160, names = "myblue")
rd = rgb(255,0,0, max = 255, alpha = 160, names = "myred")
gr = rgb(0,128,0, max = 255, alpha = 160, names = "mygreen")

# Plotting Figure 4 as boxplot

fi4_1 = ggbetweenstats(
  data = newdata,
  x = Category,
  y = d15N,
  ylab= expression(paste(delta^15,"N"," (\u2030)")),
  type = "nonparametric",
  plot.type = "box",
  pairwise.comparisons = TRUE,
  pairwise.display = "significant",
  centrality.plotting = T,
  bf.message = T
)+
```

```
  ggplot2::scale_color_manual(values = c("#FFA500", "#0000FFA0", "#FF0000A0",
"#008000A0"))

# Drop levels
newdata2=newdata
levels(newdata2$Category)

## [1] "COURT"            "FIELDS"           "MEDIEVAL FIELD"   "MEDIEVAL VIL
LAGE"

levels(droplevels(newdata2)$Category)

## [1] "COURT"            "FIELDS"           "MEDIEVAL FIELD"   "MEDIEVAL VIL
LAGE"

newdata2$Category <- as.factor(as.character(newdata2$Category))
levels(newdata2$Category)

## [1] "COURT"            "FIELDS"           "MEDIEVAL FIELD"   "MEDIEVAL VIL
LAGE"

fi4_2 = ggbetweenstats(
  data = newdata2,
  x = Category,
  y = d13C,
  ylab= expression(paste(delta^13,"C"," (\u2030)")),
  type = "nonparametric",
  plot.type = "box",
  pairwise.comparisons = TRUE,
  pairwise.display = "significant",
  centrality.plotting = T,
  bf.message = T
)+
  ggplot2::scale_color_manual(values = c("#FFA500", "#0000FFA0", "#FF0000A0",
"#008000A0"))

# Plotting Supplementary figure 4 as boxplots
plot_grid(fi4_1, fi4_2, labels = "AUTO")
```



Supplementary figure 4. Isotope values of δ15N and δ13C measured in the area of medieval manor. Categories according to medieval settlement parts and contemporary land use are described in Table 2.

```
# Plotting Supplementary figure 5 as boxplot

fi5 = ggbetweenstats(
  data = newdata,
  x = Category,
  y = P,
  ylab= "P (ppm)",
  type = "nonparametric",
  plot.type = "box",
  pairwise.comparisons = TRUE,
  pairwise.display = "significant",
  centrality.plotting = T,
  bf.message = T
)+
  ggplot2::scale_color_manual(values = c("#FFA500", "#0000FFA0", "#FF0000A0",
"#008000A0"))

# Plotting Supplementary figure 5 as boxplot
plot_grid(fi5, labels = "AUTO")
```

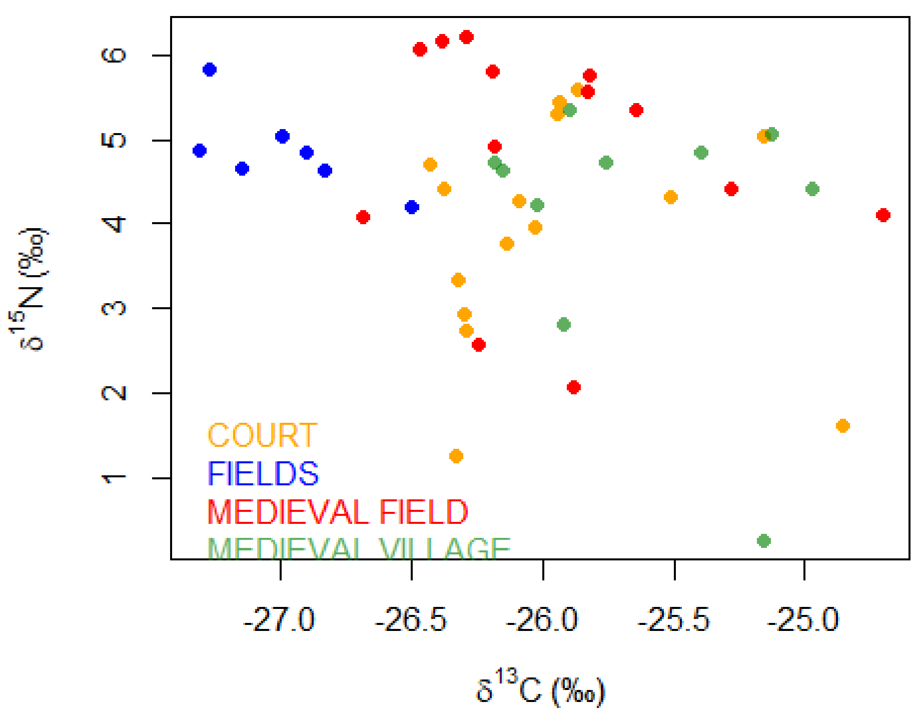


Supplementary figure 5. Phosphorus measured in the area of the medieval manor and settlement and in modern fields. Categories according to medieval settlement parts and contemporary land use are described in Table 2.

Supplementary figure 6

```
raw = read.table(file = "S1.txt", header=TRUE,
sep="\t", dec=".", check.names = F,
stringsAsFactors = T)
raw2 = raw[1:45,2:9]
newdata = raw2
# Drop levels
levels(newdata$Category)

##  [1] ""                     "bitter vetch"         "COURT"
##  [4] "einkorn"              "einkorn wheat"        "emmer wheat"
##  [7] "FIELDS"               "free-threshing wheat" "grass pea"
## [10] "hulled barley"        "lentil"               "MEDIEVAL FIELD"
## [13] "MEDIEVAL VILLAGE"     "naked barley"         "pea"

levels(droplevels(newdata)$Category)

## [1] "COURT"            "FIELDS"           "MEDIEVAL FIELD"   "MEDIEVAL VIL
LAGE"
```

```r
newdata$Category <- as.factor(as.character(newdata$Category))
data = newdata

gr = rgb(0,128,0, max = 255, alpha = 160, names = "mygreen")

barva2 = c("orange", "blue","red", gr)
barva = barva2[as.numeric(data$Category)]

# Plotting Supplementary figure 6
par(mar= c(5,5,1,1))
plot(data$d15N ~ data$d13C, data = data, pch = 16,col=barva,
     xlab = expression(paste(delta^13,"C"," (\u2030)")),
     ylab=expression(paste(delta^15,"N"," (\u2030)")), cex=1)
legend(-27.5, 2, legend=levels(data$Category),
       text.col = c("orange", "blue","red", gr),bty = "n",
       cex=1, box.col = "white")
```



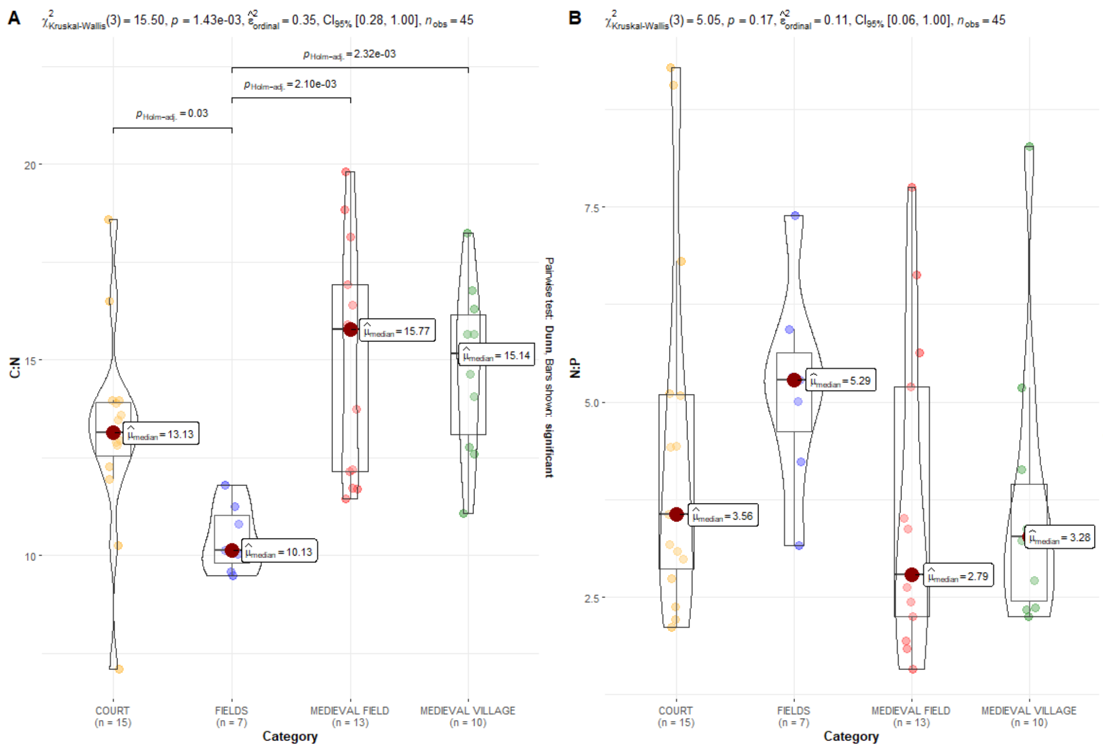

```r
#text(data$d15N ~ data$d13C, label = data$Category, cex=0.5)
```

Supplementary figure 6. Isotopic binary diagram of values in different parts of the manor and settlement and in modern fields. Categories according to medieval settlement parts and contemporary land use are described in Table 2.

Supplementary figure 7

```
library(cowplot)
library(ggstatsplot)
raw = read.table(file = "S1.txt", header=TRUE,
sep="\t", dec=".", check.names = F,
stringsAsFactors = T)

newdata = raw[1:45,c(2,3,7,8,10)]
# Drop levels
levels(newdata$Category)

##  [1] ""                    "bitter vetch"         "COURT"
##  [4] "einkorn"             "einkorn wheat"        "emmer wheat"
##  [7] "FIELDS"              "free-threshing wheat" "grass pea"
## [10] "hulled barley"       "lentil"               "MEDIEVAL FIELD"
## [13] "MEDIEVAL VILLAGE"    "naked barley"         "pea"

levels(droplevels(newdata)$Category)

## [1] "COURT"            "FIELDS"           "MEDIEVAL FIELD"   "MEDIEVAL VIL
LAGE"

newdata$Category <- as.factor(as.character(newdata$Category))

fi7_1 = ggbetweenstats(
  data = newdata,
  x = Category,
  y = C_N_ratio,
  ylab= "C:N",
  type = "nonparametric",
  plot.type = "box",
  pairwise.comparisons = TRUE,
  pairwise.display = "significant",
  centrality.plotting = T,
  bf.message = T
)+
  ggplot2::scale_color_manual(values = c("#FFA500", "#0000FFA0", "#FF0000A0",
"#008000A0"))

#N:P ratio
# selecting medieval site
raw2 = raw[1:45,]

# Phosphorus to percentage, see pXRF method in Methodology

phos = raw2[10]/10000

N_P = raw2[5]

np = N_P/phos
```

```r
colnames(np) = "N_P_ratio"

data2 = cbind(raw2, np)

newdata2 = data2

# Drop levels
levels(droplevels(newdata2)$Category)

## [1] "COURT"            "FIELDS"           "MEDIEVAL FIELD"   "MEDIEVAL VIL
LAGE"

newdata2$Category <- as.factor(as.character(newdata2$Category))

fi7_2 = ggbetweenstats(
  data = newdata2,
  x = Category,
  y = N_P_ratio,
  ylab= "N:P",
  type = "nonparametric",
  plot.type = "box",
  pairwise.comparisons = TRUE,
  pairwise.display = "significant",
  centrality.plotting = T,
  bf.message = T
)+
  ggplot2::scale_color_manual(values = c("#FFA500", "#0000FFA0", "#FF0000A0",
"#008000A0"))

# Plotting Supplementary figure 7 as boxplots
plot_grid(fi7_1, fi7_2, labels = "AUTO")
```

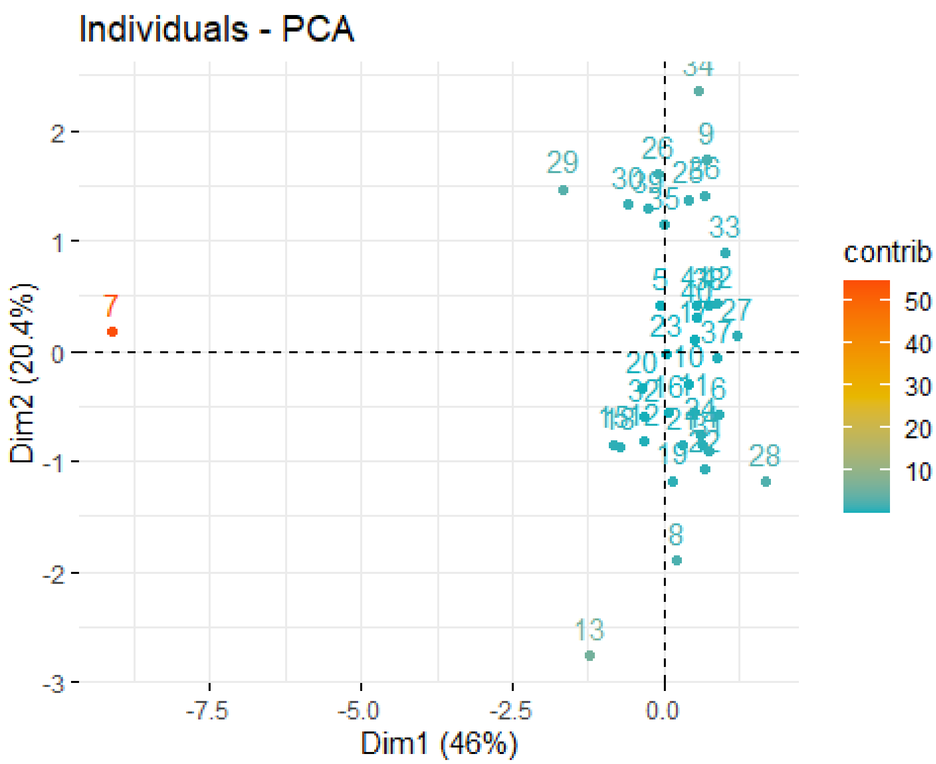


Supplementary figure 7. C:N (A) and N:P (B) ratios measured in different parts of the medieval site and in modern fields. Categories according to medieval settlement parts and contemporary land use are described in Table 2.

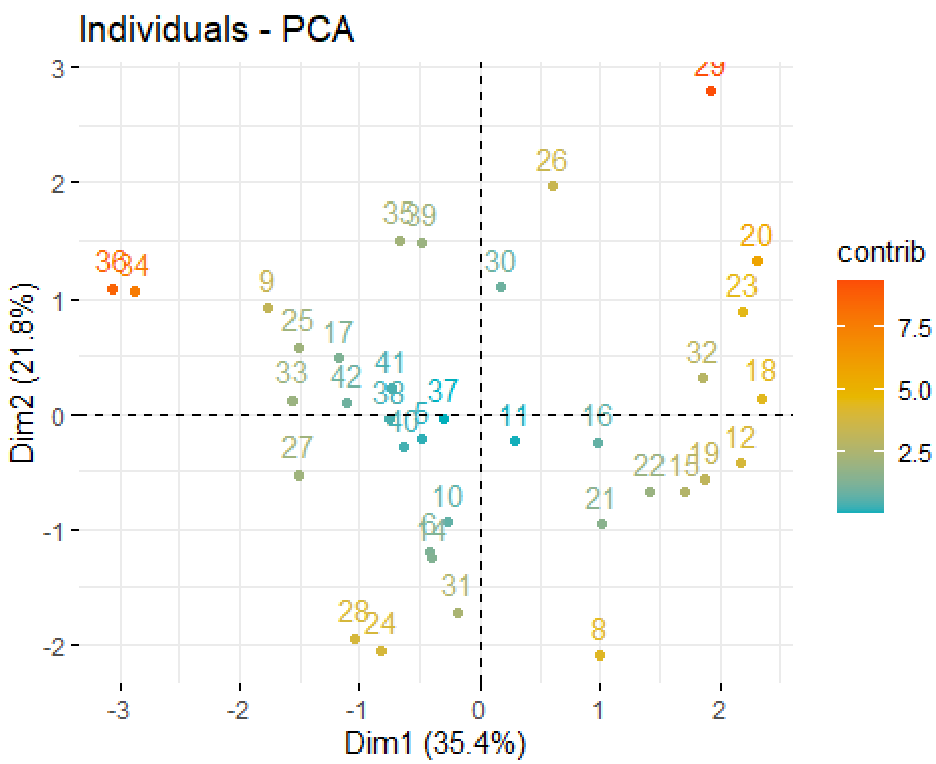
Supplementary figure 8

```
raw = read.table(file = "S1.txt", header=TRUE,
sep="\t", dec=".", check.names = F,
stringsAsFactors = T)

raw2 = raw[1:45,c(2,3,5,6,7,8,10)]

# Removing category FIELDS - too different to include in PCA
# We want to compare only archaeological site with its medieval fields.
raw3 = raw2[-c(1,2,3,4,43,44,45),]
newdata = raw3

# Drop levels
levels(newdata$Category)

##  [1] ""                 "bitter vetch"        "COURT"
##  [4] "einkorn"          "einkorn wheat"       "emmer wheat"
##  [7] "FIELDS"           "free-threshing wheat" "grass pea"
## [10] "hulled barley"    "lentil"              "MEDIEVAL FIELD"
## [13] "MEDIEVAL VILLAGE" "naked barley"        "pea"

levels(droplevels(newdata)$Category)
```

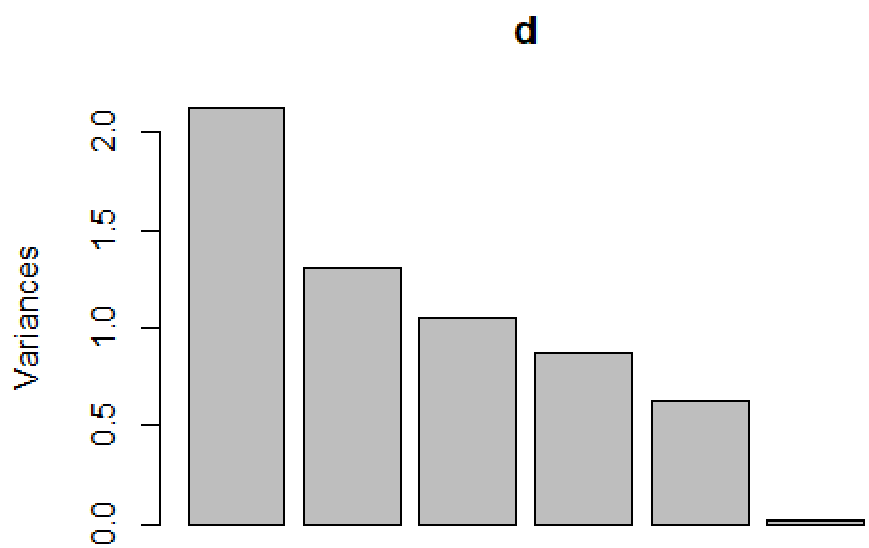
```
## [1] "COURT"            "MEDIEVAL FIELD"   "MEDIEVAL VILLAGE"

newdata$Category <- as.factor(as.character(newdata$Category))
levels(newdata$Category)

## [1] "COURT"            "MEDIEVAL FIELD"   "MEDIEVAL VILLAGE"

str(newdata)

## 'data.frame':    38 obs. of  7 variables:
##  $ d13C     : num  -25.5 -25.9 -24.9 -26.4 -25.2 ...
##  $ d15N     : num  4.33 5.59 1.6 4.7 5.04 ...
##  $ %N       : num  0.12 0.13 0.73 0.17 0.09 0.11 0.16 0.23 0.43 0.1 ...
##  $ %C       : num  1.67 1.55 13.49 1.7 1.44 ...
##  $ C_N_ratio: num  13.9 11.9 18.6 10.3 16.5 ...
##  $ Category : Factor w/ 3 levels "COURT","MEDIEVAL FIELD",..: 1 1 1 1 1 1
1 1 1 1 ...
##  $ P        : num  506 365 1073 551 284 ...

data_select = newdata[,c(1,2,3,4,5,7)]

data_normalized <- scale(data_select)
require(factoextra)

## Warning: package 'factoextra' was built under R version 4.1.3

pca = prcomp(data_normalized)
fviz_pca_ind(pca,
             col.ind = "contrib", # Color by contribution
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07") #assign gradi
ent
)
```




```
# removing 2 outliers, archaeological features

outlier1 = data_select[-c(3,9),] # dataset with removed outliers

categories = newdata[-c(3,9),] # dataset with categories

# testing again
test <- scale(outlier1)

pca = prcomp(test)
fviz_pca_ind(pca,
             col.ind = "contrib", # Color by contribution
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07") #assign gradi
ent
)
```



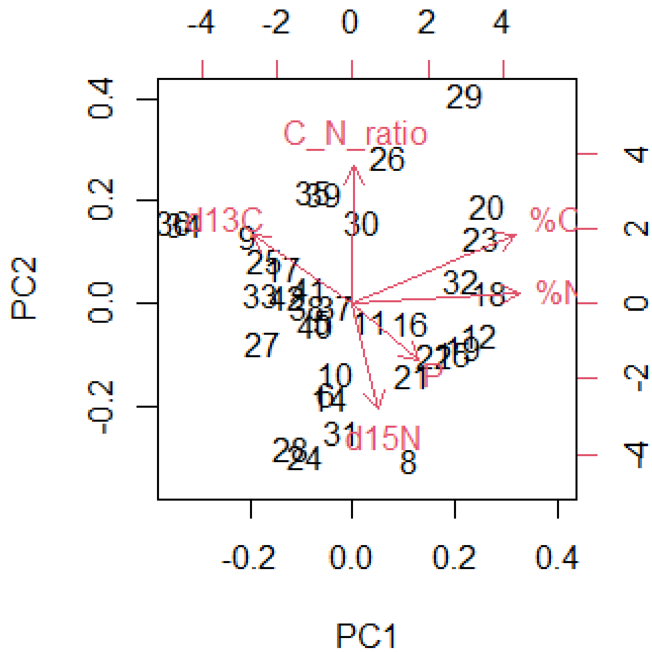

```
data_normalized2 = scale(outlier1)

d = prcomp(data_normalized2, scale=F, center=T) #scale = F
plot(d)
```



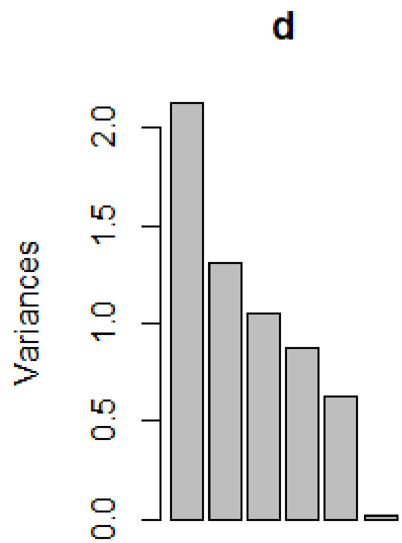

```
biplot(d)

# Summary
s <- summary(d)
s

## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5     PC6
## Standard deviation     1.4583 1.1434 1.0257 0.9355 0.7923 0.10560
## Proportion of Variance 0.3544 0.2179 0.1754 0.1459 0.1046 0.00186
## Cumulative Proportion  0.3544 0.5723 0.7477 0.8935 0.9981 1.00000

unclass(d)

## $sdev
## [1] 1.4582501 1.1434084 1.0257133 0.9355209 0.7922662 0.1055980
##
## $rotation
##                    PC1         PC2        PC3         PC4         PC5
## d13C      -0.380993324  0.33212969 -0.3569804  0.32242258 -0.71629640
## d15N       0.095136042 -0.51188572 -0.2230276  0.80309566  0.18455333
## %N         0.633893979  0.04880028 -0.3243795 -0.08167116 -0.18225359
## %C         0.617983201  0.33577948 -0.1508866  0.10185914 -0.05949162
## C_N_ratio  0.006527736  0.66258313  0.4051124  0.46615493  0.31480335
## P          0.249049788 -0.27114047  0.7285045  0.12936447 -0.56302998
##                      PC6
## d13C      -0.0078787451
```

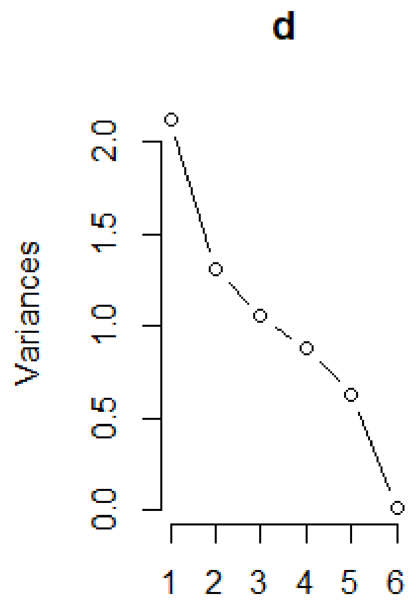
```
## d15N       0.0125813680
## %N        -0.6713332761
## %C         0.6845930247
## C_N_ratio -0.2835900970
## P          0.0005607057
##
## $center
##          d13C          d15N            %N            %C     C_N_ratio
##  2.103159e-15 -2.850247e-17 -1.491862e-16  7.285839e-17 -1.923616e-16
##             P
## -2.185752e-16
##
## $scale
##        d13C         d15N           %N           %C    C_N_ratio
P
##   0.47130622   1.35442101   0.05320446   0.77187136   2.37519221 107.23145
282
##
## $x
##           PC1         PC2         PC3         PC4          PC5           P
C6
## 5  -0.4866037 -0.21446428  0.59731525  0.29554336 -1.178720530  0.01032362
62
## 6  -0.4149165 -1.19211157 -0.66963995  0.19753086 -0.013207234  0.03466107
47
## 8   0.9986312 -2.08890402  0.59995776 -0.85657839 -0.626940401 -0.13429682
82
## 9  -1.7712969  0.91416570 -0.62811312  1.20560955  0.006998213 -0.12126994
06
## 10 -0.2750025 -0.94048328  0.77557324  0.55808894 -0.302741632  0.08143741
58
## 11  0.2858183 -0.23404212 -0.74133794 -1.04219647  0.325021465  0.01968641
67
## 12  2.1684566 -0.42590477 -0.73513310  0.40437734 -0.714671141 -0.07313998
36
## 14 -0.4091991 -1.24305752  1.04815775 -0.49894646  0.280388455  0.05127801
11
## 15  1.7011101 -0.67190646  1.16038057 -0.34433474 -1.575475980 -0.00353092
65
## 16  0.9746351 -0.24781091 -0.25738653 -0.42457977 -0.143900843 -0.07252714
16
## 17 -1.1832241  0.47857261  0.73319519 -2.39029555  0.675654081  0.00051024
53
## 18  2.3372479  0.14204534  0.36365134 -0.92847804 -0.534794451 -0.01010777
70
## 19  1.8753173 -0.57016662 -1.32171952 -0.47613048  0.005813681 -0.16546603
60
## 20  2.3022995  1.32512263 -1.25713529 -1.42188351  0.508627535  0.04284967
24
## 21  1.0193143 -0.94953294 -1.21796926  0.13122594 -0.545227447 -0.13581537
```

```
71
## 22  1.4101819 -0.67304835 -1.74846324  0.05210700  0.679344462 -0.09311317
79
## 23  2.1874771  0.88740629 -0.23988872  1.51970791  1.466312462  0.19309742
97
## 24 -0.8179847 -2.05014090  0.95718342  0.53901490 -1.000079550  0.15189889
27
## 25 -1.5067531  0.57656687 -0.95313931  0.53211646 -0.980369536  0.05107007
03
## 26  0.6154821  1.96476564 -1.91619167  1.29376199 -0.189599628  0.07408762
31
## 27 -1.5211941 -0.52921364 -0.74507328 -0.15600028  0.421602853  0.11956709
24
## 28 -1.0447339 -1.94095793 -0.09563606  0.06577105  1.396119239  0.14146437
11
## 29  1.9166847  2.79321850  1.72001702  0.03323201 -0.641531416  0.16198898
67
## 30  0.1670486  1.09346171  0.62955454  1.14054301 -1.224110357  0.05660121
97
## 31 -0.1890278 -1.71753759  1.08741173  0.84460860  0.271649026  0.04463511
79
## 32  1.8506350  0.30409520  1.27848253 -0.13346396  0.755223099  0.05308804
30
## 33 -1.5607813  0.11956826 -0.20753522  0.93313576  0.623013328 -0.03758925
54
## 34 -2.8817562  1.06221891 -0.35123489  0.86668214 -0.962878101 -0.14837342
94
## 35 -0.6636135  1.50697908  1.75125019 -0.38722744  1.010309528 -0.26054726
73
## 36 -3.0625133  1.08640052 -1.02325657 -2.84041783 -0.859672873  0.15041813
42
## 37 -0.2950720 -0.03591645 -0.33878705 -0.03056929  1.109955161  0.07480876
15
## 38 -0.7602330 -0.04153570 -1.55362005  0.11488391 -0.304084129  0.02841735
36
## 39 -0.4832010  1.48912705  1.54479841 -0.05795865  0.188114178 -0.08855811
17
## 40 -0.6443775 -0.29333761  0.63188578  0.89094164  0.268757413 -0.06654043
32
## 41 -0.7325308  0.21807793  0.49713472  0.06868611  0.609187340 -0.03988027
50
## 42 -1.1063249  0.09828042  0.62531134  0.30149236  1.195913733 -0.09113359
74

biplot(d)
```




```
# Screeplot
layout(matrix(1:2, ncol=2))
screeplot(d)
screeplot(d, type="lines")
```

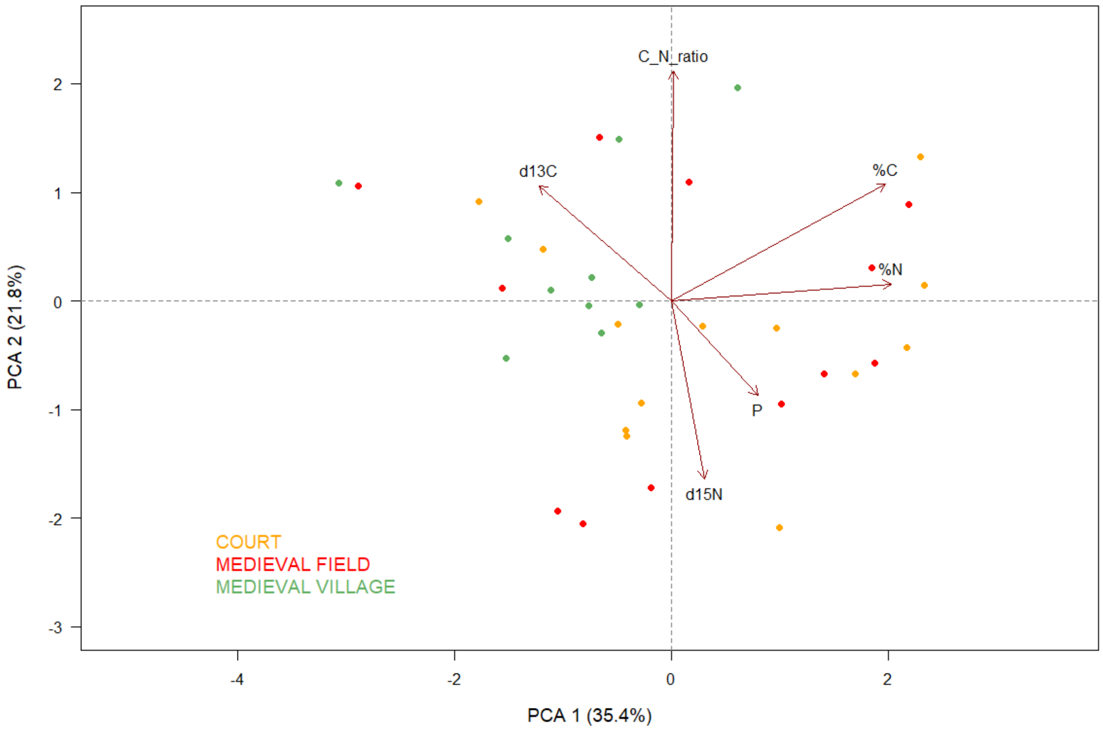





```
gr = rgb(0,128,0, max = 255, alpha = 160, names = "mygreen")

barva = c("orange", "red", gr)
barva = barva[as.numeric(categories$Category)]

# Export PCA results for Table 3
library("writexl")
sum1 = as.data.frame(print(d[["rotation"]], digits = 3))

##                PC1     PC2    PC3     PC4     PC5       PC6
## d13C      -0.38099  0.3321 -0.357  0.3224 -0.7163 -0.007879
## d15N       0.09514 -0.5119 -0.223  0.8031  0.1846  0.012581
## %N         0.63389  0.0488 -0.324 -0.0817 -0.1823 -0.671333
## %C         0.61798  0.3358 -0.151  0.1019 -0.0595  0.684593
## C_N_ratio  0.00653  0.6626  0.405  0.4662  0.3148 -0.283590
## P          0.24905 -0.2711  0.729  0.1294 -0.5630  0.000561

sum1 = sum1[1:2]
sum1$variable <- rownames(sum1)
sum2 = as.data.frame(summary(d)$importance)
sum2 = sum2[1:2]
sum2$parameter <- rownames(sum2)

sum3 = as.data.frame(d$sdev^2)


write_xlsx(sum1,"loadings.xlsx")
```

```r
write_xlsx(sum2,"pca_summary.xlsx")
write_xlsx(sum3,"pca_eigenvalues.xlsx")

# Plotting Supplementary figure 8
par(mar=c(4.5,4.5,1,1))
plot(d$x[,1], d$x[,2], xlab=paste("PCA 1 (", round(s$importance[2]*100, 1), "
%)", sep = ""),
     ylab=paste("PCA 2 (", round(s$importance[5]*100, 1), "%)", sep = ""),
     pch=16, las=1, asp=1, ylim = c(-3,2.5),
     xlim = c(-4,2.5),cex=1,  col=barva, cex.lab=1.2)

s$importance #importance of components

##                              PC1      PC2      PC3       PC4       PC5
PC6
## Standard deviation     1.45825 1.143408 1.025713 0.9355209 0.7922662 0.105
598
## Proportion of Variance 0.35442 0.217900 0.175350 0.1458700 0.1046100 0.001
860
## Cumulative Proportion  0.35442 0.572310 0.747660 0.8935300 0.9981400 1.000
000

# Add grid lines
abline(v=0, lty=2, col="grey50", lwd=1.5)
abline(h=0, lty=2, col="grey50", lwd=1.5)

# Add labels / optional
#text(d$x[,1], d$x[,2], labels=categories$Category,pos=c(1,3,4,2), font=2, ce
x=0.3)

# Get co-ordinates of variables (loadings), and multiply
l.x <- d$rotation[,1]*3.2
l.y <- d$rotation[,2]*3.2

# Draw arrows
arrows(x0=0, x1=l.x, y0=0, y1=l.y, col="darkred", length=0.1, lwd=1)

# Label position
l.pos <- l.y # Create a vector of y axis coordinates
lo <- which(l.y < 0) # Get the variables on the bottom half of the plot
hi <- which(l.y > 0) # Get variables on the top half
# Replace values in the vector
l.pos <- replace(l.pos, lo, "1")
l.pos <- replace(l.pos, hi, "3")

# Variable labels
text(l.x, l.y, labels=row.names(d$rotation),
     col="black", pos=l.pos, cex=1)
```

```
legend(-4.5, -2, legend=levels(categories$Category),
       text.col = c("orange", "red", gr),bty = "n",
       cex=1.2, box.col = "white")
```



Supplementary figure 8. Ordination diagram showing results of PCA analysis comparing different parts of the medieval site. Categories according to medieval settlement parts and contemporary land use are described in Table 2.