HABFUZZ

Command-line

A command-line tool for data-driven fuzzy modelling

A quick user's guide developed by
Christos Theodoropoulos

Updated for version 2.5 in April 2019

HABFUZZ
Command-line

Contact Information

Hellenic Centre for Marine Research
Institute of Marine Biological Resources and Inland Waters
46.7 km Athens-Sounio ave.
19013
Anavyssos
Greece
Tel. +30 22910 76335
Fax. +30 22910 76419
Email. ctheodor@hcmr.gr
URL. http://www.hcmr.gr/en/

National Technical University of Athens
Department of Water Resources and Environmental Engineering
Iroon Polytechniou 5
15780
Athens
Greece
Tel. +30 210 7722809
Fax. +30 210 7722814
Email. stamou@central.ntua.gr
URL. https://www.hydro.ntua.gr/?set_language=en

To report errors, bugs, possible amendments or anything else you would like to indicate, please contact Mr. Christos Theodoropoulos at ctheodor@central.ntua.gr

TABLE OF CONTENTS

ABBREVIATIONS

R: Response variable, the output of HABFUZZ

P1: Predictor 1, first input variable

P2: Predictor 2, second input variable

P3: Predictor 3, third input variable

P4: Predictor 4: fourth input variable

f: Fuzzy set

K: Habitat suitability

V: Flow velocity

D: Water depth

S: Substrate type

T: Water temperature

1. Overview

The concept of HABFUZZ is the following:

1. You have a set of observations relating up to four predictor variables with one response variable.
2. You have another set of the same predictors, but probably differently combined, in which the value of the response variable is unknown.
3. HABFUZZ can be used to predict the value of the response variable for each predictor combination.

You can use HABFUZZ at any case that applies to the above concept. HABFUZZ was initially developed as a habitat model and the manual is focused on predicting the habitat suitability in samples with known flow velocity, water depth, substrate and temperature and unknown habitat suitability. But you can use HABFUZZ with any topic that requires the prediction of a response variable (R) based on the combination four predictors (P1, P2, P3, P4) and using fuzzy logic. Just send us an email if you need assistance on this, at ctheodor@hcmr.gr

HABFUZZ is fully automated, you only need to prepare a single input file with your dataset to be used in the software, that's all. A small example dataset (focused on habitat modelling) is provided here to ease the understanding of the processes followed in HABFUZZ (Table 1).

> Note that to run HABFUZZ and get the output, there is no need to know its algorithms. Once you prepare the input file, all processes including the development of the fuzzy rules are internally applied by HABFUZZ.

Table 1. Example dataset for this tutorial

Observation	Flow velocity (m/s)	Water depth (m)	Substrate type
1	0.28	0.29	Boulders
2	0.05	0.08	Large stones
3	0.46	0.80	Small stones

1.1. The fuzzy logic algorithm

(This is only for reference, just proceed to section 4 to directly run HABFUZZ)

As initially proposed by Zadeh in 1965 and described in detail by Ross in 2010, the process of predicting the response variable R using four predictors and fuzzy logic can be summarized in four steps (Fig. 1):

Step 1. Fuzzification of the input variables

In this step, the user defines categories (called membership functions or fuzzy sets) for each input variable and the input values of P1, P2, P3 and P4 are assigned to one or more fuzzy sets. By this process, crisp numerical values of each input variable are converted to a fuzzy 'membership degree', ranging from 0 to 1 for each fuzzy set. Then the process continues using the fuzzy sets and their membership degrees instead of the crisp numerical inputs.

In a habitat model for example …

… a water depth value of 0.14 m may yield a membership degree of 0.7 for the 'shallow' fuzzy set and 0.28 for the 'very shallow' fuzzy set (Fig. 1).

Step 2. Application of a fuzzy operator (AND or OR) in the antecedent (IF-THEN rules)

The AND (min) or OR (max) operator is applied to each combination of variables (fuzzy sets since step 1) and the derived value is assigned to the fuzzy set of the response variable (also defined in step 1). For example, if the user defines five fuzzy sets for R (f_1, f_2, f_3, f_4, f_5), then the application of the fuzzy operator would result in

$$f_4(R) = \min\ \left(f_2(P1), f_3(P2)\right)$$
$$f_3(R) = \min\ \left(f_3(P1), f_2(P2)\right)$$
$$f_4(R) = \min\ \left(f_2(P1), f_4(P2)\right)$$

where,

f_i denotes the fuzzy set of each input and output variable (P1, P2, P3, P4 and R)

P1 is the first input variable (predictor 1)

P2 is the second input variable (predictor 2)

R is the response variable

etc., until all possible combinations of fuzzy inputs are assigned to an output fuzzy set, based on the rationale that; Rule-1: IF P1 is f_2 AND P2 is f_3 THEN R is f_4; Rule 2: IF P1 is f_3 AND P2 is f_2 THEN R is f_3 etc.

In a habitat model …

… these rules would be; **Rule 1**: IF flow velocity is f_2 (low) AND water depth is f_3 (moderate) THEN habitat suitability is f_4 (good); **Rule 2**: IF flow velocity is f_3 (moderate) AND water depth is f_2 (shallow) THEN habitat suitability is f_3 (moderate); **Rule 3:** IF flow velocity …

Step 3. Aggregation of outputs

In this step, the derived fuzzy sets of R for each rule are combined into a single fuzzy set. Usually, the OR (max) operator is applied to aggregate the same output fuzzy sets of the previous step. For example, the f_4(R) fuzzy set was observed in the previous example two times by the IF-THEN rules. The final fuzzy set representing each R class F_j would be

$$F_j = \max(f_i^1(R), f_i^2(R), \dots, f_i^v(R))$$

Step 4. Defuzzification

This final step is applied to derive a single R value, by combining the fuzzy sets of all R classes. Among the various defuzzification methods, the *centroid, maximum membership, weighted average* and *mean-max membership* methods are implemented in HABFUZZ and described below.

a. Centroid defuzzification

Often called the 'center of gravity' or 'center of area'. It can be defined by the algebraic expression

$$R = \frac{\int xf(x)dx}{\int f(x)dx}$$

which is numerically approximated in HABFUZZ by

$$R = \frac{\sum_{i=1}^{n} x_i(f(x_i))}{\sum_{i=1}^{n}(f(x_i))}$$

where,

$f(x_i)$ is the membership degree at value x_i

b. Maximum membership defuzzification

This is the maximum membership degree observed by the aggregation step:

$$R = \max(f(x))$$

c. Weighted average

This method can be used only for symmetrical output membership functions and is calculated by weighting each output membership function by its largest membership degree:

$$R = \frac{\sum_{i=1}^{n} \bar{x}_i(f(\bar{x}_i))}{\sum_{i=1}^{n}(f(\bar{x}_i))}$$

where,

$f(\bar{x}_i)$ is the membership degree at the average value $\bar{x}_i$ of each membership function

d. Mean of maximum

This method resembles the 'maximum membership' method. However, the maximum membership degree may not be unique but a range of degrees, from which the mean value is derived:

$$R = \frac{x_a + x_b}{2}$$

where,

x_a is the first value reaching the highest membership degree of the class with the highest membership and

x_b is the last value with the highest membership degree of the class with the highest membership

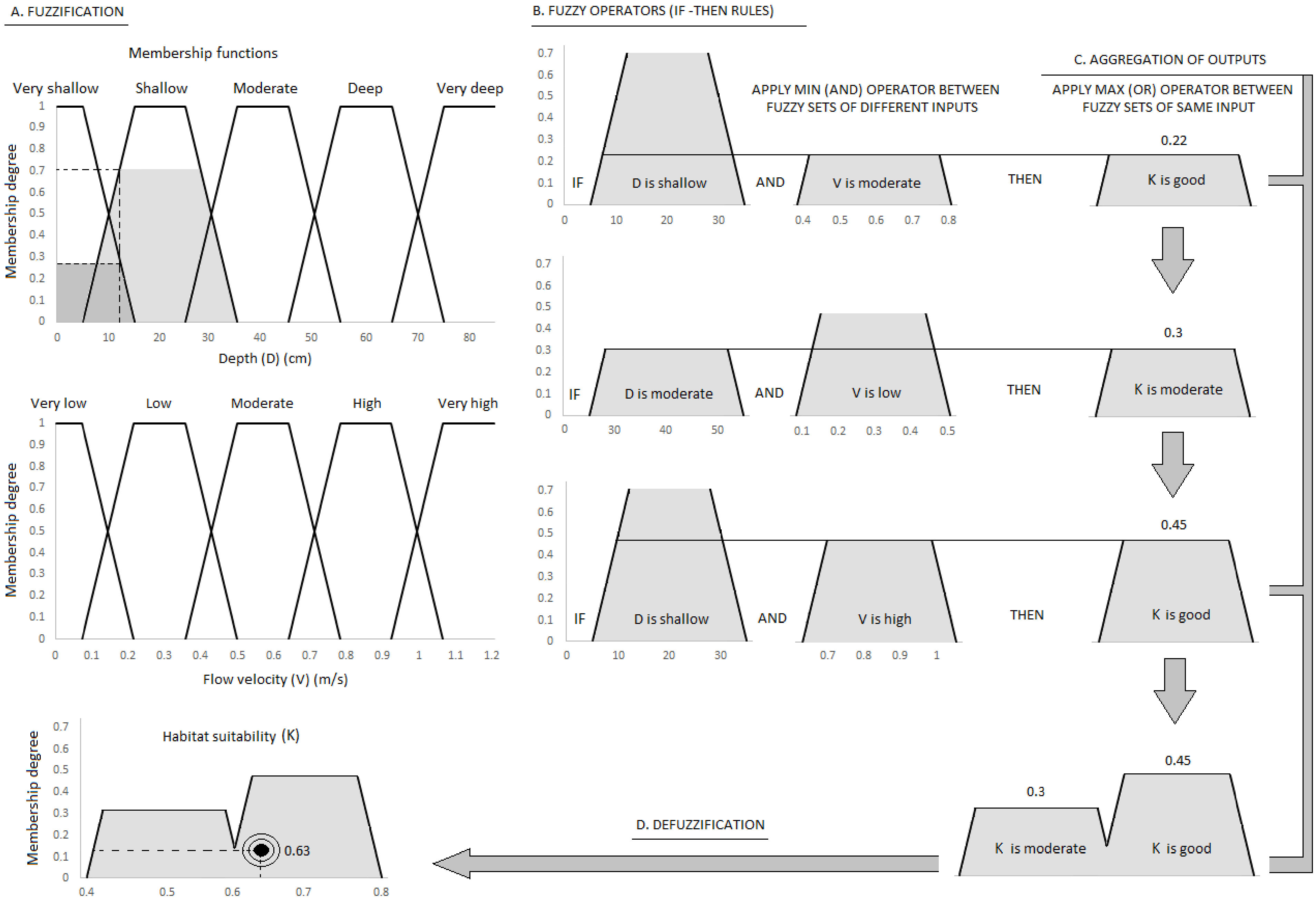

Fig. 1. The fuzzy logic algorithm used in a habitat modelling application. K: habitat suitability (response variable), V: flow velocity (predictor 1), D: water depth (predictor 2)

1.2. The fuzzy rule-based Bayesian algorithm

The fuzzy rule-based Bayesian algorithm uses the Bayesian joint probability and a classification system based on the 'expected utility' (described in Brookes et al., 2010) and can be summarized in three steps:

Step 1. Fuzzification of the input variables (predictors)

This step is the same as in the fuzzy algorithm and results in the conversion of crisp numerical values to fuzzy 'degrees of membership', ranging from 0 to 1 for each membership function (fuzzy set).

Step 2. Calculation of the Bayesian joint probability

The joint probability for interdependent events is calculated as

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$

where,

$P(A \cap B)$ is the probability of event A and event B occurring together

$P(A|B)$ is the conditional probability of event A occurring given the event B occurred

$P(B|A)$ is the conditional probability of event B occurring given the event A occurred

In this case, P1, P2, P3 and P4 are considered independent of each other and the joint probability is calculated by replacing P(A|B) with P(A).

In a habitat model …

… the joint probability for the flow velocity being 0.5 m/s and the water depth being 0.2 m, given their probabilities P(V:0.5 = 0.8) and P(D:0.2 = 0.3) is 0.8 x 0.3 = 0.24. HABFUZZ uses the fuzzy sets (fuzzified input values) from step 1 as probabilities of occurrence of each input. For example, a crisp V value of 0.28 m/s corresponds to the moderate fuzzy set by 0.2 membership degrees and to the low fuzzy set by 0.8 membership degrees. These values are used as the probabilities of occurrence of the moderate and low V sets.

Step 3. Classification of the response variable in classes

Classification is applied by using the 'expected utility' equation where each class of the response variable is scored and multiplied by the joint probability of occurrence of each class (sum of probability x score) as following:

$$EU\ (A) = \sum_{i=1}^{n} p(xi|A)U(xi)$$

where,

$EU(A)$ is the expected utility of action or event A

$P(x_i|A)$ is the probability of action or event A

$U(x_i)$ is a utility weight to convert a state to numerical values

In HABFUZZ …

… the response variable (output) is pre-defined as the habitat suitability and is already classified in five sets with a specific score as follows: [f_1: bad - U: 0.1, f_2: poor - U: 0.3, f3: moderate - U: 0.5, f4: good - U: 0.7, f5: high - U: 0.9]. These values are multiplied by the joint probability of occurrence of each habitat suitability class to develop the final prediction.

2. Dependencies

It is advised to install the GNU FORTRAN Compiler (download at https://gcc.gnu.org/wiki/GFortranBinaries) to quickly compile HABFUZZ through the relevant Windows and OS X files (however, experienced users may also use their preferred compilers). For Mac users, Xcode (download at https://developer.apple.com/xcode/) with its relevant Command Line Tools should be installed to enable compiling through the GNU FORTRAN Compiler.

3. Installing

HABFUZZ has been tested on Windows 10 - 32 bit and 64 bit operating systems, Ubuntu 16.04 and OS X 10.11 El Capitan (with Xcode 7.3.1 and Xcode 7.3.1. Command Line Tools), using the GNU FORTRAN Compiler. Depending on your operating system, follow the relevant instructions to run HABFUZZ.

3.1. Windows users

If the user needs to modify the source code of HABFUZZ, re-compilation is necessary. Using the GNU FORTRAN Compiler, you can either run the *wcompile.bat* file, or open a command window, navigate to the HABFUZZ subfolder and type the relevant commands:

```
gfortran -c fdeclarations.f95 ↵
gfortran -o habfuzz habfuzz.f95 fdeclarations.f95 reader.f95 classifier.f95 combinations.f95 ruler.f95 fuzzifier.f95 permutator.f95 rules2.f95 fuzzy.f95 fruler.f95 rules1.f95 centroid.f95 meanmax.f95 maxmem.f95 waver.f95 randomizer.f95 iterator.f95 iterator10.f95 tester.f95 ftester.f95 performance.f95 tencrossval.f95 ↵
del *.o ↵
del *.mod ↵
```

habfuzz.exe will then be replaced by the newly compiled one, being ready to run.

3.2. Linux users

Open the terminal and navigate to the HABFUZZ subfolder. If you don’t have the GNU FORTRAN Compiler, you need to be a root user (administrator) and type

```
sudo apt-get install gfortran ↵
```

to install the compiler. Having gfortran installed, the commands necessary to compile are the following:

```
gfortran -c fdeclarations.f95 ↵
gfortran habfuzz.f95 fdeclarations.f95 reader.f95 classifier.f95 combinations.f95 ruler.f95 fuzzifier.f95 permutator.f95 rules2.f95 fuzzy.f95 fruler.f95 rules1.f95 centroid.f95 meanmax.f95 maxmem.f95
```

waver.f95 randomizer.f95 iterator.f95 iterator10.f95 tester.f95 ftester.f95 performance.f95 tencrossval.f95–o habfuzz ↵

Be careful to write exactly the abovementioned commands, arranging the source files in the order given above. Then you can run HABFUZZ by typing:

./habfuzz ↵

3.3. Mac OS X users

You need to have Xcode installed together with the GNU FORTRAN Compiler and be a root user to enable compilation. Open the terminal and navigate to the HABFUZZ subfolder. To compile, you can either run the *mcompile.sh* file (which automatically applies the compilation commands) by typing:

./mcompile.sh ↵

or manually type the commands:

gfortran -c fdeclarations.f95 ↵

gfortran -o habfuzz habfuzz.f95 fdeclarations.f95 reader.f95 classifier.f95 combinations.f95 ruler.f95 fuzzifier.f95 permutator.f95 rules2.f95 fuzzy.f95 fruler.f95 rules1.f95 centroid.f95 meanmax.f95 maxmem.f95 waver.f95 randomizer.f95 iterator.f95 iterator10.f95 tester.f95 ftester.f95 performance.f95 tencrossval.f95↵

HABFUZZ can then be executed from the command line by typing

./habfuzz ↵

4. Usage

4.1. Input and output data

All input requirements of HABFUZZ are handled through a STEERING FILE, located in the steering folder (*Steering_file.xlsx*). To run HABFUZZ, you only need to prepare a train-data file and a test-data file. These files should be named traindata.txt and testdata.txt respectively and should be located at the same folder with *habfuzz.exe.*

1. traindata.txt

This file contains the data from which HABFUZZ will be trained to predict.
The file should have five columns where the four predictors -P1, P2, P3, P4- and the response variable R are stored, respectively, for each observation. Note that the first row should only contain one number describing the number of rows in the file (Fig. 2). If you don't need to include a predictor in the calculations (let's say P4), you can just assign a random, **but the same**, P4 value for all observations.

In HABFUZZ …
… the predictors are P1: flow velocity (V), P2: water depth (D), P3: substrate type (S), P4: water temperature (T) and the response variable R is the habitat suitability (K) ranging from 0 to 1. The user can change the pre-defined values if HABFUZZ is to be used in other applications (see section 4.2)

traindata.txt

```
10
0      0.19   0.05   11.9   0.274
0      0.18   0.04   21.1   0.286
1.576  0.27   0.05   11.8   0.292
0.485  0.87   0.03   12.2   0.295
1.08   0.26   0.05   11.1   0.297
0.3    0.34   0.05   12.1   0.348
0.14   0.16   0.03   11.9   0.35
0      0.06   0.07   16.8   0.361
0.19   0.64   0.05   18.8   0.369
0.75   0.19   0.04   22.1   0.369
```

First row indicates the total number of rows in the file

All other rows: First four columns contain the predictors. In HABFUZZ, second row: 0, 0.19, 0.05 and 11.9 are the values of V, D, S and T, respectively. For S values see Table 2.

All other rows: Fifth column contains the known response variable (in this case K)

Fig. 2. traindata.txt file example

2. testdata.txt

This file contains the data with unknown R values, which are to be predicted by HABFUZZ based on the training dataset. This file should have the same format as the traindata.txt but without an R column (Fig. 3). Again, if you do not wish to use P4, the P4 column should be there but with the same P4 value as the traindata.txt for all observations.

testdata.txt

```
20
0      0.16   0.04    11.6
0.1    0.35   0.07    14.6
0      0.19   0.03    14.8
0.11   0.42   0.07    17.9
0.08   0.24   0.07    22.3
0      0.07   0.022   12.2
0.05   0.14   0.024   16.8
0      0.76   0.04    21.6
0.802  0.33   0.04    11.8
0.09   0.48   0.026   12.2
0.88   0.34   0.03    23
0.1    0.52   0.04    14.9
0.006  0.24   0.024   12.5
0      0.21   0.07    11.6
0      0.03   0.022   14.8
1.444  0.53   0.05    12.5
1.42   0.42   0.05    12.3
0.15   0.39   0.03    11.9
0.95   0.51   0.07    13
0      0.19   0.05    11.9
```

testdata.txt: the same format as traindata.txt but without the response-variable column

Fig. 3. testdata.txt file example

You can use the STEERING FILE to prepare the testdata.txt …
Go to the 'steering' folder, open the *Steering_file.xlsx* and navigate to the 'prefuzz' tab. Paste the values for each predictor based on your dataset (note again that PREDICTOR 3 is not fuzzified). Then go to the 'HABFUZZ_testdata.txt' tab and save it in the habfuzz folder as testdata.txt.

The output of HABFUZZ is a file named output.txt containing a single column with all the predicted values of the response variable calculated for each input observation in the same order as with the input file (testdata.txt) and a log.txt file with the internal parameters of the prediction process. Both files are placed by the program in the HABFUZZ subfolder.

In HABFUZZ …
During the fuzzy inference process, K is initially a combination of fuzzy sets (five classes of habitat suitability - bad, poor, moderate, good, high) and through the defuzzification process it is converted into a crisp output ranging from 0 to 1. The inputs and the output suitability of HABFUZZ are depicted in Fig. 1. In the fuzzy rule-based Bayesian algorithm, K is expressed using the same five classes. Each class is assigned with a utility score (bad - 0.1, poor - 0.3, moderate - 0.5, good - 0.7, high - 0.9) and multiplied by the joint probability of each combination observed.

Table 2. Manning's n for various substrate types used in HABFUZZ

Bed material	Size (diameter)	Manning's n
Boulders	>25 cm	0.070
Large stones	12-25 cm	0.050
Small stones	6-12 cm	0.040
Large gravel	2-6 cm	0.030
Medium gravel	0.6-2 cm	0.026
Fine gravel	0.2-0.6 cm	0.024
Sand	<0.2 cm	0.022
Silt	-	0.020

4.2. Adjustment of the fuzzy-set parameters

The algorithms of HABFUZZ, either the fuzzy logic or the fuzzy rule-based Bayesian algorithm, require user-pre-defined fuzzy set classes for the predictors and the response variable. Adjustment of the fuzzy set parameters for each predictor is applied in HABFUZZ from the STEERING FILE, located in the 'steering' folder. Open the STEERING FILE (*Steering_file.xlsx*), go to the 'Fuzzy sets' tab and change the relevant parameters according to your requirements. You can see that each change you make is graphically illustrated in the relevant chart of the tab. When finished adjusting, go to the 'HABFUZZ_fuzzysets.txt' tab and save this tab as 'fuzzysets.txt' (tab delimited) in the habfuzz folder. The calculation of the IF-THEN rules is then adapted to the new fuzzy sets defined by the user. That's all you need for running HABFUZZ.

Note 1 …
The values for predictor 3 (P3) are not fuzzified. If you do not wish to include a non-fuzzified variable enter the same P3 value (e.g. 0.07) for all observations in you traindata.txt and testdata.txt files.

Note 2 - You can use the STEERING FILE to add expert judgment rules …
This works only for habitat modelling and does not include water temperature. Go to the 'steering' folder, open the *Steering_file.xlsx*, go to the 'Expert_judgment_rules' tab and adjust the habitat suitability values in the relevant column. Then go to the 'HABFUZZ_expert_traindata.txt' tab and save it in the habfuzz folder as traindata.txt.

For reference only …
The names of each fuzzy-set parameter in the source code of HABFUZZ are shown in Table 3 and Fig. 4, 5 and 6.

Table 3. The parameters that define each class interval for each predictor (P1, P2, P4) and for the response variable. These parameters need to be modified by the user if new classes are to be defined. See also Fig. 4, 5 and 6.

	P1	P2	P3	P4	Response
Class 1	uvla, uvlb	dvla, dvlb	NF	tvla, tvlb	ka, eua
Class 2	ula, ulb, ulc, uld	dla, dlb, dlc, dld	NF	tla, tlb, tlc, tld	kb, eub
Class 3	uma, umb, umc, umd	dma, dmb, dmc, dmd	NF	tma, tmb, tmc, tmd	kc, euc
Class 4	uha, uhb, uhc, uhd	dda, ddb, ddc, ddd	NF	tha, thb, thc, thd	kd, eud
Class 5	uvha, uvhb	dvda, dvdb	NF	tvha, tvhb	ke, eue

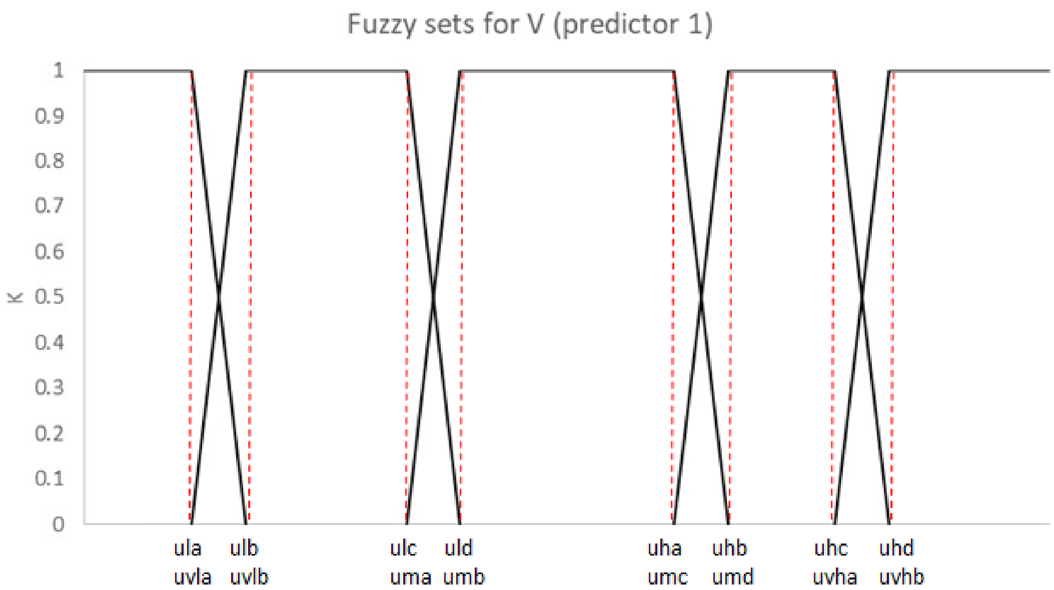

Fig. 4. Fuzzy sets for the first predictor variable (V: flow velocity in this example)

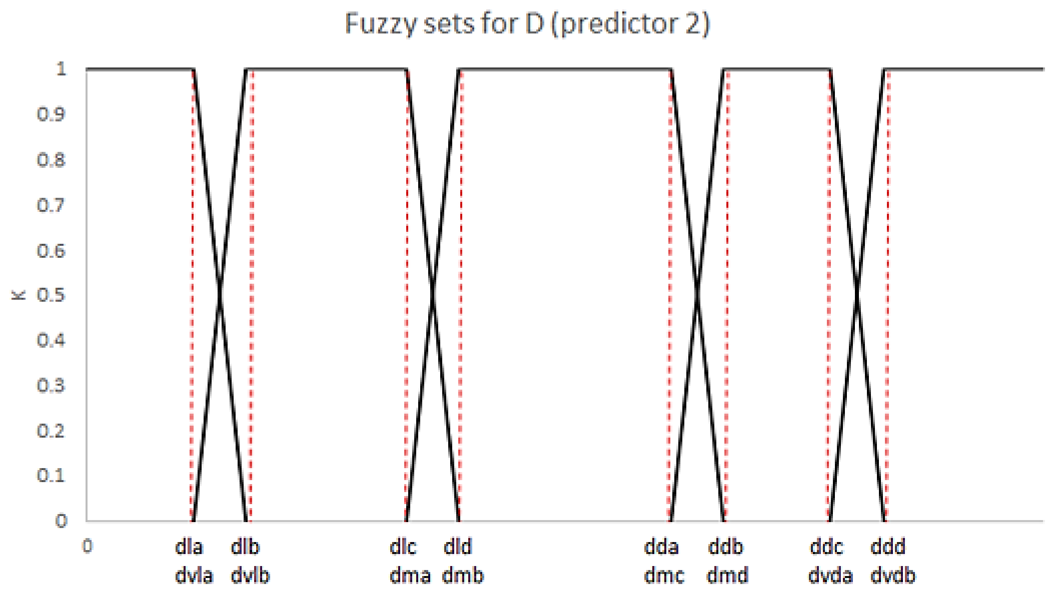

Fig. 5. Fuzzy sets for the second predictor variable (D: water depth in this example)

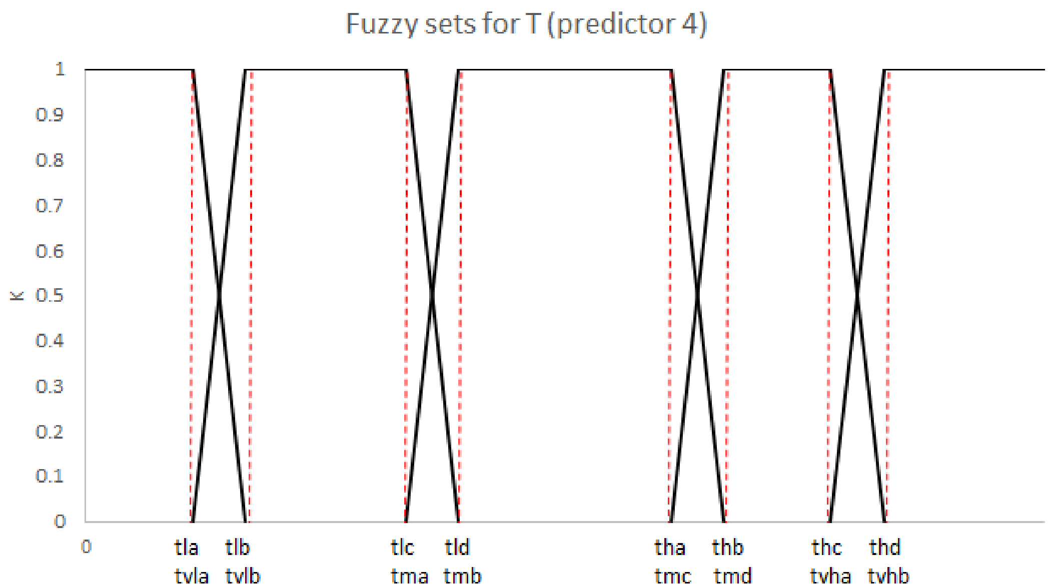

Fig. 6. Fuzzy sets for the fourth predictor variable (T: water temperature in this example)

Currently, in HABFUZZ 2.5, response variables outside the 0-1 range can be appropriately modelled only using the fuzzy rule-based Bayesian algorithm. We are working hard on expanding this option to the other algorithms.

4.3. Running HABFUZZ

(As a habitat model using the example dataset of Table 1)

After having the input files ready, run the program. The command window opens and after a short welcome message the software asks for the inference process to be implemented (Fig. 7 and 8).

Fig. 7. HABFUZZ welcome screen (v2.5)

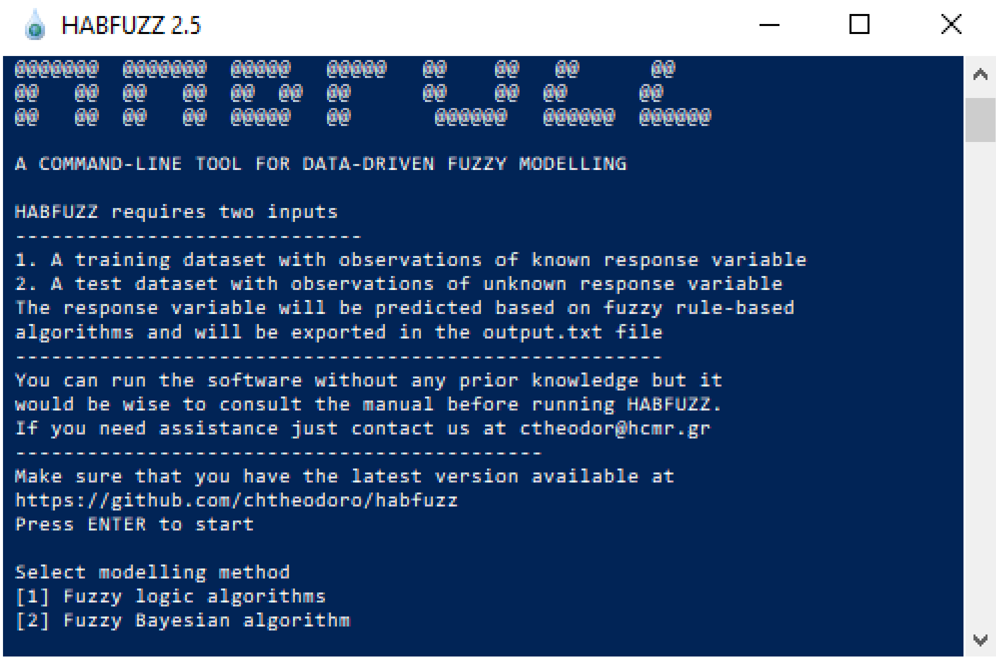

Fig. 8. Selection of modelling algorithm in HABFUZZ (v2.5)

Note again that the only essential information required to run HABFUZZ is in sections 4.1. and 4.2. The following information is here to help you understand how the program predicts the output.

After selecting the modelling algorithm, the user is asked to select the cross validation scheme. For the fuzzy rule-based Bayesian algorithm, two options are available, (i) Monte Carlo and (ii) ten-fold (Fig. 9). For the fuzzy-logic algorithms, only Monte Carlo cross validation is currently supported (Fig. 10). The cross-validation process can be bypassed by the user and directly proceed with modelling the test dataset by selecting ‘Do not cross-validate’. In the ten-fold cross validation, the initial dataset is randomly partitioned in ten equal-sized subsamples. Nine subsamples are used as the training dataset and the remaining subsample is used for model validation. This process is repeated ten times (folds), using a different subsample for validation at each iteration. The Monte Carlo scheme also includes ten iterations but at each iteration the initial dataset is randomly partitioned in two subsamples. The first subsample contains 90% of the initial data and is used for calibration, and the second subsample contains the remaining 10% and is used for validation. At each iteration, the initial data is again randomly partitioned and thus the same data may be randomly included in each subsample more than once, in contrast to the ten-fold cross validation scheme. The performance of each model is evaluated as the average percentage of the correctly classified instances (CCI) between each iteration of the ten-fold cross-validation process.

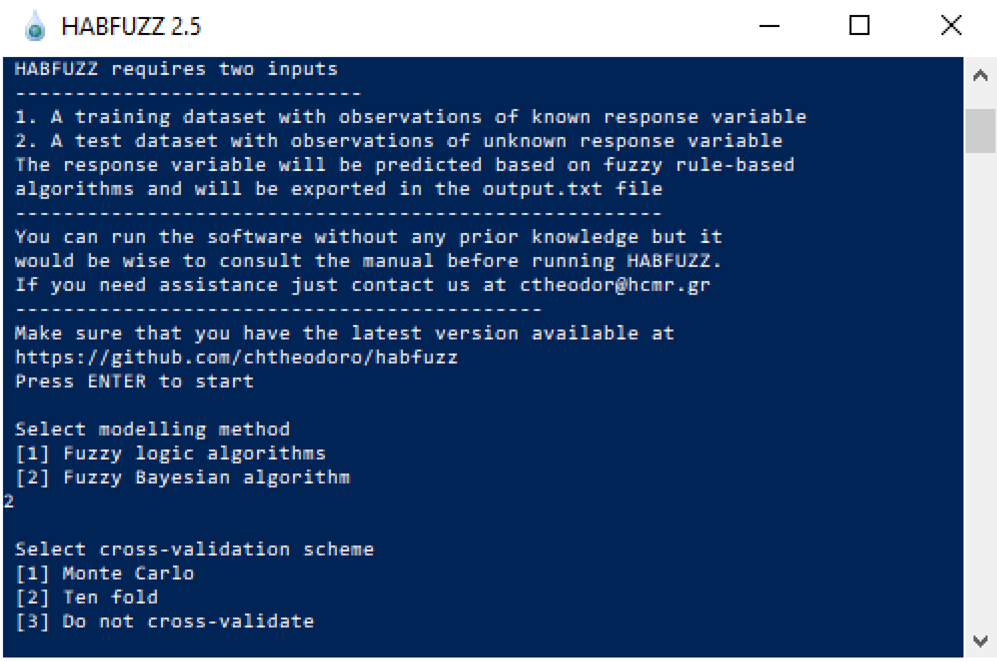

Fig. 9. Selection of the cross-validation method in the fuzzy rule-based Bayesian algorithm of HABFUZZ

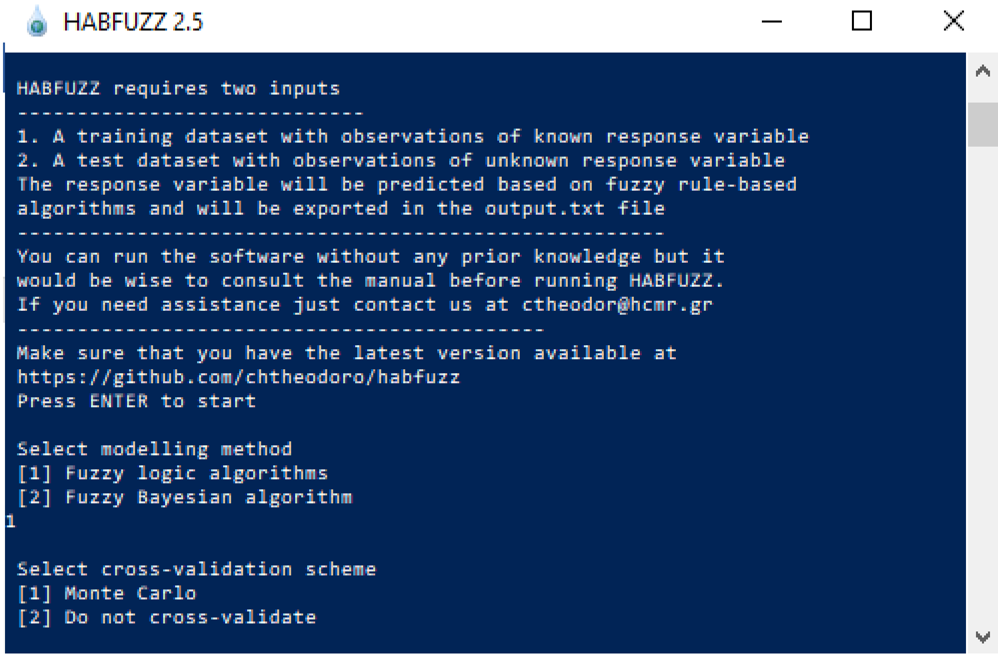

Fig. 10. Selection of the cross-validation method in the fuzzy logic algorithms of HABFUZZ

4.2.1. The fuzzy logic algorithm

If the fuzzy algorithm is selected, the program, after selecting the cross-validation scheme, prompts the user to select the desired management scenario to implement (Fig. 11). There are three scenarios available based on the method used for determining the outcome of each IF-THEN rule from the reference conditions of the program, (i) the average scenario, where the different R values for the same combinations of P1, P2 and P3 (and optionally P4) are averaged to derive the final R, (ii) the worst scenario, where the final R is derived from the minimum observed R and (iii) the optimum scenario where the final R is derived by the maximum observed R. A default scenario is also present (the moderate scenario).

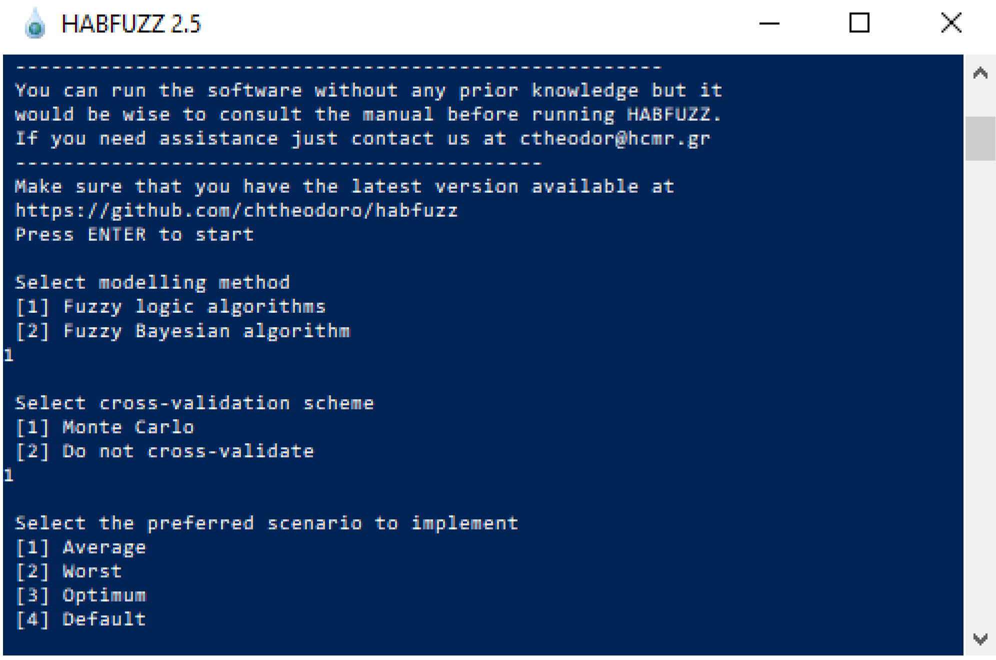

Fig. 11. Selection of the desired scenario in HABFUZZ

After selecting the desired scenario, the user is asked to select the defuzzification method (see section 1) (Fig. 12). A default method (centroid) is available. After selecting the defuzzification method, HABFUZZ calls the relevant subroutines to perform the tasks selected. The program informs the user when the process is completed and indicates the output.txt file created where the values of the response variable are stored and the log.txt file with the fuzzy membership degrees for each fuzzy set. Both files are located in the HABFUZZ subfolder.

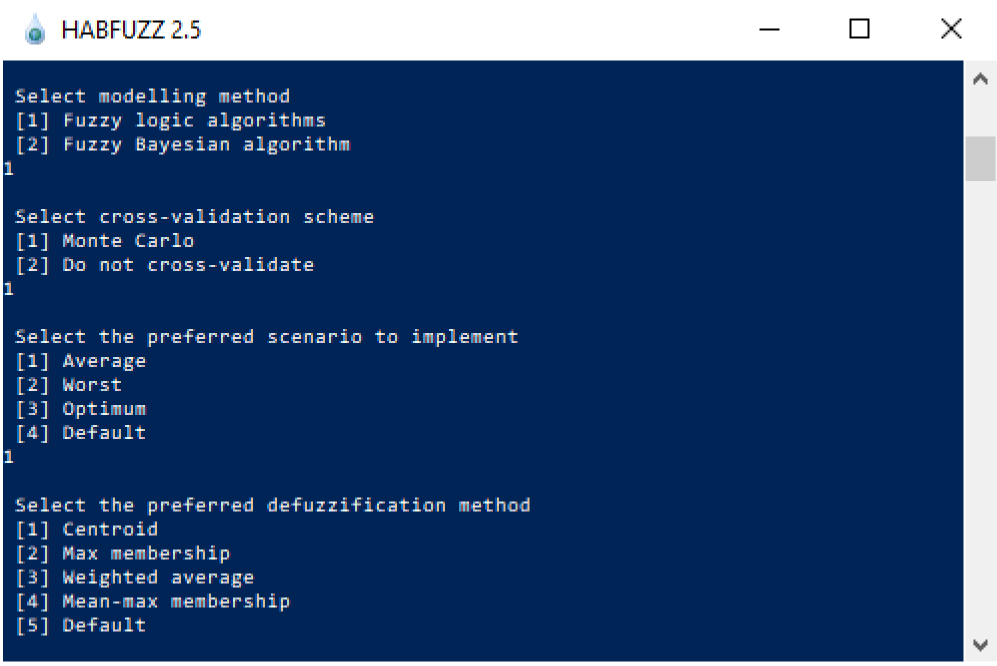

Fig. 12. Selection of the defuzzification method in HABFUZZ

Using the example data of Table 1, the crisp input values for P1: V and P2: D, are fuzzified as depicted in Table 4.

Table 4. Fuzzification results for the example dataset. Different samples-observations are shaded differently.

	Fuzzy membership classes						
Crisp inputs	V Very Low	V Low	V Moderate	D Very Shallow	D Shallow	D Moderate	D Very Deep
V = 0.28	0	0.8	0.2				
V = 0.05	1	0	0				
V = 0.46	0	0	1				
D = 0.29				0	0.333	0.667	0
D = 0.08				0.333	0.667	0	0
D = 0.80				0	0	0	1

V: Flow velocity, D: Water depth

HABFUZZ then checks each combination of inputs (their corresponding fuzzy sets) and assigns a membership degree using the AND (min) operator to the relevant R class (in this case K) they belong according to the IF-THEN rules calculated by the program itself, based on the training dataset, and depending on the selected management scenario. Let's assume that the user has chosen the moderate scenario. The membership degree of each combination to the K classes (including S) is depicted in Table 5.

Table 5. Checking the relevant IF-THEN rules and assigning membership degrees to the suitability class by applying the AND (min) operator.

V	D	S	K
Moderate (0.2)	Moderate (0.667)	Boulders (1)	-
Moderate (0.2)	Shallow (0.333)	Boulders (1)	High (0.2)
Low (0.8)	Moderate (0.667)	Boulders (1)	Moderate (0.667)
Low (0.8)	Shallow (0.333)	Boulders (1)	Good (0.333)
Very Low (1)	Very Shallow (0.667)	Large stones (1)	Good (0.667)
Very Low (1)	Shallow (0.333)	Large stones (1)	Good (0.333)
Moderate (1)	Very Deep (1)	Small stones (1)	-

V: Flow velocity, D: Water depth, S: Substrate type, K: Habitat suitability

HABFUZZ then combines the same K classes observed (aggregation step) using the OR (max) operator and the different membership degrees of all classes observed are defuzzified using one of the methods described in section 1. The results of the aggregation and defuzzification processes (in this case we have chosen the centroid defuzzification method) are depicted in Table 6.

Table 6. Aggregation of outputs using the OR (max) operator. It can be seen that observation 3 is not referred in the IF-THEN rules and a value of -1 is returned by HABFUZZ.

Observation	V	D	S	K
1	0.28	0.29	Boulders	0.622
2	0.05	0.08	Large stones	0.700
3	0.46	0.80	Small stones	-1

V: Flow velocity, D: Water depth, S: Substrate type, K: Habitat suitability

4.2.2. The fuzzy Bayesian algorithm

If the fuzzy Bayesian algorithm is selected, the program immediately calculates the response variable K according to the steps described previously and outputs two *.txt* files, the output.txt and the log.txt with the same contents as in the fuzzy inference process. Again, using the example data of Table 1, the crisp input values for V and D, are fuzzified as depicted in Table 4. The process then treats the fuzzified membership degrees as the probability of each observation occurring, suggesting for example that *the probability of K being high is the joint probability that V is moderate, D is shallow and S is boulders*. In the example dataset, this concept is shown for each observation in Tables 7, 8 and 9.

Table 7. (A) The joint probability table for the fuzzified inputs of observation 1 (S=Boulders, not shown but included). (B) Joint probability after including the probability of the habitat suitability (not shown) class for each combination.

<table>
<tr><th>(A) Observation 1</th><th colspan="3">D (P)</th></tr>
<tr><td>V (P)</td><td>Shallow (0.333)</td><td>Moderate (0.667)</td><td rowspan="3">JP = P(V) x P(D) x P(S) Substrate's P is always 1 since S is not fuzzified.</td></tr>
<tr><td>Low (0.8)</td><td>0.2664</td><td>0.5336</td></tr>
<tr><td>Moderate (0.2)</td><td>0.0666</td><td>0.1334</td></tr>
</table>

Table 7. (continued).

(B) Observation 1	D (P)						
V (P)	Shallow (0.333)				Moderate (0.667)		JP = P(V) x P(D) x P(S) x P(K)
Low (0.8)	0.1455	0.0729	0.0239	0.0239	0.2668	0.2668	
Moderate (0.2)	0.0667				-		

V: Flow velocity, D: Water depth, S: Substrate type, JP: Joint probability, K: Habitat suitability; Blue colour: High K, Green colour: Good K; Yellow colour: Moderate K, Red: Bad K

Table 8. (A) The joint probability table for the fuzzified inputs of observation 2 (S=Large stones, not shown but included). (B) Joint probability after including the probability of the habitat suitability class for each combination. JP: Joint probability.

(A) Observation 2	D (P)		
V (P)	Very Shallow (0.667)	Shallow (0.333)	JP = P(V) x P(D) x P(S)
Low (1)	0.667	0.333	

(B) Observation 2	D (P)							
V (P)	Very Shallow (0.667)			Shallow (0.333)				Joint Probability = P(V) x P(D) x P(S) x P(K)
Very Low (1)	0.1667	0.3335	0.1667	0.0639	0.1412	0.1022	0.0256	

V: Flow velocity, D: Water depth, S: Substrate type, JP: Joint probability, K: Habitat suitability, Blue colour: High suitability, Green colour: Good suitability, Yellow: Moderate suitability, Orange: Poor suitability

Table 9. Joint probability table for the fuzzified inputs of observation 2 (S=Small stones, not shown but included). Since the specific combination is nor present in the rules.f95 file, no further calculations are applied.

Observation 3	D (P)	
V (P)	Very Deep (1)	Joint Probability = P(V) x P(D) x P(S)
Moderate (1)	1	

V: Flow velocity, D: Water depth, S: Substrate type, JP: Joint probability

HABFUZZ then assigns a score at each K class to calculate the final K output (bad - 0.1, poor - 0.3, moderate - 0.5, good - 0.7, high - 0.9), using the 'expected utility' equation. Each probability from tables 6B and 7B is multiplied by the score of each relevant suitability class and all products are summed (for each observation) to derive the final habitat suitability. The results of the fuzzy Bayesian inference process are presented in Table 10.

Table 10. The fuzzy Bayesian calculation of habitat suitability using the 'expected utility (EU)' equation

Observations	Joints probability combinations							EU
1 -->	0.1455 x 0.9	0.0729 x 0.7	0.0239 x 0.5	0.0239 x 0.1	0.2668 x 0.7	0.2668 x 0.5	0.0667 x 0.9	0.577
2 -->	0.1667 x 0.9	0.3335 x 0.7	0.1667 x 0.5	0.0639 x 0.9	0.1412 x 0.7	0.1022 x 0.5	0.0256 x 0.3	0.677

A short scheme of the HABFUZZ concept is shown in Fig. 13.
You can also find helpful information in the online video tutorial at https://www.youtube.com/watch?v=ZA_NADMyMsM

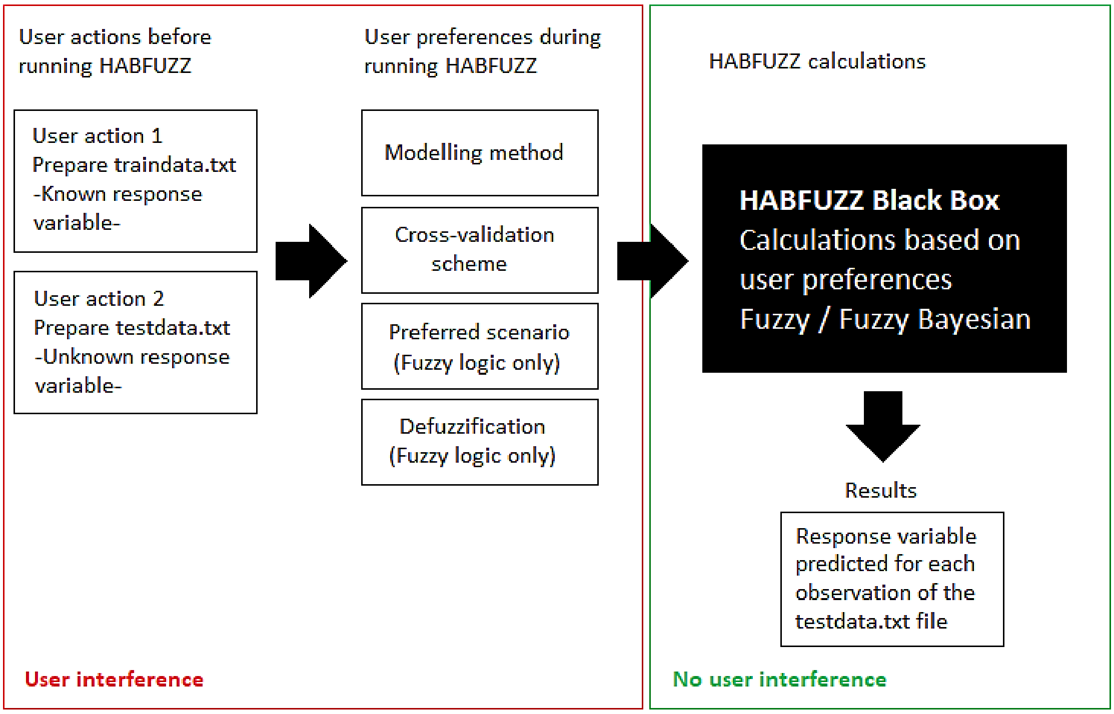

Fig. 13. Schematic representation of the HABFUZZ processes

CPU, memory and time ...
The fuzzy logic algorithms may be very time-, CPU- and memory-resources-consuming. For low-performance machines, we suggest the application of the fuzzy Bayesian algorithm, which yields accurate predictions in less time.

The STEERING FILE ...
... is only here to help you. You can create all the .txt files necessary for HABFUZZ by your own and this is also not difficult to do. However, using the STEERING FILE will save you time and effort.

References

Brookes C.J., Kumar V. and Lane S.N. 2010. A comparison of Fuzzy, Bayesian and Weighted Average formulations of an in-stream habitat suitability model. Proceedings of the International Congress on Environmental Modelling and Software, 5-8 Jul 2010, Ottawa, Canada. Available at http://www.iemss.org/iemss2010/papers/S20/S.20.07.Model%20selection%20and%20uncertainty%20A%20comparison%20of%20Fuzzy,%20Bayesian%20and%20Weighted%20Average%20formulations%20of%20an%20instream%20habitat%20suitability%20model%20-%20CHRISTOPHER%20BROOKES.pdf

Mamdani E.H. and Assilian S. 1975. An experiment in linguistic synthesis with a fuzzy logic

Controller. International Journal of Man-Machine Studies 7: 1-13.

Ross T.J. 2010. Fuzzy logic with engineering applications. Third Edition, John Wiley and Sons, UK.

Zadeh L.A. 1965. Fuzzy sets. Information and Control 8: 338–353.

APPENDIX

Manning's n for channels (Chow, 1959)

Type of Channel and Description	Minimum	Normal	Maximum
Natural streams - minor streams (top width at floodstage < 100 ft)			
1. Main Channels			
a. clean, straight, full stage, no rifts or deep pools	0.025	0.030	0.033
b. same as above, but more stones and weeds	0.030	0.035	0.040
c. clean, winding, some pools and shoals	0.033	0.040	0.045
d. same as above, but some weeds and stones	0.035	0.045	0.050
e. same as above, lower stages, more ineffective slopes and sections	0.040	0.048	0.055
f. same as "d" with more stones	0.045	0.050	0.060
g. sluggish reaches, weedy, deep pools	0.050	0.070	0.080
h. very weedy reaches, deep pools, or floodways with heavy stand of timber and underbrush	0.075	0.100	0.150
2. Mountain streams, no vegetation in channel, banks usually steep, trees and brush along banks submerged at high stages			
a. bottom: gravels, cobbles, and few boulders	0.030	0.040	0.050
b. bottom: cobbles with large boulders	0.040	0.050	0.070
3. Floodplains			
a. Pasture, no brush			
1.short grass	0.025	0.030	0.035
2. high grass	0.030	0.035	0.050
b. Cultivated areas			
1. no crop	0.020	0.030	0.040
2. mature row crops	0.025	0.035	0.045
3. mature field crops	0.030	0.040	0.050
c. Brush			
1. scattered brush, heavy weeds	0.035	0.050	0.070
2. light brush and trees, in winter	0.035	0.050	0.060
3. light brush and trees, in summer	0.040	0.060	0.080
4. medium to dense brush, in winter	0.045	0.070	0.110
5. medium to dense brush, in summer	0.070	0.100	0.160
d. Trees			
1. dense willows, summer, straight	0.110	0.150	0.200

2. cleared land with tree stumps, no sprouts	0.030	0.040	0.050
3. same as above, but with heavy growth of sprouts	0.050	0.060	0.080
4. heavy stand of timber, a few down trees, little undergrowth, flood stage below branches	0.080	0.100	0.120
5. same as 4. with flood stage reaching branches	0.100	0.120	0.160
4. Excavated or Dredged Channels			
a. Earth, straight, and uniform			
1. clean, recently completed	0.016	0.018	0.020
2. clean, after weathering	0.018	0.022	0.025
3. gravel, uniform section, clean	0.022	0.025	0.030
4. with short grass, few weeds	0.022	0.027	0.033
b. Earth winding and sluggish			
1. no vegetation	0.023	0.025	0.030
2. grass, some weeds	0.025	0.030	0.033
3. dense weeds or aquatic plants in deep channels	0.030	0.035	0.040
4. earth bottom and rubble sides	0.028	0.030	0.035
5. stony bottom and weedy banks	0.025	0.035	0.040
6. cobble bottom and clean sides	0.030	0.040	0.050
c. Dragline-excavated or dredged			
1. no vegetation	0.025	0.028	0.033
2. light brush on banks	0.035	0.050	0.060
d. Rock cuts			
1. smooth and uniform	0.025	0.035	0.040
2. jagged and irregular	0.035	0.040	0.050
e. Channels not maintained, weeds and brush uncut			
1. dense weeds, high as flow depth	0.050	0.080	0.120
2. clean bottom, brush on sides	0.040	0.050	0.080
3. same as above, highest stage of flow	0.045	0.070	0.110
4. dense brush, high stage	0.080	0.100	0.140
5. Lined or Constructed Channels			
a. Cement			
1. neat surface	0.010	0.011	0.013
2. mortar	0.011	0.013	0.015
b. Wood			
1. planed, untreated	0.010	0.012	0.014
2. planed, creosoted	0.011	0.012	0.015
3. unplanned	0.011	0.013	0.015
4. plank with battens	0.012	0.015	0.018
5. lined with roofing paper	0.010	0.014	0.017

c. Concrete			
1. trowel finish	0.011	0.013	0.015
2. float finish	0.013	0.015	0.016
3. finished, with gravel on bottom	0.015	0.017	0.020
4. unfinished	0.014	0.017	0.020
5. gunite, good section	0.016	0.019	0.023
6. gunite, wavy section	0.018	0.022	0.025
7. on good excavated rock	0.017	0.020	
8. on irregular excavated rock	0.022	0.027	
d. Concrete bottom float finish with sides of:			
1. dressed stone in mortar	0.015	0.017	0.020
2. random stone in mortar	0.017	0.020	0.024
3. cement rubble masonry, plastered	0.016	0.020	0.024
4. cement rubble masonry	0.020	0.025	0.030
5. dry rubble or riprap	0.020	0.030	0.035
e. Gravel bottom with sides of:			
1. formed concrete	0.017	0.020	0.025
2. random stone mortar	0.020	0.023	0.026
3. dry rubble or riprap	0.023	0.033	0.036
f. Brick			
1. glazed	0.011	0.013	0.015
2. in cement mortar	0.012	0.015	0.018
g. Masonry			
1. cemented rubble	0.017	0.025	0.030
2. dry rubble	0.023	0.032	0.035
h. Dressed ashlar/stone paving	0.013	0.015	0.017
i. Asphalt			
1. smooth	0.013	0.013	
2. rough	0.016	0.016	
j. Vegetal lining	0.030		0.500