Resampling Fuzzy Numbers with Statistical Applications: FuzzyResampling Package

by Maciej Romaniuk, Przemysław Grzegorzewski

Abstract

The classical bootstrap proves its usefulness in many areas of statistical inference. However, some shortcomings of this method are also known. Therefore, various bootstrap modifications and other resampling algorithms were introduced, especially in the case of real-valued data. Recently, bootstrap methods have become popular in statistical reasoning based on imprecise data often modeled by fuzzy numbers. One of the challenges faced there is to create bootstrap samples of fuzzy numbers which are similar to initial fuzzy samples but "different" at the same time. These methods are implemented in **FuzzyResampling** package and applied in different statistical functions like single-sample or two-sample tests for the mean. Besides describing the aforementioned functions, some examples of their applications as well as numerical comparisons of the classical bootstrap with the new resampling algorithms are provided in this contribution.

Introduction

The bootstrap introduced by (Efron, 1979) is one of the most important contemporary achievements of computational statistics. It proves its usefulness in various fields, like in estimating standard errors, computing confidence intervals, or hypothesis testing, especially if the actual population distribution is unknown or the desired statistical procedures are not tractable analytically, and when the sample size is not large enough to apply asymptotic methods (see, e.g, (Efron and Tibshirani, 1993)). Then Giné and Zinn (Giné and Zinn, 1990) developed a bootstrapped approximation to the Central Limit Theorem for generalized random elements which allowed to apply bootstrap for fuzzy random variables used for modeling imprecise outputs of random experiments. This very application is of extreme importance because the absence of suitable models for the distribution of fuzzy random variables makes the statistical reasoning really hard. Practice has shown that the use of the bootstrap to handle fuzzy data has been very effective (see, e.g., Gil et al. (2006); González-Rodríguez et al. (2006); Montenegro et al. (2004); Lubiano et al. (2016, 2017)).

An important disadvantage of the classical bootstrap, easily spotted especially for small sample sizes, is that the bootstrap samples almost always contain repeated values and hence their variability is lower when compared to the initial sample. This is all the more troublesome when the random distribution of the assumed statistical model is continuous, since this feature is not properly reflected with the classical bootstrap where each value has the same non-zero probability of appearing in the bootstrap sample. To overcome this problem many modifications of Efron's approach were proposed in the literature, like the smoothed bootstrap (see (Silverman and Young, 1987)). The same shortcomings and remarks apply to bootstrap methods applied in a fuzzy environment. Here would be also desirable to generate samples consisting of fuzzy observations that are almost identical to those contained in the original sample, but nevertheless to some extent different from them. Several methods have been proposed to deal with this problem (Grzegorzewski and Romaniuk, 2022b). One can divide them into two main groups. The first one comes down to the generation of new fuzzy numbers which keep some "characteristic points" of the membership functions describing initial fuzzy values. For the second group, we are interested in the preservation of some basic characteristics of the initial values like their canonical representation (Delgado et al., 1998). Both approaches seem to be fruitful in many aspects of the statistical inference (see (Grzegorzewski et al., 2019, 2020b,a; Grzegorzewski and Romaniuk, 2022b; Romaniuk, 2019; Romaniuk and Hryniewicz, 2019)) and may be helpful in improving the results of some real-life data analyses.

To allow potential users access to the aforementioned new bootstrap methods several resampling algorithms have been implemented in **FuzzyResampling** package. They have been complemented by some additional procedures which might be useful for the generation of samples of trapezoidal fuzzy numbers, for estimation of the standard error (or the MSE), and for testing hypotheses about the fuzzy mean (in the single or two samples problem), etc. The proposed package can be applied together with other tools designed for data analysis. **FuzzyResampling** is, to our best knowledge, the first package in R which provides a few resampling methods for trapezoidal fuzzy numbers together with some statistical functions based on them.

In the following, we briefly compare **FuzzyResampling** package with the existing ones, introduce a necessary notation and describe the new resampling methods. Functions implemented in the package are illustrated with examples. Finally, the proposed resampling methods are compared with the classical bootstrap in some important statistical problems (both theoretical and utilizing real-life data).

Related packages

There is an abundance of packages oriented on the classic bootstrap and its statistical applications, like **boot** with many functions and data originated from (Davison and Hinkley, 1997) including the parametric and nonparametric resampling approaches or **bootstrap** containing functions and data related to Efron and Tibshirani (1993), e.g. cross-validation or jackknife procedures.

There are also many R packages intended to handle fuzzy numbers that may be useful in statistical reasoning with fuzzy data. Let us mention, e.g., **FuzzyNumbers** for computing arithmetic operators, designing approximations, and finding possibility and necessity values for arbitrary fuzzy numbers or some of their special types. **FuzzySTs** might be helpful for fuzzification, calculation of different distances, numerical estimations of fuzzy statistical measures and bootstrap distribution of the likelihood ratio, etc. Such packages like **SAFD** (see also (Trutschnig et al., 2013)) or **Sim.PLFN** (see also (Coroianu et al., 2013)) are devoted to simulation on fuzzy numbers.

However, to our best knowledge, **FuzzyResampling** is the only package that provides the new resampling methods, different than Efron's classical approach, for trapezoidal fuzzy numbers and let apply them to various problems of statistical inference. In particular, **SAFD** provides procedures for the bootstrapped versions of the one- or two-sample tests for the mean of trapezoidal fuzzy numbers (which are related to the C-test mentioned in this paper) but they are limited to the classical bootstrap. Although it contains a function that generates "stochastically perturbated" new values based on the input data frame but it only adds some random noise without keeping track of important characteristics of the input fuzzy numbers (as it is done in the resampling methods implemented in **FuzzyResampling**).

Fuzzy numbers – notation

For a more detailed introduction to the theory of fuzzy numbers, we refer the reader to (Ban et al., 2015). In the following, we provide only some necessary concepts and notation.

A fuzzy number A is characterized by its membership function $A(x)$ which represents the degree of membership of x into A. We assume that $0 \leqslant A(x) \leqslant 1$, where $A(x) = 1$ indicates that x surely belongs to A, while $A(x) = 0$ means that x surely does not belong to A. If $A(x) \in (0,1)$ then we have a partial belongingness of x into A.

Perhaps the most often used fuzzy numbers are known as the **trapezoidal fuzzy numbers** (denoted further as TPFNs) with their membership functions given by

$$A(x) = \begin{cases} \frac{x-a_1}{a_2-a_1} & \text{if } a_1 < x \leqslant a_2, \\ 1 & \text{if } a_2 \leqslant x \leqslant a_3, \\ \frac{a_4-x}{a_4-a_3} & \text{if } a_3 \leqslant x < a_4, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $a_1, a_2, a_3, a_4 \in \mathbb{R}$, and $a_1 \leqslant a_2 \leqslant a_3 \leqslant a_4$. Here, interval $[a_2, a_3]$, called the core, contains all values which are fully compatible with the concept described by the fuzzy number A, while the interval $[a_1, a_4]$, called the support, indicates all real values that are compatible to some extent with A. If A is a TPFN it can be described completely using only four real values a_1, a_2, a_3, a_4, so we can write $A = (a_1, a_2, a_3, a_4)$. If $a_2 = a_3$ then we have a **triangular fuzzy number** (denoted further as TRFN), and $A = (a_1, a_2, a_4)$.

Another parametrization of a trapezoidal fuzzy number, which is useful, especially for generating fuzzy data and simulation studies, is given by

$$c = \frac{a_2 + a_3}{2}, \; s = \frac{a_3 - a_2}{2}, \; l = a_2 - a_1, \; r = a_4 - a_3, \tag{2}$$

where c indicates the center of the core of a fuzzy number, s is equal to its core radius (the spread of the core), l and r stand for the spread of the left and the right arm of the membership function $A(x)$, respectively. Of course, we have $c \in \mathbb{R}$, and $s, l, r \geqslant 0$.

To introduce basic arithmetic operations (such as addition, subtraction, etc.) for the fuzzy numbers, the Zadeh extension principle should be applied (see, e.g., (Ban et al., 2015)). However, in the case of TPFNs, some operations are easier to conduct, e.g., for $A = (a_1, a_2, a_3, a_4)$ and $B = (b_1, b_2, b_3, b_4)$, we have

$$A + B = (a_1 + b_1, a_2 + b_2, a_3 + b_3, a_4 + b_4), \tag{3}$$

so the result is also a TPFN.

Resampling approaches for fuzzy data

Let $\widetilde{\mathbb{X}} = (\tilde{X}_1, \ldots, \tilde{X}_n)$ be a fuzzy random sample (Puri and Ralescu, 1986) and $\widetilde{\mathbb{x}} = (\tilde{x}_1, \ldots, \tilde{x}_n)$ – its actual realization. This sample will be called the **primary sample** or the **initial sample**. For the classical Effron's bootstrap, the bootstrap samples (or **secondary samples**) are drawn randomly with replacements from the primary sample. In this way we can create b bootstrap samples and the i-th element of the j-th secondary sample is denoted by $\tilde{x}^*_{ij}$.

Because of the previously mentioned shortcomings of Efron's approach, several modifications were proposed in the case of real-valued data. New resampling methods for fuzzy data were introduced recently in (Grzegorzewski et al., 2019, 2020b,a; Romaniuk, 2019; Romaniuk and Hryniewicz, 2019). All of them are implemented in **FuzzyResampling** package. We summarize them briefly below. For a more detailed review, we refer the reader to (Grzegorzewski and Romaniuk, 2022b).

The proposed approaches can be divided into two groups. In the first one, following (2), each TPFN $\tilde{x}_i$ is described by its "characteristic points", like the left bound of the core $lc_i = a_2 = c_i - s_i$, the core width $wc_i = a_3 - a_2 = 2s_i$, and the spreads l_i and r_i. Then, given a primary sample $\widetilde{\mathbb{x}}$, four sets are created: $LC = \{lc_1, \ldots, lc_n,\}$, $WC = \{wc_1, \ldots, wc_n,\}$, $L = \{l_1, \ldots, l_n,\}$ and $R = \{r_1, \ldots, r_n,\}$. If the **d-method** (Romaniuk and Hryniewicz, 2019) is applied then four numbers lc^*, wc^*, l^*, r^* from the respective sets LC, WC, L, R are drawn to construct a new bootstrapped element $\tilde{x}^*$. This procedure is repeated n times to obtain a whole secondary sample. For the **w-method** (Romaniuk and Hryniewicz, 2019) instead of the equal probabilities on the discrete sets, the special density $w(x)$ is used to construct x^*_i. It leads to a more diversified secondary sample than for the d-method. Both methods were also applied in (Romaniuk and Hryniewicz, 2021) for interval-valued fuzzy numbers.

The second group of methods contains the following so-called "flexible" bootstrap algorithms: **VA-method** (Grzegorzewski et al., 2019, 2020a), **EW-method** (Grzegorzewski et al., 2020a), **VAF-method** (Grzegorzewski et al., 2020b) and **VAA-method** (Grzegorzewski and Romaniuk, 2022b). Contrary to **d-method** or **w-method**, flexible approaches can be applied to primary samples of any fuzzy numbers but the generated outputs consist of TPFNs. However, the generated secondary samples preserve the so-called canonical representations of the primary observations. Our flexible methods differ in the type of preserved representation, which is summarized in Table 1.

Each of the flexible methods consists of four steps. Firstly, depending on the particular method, some characteristics of fuzzy numbers – like the value $\mathrm{Val}(\tilde{x}_i)$, ambiguity $\mathrm{Amb}(\tilde{x}_i)$ (Delgado et al., 1998), expected value (i.e. the middle point of the expected interval of a fuzzy number (Dubois and Prade, 1987; Heilpern, 1992) $\mathrm{EV}(\tilde{x}_i)$, width (Chanas, 2001) $\mathrm{w}(\tilde{x}_i)$, fuzziness (Delgado et al., 1998) $\mathrm{Fuzz}(\tilde{x}_i)$, left- and right-hand ambiguities $\mathrm{Amb}_L(\tilde{x}_i), \mathrm{Amb}_U(\tilde{x}_i)$ – are calculated for each observation $\tilde{x}_i$ of the primary sample. What values are determined depends on the user's decision on which values should be preserved during the generation of the bootstrap samples. While some characteristics will be kept, others will be randomly generated from the uniform distribution. Finally, the remained values describing a TPFN are directly calculated based on the formulas depending on the resampling approach. In Table 1 we list some flexible bootstrap methods along with an indication of which parameters are preserved, which are randomly generated, and which are directly calculated.

Method	Preserved	Randomly generated	Directly calculated
VA-method	Val, Amb	s, c	l, r
EW-method	EV, w	s, c	l, r
VAF-method	Val, Amb, Fuzz	c	s, l, r
VAA-method	Val, Amb_L, Amb_U	s	c, l, r

Table 1: Brief characteristic of the flexible bootstrap methods.

Overview of FuzzyResampling package

Below we briefly discuss functions implemented in **FuzzyResampling** package (version 0.6.0, available via CRAN). All further examples in R can be self-repeated using a supplementary file.

Generation of the initial sample

When the bootstrap is applied to solve any real-life problem the provided data is automatically treated as the initial sample. However, to perform any simulation study many synthetic samples are necessary so effective data generators from various distributions are then strongly desirable. In the case of reasoning with imprecise data, we need an algorithm generating fuzzy numbers used for modeling such data. In **FuzzyResampling** package two specialized functions to generate randomly trapezoidal fuzzy numbers are available:

```
GeneratorNU (n, mu, sigma, a, b, increases = FALSE, ...)
GeneratorNExpUU (n, mu, sigma, lambda, b, c, increases = FALSE, ...)
```

The third function, `GeneratorFuzzyNumbers`, can be used for various random distributions describing the "true origin" of TPFN, the increases of its core and the support. In this case, the respective functions from **stats** package (e.g., `rnorm`), together with their parameters' names, can be directly applied.

In both cases to generate a TPFN the parametrization described by (2) is used. In particular, to obtain a single TPFN four values c, s, l, r are generated independently using random distributions described in Table 2, where N(`mu`,`sigma`) stands for the normal distribution with the mean `mu` and standard deviation `sigma`, U (0,a) denotes the uniform distribution on the interval (0, a), Exp (`lambda`) – the exponential distribution with the parameter `lambda`, etc. More details concerning various types of simulated fuzzy numbers can be found in, e.g., (Grzegorzewski et al., 2020b), (Grzegorzewski and Romaniuk, 2022a).

Function	c	s	l	r
`GeneratorNU`	N(`mu`,`sigma`)	U (0,a)	U(0,b)	U(0,b)
`GeneratorNExpUU`	N(`mu`,`sigma`)	Exp (`lambda`)	U(0,b)	U(0,c)

Table 2: Distributions used to simulate fuzzy numbers.

If we like to generate a sample of `n` fuzzy numbers using any of the considered functions we receive the output as a matrix with `n` rows and four columns corresponding to values a_1, a_2, a_3, a_4 defined in (1) (abbreviated further as "the ends") if the parameter `increases` is set to `FALSE`, or producing values $l, c - s, c + s, r$ described in (2) (abbreviated further as "the increments"). It is worth noting that the parameter `increases` plays the same role to indicate the form of the output (or input) fuzzy numbers in all functions in this package.

As an example consider the following code generating a sample of $n = 10$ trapezoidal fuzzy numbers in the "ends" mode, i.e. a_1, a_2, a_3, a_4:

```
# seed PRNG
> set.seed(1234)

> sample1 <- GeneratorNU(10, mu = 0, sigma = 1, a = 0.2, b = 0.6)
> head(sample1,2)
           [,1]       [,2]       [,3]       [,4]
[1,] -1.6023884 -1.2703882 -1.1158475 -1.0715795
[2,] -0.1709531  0.2168906  0.3304666  0.5162785
```

Resampling methods

To perform the classical Efron's bootstrap `ClassicalBootstrap` function should be applied. Functions related to other methods (VA, EW, etc.) have the common form `XXXMethod`, where `XXX` stands for the respective method abbreviation. For instance, we write `EWMethod` in the case of the EW approach. A general structure of these functions is rather intuitive:

```
XXXMethod(initialSample, b = n, increases = FALSE)
```

where `initialSample` denotes the name of a matrix of input fuzzy numbers, `b` indicates the desired size of the bootstrap sample (if `b` is not specified we assume it is equal to the initial sample n, i.e. the number of rows in the input matrix), while by `increases` we set the mode ("ends" or "increments") used for describing fuzzy numbers both in the input and output matrices.

To generate the bootstrapped sample of $b = n = 10$ values using the classical Efron's approach based on the previously created initial sample `sample1` we should type:

```
> set.seed(1234)
>
```

```
> sampleEfron <- ClassicalBootstrap(sample1)
>
> head(sampleEfron,2)
           [,1]       [,2]       [,3]       [,4]
[1,] -1.3584678 -0.8991919 -0.7285674 -0.2195319
[2,]  0.0427377  0.3439362  0.6579900  0.9603501
```

Similarly, to perform the VA method for $b = 20$ we type:

```
> set.seed(1234)
>
> sampleVA <- VAMethod(sample1, 20)
>
> head(sampleVA,2)
             [,1]       [,2]       [,3]        [,4]
[1,] -1.091883230 -1.0324841 -0.8074509 -0.06176483
[2,]  0.009038746  0.3607857  0.4974204  1.28148926
```

Calculation of the mid/spread distance

A distance between two fuzzy numbers often used in the statistical context is the so-called mid/spread distance D_θ^λ proposed in (Bertoluzza et al., 1995; Trutschnig et al., 2009). It can be designated by a function

```
BertoluzzaDistance(fuzzyNumber1, fuzzyNumber2, theta = 1/3, increases = FALSE)
```

where `fuzzyNumber1` and `fuzzyNumber2` denote vectors containing the respective fuzzy numbers, `theta` is the weight of the impact of the distance between the mid-points of α-cuts in contrast to the distance between their spreads (usually `theta=1/3` or `theta=1`).

To find the mid/spread distance between two fuzzy numbers from the sample `sample1` we apply:

```
> BertoluzzaDistance(sample1[1,],sample1[2,])
[1] 1.488457
```

Estimation of SE/MSE

Bootstrap is often applied to estimate the standard error (SE) or the mean squared error (MSE) of the considered estimator. If the resulting estimator of the mean is given by a trapezoidal fuzzy number both SE and MSE can be obtained with the following function:

```
SEResamplingMean(initialSample, resamplingMethod = "ClassicalBootstrap",
  repetitions = 100, trueMean = NA, theta = 1/3, increases = FALSE)
```

where `initialSample` indicates a matrix containing the initial sample, `resamplingMethod` denotes a resampling method, and `repetitions` stands for the number of the bootstrap samples. If `trueMean` is not set then SE is estimated by the formula

$$\widehat{\mathrm{SE}} = \sqrt{\frac{1}{\text{repetitions} - 1} \sum_{k=1}^{\text{repetitions}} D_\theta^2\left(\bar{X}^{*i}, \bar{X}^{*}\right)}, \tag{4}$$

where D_θ is the mid/spread distance, while

$$\bar{X}^{*i} = \frac{1}{n}\sum_{j=1}^{n} X^{*ij}, \qquad \bar{X}^{*} = \frac{1}{\text{repetitions}} \sum_{i=1}^{\text{repetitions}} \bar{X}^{*i}, \tag{5}$$

and where X^{*ij} is the j-th fuzzy value in the i-th bootstrap sample. Otherwise, MSE is estimated as follows

$$\widehat{\mathrm{MSE}} = \frac{1}{\text{repetitions} - 1} \sum_{k=1}^{\text{repetitions}} D_\theta^2\left(\bar{X}^{*i}, \mathbb{E}X\right), \tag{6}$$

where $\mathbb{E}X$ is the Aumann-type mean (Puri and Ralescu, 1986) of the fuzzy number given by `trueMean` expressed by the four values (depending on the mode "ends" or "increments" specified in `increases` parameter).

To estimate SE using the classical bootstrap and the previously generated `sample1` we apply:

```
> set.seed(1234)
> SEResamplingMean(sample1)
$mean
[1] -0.70650791 -0.40296248 -0.22308095  0.05157294

$SE
[1] 0.05715056
```

Here, mean is equal to $\bar{X}^*$ while SE gives $\widehat{\mathrm{SE}}$.

On the other hand, the MSE based on the VA method can be estimated as follows:

```
> set.seed(1234)
>
> SEResamplingMean(sample1,resamplingMethod = "VAMethod",trueMean = c(-0.4,-0.1,0.1,0.4))
$mean
[1] -0.4 -0.1  0.1  0.4

$SE
[1] 0.04421334
```

where SE gives actually $\widehat{\mathrm{MSE}}$.

Bootstrapped one-sample test for the mean

Bootstrap methods are often used in hypothesis testing with fuzzy data. One of such tests, denoted further as the one-sample C-test (Colubi, 2009; Lubiano et al., 2016), is designed to verify

$$H_0 : \mathbb{E}X = \mu_0 \quad \text{vs.} \quad H_1 : \mathbb{E}X \neq \mu_0, \tag{7}$$

where μ_0 is a fixed fuzzy number assumed as the true fuzzy mean. This test can be invoked with the following function:

```
OneSampleCTest(initialSample,mu_0,numberOfSamples = 100,theta = 1/3,
  resamplingMethod = "ClassicalBootstrap",increases = FALSE)
```

where initialSample is a matrix describing the initial sample, mu_0 is the hypothetical mean μ_0, numberOfSamples is the number of bootstrap samples and resamplingMethod specifies the resampling method. As the output, we receive the estimated *p-value* which can be used to make a final decision (i.e. to reject or not the null hypothesis H_0).

To check whether the true mean of the population represented by sample1 is a TPFN mu_0 parametrized by $(-0.4, -0.1, 0.1, 0.4)$ we can apply the C-test based on the classical bootstrap as follows:

```
> set.seed(1234)
> OneSampleCTest(sample1, mu_0 = c(-0.4,-0.1,0.1,0.4))
[1] 0.31
```

To perform this test based on the VA method we apply:

```
> set.seed(1234)
>
> OneSampleCTest(sample1, mu_0 = c(-0.4,-0.1,0.1,0.4),
+                numberOfSamples = 1000, resamplingMethod = "VAMethod")
[1] 0.253
```

In both cases, there is no reason to reject the null hypothesis because the estimated p-values are greater than any reasonable significance level (like the traditional 0.05).

Bootstrapped two-sample test for the mean

A similar procedure can be applied to check whether there is a significant difference between the means of the two independent fuzzy samples. More precisely, we consider the so-called two-sample C-test (Lubiano et al., 2016) to verify

$$H_0 : \mathbb{E}X_1 = \mathbb{E}X_2 \quad \text{vs.} \quad H_1 : \mathbb{E}X_1 \neq \mathbb{E}X_2, \tag{8}$$

where $\mathbb{E}X_1$ and $\mathbb{E}X_2$ are the Aumann-type means of the first and second sample, respectively. Here the desired p-value can be calculated by the following function

```
TwoSampleCTest(initialSample1,initialSample2,numberOfSamples = 100,
 theta = 1/3,resamplingMethod = "ClassicalBootstrap",increases = FALSE)
```

where `initialSample1` and `initialSample2` are matrices corresponding to the first and the second initial sample, respectively, while other parameters remain identical as for `OneSampleCTest`.

Given previously generated `sample1` and `sample2` generated with a slight shift (i.e. for `mu = 0.2`) we verify (8) using the two-sample C-test based on the classical bootstrap. Thus we type the following commands:

```
> set.seed(1234)
> sample2 <- GeneratorNU(10, mu = 0.2, sigma = 1, a = 0.2, b = 0.6)
>
> TwoSampleCTest(sample1, sample2,numberOfSamples = 1000)
[1] 0.678
```

Thus obtaining so high p-value there is no reason to reject H_0.

Comparison of the resampling methods with FuzzyResampling package

Now we numerically compare the introduced resampling methods with the classical bootstrap using various statistical approaches like the standard error estimation or the power analysis.

Method comparison with SE/MSE

To compare the statistical properties of the considered resampling methods we firstly use SE/MSE for the mean. In some papers, like (Grzegorzewski et al., 2020a,b; Grzegorzewski and Romaniuk, 2022b)), it is stated that the proposed new resampling methods have usually smaller errors than the classical bootstrap. To check it, one may use the following `SEResamplingMean` function

```
ComparisonSEMean (generator, sampleSize = 10, numberOfSamples = 100,
  repetitions = 100,trueMean = NA, theta = 1/3, ...)
```

where `generator` indicates the algorithm applied to simulate fuzzy numbers (see Tab. 2), ... specifies some additional parameters required by `generator` and `numberOfSamples` stands for the number of the experiment repetitions performed to minimize the undesired influence of random fluctuations.

To estimate SE for a small (e.g. $n = 10$) or moderate (e.g. $n = 50$) initial sample size we proceed as follows:

```
> set.seed(1234567)

> outputSEsample3 <- ComparisonSEMean(generator = "GeneratorNExpUU",sampleSize = 10,
+ numberOfSamples = 10000,repetitions = 100,mu = 0,sigma = 4,lambda = 4,b = 0.4,c = 0.8)

> round(outputSEsample3, digits = 5)

ClassicalBootstrap  VAMethod  EWMethod  VAFMethod  DMethod  WMethod VAAMethod
           0.10756   0.10757   0.10753    0.10751  0.10763  0.10664  0.10764
```

As it is seen the estimated SE for the new resampling methods may be significantly lower (even about 1%) than for the classical bootstrap. These results confirm the remarks given in (Grzegorzewski and Romaniuk, 2022b) that "the w-method is usually the best approach for small sample sizes, while the VAF-method is the best for moderate and big samples. In general, when considering various models, it appears that the VAF-method and EW-method behave in a rather stable manner, while the behavior of the VAA-method depends strongly on the type of data." Some other examples can be found in the supplemental file.

A similar comparison of the bootstrap methods can be done using the MSE:

```
> set.seed(1234567)

> outputMSEsample2 <- ComparisonSEMean(generator = "GeneratorNExpUU",sampleSize = 10,
+ numberOfSamples = 10000, repetitions = 100, trueMean = c(-0.45,-0.25,0.25,0.65),
+ mu = 0, sigma = 4, lambda = 4, b = 0.4, c = 0.8)

> round(outputMSEsample1, digits = 5)
```

```
ClassicalBootstrap  VAMethod  EWMethod  VAFMethod  DMethod  WMethod  VAAMethod
           0.16495   0.16499   0.16482    0.16470  0.16496  0.16315  0.16519
```

Here the relative difference is about 1.1% if the w-method is compared with the classical bootstrap.

Test size estimation

We can also compare the empirical size (i.e. the maximal percentage of rejections if the null hypothesis holds) of the one-sample C-test based on different resampling approaches (see also (Grzegorzewski et al., 2020b)). To proceed with such a study the following special function based on `OneSampleCTest` is available:

```
ComparisonOneSampleCTest (generator,mu_0,shift=0,sampleSize = 10,numberOfSamples = 10,
 initialSamples = 100,theta = 1/3,significance = 0.05, ...)
```

It generates a sequence of initial samples (their number is given in `initialSamples` and the size is determined by `sampleSize`) for fuzzy numbers of the type specified by `generator`. Then some deterministic shift of the size `shift` is added to each fuzzy observation in these samples. Next, function `OneSampleCTest` is executed to calculate the p-value for each combination of the initial sample and resampling method. Then, by comparing the p-value with the assumed significance level `significance` we make a decision whether to reject the null hypothesis H_0 specified in (7) or not. The output of this procedure is the percentage of rejections in the sequence of experiments. To estimate the test size one should set `shift=0`. All other parameters of this function are passed to `OneSampleCTest` or the respective `generator`.

Let us consider the following simulation experiment:

```
> set.seed(1234567)

> outputCSizetest1 <- ComparisonOneSampleCTest(generator="GeneratorNU",
+ mu_0 = c(-0.4,-0.1,0.1,0.4), sampleSize = 10,numberOfSamples = 100,
+ initialSamples = 1000,mu = 0, sigma = 1,a = 0.2,b = 0.6)

> round(outputCSizetest1, digits = 4)

ClassicalBootstrap VAMethod EWMethod VAFMethod DMethod WMethod  VAAMethod
             0.038    0.035    0.040     0.035   0.037   0.040   0.034

# check the distance to the "true value" 0.05
> round(outputCSizetest1-0.05, digits = 4)

ClassicalBootstrap VAMethod EWMethod VAFMethod DMethod WMethod  VAAMethod
            -0.012   -0.015   -0.010    -0.015  -0.013  -0.010   -0.016
```

It is seen that the empirical sizes of the C-test combined with the new resampling methods might be definitely closer to the assumed significance level of 0.05 than obtained for Efron's bootstrap. Some other examples can be found in the supplemental file. The plot showing the estimated percentage of rejections as a function of the significance level is shown in Fig. 1. Because the distance between these two values (i.e. the percentage of rejections and the significance level) is usually low, the respective differences are additionally plotted in Fig. 2 for all the resampling methods.

Our experiments confirm the conclusion stated in (Grzegorzewski et al., 2020a) that the classical bootstrap was never the winner in such comparisons. New resampling methods usually behave much better – especially the d-method and the VA-method. The EW-method behaves always in a relatively stable manner and never drops below the third place.

Test power analysis

Our package enables also the power comparison for the one-sample C-test combined with different resampling methods. This is possible by using the `ComparisonOneSampleCTest` function which determines the percentage of the null hypothesis rejection under increasing shift added to the initial sample:

```
ComparePowerOneSampleCTest (generator,mu_0,shiftVector,sampleSize = 10,
 numberOfSamples = 10,initialSamples = 100,theta = 1/3,significance = 0.05, ...)
```

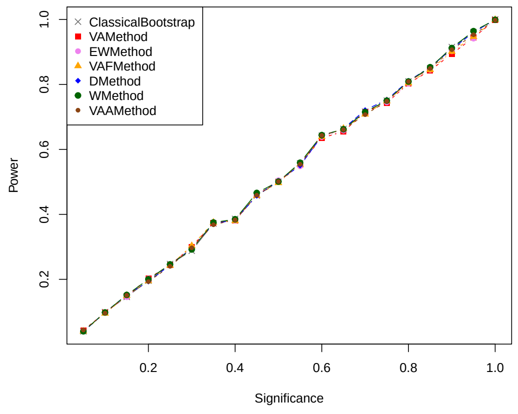

Figure 1: Test size depending on the significance level for the one-sample C-test based on the samples generated with `GeneratorNU`.

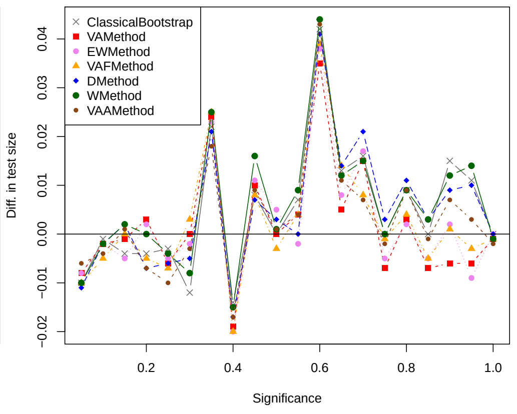

Figure 2: Differences in test size between the resampling methods and the significance level.

This procedure produces a matrix with the successive shift values, set in the vector `shiftVector`, specified in the rows and the corresponding percentages of rejections for the respective resampling methods given in columns.

Below we show how to apply this function to comparisons based on very small samples ($n = 5$). To minimize undesired random effects the following experiment was repeated 10000 times:

```
# prepare vector of shifts
> shifts <- seq(0.1,1,by = 0.05)

> set.seed(1234567)

> outputCPowerTest1 <- ComparePowerOneSampleCTest(generator="GeneratorNU",
+ mu_0 = c(-0.4,-0.1,0.1,0.4),shiftVector = shifts,sampleSize = 5,numberOfSamples = 100,
+ initialSamples = 10000,mu = 0,sigma = 1,a = 0.2,b = 0.6)

> head(round(outputCPowerTest1, digits = 4),4)

     ClassicalBootstrap VAMethod EWMethod VAFMethod DMethod WMethod VAAMethod
0.1              0.0274   0.0302   0.0305    0.0314  0.0288  0.0276    0.0278
0.15             0.0278   0.0321   0.0321    0.0305  0.0304  0.0293    0.0297
0.2              0.0315   0.0357   0.0337    0.0325  0.0339  0.0326    0.0302
0.25             0.0386   0.0393   0.0386    0.0391  0.0395  0.0397    0.0366
```

The results given both in the output matrix and in Fig. 3 show that the new resampling methods have bigger power than the classical bootstrap approach. To highlight the results better, differences between the power curves for each of the new resampling methods and Efron's bootstrap are plotted in Fig. 4. Observing these results we can agree with the conclusions of simulations reported in (Grzegorzewski and Romaniuk, 2022b), that "In general (i.e. including other models) this method *(VA-method)* was usually one of the best, especially for smaller sample sizes or a lower number of bootstrap replications (e.g. $b = 100$). The VAA-method or the EW-method were usually just after VA-method; the d-method, w- and the VAA-method were neither too good nor too bad, but the classical bootstrap (as compared with other resampling methods) was usually the worse method in hypotheses testing."

Two-sample test – a real-life case

Finally, we compare the resampling methods combined with the two-sample C-test applied to some real-life imprecise data. Such data may appear naturally wherever we deal with expert opinions expressed in everyday language. In our study, we consider the opinions of the experts related to the Gamonedo cheese quality (a kind of blue cheese produced in Asturias, Spain). Their inherently imprecise assessments were modeled by the TPFNs (Ramos-Guajardo et al., 2019).

To check whether two given experts, say A and C, differ significantly in their opinions, we verify the null hypothesis (8) of no difference (Grzegorzewski et al., 2020a). We can perform the desired test using the following function based on `TwoSampleCTest`:

```
CompareRealCaseTwoSample(numberOfSamples, numberOfIterations)
```

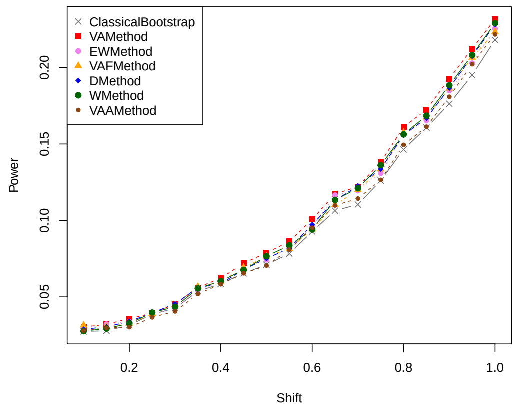

Figure 3: Power curves of the one-sample C-test based on the samples generated with `GeneratorNU`.

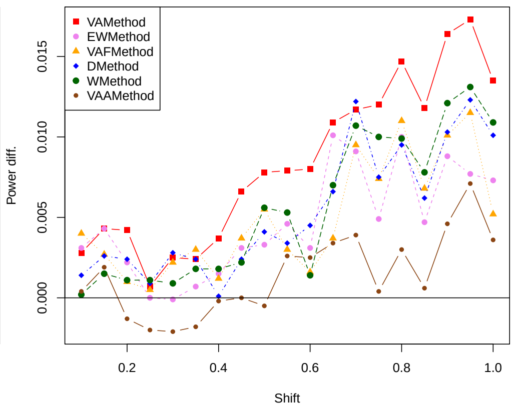

Figure 4: Differences in power curves between the new resampling methods and the classical approach.

which delivers p-values of the two-sample C-test for all resampling methods. To eliminate undesired randomness each experiment (i.e. the bootstrap procedure for the single test) is repeated `numberOfIterations` times and then the obtained p-values are averaged. The parameter `numberOfSamples` is passed directly to `TwoSampleCTest`.

Then the following function is applied:

```
> set.seed(1234567)
> realCaseP <- CompareRealCaseTwoSample(numberOfSamples=100,numberOfIterations=10000)
>
> round(realCaseP,5)
     ClassicalBootstrap VAMethod EWMethod VAFMethod DMethod WMethod VAAMethod
     0.82055            0.91213  0.88388  0.85542   0.82179 0.82212 0.85455
```

Assuming any reasonable significance level the null hypothesis H_0 is not rejected. Thus, whatever considered resampling method is applied, we may conclude that given two experts do not differ significantly in their opinions. However, the new resampling methods lead usually to bigger p-values than the classical bootstrap (i.e. we are "more sure" that H_0 can be accepted).

Conclusions

The main goal of the proposed resampling algorithms for fuzzy samples of trapezoidal fuzzy numbers is to overcome the shortcomings typical to the classical bootstrap. New approaches are implemented in **FuzzyResampling** package and are supplemented with some ready-to-use functions useful in statistical inference based on imprecise data. It is shown that in some cases analyzed in this paper and related literature the suggested bootstrap algorithms are more effective than the classical one.

Obviously, many problems are still open. New methods and algorithms that would be helpful in solving many problems faced by practitioners are still needed. In particular, an in-depth study on the so-called "epistemic bootstrap" (Grzegorzewski and Romaniuk, 2021) would be useful to handle "epistemic data" instead of the "ontic data" (Couso and Dubois, 2014) discussed in this paper.

Bibliography

A. Ban, L. Coroianu, and P. Grzegorzewski. *Fuzzy Numbers: Approximations, Ranking and Applications*. Polish Academy of Sciences, Warsaw, 2015. [p2]

C. Bertoluzza, N. Corral, and A. Salas. On a new class of distances between fuzzy numbers. *Mathware and Soft Computing*, 2(2):71–84, 1995. URL `https://eudml.org/doc/39054`. [p5]

S. Chanas. On the interval approximation of a fuzzy number. *Fuzzy Sets and Systems*, 122:353–356, 2001. [p3]

A. Colubi. Statistical inference about the means of fuzzy random variables: Applications to the analysis of fuzzy- and real-valued data. *Fuzzy Sets and Systems*, 160:344–356, 2009. [p6]

L. Coroianu, M. Gagolewski, and P. Grzegorzewski. Nearest piecewise linear approximation of fuzzy numbers. *Fuzzy Sets and Systems*, 233:26–51, 2013. ISSN 0165-0114. doi: https://doi.org/10.1016/j.fss.2013.02.005. Theme: Fuzzy numbers and statistics. [p2]

I. Couso and D. Dubois. Statistical reasoning with set-valued information: Ontic vs. epistemic views. *International Journal of Approximate Reasoning*, 55:1502–1518, 2014. [p10]

A. C. Davison and D. V. Hinkley. *Bootstrap Methods and their Application*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997. doi: 10.1017/CBO9780511802843. [p2]

M. Delgado, M. Vila, and W. Voxman. On a canonical representation of a fuzzy number. *Fuzzy Sets and Systems*, 93:125–135, 1998. [p1, 3]

D. Dubois and H. Prade. The mean value of a fuzzy number. *Fuzzy Sets and Systems*, 24:279–300, 1987. [p3]

B. Efron. Bootstrap methods: Another look at the jackknife. *Annals of Statistics*, 7:1–26, 1979. [p1]

B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Number 57 in Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, Boca Raton, Florida, USA, 1993. [p1, 2]

M. Gil, M. Montenegro, G. González-Rodríguez, A. Colubi, and M. Casals. Bootstrap approach to the multi-sample test of means with imprecise data. *Computational Statistics and Data Analysis*, 51: 148–162, 2006. [p1]

E. Giné and J. Zinn. Bootstrapping general empirical measures. *Annals of Probability*, 18(2):851–869, 1990. [p1]

G. González-Rodríguez, M. Montenegro, A. Colubi, and M. Gil. Bootstrap techniques and fuzzy random variables: Synergy in hypothesis testing with fuzzy data. *Fuzzy Sets and Systems*, 157: 2608–2613, 2006. [p1]

P. Grzegorzewski and M. Romaniuk. Epistemic bootstrap for fuzzy data. In *Joint Proceedings of IFSA-EUSFLAT-AGOP 2021 Conferences*, pages 538–545. Atlantis Press, 2021. [p10]

P. Grzegorzewski and M. Romaniuk. Bootstrapped Kolmogorov-Smirnov test for epistemic fuzzy data. In D. Ciucci, I. Couso, J. Medina, D. Ślęzak, D. Petturiti, B. Bouchon-Meunier, and R. R. Yager, editors, *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 494–507, Cham, 2022a. Springer International Publishing. ISBN 978-3-031-08974-9. [p4]

P. Grzegorzewski and M. Romaniuk. Bootstrap methods for fuzzy data. In K. T. Atanassov, V. Atanassova, J. Kacprzyk, A. Kałuszko, M. Krawczak, J. W. Owsiński, S. S. Sotirov, E. Sotirova, E. Szmidt, and S. Zadrożny, editors, *Uncertainty and Imprecision in Decision Making and Decision Support: New Advances, Challenges, and Perspectives*, pages 28–47, Cham, 2022b. Springer International Publishing. ISBN 978-3-030-95929-6. [p1, 3, 7, 9]

P. Grzegorzewski, O. Hryniewicz, and M. Romaniuk. Flexible bootstrap based on the canonical representation of fuzzy numbers. In *Proceedings of EUSFLAT 2019*. Atlantis Press, 2019. ISBN 978-94-6252-770-6. doi: https://doi.org/10.2991/eusflat-19.2019.68. [p1, 3]

P. Grzegorzewski, O. Hryniewicz, and M. Romaniuk. Flexible resampling for fuzzy data. *International Journal of Applied Mathematics and Computer Science*, 30(2):281–297, 2020a. [p1, 3, 7, 8, 9]

P. Grzegorzewski, O. Hryniewicz, and M. Romaniuk. Flexible bootstrap for fuzzy data based on the canonical representation. *International Journal of Computational Intelligence Systems*, 13:1650–1662, 2020b. [p1, 3, 4, 7, 8]

S. Heilpern. The expected value of a fuzzy number. *Fuzzy Sets and Systems*, 47:81–86, 1992. [p3]

M. A. Lubiano, M. Montenegro, B. Sinova, S. de la Rosa de Sáa, and M. A. Gil. Hypothesis testing for means in connection with fuzzy rating scale-based data: algorithms and applications. *European Journal of Operational Research*, 251:918–929, 2016. [p1, 6]

M. A. Lubiano, A. Salas, C. Carleos, and M. A. de la Rosa de Sáa, S.and Gil. Hypothesis testing-based comparative analysis between rating scales for intrinsically imprecise data. *International Journal of Approximate Reasoning*, 88:128–147, 2017. [p1]

M. Montenegro, A. Colubi, M. Casals, and M. Gil. Asymptotic and bootstrap techniques for testing the expected value of a fuzzy random variable. *Metrika*, 59:31–49, 2004. [p1]

M. Puri and D. Ralescu. Fuzzy random variables. *Journal of the Mathematical Analysis and Applications*, 114:409–422, 1986. [p3, 5]

A. Ramos-Guajardo, A. Blanco-Fernández, and G. González-Rodríguez. Applying statistical methods with imprecise data to quality control in cheese manufacturing. In P. Grzegorzewski, A. Kochanski, and J. Kacprzyk, editors, *Soft Modeling in Industrial Manufacturing*, pages 127–147. Springer, 2019. [p9]

M. Romaniuk. On some applications of simulations in estimation of maintenance costs and in statistical tests for fuzzy settings. In A. Steland, E. Rafajłowicz, and O. Okhrin, editors, *Stochastic Models, Statistics and Their Applications*, pages 437–448. Springer International Publishing, 2019. [p1, 3]

M. Romaniuk and O. Hryniewicz. Interval-based, nonparametric approach for resampling of fuzzy numbers. *Soft Computing*, 23:5883–5903, 2019. [p1, 3]

M. Romaniuk and O. Hryniewicz. Discrete and smoothed resampling methods for interval-valued fuzzy numbers. *IEEE Transactions on Fuzzy Systems*, 29:599–611, 2021. ISSN 1941-0034. doi: 10.1109/TFUZZ.2019.2957253. [p3]

B. W. Silverman and G. A. Young. The bootstrap: To smooth or not to smooth? *Biometrika*, 74(3): 469–479, 1987. [p1]

W. Trutschnig, G. González-Rodríguez, A. Colubi, and M. Gil. A new family of metrics for compact, convex (fuzzy) sets based on a generalized concept of mid and spread. *Information Sciences*, 179: 3964–3972, 2009. [p5]

W. Trutschnig, M. A. Lubiano, and J. Lastra. *SAFD — An R Package for Statistical Analysis of Fuzzy Data*, pages 107–118. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. doi: 10.1007/978-3-642-30278-7_10. [p2]

Maciej Romaniuk
Systems Research Institute, Polish Academy of Sciences
Newelska 6, 01-447 Warsaw
Poland
(0000-0001-9649-396X)
mroman@ibspan.waw.pl

Przemysław Grzegorzewski
Faculty of Mathematics and Information Science, Warsaw University of Technology
Koszykowa 75, 00-662 Warsaw
Poland
Systems Research Institute, Polish Academy of Sciences
Newelska 6, 01-447 Warsaw
Poland
(0000-0002-5191-4123)
pgrzeg@pw.edu.pl