QInfoMating Manual v. 1.2.2

Antonio Carvajal-Rodríguez

Departamento de Bioquímica, Genética e Inmunología

Universidad de Vigo, Vigo 36310, Spain

Email: acraaj@uvigo.es

Web: https://noosdev0.github.io/QInfoMating/

TABLE OF CONTENTS

QInfomating is a software that performs sexual selection and assortative mating tests for continuous and discrete data. It performs also model selection and multimodel inference to study mate competition and mate choice models and their sexual selection and assortative mating effects.

VERSION

Current version is 1.2.2 (May 2025). In this version:

1) Fixed a minor issue when reading csv file format.

2) Added warning for Linux and macOS users to ensure that executables have execution permission.

In previous version is 1.2.1 (February 2025). In this version:

1) The interface is enhanced with a table displaying the data loaded from the input file and a panel summarizing the main results.

2) The interface is now also available for macOS.

3) An output file is added for the explanation of the models.

4) The estimation of mate choice models is only performed if the mating matrices are square (equal number of female and male classes).

In previous version 1.0.1 (October 2024):

1) The width w of each class calculated as w=ceil(range/K) fails when range is less than 1, so now the range is calculated as w=(range/K).

In previous version 1.0.0 (August 2023):

1) The name of the program has changed to reflect the inclusion of quantitative data analysis. The new name is QInfomating.

2) Added a new input format for quantitative data.

3) The format of the output files has been improved so that the results now look clearer and distinguish between statistical tests and model estimation.

In previous version 0.4:

1) The standard error of each estimate is given for the best fit model which is now included in the brief output file. How the variance of the propensity estimators is calculated is now explained in a new section at the end of this manual.

2) Changed the default zeros i.e. the value that replaces the zero when encountered in the mating counts is set by default to $min(10^{-6}, q_{ij})$.

3) Fixed a bug in S2-2P model that was low-biasing the parameter estimates for this model.

4) Some restrictions has been added before the analysis is done:

      4-1) If the mating class (i,j) has expected random mating frequency q_{ij} such that $q_{ij} \times$ sample size < 1 then the program ends the execution without doing the analysis while warning that sample size is too low for distinguishing from random mating.

      4-2) For a given number of mating classes the lower bound for the sample size is computed following the coupon collector's problem solution (see the new manual section about number of classes and sample size). If the number of matings (the sample size) in the input file does not reach the lower bound the program ends. However, these restrictions can be skipped (see next point).

5) Added a new command line argument for skipping the above restrictions. By default the argument is -force 0 so that the execution is blocked if any of 4-1) or 4-2) happens. However, if the argument -force 1 is given then the program execution is performed in any case.

In previous <u>version 0.3.2</u>:

1) The squared unconditional standard error changed to the unconditional standard error (USE, without squaring)

2) When the parameter estimates are normalized by the mean, the USEs are normalized as well. This was not happening in previous versions.

3) When the parameter estimates are not normalized, the parameters are scaled by the model fixed ones. The resulting output coincides with the model definition in the models file. The USEs are scaled as well.

4) The default output was changed to be non-normalized.

In previous <u>version 0.3.1</u>:

1) The output was divided in three different files. One with the tested model set (Models_InfoMating.out), other with the full output (Full_InfoMating_norm.out) and finally the output with just the matrix for the multimodel inference parameters (Brief_InfoMating.out).

In previous <u>version 0.3.0</u>:

1) New kind of mate choice models added. The choice bias models permit a bias in the preference under a similarity model, i.e. an individual may prefer mating with an equal-size or a higher-size partner.

2) The output presentation was improved with explicit tables of parameter estimates for each kind o model.

In previous <u>version 0.2</u>:

1) The notation for the mate choice models changed. Now noted as C-num_of_parameters.

2) A new category of models has been added. The new models permit a clearer distinction between competition (intra-sexual) and mate choice (inter-sexual) parameters. The notation of the new models include the heading: SfemC-numparams or SmaleC-numparams.

3) Added the possibility of selecting different subset of models when performing the analysis.

THE PROGAM

Download

The program can be downloaded from

https://acraaj.webs.uvigo.es/InfoMating/QInfoMating_v1.2.zip

USER-FRIENDLY INTERFACE AND READY-TO-USE EXECUTABLES

The program can be called using a user-friendly interface (UFI) or directly from the command line. The user-friendly interface is available for Windows, Ubuntu and macOS.

The UFI requires that the command line executable be present in the same folder. This is because UFI simply starts the command line executable with user-defined parameters. The UFI executable is called PyQInfoMating and the command line executable is called QInfoMating (which does not start with Py).

Both UFI and command line runs produce the same output files, as explained in the Output section.
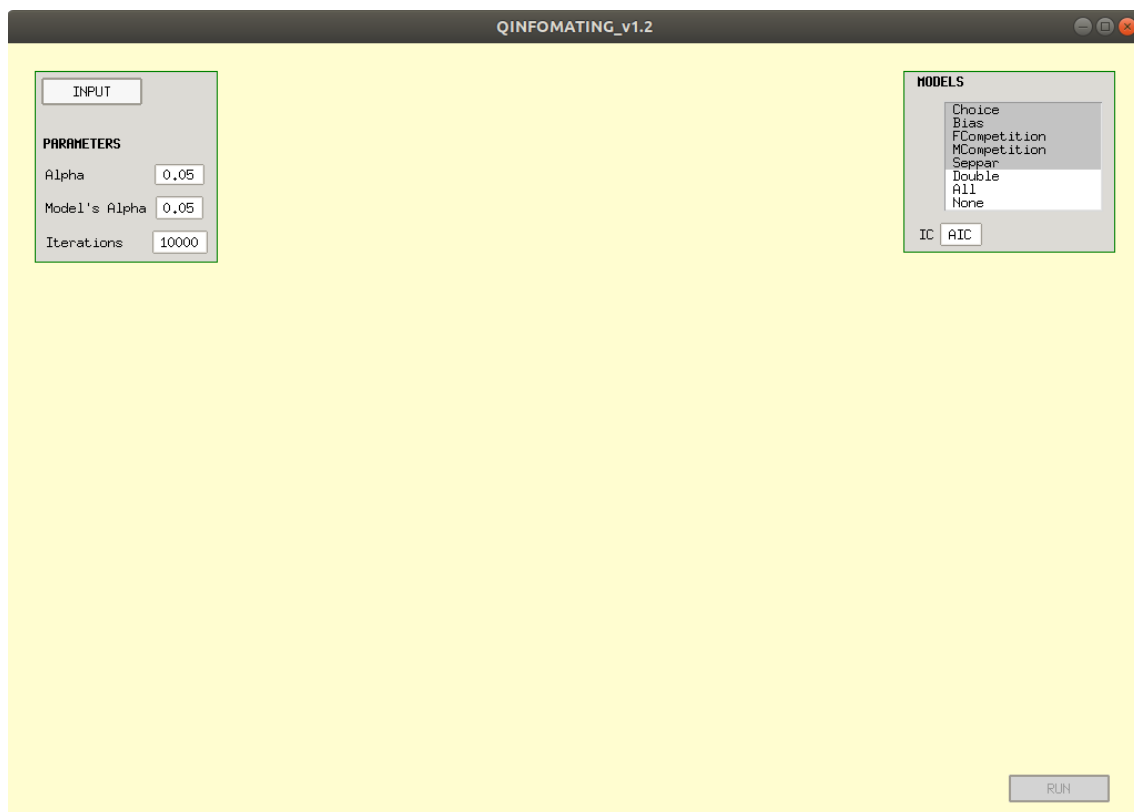
Figure 1. Friendly user QInfomating interface.

Windows

Friendly user interface

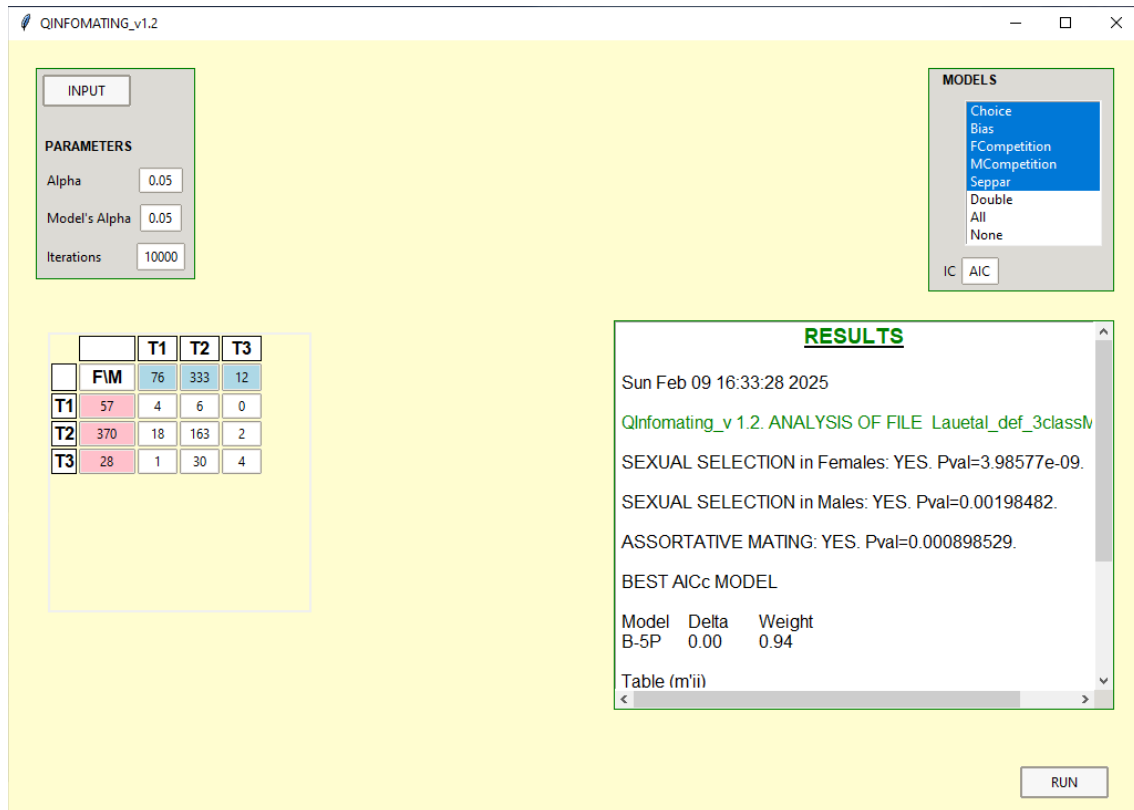To execute the program just double click PyQInfoMating.exe and a interface similar to the one in Figure 1 will appear. The user must choose and adequate input file (see the input file section below) before the program can be executed via the *Run* button. The pre-selected models are the models by default than can be changed for any single model or combination of them (only for discrete data or continuous data discretized by the program, see the Analysis of Continuous data section).

The remaining parameters are explained in the Program Options section.

Please note that for the program to work, the command line executable must be located in the same directory as the interface one. That is, PyQInfoMating.exe and QInfoMating.exe must be present in the same folder.

Command line

If you prefer to directly run the program without the interface then you can use the QInfoMating.exe (without the Py) by double clicking for running the program under default options. Alternatively, you can go to the command prompt (cmd.exe) and type

8

QInfoMating

You can also access the Run command by pressing the Windows logo key ⊞+r
then drag and drop the .exe file from your folder and add the desired arguments, e.g. if
the QInfoMating.exe is in the folder QInfoMating then after drag and drop you would
have

C:\QInfoMating\QInfoMating.exe

now add the desired arguments and then hit the Intro key. For example:

C:\QInfoMating\QInfoMating.exe -input QInfomating.txt

If you have data stored in QInfomating.txt file in the same folder as the program or

C:\QInfoMating\QInfoMating.exe -path ../InputFile_Examples/ -input
QInfoMating.txt

if the file is in the subfolder InputFile_Examples of a higher level directory.

See the Program usage and command line arguments sections and examples below for
more options.

Linux (Ubuntu)

Friendly user interface

To run the program, navigate in the terminal to the folder where the PyQInfoMating and
QInfoMating programs are located and type ./PyQInfoMating to get an interface similar
to Figure 1. The user must choose and adequate input file (see the input file section
below) before the program can be executed via the *Run* button. The pre-selected models
are the models by default than can be changed for any single model or combination of
them (only for discrete data, see the Analysis of Continuous data section). The
remaining parameters are explained in the Program Options section.

*As before, PyQInfoMating and QInfoMating programs must be present in the same
folder. Also note that both executables must have execute permission.*

Command line

If your data is already stored in a file called QInfomating.txt (with discrete data format)
or in a file called QInfomating.csv (in a continuous data format, i.e. format 4 or 5) you
just need to type from the console or terminal:

./QInfoMating

If you have both types of files in the same folder as the executable, then by default the
program read the QInfomating.txt file. If you prefer that the program chooses the
QInfomating.csv file you should add the tag -Q typing:

./QInfoMating -Q

If your input file is called XYdata.csv and is in the subfolder InputFile_Examples of a higher level directory then you can type

./QInfoMating -path ../InputFile_Examples/ -input XYdata.csv

See the Program usage and command line arguments sections and examples below for more options.

macOS

Friendly user interface

To run the program, navigate in the terminal to the folder where the PyQInfoMating and QInfoMating programs are located and type ./PyQInfoMating to get an interface similar to Figure 1. *As before, <u>PyQInfoMating and QInfoMating programs must be present in the same folder</u>. Also note that both executables must have execute permission.*

Command line

As before you can run the program by typing from the console or terminal:

./QInfoMating

and the desired arguments (see the command line section).

Compilation

To run the program on any Unix/Linux/macOS system you can also compile it, which is as simple as going in the terminal to the source folder of the program, which should include a file called Makefile, and simply typing make, which should produce a QInfoMating executable file that can be run following the same steps explained in the Linux command line section above.

But please note that if you compile and then want to use the friendly interface, keep in mind that the obtained PyQInfoMating_v1.2.2 executable must be appended with _U if the system is Ubuntu-Linux (PyQInfoMating_v1.2.2_U) or _macOS if it is macOS (PyQInfoMating_v1.2.2_macOS).

INPUT FILES

Sample files for discrete and continuous data are included in the program's distribution folder.

Discrete data

The program requires an input file (.txt extension) containing the mating data (Figure 2). It accepts various formats including the format of the JMating software (Carvajal-Rodríguez and Rolan-Alvarez, 2006).

A

```
# format number
0
# num of types
2
#premating male numbers
720 720
#premating female numbers
720 720
# matings by rows (females)
611 87
247 759
```

B

```
# format number
1
# num of types
3
# matings by rows (females)
102 40 60
40 120 230
30 70 77
```

C

```
# format number
2
# num of types
2 3
#premating male numbers
500 500 500
#premating female numbers
720 720
# matings by rows (females)
611 87 300
247 759  120
```

D

```
# format number
3
# num of types
2 3
# matings by rows (females)
611 87 300
247 759  120
```

E

```
# format number
15
# num of types
2 3
#premating male numbers
0.333 0.333 0.334
#premating female numbers
0.5   0.5
# matings by rows (females)
611 87 300
247 759  120
```

Figure 2. Different formats allowable for discrete data in QInfoMating. A: Format number 0, the same as JMating. B: Format 1, similar to format 0 without premating information. C: Format 2, the same as format 0 but the number of types for females and males can be different. D: Format 4, the format 2 without premating information. E: Format 15, the format 2 with the premating information in the form of relative frequencies.

Continuous data

The program requires an input file (.csv extension) containing the mating data. The input file must be a csv file (Figure 3). It accepts two formats. Format 4 (Figure 3A) should include population (premating data) and mating data. Format 5 (Figure 3B) includes only mating data. The first line begins with the tag #Format, the second should include the number of the format (4 or 5). The third line should have the tag # followed by any informative word as Data, Nfem or pairs. The fourth line under the format 4 has three numbers indicating the number of females in the premating data, the number of males and the number of mating pairs, respectively. Under the format 5, the fourth line only has the number of pairs. Line five have the tag # followed by any informative names and the remaining lines from line six to the end include the data.

Under the format 4 the first column includes female population data, the second the male population data, the third column includes the mated females and the fourth the mated male corresponding to each pair. The first two columns can have different length i.e. different number of females and males in the population data.

Under the format 5 the first two columns correspond to mating females and males. The columns with the pairing data must have the same length.

A

	A	B	C	D	
1	#Format				
2	4				
3	#Nfem	Nmales	pairs		
4	25	20	10		
5	#Pop fems	Pop Males	Mat fems	Mat males	
6	9.7	7	9.7	7	
7	7	5.7	7	5.7	
8	8.8	8.5	8.8	8.5	
9	6.8	7.9	6.8	7.9	
10	10.6	7.5	10.6	7.5	
11	6.1	7.8	6.1	7.8	
12	6.8	7.1	6.8	7.1	
13	8.3	8.1	8.3	8.1	
14	9.3	8.2	9.3	8.2	
15	7.9	7.3	7.9	7.3	
16	5.2	6.5			
17	8.7	7.2			
18	6.7	8.2			
19	9.2	6.1			
20	8.3	8.1			
21	9.6	7			
22	9.4	8.7			
23	10.3	8			
24	7.7	6.3			
25	9.9	6.5			
26	9.6				
27	8.9				
28	8.5				
29	10.8				
30	9.1				

B

	A	B
1	#Format	
2	5	
3	#pairs	
4	10	
5	#Mat fems	Mat males
6	9.7	7
7	7	5.7
8	8.8	8.5
9	6.8	7.9
10	10.6	7.5
11	6.1	7.8
12	6.8	7.1
13	8.3	8.1
14	9.3	8.2
15	7.9	7.3
16		

Figure 3. Different formats allowable for continuous data. A: Format 4. B: Format 5.

ANALYSIS OF CONTINUOUS DATA

Sexual selection tests

Sexual selection tests for quantitative traits are applied following (Carvajal-Rodríguez, 2024).

Assortative mating tests

Assortative mating tests for quantitative traits are applied following (Carvajal-Rodriguez, 2024).

Grouped frequency distribution

If any of the above tests are significant the continuous data is discretized for performing multimodel inference (see the discrete data section). The discretization is performed as follows:

1- The range of the data set is calculated (maximum value *Max* - minimum value *Min*).

2- The formula (1) that relates number of classes *K* and sample size *n'* (see the section at the end of this manual) is utilized for defining the maximum number of classes given the mating sample size. However, that formula is for uniform frequencies, whereas if there are low frequencies, the required sample size would be much larger. Therefore, we conservatively assume a sample size more than one order of magnitude larger than that required for the uniform frequency case.

$$K[\ln(K)+0.5777]+\frac{1}{2} \leq \frac{n'}{20\ln(K)}$$

3- The width *w* of each class is computed as *w*= (range/*K*).

4-Make the classes. The first class is always [*Min,Min*+w) and the last class is [*x,Max*+w).

5- Compute the frequency for each class. Check it is represented in each sex, if not decrease *K* and repeat from 3.

Note that under this method the number of classes is the same for females and males. Once the data has been classified into classes, a table of matings is created and the program proceeds to model selection. However, the option to select model types through the interface only works for discrete data. If the data are continuous, the set of models to test depends on the results of the quantitative statistical tests. For example, if assortative mating and sexual selection are detected in both sexes, the Double Effect models will also be tested together with Choice, Bias and Competition regardless of whether they are selected or not in the menu.

ANALYSIS OF DISCRETE DATA

Statistical tests

Sexual selection and assortative mating tests for discrete traits are applied following (Carvajal-Rodríguez, 2018).

Mating models

The program can generate different kind of non-random mating models (Carvajal-Rodríguez 2020). In the models, females are represented by rows, and males by columns. There are three main type of models.

1.- *Mate competition models*

These models can still be divided in female competition, male competition or competition in both sexes. In the female competition models, the mating parameter(s) distinguishes some females (rows) from others and may produce female sexual selection. The model name starts with Sfem and then a number indicating the number of parameters, e.g. for a one parameter model: Sfem-1P. The parameters are named with the letter 'a' and they are repeated row-wise.

Similarly in the male competition models, the mating parameter(s) distinguishes some type of males (columns) from others and may produce male sexual selection. The model name starts with Smale and then a number indicating the number of parameters, e.g. for a one parameter model: Smale-1P. The parameters are named with the letter 'b' and they are repeated column-wise.

2.- *Mate choice models (with or without bias)*

In the mate choice models the parameter(s) are in the diagonal (named with the letter 'c') and may produce assortative mating patterns. The model name starts with C and then a number indicating the number of parameters, e.g. for a one parameter model: C-1P.

In the choice bias models the choice parameters ('c') are in the diagonal and the bias parameters ('B') in the subdiagonal. The model name starts with B and then a number indicating the number of parameters as in B-2P.

3.- *Double effect models*

There are also some models that have both kind of parameters i.e. mate competition and mate choice parameters. Finally, for some models, each parameter may produce a double effect i.e. sexual selection plus assortative mating. The user can decide to run some of these model subsets or all of them (see the 'models_' tags in the command line section).

The random mating model (M_0) and the saturated model (M_{sat}) are always performed (see the tag 'models_none' in the command line section).

Non-square matrices (different number of female and male classes)

Mate choice models and double models are expected to have the same number of classes in males and females to study assortative mating. When the number of classes is different, only sexual selection models are calculated. If you still want these models to be calculated, the missing mating classes must be filled with a value of 0 or with the value that the user wants to replace the zero. This value can also be configured using the

-zeros tag on the command line. However, non-zero population numbers for missing classes must also be provided to allow model estimation.

User-defined models

In addition to the above models automatically generated by the program, the user can define her/his own. This option is only available under the command line version. This is recognized at the input, when the argument with the tag -user is found followed by a file name. The file (Figure 4) must contain the number of user-defined models, the number of female and male types at each model and finally the values of the mutual mating propensities (positive numbers).

```
# number of models
3

# MODEL 1 from pooled data in Author et al 2014

# num of types
2   2

# mutual propensities by rows (females)
1    1
1    0.6


# MODEL 2 best model from analysis in Author et al 2015

# num of types and number of parameter (c)
2    2            1
# mutual propensities by rows (females)
0.81   1
1       0.63


# MODEL 3

# num of types
2 2
# mutual propensities by rows (females)
3     0.45
1     0.45
```

Figure 4. User-defined model input file example.

The number of types always correspond to two numbers, one for female the other for male. However, a third number can appear (see MODEL 2 in Figure 4) indicating the number of parameters when it is not obvious. This could be necessary because by default, the program considers as different parameters those numbers that are different from 1. It could happen that different values correspond to only 1 parameter as in (1+c, 1- c). In this case, the use of the third value after the number of types allows to indicate the correct number of parameters.

The model input file must be in the same folder that the data input file.

PROGRAM OPTIONS

Default values

Calling the program without arguments, i.e.

QInfoMating.exe

It is equivalent to calling

QInfoMating.exe -path ./ -input QInfoMating.txt -user " " -zeros 1e-6 -numiter

10000 -SL 0.05 -SLmodel 0.05 -rdistrib 0 -output " " -models_none 0 -

models_all 0 -verbose 0 -force 0

If the program does not find an input file called QInfoMating.txt in the same folder then

it looks for QInfoMating.csv.

If both kinds of files exist in the same folder and we want to read the csv file then we

must make the calling with just the argument -Q, i.e.

QInfoMating.exe -Q

The csv file should be in one of the two continuous data formats (format 4 or 5).

Thus, for the basic execution (double click under windows) we only need a file called

QInfoMating.txt containing mating data in any of the formats indicated in Figure 2 or a

file called QInfoMating.csv with data in the formats from Figure 3.

Note that without premating information, i.e. with formats 1, 3 or 5 (Figure 2-B, -D,

Figure 3-B) the program does not perform model selection but computes the J_{PSI} value

and its significance (Carvajal-Rodríguez, 2024, 2018).

Command line options

The following arguments can be passed to the program:

-force <BOOLEAN> When set to 1, it forces the program to execute even if the

population frequencies or the sample size are not enough for providing minimal

reliable estimates. By default is 0.

-IC <STRING> Information Criterion. By default is AICc (AIC with sample size

correction). Can be changed to KICc or BIC.

-input <STRING> Specifies the input file name. By default is QInfoMating.txt. If given

it overwrites the argument -Q.

-models_choice <BOOLEAN> When set to 1, it tests against models with choice

parameter(s) in the diagonal. These type of models may produce assortative

mating. The models are avoided when the tag is set to 0, or when the number of female and male classes is different. By default is 1.

-models_bias <BOOLEAN> When set to 1, it tests against models with choice parameter(s) in the diagonal and choice bias parameters in the subdiagonal. This type of models may produce assortative mating with or without bias. The models are avoided when the tag is set to 0, or when the number of female and male classes is different. By default is 1.

-models_femcompet <BOOLEAN> When set to 1, it tests against models with parameter(s) in one or more rows. These type of models may produce female sexual selection. The models are avoided when the tag is set to 0. By default is 1.

-models_malecompet <BOOLEAN> When set to 1, it tests against models with parameter(s) in one or more columns. These models may produce male sexual selection. The models are avoided when the tag is set to 0. By default is 1.

-models_seppar <BOOLEAN> When set to 1, it tests against models having at least two parameters, one for male or female competition (columns or rows) and other for mate choice (diagonal). These models may produce sexual selection linked to one parameter and assortative mating linked to the other. The models are avoided when the tag is set to 0, or when the number of female and male classes is different. By default is 1.

-models_double <BOOLEAN> When set to 1, it tests against models having at least one parameter that may have, per se, a double effect of producing sexual selection plus assortative mating. The models are avoided when the tag is set to 0, or when the number of female and male classes is different. By default is 0.

-models_all < BOOLEAN > When set to 1, it tests against all the above models. If the tag is set to 0 only the models having their tag to 1 are performed. By default is 0.

-models_none <BOOLEAN> When set to 1, it performs only the random mating and the saturated models plus any user-defined model. Note that, if both the tag 'all' and the tag 'none' are set to 1, the tag 'all' predominates so that all models are performed. By default is 0.

-normalize <BOOLEAN> When set to 1, the mating propensities in the output file are divided by the mean propensity. By default is 0.

-numiter <INTEGER> Defines the number of iterations for the randomization test when
computing the significance of the discrete statistical tests (*J* tests). By default is
10,000.

-output <STRING> Specifies the output file names. See the output section below.

-path <STRING> Specifies the path to the input file if it is not in the same folder as the
executable.

-Q. Don't require any value. It changes the default input file to be QInfoMating.csv.

-rdistrib <BOOLEAN> When set to 1, it writes the discrete distribution of *J* values
under random mating (file RandDistrib.txt). By default is 0.

-SL <DOUBLE> Defines the significance level for the Chi-square and *t* tests.

-SLmodel <DOUBLE> The value (1 - SLmodel), defines the minimum weight for
considering the best model as the only one (i.e. not perfoming multimodel
inference). For example, if -SLmodel 0.05 then if the best model have at least a
weight of 0.95 then it would be considered the only model for performing the
parameter inference. If the best model have less than 0.95 weight, then
multimodel inference will be performed. SLmodel is also utilized for the list of
models in the multimodel output. Only models having weight of at least
SLmodel are given. By default SLmodel=0.05.

-user < STRING > Specifies the file name that contains the user-defined models.

-zeros <DOUBLE> Indicates which value must replace the zero when encountered in
the mating counts. By default is 10^{-6}.

Examples of command line arguments

Input file location in Linux/macOS

If the input file is in the same folder as the program:

> *-inputfile QdataFormat4.csv -numiter 10000 -SL 0.05*
>
> or also
>
> *-path ./ -inputfile QdataFormat4.csv -numiter 10000 -SL 0.05*

If input file is in a data folder at the same level as the program folder:

> *-path ../data/ -inputfile QdataFormat4.csv -numiter 10000 -SL 0.05*

If the input file is in a subfolder and skipping unnecesary default parameters:

> *-path ./data/Models/ -inputfile QdataFormat5.txt*

Input file location in Windows

     -path C:\data\Models\ -inputfile ToyExample0.txt -output Toy0_Results.txt

Models tested

Under continuous data, the models tested by default depend on the tests that were significant. However, the user can still define the kind of models to test. To choose the desired models the user need to set to 0 the unwanted models that are by default to 1 and viceversa.

Note that the random mating (M_0) and the saturated (M_{sat}) models are always tested.

If we want to test only choice models (including bias)

     -inputfile QdataFormat4.csv -models_femcompet 0 -models_malecompet 0 -models_seppar 0

If we want to test only female sexual selection models

     -inputfile QdataFormat4.csv -models_femcompet 1 -models_malecompet 0 -models_seppar 0 -
     models_choice 0 -models_bias 0

If we don't want to test any model but M_0 and M_{sat}.

     -inputfile QdataFormat4.csv -models_none 1

If we want to test all models

     -inputfile QdataFormat4.csv -models_all 1

Information criteria

If we want to use the bayesian information criterion instead of the AIC.

     -inputfile QdataFormat4.csv -IC BIC

OUTPUT

The output will be written to a folder called QIM_Results which will be created if it doesn't exist. If the input format does not include premating information (population data) QInfomating only performs the assortative mating tests (continuous and/or discrete).

Regardless of the data type, the program produces a file with the results summary. If the name of the input file was X.txt then the summary output file will be called *QISummary_X.txt .out*. The following output files are also generated.

Discrete data

For formats that contain premating data, the program generates four different files. If the name of the input file was X.txt then the output files will be called:

Discrete_Tests_QIn_X.txt.out

AICc_BestFitModel_QIn_X.txt.out

AICc_MultiModel_QIn_X.txt.out

Models_Explained_QIn_X.txt.out

Continuous data

If the input file is in format 4 and is called X.csv. The program performs the statistical tests for sexual selection and assortative mating and produces a file called

Quantitative_tests_QInFile_X.csv.out

If there were no significant tests this is the only additional file produced. On the contrary, if any test was significant the program will classify the data in classes and perform the discrete analysis so also producing the following files

Class_Intervals_QInFile_X.csv.out

Discrete_Tests_QIn_X.csv.out

AICc_BestFitModel_QIn_X.csv.out

AICc_MultiModel_QIn_X.csv.out

Models_Explained_QIn_X.csv.out

Mating table from continuous data

The *Class_Intervals_QInFile_* file contains the class intervals generated by the program, jointly with the matings table for these classes in the JMating format (Carvajal-Rodríguez and Rolan-Alvarez, 2006) so that the user can copy this table and repeat the analysis directly for discrete data selecting the desired models in the interface or with other program as JMating.

NUMBER OF CLASSES AND SAMPLE SIZE

There is a relationship between the number of classes we may observe in a sample and the sample size. This is also known as the coupon collector's problem (Lewontin and Prout, 1956). The lower bound for the sample size n' required to observe at least one element of each class from K classes is approximated by

$$n' \geq K[\ln(K) + \gamma] + \frac{1}{2} \quad (1)$$

where $\gamma = 0.5777$ is the Euler- Mascheroni constant. In Table 1 the lower bound is given for different number of mating classes. However, the formula (1) is derived assuming equal frequencies for the classes which, if not true, would require higher sample size depending on the random mating frequency distribution. Thus, under a general probability distribution the required sample size is (Flajolet et al., 1992).

$$n' = \sum_{q=0}^{K-1} (-1)^{K-1-q} \sum_{|J|=q} \frac{1}{1-P_J}$$

with $P_J = \sum_{j \in J} P_j$

If we assume uniform probabilities we have $P_j = 1/K$ and so (2) becomes

$$n' = (-1)^{K-1-q} + \sum_{q=1}^{K-1} (-1)^{K-1-q} \binom{K}{q} \frac{K}{K-q} \qquad (2)$$

that gives the exact computation for the lower bound in (1).

For example, if $K = 2$ we would have the maximum cardinality of J, $|J| = q = 1$ so $J = \{p_1\}$ or $J = \{p_2\}$ so $P_{J=\{pi\}} = p_i$ and the formula (2) gives $n' = -1 + 1[\ 1/(1-p_1) + 1/(1-p_2)]$ that under the uniform case $p_1 = p_2 = 0.5$ provides the lower bound $n' = 3$. Note that as some p_i tends to 1, n' tends to infinite. Thus, the more extreme the frequencies the higher the sample size required to capture all classes.

In the context of the estimation of the mating propensities, if under the random mating distribution we don't have the sample size required by (2) we expect some classes to have zero counts just by random so we cannot provide a reliable estimate of mating propensity for this class and the parameter value will appear as a 0. However, if a model have the same parameter for several classes, the zero class may have an estimate although that model would probably has a very low weight within the set of candidate models.

Table 1. Sample size lower bound for observing *K* classes.

K	Sample size lower bound

2	3
4	8
9	25
16	54
25	95
36	150

VARIANCE OF THE MULTINOMIAL PROPENSITY PARAMETERS

The variance of the maximum likelihood estimate of the expected frequency q'_{ij} of each mating class under sample size n' is

$$V(\hat{q}'_{ij}) = \frac{\hat{q}'_{ij}(1-\hat{q}'_{ij})}{n'} = \frac{n'_{ij}(n'-n'_{ij})}{n'^3}$$

with $\quad \hat{q}'_{ij} = \frac{n'_{ij}}{n'}$

since we ignore the value of q'_{ij}.

The relationship between the mating parameter of interest c and the corresponding frequencies in the sample (Carvajal-Rodríguez, 2020) is

$$\hat{c} = \frac{\lambda(c)}{\lambda(1)} = \frac{n'_c q_1}{q_c n'_1} \quad \text{with} \quad \frac{n'}{M} = \frac{n'_1}{q_1} \quad \text{then} \quad \hat{c} = \frac{n'_c M}{q_c n'}$$

with $\quad M = \sum_{i \in K} q_i m'_i \quad$ the mean propensity in the model with parameters m'_i and the random mating population frequencies q_i.

so $\quad \hat{c} = \frac{\hat{q}'_c M}{q_c} \quad$ and then we obtain the variance of the maximum likelihood estimate of

c as

$$V(\hat{c}) = \left(\frac{M}{q_c}\right)^2 \frac{n'_c(n'-n'_c)}{n'^3} = \left(\frac{M}{q_c}\right)^2 \frac{\hat{q}'_c(1-\hat{q}'_c)}{n'} \qquad (3).$$

It is clear that the higher the sample size the lower the variance. On the contrary, the lower the expected random mating frequency q_c of the mating classes having mating parameter c, the higher the variance.

UNCONDITIONAL STANDARD ERROR

If we have various candidate models with different weights w_k we may obtain an upper bound of the variance of the parameter c by the geometric mean over these models (Buckland et al., 1997). In addition we may consider the model misspecification for a given model k as

$$\beta = \hat{c}_k - \bar{\hat{c}}$$

with $\quad \bar{\hat{c}} = \dfrac{\sum_k w_k \hat{c}_k}{\sum_k w_k}$

Then a conservative estimate for the (unconditional) standard error of the estimator of c not conditioned for the specific model is (Symonds and Moussalli, 2011)

$$USE_u(\hat{c}) = \sum_k w_k \sqrt{vV(\hat{c}) + \beta^2}$$

where $V(c)$ is the variance as computed in the previous section and v is the overdispersion.

REFERENCES

Buckland, S.T., Burnham, K.P., Augustin, N.H., 1997. Model Selection: An Integral Part of Inference. Biometrics 53, 603–618. https://doi.org/10.2307/2533961

Carvajal-Rodríguez, A., 2024. Unifying quantification methods for sexual selection and assortative mating using information theory. Theor. Popul. Biol. 158, 206–215. https://doi.org/10.1016/j.tpb.2024.06.007

Carvajal-Rodríguez, A., 2020. Multi-model inference of non-random mating from an information theoretic approach. Theor. Popul. Biol. 131, 38–53. https://doi.org/10.1016/j.tpb.2019.11.002

Carvajal-Rodríguez, A., 2018. Non-random mating and information theory. Theor. Popul. Biol. 120, 103–113. https://doi.org/10.1016/j.tpb.2018.01.003

Carvajal-Rodríguez, A., Rolan-Alvarez, E., 2006. JMATING: a software for the analysis of sexual selection and sexual isolation effects from mating frequency data. BMC Evol Biol 6, 40.

Flajolet, P., Gardy, D., Thimonier, L., 1992. Birthday paradox, coupon collectors, caching algorithms and self-organizing search. Discrete Appl. Math. 39, 207–229. https://doi.org/10.1016/0166-218X(92)90177-C

Lewontin, R.C., Prout, T., 1956. Estimation of the Number of Different Classes in a Population. Biometrics 12, 211–223. https://doi.org/10.2307/3001762

Symonds, M.R.E., Moussalli, A., 2011. A brief guide to model selection, multimodel inference and model averaging in behavioural ecology using Akaike's information criterion. Behav. Ecol. Sociobiol. 65, 13–21. https://doi.org/10.1007/s00265-010-1037-6