From Spider-Man to Hero – Archetypal Analysis in R

Manuel J. A. Eugster
Ludwig-Maximilians-Universität München

Friedrich Leisch
Ludwig-Maximilians-Universität München

Abstract

This introduction to the R package **archetypes** is a (slightly) modified version of Eugster and Leisch (2009), published in the *Journal of Statistical Software*.

Archetypal analysis has the aim to represent observations in a multivariate data set as convex combinations of extremal points. This approach was introduced by Cutler and Breiman (1994); they defined the concrete problem, laid out the theoretical foundations and presented an algorithm written in Fortran. In this paper we present the R package **archetypes** which is available on the Comprehensive R Archive Network. The package provides an implementation of the archetypal analysis algorithm within R and different exploratory tools to analyze the algorithm during its execution and its final result. The application of the package is demonstrated on two examples.

Keywords: archetypal analysis, convex hull, R.

1. Introduction

The Merriam-Webster Online Dictionary (2008) defines an archetype as *the original pattern or model of which all things of the same type are representations or copies.* The aim of archetypal analysis is to find "pure types", the archetypes, within a set defined in a specific context. The concept of archetypes is used in many different areas, the set can be defined in terms of literature, philosophy, psychology and also statistics. Here, the concrete problem is to find a few, not necessarily observed, points (archetypes) in a set of multivariate observations such that all the data can be well represented as convex combinations of the archetypes. The title of this article illustrates the concept of archetypes on the basis of archetypes in literature: the *Spider-Man* personality belongs to the generic *Hero* archetype, and archetypal analysis tries to find this coherence.

In statistics archetypal analysis was first introduced by Cutler and Breiman (1994). In their paper they laid out the theoretical foundations, defined the concrete problem as a nonlinear least squares problem and presented an alternating minimizing algorithm to solve it. It has found applications in different areas, with recently grown popularity in economics, e.g., Li, Wang, Louviere, and Carson (2003) and Porzio, Ragozini, and Vistocco (2008). In spite of the rising interest in this computer-intensive but numerically sensitive method, no "easy-to-use" and freely available software package has been developed yet. In this paper we present the software package **archetypes** within the R statistical environment (R Development Core Team 2008) which provides an implementation of the archetypal analysis algorithm. Additionally, the package provides exploratory tools to visualize the algorithm during the minimization steps and its final result. The newest released version of **archetypes** is always available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=archetypes`.

The paper is organized as follows: In Section 2 we outline the archetypal analysis with its different conceptual parts. We present the theoretical background as far as we need it for a sound introduction of our implementation; for a complete explanation we refer to the original paper. Section 3 demonstrates how to use **archetypes** based on a simple artificial data set, with details about numerical problems and the behavior of the algorithm. Section 4 presents a simulation study to show how the implementation scales with numbers of observations, attributes and archetypes. In Section 5 we show a real word example – the archetypes of human skeletal diameter measurements. Section 6 concludes the article with future investigations.

2. Archetypal analysis

Given is an $n \times m$ matrix X representing a multivariate data set with n observations and m attributes. For a given k the archetypal analysis finds the matrix Z of k m-dimensional archetypes according to the two fundamentals:

(1) The data are best approximated by convex combinations of the archetypes, i.e., they minimize

$$\text{RSS} = \|X - \alpha Z^\top\|_2$$

with α, the coefficients of the archetypes, an $n \times k$ matrix; the elements are required to be greater equal 0 and their sum must be 1, i.e., $\sum_{j=1}^{k} \alpha_{ij} = 1$ with $\alpha_{ij} \geq 0$ and $i = 1, \ldots, n$. $\|\cdot\|_2$ denotes an appropriate matrix norm.

(2) The archetypes are convex combinations of the data points:

$$Z = X^\top \beta$$

with β, the coefficients of the data set, a $n \times k$ matrix where the elements are required to be greater equal 0 and their sum must be 1, i.e., $\sum_{i=1}^{n} \beta_{ji} = 1$ with $\beta_{ji} \geq 0$ and $j = 1, \ldots, k$.

These two fundamentals also define the basic principles of the estimation algorithm: it alternates between finding the best α for given archetypes Z and finding the best archetypes Z for given α; at each step several convex least squares problems are solved, the overall RSS is reduced successively.

With a view to the implementation, the algorithm consists of the following steps:

Given the number of archetypes k:

1. Data preparation and initialization: scale data, add a dummy row (see below) and initialize β in a way that the the constraints are fulfilled to calculate the starting archetypes Z.

2. Loop until RSS reduction is sufficiently small or the number of maximum iterations is reached:

2.1. Find best α for the given set of archetypes Z: solve n convex least squares problems $(i = 1, \dots, n)$

$$\min_{\alpha_i} \frac{1}{2} \|X_i - Z\alpha_i\|_2 \text{ subject to } \alpha_i \geq 0 \text{ and } \sum_{j=1}^{k} \alpha_{ij} = 1.$$

2.2. Recalculate archetypes $\tilde{Z}$: solve system of linear equations $X = \alpha \tilde{Z}^\top$.

2.3. Find best β for the given set of archetypes $\tilde{Z}$: solve k convex least squares problems $(j = 1, \dots, k)$

$$\min_{\beta_j} \frac{1}{2} \|\tilde{Z}_j - X\beta_j\|_2 \text{ subject to } \beta_j \geq 0 \text{ and } \sum_{i=1}^{n} \beta_{ji} = 1.$$

2.4. Recalculate archetypes Z: $Z = X\beta$.

2.5. Calculate residual sum of squares RSS.

3. Post-processing: remove dummy row and rescale archetypes.

The algorithm has to deal with several numerical problems, i.e. systems of linear equations and convex least squares problems. In the following we explain each step in detail.

Solving the convex least squares problems: In Step 2.1 and 2.3 several convex least squares problems have to be solved. Cutler and Breiman (1994) use a penalized version of the non-negative least squares algorithm by Lawson and Hanson (1974) (as general reference see,e.g., Luenberger 1984). In detail, the problems to solve are of the form $\|u - Tw\|_2$ with u, w vectors and T a matrix, all of appropriate dimensions, and the non-negativity and equality constraints. The penalized version adds an extra element M to u and to each observation of T; then

$$\|u - Tw\|_2 + M^2\|1 - w\|_2$$

is minimized under non-negativity restrictions. For large M, the second term dominates and forces the equality constraint to be approximately satisfied while maintaining the non-negativity constraint. The hugeness of the value M varies from problem to problem and thus can be seen as a hyperparameter of the algorithm. Default value in the package is 200.

Solving the system of linear equations: In Step 2.2 the system of linear equations

$$\tilde{Z} = \alpha^{-1}X$$

has to be solved. A lot of methods exist, one approach is the Moore-Penrose pseudoinverse which provides an approximated unique solution by a least squares approach: given the pseudoinverse α^+ of α,

$$\tilde{Z} = \alpha^+ X,$$

is solved. Another approach is the usage of QR decomposition: $\alpha = QR$, where Q is an orthogonal and R an upper triangular matrix, then

$$\tilde{Z} = Q^\top X R^{-1},$$

is solved. Default approach in the package is the QR decomposition using the `solve()` function.

Calculating the residual sum of squares: In Step 2.5 the RSS is calculated. It uses the spectral norm (see, e.g., Golub and Loan 1996). The spectral norm of a matrix X is the largest singular value of X or the square root of the largest eigenvalue of X^*X,

$$\|X\|_2 = \sqrt{\lambda_{max}(X^*X)},$$

where X^* is the conjugate transpose of X.

Avoiding local minima: Cutler and Breiman (1994) show that the algorithm converges in all cases, but not necessarily to a global minimum. Hence, the algorithm should be started several times with different initial archetypes. It is important that these are not too close together, this can cause slow convergence or convergence to a local minimum.

Choosing the correct number of archetypes: As in many cases there is no rule for the correct number of archetypes k. A simple method the determine the value of k is to run the algorithm for different numbers of k and use the "elbow criterion" on the RSS where a "flattening" of the curve indicates the correct value of k.

Approximation of the convex hull: Through the definition of the problem, archetypes lie on the boundary of the convex hull of the data. Let N be the number of data points which define the boundary of the convex hull, then Cutler and Breiman (1994) showed: if $1 < k < N$, there are k archetypes on the boundary which minimize RSS; if $k = N$, exactly the data points which define the convex hull are the archetypes with RSS $= 0$; and if $k = 1$, the sample mean minimizes the RSS. In practice, these theoretical results can not always be achieved as we will see in the following two sections.

3. Using package archetypes

The package is loaded into R using the `library()` or `require()` command:

```
R> library("archetypes")
```

```
Loading required package: nnls
```

It requires the packages **nnls** (Mullen and van Stokkum 2007) for solving the convex least squares problems.

We use a simple artificial two-dimensional data set to explain the usage of the implementation, and the behavior of the archetypal analysis at all. The advantage is that we can imagine the results and simply visualize them, Section 5 then shows a more realistic example. Note that in the following the plot commands do not produce exactly the shown graphics concerning primitives coloring, width, etc.; to increase readability we have reduced the presented commands to the significant arguments. E.g., if we use a different plotting symbol in a scatter plot the corresponding (`pch = ...`) will be omitted.

```
R> data("toy")
R> plot(toy)
```

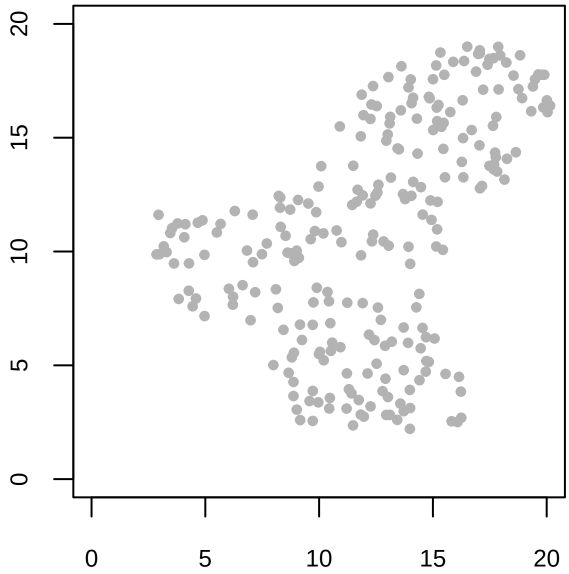

Figure 1: Data set `toy`.

Data set `toy` consists of the two attributes x and y, and 250 observations. According to the shape of the data, it seems to be a good idea to apply archetypal analysis with $k = 3$ archetypes.

```
R> suppressWarnings(RNGversion("3.5.0"))
R> set.seed(1986)
R> a <- archetypes(toy, 3, verbose = TRUE)

1: rss = 0.01676035, improvement = 0.06191059
2: rss = 0.01159216, improvement = 0.00516819
3: rss = 0.00926030, improvement = 0.00233186
4: rss = 0.00788877, improvement = 0.00137153
5: rss = 0.00734550, improvement = 0.00054327
6: rss = 0.00748457, improvement = -0.00013907
```

During the fit, the function reports its improvement and stops after a maximum number of iterations (default is `maxIterations = 100`) or if the improvement is less than a defined value (default is `minImprovement = sqrt(.Machine$double.eps)`). As basis for our further research, the implementation is a flexible framework where the problem solving mechanisms of the individual steps can be exchanged. The default values are the “original ones” described in the previous section (`family = archetypesFamily()`). The result is an S3 `archetypes` object,

```
R> a

Archetypes object

archetypes(data = toy, k = 3, verbose = TRUE)
```

```
Convergence after 6 iterations
with RSS = 0.007484568.
```

containing the three final archetypes:

```
R> parameters(a)

             x         y
[1,] 14.733401  2.315634
[2,]  2.859105  9.905305
[3,] 18.695236 18.673817
```

The `plot()` function visualizes archetypes for two-dimensional data sets; for higher-dimensional data sets parallel coordinates are used.

```
R> xyplot(a, toy, chull = chull(toy))
R> xyplot(a, toy, adata.show = TRUE)
```

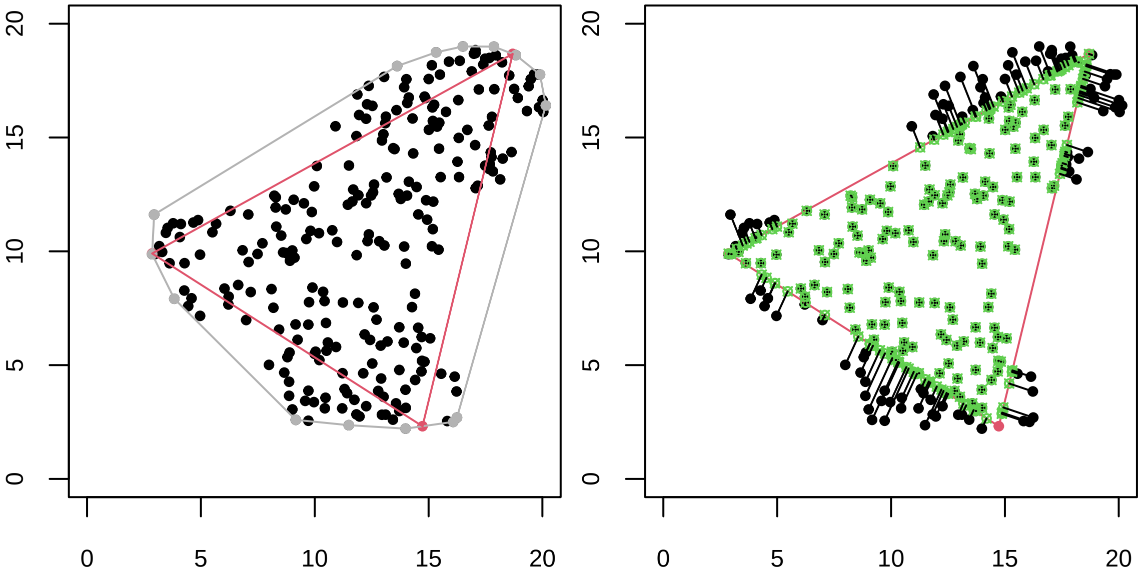

Figure 2: Visualization of three archetypes.

The left plot of Figure 2 shows the archetypes, their approximation of the convex hull (red dots and lines) and the convex hull (black dots and lines) of the data. The right plot of Figure 2 additionally shows the approximation of the data through the archetypes and the corresponding α values (green symbols, and grey connection lines); as we can see, all data points outside the approximated convex hull are mapped on its boundary. This plot is based on an idea and MATLAB source code of Pailthorpe (2008).

With `saveHistory = TRUE` (which is set per default) each step of the execution is saved and we can examine the archetypes in each iteration using the `a$history` memento object; the initial archetypes, for example, are `a$history$get(0)`. This can be used to create an "evolution movie" of the archetypes,

```
R> movieplot(a, toy)
```

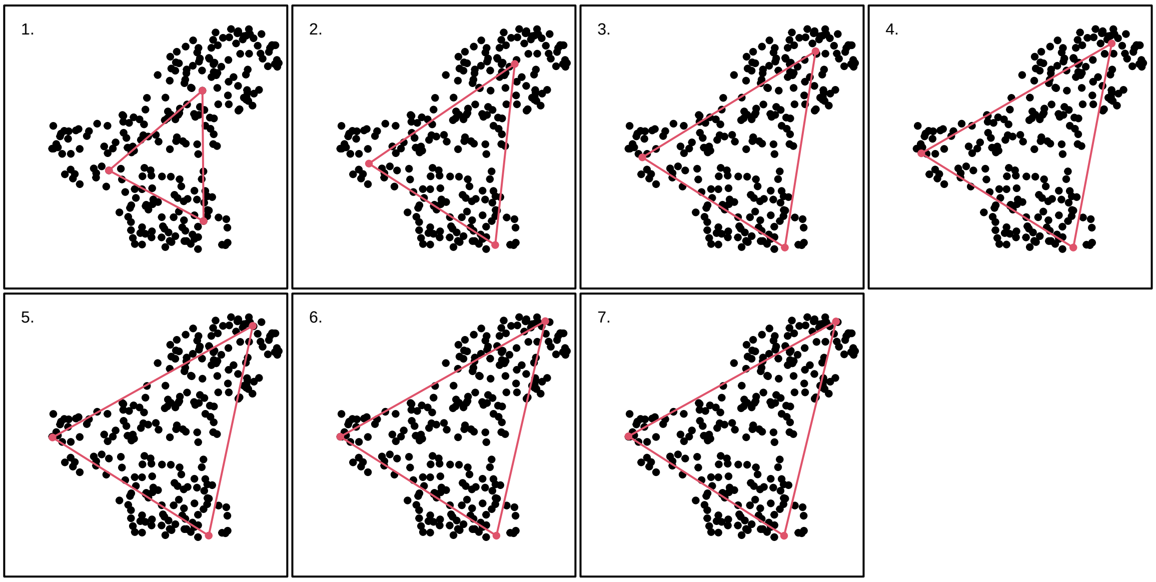

Figure 3: Visualization of the algorithm iterations.

The figure shows the plots of the seven steps (the random initialization and the six iterations) from top to bottom and left to right. In each step the three archetypes move further to the three corners of the data set. A movie of the approximated data is shown when setting parameter `show = "adata"`[1].

In the previous section we mentioned that the algorithm should be started several times to avoid local minima. This is done using the `stepArchetypes()` function; it passes all arguments to the `archetypes()` function and additionally has the argument `nrep` which specifies the number of repetitions.

```
R> set.seed(1986)
R> a4 <- stepArchetypes(data = toy, k = 3, verbose = FALSE, nrep = 4)
```

The result is an S3 `stepArchetypes` object,

```
R> a4

StepArchetypes object

stepArchetypes(data = toy, k = 3, nrep = 4, verbose = FALSE)
```

where `summary()` provides an overview of each repetition by showing the final residual sum of squares and number of iterations:

[1] Real animations are as Flash movies along with this paper and available from http://www.jstatsoft.org/v30/i08/.

```
R> summary(a4)

StepArchetypes object

stepArchetypes(data = toy, k = 3, nrep = 4, verbose = FALSE)

k=3:
Convergence after 6 iterations
with RSS = 0.007484568.
Convergence after 19 iterations
with RSS = 0.007269786.
Convergence after 14 iterations
with RSS = 0.007269673.
Convergence after 13 iterations
with RSS = 0.007274116.
```

There are no huge differences in the residual sum of squares, thus if there are different local minima then they are all equally good. But the following plot shows that the repetition starts all nearly found the same final archetypes (and thus the same local minima),

```
R> xyplot(a4, toy)
```

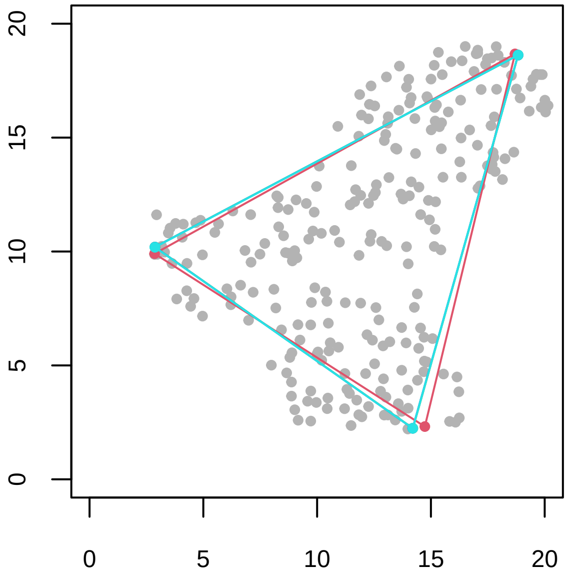

Figure 4: Visualization of different repetitions.

However, the model of repetition 3 has the lowest residual sum of squares and is the best model:

```
R> bestModel(a4)

Convergence after 14 iterations
with RSS = 0.007269673.
```

At the beginning of the example we decided by looking at the data that three archetypes may be a good choice. Visual inspection does not necessarly lead to the best choice, and is not an option for higher-dimensional data sets. As already mentioned in the previous section, one simple way to choose a good number of archetypes is to run the algorithm for different numbers of k and use the "elbow criterion" on the residual sum of squares. The `stepArchetypes()` function allows a vector as value of argument k and executes for each k_i the `archetypes()` function *nrep* times.

```
R> set.seed(1986)
R> as <- stepArchetypes(data = toy, k = 1:10, verbose = FALSE, nrep = 4)
```

```
There were 23 warnings (use warnings() to see)
```

The occurred warnings indicate that errors occured during the execution, in this case, singular matrices in solving the linear equation system in Step 2.2 as from $k = 4$:

```
R> warnings()
```

```
Warnings:
1: In archetypes(..., k = k[i], verbose = verbose) ... :
  k=4: Error in qr.solve(alphas %*% t(alphas)): singular matrix 'a' in solve
2: In archetypes(..., k = k[i], verbose = verbose) ... :
  k=5: Error in qr.solve(alphas %*% t(alphas)): singular matrix 'a' in solve
3: In archetypes(..., k = k[i], verbose = verbose) ... :
  k=5: Error in qr.solve(alphas %*% t(alphas)): singular matrix 'a' in solve
[...]
```

In these cases the residual sum of squares is `NA`:

```
R> rss(as)
```

```
             r1          r2          r3          r4
k1  0.075569637 0.075569637 0.075569637 0.075569637
k2  0.047510401 0.047510402 0.047510404 0.047510417
k3  0.007370771 0.007370765 0.007370774 0.007565407
k4  0.005124791          NA 0.004826269 0.004970543
k5  0.005241236          NA          NA          NA
k6           NA          NA          NA          NA
k7           NA 0.001216349          NA          NA
k8           NA          NA          NA          NA
k9           NA          NA          NA          NA
k10          NA          NA          NA          NA
```

And all errors occured during the first iteration,

```
R> t(sapply(as, function(a) sapply(a, '[[', 'iters')))
```

```
      [,1] [,2] [,3] [,4]
 [1,]    2    2    2    2
 [2,]   13   17   14   19
 [3,]   72   72   72    7
 [4,]   20    1   50   21
 [5,]   66    1    1    1
 [6,]    1    1    1    1
 [7,]    1  100    1    1
 [8,]    1    1    1    1
 [9,]    1    1    1    1
[10,]    1    1    1    1
```

which is an indication for an afflicted random initialisation. If warnings occur within repetitions for k_i archetypes, the pretended meaningful solutions (according to the RSS) have to be examined carefully; Section 3.1 illustrates a pretended meaningful solution where the plot then uncovers problems. But up to $k = 5$ there is always at least one start with a meaningful result and the residual sum of squares curve of the best models shows that by the "elbow criterion" three or maybe seven is the best number of archetypes:

```
R> screeplot(as)
```

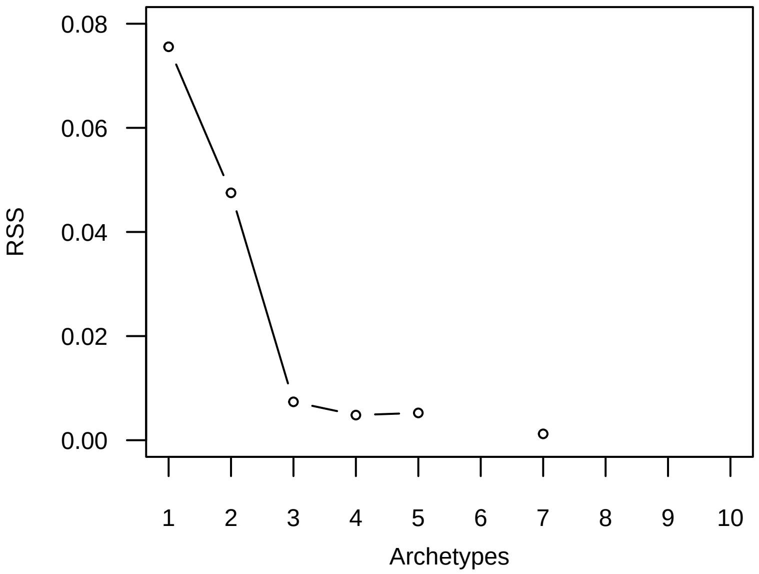

Figure 5: Screeplot of the residual sum of squares.

We already have seen the three archetypes in detail; the seven archetypes of the best repetition and their approximation of the data are:

```
R> a7 <- bestModel(as[[7]])
R> xyplot(a7, toy, chull = chull(toy))
R> xyplot(a7, toy, adata.show = TRUE)
```

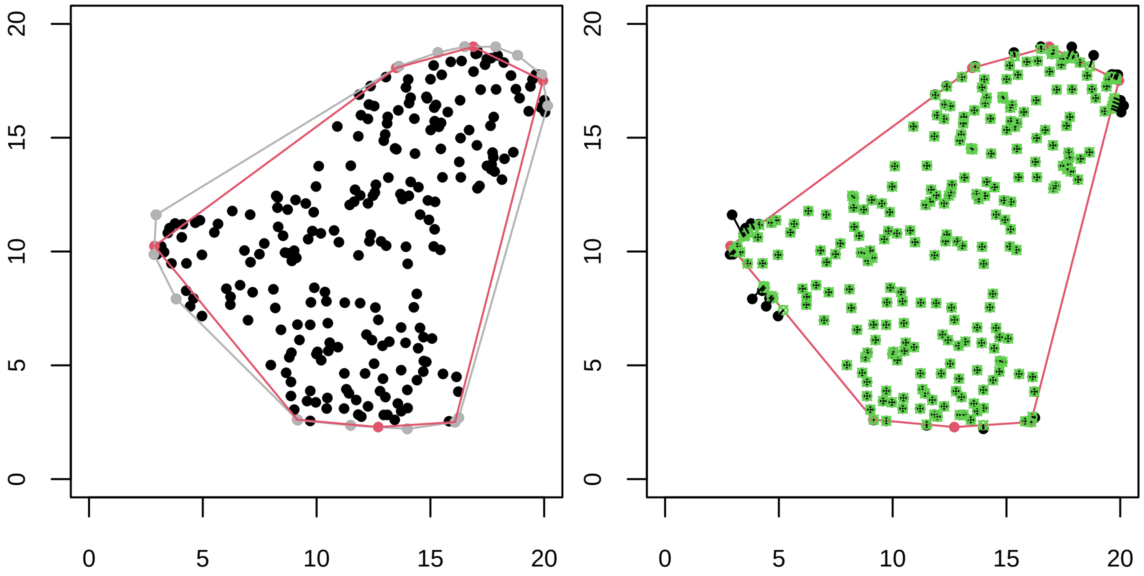

Figure 6: Visualization of seven archetypes.

The approximation of the convex hull is now clearly visible.

3.1. Alternative numerical methods

As we mentioned in Section 2, there are many ways to solve linear equation systems. One other possibility is the Moore-Penrose pseudoinverse:

```
R> set.seed(1986)
R> gas <- stepArchetypes(data = toy, k = 1:10,
+                        family = archetypesFamily("original",
+                        zalphasfn = archetypes:::ginv.zalphasfn),
+                        verbose = FALSE, nrep = 4)

Loading required package: MASS
There were 23 warnings (use warnings() to see)
```

We use the `ginv()` function from the **MASS** package to calculate the pseudoinverse. The function ignores ill-conditioned matrices and “just solves the linear equation system”, but the `archetypes` function throws warnings of ill-conditioned matrices if the matrix condition number κ is bigger than an upper bound (default is `maxKappa = 1000`):

```
R> warnings()

Warnings:
1: In archetypes(..., k = k[i], verbose = verbose) ... :
  k=4: alphas > maxKappa
2: In archetypes(..., k = k[i], verbose = verbose) ... :
  k=5: alphas > maxKappa
```

```
3: In archetypes(..., k = k[i], verbose = verbose) ... :
  k=5: alphas > maxKappa
[...]
```

In comparison with the QR decomposition, the warnings occured for the same number of archetypes k_i during the same repetition. In most of these cases the residual sum of squares is about 12,

```
R> rss(gas)
```

```
             r1          r2          r3          r4
k1  0.075569637 0.075569637 0.075569637 0.075569637
k2  0.047510401 0.047510402 0.047510404 0.047510417
k3  0.007370771 0.007370765 0.007370774 0.007565407
k4  0.005124791 0.134572123 0.004826269 0.004970543
k5  0.005241236 0.005368693 0.090090637 0.005121524
k6  0.005394369 0.115877068 0.004796068 0.089518648
k7  0.158016364 0.001216349 0.001182433 0.004105327
k8  0.150116233 0.005618620 0.087120299 0.159113751
k9  0.108938969 0.129585851 0.115504499 0.005207711
k10 0.102716292 0.125476701 0.096505495 0.120292145
```

and the randomly chosen initial archetypes “collapse” to the center of the data as we exemplarily see for $k = 9$, $r = 3$:

```
R> movieplot(gas[[9]][[3]], toy)
```

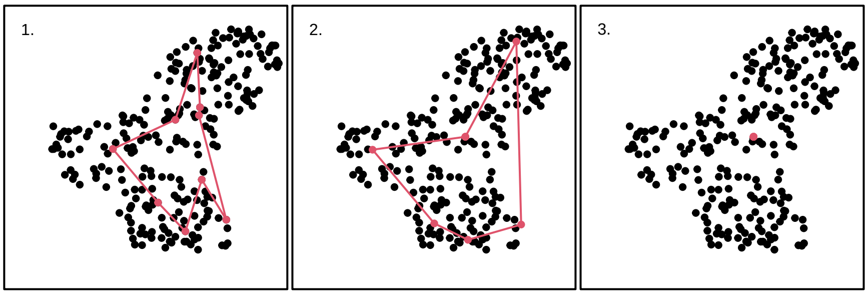

Figure 7: Visualization of algorithm iterations until “collapsing”.

The figure shows the four steps (from left to right), the random initialization and the three iterations until all archetypes are in the center of the data.

All other residual sum of squares are nearly equivalent to the ones calculated with QR decomposition. Further investigations would show that three or maybe seven is the best number of archetypes, and in case of $k = 3$ nearly the same three points are the best archetypes. An interesting exception is the case $k = 7, r = 2$; the residual sum of squares is exactly the same, but not the archetypes. The plots of the archetypes and their approximation of the data:

```
R> ga7 <- bestModel(gas[[7]])
R> xyplot(ga7, toy, chull = chull(toy))
R> xyplot(ga7, toy, adata.show = TRUE)
```

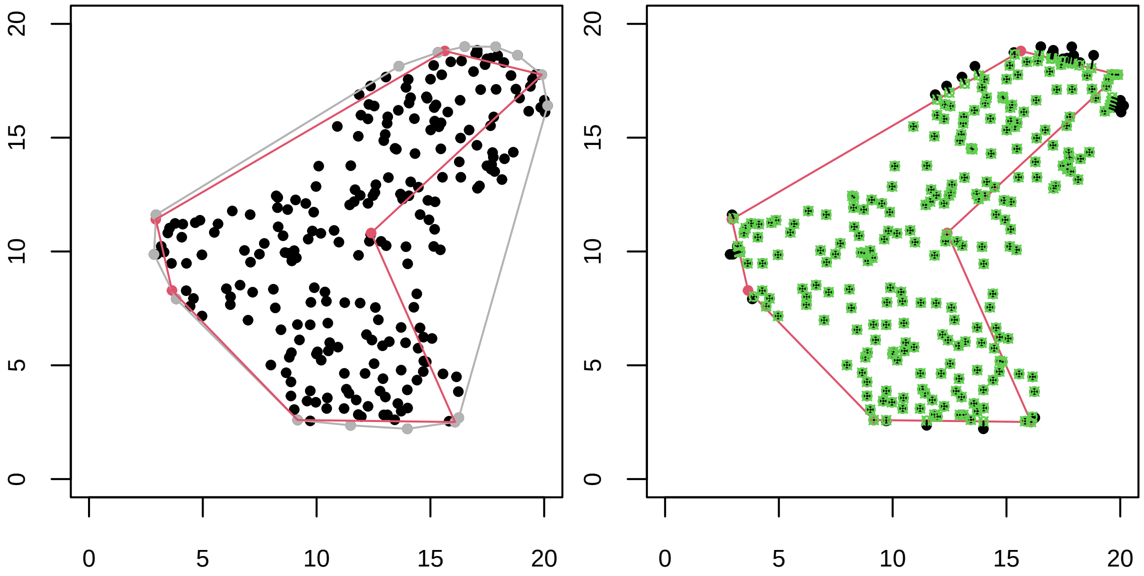

Figure 8: Visualization of seven archetypes; one archetype "collapsed".

Interesting is the one archetype in the center of the data set and especially the approximation of the data in the right area of it. As the data are approximated by a linear combination of archetypes and non-negative α, the only possibility for this kind of approximation is when α for this archetype is always zero:

```
R> apply(coef(ga7, 'alphas'), 2, range)
```

```
     [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
[1,]    0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[2,]    0 0.9999845 0.9548158 0.9999926 0.9789887 0.9999594 0.9940115
```

As we can see, α of archetype 1 (column one) is 0 for all data points. Theoretically, this is not possible, but ill-conditioned matrices during the fit process lead to such results in practice. The occurred warnings (`k=7:  alphas > max.kappa`) notify that solving the convex least squares problems lead to the ill-conditioned matrices. Our simulations showed that this behavior mostly appears when requesting a relatively large number of archetypes in relation to size of the data set.

4. Computational complexity

In Section 1 we stated that the archetypal analysis algorithm is computer-intensive; in this section we now present a simulation to show how the implementation scales with numbers of observations, attributes and archetypes. In general, the speed of the algorithm is determined by the efficiency of the convex least squares method. The penalized non-negative least squares method is quite slow, but is appealing because it can be used when the number of attributes is larger than the number of observations (according to Cutler and Breiman 1994).

The simulation setup is the following (cf. the *simulation problem* by Hothorn, Leisch, Zeileis, and Hornik 2005): we consider the multivariate standard normal distribution as data generating process (spherical data). A data set is generated for each combination of $m = 5, 10, 25, 50$ attributes, $n = 100, 1000, 10000, 10000$ observations and 20 replications in each case. On each data set $k = 1, \dots, 10$ archetypes are fitted; stop criteria are 100 iterations or an improvement less than `sqrt(.Machine$double.eps)`. Each m, n, k configuration is fitted with randomly chosen initial archetypes. For each of the $m \times n \times k \times 20$ fits the computation time and the number of iterations until convergence are measured.

We present two main results of the simulation: (1) the computation time per iteration, i.e., focus on the efficiency of the convex least squares methods; (2) the computation time per fit, where the convergence behavior matters. The results are presented in Figure 9 and Figure 10, respectively. Both show four panels, one for each possible number of observations and each panel contains four lines, one for each possible number of attributes. The x-axis is the number of archetypes and the y-axis the measured value; notice that the scale of the y-axis varies between the panels.

Figure 9 shows the computation time per iteration in seconds, i.e., the computation time per fit divided by the number of iterations until convergence. In general the behavior is as expected: in case of fixed n and m (each individual line), increasing number of archetypes imply a linear increase of the computation time. Similarly, in case of fixed k and m (dots of same color and x-axis position, different panels), increasing observations imply approximately 10 times more computation time. And in the case of fixed k and n (dots of different colors, same x-axis position and panel), increasing attributes indicate a polynomial increase of the computation time. Noticeable is that in high dimensions ($n = 10000, 100000$ and $m = 50$) two, three and four archetypes need more computation time than the remaining numbers of archetypes.

Figure 10 shows the (median) computation time per fit in seconds. The figure shows that an increasing number of observations approximately imply a linear increase of the computation time. By contrast, for fixed n the computation time is approximately constant in k. Except from $n = 100000$ the peak at two archetypes is strongly distinct. Currently we have no reasonable explanation and more detailed simulations need to be done.

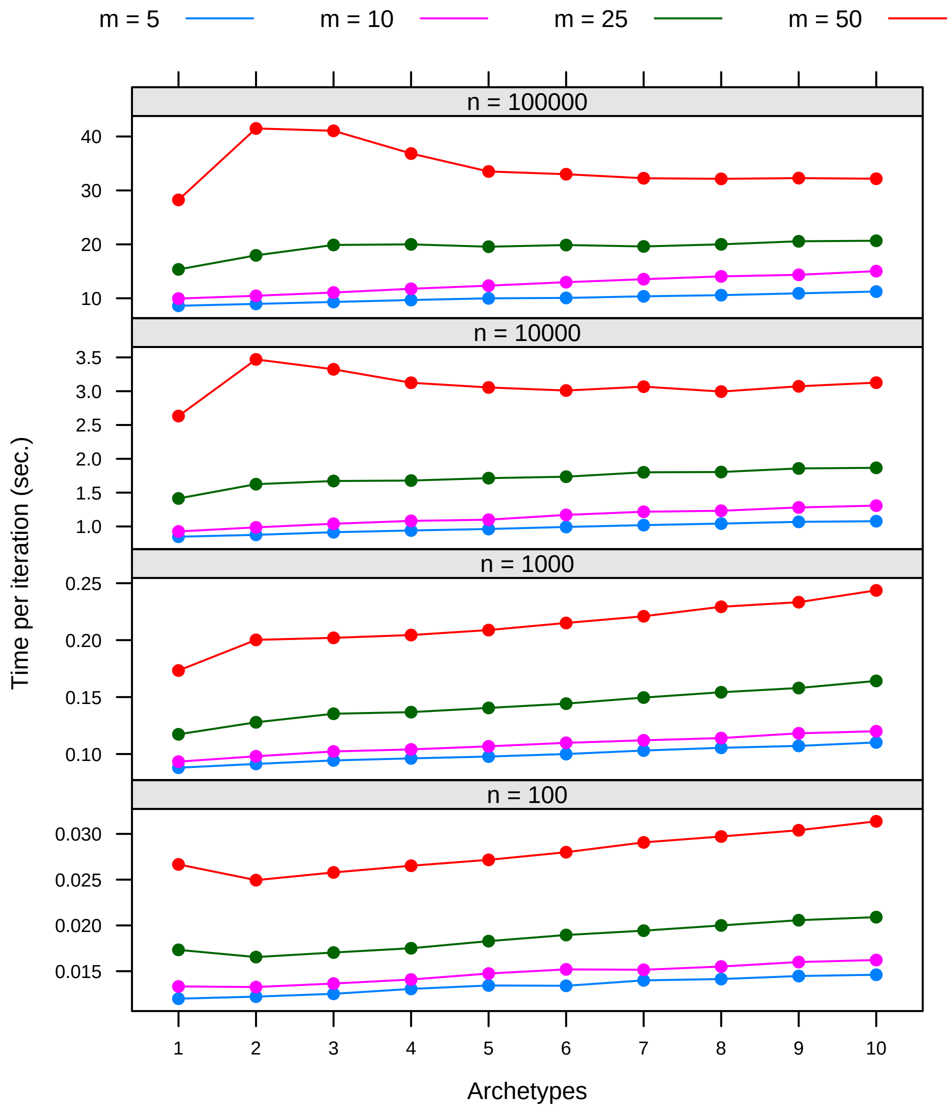

Figure 9: Computation time per iteration in seconds.

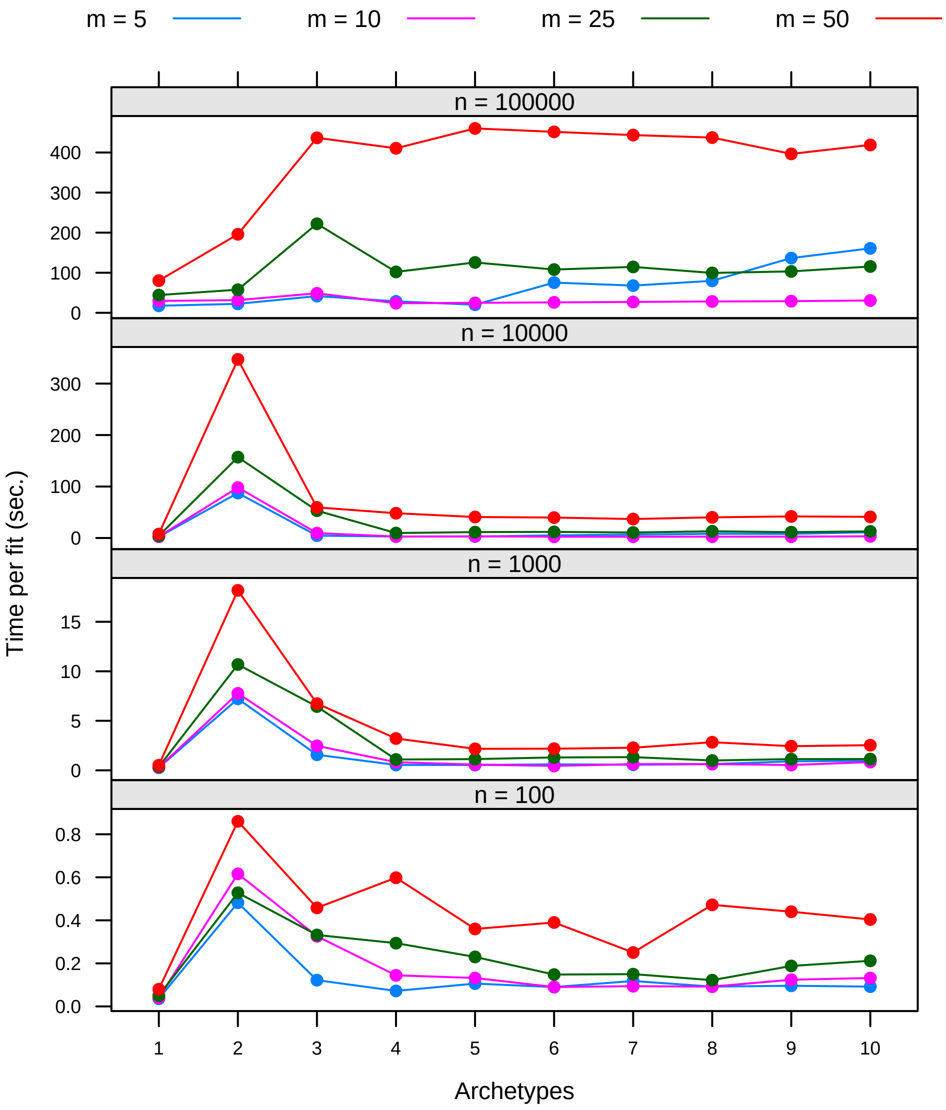

Figure 10: Computation time per fit in seconds.

5. Example: Skeletal archetypes

In this section we apply archetypal analysis on an interesting real world example: in Heinz, Peterson, Johnson, and Kerk (2003) the authors took body girth measurements and skeletal diameter measurements, as well as age, weight, height and gender on 247 men and 260 women in their twenties and early thirties, with a scattering of older man and woman, and all physically active. The full data are available within the package as `data("body")`, but we are only interested in a subset, the skeletal measurements and the height (all measured in centimeters),

```
R> data("skel")
R> skel2 <- subset(skel, select = -Gender)
```

The skeletal measurements consist of nine diameter measurements: biacromial (`Biac`), shoulder diameter; biiliac (`Biil`), pelvis diameter; bitrochanteric (`Bitro`) hip diameter; chest depth between spine and sternum at nipple level, mid-expiration (`ChestDp`); chest diameter at nipple level, mid-expiration (`ChestDiam`); elbow diameter, sum of two elbows (`ElbowDiam`); wrist diameter, sum of two wrists (`WristDiam`); knee diameter, sum of two knees (`KneeDiam`); ankle diameter, sum of two ankles (`AnkleDiam`). See the original publication for a full anatomical explanation of the skeletal measurements and the process of measuring. We use basic elements of *Human Modeling and Animation* to model the skeleton and create a schematic representation of an individual, e.g., `skeletonplot(skel2[1, ])` for observation number one. The function `jd()` (for “John Doe”) uses this plot and shows a generic individual with explanations of the measurements:

```
R> jd()
```

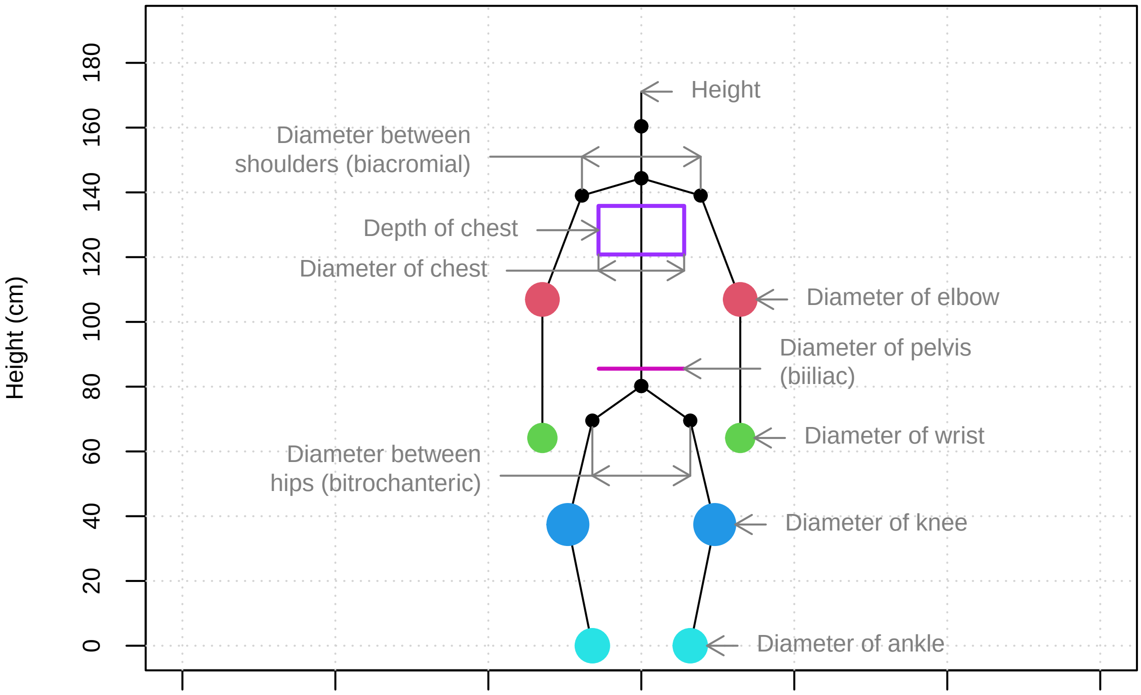

Figure 11: Generic skeleton with explanations of the measurements.

For visualizing the full data set, parallel coordinates with axes arranged according to the “natural order” are used,

```
R> pcplot(skel2)
```

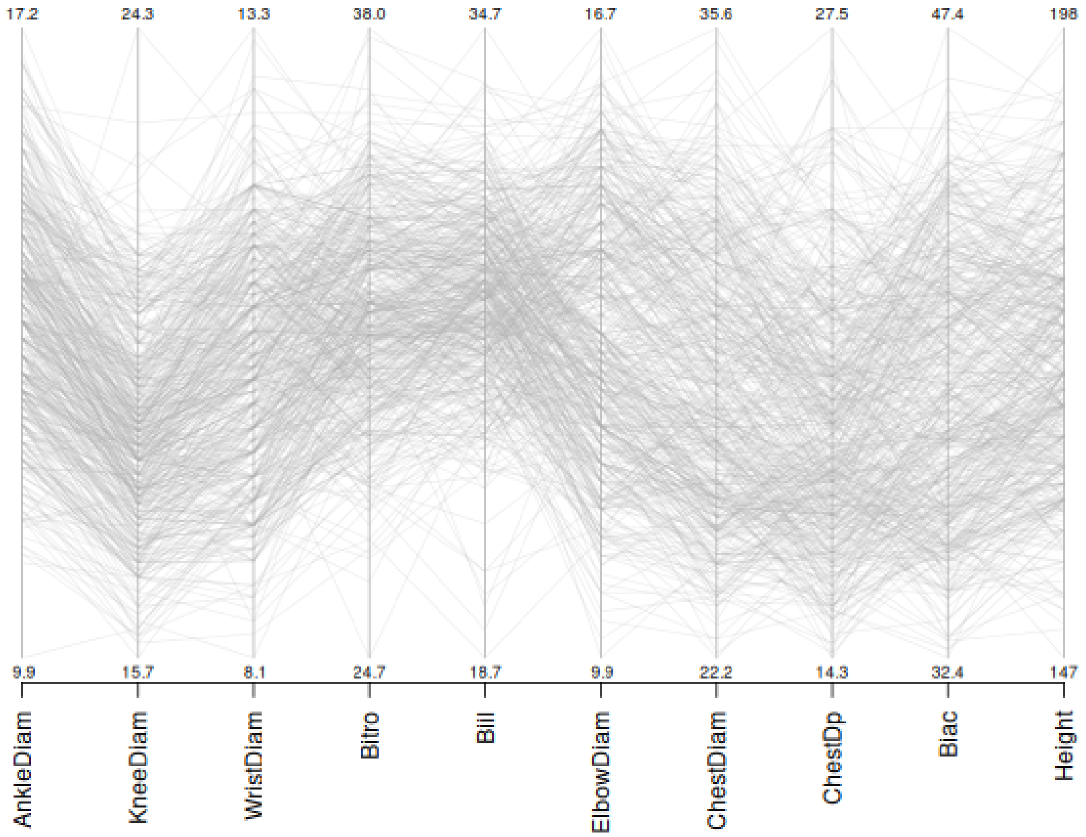

Figure 12: Parallel coordinates of the `skel2` data set.

At first view no patterns are visible and it is difficult to guess a meaningful number of archetypes. Therefore, we calculate the archetypes for $k = 1, \ldots, 15$ with three repetitions each time,

```
R> set.seed(1981)
R> as <- stepArchetypes(skel2, k = 1:15, verbose = FALSE, nrep = 3)
```

```
There were 12 warnings (use warnings() to see)
```

The warnings indicate ill-conditioned matrices as from $k = 11$, but, not as in the previous section, the residual sum of squares contains no `NA` values. The corresponding curve of the best model in each case is:

```
R> screeplot(as)
```

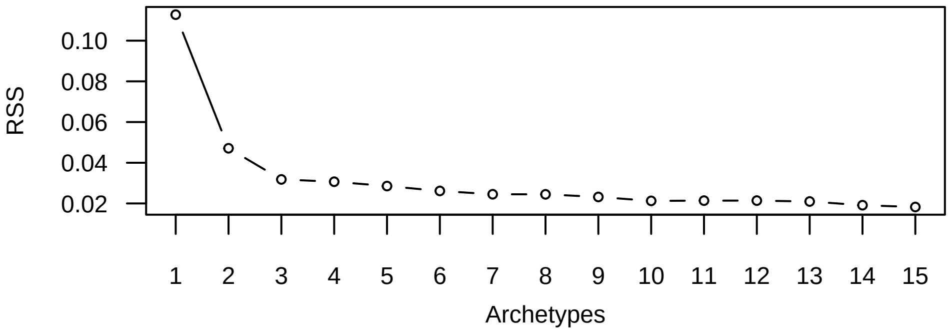

Figure 13: Screeplot of the residual sum of squares.

And according to the "elbow criterion" $k = 3$ or maybe $k = 7$ is the best number of archetypes. Corresponding to Occam's razor we proceed with three archetypes,

```
R> a3 <- bestModel(as[[3]])
```

The three archetypes are (transposed for better readability):

```
R> t(parameters(a3))
```

```
                 [,1]      [,2]       [,3]
AnkleDiam   13.184998  15.96133  12.022492
KneeDiam    18.590612  21.04518  16.435389
WristDiam    9.764051  12.23784   9.284341
Bitro       34.451169  34.45469  27.264701
Biil        31.222383  29.34122  22.522171
ElbowDiam   12.230591  15.74044  11.291717
ChestDiam   26.150773  32.77958  24.304209
ChestDp     18.392200  22.90516  15.952083
Biac        36.312352  43.88833  34.582899
Height     167.007496 186.68539 157.801659
```

Or as a barplot in relation to the original data:

```
R> barplot(a3, skel2, percentiles = TRUE)
```

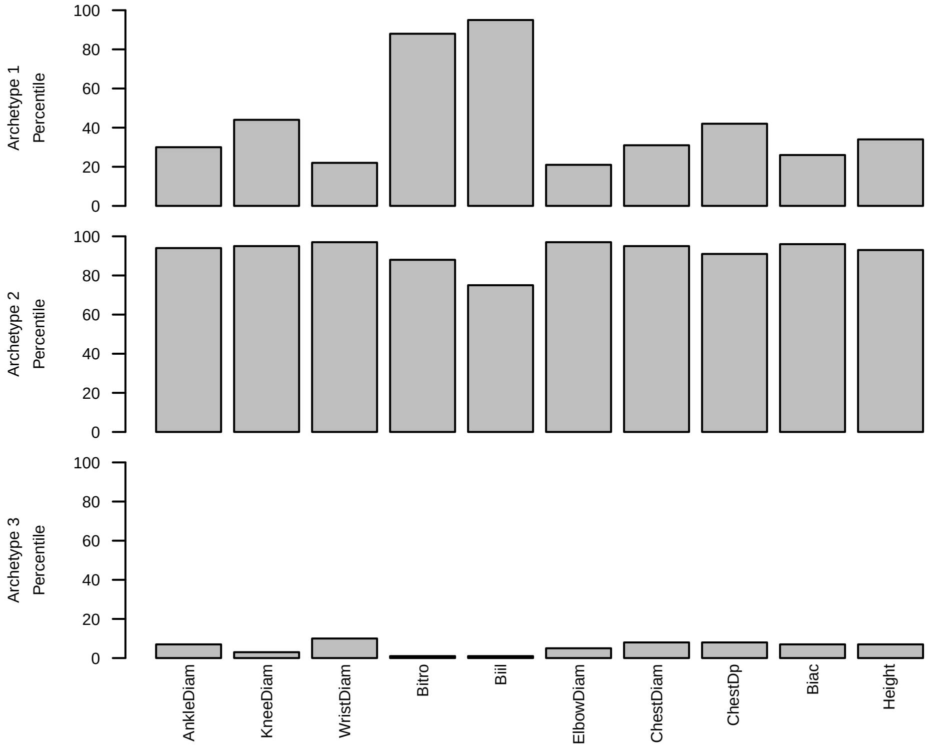

Figure 14: Barplot visualizing the three archetypes.

Archetype 2 (gray) represents individuals which are "huge" in all measurements; on the other hand, archetype 3 (lightgray) represents individuals which are "small". Archetype 1 (darkgray) represents individuals with average measures except the bitrochanteric and biiliac – the meaning of this is best visible when looking at the data with gender information (men are blue, women are green colored, with alpha transparency) and the archetypes (red),

```
R> pcplot(a3, skel2, data.col = as.numeric(skel$Gender))
```

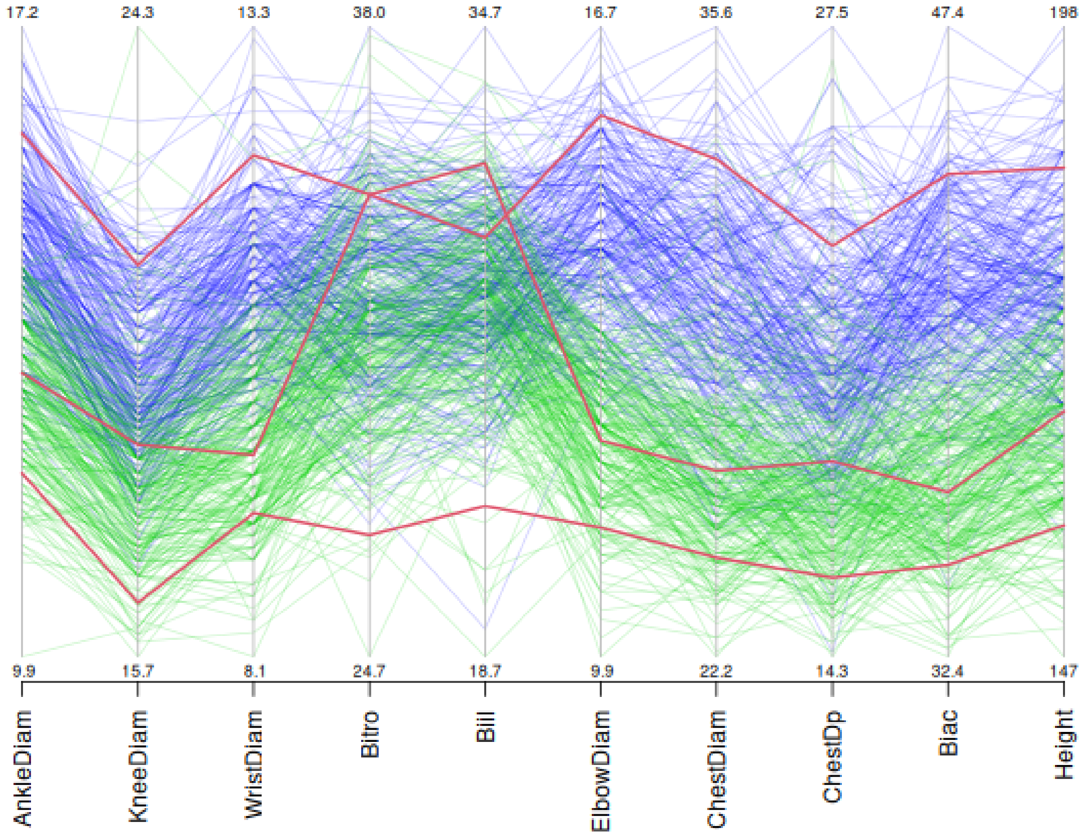

Figure 15: Parallel coordinates with colors related to the gender and the three archetypes.

Archetype 2 reflects the primary difference between men and women in body structure – the comparatively wider hip and pelvis of women. A verification of this interpretation can be done by looking at the coefficients α and see how much each archetype contributes to the approximation of each individual observation. For three archetypes, a ternary plot is a usable graphical representation (e.g., package **vcd** by Meyer, Zeileis, and Hornik 2008):

```
R> ternaryplot(coef(a3, 'alphas'), col = as.numeric(skel$Gender))
```

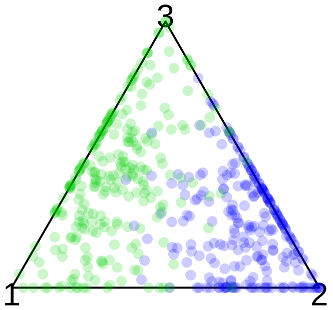

Figure 16: Ternary plot visualizing the α, colored according to the gender.

Clearly, males cluster close to archetype 2 and women mixes mainly the first and the third archetype. Therefore, males are primarly approximated by archetype 2, women by linear combination of archetypes 1 and 3. For more than three archetypes parallel coordinates with an axis for each archetype projecting the corresponding coefficients (in range $[0, 1]$) can be used to investigate the coefficients α.

Finally, the `skeletonplot()` visualizes the three skeleton archetypes:

```
R> skeletonplot(parameters(a3))
```

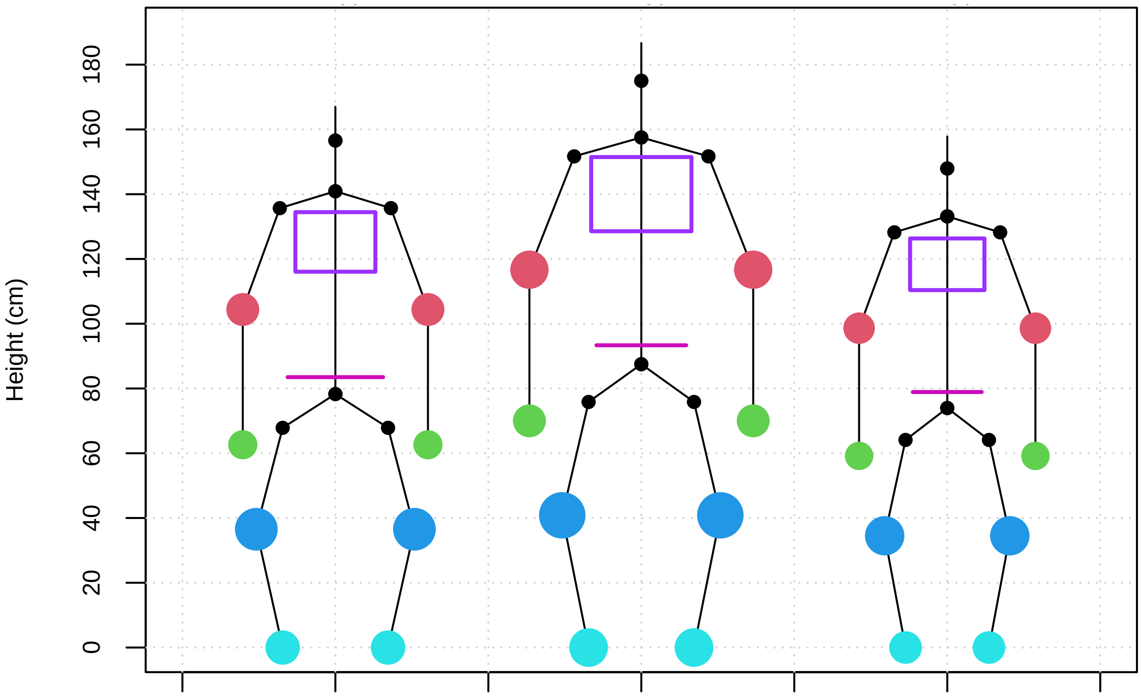

Figure 17: Skeleton plot visualizing the three archetypes.

The left skeleton visualizes archetype 2 with the wider hip and pelvis; the middle skeleton visualizes archetype 1 which is "huge", the right skeleton visualizes archetype 3 which is "small" in all measurements. Assume that we want to automatically fabricate blue worker overalls from given templates (e.g., paper patterns). Combinations of archetypes 1 and 3 gives overalls in different sizes which each have similar width at shoulder and waist. If we add archetype 2, we get overalls with a wider waist (compared to the shoulder).

6. Summary and outlook

In Section 2 we explained the different steps of the algorithms according to the original paper. However, for each problem a number of methods exist and the choice of the right method often depends on the concrete data set. In Section 3 we have already used the `archetypesFamily()` function to use different linear equations solver, but the function has to be extendend to allow abritary problem solving blocks for each step of the algorithm. Of primary interest is the usage of different optimization methods to solve the optimization problem.

Additionally, the two fundamentals defined in Section 2 can be generalized to allow for example arbitrary loss functions, arbitrary conditions on the coefficients α or arbitrary matrix norms. As the algorithm strategy is similar to the least squares formulation of the k-means algorithm, this leads to a general framework for this class of k-means-like algorithms (k-means, k-median, Fuzzy k-means, etc.; see, e.g., Steinly 2006).

Altogether, the short-term goal for the package **archetypes** was the implementation of a general framework with interchangeable blocks for the concrete problem solving mechanisms of each algorithm step. Now, further work is the design of a clean archetypes family object, especially with a view to our long-term goal, the generalization towards the class of k-means-like algorithms.

Computational details

```
R> sessionInfo()

R version 4.4.1 (2024-06-14)
Platform: x86_64-pc-linux-gnu
Running under: Ubuntu 24.04.1 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/libopenblasp-r0.3.26.so;  LAPACK versio

Random number generation:
 RNG:     Mersenne-Twister
 Normal:  Inversion
 Sample:  Rounding

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

time zone: Etc/UTC
tzcode source: system (glibc)

attached base packages:
[1] grid      stats4    stats     graphics  grDevices utils     datasets
[8] methods   base
```

```
other attached packages:
[1] vcd_1.4-13         archetypes_2.2-0.1 nnls_1.6           modeltools_0.2-23

loaded via a namespace (and not attached):
 [1] xfun_0.48        lattice_0.22-6   maketools_1.3.1  zoo_1.8-12      
 [5] knitr_1.48       buildtools_1.0.0 compiler_4.4.1   lmtest_0.9-40   
 [9] sys_3.4.3        tools_4.4.1      colorspace_2.1-1 MASS_7.3-61     
```

The newest released version of **archetypes** is always available from the Comprehensive R Archive Network at `http://CRAN.R-Project.org/package=archetypes`. The development version is hosted on R-Forge as project **Archetypal Analysis (archetypes)** and available from `http://r-forge.r-project.org/projects/archetypes`. The source code is documented using the Roxygen documentation system for R (Danenberg and Eugster 2008). An up-to-date version of this manuscript is contained in the package as a vignette.

Acknowledgments

The authors thank Bernard Pailthorpe, University of Queensland, for providing his MATLAB code for comparison with our code.

References

Cutler A, Breiman L (1994). "Archetypal Analysis." *Technometrics*, **36**(4), 338–347.

Danenberg P, Eugster MJA (2008). ***roxygen**: Literate Programming in R.* R package version 0.1, URL `http://CRAN.R-project.org/package=roxygen`.

Eugster MJA, Leisch F (2009). "From Spider-Man to Hero – Archetypal Analysis in R." *Journal of Statistical Software*, **30**(8), 1–23. URL `http://www.jstatsoft.org/v30/i08`.

Golub GH, Loan CFV (1996). *Matrix Computations.* Third edition. The Johns Hopkins University Press.

Heinz G, Peterson LJ, Johnson RW, Kerk CJ (2003). "Exploring Relationships in Body Dimensions." *Journal of Statistics Education*, **11**(2). URL `http://www.amstat.org/publications/jse/v11n2/datasets.heinz.html`.

Hothorn T, Leisch F, Zeileis A, Hornik K (2005). "The Design and Analysis of Benchmark Experiments." *Journal of Computational and Graphical Statistics*, **14**(3), 675–699.

Lawson CL, Hanson RJ (1974). *Solving Least Squares Problems.* Prentice Hall.

Li S, Wang P, Louviere J, Carson R (2003). "Archetypal Analysis: A New Way to Segment Markets Based on Extreme Individuals." In *A Celebration of Ehrenberg and Bass: Marketing Knowledge, Discoveries and Contribution. Proceedings of the ANZMAC 2003 Conference, December 1-3, 2003*, pp. 1674–1679.

Luenberger DG (1984). *Linear and Nonlinear Programming.* Second edition. Addison-Wesley.

Merriam-Webster Online Dictionary (2008). "Archetype." In *Merriam-Webster Online Dictionary*. Retrieved October 6, 2008, from http://www.merriam-webster.com/dictionary/archetype.

Meyer D, Zeileis A, Hornik K (2008). ***vcd**: Visualizing Categorical Data*. R package version 1.0-9, URL http://CRAN.R-project.org/package=vcd.

Mullen KM, van Stokkum IHM (2007). ***nnls**: The Lawson-Hanson Algorithm for Non-Negative Least Squares (NNLS)*. R package version 1.1, URL http://CRAN.R-project.org/package=nnls.

Pailthorpe B (2008). "MATLAB Source Code for Archetypal Analysis." MATLAB source code and personal communication.

Porzio GC, Ragozini G, Vistocco D (2008). "On the Use of Archetypes as Benchmarks." *Applied Stochastic Models in Business and Industry*, **24**(5), 419–437.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Steinly D (2006). "K-means Clustering: A Half-Century Synthesis." *British Journal of Mathematical and Statistical Psychology*, pp. 1–34.

Affiliation:

Manuel J. A. Eugster
Department of Statistics
Ludwig-Maximilians-Universität München
80539 Munich, Germany
E-mail: Manuel.Eugster@stat.uni-muenchen.de
URL: http://www.statistik.lmu.de/~eugster/

Friedrich Leisch
Department of Statistics
Ludwig-Maximilians-Universität München
80539 Munich, Germany
E-mail: Friedrich.Leisch@R-project.org
URL: http://www.statistik.lmu.de/~leisch/