Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. doi: 10.18637/jss.v000.i00

Modelling Prehistoric Iconographic Compositions. The R package decorr

Thomas Huet
UMR 5140

Jose Pozo

Craig Alexander

Abstract

By definition, Prehistorical societies are characterised by the absence of a writing system. During, the largest part of human history – from far – symbolic expressions belong to illiterate societies which express themself with rock-art paintings, pottery decorations, figurines and statuary, etc., and a lot of now disappeared carved woods, textil design, etc. At the composition level, recognition of meaningful associations of signs and reccurent patterns indicate clearly the existence of social conventions in the way to display and to read these expressions. We present the **decorr** R package which grounds concepts, methods and tools to analyse any ancient graphical systems. Our asumption is that i) any graphical system is a spatial distribution of features, and ii) these features have possibly any meaningful relationships one with another depending on their pairwise spatial proximities. To model the graphical compositions we employ concepts coming from the Graph Theory. To ensure the feasability of this type of analysis, we propose a GIS-based method for inputs and a serie of functions for data management.

Keywords: Semiotics, Iconography, Prehistory, Graphs, Spatial Analysis, Binary Topological Relationships (*birel*), R.

1. Introduction

Symbolic practices is a characteristic trait of human societies. Even discussed, such practices seems to start between 233,000 to 800,00 BC (d'Errico and Nowell 2000), covering more than 97% of total human societies time span. Symbolism cover a large range of practices, from ochre deposit in a tomb, to menhir alignements, passing through wall fresco. This latter, what might be called "iconographical practices", probably shows the most complex and interesting testimonies of past societies symbolism. For decades, its study was linked to history of religion because commonly seen as closely linked to cultual practices and believes. Since the *New Archaeology* developpement during the 60's (Clarke 2014), symbolic expressions start to be

studied with the same formal methods (statistics, seriations, distribution maps, etc.) as any another aspect of social organisation: settlement patterns, tools *chaine opératoire*, susbsitence strategies, etc. (Renfrew and Bahn 1991). But unlike many aspects of the material culture where technological requirements and technical efficiency determine the choice of the raw material and the shape of the object – a flint blade for cutting, a pottery for containing, a house for living –, the function of an iconographic composition cannot be drawn directly from itself. Furthermore, there is no specific and controled vocabularies to describe hand-made images. There is always a difficulty to decribe an iconographical content with alphanumerical signs. Whether these last decades study of ancient iconography had undergone significative improvements at the site scale – with GIS/database statistics, paleoenvironmental restitutions, etc. – and at the sign scale with the development of archaeological sciences – radiocarbon dating, use-wear analysis, elemental analysis, etc. –, these improvement do not necessarily help to understand the semantic content of the iconography.
Semantics or semiotics can be defined as a system of conventional features – called signs – organised also in conventional manners. Until our days, formal methods to study ancient iconography semantics have been mostlty been grounded – explicitly or not – on the prime principle of Saussurian linguistic: the '*linearity of the signifier*' (De Saussure 1989). Writing is one of the most rational semiographical system with a clear distinction between signified and signifier and the development of the signified on a linear axis. Even if we do not understand the meaning of the signs, writing can easily be modelled with Graph theory and recurent patterns can be identified. For example, the 3-letters word `"art"` can be modelled with three vertices (`a`, `r`, `t`) and two edges (one between `a` and `r`, the other between `r` and `t`). In R, these features, concatenated in this order with a `paste0()`, is `art`, and not `rat` (Figure 1).

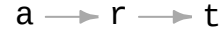

Figure 1: Concatenate of 'a', 'r' and 't' graphical units is 'art'

But, as stated, in Prehistorical the writing system does not exists. Spatial relationships between graphical features, or graphical units, are not necessarily linear and directed but could most probably be more multi-directional and undirected: the direction of the interactions of pairwise graphical units can be in any order (Figure 2).

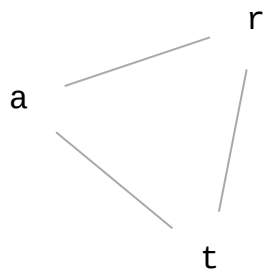

Figure 2: Potential spatial relations between graphical units

Because of the inherent variability of iconography, and because graphical and spatial proximities between graphical units are generally not quantified, applying the Saussurian model to any structured prehistorical graphical content had led to considerable problems:

- unexplicit spatial groupings of graphical units – like graphical units grouped into *figures*, *figures* grouped into *patterns*, *patterns* grouped into *motives*, etc. – with tedious number of groups
- consistency, proximities and relationships between these groups are often implicit and not quantified
- studies develop proper descriptive vocabularies, singular relationships of grouping, idosyncratic methods at site-dependent or period-dependent scales

These issues limit drastically the possibility to conduct cross-cultural comparisons and to draw a synthesis of humankind's symbolism at a large scale and over the long-term.

According to us, prehistorical composition – like any formal system – are spatial features (mostly shapes) related one with the other depending on rules of spatial proximities. Then, the First Law of Geography: "*everything is related to everything else, but near things are more related than distant things*" (Tobler 1970) should be more operative than the Saussurian '*linearity of the signifier*' principle (De Saussure 1989). To get these '*near things*', or features proximities, we use concepts coming from the Graph Theory, which offers a general framework to manage neighbourhoods. Furthermore, we address the questions of managing graphical units attributes (for example an anthropomorph with a sword *vs* an anthropomorph with a spear) and superimpositions of graphical units.

In this article we present the R package **decorr**. Its purpose is to formalise a method and offer tools to analyse any graphical content (here: decorations) with methods coming from raph Theory, GIS and binary topological relationships (*birel*) modeling. This package has been grounded on the seminal work of Alexander (2008) and its first IT implementation by Huet (2018).

2. Graph theory Model

Graph theory offers a conceptual framework and indices (global at the entire graph scale, local at the vertex scale) to deal with notions of networks, relationships and neighbourhoods. Graphical units (GUs) can be modelled as vertices (nodes) separated one with another by undecorated areas. Their spatial relations can be modelled as edges. The different levels of GUs spatial organisation can be retrieve by graphs analysis (Graph Theory) and a spatial (GIS) analysis.

Graph elements, nodes and edges – repectively GUs and connexions between GUs – are created on a GIS interface. Indeed, for large series of decorated supports, GIS offers the most suitable and flexible interface to register all GUs and to get their coordinates, makes easy the updating of GUs and GUs relation and opens to develop further spatial analysis. The decoration image is considerated as the basemap of the project and will cover the region of interest of the analysis (ie, all the graphical content). The decoration image can be binarized: GUs are considerated active, the undecorated parts of the support – the background – are considerated inactive. The polygonisation of the GUs (`POLYGONS`), including the border of the stelae, allows to calculated their Voronoi seeds (ie, their centroid) and Voronoi cells (ie, Thiessen polygons, their area of influence). So, the entire decoration image is tiled and condidered as a Voronoi diagram. Centroids of the GUs (`POINTS`) are calculated and considered as the graph nodes,

their locations (x and y coordinates, measured in pixels) are relative to each decoration. Exist edges (`LINES`) between pairs of GUs when their Voronoi cells share a common border (*birel*: 'touches').

The entire graphical content of the decoration is mapped with a geometric graph – or planar graph – of GUs centroids (nodes) and GUs proximity links (edges). Thus, a decoration graph is a one component graph. This graph is the dual graph of the decoration Voronoi diagram (Figure 3).

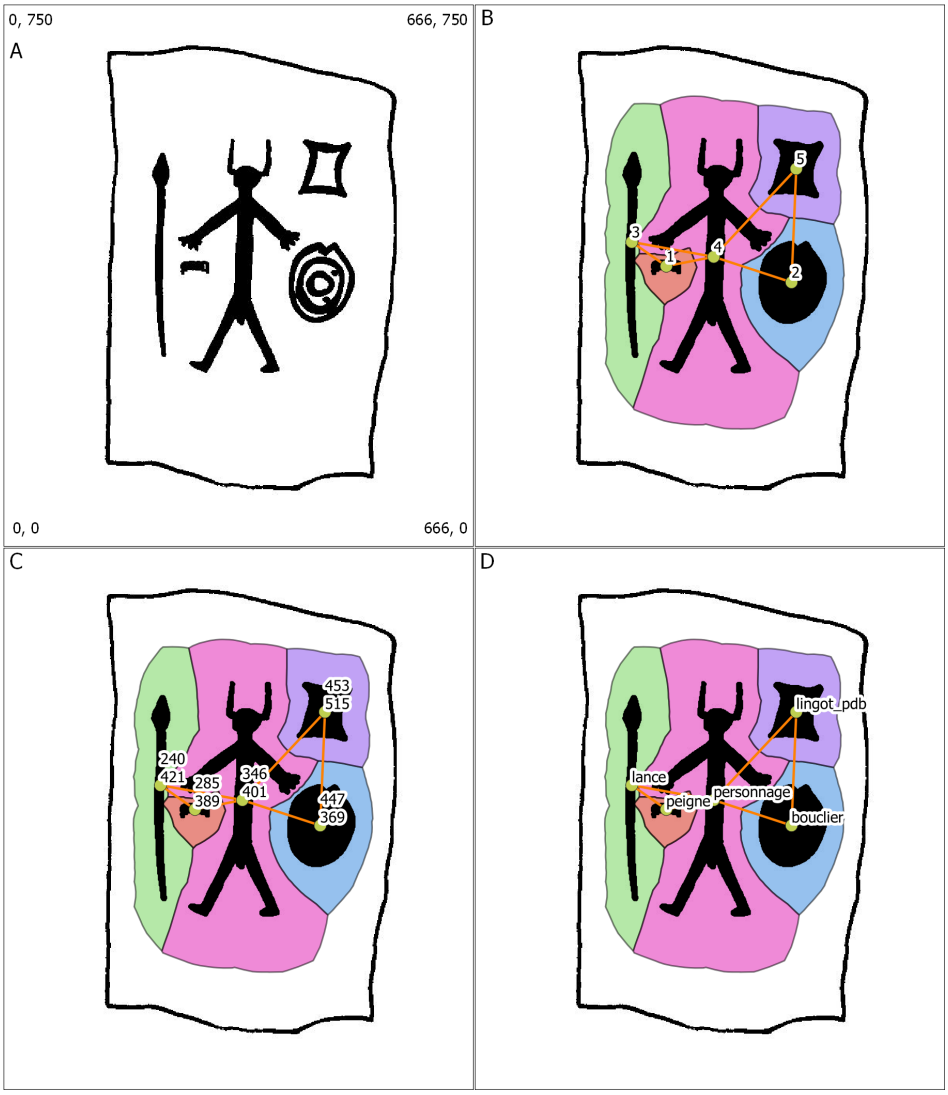

Figure 3: GIS interface. A) Original decoration of the Late Bronze Age Cerro Muriano 1 stelae (drawing: Díaz-Guardamino Uribe (2010)) opened in new GIS project without any projection system with its extent (`xmin, xmax, ymin, ymax` in pixels); B) Polygonisation of the GUs, calculation of their centroids (nodes), calculation of the Voronoi cells, calculation of the proximity links (edges); C) plot of the relative x and y coordinates of the GUs centroids; D) plot of the values in the `type` of the GUs

Whether this theoretical process includes image binarization, polygonization and *birel* analy-

sis, a simplier solution will be to create directly centroids on the GUs and to draw the edges manually. Then x and y coordinates of nodes can be easily retrieved.

The notation of with nodes allows also to avoid, empirical and top-down hierachical typologies . According to us, it is more convenient to record GUs (eg, an anthropomorph) and GUs varietes (an anthropomorph wearing an helmet) with different spatialized nodes (firstly within a GIS) rather than multiplying hierarchical levels of categories (e.g., a category of graphical units with n types, a type with n subtypes, a subtype with n varieties, etc.) that lead also to tedious idiosynchratic inventories and difficult possible comparisons *within* and *between* these categories. For example, with the two types of GUs `casque` (helmet) and `personnage` (anthropomorphe) and the *attribute* edge it is possible to describe: (1) a case where an anthropomorph wears a helmet (like for the Cerro Muriano steale, Figure 4 left), (2) a case where an anthropomorph is separated from a helmet (like for the Zarza de Montanchez, Figure 4 right). With a top-down hierachical typology, three types of GUs would have been required: (1) anthropomorph wearing a helmet (Cerro Muriano); (2) anthropomorph and (3) helmet (Zarza de Montanchez).

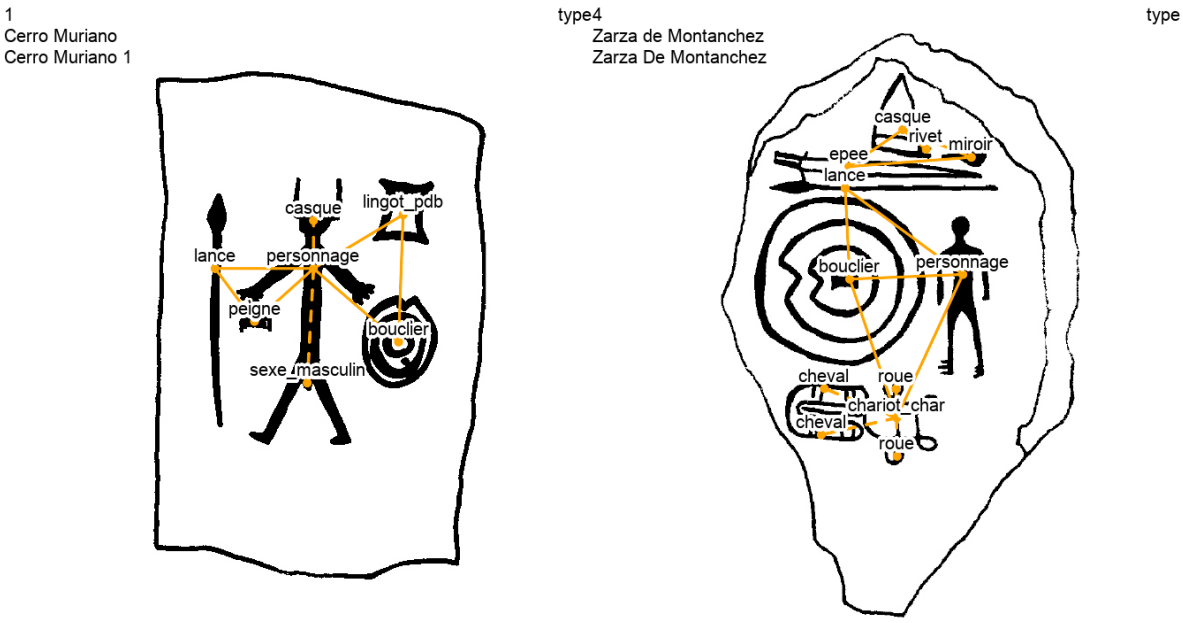

Figure 4: Cerro Muriano 1 and Zarza de Montanchez decorations.

2.1. Node types

Nodes are uniquely identified by a number. But this does not identify or classify the nodes by the corresponding GU meaning or type. For this, name nodes could be assigned. However this presents two problems: nodes with repeated GU type in the same graph could not be handled and comparisons by different GU classifications would be hindered. For this reason, the R package decorr considers that the GU type is assigned in a node attribute or variable. Thus, at least one node variable must be defined in order to analyse and compare iconographical

compositions. By default this node variable is named `type`. In this article examples, we consider node `type` values ('anthropomorph', 'sword', etc.), but the package user can add other variables (`technique`, `color`, etc.), which would produce complementary analyses.

2.2. Egdes types

The graph edges can represent different types of relationships between GUs. This relationship type is encoded in an edge variable named `type`. The current package version considers the most common types:

Normal edges. Two contemporaneous nodes having a Voronoi cell sharing a border have a common edge. By convention, this edge is tagged `'='` (`type = '='`), represented with an orange plain line and its textual notation is `'-=-'`. For example: `personnage -=- bouclier` means that the nodes `personnage` and `bouclier` have a common border (Figure 5). As said, in Prehistorical iconography, the read direction is not known. Thus this edge type is undirected, ie `personnage -=- bouclier` is equal to `bouclier -=- personnage`.

Attribute edges. Frequently a node can be divided into a main unit (eg, an anthropomorph) and one or various attribute units (eg, a helmet, a male sex). To record this ordered set, the edge between a main and an attribute node is tagged `'+'` (`type = '+'`), represented with an orange dashed line and its textual notation is `'-+-'`. For example `personnage -+- casque` means that the main node `personnage` has the attribute node `casque`. The edge is directed since `personnage -+- casque` (the anthropomorph has an helmet) is not equal to `casque -+- personnage` (the helmet has an anthropomorph) (Figure 5).

Overlap edges. Sometimes a graphical composition shows decoration phases. These decoration phases can be identified by observing the superimpositions between different nodes or by other means (differences in the techniques or in the patina for example). In the first case, two nodes belonging to two different phase will have a common part (the overlapped intersection) while in the second case the two nodes will not necessarily share a part of their surface. To record this relative chronology, an edge between two non-contemporaneous nodes is tagged `'>'` (`type = '>'`), represented with a blue plain line and its textual notation is `'->-'`. For example `ecriture ->- bouclier` means that `ecriture` is more recent than the `bouclier`, or *overlaps* the`bouclier` in the stratigraphical sense. The edge is directed since `ecriture ->- bouclier` (ie, `ecriture` *overlaps* `bouclier`) is not equal to `bouclier ->- ecriture` (ie, `bouclier` *overlaps* `ecriture`) (Figure 6).

This stratigraphical information (node 1 *over* node 2, or node 2 *under* node 1) helps to understand the relative chronology between nodes and must be recorded. The analysis of the iconographical content can be stratified (stratified analysis) and performed on each different phases of decoration separately.

These three relationship types are resumed with their *birel* equivalencies and chronological assumptions in Table 1. The equality conditions between nodes and between edges are resumed in Table 2, taking into account the directness of each edge type.

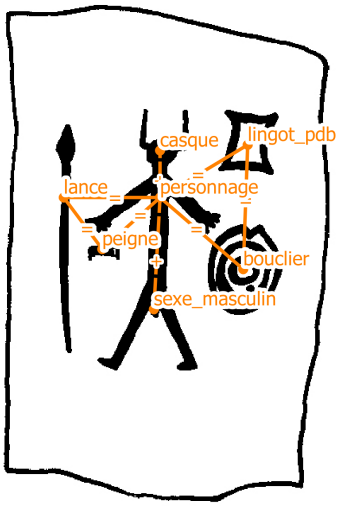

Figure 5: GIS interface, Cerro Muriano 1 decoration. The nodes `casque` (helmet) and `sexe_masculin` (male sex) are two nodes attributes of the node `personnage` (anthropomorph).

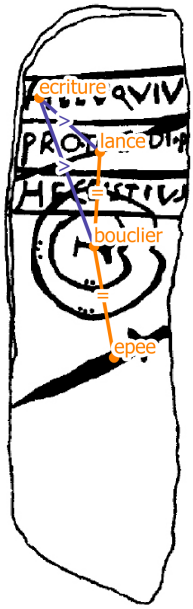

Figure 6: GIS interface, Ibahernando decoration. The node `ecriture` (writing) overlaps `lance` (spear) and the upper part of the `bouclier` (shield).

Node 1	Edge type	Node 2	Edge directness	*Birel*	Description
`A`	`=`	`B`	Undirected	$A \cap B = \emptyset$	`A` and `B` contemporaneous and disjoint.
`A`	`+`	`B`	Directed	$A \cap B = A$	`A` and `B` contemporaneous. `B` is an attribute of `A`.
`A`	`>`	`B`	Directed	Any $A \cap B$	`A` more recent than `B`. Overlapping or disjoint.

Table 1: Synthesis for the different types of relations between GUs.

Element class	Element 1	Element 2	is.equal?
node	epee	epee	True
node	epee	lance	False
edge	epee –=– lance	epee –=– lance	True
edge	epee –=– lance	lance –=– epee	True
edge	personnage –+– bouclier	personnage –+– bouclier	True
edge	personnage –+– bouclier	bouclier –+– personnage	False
edge	ecriture –>– bouclier	ecriture –>– bouclier	True
edge	ecriture –>– bouclier	bouclier –>– ecriture	False
edge	epee –=– lance	epee –=– bouclier	False
edge	personnage –=– bouclier	personnage –+– bouclier	False

Table 2: Example of equalities and differences for nodes and edges

3. The R package decorr

The **decorr** package can be downloaded from GitHub

```
R>   devtools::install_github("zoometh/iconr", build_vignettes=TRUE)
```

3.1. External package dependencies

The **decorr** package imports the following packages:

- **magick** for image manipulation (Ooms 2018)
- **igraph** for graph and network analysis (Csardi and Nepusz 2006)
- **rgdal** to read *shapefiles* of nodes and edges (Bivand, Keitt, and Rowlingson 2019)
- **grDevices** for colors and font plotting, **graphics** for graphics, **utils** and **methods** for formally defined methods and *varia* methods (all combinations, etc.) (R Core Team 2019)

3.2. Data

A small testing dataset is provided with the **decorr** package, stored in its `extdata` folder. It contains dataframes of nodes' coordinates, edges' coordinates, decorations' listing in different

formats (`.tsv`, `.csv` or *shapefiles*), and the decoration drawings (`.jpg`) of five Late Bronze Age stelae belonging to the so-called 'Warrior stelae' family – with about 140 stelae – mostly concentrated in the Southwest Iberian peninsula (Pérez 2001).

Imgs dataframe

The inventory of decorations is stored in the comma-separated values file `imgs.csv` and the tab-separated values files `imgs.tsv`. Any of them can be read as a dataframe (Table 3),

```
R> imgs <- read.table(system.file("extdata", "imgs.tsv", package = "decorr"),
+                     sep="\t", stringsAsFactors = FALSE)
```

The field `imgs$idf` is the short name of the decoration, useful during statistical analysis. The primary key of each decoration is the concatenate of `imgs$site` and `imgs$decor`. These keys will allow joints with the other dataframes (nodes and edges).

idf	site	decor	img
1	Cerro Muriano	Cerro Muriano 1	Cerro_Muriano.Cerro_Muriano_1.jpg
2	Torrejon Rubio	Torrejon Rubio 1	Torrejon_Rubio.Torrejon_Rubio_1.jpg
3	Brozas	Brozas	Brozas.Brozas.jpg
4	Zarza de Montanchez	Zarza De Montanchez	Zarza_de_Montanchez.Zarza_De_Montanchez.jpg
5	Ibahernando	Ibahernando	Ibahernando.Ibahernando.jpg

Table 3: The studied corpus, the `imgs.tsv` dataframe

At first the drawings dataset (Díaz-Guardamino Uribe 2010) can be checked by using the information in the `imgs` dataframe and the **magick** package (Figure 7)

```
R> pth <- system.file("extdata", package = "decorr")
R> lims <- list()
R> for(i in 1:nrow(imgs)){
+    i1 <- image_read(paste0(pth,"/",imgs$img[i]))
+    lbl.txt <- paste0(paste(imgs[i,], collapse = "\n"), "\n",
+                      image_info(i1)$width, "*", image_info(i1)$height, " px")
+    lims[[i]] <- image_annotate(i1, lbl.txt, location = "northwest",
+                                size = 28, color = "red")
+  }
R> out.img <- image_append(c(image_append(c(lims[[1]], lims[[2]], lims[[3]])),
+                            image_append(c(lims[[4]], lims[[5]]))),
+                          stack = TRUE)
R> par(mar=c(0, 0, 0, 0))
R> plot(out.img)
```

To construct a graph overlapping the decoration images listed in the `imgs` dataframe, the first step is to load nodes and edges dataframes.

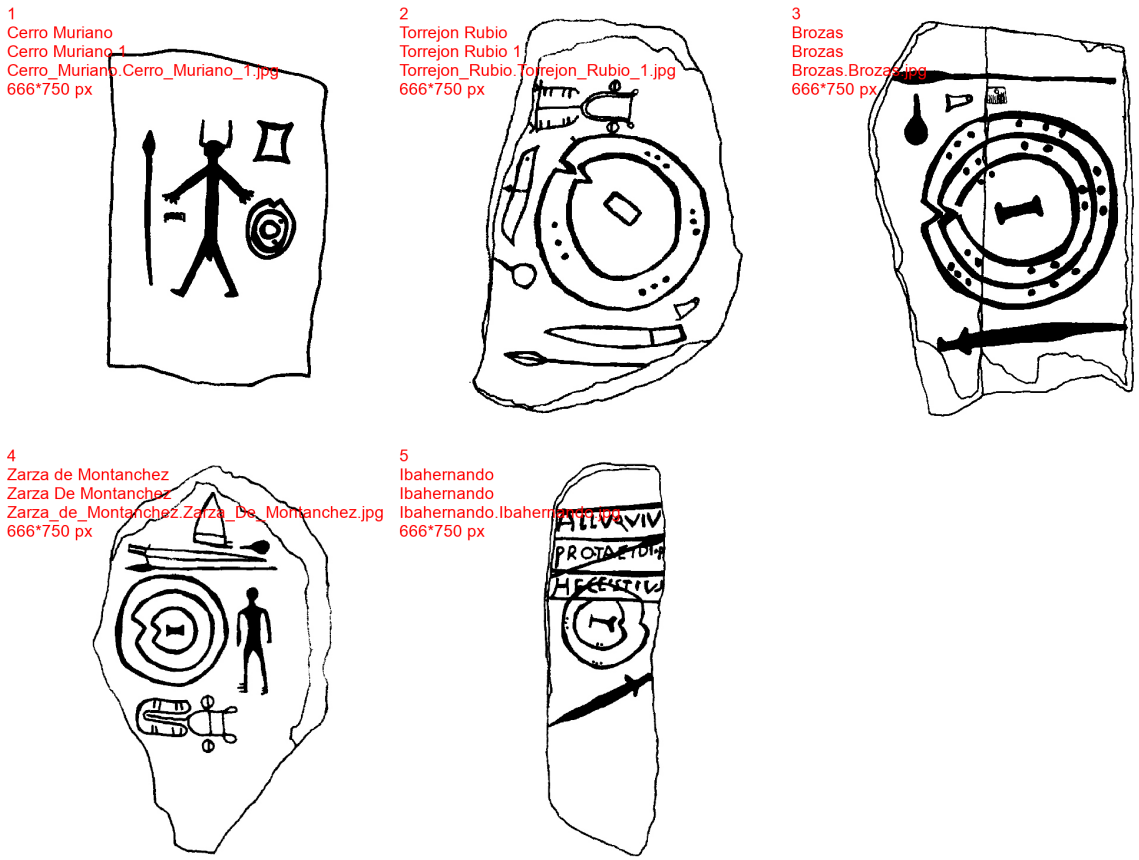

Figure 7: Decoration images of the training dataset

Nodes dataframe

The nodes for the five decorations are stored in different formats in `nodes.csv`, `nodes.tsv`, and `nodes.shp`. This can be read by

```
R> nodes <- read.table(system.file("extdata", "nodes.csv", package = "decorr"),
+                       sep=";", stringsAsFactors = FALSE)
```

generating a dataframe containing the required minimum variables for the analysis (Table 4).

Since a prehistorical site can have various decorated objects, the primary key of the decoration is based on two fields: `nodes$site` and `nodes$decor`. The `nodes$id` is the identifier of the node. The `nodes$type` field is the default variable for further statistical analysis. Here, `nodes$type` refers to the typology of the nodes (anthropomorph, weapons, etc.). The `nodes$x` and `nodes$y` columns refer to the x and y coordinates of the nodes. As said, in the first place theses coordinates come from the GIS where the origin (0, 0) is the bottom-left corner, while for any R matrices (rasters, grids, dataframes, etc.) this origin is top-left (see Figure 3. A). To recover the local y value of nodes and edges on the decoration image, the absolute value y value and the image height, as a constant offset, are computed.

site	decor	id	type	x	y
Cerro Muriano	Cerro Muriano 1	1	personnage	349.81	-298.32
Cerro Muriano	Cerro Muriano 1	2	casque	349.81	-243.99
Cerro Muriano	Cerro Muriano 1	3	lance	238.46	-298.32
Cerro Muriano	Cerro Muriano 1	4	bouclier	446.02	-381.17
Cerro Muriano	Cerro Muriano 1	5	peigne	283.00	-358.01
Cerro Muriano	Cerro Muriano 1	7	sexe_masculin	342.69	-427.49
Cerro Muriano	Cerro Muriano 1	8	lingot_pdb	451.15	-237.48
Torrejon Rubio	Torrejon Rubio 1	1	chariot_char	306.30	-177.12
Torrejon Rubio	Torrejon Rubio 1	2	bouclier	374.58	-347.81
Torrejon Rubio	Torrejon Rubio 1	3	arc	221.62	-303.82
Torrejon Rubio	Torrejon Rubio 1	4	miroir	211.77	-458.10
Torrejon Rubio	Torrejon Rubio 1	5	fibule	490.12	-513.24
Torrejon Rubio	Torrejon Rubio 1	6	epee	366.70	-563.14
Torrejon Rubio	Torrejon Rubio 1	7	lance	298.43	-607.78
Torrejon Rubio	Torrejon Rubio 1	8	cheval	251.16	-141.67
Torrejon Rubio	Torrejon Rubio 1	9	cheval	256.41	-209.95
Torrejon Rubio	Torrejon Rubio 1	10	roue	364.51	-138.76
Torrejon Rubio	Torrejon Rubio 1	11	roue	363.64	-209.28
Torrejon Rubio	Torrejon Rubio 1	12	fleche	185.59	-312.46
Brozas	Brozas	1	lance	354.11	-123.36
Brozas	Brozas	2	peigne	346.35	-151.84
Brozas	Brozas	3	fibule	279.04	-162.19
Brozas	Brozas	4	miroir	211.74	-206.20
Brozas	Brozas	5	bouclier	392.94	-343.40
Brozas	Brozas	6	epee	387.76	-564.72
Zarza de Montanchez	Zarza De Montanchez	1	casque	350.73	-141.33
Zarza de Montanchez	Zarza De Montanchez	2	miroir	428.21	-172.67
Zarza de Montanchez	Zarza De Montanchez	3	epee	289.79	-183.12
Zarza de Montanchez	Zarza De Montanchez	4	lance	285.43	-207.49
Zarza de Montanchez	Zarza De Montanchez	5	bouclier	290.66	-310.23
Zarza de Montanchez	Zarza De Montanchez	6	personnage	418.64	-305.00
Zarza de Montanchez	Zarza De Montanchez	7	chariot_char	343.76	-466.94
Zarza de Montanchez	Zarza De Montanchez	8	cheval	261.93	-433.85
Zarza de Montanchez	Zarza De Montanchez	9	cheval	258.44	-486.09
Zarza de Montanchez	Zarza De Montanchez	10	roue	342.95	-434.16
Zarza de Montanchez	Zarza De Montanchez	11	roue	344.21	-510.23
Zarza de Montanchez	Zarza De Montanchez	12	rivet	377.14	-163.52
Ibahernando	Ibahernando	1	ecriture	284.11	-131.23
Ibahernando	Ibahernando	2	lance	351.60	-191.56
Ibahernando	Ibahernando	3	bouclier	345.04	-296.60
Ibahernando	Ibahernando	4	epee	367.36	-420.68

Table 4: Nodes (from `nodes.csv` dataframe)

Edges dataframe

Analogously, the edges for the five decorations are stored in different formats in `edges.csv`, `edges.tsv`, and `edges.shp`. This can be read by

```
R> edges <- read.table(system.file("extdata", "edges.csv",
+                                  package = "decorr"),
+                      sep=";",stringsAsFactors = FALSE)
```

generating the corresponding edges dataframe (Table 5).

site	decor	a	b	type
Cerro Muriano	Cerro Muriano 1	1	4	=
Cerro Muriano	Cerro Muriano 1	1	5	=
Cerro Muriano	Cerro Muriano 1	3	5	=
Cerro Muriano	Cerro Muriano 1	1	2	+
Cerro Muriano	Cerro Muriano 1	1	7	+
Cerro Muriano	Cerro Muriano 1	3	1	=
Cerro Muriano	Cerro Muriano 1	1	8	=
Cerro Muriano	Cerro Muriano 1	4	8	=

Table 5: Edges of Cerro Muriano 1 (from `edges.csv` dataframe)

The fields `edges$site` and `edges$decor` are the primary key of decoration. The fields `edges$a` and `edges$b` are the equivalent to columns *from* and *to* in Graph theory. Even if undirected graphs will the most common in further studies, this direction helps to distinguish between nodes. The column `edges$a` is the identifier of *starting node* or *main node* or *overlapping node*. The column `edges$b` is the identifier of the *ending node* or *attribue node* or *overlapped node*. The column `edges$type` is the type of relation (normal, attribute, overlap, etc.) between the node a and the node b. There is no need to record the coordinates of the edges, these coordinates are extracted from the nodes dataframe.

The **decorr** package functions can be divided into:

1. management functions
2. graphical functions
3. single decoration functions
4. comparisons functions

Once these dataframes loaded, the list of decoration graphs can be calculated with `list_dec()`.

3.3. Management functions

The management functions are `list_dec()`, `contemp_nds()` and `named_elements()`

```
list_dec(imgs,
         nodes,
         edges)
```

Once imgs, nodes and edges dataframes loaded, the `list_dec()` function allows to create graphs for all decorations within these dataframes. The result is a list of `igraph` objects. As an example, the first graph of the list can be plotted with the `igraph::plot()` function. By default, the graphs spazialisation (`layout`) is based on `x` and `y` columns, when exist. This is our case since we work with geometric graphs. For a same graph, very different layouts (*rÃľalisations*) exist (Mathis 2003)(graph drawing) (Figure 8)

```
R> lgrph <- list_dec(imgs, nodes, edges)
R> g <- lgrph[[1]]
R> par(mar = c(0, 0, 0, 0), mfrow = c(1, 2))
R> coords <- layout.fruchterman.reingold(g)
R> plot(g,
+       vertex.size = 14,
+       vertex.label.family = "Helvetica",
+       edge.color = "orange",
+       vertex.label.cex = .8
+  )
R> plot(g,
+       layout = coords,
+       vertex.size = 4 + (degree(g)*10),
+       vertex.label.family = "Helvetica",
+       edge.color = "orange",
+       vertex.label.cex = .8
+  )
```

It occurs that some nodes are non-contemporaneous one with the others, like for the Ibahernando stelae. This stelae have been found reused as a funerary steale during Roman times (Basch 1966). A latin inscription (`ecriture`, node 1) have been carved over a spear (`lance`, node 2) and overlaps partially a V-notched shield (`bouclier`, node 3). The edges between node 1 and node 2, and the edge between node 1 and node 3, are *overlap* edges (Figure 9).

```
R> par(mar=c(0,0,0,0))
R> nds.df <- read_nds(site = "Ibahernando",
+                     decor = "Ibahernando",
+                     doss = system.file("extdata",
+                                        package = "decorr"))
R> eds.df <- read_eds(site = "Ibahernando",
+                     decor = "Ibahernando",
+                     doss = system.file("extdata",
+                                        package = "decorr"))
R> img.graph <- plot_dec_grph(nodes = nds.df,
+                              edges = eds.df,
+                              site = "Ibahernando",
+                              decor = "Ibahernando",
+                              lbl.size = 2,
+                              doss = system.file("extdata",
+                                                 package = "decorr"))
```

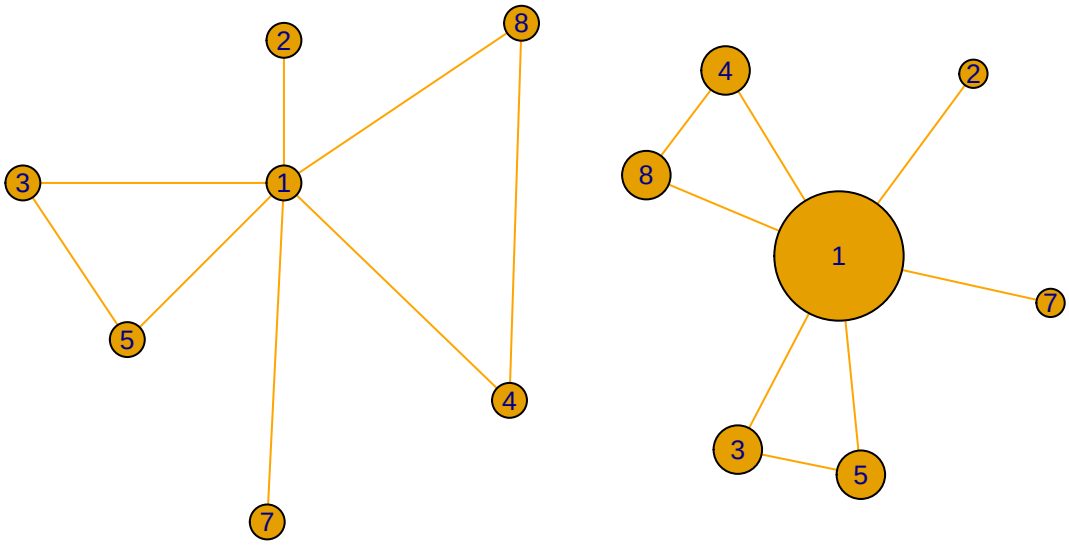

Figure 8: R interface. Plot of the first graph of the list (Cerro Muriano 1 decoration) with graph drawing on x, y coordinates (left), and force-directed graph drawing with nodes sized on their degree centralities (right)

```
R> plot(magick::image_read(img.graph))
```

In this case, the non-contemporaneous layers of decoration should be removed before the comparison process. To do so, the original graph (1-component) will be splitted into different sub-graphs (n-component) by selecting `>` edges. The studied graph component will be retrieved with the component membership of a selected node in the `contemp_nds()` function parameters.

To study only the Late Bronze Age iconographic layer of the Ibahernando steale, we can choose the Late Bronze Age node 4, the figure of sword (`epee`) dated to the middle and final stages of Late Bronze Age (ca 1250-950 BC). This node is supposed to be contemporaneous to node 2 and node 3 (Figure 10).

```
R> par(mar=c(0,0,0,0))
R> selected.nd <- 4
R> nds.df <- read_nds(site = "Ibahernando",
+                     decor = "Ibahernando",
+                     doss = system.file("extdata",
+                                        package = "decorr"))
R> eds.df <- read_eds(site = "Ibahernando",
```

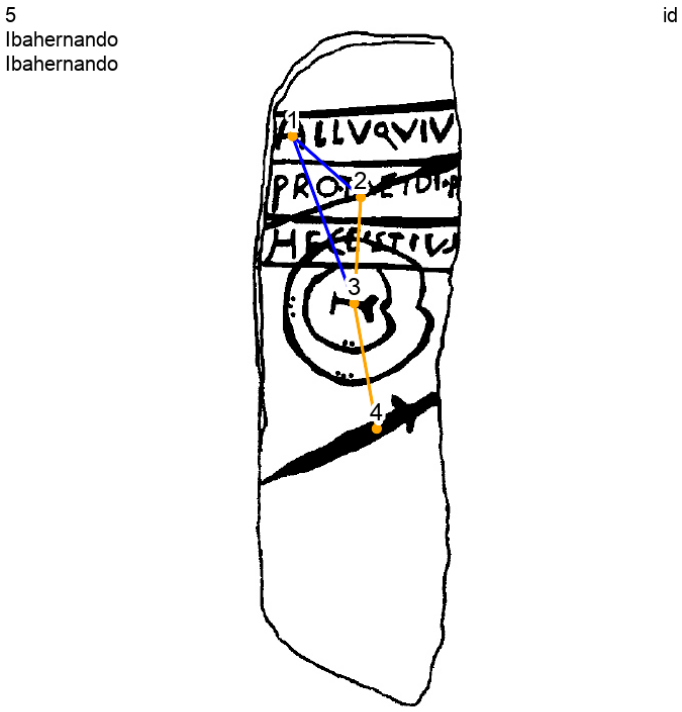

Figure 9: Ibahernando decoration, node 1 overlaps node 2 and node 3

```
+                     decor = "Ibahernando",
+                     doss = system.file("extdata",
+                                        package = "decorr"))
R> l_dec_df <- contemp_nds(nds.df, eds.df, selected.nd)
R> Ibahernando <- plot_dec_grph(nodes = nds.df,
+                                edges = eds.df,
+                                site = "Ibahernando",
+                                decor = "Ibahernando",
+                                nd.var = "type",
+                                lbl.color = "brown",
+                                lbl.size=2.2,
+                                doss = system.file("extdata",
+                                                   package = "decorr"))
R> Ibahernando.img <- magick::image_read(Ibahernando)
R> Ibahernando.contemp <- plot_dec_grph(nodes = l_dec_df[[1]],
+                                       edges = l_dec_df[[2]],
+                                       site = "Ibahernando",
+                                       decor = "Ibahernando",
+                                       nd.var = "type",
+                                       lbl.color = "brown",
+                                       lbl.size=2.2,
+                                       doss = system.file("extdata",
```

```
+                                                                    package = "decorr"))
R> Ibahernando.contemp.img <- magick::image_read(Ibahernando.contemp)
R> out <- magick::image_append(c(Ibahernando.img, Ibahernando.contemp.img))
R> plot(out)
```

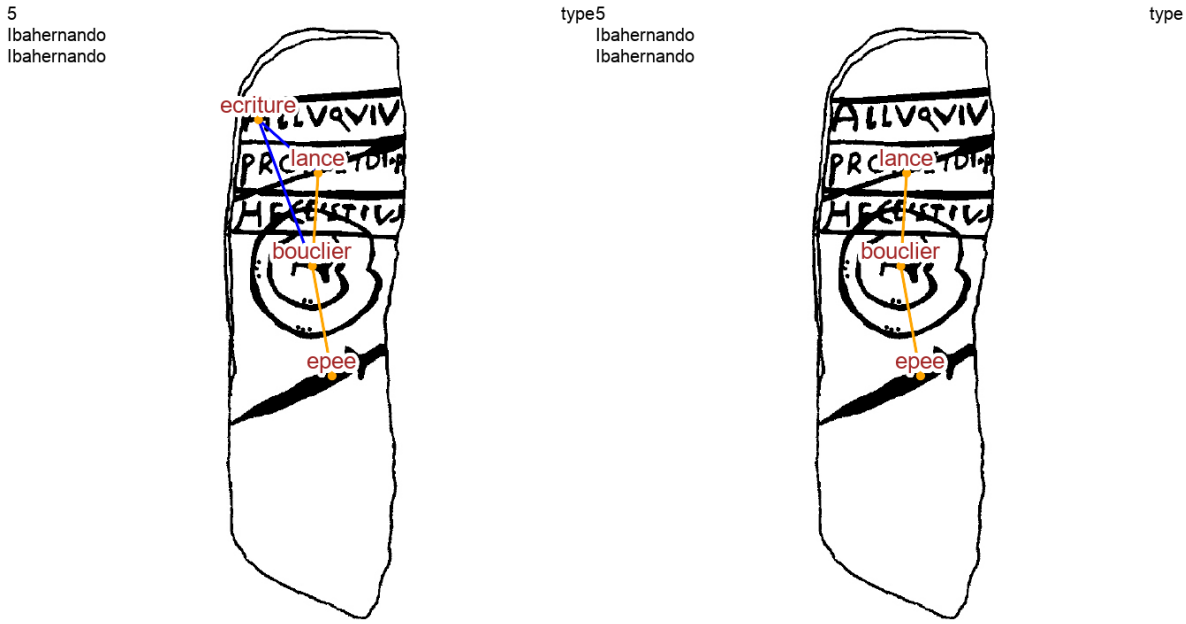

Figure 10: Ibahernando stelae before and after the selection of node 4 (sword) graph component

```
named_elements(grph,
               focus = "edges",
               nd.var = "type")
```

The `named_elements()` function allows to calculate the textual expression of graph elements (nodes and edges) with a disambiguation of these elements – adding '#' suffixes – when several nodes or edges have the same values.

```
R> lgrph <- list_dec(imgs, nodes, edges)
R> named_elements(lgrph[[2]], focus = "edges", nd.var="type")
## [1] "epee-=-fibule" "bouclier-=-miroir"
## [3] "epee-=-lance" "bouclier-=-chariot_char"
## [5] "chariot_char-+-cheval" "chariot_char-+-cheval#"
## [7] "arc-=-bouclier" "arc-=-chariot_char"
## [9] "arc-=-miroir" "bouclier-=-fibule"
## [11] "bouclier-=-epee" "epee-=-miroir"
## [13] "arc-+-fleche" "chariot_char-+-roue"
## [15] "chariot_char-+-roue#"
```

```
R> as.character(named_elements(lgrph[[2]], focus = "nodes", nd.var="type"))
## [1] "chariot_char" "bouclier" "arc" "miroir"
## [5] "fibule" "epee" "lance" "cheval"
## [9] "cheval#" "roue" "roue#" "fleche"
```

3.4. Graphical functions

The **decorr** package has two graphical functions: `labels_shadow()` and `side_plot()`.

```
labels_shadow(x, y = NULL, labels,
              col = "black", bg = "white",
              theta = seq(0, 2 * pi, length.out = 50),
              r = 0.1,
              cex = 1, ...)
```

`labels_shadow()` is a re-use of the `shadowtext()` function from the **TeachingDemos** package (Snow 2020). It plots labels (`labels`) at specific coordinates (`x, y`) with a contrasting buffer to make them more visible.

```
side_plot(grp, doss, nd.var, focus = "nodes",
          nd.color = c("orange", "red"),
          nd.size = c(0.5, 1),
          ed.color = c("orange", "red"),
          ed.width = c(1, 2),
          lbl.size = 0.5)
```

`side_plot()` allows to plot two decoration graphs side-by-side for elements (nodes or edges) comparisons. It take a list of two graph (`grp`), the path of the working directory (`doss`) and the name of the studied node variable (`nd.var`) and return an image.

3.5. Single decoration functions

Functions allowing to create a geometric graph for a single decoration are `read_nds()` and `read_eds()`. These functions allow to read respectively a file of nodes and a file of edges (`.csv`, `.tsv` or `.shp` files).

```
read_nds(site,
         decor,
         doss = getwd(),
         nodes = "nodes",
         dev = ".tsv")
```

`read_nds()` is close to the R native `read.table()` function, but allows to read *shapefiles* of nodes. For example, the nodes of the Torrejon Rubio 1 decoration can be read from the the *shapefile*, and the 6^{th} node – a sword (`epee`) – displayed (Table 6).

```
R> caption <- "Torrejon Rubio 1 decoration node 6: the sword."
R> sit <- "Torrejon Rubio" ; dec <- "Torrejon Rubio 1"
R> nds.df <- read_nds(site = sit, decor = dec, dev = ".shp",
+                     doss = system.file("extdata", package = "decorr"))
R> print(xtable::xtable(nds.df[nds.df$id == 6, ],
+                       caption = caption,
+                       label = "tab:nd",
+                       size = 7),
+        include.rownames = FALSE,
+        table.placement = "!htbp")
```

site	decor	id	type	x	y
Torrejon Rubio	Torrejon Rubio 1	6	epee	366.70	-563.14

Table 6: Torrejon Rubio 1 decoration node 6: the sword.

```
read_eds(site,
         decor,
         doss = getwd(),
         edges = "edges",
         nodes = "nodes",
         dev = ".tsv")
```

`read_eds()` permits to read a `.csv` or `.tsv` dataframe, or a *shapefiles*, of edges. When the data source is a dataframe, the coordinates of the edges are retrieved from the nodes dataframe (Table 7).

```
R> caption <- paste0("First edge of the Torrejon Rubio 1 decoration, ",
+                    "the proximity between the sword (node 6) ",
+                    "and the fibula (node 5)")
R> eds.df <- read_eds(site = sit, decor = dec, dev = ".tsv",
+                     doss = system.file("extdata", package = "decorr"))
R> print(xtable::xtable(eds.df[1,],
+                       caption = caption,
+                       label = "tab:ed",
+                       size = 7),
+        include.rownames = FALSE,
+        table.placement = "!htbp")
```

site	decor	a	b	type	xa	ya	xb	yb
Torrejon Rubio	Torrejon Rubio 1	6	5	=	366.70	-563.14	490.12	-513.24

Table 7: First edge of the Torrejon Rubio 1 decoration, the proximity between the sword (node 6) and the fibula (node 5)

Once, the imgs, nodes and edges dataframes have been read, a geometric graph can be build and plotted over a specific decoration image with the `plot_dec_grph()` function.

```
plot_dec_grph(nodes = NULL,
              edges = NULL,
              site,
              decor,
              doss = getwd(),
              nd.var = 'id',
              nd.color = 'orange', nd.size = 1.7,
              lbl.color = 'black', lbl.size = 1.2,
              ed.color = c("orange", "blue"), ed.lwd = 4,
              img.format = "png")
```

The parameters `nodes` and `edges` concern respectively the nodes and edges dataframes. The parameters `site` and `decor` allow to choose a specific decoration. The `nd.var` parameter allows to decide which field of the nodes will be displayed as the label, by default this is the `nodes$id` field. For example, we can plot the graph of the Torrejon Rubio 1 decoration with the type of the nodes (Figure 11).

```
R> par(mar=c(0, 0, 0, 0))
R> sit <- "Torrejon Rubio" ; dec <- "Torrejon Rubio 1"
R> nds.df <- read_nds(site = sit, decor = dec, dev = ".tsv",
+                     doss = system.file("extdata", package = "decorr"))
R> eds.df <- read_eds(site = sit, decor = dec, dev = ".tsv",
+                     doss = system.file("extdata", package = "decorr"))
R> img.graph <- plot_dec_grph(nodes = nds.df,
+                              edges = eds.df,
+                              site = sit,
+                              decor = dec,
+                              doss = system.file("extdata",
+                                                 package = "decorr"),
+                              nd.var = "type", lbl.size = 2)
R> out <- magick::image_read(img.graph)
R> plot(out)
```

3.6. Decoration comparison functions

These functions allow to compare different graph decorations. The function `list_compar()` is used to compare common nodes and common edges depending on the node variable `nd.var`.

```
list_compar(lgrph,
            nd.var = "type",
            verbose = FALSE)
```

Whereas the `nd.var` (by default, the column `type`) refers to the nodes and can be any other column create by the user (e.g., `technique`, `color`, etc.), the comparisons always take into

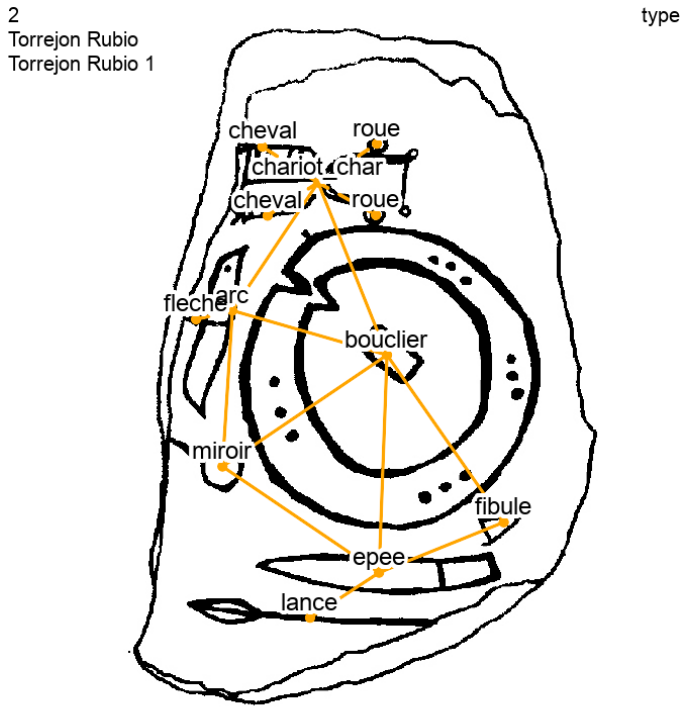

Figure 11: Torrejon Rubio 1 decoration

account the field `type` (=, +, >, ...) of the edges. Comparisons are done between one or more pairwise of decorations. In the training dataset, there are five (5) decorations in the default dataset, so there is $\frac{5!}{(5-2)!2!} = 10$ possible pairwise comparisons. This function calls `nds_compar()` and `eds_compar()`.

```
nds_compar(grphs,
           nd.var = "type")

eds_compar(grphs,
           nd.var = "type")
```

The functions `nds_compar()` and `eds_compar()` take a pair of graphs `grphs` and return respectively the common nodes and the common edges based on the `nd.var` field of nodes (by default, the column `type`).

The function `plot_compar()` allows to plot and save two figures side-by-side for one pairwise, or more, of decorations identifying common elements.

```
plot_compar(listg, graph2 = NULL, focus = "nodes",
            doss = getwd(),
            nd.color = c("orange", "red"), nd.size = c(0.5, 1),
            ed.color = c("orange", "red"), ed.width = c(1, 2),
            lbl.size = 0.5, img.format = "png", res = 300)
```

`plot_compar()` takes a list of graphs `listg`, created previously by `list_compar()`, and a vector with the ids of the graphs to compare (`graph2`). The function creates an image for each pairwise comparison with the decorations plotted side-by-side with elements: common nodes (`focus = "nodes"`, by default) or common edges (`focus = "edges"`). For each pairwise, if an element exists only in one decoration, it is displayed in orange by default (`nd.color[1]` or `ed.color[1]`). When an element is present on the two decorations, it is displayed in red by default (`nd.color[2]` or `ed.color[2]`). The function returns the paths to the created images.

Let us compare the decoration 1 and the decoration 4 and identify common edges (Figure 12).

```
R> par(mar=c(0,0,0,0))
R> g.compar <- list_compar(lgrph, "type")
R> eds_compar <- plot_compar(g.compar, c(1, 4),
+                            focus = "edges",
+                            doss = system.file("extdata", package = "decorr"))
R> plot(magick::image_trim(magick::image_read(eds_compar)))
```

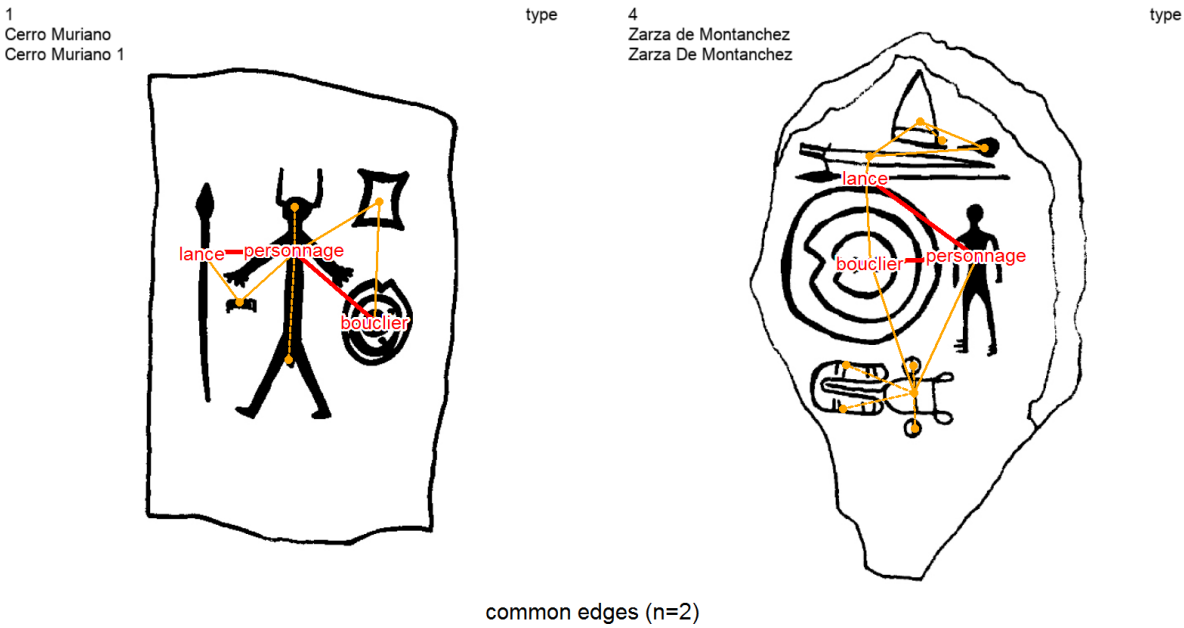

Figure 12: Comparison between decoration 1 (Cerro Muriano 1) and decoration 4 (Zarza de Montsanchez)

The comparison shows that the decoration 1 (Cerro Muriano 1) and the decoration 4 (Zarza de Montsanchez) have two (2) common edges: `lance -=- personnage` and `bouclier -=- personnage`.

The function `same_elements()` allows to count matching nodes and matching edges between decoration pairwises.

```
same_elements(lgrph,
              nd.var = "type",
              focus = "nodes")
```

The `same_elements()` function take the list of graphs (`lgrph`) produced by `list_dec()` and the node variable on which the comparison will be done, by default `type` (`nd.var = "type"`). The result is a square matrix between all pairwise comparisons with the number of common elements (nodes or edges in the cells). The diagonal of the square matrix shows the total number of element for a given decoration. For example, we can compute the matrix of common edges (Table 8).

```
R> df.same_edges <- same_elements(lgrph, "type", "edges")
R> caption <- "Number of common edges between all graph decoration pairwises"
R> print(xtable::xtable(df.same_edges,
+                       caption = caption,
+                       label = "tab:mat.ed",
+                       size = 8,
+                       digits = c(0)),
+        table.placement="!htbp",
+        include.rownames = TRUE)
```

	1	2	3	4	5
1	8	0	1	2	0
2	0	15	3	7	1
3	1	3	10	1	2
4	2	7	1	14	1
5	0	1	2	1	4

Table 8: Number of common edges between all graph decoration pairwises

The edges comparisons between the decoration 1 and the decoration 4 show that they have two (2) common edges, see also Figure 12.

4. Illustrations

In order to demonstrate the main insight of a graph-based analysis of the decorations, we will compare two classifications. The first classification is based on the counts of common nodes. Since the precise location of the nodes is usualy not registred, his is the most commonly analysis applied on large series. The second classification is based on the presence of common edges.

At first, the Late Bronze Age phase of Ibahernando stelae must be selected (ie, the Roman latin writing must be removed). After the selection of the graph elements of Ibahernando stelae (decoration 5), nodes and edges of the node 2 connected component, the lance (spear), is selected with `contemp_nds()`, a new graph object is created (`grph`) and used to update the list of graphs (`lgrph`).

```
R> dataDir <- system.file("extdata", package = "decorr")
R> nds.df <- read_nds(site = "Ibahernando",
+                     decor = "Ibahernando",
+                     doss = dataDir)
R> eds.df <- read_eds(site = "Ibahernando",
+                     decor = "Ibahernando",
+                     doss = dataDir)
R> l_dec_df <- contemp_nds(nds.df, eds.df, selected.nd = 2)
R> l_dec_df[[1]]$site <- l_dec_df[[1]]$decor <- NULL
R> l_dec_df[[2]]$site <- l_dec_df[[2]]$decor <- NULL
R> grph <- graph_from_data_frame(l_dec_df[[2]], vertices = l_dec_df[[1]],
+                                directed = FALSE)
R> grph$site <- grph$decor <- "Ibahernando"
R> grph$name <- 5
R> lgrph[[5]] <- grph
```

Then, `same_elements()` is done on nodes and edges (Table 9)

```
R> df.same_edges <- same_elements(lgrph, "type", "edges")
R> df.same_nodes <- same_elements(lgrph, "type", "nodes")
```

Common nodes

	1	2	3	4	5
1	7	2	3	4	2
2	2	12	5	9	3
3	3	5	6	4	3
4	4	9	4	12	3
5	2	3	3	3	3

Common edges

	1	2	3	4	5
1	8	0	1	2	0
2	0	15	3	7	1
3	1	3	10	1	2
4	2	7	1	14	1
5	0	1	2	1	2

Table 9: Tables showing the number of common nodes and common edges between each pair of decorations

Once the heatmap matrices calculated, the distance matrices are calculated with the native `dist()` function (R Core Team 2019) (Table 10).

Distance matrix on nodes

	1	2	3	4	5
1		12.45	5.92	11.14	5.29
2			8.72	4.80	11.00
3				9.22	3.87
4					11.05
5					

Distance matrix on edges

	1	2	3	4	5
1		17.86	12.00	15.17	8.43
2			15.20	11.00	15.30
3				16.37	8.31
4					14.53
5					

Table 10: Distance matrices on nodes and common edges between each pair of decorations

For both nodes and edges, the results show that the most important differences are between the decoration 1 and the decoration 2: for nodes, dist = 12.45 ; for edges, dist = 17.86. At

the opposite, the decorations showing the more similar compositions are decoration 3 and decoration 5: for nodes, dist = 3.87 ; for edges, dist = 8.31. The compared decorations, with their common elements, can be plotted with `plot_compar()`. A hierachical clustering (HC) of the distance matrices, with the native function `hclust()` (R Core Team 2019), summarize these similarites/dissimilarities (Figure 13)

```
R> par(mfrow=c(1, 2), mar=c(0.2,4,1.2,0.5))
R> hc.nds <- hclust(dist.nodes, method = "ward.D")
R> hc.eds <- hclust(dist.edges, method = "ward.D")
R> plot(hc.nds, main = "common nodes", cex = 1.1)
R> plot(hc.eds, main = "common edges", cex = 1.1)
```

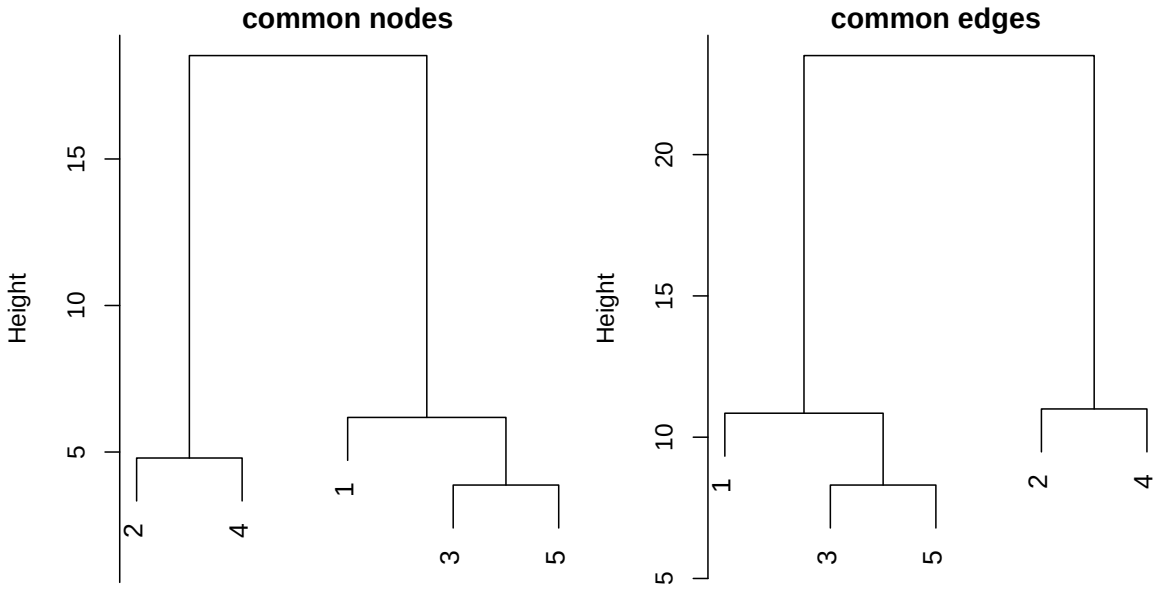

Figure 13: Hierachical clusterings on decoration common elements

Even if in this example the clustering of decorations on common nodes and the clustering of decorations on common edges give the same results, another distribution of the same nodes, could give two different clusterings. The clustering on common nodes and the clustering on common edges can be compared with the `tanglegram()` function from the **dendextend** package (Galili 2015) (Figure 14)

```
R> suppressPackageStartupMessages(library(dendextend))
R> suppressPackageStartupMessages(library(dplyr))
R> par(mfrow=c(1, 2))
R> dend.nds <- as.dendrogram (hc.nds)
R> dend.eds <- as.dendrogram (hc.eds)
R> dendlist(dend.nds, dend.eds) %>%
+    untangle(method = "step1side") %>%
+    tanglegram(columns_width = c(6, 1, 6),
+               main_left = "common nodes",
```

```
+                main_right = "common edges",
+                lab.cex = 1.5,
+                cex_main = 1.5,
+                highlight_branches_lwd = F)
```

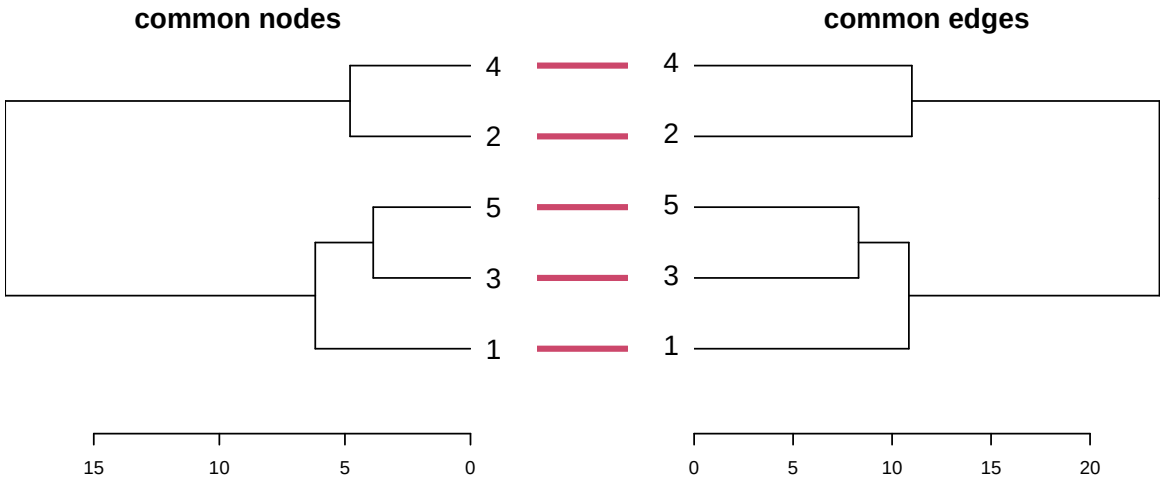

Figure 14: Comparison of hierachical clusterings on decoration common elements, nodes (left) and edges (right)

5. Summary and discussion

In Archaeology, the iconography has been considered as a privileged elements to the identification of cultural facies (Renfrew and Bahn 1991). By integrating the recording, the management and the analysis process in a simple manner, within the framework of graph Theory and binary topological telationships (*birel*), the **decorr** framework opens possibilities of studying graphical systems with normalized indexes over large series (Later Paleolithic cave paintings, *cups-and-rings* art, Valcamonica Iron Age *pittoti*, etc.) a long period of time, and on heterogeneous source data.

The **decorr** framework is grounded on a parsimonious model, where Tobler's '*near things*' (Tobler 1970) of any iconographical content is recorded with a minimal of *a priori* definitions. Those definitions only concern the graph elements: graphical units variables (ie, nodes variables) and types of relations they share (edges). The current edge notation with *normal* edge, *attribute* edge, and *overlap* edge allow to describe most cases of Prehistorical iconography, but can also be extended to accept new types like, for example, the *interact* edge when two graphical units interact (ie, when an action is visible). For the nodes, we have only considered the iconographical `type` (`personnage`, `casque`, etc.) as the studied variable (`nd.var`), but the package user can create and select any other structured vocabularies: `technique`, `color`, etc. Tree structures for hierarchical datasets allow generalization processes (up, the parent level) and specification processes (down, the children level). Such a formalism can be used to weight the differences between nodes, to conduct analysis with different level of precision or to overcome idiosynchratic typologies issues. To record the relative chronology

of archaeological layers (in a case of an excavation) or the relative chronology of iconographical layers (in case of a decorated support), archaeologists use quite commonly a hierachical dataset to model : the Harris matrix. The *overlap* edge, by recording the anterior-posterior relation between pairwises of graphical units, is first operator permiting to trace the Harris matrix directly of the studied decoration. The Harris matrix can be then traced with R data tree package (**data.tree**, Glur (2019)) and integrated into an interactive Reingold-Tilford tree diagram (**collapsibleTree**, Khan (2018)).

Here, the undecorated parts of the support – the background – is inactive and considered as homogeneous it is possible to consider but a crack on a rock or a carene on a pottery (Huet 2018), like relevant topologic entities to understand the iconographical composition. Then a new class of nodes (eg, topographical units), beside the graphical units, can be registred to create a n-mode graph (graph approach), or adding a new raster (eg, a Digital Elevation Model) within a decoration GIS project file (geostatistical approach).

Acknowledgments

A part of the IT development of the **decorr** package has been done in the frame of a postdoctoral position (2015-16) supported by LabEx ARCHIMEDE from "Investissement d'Avenir" program ANR-11-LABX-0032-01.

References

Alexander C (2008). "The Bedolina map – an exploratory network analysis." In A Posluschny, K Lambers, I Herzog (eds.), *Layers of Perception. Proceedings of the 35th International Conference on Computer Applications and Quantitative Methods in Archaeology (CAA), Berlin, 2.-6. April 2007*, pp. 366–371. Koll. Vor- u. FrÃijhgesch. doi:https://doi.org/10.11588/propylaeumdok.00000512.

Basch MA (1966). *Las estelas decoradas del Suroeste peninsular*, volume 8. Editorial CSIC-CSIC Press.

Bivand R, Keitt T, Rowlingson B (2019). *rgdal: Bindings for the 'Geospatial' Data Abstraction Library*. R package version 1.4-7, URL https://CRAN.R-project.org/package=rgdal.

Clarke DL (2014). *Analytical archaeology*. Routledge.

Csardi G, Nepusz T (2006). "The igraph software package for complex network research." *InterJournal*, **Complex Systems**, 1695. URL http://igraph.org.

De Saussure F (1989). *Cours de linguistique générale*, volume 1. Otto Harrassowitz Verlag.

d'Errico F, Nowell A (2000). "A new look at the Berekhat Ram figurine: implications for the origins of symbolism." *Cambridge Archaeological Journal*, **10**(1), 123–167.

Díaz-Guardamino Uribe M (2010). *Las estelas decoradas en la Prehistoria de la Península Ibérica*. Ph.D. thesis, Universidad Complutense de Madrid, Servicio de Publicaciones.

Galili T (2015). "dendextend: an R package for visualizing, adjusting, and comparing trees of hierarchical clustering." *Bioinformatics*. doi:10.1093/bioinformatics/btv428. URL https://academic.oup.com/bioinformatics/article/31/22/3718/240978/dendextend-an-R-package-for-visualizing-adjusting.

Glur C (2019). *data.tree: General Purpose Hierarchical Data Structure*. R package version 0.7.11, URL https://CRAN.R-project.org/package=data.tree.

Huet T (2018). "Geometric graphs to study ceramic decoration." In M Matsumoto, E Uleberg (eds.), *Exploring Oceans of Data, proceedings of the 44th Conference on Computer Applications and Quantitative Methods in Archaeology, CAA 2016*, pp. 311–324. Archaeopress.

Khan A (2018). *collapsibleTree: Interactive Collapsible Tree Diagrams using 'D3.js'*. R package version 0.1.7, URL https://CRAN.R-project.org/package=collapsibleTree.

Mathis P (2003). "Puissance et insuffisances des graphes pour la description et la modélisation des réseaux." *Graphes et réseaux, modélisation multiniveau*, pp. 19–47.

Ooms J (2018). "Magick: advanced graphics and image-processing in R." *CRAN. R package version*, **1**.

Pérez SC (2001). *Estelas de guerrero y estelas diademadas: la precolonialzación y formación del mundo tartésico*. Ediciones Bellaterra.

R Core Team (2019). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Renfrew C, Bahn PG (1991). *Archaeology: theories, methods and practice*, volume 2. Thames and Hudson London.

Snow G (2020). *TeachingDemos: Demonstrations for Teaching and Learning*. R package version 2.12, URL https://CRAN.R-project.org/package=TeachingDemos.

Tobler WR (1970). "A computer movie simulating urban growth in the Detroit region." *Economic geography*, **46**(sup1), 234–240.

Affiliation:

Thomas Huet
CNRS-UMR 5140
Archeologie des Societes Mediterraneennes
Universite Paul Valery
route de Mende
Montpellier 34199, France
E-mail: thomashuet7@gmail.com

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
doi:10.18637/jss.v000.i00 *Accepted:* yyyy-mm-dd