CODECHECK certificate 2020-025

https://doi.org/10.5281/zenodo.4279275

Item	Value
Title	The application of Local Indicators for Categorical Data (LICD) to explore spatial dependence in archaeological spaces
Authors	Francesco Carrer , Tomasz M. Kossowski , Justyna Wilk , Michał B. Pietrzak , Roger S. Bivand
Reference	https://doi.org/10.1016/j.jas.2020.105306
Codechecker	Daniel Nüst
Date of check	2020-11-19 12:00:00
Summary	This workflow was very straightforward to check following the authors' README. All figures stored in the repository could be recreated and match the ones given in the manuscript.
Repository	https://github.com/codecheckers/LICD_article

Table 1: CODECHECK summary

Output	Comment	Size (b)
`Torridge_HLC.jpeg`	Figure 3	498208
`Torridge_LICD.jpeg`	Figure 4	176077
`jc_out.csv`	data for Table 2	2168
`HLC_output.Rout`	Verbatim console output of LICD-Archaeo-HCL.R captured by sink, including messages	12334
`barmose_jc_out.csv`	data for Table 3	738
`Barmose_Grid_Cores.jpeg`	Figure 5	154525
`Barmose_LICD_class.jpeg`	Figure 6	144274
`Grid_output.Rout`	Verbatim console output of LICD-Archaeo-Grid.R captured by sink, including messages	12226

Table 2: Summary of output files generated

Summary

This workflow was very straightforward to check following the authors' README. All figures stored in the repository could be recreated and match the ones given in the manuscript.

CODECHECKER notes

I forked the GitHub repo provided by the authors into the codecheckers organisation: codecheckers/LICD_article. This evaluation is based on the commit `2665bb2761eb00d8fd929d390682c76e628c4f8b`. The authors deposited the contents of the repository on Zenodo at https://doi.org/10.5281/zenodo.4283766.

The repository contains some R code and a number of `.jpeg` files. Data is downloaded as part of the scripts.

I went through the steps in the README, and only had to install a few packages that were not available on my system yet. Using the provided `DESCRIPTION` file, I installed dependencies with `devtools::install_deps()`. Thanks to the solid documentation within the scripts and the readable code, I also knew to use the development version of the package `mapview`.

This took around 25 minutes to complete on my laptop (8 cores, 40 GB RAM, SSD). From the diff between the captures outputs, it's quite easy to see the differences in the computing environments, but that no relevant problems occurred during execution. I briefly confirmed the workflow also runs in an `rocker/geospatial:4.0.3` container.

Recommendations to authors

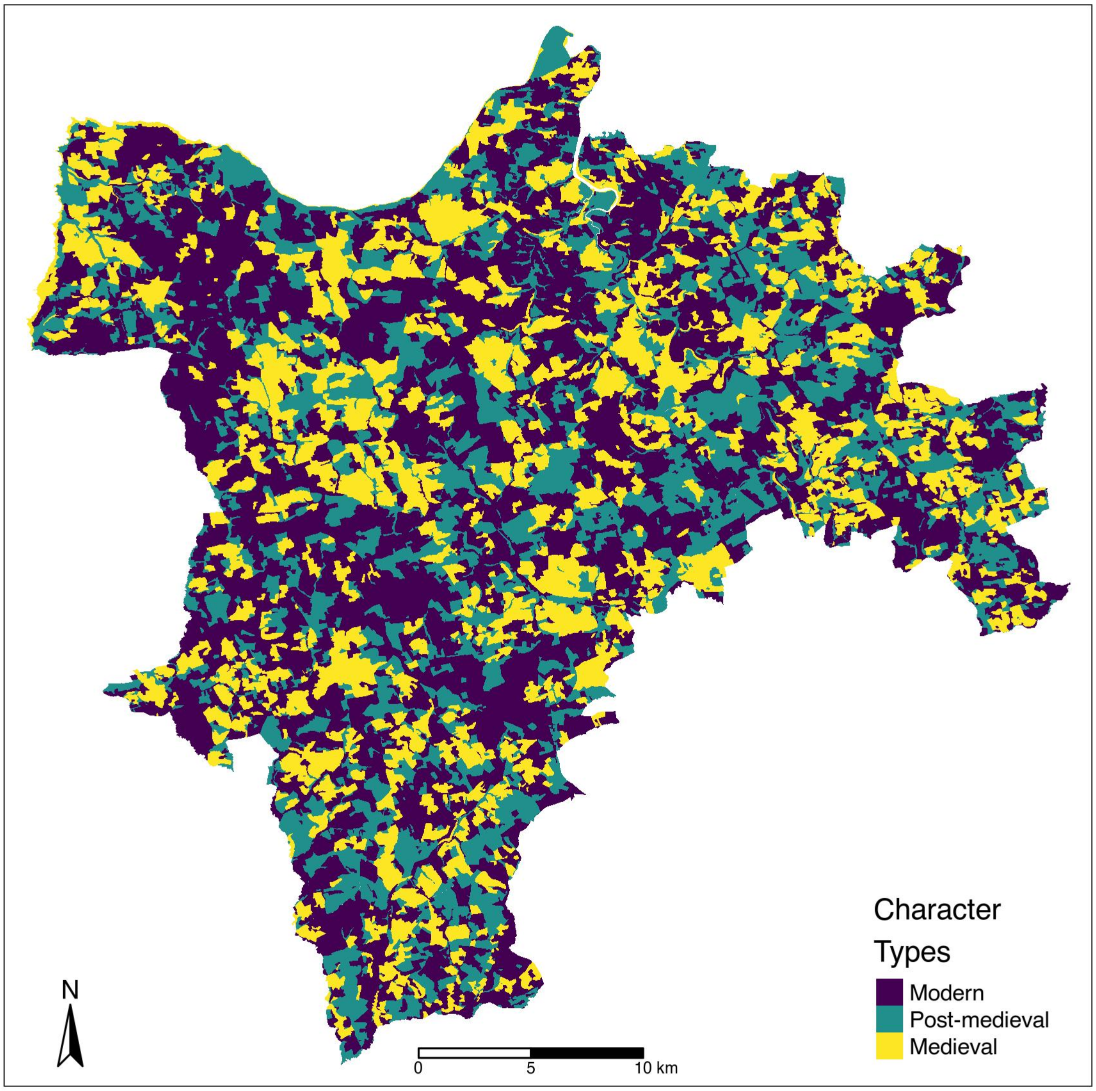
The code was well documented and worked flawlessly on my local machine, there are a few things the authors could consider for their next workflow publication, some of which the authors already picked up in a revision of their repository.

- (authors added `DESCRIPTION` file) use `renv` package or a `DESCRIPTION` file to define the dependencies (= use R package structure)
- (authors updated README) document in README that development version of package `mapview` is needed
- (authors updated README) the interactive map is a bonus compared to the paper - mention how to get to it in the README (the `.zip` file)
- (authors created Zenodo record) deposit a copy of your repo on Zenodo to make the workflow citable and archived
- The data could possibly be republished with the workflow just to be sure, since it is available under open licenses; I can't judge the long term availability of the used download URLs

Manifest files

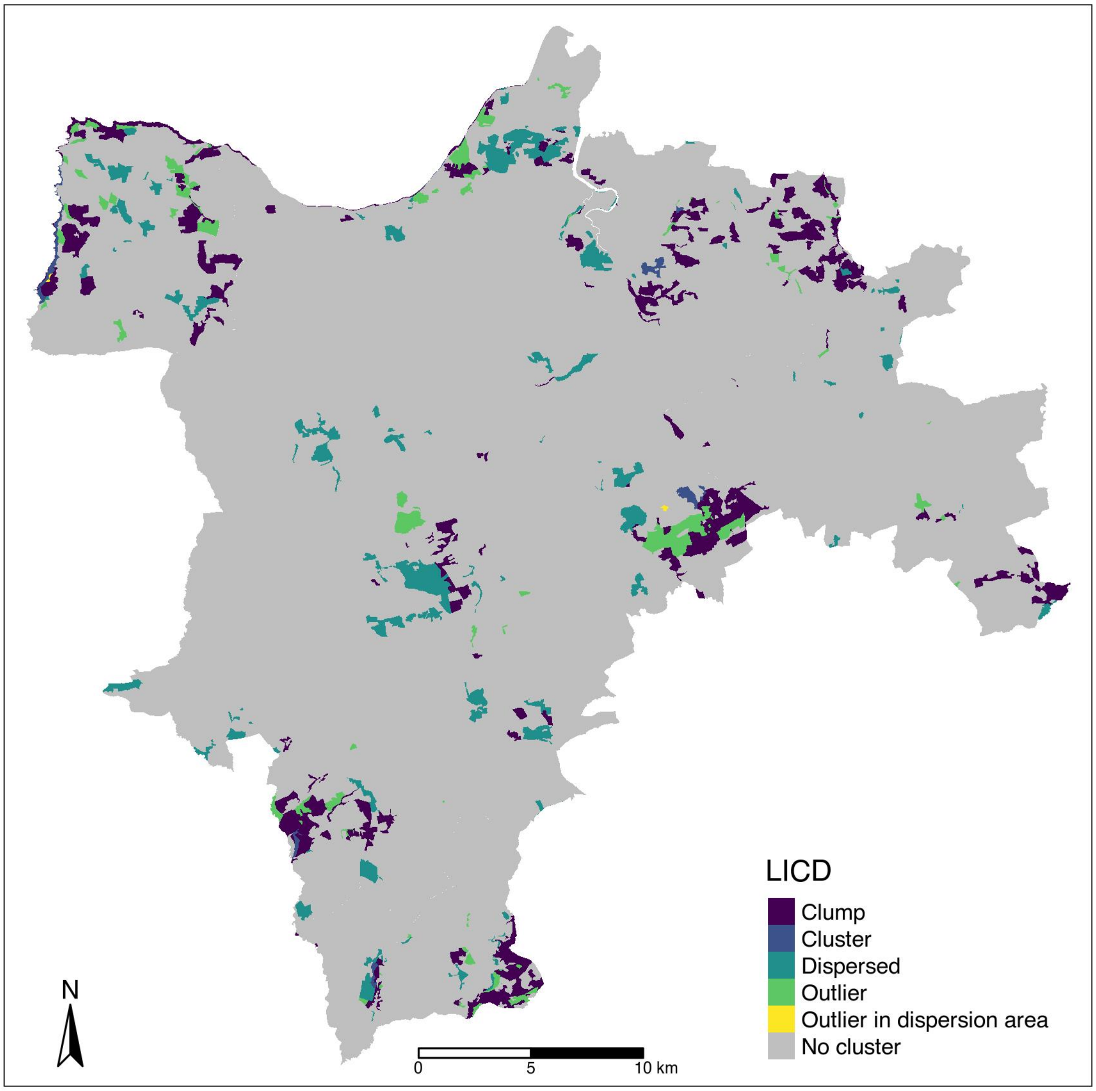
Torridge_HLC.jpeg

Comment: Figure 3



Torridge_LICD.jpeg

Comment: Figure 4



jc_out.csv

Summary statistics of tabular data:

```
-- Data Summary ------------------------
                          Values
Name                       read.csv(path)
Number of rows             18
Number of columns          7
-----------------------
Column type frequency:
 character                2
 numeric                  5
------------------------
Group variables            None

-- Variable type: character --------------------------------
 skim_variable n_missing complete_rate   min   max empty
1 order                 0             1     5     6     0
2 JCS                   0             1    13    27     0
 n_unique whitespace
1        3          0
2        6          0

-- Variable type: numeric ----------------------------------
 skim_variable n_missing complete_rate    mean      sd
1 Joincount             0             1 524.    353.
2 Expected              0             1 524.    374.
3 Variance              0             1  34.6    37.0
4 z.value               0             1   1.22    6.04
5 pvalue                0             1   0.416   0.443
        p0         p25     p50     p75   p100 hist
1  5.48e+ 1 309.        422.    811.    1138.
2  5.32e+ 1 289.        407.    839.    1147.
3  1.57e+ 0  10.3        18.9    46.8    139.
4 -8.68e+ 0  -3.07        0.665   5.24    11.3
5  4.08e-30   0.0000241   0.254   0.995    1
```

HLC_output.Rout

```
> sink(zz, type = "message")

> # Loading libraries
> library(sf)

> library(spdep)

> library(tmap)

> library(units)

> library(Matrix)

> # Importing Devon HLC shapefile
> zipfile <- "https://archaeologydataservice.ac.uk/catalogue/adsdata/arch-2090-1/dissemination/zip/rawhlc.zip"

> ## subject to https://archaeologydataservice.ac.uk/advice/termsOfUseAndAccess.xhtml
> ## It can be cited by https://doi.org/10.5284/1032952
> 
> td  .... [TRUNCATED] 

> download.file(zipfile, destfile=file.path(td, "rawhlc.zip"))
trying URL 'https://archaeologydataservice.ac.uk/catalogue/adsdata/arch-2090-1/dissemination/zip/rawhlc.zip'
Content type 'application/zip' length 23188797 bytes (22.1 MB)
==================================================
downloaded 22.1 MB


> fls <- unzip(file.path(td, "rawhlc.zip"), exdir=td, overwrite=TRUE)

> devon_hlc <- st_read(file.path(td, "rawhlc.shp"), crs=27700)
Reading layer `rawhlc' from data source `/tmp/RtmpDaLJtN/rawhlc.shp' using driver `ESRI Shapefile'
Simple feature collection with 49485 features and 36 fields
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 220832 ymin: 34922.78 xmax: 337602.9 ymax: 148712.1
projected CRS:  OSGB 1936 / British National Grid

> ## clip to Torridge District boundary
> 
> torridge_bys <- "https://raw.githubusercontent.com/digital-land/boundary-collection/master/collection/loc ..." ... [TRUNCATED] 

> ## subject to https://www.nationalarchives.gov.uk/doc/open-government-licence/version/3/
> bdy <- st_read(torridge_bys)
Reading layer `index' from data source `https://raw.githubusercontent.com/digital-land/boundary-collection/master/collection/local-authority/E0700
Simple feature collection with 1 feature and 10 fields
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: -4.680689 ymin: 50.64654 xmax: -3.883846 ymax: 51.20254
geographic CRS: WGS 84

> ## initially created with sf linked to GEOS 3.9.0dev, with OverlayNG
> ## without OverlayNG, intersection fails because of topology errors
> ## req .... [TRUNCATED] 

> ## Create ordered factor
> 
> 
> hlc$class <- ifelse(hlc$PERIOD1 %in% c("Modern", "Post-medieval"), hlc$PERIOD1, "Medieval")

> hlc$class <- ordered(hlc$class, levels=c("Modern", "Post-medieval", "Medieval"))

> ## Map classes
> 
> HLC_map <- tm_shape(hlc) +
+   tm_fill("class", palette="viridis", title="Character\nTypes") +
+   tm_compass(position=c("left ..." ... [TRUNCATED] 

> # tiff("Torridge_HLC.tiff", width=15, height=15, units="cm", res=300)
> # HLC_map
> # dev.off()
> jpeg("Torridge_HLC.jpeg", width=15, height=15, uni .... [TRUNCATED] 

> HLC_map

> dev.off()
null device 
          1 
```

```
> ## Create neighbours
> nb1 <- poly2nb(hlc, snap=4, row.names=as.character(hlc$ID))

> hlc.nb <- nblag(nb1, 3) ## higher orders

> hlc.mat <- as(nb2listw(nblag_cumul(hlc.nb), style="B"), "CsparseMatrix")

> ## Join-Count Statistics
>
> jc.hlc <- vector(mode="list", length=length(hlc.nb))

> jc.hlc.p <- vector(mode="list", length=length(hlc.nb))

> for (i in 1:length(hlc.nb)) {
+   jc.hlc[[i]] <- joincount.multi(hlc$class, nb2listw(hlc.nb[[i]]))
+   jc.hlc.p[[i]] <- pnorm(jc.hlc[[i]][,4], lower .... [TRUNCATED]

> ## Exporting output
>
> jcs <- do.call("rbind", jc.hlc)[-c(7, 14, 21),]

> jcps <- do.call("c", jc.hlc.p)[-c(7, 14, 21)]

> (jc_out <- data.frame(order=rep(c("First", "Second", "Third"), each=6), JCS=rownames(jcs), as.data.frame(cbind(jcs, pvalue=jcps)), row.names=NUL
   order                         JCS  Joincount   Expected   Variance     z.value       pvalue
1   First               Modern:Modern  771.12427  839.00207 102.175599 -6.71512609 1.000000e+00
2   First Post-medieval:Post-medieval  345.60261  391.89976  59.981162 -5.97787566 1.000000e+00
3   First           Medieval:Medieval   54.80680   53.16675   9.973976  0.51930739 3.017732e-01
4   First        Post-medieval:Modern 1091.05970 1147.26492 139.398052 -4.76045268 9.999990e-01
5   First             Medieval:Modern  514.64644  422.73095  65.680018 11.34154148 4.084633e-30
6   First      Medieval:Post-medieval  365.76018  288.93556  48.634518 11.01610665 1.597925e-28
7  Second               Modern:Modern  839.46923  839.00207  30.997060  0.08390832 4.665647e-01
8  Second Post-medieval:Post-medieval  418.01951  391.89976  18.034020  6.15067428 3.857711e-10
9  Second           Medieval:Medieval   71.31331   53.16675   2.955733 10.55508654 2.405980e-26
10 Second        Post-medieval:Modern 1091.47783 1147.26492  41.291714 -8.68164898 1.000000e+00
11 Second             Medieval:Modern  425.29930  422.73095  19.659496  0.57925259 2.812094e-01
12 Second      Medieval:Post-medieval  297.42082  288.93556  14.480309  2.22985407 1.287857e-02
13  Third               Modern:Modern  824.51919  839.00207  18.046848 -3.40921464 9.996742e-01
14  Third Post-medieval:Post-medieval  394.30714  391.89976  10.147943  0.75571103 2.249112e-01
15  Third           Medieval:Medieval   60.36025   53.16675   1.569199  5.74250544 4.664294e-09
16  Third        Post-medieval:Modern 1137.75125 1147.26492  21.881608 -2.03379977 9.790141e-01
17  Third             Medieval:Modern  435.02278  422.73095  10.870555  3.72812692 9.645411e-05
18  Third      Medieval:Post-medieval  291.03939  288.93556   7.837674  0.75148029 2.261818e-01

> write.csv(jc_out, "jc_out.csv", row.names=FALSE)

> ###########################################
> ## Boots' LICD (from Bivand et al. 2017) ##
> ###########################################
>
> # For h .... [TRUNCATED]
                  [,1]
Modern        0.5167038
Post-medieval 0.3531658
Medieval      0.1301304

> areas <- aggregate(st_area(hlc), list(hlc$class), sum)

> areas$x <- set_units(areas$x, "km2")

> areas$props <- drop_units(areas$x/sum(areas$x))

> areas
       Group.1               x     props
1        Modern 451.1156 [km^2] 0.4593523
2 Post-medieval 271.3623 [km^2] 0.2763170
3      Medieval 259.5909 [km^2] 0.2643307

> adata <- as.numeric(hlc$class) #factor no longer necessary, now numeric

> source("local_JC0.R")

> res <- local_JC0(obj=hlc, lagsmat=hlc.mat, varname="class", numvar=adata, p=p)

> local_comp <- res[[1]]

> JC.pvalue_seq <- res[[2]]

> #### STEP 2: local configuration
>
```

```
> local_config <- matrix(0,length(adata),1)

> colnames(local_config) <- c("cluster-dispersion")

> for (j in 1:length(adata)){#for cluster is 1, for dispersion -1, otherwise 0
+   if (min(JC.pvalue_seq[j,])<1-(1-0.05)^(1/3)){ ###CHANGE
+     ifels .... [TRUNCATED]

> # Combination of local composition and local configuration
> Type <- character(length=length(adata))

> C <- cbind(local_comp, local_config)

> for (i in 1:length(adata)){
+   ifelse(C[i,1] == 1 && C[i,2] == 1, Type[i] <- "Cluster",
+          ifelse(C[i,1] == 1 && C[i,2] == 0, Type[i] <- "C ..." ... [TRUNCATED]

> # Plot LICD - TIFF + JPEG
> Type1 <-  Type

> hlc$Type <- Type

> is.na(Type1) <- Type1 == "No cluster"

> hlc$Type1 <- factor(Type1)

> LICD_map <- tm_shape(hlc) +
+   tm_fill("Type1", palette="viridis", title="LICD", textNA="No cluster") +
+   tm_compass(position=c("left", "bottom ..." ... [TRUNCATED]

> # tiff("Torridge_LICD.tiff", width=15, height=15, units="cm", res=300)
> # LICD_map
> # dev.off()
> jpeg("Torridge_LICD.jpeg", width=15, height=15,  .... [TRUNCATED]

> LICD_map

> dev.off()
null device
         1

> # HLC & LICD
>
> both <- LICD_map + tm_facets("class", nrow=2)

> jpeg("Torridge_HLC_LICD.jpeg",width=30,height=25,units="cm",res=300)

> both

> dev.off()
null device
         1

> # mapview installed from "r-spatial/mapview" after #336 #327 #323
> library(mapview)

> packageVersion("mapview")
[1] ‘2.9.4’

> if (unname(sf_extSoftVersion()["GDAL"]) >= "3.1.0") mapviewOptions(fgb = FALSE)

> file.remove("HLC_map.html")
[1] FALSE

> file.remove("HLC_map.zip")
[1] TRUE

> cl <- mapview(hlc, zcol="class")

> ty <- mapview(hlc, zcol="Type")

> mapshot(cl + ty, url = paste0(getwd(), "/HLC_map.html"))

> zip("HLC_map.zip", "HLC_map.html")

> file.remove("HLC_map.html")
[1] TRUE

> sessionInfo()
R version 4.0.3 (2020-10-10)
```

```
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 20.04 LTS

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/liblapack.so.3

locale:
[1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=de_DE.UTF-8        LC_COLLATE=en_US.UTF-8
[5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=en_US.UTF-8    LC_PAPER=de_DE.UTF-8       LC_NAME=C
[9] LC_ADDRESS=C               LC_TELEPHONE=C             LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] stars_0.4-3         abind_1.4-5         spatstat_1.64-1     rpart_4.1-15        nlme_3.1-150        spatstat.data_1.5-2
[7] archdata_1.2        mapview_2.9.4       Matrix_1.2-18       units_0.6-7         tmap_3.2            spdep_1.1-5
[13] spData_0.3.8        sp_1.4-4            sf_0.9-6

loaded via a namespace (and not attached):
 [1] colorspace_2.0-0        leafem_0.1.3            deldir_0.2-3            ellipsis_0.3.1          class_7.3-17
 [6] leaflet_2.0.3           rprojroot_2.0.2         dadjoke_0.1.2           cranlike_1.0.2          satellite_1.0.2
[11] base64enc_0.1-3         fs_1.5.0                dichromat_2.0-0         rstudioapi_0.13         farver_2.0.3
[16] remotes_2.2.0           bit64_4.0.5             fansi_0.4.1             codetools_0.2-18        splines_4.0.3
[21] knitr_1.30              polyclip_1.10-0         pkgload_1.1.0           jsonlite_1.7.1          tmaptools_3.1
[26] png_0.1-7               compiler_4.0.3          httr_1.4.2              assertthat_0.2.1        cli_2.2.0
[31] leaflet.providers_1.9.0 htmltools_0.5.0         prettyunits_1.1.1       tools_4.0.3             coda_0.19-4
[36] glue_1.4.2              dplyr_1.0.2             rappdirs_0.3.1          gmodels_2.18.1          Rcpp_1.0.5
[41] raster_3.4-5            vctrs_0.3.5             svglite_1.2.3.2         gdata_2.18.0            debugme_1.1.0
[46] leafsync_0.1.0          crosstalk_1.1.0.1       lwgeom_0.2-5            xfun_0.19               stringr_1.4.0
[51] ps_1.4.0                testthat_3.0.0          parsedate_1.2.0         lifecycle_0.2.0         gtools_3.8.2
[56] devtools_2.3.2          goftest_1.2-2           XML_3.99-0.5            LearnBayes_2.15.1       MASS_7.3-53
[61] scales_1.1.1            spatstat.utils_1.17-0   clisymbols_1.2.0        parallel_4.0.3          expm_0.999-5
[66] rematch2_2.1.2          RColorBrewer_1.1-2      yaml_2.2.1              curl_4.3                memoise_1.1.0
[71] gdtools_0.2.2           stringi_1.5.3           RSQLite_2.2.1           desc_1.2.0              leafpop_0.0.6
[76] e1071_1.7-4             crancache_0.0.0.9001    boot_1.3-25             pkgbuild_1.1.0          repr_1.1.0
[81] systemfonts_0.3.2       rlang_0.4.8             pkgconfig_2.0.3         prompt_1.0.0            evaluate_0.14
[86] lattice_0.20-41         tensor_1.5              purrr_0.3.4             htmlwidgets_1.5.2       bit_4.0.4
[91] processx_3.4.4          tidyselect_1.1.0        magrittr_2.0.1          R6_2.5.0                generics_0.1.0
[96] DBI_1.1.0               mgcv_1.8-33             pillar_1.4.7            withr_2.3.0             tibble_3.0.4
[101] crayon_1.3.4            uuid_0.1-4              KernSmooth_2.23-18      rmarkdown_2.5           usethis_1.6.3
[106] grid_4.0.3              blob_1.2.1              callr_3.5.1             git2r_0.27.1            webshot_0.5.2
[111] digest_0.6.27           classInt_0.4-3          brew_1.0-6              stats4_4.0.3            munsell_0.5.0
[116] viridisLite_0.3.0       skimr_2.1.2             sessioninfo_1.1.1

> sf_extSoftVersion()
         GEOS           GDAL         proj.4 GDAL_with_GEOS     USE_PROJ_H
      "3.8.1"        "3.1.3"        "7.1.1"         "true"         "true"

> sink(type = "message")

> sink()
```

barmose_jc_out.csv

Summary statistics of tabular data:

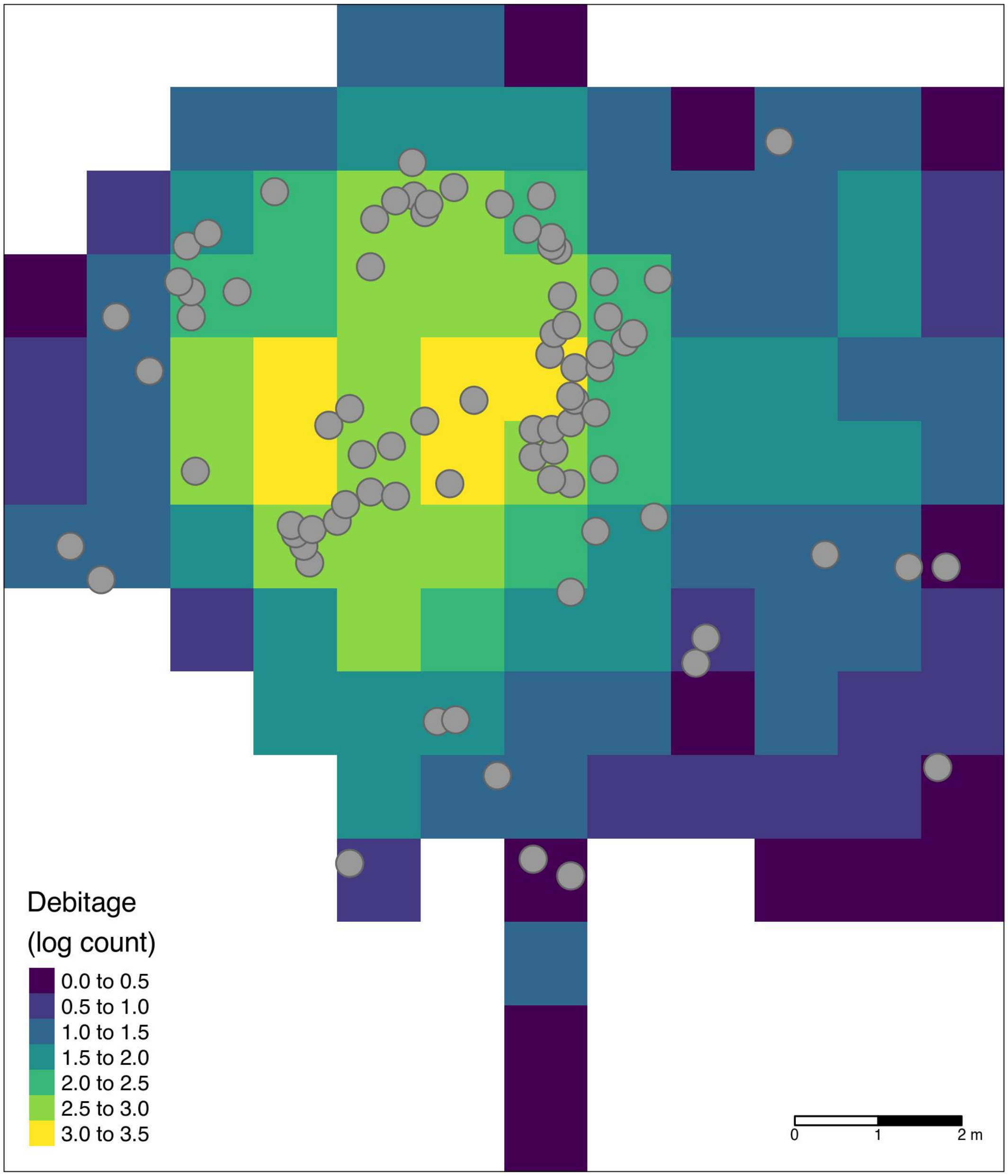
```
-- Data Summary ------------------------
                          Values
Name                       read.csv(path)
Number of rows             6
Number of columns          7
-----------------------
Column type frequency:
 character                2
 numeric                  5
------------------------
Group variables            None

-- Variable type: character --------------------------------
 skim_variable n_missing complete_rate   min   max empty
1 order                 0             1     5     6     0
2 JCS                   0             1     9    15     0
 n_unique whitespace
1        2          0
2        3          0

-- Variable type: numeric ----------------------------------
 skim_variable n_missing complete_rate   mean    sd
1 Joincount             0             1 17.8   7.49
2 Expected              0             1 17.8   8.75
3 Variance              0             1  0.804 0.568
4 z.value               0             1  0.583 2.42
5 pvalue                0             1  0.404 0.451
        p0     p25    p50    p75   p100 hist
1  7.72     11.8    21.3   23.3   24.0
2  6.63     10.5    22.1   24.1   24.7
3  0.335     0.489   0.533  0.977  1.83
4 -3.10     -0.727   0.848  2.37   3.28
5  0.000522  0.0117  0.302  0.757  0.999
```

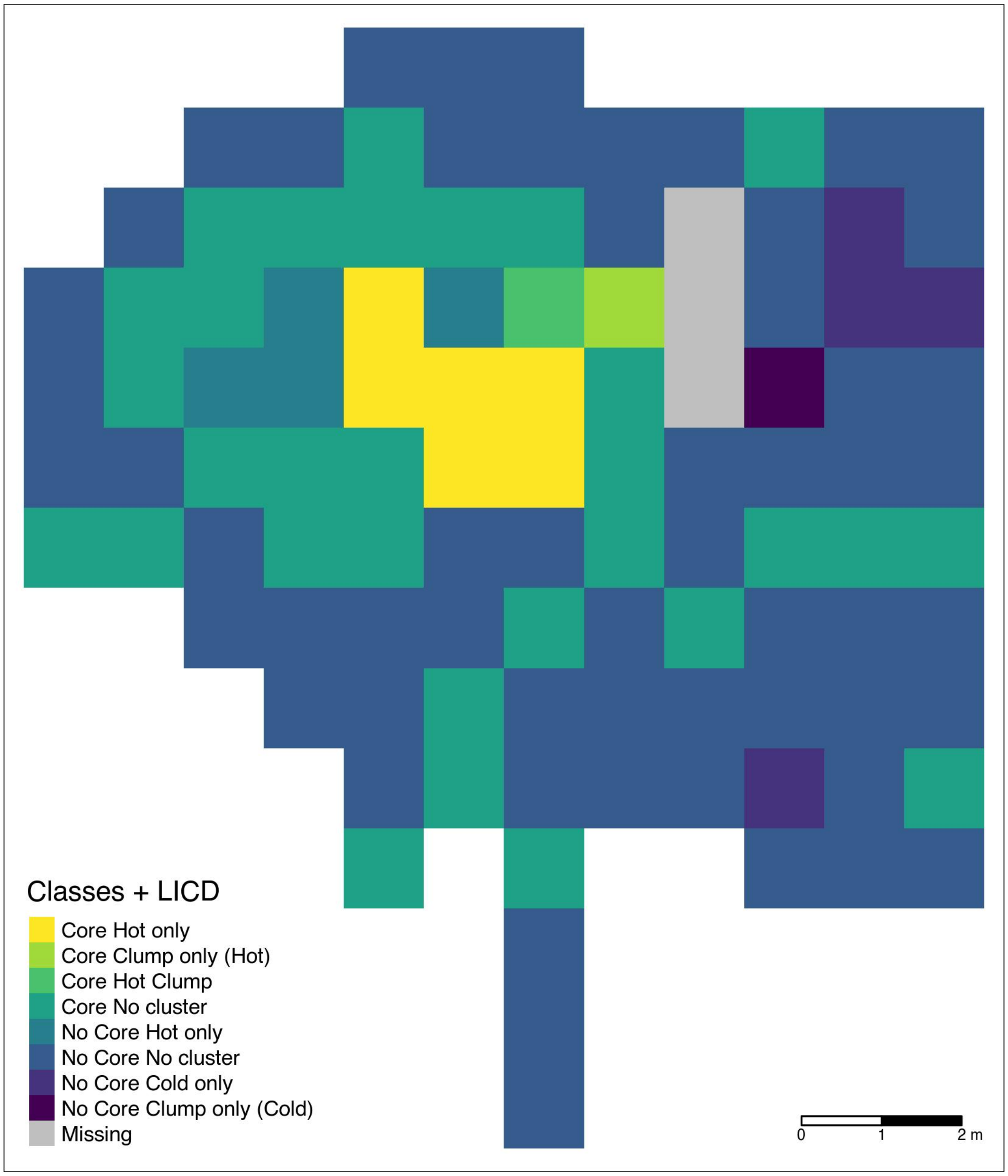
Barmose_Grid_Cores.jpeg

Comment: Figure 5



Barmose_LICD_class.jpeg

Comment: Figure 6



Grid_output.Rout

```
> sink(zz, type = "message")

> # Import data from CRAN
> library(archdata)

> data("BarmoseI.grid")

> data("BarmoseI.pp")

> Easts <- sort(unique(BarmoseI.grid$East))

> Norths <- sort(unique(BarmoseI.grid$North))

> BarmoseI_Cores.pp <- BarmoseI.pp[BarmoseI.pp[, "Label"]=="Cores",]

> library(spatstat)

> BarmoseI_Cores.ppp <- ppp(BarmoseI_Cores.pp$East, BarmoseI_Cores.pp$North, window=owin(xrange=c(0, max(Easts)+1), yrange=c(0, max(Norths)+1)))

> summary(BarmoseI_Cores.ppp)
Planar point pattern:  81 points
Average intensity 0.4821429 points per square unit

Coordinates are given to 2 decimal places
i.e. rounded to the nearest multiple of 0.01 units

Window: rectangle = [0, 12] x [0, 14] units
Window area = 168 square units

> # Convert to stars array
> library(stars)

> packageVersion("stars")
[1] ‘0.4.3’

> BarmoseI.grid1 <- BarmoseI.grid

> if (packageVersion("stars") < "0.4.4") {
+   BarmoseI.grid1$North <- BarmoseI.grid1$North + 1
+ } else {
+   BarmoseI.grid1$North <- BarmoseI.grid1$ .... [TRUNCATED]

> # data registed to SW cell corner, stars bug 0.4-2, 0.4-3 and 0.4-4 before
> # late October 2020; stars >= 0.4-2 required by current tmap;
> # stars .... [TRUNCATED]

> # impose a plate carree projection to satisfy tmap
> st_crs(rast) <- 32662

> rast1 <- rast

> rast1$logp1_Debitage <- log10(rast1$Debitage+1)

> # convert to sf data.frame
> library(sf)

> barmose0 <- st_as_sf(rast)

> barmose <- barmose0[!is.na(barmose0$Debitage),]

> cores <- st_as_sf(BarmoseI_Cores.pp, coords=c("East", "North"))

> # impose a plate carree projection to satisfy tmap
> st_crs(cores) <- 32662

> jpeg("Barmose_Grid_Check.jpeg", width=15, height=17, units="cm", res=300)

> opar <- par(no.readonly=TRUE)

> par(mar=c(0,0,0,0)+0.1)

> plot(BarmoseI_Cores.ppp, chars=24, pt.bg="grey", cex=0.7, legend=FALSE)

> abline(v=Easts, lwd=0.5, lty=2)

> abline(h=Norths, lwd=0.5, lty=2)

> points(North ~ East, BarmoseI.grid, pch=3)
> plot(st_geometry(barmose), add=TRUE, lwd=0.5, border="orange")

> par(opar)

> dev.off()
null device
         1

> library(tmap)

> Log_Deb_map <- tm_shape(rast1, unit="m") +
+   tm_raster("logp1_Debitage", n=7, palette="viridis",
+             title="Debitage\n(log count)") +
.... [TRUNCATED]

> jpeg("Barmose_Grid_Cores.jpeg", width=15, height=17, units="cm", res=300)

> Log_Deb_map

> dev.off()
null device
         1

> barmose$cores <- sapply(st_intersects(barmose, cores), length)

> barmose$class <- factor((barmose$cores>0)+0, levels=c(0, 1), labels=c("No Core", "Core"))

> sum(barmose$cores) # 86 for stars 0.4-4 and using sf::st_intersects
[1] 84

> table(barmose$cores)

0  1  2  3  4  5  6  7
69 19  8  2  5  2  1  1

> #  0  1  2  3  4  5  6  9
> # 69 18 11  2  2  1  3  1
> table(barmose$class)

No Core    Core
    69      38

> # NC  C
> # 69 38
>
> class_map <- tm_shape(barmose, unit="m") +
+   tm_fill("class", palette="viridis") +
+   tm_shape(cores, unit="m") +
+   .... [TRUNCATED]

> jpeg("Barmose_class_Cores.jpeg", width=15, height=17, units="cm", res=300)

> class_map

> dev.off()
null device
         1

> # Create neighbours
>
> ## Contiguity neighbours l-order, l = 3 - USE ONLY WHEN WINDOW IS LARGER THAN 1ST ORDER!
>
> library(spdep)

> nb1 <- poly2nb(barmose)

> barmose.nb <- nblag(nb1, 2) ## higher orders

> barmose.mat <- as(nb2listw(nblag_cumul(barmose.nb), style="B"), "CsparseMatrix")

> # Join-Count Statistics
> ## JC for contiguity
>
> jc.barmose <- vector(mode="list", length=length(barmose.nb))

> jc.barmose.p <- vector(mode="list", length=length(barmose.nb))

> for (i in 1:length(barmose.nb)) {
+   jc.barmose[[i]] <- joincount.multi(barmose$class, nb2listw(barmose.nb[[i]]))
+   jc.barmose.p[[i]] <- pnorm(jc .... [TRUNCATED]
> ## Exporting output
>
> jcs <- do.call("rbind", jc.barmose)[-c(4, 8),]

> jcps <- do.call("c", jc.barmose.p)[-c(4, 8)]

> (jc_out <- data.frame(order=rep(c("First", "Second"), each=3), JCS=rownames(jcs), as.data.frame(cbind(jcs, pvalue=jcps)), row.names=NULL))
  order             JCS Joincount  Expected  Variance    z.value       pvalue
1  First No Core:No Core 24.044048 22.132075 0.5678672  2.5372203 0.0055868304
2  First       Core:Core  8.915476  6.632075 0.4851153  3.2783827 0.0005220186
3  First    Core:No Core 20.540476 24.735849 1.8256763 -3.1049791 0.9990485376
4 Second No Core:No Core 22.001997 22.132075 0.4986978 -0.1841983 0.5730710605
5 Second       Core:Core  7.720851  6.632075 0.3352042  1.8805452 0.0300169049
6 Second    Core:No Core 23.777151 24.735849 1.1140353 -0.9083061 0.8181417412

> write.csv(jc_out, "barmose_jc_out.csv", row.names=FALSE)

> ##########################################
> ## Boots' LICD (from Boots 2003) ##
> ##########################################
> ## Set column with .... [TRUNCATED]

> adata <- factor(barmose[[clm]]) #object with "levels" (factor)

> #### STEP 1: local composition
> p <- (as.matrix(summary(adata)))/length(adata) #probabilities of each "type"

> adata <- as.numeric(adata) #factor no longer necessary, now numeric

>   ## Routine 1 ##
>   # cluster has 5 columns: 1 - number of units of the "type" as the unit j in the "window
>   # 2 - probability of the "type", 3 ..." ... [TRUNCATED]

> c1 <- c2 <- c3 <- c4 <- c5 <- numeric(length(adata))

> for (i in 1:length(adata)) {
+     c1[i] <- (barmose.mat[i,] %*% ifelse(adata==2,1,0))+ifelse(adata[i]==2,1,0)
+     c2[i] <- p[2]
+     c3[i] <- su .... [TRUNCATED]

> cluster <- cbind(c1, c2, c3, c4, c5)

>   cluster[is.nan(cluster)]<- 1

>   ## End of routine 1 ##
>
>   ### Custer-outlier analysis -> result of local composition ###
>   sc <- 1-(1-0.05)^(1/2) #Sidak correction, 0.05  .... [TRUNCATED]

>   local_comp <- ifelse(cluster[,4]< sc, 1, ifelse(cluster[,5]< sc, 0, -1))

>   # 1 for black, 0 for white, -1 - black-white
>
>   #### STEP 2: local configuration
>
>   ## Routine 2  ##
>   ## We built empirical distrib .... [TRUNCATED]

> for (j in 1:length(adata))  {
+     barmose.mat.1 <- barmose.mat[j,] #extracting a row from weights matrix
+     ktore <- which(barmose.mat.1!=0, ar .... [TRUNCATED]

>   ## End of routine 2 ##
>
>   colnames(JC.pvalue_seq) <- c("1:1X>=x","0:0X>=x", "1:0X>=x")

>   local_config <- matrix(3,length(adata),1)

>   scJC <- 1-(1-0.05)^(1/3) # Sidak correction JC - 3 tests!

>   #scJC <- 0.05 #standard
>
>   ### Routine 3 Local configuration
>
>   local_config <- matrix(nrow=length(adata), ncol=1)

> for (j in 1:length(adata)) {#for black is 1, for white is 0, for black-white is -1, otherwise -2
+     if (min(JC.pvalue_seq[j,])<scJC){
+       ife .... [TRUNCATED]
```

```
>   ## End of routine 3 ##
>   colnames(local_config) <- c("cluster-dispersion")

> # Combination of local composition and local configuration
> Type <- character(length(adata))

> C <- cbind(local_comp, local_config)

> for (i in 1:length(adata)){
+   ifelse(C[i,1] == 1 && C[i,2] == 1, Type[i] <- "Hot Clump",
+          ifelse(C[i,1] == 1 && (C[i,2] == -2 || C[i,2]  .... [TRUNCATED]

> barmose$Type <- factor(Type)

> types_map <- tm_shape(barmose, unit="m") +
+   tm_fill("Type", palette="viridis") +
+   tm_scale_bar(breaks=c(0,1,2), position=c("right", "bottom" .... [TRUNCATED]

> jpeg("Barmose_types_Cores.jpeg", width=15, height=17, units="cm", res=300)

> types_map

> dev.off()
null device
         1

> # Plot Cores + LICD - TIFF + JPEG
>
> LICDClass <- interaction(barmose$class, barmose$Type, sep=" ")

> barmose$LICDClass <- factor(LICDClass, levels=c("Core Hot only", "Core Clump only (Hot)", "Core Hot Clump", "Core No cluster", "No Core Hot only

> LICDClass_map <- tm_shape(barmose, unit="m") +
+   tm_fill("LICDClass", palette="-viridis", title="Classes + LICD") +
+   tm_scale_bar(breaks=c(0, .... [TRUNCATED]

> jpeg("Barmose_LICD_class.jpeg", width=15, height=17, units="cm", res=300)

> LICDClass_map

> dev.off()
null device
         1

> sessionInfo()
R version 4.0.3 (2020-10-10)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 20.04 LTS

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/liblapack.so.3

locale:
[1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=de_DE.UTF-8        LC_COLLATE=en_US.UTF-8
[5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=en_US.UTF-8    LC_PAPER=de_DE.UTF-8       LC_NAME=C
[9] LC_ADDRESS=C               LC_TELEPHONE=C             LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] stars_0.4-3         abind_1.4-5         spatstat_1.64-1     rpart_4.1-15        nlme_3.1-150        spatstat.data_1.5-2
[7] archdata_1.2        mapview_2.9.4       Matrix_1.2-18       units_0.6-7         tmap_3.2            spdep_1.1-5
[13] spData_0.3.8        sp_1.4-4            sf_0.9-6

loaded via a namespace (and not attached):
 [1] colorspace_2.0-0        leafem_0.1.3            deldir_0.2-3            ellipsis_0.3.1          class_7.3-17
 [6] leaflet_2.0.3           rprojroot_2.0.2         dadjoke_0.1.2           cranlike_1.0.2          satellite_1.0.2
[11] base64enc_0.1-3         fs_1.5.0                dichromat_2.0-0         rstudioapi_0.13         farver_2.0.3
[16] remotes_2.2.0           bit64_4.0.5             fansi_0.4.1             codetools_0.2-18        splines_4.0.3
[21] knitr_1.30              polyclip_1.10-0         pkgload_1.1.0           jsonlite_1.7.1          tmaptools_3.1
[26] png_0.1-7               compiler_4.0.3          httr_1.4.2              assertthat_0.2.1        cli_2.2.0
[31] leaflet.providers_1.9.0 htmltools_0.5.0         prettyunits_1.1.1       tools_4.0.3             coda_0.19-4
[36] glue_1.4.2              dplyr_1.0.2             rappdirs_0.3.1          gmodels_2.18.1          Rcpp_1.0.5
[41] raster_3.4-5            vctrs_0.3.5             svglite_1.2.3.2         gdata_2.18.0            debugme_1.1.0
[46] leafsync_0.1.0          crosstalk_1.1.0.1       lwgeom_0.2-5            xfun_0.19               stringr_1.4.0
[51] ps_1.4.0                testthat_3.0.0          parsedate_1.2.0         lifecycle_0.2.0         gtools_3.8.2
[56] devtools_2.3.2          goftest_1.2-2           XML_3.99-0.5            LearnBayes_2.15.1       MASS_7.3-53
[61] scales_1.1.1            spatstat.utils_1.17-0   clisymbols_1.2.0        parallel_4.0.3          expm_0.999-5
```

```
[66] rematch2_2.1.2          RColorBrewer_1.1-2      yaml_2.2.1              curl_4.3                memoise_1.1.0
[71] gdtools_0.2.2           stringi_1.5.3           RSQLite_2.2.1           desc_1.2.0              leafpop_0.0.6
[76] e1071_1.7-4             crancache_0.0.0.9001    boot_1.3-25             pkgbuild_1.1.0          repr_1.1.0
[81] systemfonts_0.3.2       rlang_0.4.8             pkgconfig_2.0.3         prompt_1.0.0            evaluate_0.14
[86] lattice_0.20-41         tensor_1.5              purrr_0.3.4             htmlwidgets_1.5.2       bit_4.0.4
[91] processx_3.4.4          tidyselect_1.1.0        magrittr_2.0.1          R6_2.5.0                generics_0.1.0
[96] DBI_1.1.0               mgcv_1.8-33             pillar_1.4.7            withr_2.3.0             tibble_3.0.4
[101] crayon_1.3.4            uuid_0.1-4              KernSmooth_2.23-18      rmarkdown_2.5           usethis_1.6.3
[106] grid_4.0.3              blob_1.2.1              callr_3.5.1             git2r_0.27.1            webshot_0.5.2
[111] digest_0.6.27           classInt_0.4-3          brew_1.0-6              stats4_4.0.3            munsell_0.5.0
[116] viridisLite_0.3.0       skimr_2.1.2             sessioninfo_1.1.1

> sf_extSoftVersion()
         GEOS           GDAL         proj.4 GDAL_with_GEOS     USE_PROJ_H
      "3.8.1"        "3.1.3"        "7.1.1"         "true"         "true"

> sink(type = "message")

> sink()
```

Citing this document

Daniel Nüst (2020). CODECHECK Certificate 2020-025. Zenodo. https://doi.org/10.5281/zenodo.4279275

About CODECHECK

This certificate confirms that the codechecker could independently reproduce the results of a computational analysis given the data and code from a third party. A CODECHECK does not check whether the original computation analysis is correct. However, as all materials required for the reproduction are freely available by following the links in this document, the reader can then study for themselves the code and data.

About this document

This document was created using R Markdown using the `codecheck` R package. `make codecheck.pdf` will regenerate the report file.

```r
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04 LTS
## 
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/liblapack.so.3
## 
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C              
##  [3] LC_TIME=de_DE.UTF-8        LC_COLLATE=en_US.UTF-8    
##  [5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=en_US.UTF-8   
##  [7] LC_PAPER=de_DE.UTF-8       LC_NAME=C                 
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C            
## [11] LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C       
## 
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets 
## [6] methods   base     
## 
## other attached packages:
##  [1] readr_1.4.0          tibble_3.0.4        
##  [3] xtable_1.8-4         yaml_2.2.1          
##  [5] rprojroot_2.0.2      knitr_1.30          
##  [7] codecheck_0.0.0.9010 parsedate_1.2.0     
##  [9] R.cache_0.14.0       gh_1.2.0            
## 
## loaded via a namespace (and not attached):
##  [1] git2r_0.27.1      compiler_4.0.3    pillar_1.4.7     
##  [4] base64enc_0.1-3   R.methodsS3_1.8.1 R.utils_2.10.1   
##  [7] tools_4.0.3       rorcid_0.6.4      digest_0.6.27    
## [10] jsonlite_1.7.2    evaluate_0.14     memoise_1.1.0    
## [13] lifecycle_0.2.0   pkgconfig_2.0.3   rlang_0.4.9      
## [16] cli_2.2.0         crul_1.0.0        curl_4.3         
## [19] xfun_0.19         withr_2.3.0       repr_1.1.0       
## [22] dplyr_1.0.2       httr_1.4.2        stringr_1.4.0    
## [25] hms_0.5.3         fauxpas_0.5.0     generics_0.1.0
## [28] fs_1.5.0          vctrs_0.3.5       tidyselect_1.1.0
## [31] glue_1.4.2        httpcode_0.3.0    R6_2.5.0
## [34] fansi_0.4.1       rmarkdown_2.6     tidyr_1.1.2
## [37] skimr_2.1.2       whisker_0.4       purrr_0.3.4
## [40] magrittr_2.0.1    osfr_0.2.8        htmltools_0.5.0
## [43] ellipsis_0.3.1    assertthat_0.2.1  utf8_1.1.4
## [46] stringi_1.5.3     crayon_1.3.4      R.oo_1.24.0
```