Introduction to identifying functional residues in an alignment using BGAFUN

Iain Wallace

April 22, 2010

Proteins that evolve from a common ancestor can change functionality over time, and it is important to be able identify residues that cause this change. There are numerous different methods that can be used to solve this problem.

This package contains the functions to use supervised multivariate statistical method, Between Group Analysis (BGA) [Culhane *et al.*, 2005] to identify these residues from families of proteins with different substrate specifities using multiple sequence alignments.

The main steps involved are

1. Reading in the alignment

2. Defining groups

3. Convert alignment into a matrix

4. Run the Between Groups Analysis

5. Plot the results

6. Identify important positions in the Alignment

1 Reading in the alignment

The sequences are read in using the seqinR [Charif & Lobry, 2006] commands

```
> library(bgafun)
> LDH <- read.alignment(file = system.file("sequences/LDH-MDH-PF00056.fasta",
+     package = "bgafun"), format = "fasta")
> class(LDH)

[1] "alignment"
```

2   Defining groups

Next, a factor assigning each sequence to a group has to be defined. One
method of doing this is shown in the code below. The alignment object is
first converted into a matrix, in which the rownames are sequence names

```
> library(bgafun)
> data(LDH)
> LDH.amino = convert_aln_amino(LDH)
> LDH.groups = rownames(LDH.amino)
> LDH.groups[grep("LDH", LDH.groups)] = "LDH"
> LDH.groups[grep("MDH", LDH.groups)] = "MDH"
> LDH.groups = as.factor(LDH.groups)
> LDH.groups

 [1] MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH
[20] MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH MDH LDH
[39] LDH LDH LDH LDH LDH LDH LDH
Levels: LDH MDH
```

3   Convert alignment into a matrix

The alignment has to be converted into a matrix that is suitable for Between
Group Analysis. There are two different encoding schemes available in this
package

1. Binary

2. Amino Acid Properties

3.1 Binary Encoding Scheme

Each column in the sequence alignment is represented by a binary vector of length 20 which represents the presence or absence of a particular amino acid at this position. This results in a data matrix of dimension NxLx20 where N is the number of sequences and L is the length of the alignment. Gaps are removed from the matrix using the remove_gaps_groups function. This removes columns that contain more than a certain percentage, gap_fraction, of gaps in a column in the alignment in any of the defined groups.

```
> library(bgafun)
> data(LDH)
> data(LDH.groups)
> LDH.amino = convert_aln_amino(LDH)
> dim(LDH.amino)

[1]    45 3600

> LDH.amino.gapless = remove_gaps_groups(LDH.amino, LDH.groups)
> dim(LDH.amino.gapless)

[1]    45 2820
```

We found that it is important to add pseudo count as otherwise the absence of amino acids in a column dominates the analysis. There are two methods of adding pseudocounts:

1. Add 1 to all elements in the matrix

   ```
   > library(bgafun)
   > data(LDH.amino.gapless)
   > LDH.pseudo = LDH.amino.gapless + 1
   > dim(LDH.pseudo)

   [1]    45 2820
   ```

2. Use the add_pseudo_counts function

   ```
   > library(bgafun)
   > data(LDH.amino.gapless)
   > LDH.pseudo = add_pseudo_counts(LDH.amino.gapless, LDH.groups)
   > dim(LDH.pseudo)

   [1]    45 2820
   ```

3.2   Amino Acid Properties Encoding Scheme

Each residue in the alignment is represented by a vector of five continuous variables as given by Atchley et al [Atchley *et al.*, 2005]. They applied a multivariate statistic approach to reduce the information in 494 amino acid attributes into a set of five factors for each amino acid.

1. Factor A is termed the polarity index. It correlates well with a large variety of descriptors including the number of hydrogen bond donors, polarity versus nonpolarity, and hydrophobicity versus hydrophilicity.

2. Factor B is a secondary structure index. It represents the propensity of an amino acid to be in a particular type of secondary structure, such as a coil, turn or bend versus the frequency of it in an a-helix.

3. Factor C is correlated with molecular size, volume and molecular weight.

4. Factor D reflects the number of codons coding for an amino acid and amino acid composition. These attributes are related to various physical properties including refractivity and heat capacity.

5. Factor E is related to the electrostatic charge.

Columns in the alignment containing more than 80% gaps are removed. When using the AAP encoding, the remaining gaps were replaced with the average value for the subfamily in that column using the function average_cols_aap

```
> library(bgafun)
> data(LDH)
> data(LDH.groups)
> LDH.aap = convert_aln_AAP(LDH)
> dim(LDH.aap)

[1]  45 900

> LDH.aap.ave = average_cols_aap(LDH.aap, LDH.groups)
> dim(LDH.aap.ave)

[1]  45 690
```

4 Run the Between Groups Analysis

Two different ordination techniques can be used, either Principal Components Analysis or Correspondance Analysis. PCA is the most suitable ordination technique for the AAP encoding as it can be used to ordinate data that contains continuous variables. CA, on the other hand, can only be used to ordinate data that contains positive integer values, such as the binary representation of a sequence alignment. To run the analysis with PCA the run_between_pca function is used, It requires an binary amino acid matrix which is used to calculate sequence weights, the matrix to be analysised and the group definition

```
> library(bgafun)
> data(LDH)
> data(LDH.groups)
> data(LDH.amino.gapless)
> data(LDH.aap.ave)
> LDH.aap.ave.bga = run_between_pca(LDH.amino.gapless, LDH.aap.ave,
+     LDH.groups)
> class(LDH.aap.ave.bga)

[1] "pca" "bga"
```

To run the analysis with CA, without weights and with very simple pseudocounts

```
> library(bgafun)
> data(LDH.groups)
> data(LDH.amino.gapless)
> LDH.binary.bga = bga(t(LDH.amino.gapless + 1), LDH.groups)
```

5 Plot the results

The results are plotted using the MADE4 functions. When the labels overlab, like below, the easiest method to clean them is to use Photoshop.

```
> plot(LDH.aap.ave.bga)
```

```
 [1] 191 192 194 311 376 378 380 384 401 483 485 523 586 587 588 589 590 591 637
[20] 640
```
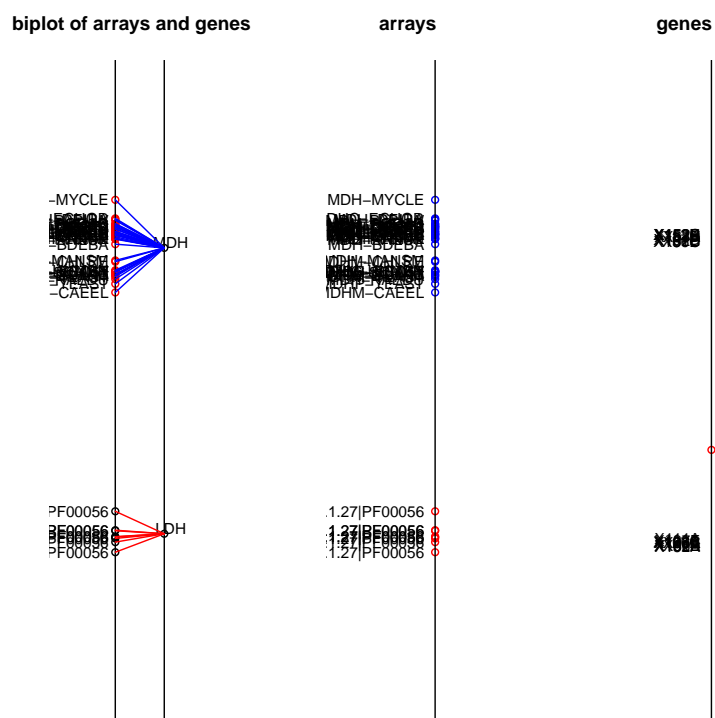

Figure 1: PCA plot of the LDH example

6 Identify important positions in the Alignment

Important positions for two groups can be identified by finding the most extreme positions on the axis using the top_residues_2_groups function

```
> top_res = top_residues_2_groups(LDH.binary.bga)
> names(top_res) = sub("X", "", names(top_res))
```

Important positions for three groups can be identified by finding the most extreme positions that are plotted same direction as the sequences on the graph

Now, to look at the amino acid content in the alignment that the analsyis has identified one can use the following code. It creates a profile string for each of the groups. This shows how many of each type of amino acids are present in each group at every position. The

```
> LDH.profiles = create_profile_strings(LDH.amino, LDH.groups)
> LDH.profiles[, colnames(LDH.profiles) %in% names(top_res)]
```

	4A	8A	9A	9G	12I	14M	16L	21L	45I	61E	61V	71F	89D	96S	102T	106R	106P	107R
LDH	0	1	0	8	0	8	0	8	0	8	0	8	8	8	8	8	0	0
MDH	28	33	30	2	34	0	34	0	24	0	32	0	0	0	1	0	31	35

	111E	111M	114L	115D	119R	127Q	128G	130A	136A	136S	152N	152D	153T	154N	155T
LDH	8	0	8	1	8	0	0	0	0	8	0	8	0	0	8
MDH	2	32	0	34	1	23	23	27	23	1	35	0	25	23	1

	157I	157V	159W	162A	162S	173P	179G	180S
LDH	0	8	8	0	8	8	8	8
MDH	33	1	0	23	0	0	0	1

7 Bib

References

[Atchley *et al.*, 2005] Atchley, W. R., Zhao, J., Fernandes, A. D. & Druke, T. (2005). Solving the protein sequence metric problem. *Proc Natl Acad Sci U S A*, 102(18), 6395–400.

[Charif & Lobry, 2006] Charif, D. & Lobry, J.R. (2006). Seqinr 1.0-2: a contributed package to the r project for statistical computing devoted to biological sequences retrieval and analysis. In *Structural approaches to sequence evolution: Molecules, networks, populations* (Bastolla, U., Porto, M., Roman, H.E. & Vendruscolo, M., Hrsg.). Springer.

[Culhane *et al.*, 2005] Culhane, A. C., Thioulouse, J., Perriere, G. & Higgins, D. G. (2005). Made4: an r package for multivariate analysis of gene expression data. *Bioinformatics*, 21(11), 2789–90.