VELIMIR GAYEVSKIY

OBSTRUCT MANUAL — VERSION 1.0

HTTP://GODDARDLAB.AUCKLAND.AC.NZ/OBSTRUCT

http://goddardlab.auckland.ac.nz/obstruct

Manual last updated: December 2013

Contents

Introduction

Introduction and Purpose

ObStruct is a tool for the objective analysis of population structure within outputs derived from the population genetics software packages InStruct [Gao et al., 2007], structure [Pritchard et al., 2000] and baps [Corander et al., 2004, 2008]. These software packages use Bayesian methods to test for population structure in genetic data from a number of individuals in a number of sampled populations. The Bayesian approach allows the probabilistic assignment of individuals to any number of computationally inferred populations within each of which there is free gene flow. structure attempts to do this by clustering individuals within each inferred population so as to maximize Hardy-Weinberg equilibrium; InStruct calculates the expected genotype frequencies given the rates of inbreeding within each inferred population; and baps relies on the differences in allele frequencies to partition individuals. The main benefit of this is that individuals no longer have to be *a priori* assigned to populations, the genetic data is presented to the software and if there are groups of individuals which are closely related to one another, the methods will tend to cluster them together within one of the inferred populations. We can thus define these proportions of clustering to each inferred population as **ancestry profiles** where each individual has a certain percentage of 'fit' to each inferred population.

Ancestry profiles produced by structure and InStruct are typically visualized in the distruct [Rosenberg, 2003] software while baps produces its own similar plots which simply assigns a unique color to each inferred population and creates a column graph where each individual is a column made up of the proportions of ancestry to each inferred population. Individuals from each sampled population are then grouped together and separated from the other populations. If there is population structure within the sampled populations, individuals within them will tend to have high proportions of ancestry to unique inferred populations, i.e. consist of large blocks of unique colors. This technique is an excellent tool for visualizing the data and making conclusions, but it lacks the crucial step of objectivity. For example, how do you judge the differing amounts of structure within each of the sampled populations given different population sizes and rates of admixture? How do you compare different data sets to determine their relative levels of structure? How do you test whether sampled populations are significantly different from one another or whether they come from a single population?

ObStruct focuses on the objective analysis of these ancestry profiles to answer the questions posed above and many others. At the core of ObStruct is the R^2 statistic which determines the variability within sampled (predefined by the user) populations and the variability between them and goes on to produce a single number (R^2) which indicates what proportion of the variability in the data is explained by the predefined populations. ObStruct further tests statistical significance by permuting the assignment of

ancestry profiles to individuals and performs additional analyses of population structure to further delve into which populations within the data are contributing to the overall level of structure in the data.

ObStruct *Software Overview*

This section will describe the process that ObStruct takes to generate its outputs in a step-by-step manner.

After ObStruct is run from the command line with the parameters specified in the ObStruct Input section below, it proceeds to load the input file[1]. If InStruct, structure or baps outputs are specified as the input files, ObStruct will parse them for the the ancestry profiles. This parsing includes compiling a list of all individuals within the data, determining their sampled population classifications then associating their ancestry profiles to them. This information is then exported to a csv (comma-separated values) file. The reason for parsing into a csv file is that it allows a single standard format for ObStruct to work with which is easy to edit for users wishing to change predefined population[2] membership, add/remove individuals or create simulated data. In these cases, the edited or created csv files can be loaded directly into ObStruct, bypassing the parsing step altogether.

[1] Checks are performed to determine whether the parameters are correct given the populations genetics software used and whether input files exist.

[2] They are called predefined populations due to the ability of the user to classify individuals within their input data into any population they wish.

ObStruct computes the R^2 values for different combinations of individuals. To assess the overall structure it computes the R^2 value for the whole data set. Next, it computes the R^2 values for all pairs of predefined populations, the R^2 values for the remaining individuals after removal of single predefined populations, and the R^2 value for the data after removal of single inferred populations.

The significance of the value observed for the entire dataset and pairwise predefined populations comparisons is assessed by a permutation approach. For the whole data set, ObStruct permutes the population membership of individuals 10,000 times and computes the R^2 value for each permutation. The approximate p-value is then the proportion of permuted R^2 values exceeding the original R^2 value. For the pairwise assessment, 1,000 permutations are executed for each test, and a Bonferroni correction is applied to the p-value to account for multiple pairwise testing.

ObStruct does not provide p-values for calculations of R^2 when each predefined/inferred population is removed in turn because significance for these cases is reflected in the pairwise testing. For example, if only a single predefined population is driving the structure within the data, all pairwise comparisons not involving this structured population will show very little structure with non-significant p-values.

Finally, ObStruct generates an R script. This script can simply be opened in the statistical software R and run to produce three figures visualizing the structure within the data using Canonical Discriminant Analysis (CDA).

ObStruct *Input*

ObStruct is a Perl script which must be run from the command line.

In Windows you will need to open a command prompt which is capable of running Perl. If you are using Strawberry Perl[3], this can be found in the Start Menu in the Strawberry Perl folder labelled 'Perl (command line)'.

[3] Strawberry Perl is the recommended Perl interpreter for Windows, download it here.

Both Apple OS X and Linux come pre-loaded with Perl so just open Terminal and you are ready to go.

To run ObStruct, you simply need to enter the string specified in each of the "Using a <software> Output" sections below but replace each of the parameters (the ones with dashes) with a value for that parameter from the associated parameter description tables.

ObStruct comes with four sample input files: the InStruct output `instruct-yeastdata.txt`; the structure output `structure-humandata-k-is-6`; the baps output `baps-humandata.txt`; and, a simulation output `sim.csv`. Examples of input strings for each of these input files for both Apple OS X/Linux and Windows can be found in the "Using a <software> Output" sections below. Due to the way the two human datasets were created and run through structure and baps, you should only compare the overall R2 value and R2 patterns seen in the ObStruct output. A more detailed explanation of this can be found in the Extra Information section at the end of this document.

If you have your output file ready to run through ObStruct or wish to use one of the example files then jump to the correct section based on the software you used to generate it:

Alternatively, read on for more information about these output files and how you can create your own in the Input Files from InStruct, Structure & baps and Creating a CSV File for Input sections.

Input Files from INSTRUCT, STRUCTURE *&* BAPS

STRUCTURE

IF YOU ARE using a STRUCTURE output file as your input into OBSTRUCT, you must navigate to the project output directory of your STRUCTURE run and find the Results folder. Within this folder you will see a number of files the first of which ends with `_run_1_f` and the rest iterate the 1 for each additional file. These files have the following header:

```
-------------------------------------
STRUCTURE by Pritchard, Stephens and Donnelly (2000)
and Falush, Stephens and Pritchard (2003)
Code by Pritchard, Falush and Hubisz
Version 2.3.2.1 (Oct 2009)
-------------------------------------
```

Find the file which contains the number of inferred populations you are interested in and copy it to the working directory. This file can be specified as the input file into OBSTRUCT.

INSTRUCT

IF YOU ARE using a INSTRUCT output file as your input into OBSTRUCT, copy the INSTRUCT output file to the working directory. The INSTRUCT output file has the following header:

```
=============================================
InStruct by Gao, Williamson and Bustamante (2007)
Code by Hong Gao
Version 1.0 (May. 2007)
=============================================
```

This file can be specified as the input file into OBSTRUCT.

BAPS

IF YOU ARE using an BAPS output file as your input into OBSTRUCT, copy the BAPS output file to the working directory. BAPS runs its analyses in a two-step manner where first a "Population mixture analysis" is performed followed by a secondary "Population admixture" analysis. Please perform both analyses and make sure they are

output one after another to the same output file from BAPS. The BAPS output file has a header similar to this:

> RESULTS OF INDIVIDUAL LEVEL MIXTURE ANALYSIS:
>
> Data file: microsats.txt
>
> Number of clustered individuals: 1484
>
> Number of groups in optimal partition: 6
>
> Log(marginal likelihood) of optimal partition: -89760.5629
>
> Best Partition:

This file can be specified as the main input file into OBSTRUCT. Since BAPS does not specify the population membership of each individual in its main output file, you will need to also specify the population index file you used to run BAPS. More information on how to do this can be found in the Using a BAPS Output section below.

Population Classifications

INSTRUCT, STRUCTURE & BAPS do not print population names as words in their outputs. Since OBSTRUCT can only work with what it is given, parsed output files from INSTRUCT, STRUCTURE & BAPS will retain the numbers assigned to the predefined populations by these software. This leaves the user of OBSTRUCT with two options, either go back to the outputs from INSTRUCT, STRUCTURE or BAPS and determine which predefined population matches to which number in the OBSTRUCT output, or rename the predefined populations in the parsed csv file that OBSTRUCT produces and input the file as `-csv` for the software parameter[4]. More information on how to do this is in the section below.

[4] You can use letters, numbers, dashes, underscores and periods for population names if renaming within the csv file.

Creating a CSV File for Input

YOU WILL RECALL from the OBSTRUCT Software Overview section that OBSTRUCT parses the input files from STRUCTURE, INSTRUCT or BAPS to generate a standard input csv file which it then operates on. A csv file contains rows of comma-separated values where each row contains an individual's information separated by a comma and there are as many rows as there are individuals.

The csv file produced by OBSTRUCT is as follows: each row of the csv file contains one individual; the first column is the individual's

name, the second is its population name/number[5] and the remaining columns are the ancestry values of the individual to Inferred population 1, Inferred population 2, etc.

This is an example of this format for real data:

[5] Population names can be unique numbers (as generated by INSTRUCT, STRUCTURE or BAPS) and/or letters that may contain numbers, dashes, underscores and periods.

```
HB_C_BROOKLANDS_1,0,0.030,0.007,0.099,0.015,0.026,0.014,0.028,0.009,0.756,0.017
HB_C_BROOKLANDS_10,0,0.039,0.257,0.006,0.308,0.055,0.009,0.029,0.123,0.075,0.099
HB_C_BROOKLANDS_11,0,0.011,0.008,0.017,0.008,0.912,0.008,0.013,0.008,0.008,0.008
WA_C_CODDINGTON_1,1,0.947,0.005,0.005,0.005,0.005,0.006,0.005,0.005,0.011,0.005
WA_C_CODDINGTON_10,1,0.005,0.004,0.960,0.004,0.005,0.004,0.004,0.004,0.005,0.004
WA_C_CODDINGTON_11,1,0.004,0.004,0.958,0.005,0.005,0.004,0.005,0.004,0.005,0.005
```

In the above example we have 6 individuals and their ancestry profiles to 10 inferred populations. The first column[6] contains the individual names, the second column contains the population assignment[7] and the remaining columns contain ancestry values to each of the inferred populations in these data (10 in total) and should sum to 1.

[6] The first piece of information in each row followed by a comma

[7] In this case the first three individuals belong to predefined population 0 while the last three belong to predefined population 1.

For studies involving simulation or other computational creation of ancestry profiles, simply print them in the format specified above and OBSTRUCT will read them and process them when csv is specified as the software input.

Note that you may experience problems with editing or creating these csv files in Microsoft Excel. Excel occasionally adds non-standard characters to csv files which may not be able to be read by OBSTRUCT. If you get errors after editing a csv file in Excel, try using a plain text editor with regular expressions instead.

Using a STRUCTURE *Output*

To RUN OBSTRUCT WITH an output file produced by STRUCTURE, you will need to call OBSTRUCT from the command line with several parameters[8]. The following line is the generic input you will need to enter into the command line with each of the parameters in this line explained in the table below it:

[8] Parameters are otherwise known as arguments or options. They specify a vital piece of information for the software to run.

```
perl /Path/To/ObStruct.pl -structure -working_directory -input_file -output_file
```

Parameter	Description
`/Path/To/ObStruct.pl`	The location of the ObStruct Perl script. To make sure the formatting of the directory path is correct, drag in the Perl script file into the terminal/command line input. This will print the path in the correct format for your operating system.
`-structure`	This simply specifies to OBSTRUCT that you wish it to analyze a STRUCTURE output file.
`-working_directory`	OBSTRUCT only works within a single directory. The directory you specify here will house your input files and will be where output files are written. Note that it **should end with a slash**. As before, you can simply drag a folder into your command line to have the full path come up in the correct format for your operating system. If there are any spaces in this directory path, you will need to escape them with quotation marks in Windows (e.g. -C:\Working" "Directory\) and a backslash in Apple OS X/Linux (e.g. -/Users/Smith/Working\ Directory/).
`-input_file`	The filename of the input file. This input file must be in the working directory. Note that you **do not need to specify a path** for this file as you have already specified the working directory.
`-output_file`	The filename for the outputs of OBSTRUCT. These will be saved in the working directory as two files ending in .csv and .r with the filename specified in this parameter. Once again, you **do not need to specify a path** for this parameter.

Table 1: Explanation of the parameters to be used when running OBSTRUCT with a STRUCTURE output file.

Examples of Input Strings

Apple OS X & Linux

```
perl /Users/Smith/Desktop/ObStruct.pl -structure -/Users/Smith/Desktop/ObStruct\ Data/
-structure-humandata-k-is-6 -structure-humandata-k-is-6-output
```

Windows

```
perl C:\Users\Smith\Desktop\ObStruct.pl -structure -C:\Users\Smith\Desktop\ObStruct" "Data\
-structure-humandata-k-is-6 -structure-humandata-k-is-6-output
```

Using a InStruct *Output*

To run ObStruct with an output file produced by InStruct, you will need to call ObStruct from the command line with several parameters[9]. The following line is the generic input you will need to enter into the command line with each of the parameters in this line explained in the table below it:

[9] Parameters are otherwise known as arguments or options. They specify a vital piece of information for the software to run.

```
perl /Path/To/ObStruct.pl -instruct -working_directory -input_file -output_file -K
```

Parameter	Description
`/Path/To/ObStruct.pl`	The location of the ObStruct Perl script. To make sure the formatting of the directory path is correct, drag in the Perl script file into the terminal/command line input. This will print the path in the correct format for your operating system.
`-instruct`	This simply specifies to ObStruct that you wish it to analyze a InStruct output file.
`-working_directory`	ObStruct only works within a single directory. The directory you specify here will house your input files and will be where output files are written. Note that it **should end with a slash**. As before, you can simply drag a folder into your command line to have the full path come up in the correct format for your operating system. If there are any spaces in this directory path, you will need to escape them with quotation marks in Windows (e.g. -C:\Working" "Directory\) and a backslash in Apple OS X/Linux (e.g. -/Users/Smith/Working\ Directory/).
`-input_file`	The filename of the input file. This input file must be in the working directory. Note that you **do not need to specify a path** for this file as you have already specified the working directory.
`-output_file`	The filename for the outputs of ObStruct. These will be saved in the working directory as two files ending in .csv and .r with the filename specified in this parameter. Once again, you **do not need to specify a path** for this parameter.
`-K`	This parameter can either be: -[number] or -optimal. InStruct prints its output into a single large text file which includes a range of K values and a number of chains. You have the option to use a specific value of K or to automatically use the optimal value of K as determined by InStruct at the end of its output. Note that **K needs to be greater than 1**.

Table 2: Explanation of the parameters to be used when running ObStruct with a InStruct output file.

Examples of Input Strings

Apple OS X & Linux

```
perl /Users/Smith/Desktop/ObStruct.pl -instruct -/Users/Smith/Desktop/ObStruct\ Data/
-instruct-yeastdata.txt -instruct-yeastdata-output -optimal

perl /Users/Smith/Desktop/ObStruct.pl -instruct -/Users/Smith/Desktop/ObStruct\ Data/
-instruct-yeastdata.txt -instruct-yeastdata-output -5
```

Windows

```
perl C:\Users\Smith\Desktop\ObStruct.pl -instruct -C:\Users\Smith\Desktop\ObStruct" "Data\
-instruct-yeastdata.txt -instruct-yeastdata-output -optimal

perl C:\Users\Smith\Desktop\ObStruct.pl -instruct -C:\Users\Smith\Desktop\ObStruct" "Data\
-instruct-yeastdata.txt -instruct-yeastdata-output -5
```

Using a BAPS *Output*

To run ObStruct with an output file produced by baps, you will need to call ObStruct from the command line with several parameters[10]. The following line is the generic input you will need to enter into the command line with each of the parameters in this line explained in the table below it:

[10] Parameters are otherwise known as arguments or options. They specify a vital piece of information for the software to run.

```
perl /Path/To/ObStruct.pl -baps -working_directory -input_file -output_file -population_index_file
```

Parameter	Description
`/Path/To/ObStruct.pl`	The location of the ObStruct Perl script. To make sure the formatting of the directory path is correct, drag in the Perl script file into the terminal/command line input. This will print the path in the correct format for your operating system.
`-baps`	This simply specifies to ObStruct that you wish it to analyze a baps output file.
`-working_directory`	ObStruct only works within a single directory. The directory you specify here will house your input files and will be where output files are written. Note that it **should end with a slash**. As before, you can simply drag a folder into your command line to have the full path come up in the correct format for your operating system. If there are any spaces in this directory path, you will need to escape them with quotation marks in Windows (e.g. -C:\Working" "Directory\) and a backslash in Apple OS X/Linux (e.g. -/Users/Smith/Working\ Directory/).
`-input_file`	The filename of the input file. This input file must be in the working directory. Note that you **do not need to specify a path** for this file as you have already specified the working directory.
`-output_file`	The filename for the outputs of ObStruct. These will be saved in the working directory as two files ending in .csv and .r with the filename specified in this parameter. Once again, you **do not need to specify a path** for this parameter.
`-population_index_file`	The baps output file does not specify the predefined population memberships of each individual so it needs to be specified separately. Simply take the population index file you used to run baps and place it in the working directory then specify the filename here.

Table 3: Explanation of the parameters to be used when running ObStruct with a baps output file.

Examples of Input Strings

Apple OS X & Linux

```
perl /Users/Smith/Desktop/ObStruct.pl -baps -/Users/Smith/Desktop/ObStruct\ Data/
-baps-humandata.txt -baps-humandata-output -baps-population-index.txt
```

Windows

```
perl C:\Users\Smith\Desktop\ObStruct.pl -baps -C:\Users\Smith\Desktop\ObStruct" "Data\
-baps-humandata.txt -baps-humandata-output -baps-population-index.txt
```

Important Note Regarding p-values

BAPS PRODUCES a *p*-value for each ancestry profile indicating whether admixture events within the profiles are significant or not. At present, OBSTRUCT ignores these *p*-values and parses the ancestry profiles as they are output by BAPS. A future version of OBSTRUCT may include a flag for disregarding admixture events which BAPS determines non-significant.

Using a CSV File

To RUN OBSTRUCT WITH a CSV file previously produced by OBSTRUCT or one that you made yourself, you will need to call OBSTRUCT from the command line with several parameters[11]. The following line is the generic input you will need to enter into the command line with each of the parameters in this line explained in the table below it:

[11] Parameters are otherwise known as arguments or options. They specify a vital piece of information for the software to run.

```
perl /Path/To/ObStruct.pl -csv -working_directory -input_file -output_file
```

Parameter	Description
`/Path/To/ObStruct.pl`	The location of the ObStruct Perl script. To make sure the formatting of the directory path is correct, drag in the Perl script file into the terminal/command line input. This will print the path in the correct format for your operating system.
`-csv`	This simply specifies to OBSTRUCT that you wish it to analyze a csv file.
`-working_directory`	OBSTRUCT only works within a single directory. The directory you specify here will house your input files and will be where output files are written. Note that it **should end with a slash**. As before, you can simply drag a folder into your command line to have the full path come up in the correct format for your operating system. If there are any spaces in this directory path, you will need to escape them with quotation marks in Windows (e.g. -C:\Working" "Directory\) and a backslash in Apple OS X/Linux (e.g. -/Users/Smith/Working\ Directory/).
`-input_file`	The filename of the input file. This input file must be in the working directory. Note that you **do not need to specify a path** for this file as you have already specified the working directory.
`-output_file`	The filename for the outputs of OBSTRUCT. These will be saved in the working directory as two files ending in .csv and .r with the filename specified in this parameter. Once again, you **do not need to specify a path** for this parameter.

Table 4: Explanation of the parameters to be used when running OBSTRUCT with a csv file.

Examples of Input Strings

Apple OS X & Linux

```
perl /Users/Smith/Data/ObStruct.pl -csv -/Users/Smith/Desktop/ObStruct\ Data/ -sim.csv
-sim-output
```

Windows

```
perl C:\Users\Smith\Desktop\ObStruct.pl -csv -C:\Users\Smith\Desktop\ObStruct" "Data\ -sim.csv
-sim-output
```

ObStruct *Output*

The overview in the Introduction outlines the process that ObStruct takes to generate its outputs. This section will provide examples of the outputs of ObStruct and explain them. ObStruct generates a total of three files from a run – a csv file of the parsed ancestry profiles[12], an R script for generating CDA plots and another csv file containing the R^2 calculations and significances for your data. The first of the output files is named the same as the input filename but with a .csv extension while the second and third are named based on the output filename specified for the run with .r and .csv extensions, respectively.

[12] For a run where csv is specified as the software this file is not generated.

Parsed Ancestry Profiles CSV

When inputting the outputs of InStruct, structure or baps runs into ObStruct, the first step is to parse[13] them into a csv file which is then used as the true input for ObStruct automatically. To understand this format and to create your own csv file see the Creating a CSV File for Input section. You do not generally need to look at or touch this file unless you need to change population assignments or add/remove data.

[13] Parsing means reading in the input file and processing it.

ObStruct *Main Output CSV*

The primary output from ObStruct contains the calculated R^2 values for: the overall data, all pairwise comparisons of predefined populations, each inferred/predefined population removed in turn and the permutation significance values of the overall data and pairwise comparisons. This output is printed as a csv file which is an ideal simple spreadsheet format. This file can be opened in any text editor, but is most suited to opening in spreadsheet software such as Microsoft Excel. Below is an example of this output in Microsoft

Excel from a dataset of 1,484 humans sampled from 7 continents and analyzed for population structure using STRUCTURE.

Overall R2 for your data	0.76						
p-value	<0.0001						
6	NA	0.92 (<0.001)	0.87 (<0.001)	0.82 (<0.001)	0.86 (<0.001)	0.84 (<0.001)	0.54 (<0.001)
1	0.92 (<0.001)	NA	0.83 (<0.001)	0.29 (<0.001)	0.84 (<0.001)	0.23 (<0.001)	0.52 (<0.001)
4	0.87 (<0.001)	0.83 (<0.001)	NA	0.60 (<0.001)	0.63 (<0.001)	0.74 (<0.001)	0.25 (<0.001)
3	0.82 (<0.001)	0.29 (<0.001)	0.60 (<0.001)	NA	0.69 (<0.001)	0.30 (<0.001)	0.45 (<0.001)
7	0.86 (<0.001)	0.84 (<0.001)	0.63 (<0.001)	0.69 (<0.001)	NA	0.82 (<0.001)	0.55 (<0.001)
2	0.84 (<0.001)	0.23 (<0.001)	0.74 (<0.001)	0.30 (<0.001)	0.82 (<0.001)	NA	0.52 (<0.001)
5	0.54 (<0.001)	0.52 (<0.001)	0.25 (<0.001)	0.45 (<0.001)	0.55 (<0.001)	0.52 (<0.001)	NA
R2 without predefined population 6	0.72						
R2 without predefined population 7	0.72						
R2 without predefined population 1	0.74						
R2 without predefined population 2	0.75						
R2 without predefined population 4	0.76						
R2 without predefined population 3	0.77						
R2 without predefined population 5	0.85						
R2 without inferred population 5	0.75						
R2 without inferred population 6	0.75						
R2 without inferred population 4	0.75						
R2 without inferred population 1	0.75						
R2 without inferred population 3	0.77						
R2 without inferred population 2	0.81						

The first two lines of this output contain the overall R^2 value for your data and the associated p-value. When `p-value <0.0001` is printed, none of the 10,000 permutations of the data resulted in an R^2 value larger than the observed value, meaning more permutations would be needed to obtain a more accurate p-value[14].

The next block in the output is the pairwise matrix of predefined populations. The first column contains the predefined population labels with the same order of populations along the top of the matrix. Values within the matrix are the R^2 values for each pairwise comparison with significance in brackets[15]. `NA` values are, of course, when a population is compared with itself, which is not calculated and therefore not applicable.

Next we have two sets of R^2 values for the rest of the data when each predefined/inferred population is left out in turn. These are sorted by R^2 value from smallest to largest. The populations which decrease the R^2 value from the overall value when removed from the data have a greater than average contribution to the structure within the data. These populations are likely the most structured since without them the rest of the data becomes more variable. Likewise, the

[14] This would be computationally intensive as it could take billions of permutations before one is more structured by chance.

[15] 1,000 permutations are performed for pairwise comparisons to save computing time.

populations that increase the R^2 value from the overall value when removed contribute least to structure – i.e. they contain a large amount of admixture. The calculation of R^2 takes into account population size so a small but highly structured predefined population may not be listed as the most structured in favor of a larger slightly less structured population; this acts to account for sampling variability.

To provide a comparison against the highly structured human data, below we see the same output for simulation data where all five predefined populations contain no significant structure between them.

Overall R2 for your data	0.01				
p-value	0.1488				
1	NA	0.00 (1)	0.01 (1)	0.00 (1)	0.01 (1)
4	0.00 (1)	NA	0.01 (1)	0.00 (1)	0.01 (0.87)
0	0.01 (1)	0.01 (1)	NA	0.01 (1)	0.01 (0.90)
3	0.00 (1)	0.00 (1)	0.01 (1)	NA	0.01 (0.42)
2	0.01 (1)	0.01 (0.87)	0.01 (0.90)	0.01 (0.42)	NA
R2 without predefined population 2	0.01				
R2 without predefined population 0	0.01				
R2 without predefined population 3	0.01				
R2 without predefined population 1	0.01				
R2 without predefined population 4	0.01				
R2 without inferred population 2	0.01				
R2 without inferred population 3	0.01				
R2 without inferred population 5	0.01				
R2 without inferred population 1	0.01				
R2 without inferred population 4	0.01				

R Script for Visualization of Ancestry Profiles

The R script created by ObStruct is used to produce visualizations of the ancestry profiles using canonical discriminant analysis (CDA) [Gittins, 1985]. A CDA searches for a transformation of the ancestry profiles so that the variation and difference between the individuals is mapped to a smaller space of independent factors. The data are transformed such that the data median is located at the origin. This

helps visualize the data easier. The `candisc` package [Friendly, 2007] for the statistical software R provides the transformations and visualizations. In order to run this script, install R and open the .r file in it from the **same directory in which it was created**[16]. You should see a window with some code in it. You can execute the script in two different ways:

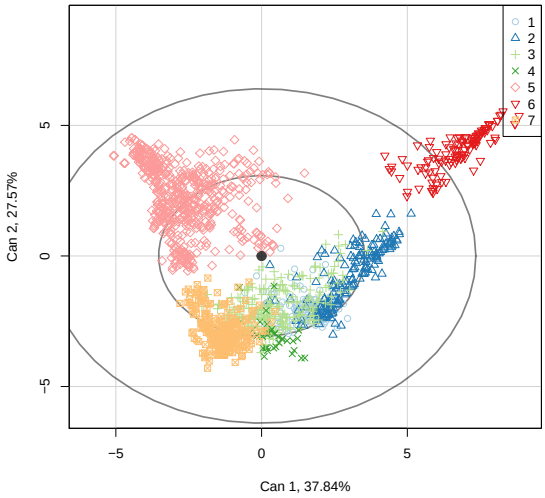
1. Select all of this code by pressing Ctrl+A in Windows or Command+A in Apple OS X. Press Ctrl+R in Windows or Command+Enter in Apple OS X to execute the script.

2. Use the command line of R. Type `getwd()` to check if you are in the right working directory. If not, set the right working directory by typing `setwd("working_directory")` and replace `working_directory` with the directory your files are contained in. Note that R permits autofill by using the tab key. To execute the script simply type `source("script_file.r")` and replace `script_file.r` with the name of the ObStruct output file.

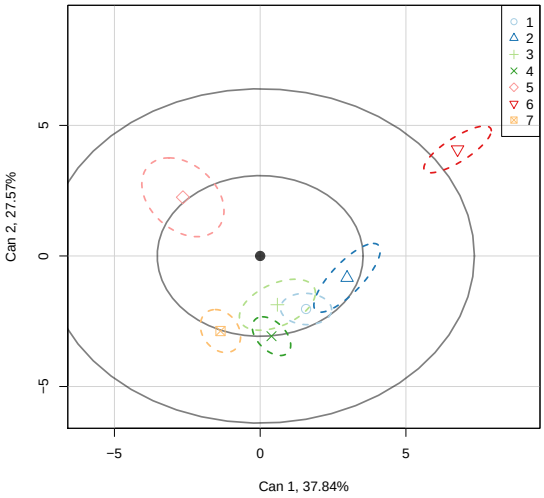
If this is your first time running a CDA analysis, you will need to uncomment the top 6 lines beginning with `# install` by removing the hash (#) symbol. This will force R to install the packages required to run the analysis.

If the script is successful, you will see a folder called 'graphs' in the same directory as the script. This folder will contain three separate visualizations with the following filenames: `candisc_all_data.pdf`, `candisc_ellipse.pdf & candisc_heplot.pdf`. Below are examples of these three visualizations for a dataset of 1,484 humans sampled from 7 continents and analyzed for population structure using Structure.

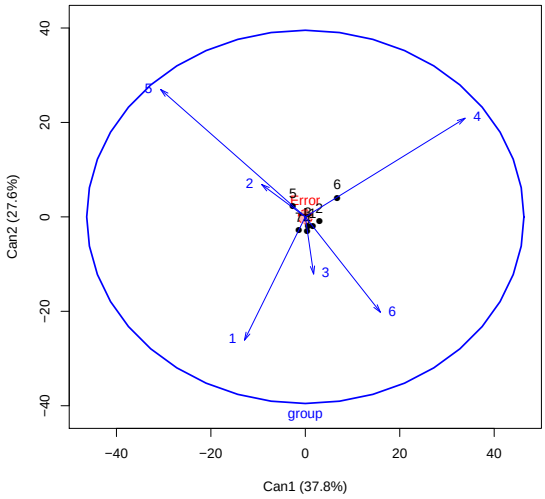
[16] The R script defines the working directory as the directory in which the script was created and in which the ObStruct outputs are housed. If you wish to run the script from another directory, you will need to update the line setwd(<directory>) in the script to the directory of the ObStruct outputs.



(a) `candisc_all_data.pdf`



(b) `candisc_ellipse.pdf`



(c) `candisc_heplot.pdf`

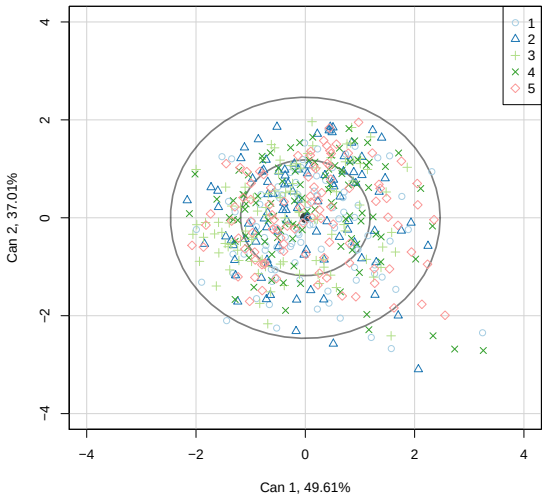
The plots visualize the data with respect to the two dimensions[17] covering the highest variability. The variability is shown as a percentage behind the axis labels. In this example the two best variables cover about 65% of the variability in the data. The first plot (candisc_all_data.pdf) shows every individual within the data colored by their predefined population. The inner grey ellipse contains 50% of all individuals while the outer grey ellipse contains 95% of all individuals. The second plot (candisc_ellipse.pdf) reduces the amount of information by only showing the median and the 50% ellipse for each predefined population. This visualization provides a better indication of the separation of populations and the within group variation. The third plot (candisc_heplot.pdf) is a so-called HE-plot. The outer blue ellipsoid labelled group reflects the variation of the group means around the grand mean[18] while the red circle reflects the pooled within-group dispersion and covariation[19]. A small red circle in a large blue ellipsoid indicates that the data are well discriminated by the two canonical dimensions. If the red circle fills the blue ellipsoid we can assume that the variation of the group means around the grand mean can be explained by the within-group dispersion and covariation. Black dots indicate the group means of the respective populations. Blue arrows indicate the correlation with the linear combinations of the canonical dimensions. For instance, a horizontal line would indicate that the inferred population is strongly correlated with the first canonical dimension and uncorrelated with the second canonical dimension.

Examples 1(a)–1(c) show the plots for the human data. We see that the groups are well discriminated by the first two canonical dimensions and that there is a high correlation between the inferred populations and the predefined populations. Examples 1(d)–1(f) show the plots for an example with no significant difference within the predefined populations, i.e. in this example the data cannot discriminate the predefined populations. These data have been generated in a simulation study where five populations are sampled just after splitting off from their common ancestor, i.e. no divergence has occurred yet and the individuals are effectively from the same population.

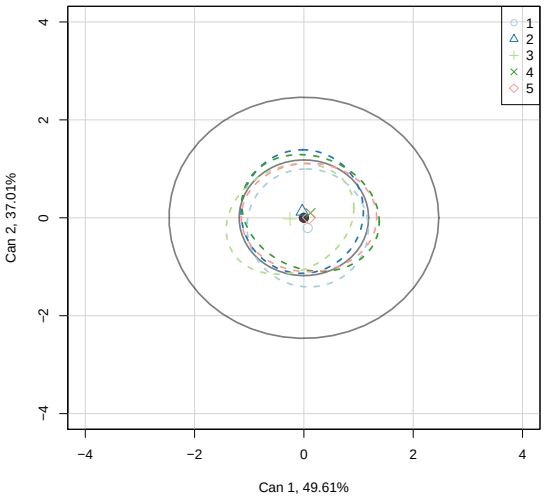
[17] Labelled Can 1 and Can 2.

[18] The numerator of the R^2 statistic.

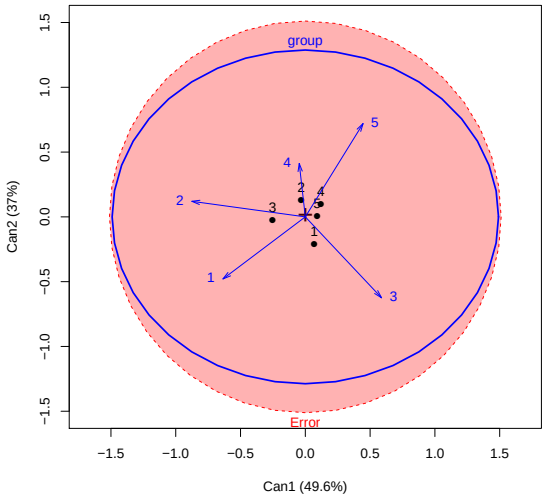
[19] The denominator of the R^2 statistic.



(d) `candisc_all_data.pdf`



(e) `candisc_ellipse.pdf`



(f) `candisc_heplot.pdf`

Extra Information

Comment 1: InStruct has a bug where it does not print the name and population membership of the last individual in its output. ObStruct corrects this by printing "Unknown" and the same population as the individual before the last one. The results are unaffected since the last individual in the data should almost always be in the same predefined population as the one previous to it and the name of individuals does not matter for analysis.

Comment 2: When only two predefined populations exist in the data, ObStruct will still perform its analysis but will not carry out pairwise predefined population calculations/permutations or the removal of each predefined population with recalculation of R^2. These methods require three or more predefined populations.

Comment 3: ObStruct comes with two example outputs from structure and baps for the same human data. Running both of these through ObStruct will produce similar overall R2 values and significance values for all tests. However, the predefined population membership of each individual is different for the two datasets so while both have the same number of individuals grouped in the same populations, the number associated with, for example Africa, is different for each data set. This means that the "R2 without predefined population 1" results from ObStruct will not be comparable for these two data sets as well as the pairwise matrices. In a situation such as this where one might wish to compare the two software packages, it would be advised to take the csv file produced by ObStruct for each of these runs and determine which of the predefined population numbers within them correspond to which continent/region in the data and change these numbers to these regions. Then, both updated csv files can be run through ObStruct again and the correct predefined population names will be printed for each data set to allow easy comparison.

Bibliography

J Corander, P Waldmann, P Marttinen, and M J Sillanpaa. BAPS 2: enhanced possibilities for the analysis of genetic population structure. *Bioinformatics*, 20(15):2363–2369, October 2004.

Jukka Corander, Pekka Marttinen, Jukka Sirén, and Jing Tang. Enhanced Bayesian modelling in BAPS software for learning genetic structures of populations. *BMC bioinformatics*, 9(1):539, 2008.

Michael Friendly. HE Plots for Multivariate Linear Models. *Journal of Computational and Graphical Statistics*, 16(2):421–444, June 2007.

H Gao, S Williamson, and C D Bustamante. A Markov Chain Monte Carlo Approach for Joint Inference of Population Structure and Inbreeding Rates From Multilocus Genotype Data. *Genetics*, 176(3):1635–1651, May 2007.

R Gittins. Canonical analysis: a review with applications in ecology. 1985.

J K Pritchard, M Stephens, and P Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, June 2000.

Noah A Rosenberg. distruct: a program for the graphical display of population structure. *Molecular Ecology Notes*, 4(1):137–138, December 2003.