 doi:10.20944/preprints202210.0464.v1

Article

Some Estimators and their Properties Following Kabirian-based Optinalysis

Kabir Bindawa Abdullahi

Department of Biology, Faculty of Natural and Applied Sciences, Umaru Musa Yar'adua University, P.M.B., 2218 Katsina, Katsina State, Nigeria; kabir.abdullahi@umyu.edu.ng or kabirnamallam@gmail.com

Abstract: Good estimators are characterized as robust, unbiased, efficient, and consistent. However, the commonly used estimators are weak or lack one or more of these properties. In this article, eight (8) estimators for statistical and geometrical estimations of symmetry/asymmetry, similarity/dissimilarity, identity/unidentity, and feature transformation were proposed following Kabirian-based optinalysis and other operations. The proposed estimators are characterized as invariant (robust) under scaling, location shift, and rotation or reflection. A computing code was written in python language for each of the proposed estimators so that peers can have working codes for application and performance evaluation.

Keywords: Kabirian-based optinalysis; estimators; properties; computing codes

1. Introduction

Good estimators are characterized as robust (invariant), unbiased, efficient, and consistent [1], [2]. However, the commonly used estimators of symmetry/asymmetry, similarity/dissimilarity/distance, and identity/unidentity estimation lack one or more of these properties. Methods of symmetry/asymmetry detection of shapes and distributions and those of similarity/dissimilarity/deviation/distance measures between objects, shapes, and distributions have been developed since earlier times. However, these methods can either or not be characterized as scale-invariant, location-invariant, and scale-and-location-invariant [3], [4], [5], [6], [7]

Symmetry or asymmetry detectors are good candidates for shape analysis and are very useful tools in object detection and recognition in many situations. Symmetry is characterized as the invariance of objects or properties under a set of operations. Moreover, fast and effective symmetry recognition is still a difficult problem in computer vision [3], [8]. In image analysis, most of the symmetry detectors are invariant to scaling and rotation but not to the contrast and brightness of images, but some asymmetry detectors are invariant to image contrast [6].

Similarity, distance, or deviation measures are the core components used by distance-based clustering algorithms that placed dissimilar data points into different clusters, while similar data points are placed in the same clusters [5]. Similarly, deviation measures of an independent and identically distributed random variable(s) are underpinned by the measures of statistical dispersion. Some of the major disadvantages of commonly used similarity, distance or deviation measures include outlier sensitivity, lack of invariance to linear transformation, and Low accuracy for high-dimensional datasets [5], [7].

The estimators of symmetry/asymmetry, similarity/dissimilarity/distance, and identity/unidentity have one common mathematical feature of automorphism or isomorphism [3]. Therefore, a conceptually and theoretically good estimator of symmetry/asymmetry, similarity/dissimilarity/distance, and identity/unidentity should be proven as a bijection function. Almost all the estimators have failed to meet this important operation. Recently, the emergence of Kabirian-based optinalysis could be utilized for the development of alternative estimators [3]. Kabirian-based optinalysis is a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

function that isoreflectively or autoreflectively compares the similarity, symmetry, and identity between two mathematical structures as an optic-like (mirror-like) reflection of each other about a mid-point or symmetrical line [8].

In this paper, Kabirian-based optinalysis, coupled with other operations were used. About eight (8) estimators for statistical and geometrical symmetry/asymmetry, similarity/dissimilarity/distance, identity/unidentity estimations, and feature transformation were proposed.

2. Methods

2.1. Proposal 1: Statistical Symmetry

2.1.1. Definition

Statistical symmetry is a mirror reflection of statistically defined data points to itself. Under Kabirian-based optinalysis, it is the autoreflectivity of data points to itself. Statistical symmetry refers to the theoretical ordering, with or without centering the data and optinalysing the established autoreflective pair for a given variable.

2.1.2. Computational steps and algorithmic procedure

Suppose we have a set of variables $X = (x_3, x_1, x_2, \ldots\ldots, x_n)$. Let the order of algorithmic transformations t_c and t_o as centering and ordering of the data X respectively. The optinalysis-based statistical symmetry implies the following steps:

First step 1: Centering the data X (i.e., location removal). This is optional, depending on the task. By centering the variable, two distinct sets of positive and negative integers were obtained.

$$t_c(X) = (x_3, x_1, x_2, \ldots\ldots, x_n)$$

Second step 2: Establish a theoretical order and the autoreflective pair of symmetry (shape) for the $t_c(X)$ variable. Note that numerical values are theoretically arranged in ascending or descending order. The two distinct separations of the integers into a positive and negative form the basis for the establishment of the autoreflective pair. For the efficiency of the result, absolute estimates of the centered data are used.

$$\left(t_{c,o}(X)\middle|t_{c,o}(X')\right) = (x_1 \leq, x_2 \leq, x_3 \ldots \leq x_{\frac{n-1}{2}}, \hat{x}_{\frac{n+1}{2}}, x'_{\frac{n-1}{2}} \ldots \geq x'_3, \geq x'_2, \geq x'_1)$$

$$\left(t_{c,o}(X)\middle|t_{c,o}(X')\right) = (x_1 \geq, x_2 \geq, x_3 \ldots \geq x_{\frac{n-1}{2}}, \hat{x}_{\frac{n+1}{2}}, x'_{\frac{n-1}{2}} \ldots \leq x'_{n-2}, \leq x'_{n-1}, \leq x'_n)$$

Third step 3: Optinalyse (by Kabirian-based optinalysis [3]) the autoreflective pair of $\left(t_{c,o}(X)\middle|t_{c,o}(X')\right)$.

$$f\colon t_{c,o}(X) \overset{\delta}{\twoheadrightarrow} t_{c,o}(X') \twoheadrightarrow R$$

$$f\colon \begin{bmatrix} t_{c,o}(X) = \left(x_1, x_2, x_3, \ldots\ldots, x_{\frac{n-1}{2}}\right) & \delta = \left(\hat{x}_{\frac{n+1}{2}}\right) & t_{c,o}(X') = \left(x'_{\frac{n-1}{2}}, \ldots\ldots, x'_3, x'_2, x'_1\right) \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, r_2, r_3, \ldots\ldots, r_{\frac{n-1}{2}}, & r_{\frac{n+1}{2}}, & r_{\frac{n+3}{2}}, \ldots\ldots, r_{n-2}, r_{n-1}, r_n) \end{bmatrix}$$

Such that $t_{c,o}(X), t_{c,o}(X'), \delta \,\&\, R \in \mathbb{R}$; $r_1 \neq 0$, $n \in \mathbb{N}$; $t_{c,o}(X) \,\&\, t_{c,o}(X') \in Y$ and $t_{c,o}(X) \,\&\, t_{c,o}(X')$ are autoreflective pairs about a central point δ.

By Kabirian-based optinalysis [3], the Kabirian coefficient of similarity ($KC_{Ssym.}$) and its derivatives satisfied the Y-rule of Kabirian-based automorphic optinalysis.

$$\begin{matrix} KC1_{Ssym.}(X, X') \\ \ldots \\ KC2_{Ssym.}(X', X) \end{matrix} \rightleftharpoons P_{Ssym.}(X, X') = P_{Ssym.}(X', X) \rightleftharpoons P_{Sasym.}(X, X') = P_{Sasym.}(X', X)$$

where $X, X' \in \mathbb{R}$.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

The two possible Kabirian bi-coefficients ($KC1_{Ssym.}$ & $KC2_{Ssym.}$) function on two different, but inverse optinalytic scales.

2.1.3. Scale and scaloc-invariant statistical symmetry

Statistical symmetry is called scale invariance if the efficient location parameter is not removed from the variable, while it is called scaloc-invariant if the efficient location parameter is removed from the variable.

2.1.4. General properties of statistical symmetry

i. It is based on the entire observations of variables, unlike percentile-based or decile-based statistics. Therefore, extreme maximum and minimum values are not discarded or trimmed.
ii. It applies to variable(s) from the set of real numbers.
iii. Statistical symmetry is scale-invariant (i.e., robust to scale, and unitless) estimator.

Supposed we have an a scaling of a variable $x = (x_1, x_2, x_3, \ldots\ldots, x_n)$.

$$KC_{Ssym.}(x) = KC_{Ssym.}(ax) = KC_{Ssym.}(-ax)$$
$$P_{Ssym.}(x) = P_{Ssym.}(ax) = P_{Ssym.}(-ax)$$
$$P_{Sasym.}(x) = P_{Sasym.}(ax) = P_{Sasym.}(-ax)$$

where $x, a \in \mathbb{R}; a \neq 0$.

Statistical symmetry is a location-invariant estimator, only if, the efficient location parameter is removed from the variable. For instance, taking the absolute distances from the mean.

iv. Supposed we have a variable $x = (x_1, x_2, x_3, \ldots\ldots, x_n)$, then it is shifted by a b location and returned as $\hat{x} = (x_1 + b, x_2 + b, x_3 + b, \ldots\ldots, x_n + b)$. The location-invariance implies:

$$KC_{Ssym.}(x) = KC_{Ssym.}(\hat{x} - \mu) \neq KC_{Ssym.}(\hat{x})$$
$$P_{Ssym.}(x) = P_{Ssym.}(\hat{x} - \mu) \neq P_{Ssym.}(\hat{x})$$
$$P_{Sasym..}(x) = P_{Sasym.}(\hat{x} - \mu) \neq P_{Sasym.}(\hat{x})$$

where $x, \hat{x}, b \in \mathbb{R}$; μ is the mean estimate of $\hat{x}$.

v. Where location and scale properties are combined (i.e., scaloc-transform distribution), statistical symmetry is its scaloc-invariant and/or scale-invariant estimator.

Supposed we have an a scaling and b location shift of a variable $x = (x_1, x_2, x_3, \ldots\ldots, x_n)$.

Here is a situation (of scaloc-transform distribution) where only the scaloc-invariant statistical symmetry is an efficient estimator.

$$KC_{Ssym.}(x) = KC_{Ssym.}\{(xa + b) - \mu\} = KC_{Ssym.}\{(xa - b) - \mu\}$$
$$P_{Ssym.}(x) = P_{Ssym.}\{(xa + b) - \mu\} = P_{Ssym.}\{(xa - b) - \mu\}$$
$$P_{Sasym.}(x) = P_{Sasym.}\{(xa + b) - \mu\} = P_{Sasym.}\{(xa - b) - \mu\}$$

where $x, a, b \in \mathbb{R}$; $a \neq 0$; μ is the mean estimate of $(xa \pm b)$; and the resultant effect is a location property.

Here is a situation (of scaloc-transform distribution) where both the scaloc-invariant and scale-invariant statistical symmetries are efficient estimators.

$$KC_{Ssym.}(x) = KC_{Ssym.}\{a(x + b) - \mu\} = KC_{Ssym.}\{-a(x + b) - \mu\}$$
$$P_{Ssym.}(x) = P_{Ssym.}\{a(x + b) - \mu\} = P_{Ssym.}\{-a(x + b) - \mu\}$$
$$P_{Sasym.}(x) = P_{Sasym.}\{a(x + b) - \mu\} = P_{Sasym.}\{-a(x + b) - \mu\}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

where $x, a, b \in \mathbb{R}$; $a \neq 0$; μ is the mean estimate of $\pm a(x + b)$; and the resultant effect is a scale property.

vi. Statistical symmetry is invariant to sample size or multiple repeats of a univariate dataset. But the univariate sample size invariance is effective to $P_{Ssym.}$ and $P_{Sasym.}$, and not to $KC_{Ssym.}$.

Supposed we have a c duplicate of a variable $x = (x_1, x_2, x_3, \ldots, x_n)$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Ssym.}(x) \neq KC_{Ssym.}([x] * c)$$
$$P_{Ssym.}(x) = P_{Ssym.}([x] * c)$$
$$P_{Sasym.}(x) = P_{Sasym.}([x] * c)$$

where $x \in \mathbb{R}; c \in \mathbb{N}$.

vii. Statistical symmetry is invariant to sample size or multiple repeats of multivariate datasets. But the multivariate sample size invariance is effective to $P_{Ssym.}$ and $P_{Sasym.}$, and not to $KC_{Ssym.}$.

Supposed we have a a duplicate of variables $x = (x_1, x_2, x_3, \ldots, x_n)$, $y = (y_1, y_2, y_3, \ldots, y_n)$, $z = (z_1, z_2, z_3, \ldots, z_n)$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Ssym.}([[x], [y], [z]]) \neq KC_{Ssym.}([[x], [y], [z]] * c)$$
$$P_{Ssym.}([[x], [y], [z]]) = P_{Ssym.}([[x], [y], [z]] * c)$$
$$P_{Sasym.}([[x], [y], [z]]) = P_{Sasym.}([[x], [y], [z]] * c)$$

where $x, y, z \in \mathbb{R}; c \in \mathbb{N}$.

2.1.5. Python code

Get the python code at:

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/symmetry_estimators.ipynb

Input guide: **symmetry***([data, centering, ordering, print])*

Input options:

- ✓ **for data:** *list of numerical values from a set of real numbers*.
- ✓ **for centering:** "*allow*", or "*never*".
- ✓ **for ordering:** "*ascend*", "*descend*", or "*never*".
- ✓ **for print:** "*kc*", "*psym*", "*pasym*", "*kcalt1*", "*kcalt2*", or "*kcalt*".

Examples:

print("*Kabirian coefficient =*", **symmetry**([*sorted([2,4,6,8,4,2,4,5,6])*, "*centering:allow*", "*ordering:never*", "*print:kc*"]))

print("*Probability of symmetry =*", **symmetry**([*sorted([2,4,6,8,4,2,4,5,6])*, "*centering:allow*", "*ordering:never*", "*print:psym*"]))

print("Probability of asymmetry =", **symmetry**([*sorted([2,4,6,8,4,2,4,5,6]), "centering:allow", "ordering:never", "print:pasym"])*)

print("Alt1. Kabirian coefficient =", **symmetry**([*sorted([2,4,6,8,4,2,4,5,6]), "centering:allow", "ordering:never", "print:kcalt1"])*)

print("Alt2. Kabirian coefficient =", **symmetry**([*sorted([2,4,6,8,4,2,4,5,6]), "centering:allow", "ordering:never", "print:kcalt2"])*)

print("Alt. Kabirian coefficient =", **symmetry**([*sorted([2,4,6,8,4,2,4,5,6]), "centering:allow", "ordering:never", "print:kcalt"])*)

2.1.6. Drawbacks and limitations of statistical symmetry

The following are some of the identified drawbacks and limitations of statistical symmetry:

i. The given random ordering of elements of the list of the variable(s) is not preserved, thus an efficient theoretical ordering (i.e., ascend or descend sorting) has to be adopted or used.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

ii. The two possible Kabirian bi-coefficients do not function on the same optinalytic scale. For comparison of results, estimates with the mixed Kabirian coefficients should either be translated forward or otherwise uniformed by backward alternate translation.

2.2. Proposal 2: Geometrical Symmetry

2.2.1. Definition

Geometrical symmetry is a mirror reflection of a geometrically defined data point to itself. Under optinalysis, it is the autoreflectivity of data points to itself. Geometrical symmetry refers to the conceptual ordering, with or without centering the data and optinalysing the established autoreflective pair for a given variable.

2.2.2. Computational steps and algorithmic procedure

Suppose we have a sequence of variable $Y = (y_2, y_1, y_3, \ldots\ldots, y_n)$. Let the order of algorithmic transformations t_d and t_c as centering and ordering of the data Y respectively. The optinalysis-based geometrical symmetry implies the following steps:

First step 1: Center the data Y (i.e., location removal). This is optional, depending on the task.

$$t_c(Y) = (y_2, y_1, y_3, \ldots\ldots, y_n)$$

Second step 2: Establish a conceptual order and the autoreflective pair of symmetry (shape) for the $t_c(Y)$ variable.

$$\left(t_{c,o}(Y)\middle|t_{c,o}(Y')\right) = (y_2, y_1, y_3, \ldots\ldots y_{\frac{n-1}{2}}, \hat{y}_{\frac{n+1}{2}}, y'_{\frac{n-1}{2}} \ldots\ldots, y'_3, y'_1, y'_2))$$

Third step 3: Optinalyse (by Kabirian-based optinalysis [3]) the autoreflective pair of $\left(t_{c,o}(Y)\middle|t_{c,o}(Y')\right)$.

$$f\colon t_{c,o}(Y) \overset{\delta}{\twoheadrightarrow} t_{c,o}(Y') \twoheadrightarrow R$$

$$f\colon \begin{bmatrix} t_{c,o}(Y) = \left(y_1, y_2, y_3, \ldots\ldots, y_{\frac{n-1}{2}}\right) & \delta = \left(\hat{y}_{\frac{n+1}{2}}\right) & t_{c,o}(Y') = \left(y'_{\frac{n-1}{2}}, \ldots\ldots, y'_3, y'_2, y'_1\right) \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, r_2, r_3, \ldots\ldots, r_{\frac{n-1}{2}}, & r_{\frac{n+1}{2}}, & r_{\frac{n+3}{2}}, \ldots\ldots, r_{n-2}, r_{n-1}, r_n) \end{bmatrix}$$

Such that $t_{c,o}(Y), t_{c,o}(Y'), \delta \,\&\, R \in \mathbb{R}$; $r_1 \neq 0$; $n \in \mathbb{N}$; $t_{c,o}(Y) \,\&\, t_{c,o}(Y') \in Y$ and $t_{c,o}(Y) \,\&\, t_{c,o}(Y')$ are autoreflective pairs about a central point δ.

By Kabirian-based optinalysis [3], the Kabirian coefficient of similarity $(KC_{Gsym.})$ and its derivatives satisfied the Y-rule of Kabirian-based automorphic optinalysis.

$$\begin{matrix} KC1_{Gsym.}(Y, Y') \\ \ldots \\ KC2_{Gsym.}(Y', Y) \end{matrix} \rightleftharpoons P_{Gsym.}(Y, Y') = P_{Gsym.}(Y', Y) \rightleftharpoons P_{Gasym.}(Y, Y') = P_{Gasym.}(Y', Y)$$

where $Y, Y' \in \mathbb{R}$.

The two possible Kabirian bi-coefficients $(KC1_{Gsym.} \,\&\, KC2_{Gsym.})$ function on two different, but inverse optinalytic scales.

2.2.3. Scale and scaloc-invariant geometrical symmetry

Geometrical symmetry is called scale invariance if the efficient location parameter is not removed from the variable, while it is called scaloc-invariant if the efficient location parameter is removed from the variable.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

2.2.4. General properties of geometrical symmetry

i. It is based on the entire observations of variables, unlike percentile-based or decile-based statistics. Therefore, extreme maximum and minimum values are not discarded or trimmed.

ii. It applies to variable(s) from the set of real numbers.

iii. Geometrical symmetry is scale-invariant (i.e., robust to scale, and unitless) estimator.

Supposed we have an a scaling of a variable $y = (y_1, y_2, y_3, \ldots, y_n)$.

$$KC_{Gsym.}(y) = KC_{Gsym.}(ay) = KC_{Gsym.}(-ay)$$

$$P_{Gsym.}(y) = P_{Gsym.}(ay) = P_{Gsym.}(-ay)$$

$$P_{Gasym.}(y) = P_{Gasym.}(ay) = P_{Gasym.}(-ay)$$

where $y, a \in \mathbb{R};\ a \neq 0$.

iv. Geometrical symmetry is a location-invariant estimator, only if, the efficient location parameter is removed from the variable. For instance, taking the absolute distances from the mean.

Supposed we have a variable $y = (y_1, y_2, y_3, \ldots, y_n)$, then it is shifted by a b location and returned as $\hat{y} = (y_1 + b, y_2 + b, y_3 + b, \ldots, y_n + b)$. The location-invariance implies:

$$KC_{Gsym.}(y) = KC_{Gsym.}(\hat{y} - \mu) \neq KC_{Gsym.}(\hat{y})$$

$$P_{Gsym.}(y) = P_{Gsym.}(\hat{y} - \mu) \neq P_{Gsym.}(\hat{y})$$

$$P_{Gasym..}(y) = P_{Gasym.}(\hat{y} - \mu) \neq P_{Gasym.}(\hat{y})$$

where $y, \hat{y}, b \in \mathbb{R};\ \mu$ is the mean estimate of $\hat{y}$.

v. Where location and scale properties are combined (i.e., scaloc-transform distribution), geometrical symmetry is its invariant scaloc-invariant and/or scale-invariant estimator.

Supposed we have an a scaling and b location shift of a variable $y = (y_1, y_2, y_3, \ldots, y_n)$.

Here is a situation (of scaloc-transform distribution) where only the scaloc-invariant geometrical symmetry is an efficient estimator.

$$KC_{Gsym.}(y) = KC_{Gsym.}\{(ya + b) - \mu\} = KC_{Gsym.}\{(ya - b) - \mu\}$$

$$P_{Gsym.}(y) = P_{Gsym.}\{(ya + b) - \mu\} = P_{Gsym.}\{(ya - b) - \mu\}$$

$$P_{Gasym.}(y) = P_{Gasym.}\{(ya + b) - \mu\} = P_{Gasym.}\{(ya - b) - \mu\}$$

where $y, a, b \in \mathbb{R};\ a \neq 0;\ \mu$ is the mean estimate of $(ya \pm b)$.

Here is a situation (of scaloc-transform distribution) where both the scaloc-invariant and scale-invariant geometrical symmetries are efficient estimators.

$$KC_{Gsym.}(y) = KC_{Gsym.}\{a(y + b) - \mu\} = KC_{Gsym.}\{-a(y + b) - \mu\}$$

$$P_{Gsym.}(y) = P_{Gsym.}\{a(y + b) - \mu\} = P_{Gsym.}\{-a(y + b) - \mu\}$$

$$P_{Gasym.}(y) = P_{Gasym.}\{a(y + b) - \mu\} = P_{Gasym.}\{-a(y + b) - \mu\}$$

where $y, a, b \in \mathbb{R};\ a \neq 0;\ \mu$ is the mean estimate of $\pm a(y + b)$.

vi. Geometrical symmetry is population-independent and variant to sample size or multiple repeats of a univariate dataset.

Supposed we have a c duplicate of a variable $y = (y_1, y_2, y_3, \ldots, y_n)$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Gsym.}(y) \neq KC_{Gsym.}([y] * c)$$

$$P_{Gsym.}(y) \neq P_{Gsym.}([y] * c)$$

$$P_{Gasym.}(y) \neq P_{Gasym.}([y] * c)$$

where $x \in \mathbb{R}; c \in \mathbb{N}$.

vii. Geometrical symmetry is a variant to sample size or multiple repeats of multivariate datasets.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

Supposed we have a c duplicate of variables $x = (x_1, x_2, x_3, \ldots., x_n)$, $y = (y_1, y_2, y_3, \ldots., y_n)$, $z = (z_1, z_2, z_3, \ldots., z_n)$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Gsym.}([[x],[y],[z]]) \neq KC_{Gsym.}([[x]*c,[y]*c,[z]*c])$$
$$P_{Gsym.}([[x],[y],[z]]) \neq P_{Gsym.}([[x]*c,[y]*c,[z]*c])$$
$$P_{Gasym.}([[x],[y],[z]]) \neq P_{Gasym.}([[x]*c,[y]*c,[z]*c])$$

where $x, y, z \in \mathbb{R}; c \in \mathbb{N}$.

2.2.5. Python code

Get the python code at:

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/symmetry_estimators.ipynb

Input guide: **symmetry***([data, centering, ordering, print])*

Input options:

- ✓ **for data:** *list of numerical values from a set of real numbers*.
- ✓ **for centering:** "*allow*", or "*never*".
- ✓ **for ordering:** "*ascend*", "*descend*", or "**never**".
- ✓ **for print:** "*kc*", "*psym*", "*pasym*", "*kcalt1*", "*kcalt2*", or "*kcalt*".

Examples:

print("*Kabirian coefficient =*", **symmetry***([[2,4,6,8,4,2,4,5,6], "centering:allow", "ordering:never", "print:kc"]))*

print("Probability of symmetry =", **symmetry***([[2,4,6,8,4,2,4,5,6], "centering:allow", "ordering:never", "print:psym"]))*

print("Probability of asymmetry =", **symmetry***([[2,4,6,8,4,2,4,5,6], "centering:allow", "ordering:never", "print:pasym"]))*

print("Alt1. Kabirian coefficient =", **symmetry***([[2,4,6,8,4,2,4,5,6], "centering:allow", "ordering:never", "print:kcalt1"]))*

print("Alt2. Kabirian coefficient =", **symmetry***([[2,4,6,8,4,2,4,5,6], "centering:allow", "ordering:never", "print:kcalt2"]))*

print("Alt. Kabirian coefficient =", **symmetry***([[2,4,6,8,4,2,4,5,6], "centering:allow", "ordering:never", "print:kcalt"]))*

2.2.6. Drawbacks and limitations of geometrical symmetry

The following are some of the identified drawbacks and limitations of geometrical symmetry:

i. The given random ordering (sequence) of elements of the list of the variable(s) should be preserved, otherwise, a conceptual ordering has to be established.

ii. The two possible Kabirian bi-coefficients do not function on the same optinalytic scale. For comparison of results, estimates with the mixed Kabirian coefficients should either be translated forward or otherwise uniformed by backward alternate translation.

2.3. Proposal 3: Statistical Mirroring

2.3.1. Definition

Statistical mirroring is the measure of deviation or proximity of data points from a defined location of a given distribution. Under Kabirian-based optinalysis, statistical mirroring is the isoreflectivity of data points to a defined statistical mirror (i.e., a defined and amplified location estimate of the distribution through a defined length). Statistical mirroring refers to the theoretical ordering, with or without centering the data and optinalysing the established isoreflective pair for a given variable.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

2.3.2. Computational steps and algorithmic procedure

Suppose we have a set of variables $X = (x_1, x_3, x_2, \ldots\ldots, x_n)$. Let the order of algorithmic transformations t_c and t_o as centering and ordering of the data X respectively. The optinalysis-based statistical mirroring implies the following steps:

First step 1: Centering the data X (i.e., location removal). This is optional, depending on the task.

$$t_c(X) = (x_1, x_2, x_3, \ldots\ldots, x_n)$$

Second step 2: Establish a theoretical order for the $t_c(X)$ variable. Note that numerical values are theoretically arranged in ascending or descending order.

$$t_{c,o}(X) = (x_1 \leq, x_2, \leq\ x_3, \leq. \ldots\ldots, \leq x_n)$$

or alternatively

$$t_{c,o}(X) = (x_1 \geq, x_2, \geq x_3, \geq. \ldots\ldots, \geq x_n)$$

Second step 3: Design an efficient statistical mirror. A statistical mirror refers to a defined and amplified location estimate (e.g., mean, median, maximum, range, etc) of the distribution through a defined length. Different types of statistical mirrors can be designed, but the choice depends on the task to be performed.

$$P = (p_1, p_2, p_3, \ldots\ldots, p_n)$$

Third step 4: Choose an efficient pairing style (reflection) and establish the isoreflective pair between $t_{c,o}(X)$ onto P about δ. For instance,

Head-to-head pairing or reflection is given as:

$$\left(t_{c,o}(\vec{X})\middle|\overleftarrow{P}\right) = (x_1 \leq, x_2, \leq x_3, \leq. \ldots\ldots, \leq x_n, \delta, p_n, \ldots\ldots, p_3, p_2, p_1)$$

$$\left(t_{c,o}(\vec{X})\middle|\overleftarrow{P}\right) = (x_1 \geq, x_2, \geq x_3, \geq. \ldots\ldots, \geq x_n, \delta, p_n, \ldots\ldots, p_3, p_2, p_1)$$

Tail-to-tail pairing or reflection is given as:

$$\left(t_{c,o}(\vec{X})\middle|\overleftarrow{P}\right) = (x_n, \ldots\ldots \leq, x_3, \leq x_2, \leq x_1, \delta, p_1, p_2, p_3, \ldots\ldots, p_n)$$

$$\left(t_{c,o}(\vec{X})\middle|\overleftarrow{P}\right) = (x_n, \ldots\ldots \geq, x_3, \geq x_2, \geq x_1, \delta, p_1, p_2, p_3, \ldots\ldots, p_n)$$

Fifth step 5: Optinalyse (by Kabirian-based optinalysis [3]) the isoreflective pair of $\left(t_{c,o}(\vec{X})\middle|\overleftarrow{P}\right)$ or $\left(t_{c,o}(\vec{X})\middle|\overleftarrow{P}\right)$ about a mid-point δ.

$$f\colon t_{c,o}(X) \overset{\delta}{\twoheadrightarrow} P \twoheadrightarrow R$$

$$f\colon \begin{bmatrix} t_{c,o}(X) = (x_1 \leq, x_2, \leq x_3, \leq. \ldots\ldots, \leq x_n) & \overset{\delta}{\twoheadrightarrow} & P = (p_n, \ldots\ldots, p_3, p_2, p_1) \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, r_2, r_3, \ldots\ldots, r_n, & r_{n+1}, & r_{n+2}, r_{n+3}, r_{n+4}, \ldots\ldots, r_{2n+1}) \end{bmatrix}$$

Or in an alternative way,

$$f\colon \begin{bmatrix} t_{c,o}(X) = (x_1 \geq, x_2, \geq x_3, \geq. \ldots\ldots, \geq x_n) & \overset{\delta}{\twoheadrightarrow} & P = (p_n, \ldots\ldots, p_3, p_2, p_1) \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, r_2, r_3, \ldots\ldots, r_n, & r_{n+1}, & r_{n+2}, r_{n+3}, r_{n+4}, \ldots\ldots, r_{2n+1}) \end{bmatrix}$$

Such that $X, P\ \&\ R \in \mathbb{R}$; $r_1 \neq 0$; $n \in \mathbb{N}$; and $t_{c,o}(X)\ \&\ P$ are isoreflective pairs. $\delta = 0$ is by default operation, except under optinalytic normalization. P could be the average (mean), median, mode, maximum, minimum, etc of the distribution.

By Kabirian-based optinalysis [3], the Kabirian coefficient of proximity/similarity ($KC_{Sprox.}$) and its derivatives satisfied the Y-rule of Kabirian-based isomorphic optinalysis.

$$\begin{matrix} KC1_{Sprox.}(X,P) \\ \ldots \\ KC2_{Sprox.}(P,X) \end{matrix} \rightleftharpoons P_{Sprox.}(X,P) = P_{Sprox.}(P,X) \rightleftharpoons P_{Sdev.}(X,P) = P_{Sdev.}(P,X)$$

where $X, P \in \mathbb{R}$.

The two possible Kabirian bi-coefficients ($KC1_{Sprox.}\ \&\ KC2_{Sprox.}$) function on two different, but inverse optinalytic scales.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

2.3.3. Different approaches to statistical mirroring

Suppose that $f: X \overset{\delta}{\twoheadrightarrow} P \twoheadrightarrow R$, where X is the observations, P is the statistical mirror, R is the optiscale. It is called:

a. *A statistical meanic mirroring*, if $P = M_n$ (i.e., the mean of the distribution of X). It is the measure of proximity or deviation (how close or far) the data points are from its estimate of the mean.
b. *A statistical medianic mirroring*, if $P = M_d$ (i.e., the median of the distribution of X). It is the measure of proximity or deviation (how close or far) the data points are from its median estimate.
c. *A statistical modalic mirroring*, if $P = M_o$ (i.e., the mode of the distribution of X). It is the measure of proximity or deviation (how close or far) the data points are from its mode estimate.
d. *A statistical maximalic mirroring*, if $P = M_x$ (i.e., the maximum of the distribution of X). It is the measure of proximity or deviation (how close or far) the data points are from their maximum estimate.
e. *A statistical minimalic mirroring*, if $P = M_y$ (i.e., the minimum of the distribution of X). It is the measure of proximity or deviation (how close or far) the data points are from their minimum estimate.
f. *A statistical rangic mirroring*, if $P = M_{x-y}$ (i.e., the range of the distribution of X). It is the measure of proximity or deviation (how close or far) the data points are from their range estimate.
g. *A statistical reference mirroring*, if $P = R_f$ (i.e., a reference value outside the distribution of X). It is the measure of proximity or deviation (how close or far) the data points are from their reference estimate value.
h. *An endo-statistical mirroring*, if $P = M_n, M_d, M_o, M_x, M_y, M_{x-y}$ (of a location estimate).
i. *An exo-statistical mirroring*, if $P \neq M_n, M_d, M_o, M_x, M_y, M_{x-y}$ (of a location estimate).

2.3.4. Scale and scaloc-invariant statistical mirroring

Statistical mirroring is called scale invariance if the efficient location parameter is not removed from the variable, while it is called scaloc-invariant if the efficient location parameter is removed from the variable.

2.3.5. General properties of statistical mirroring

i. It is based on the entire observations of variables, unlike percentile-based or decile-based statistics. Therefore, extreme maximum and minimum values are not discarded or trimmed.
ii. It applies to variable(s) from the set of real numbers (such as discrete or continuous variables containing either or both negative and positive values).
iii. It involves a measure of spread around a defined location estimate (such as mean, median, maximum, minimum, and range) and other attributes.
iv. Statistical mirroring is scale-invariant (i.e., robust to scale, and unitless) estimator.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

Supposed we have an a scaling of a variable $x = (x_1, x_2, x_3, \ldots, x_n)$ and its statistical mirror as $p = (p_1, p_2, p_3, \ldots, p_n)$, where p is the amplified estimate or reference of the associated dataset.

$$KC_{Sprox.}(x, p) = KC_{Sprox.}(ax, p) = KC_{Sprox.}(-ax, p)$$
$$P_{Sprox.}(x, p) = P_{Sprox.}(ax, p) = P_{Sprox.}(-ax, p)$$
$$P_{Sdev.}(x, p) = P_{Sdev.}(ax, p) = P_{Sdev.}(-ax, p)$$

where $x, p, a \in \mathbb{R}; a \neq 0$.

i. Statistical mirroring is a location-invariant estimator, only if, the efficient location parameter is removed from the variable. For instance, taking the absolute distances from the mean.

Supposed we have a variable $x = (x_1, x_2, x_3, \ldots, x_n)$, then it shifted by a b location and returned as $\hat{x} = (x_1 + b, x_2 + b, x_3 + b, \ldots, x_n + b)$, and its statistical mirror as $p = (p_1, p_2, p_3, \ldots, p_n)$, where p is the amplified estimate or reference of the associated dataset. The location-invariance implies:

$$KC_{Ssym.}(x, p) = KC_{Ssym.}(\hat{x} - \mu, p) \neq KC_{Ssym.}(\hat{x}, p)$$
$$P_{Ssym.}(x, p) = P_{Ssym.}(\hat{x} - \mu, p) \neq P_{Ssym.}(\hat{x}, p)$$
$$P_{Sasym..}(x, p) = P_{Sasym.}(\hat{x} - \mu, p) \neq P_{Sasym.}(\hat{x}, p)$$

where $x, \hat{x}, p, b \in \mathbb{R}$; μ is the mean estimate of $\hat{x}$.

ii. Where location and scale properties are combined (i.e., scaloc-transform distribution), statistical mirroring is its invariant scaloc-invariant and/or scale-invariant estimator.

Supposed we have an a scaling and b location shift of a variable $x = (x_1, x_2, x_3, \ldots, x_n)$ and its statistical mirror as $p = (p_1, p_2, p_3, \ldots, p_n)$, where p is the amplified estimate or reference of the associated dataset.

Here is a situation (of scaloc-transform distribution) where only the scaloc-invariant statistical mirroring is an efficient estimator.

$$KC_{Sprox.}(x, p) = KC_{Sprox.}\{(xa + b) - \mu, p\} = KC_{Sprox.}\{(xa - b) - \mu, p\}$$
$$P_{Sprox.}(x, p) = P_{Sprox.}\{(xa + b) - \mu, p\} = P_{Sprox.}\{(xa - b) - \mu, p\}$$
$$P_{Sdev.}(x, p) = P_{Sdev.}\{(xa + b) - \mu, p\} = P_{Sdev.}\{(xa - b) - \mu, p\}$$

where $x, p, a, b \in \mathbb{R}$; $a \neq 0$; μ is the mean estimate of $(xa \pm b)$; and the resultant effect is a location property.

Here is a situation (of scaloc-transform distribution) where both the scaloc-invariant and scale-invariant statistical mirroring are efficient estimators.

$$KC_{Sprox.}(x, p) = KC_{Sprox.}\{a(x + b) - \mu, p\} = KC_{Sprox.}\{-a(x + b) - \mu, p\}$$
$$P_{Sprox.}(x, p) = P_{Sprox.}\{a(x + b) - \mu, p\} = P_{Sprox.}\{-a(x + b) - \mu, p\}$$
$$P_{Sdev.}(x, p) = P_{Sdev.}\{a(x + b) - \mu, p\} = P_{Sdev.}\{-a(x + b) - \mu, p\}$$

where $x, p, a, b \in \mathbb{R}$; $a \neq 0$; μ is the mean estimate of $\pm a(x + b)$; and the resultant effect is a scale property.

iii. Statistical mirroring is a variant of pericentral rotation (alternate reflection), except for meanic statistical mirroring.

Supposed we have a statistically ordered variable $x = (x_1, x_2, x_3, \ldots, x_n)$ and its statistical mirror as $p = (p_1, p_2, p_3, \ldots, p_n)$, where p is the amplified estimate or reference of the associated dataset.

$$KC_{Sprox.}(\tilde{x})|\vec{p}) \neq KC_{Sprox.}(\vec{x}|\tilde{p})$$
$$P_{Sprox.}(\tilde{x})|\vec{p}) \neq P_{Sprox.}(\vec{x}|\tilde{p})$$
$$P_{Sdev.}(\tilde{x})|\vec{p}) \neq P_{Sdev.}(\vec{x}|\tilde{p})$$

An exception is the case of meanic statistical mirroring

$$KC_{Smnprox.}(\tilde{x})|\vec{p}) = KC_{Smnprox.}(\vec{x}|\tilde{p})$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

$$P_{Smnprox.}(\tilde{x})|\tilde{p}) = P_{Smnprox.}(\vec{x}|\tilde{p})$$
$$P_{Smndev.}(\tilde{x})|\tilde{p}) = P_{Smndev.}(\vec{x}|\tilde{p})$$

where $x, p \in \mathbb{R}$.

iv. Statistical mirroring is invariant to sample size or multiple repeats of a univariate dataset. But the univariate sample size invariance is effective to $P_{Sprox.}$ and $P_{Sdev.}$, and not to $KC_{Sprox.}$.

Supposed we have a c duplicate of variable $x = (x_1, x_2, x_3, \ldots., x_n)$ and its statistical mirror as $p = (p_1, p_2, p_3, \ldots., p_n)$, where p is the amplified estimate or reference of the associated dataset.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Sprox.}(x, p) \neq KC_{Sprox.}([x] * c, p)$$
$$P_{Sprox.}(x, p) = P_{Sprox.}([x] * c, p)$$
$$P_{Sdev.}(x, p) = P_{Sdev.}([x] * c, p)$$

where $x, p \in \mathbb{R}; c \in \mathbb{N}$

v. Statistical mirroring is variant to sample size or multiple repeats of multivariate datasets, except for meanic statistical mirroring, and is effective to $P_{Smnprox.}$ and $P_{Smndev.}$, and not to $KC_{Smnprox.}$.

Supposed we have a c duplicate of variables $x = (x_1, x_2, x_3, \ldots., x_n)$, $y = (y_1, y_2, y_3, \ldots., y_n)$, $z = (z_1, z_2, z_3, \ldots., z_n)$, and their statistical mirror as $p = (p_1, p_2, p_3, \ldots., p_n)$, where p is the amplified mean estimate of the associated dataset.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Sprox.}([[x], [y], [z]], p) \neq KC_{Sprox.}([[x], [y], [z]] * c, p)$$
$$P_{Sprox.}([[x], [y], [z]], p) \neq P_{Sprox.}([[x], [y], [z]] * c, p)$$
$$P_{Sdev.}([[x], [y], [z]], p) \neq P_{Sdev.}([[x], [y], [z]] * c, p)$$

An exception is the case of meanic statistical mirroring

$$KC_{Smnprox.}([[x], [y], [z]], p) \neq KC_{Smnprox.}([[x], [y], [z]] * c, p)$$
$$P_{Smnprox.}([[x], [y], [z]], p) = P_{Smnprox.}([[x], [y], [z]] * c, p)$$
$$P_{Smndev.}([[x], [y], [z]], p) = P_{Smndev.}([[x], [y], [z]] * c, p)$$

where $p, q, r, s \in \mathbb{R}; c \in \mathbb{N}$.

2.3.6. Python code

Get the python code at;

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/mirroring_estimators.ipynb

Input guide: **mirroring**(*[data, principal_value, centering, ordering, pairing, print]*)

Input options:

- ✓ ***for data:*** *list of numerical values from a set of real numbers.*
- ✓ ***for principal_value:*** *"mean", "median", "mode", "max", "min", "range", or numerical_value,*
- ✓ ***for centering:*** *"allow", or "never".*
- ✓ ***for ordering:*** *"ascend", "descend", or "never".*
- ✓ ***for pairing:*** *"H_H", or "T_T".*
- ✓ ***for print:*** *"kc", "pprox", "pdev", "kcalt1", "kcalt2", or "kcalt".*

Example 1:

data = [2,-4,6.12,8,4,0.2,4,5,6,24,12,-2,0,-3,4,-1.05,13.33]

print("Kabirian coefficient =", **mirroring**(*[data, "principal_value:mean", "centering:allow", "ordering:ascend", "pairing:H_H", "print:kc"])*)

print("Probability of proximity =", **mirroring**(*[data, "principal_value:mean", "centering:allow", "ordering:ascend", "pairing:H_H", "print:pprox"])*)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

*print("Probability of deviation =", **mirroring**([data, "principal_value:mean", "centering:allow", "ordering:ascend", "pairing:H_H", "print:pdev"]))*

*print("Alt1. Kabirian coefficient =", **mirroring**([data, "principal_value:mean", "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt1"]))*

*print("Alt2. Kabirian coefficient =", **mirroring**([data, "principal_value:mean", "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt2"]))*

*print("Alt. Kabirian coefficient =", **mirroring**([data, "principal_value:mean", "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt"]))*

Example 2:

*print("Kabirian coefficient =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:ascend", "pairing:H_H", "print:kc"]))*

*print("Probability of proximity =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:ascend", "pairing:H_H", "print:pprox"]))*

*print("Probability of deviation =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:ascend", "pairing:H_H", "print:pdev"]))*

*print("Alt1. Kabirian coefficient =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt1"]))*

*print("Alt2. Kabirian coefficient =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt2"]))*

*print("Alt. Kabirian coefficient =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt"]))*

Example 3:

*print("Kabirian coefficient =", **mirroring**([data, 5.123, "centering:allow", "ordering:ascend", "pairing:H_H", "print:kc"]))*

*print("Probability of proximity =", **mirroring**([data, 5.123, "centering:allow", "ordering:ascend", "pairing:H_H", "print:pprox"]))*

*print("Probability of deviation =", **mirroring**([data, 5.123, "centering:allow", "ordering:ascend", "pairing:H_H", "print:pdev"]))*

*print("Alt1. Kabirian coefficient =", **mirroring** ([data, 5.123, "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt1"]))*

*print("Alt2. Kabirian coefficient =", **mirroring** ([data, 5.123, "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt2"]))*

*print("Alt. Kabirian coefficient =", **mirroring** ([data, 5.123, "centering:allow", "ordering:ascend", "pairing:H_H", "print:kcalt"]))*

2.3.7. Drawbacks and limitations of statistical mirroring

The following are some of the identified drawbacks and limitations of statistical mirroring:

i. The given random ordering of elements of the list of the variable(s) is not preserved, thus an efficient theoretical ordering (i.e., ascend or descend sorting) has to be adopted or used.

ii. A suitable and efficient pairing style or alternate reflection has to be chosen and adopted for repeatability and comparison of results. This excludes only statistical meanic mirroring.

iii. The two possible Kabirian bi-coefficients do not function on the same optinalytic scale. For comparison of results, estimates with the mixed Kabirian coefficients should either be translated forward or otherwise uniformed by backward alternate translation.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

2.4. Proposal 4: Geometrical Mirroring

2.4.1. Definition

Geometrical mirroring is the measure of deviation or proximity of geometrical data points from a defined location of a given sequence. Under Kabirian-based optinalysis, geometrical mirroring is the isoreflectivity of data points to a defined geometrical mirror (i.e., a defined and amplified location estimate of the sequence through a defined length). Geometrical mirroring refers to the conceptual ordering, with or without centering the data and optinalysing the established isoreflective pair for a given variable. Geometrical mirroring is a method of shape and sequence analysis

2.4.2. Computational steps and algorithmic procedure

Suppose we have a sequence of variable $Y = (y_2, y_3, y_1, \ldots\ldots, y_n)$. Let the order of algorithmic transformations t_c and t_o as centering and ordering of the data X respectively. The optinalysis-based geometrical mirroring implies the following steps:

First step 1: Centering the data Y (i.e., location removal). This is optional, depending on the task.

$$t_c(Y) = (y_2, y_3, y_1, \ldots\ldots, y_n)$$

Second step 2: Establish a conceptual order of sequence for the $t_c(Y)$ variable.

$$t_{c,o}(Y) = (y_1, y_2, y_3, \ldots\ldots, y_n)$$

Third step 3: Design an efficient geometrical mirror. A geometrical mirror refers to a defined and amplified estimate (e.g., mean, median, maximum, range, etc) of the distribution or others through a defined length. Different types of geometrical mirrors can be designed, but the choice depends on the task to be performed.

$$P = (p_1, p_2, p_3, \ldots\ldots, p_n)$$

Third step 4: Choose an efficient pairing style (reflection) and establish the isoreflective pair between $t_{c,o}(Y)$ and P about δ. For instance,

Head-to-head pairing or reflection is given as:

$$\left(t_{c,o}(\overleftarrow{Y})\middle|\vec{P}\right) = (y_1, y_2, y_3, \ldots\ldots, y_n, \delta, p_n, \ldots\ldots, p_3, p_2, p_1)$$

Tail-to-tail pairing or reflection is given as:

$$\left(t_{c,o}(\vec{Y})\middle|\overleftarrow{P}\right) = (y_n, \ldots\ldots, y_3, y_2, y_1, \delta, p_1, p_2, p_3, \ldots\ldots, p_n)$$

Fifth step 5: Optinalyse (by Kabirian-based optinalysis [3]) the isoreflective pair of $\left(t_{c,o}(\overleftarrow{Y})\middle|\vec{P}\right)$ or $\left(t_{c,o}(\vec{Y})\middle|\overleftarrow{P}\right)$ about a mid-point δ.

$$f\colon t_{c,o}(Y) \overset{\delta}{\twoheadrightarrow} P \twoheadrightarrow R$$

$$f\colon \begin{bmatrix} t_{c,o}(Y) = (y_n, \ldots\ldots, y_3, y_2, y_1) & \overset{\delta}{\twoheadrightarrow} & P = (p_1, p_2, p_3, \ldots\ldots p_n) \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, r_2, r_3, \ldots\ldots, r_n, & r_{n+1}, & r_{n+2}, r_{n+3}, r_{n+4}, \ldots\ldots, r_{2n+1}) \end{bmatrix}$$

Such that $Y, P \,\&\, R \in \mathbb{R}$; $R_1 \neq 0$; $n \in \mathbb{N}$; and $t_{c,o}(Y) \,\&\, P$ are isoreflective pairs. $\delta = 0$ is by default operation, except under optinalytic normalization. P could be the average (mean), median, mode, maximum, minimum, etc of the distribution.

By Kabirian-based optinalysis [3], the Kabirian coefficient of proximity/similarity ($KC_{Gprox.}$) and its derivatives satisfied the Y-rule of Kabirian-based isomorphic optinalysis.

$$\begin{matrix} KC1_{Gprox.}(Y,P) \\ \ldots \\ KC2_{Gprox.}(P,Y) \end{matrix} \rightleftharpoons P_{Gprox.}(Y,P) = P_{Gprox.}(P,Y) \rightleftharpoons P_{Gdev.}(Y,P) = P_{Gdev.}(P,Y)$$

where $Y, P \in \mathbb{R}$.

The two possible Kabirian bi-coefficients ($KC1_{Gprox.} \,\&\, KC2_{Gprox.}$) function on two different, but inverse optinalytic scales.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

2.4.3. Different approaches to geometrical mirroring

Suppose that $f: Y \overset{\delta}{\twoheadrightarrow} P \twoheadrightarrow R$, where Y is the observations, P is the geometrical mirror, R is the optiscale. It is called:

a. *A geometrical meanic mirroring*, if $P = M_n$ (i.e., the mean of the distribution of Y). It is the measure of proximity or deviation (how close or far) the data points are from its mean estimate.
b. *A geometrical medianic mirroring*, if $P = M_d$ (i.e., the median of the distribution of Y). It is the measure of proximity or deviation (how close or far) the data points are from its median estimate.
c. *A geometrical modalic mirroring*, if $P = M_o$ (i.e., the mode of the distribution of Y). It is the measure of proximity or deviation (how close or far) the data points are from its mode estimate.
d. *A geometrical maximalic mirroring*, if $P = M_x$ (i.e., the maximum of the distribution of Y). It is the measure of proximity or deviation (how close or far) the data points are from their maximum estimate.
e. *A geometrical minimalic mirroring*, if $P = M_y$ (i.e., the minimum of the distribution of Y). It is the measure of proximity or deviation (how close or far) the data points are from their minimum estimate.
f. *A geometrical rangic mirroring*, if $P = M_{x-y}$ (i.e., the range of the distribution of Y). It is the measure of proximity or deviation (how close or far) the data points are from their range estimate.
g. *A geometrical reference mirroring*, if $P = R_f$ (i.e., a reference value outside the distribution of Y). It is the measure of proximity or deviation (how close or far) the data points are from their reference estimate value.
h. *An endo-geometrical mirroring*, if $P = M_n, M_d, M_o, M_x, M_y, M_{x-y}$ (of a location estimate).
i. *An exo-geometrical mirroring*, if $P \neq M_n, M_d, M_o, M_x, M_y, M_{x-y}$ (of a location estimate).

2.4.4. Scale and scaloc-invariant geometrical mirroring

Geometrical mirroring is called scale invariance if the efficient location parameter is not removed from the variable, while it is called scaloc-invariant if the efficient location parameter is removed from the variable.

2.4.5. General properties of geometrical mirroring

i. It is based on the entire observations of variables, unlike percentile-based or decile-based statistics. Therefore, extreme maximum and minimum values are not discarded or trimmed.
ii. It applies to variable(s) from the set of real numbers (such as discrete or continuous variables containing either or both negative and positive values),
iii. It involves a measure of spread around a defined location estimate (such as mean, median, maximum, minimum, and range) and other attributes.
iv. Geometrical mirroring is scale-invariant (i.e., robust to scale, and unitless) estimator.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

Supposed we have an a scaling of a variable $y = (y_1, y_2, y_3, \ldots, y_n)$ and its geometrical mirror as $p = (p_1, p_2, p_3, \ldots, p_n)$, where p is the amplified estimate or reference of the associated dataset.

$$KC_{Gprox.}(y, p) = KC_{Gprox.}(ay, p) = KC_{Gprox.}(-ay, p)$$
$$P_{Gprox.}(y, p) = P_{Gprox.}(ay, p) = P_{Gprox.}(-ay, p)$$
$$P_{Gdev.}(y, p) = P_{Gdev.}(ay, p) = P_{Gdev.}(-ay, p)$$

where $y, p, a \in \mathbb{R}; a \neq 0$.

i. Geometrical mirroring is a location-invariant estimator, only if, the efficient location parameter is removed from the variable. For instance, taking the absolute distances from the mean.

Supposed we have a variable $y = (y_1, y_2, y_3, \ldots, y_n)$, then it shifted by a b location and returned as $\hat{y} = (y_1 + b, y_2 + b, y_3 + b, \ldots, y_n + b)$, and its statistical mirror as $p = (p_1, p_2, p_3, \ldots, p_n)$, where p is the amplified estimate or reference of the associated dataset. The location-invariance implies:

$$KC_{Gsym.}(y, p) = KC_{Gsym.}(\hat{y} - \mu, p) \neq KC_{Gsym.}(\hat{y}, p)$$
$$P_{Gsym.}(y, p) = P_{Gsym.}(\hat{y} - \mu, p) \neq P_{Gsym.}(\hat{y}, p)$$
$$P_{Gasym..}(y, p) = P_{Gasym.}(\hat{y} - \mu, p) \neq P_{Gasym.}(\hat{y}, p)$$

where $y, \hat{y}, p, b \in \mathbb{R}$; μ is the mean estimate of $\hat{y}$.

ii. Where location and scale properties are combined (i.e., scaloc-transform distribution), geometrical mirroring is its invariant scaloc-invariant and/or scale-invariant estimator.

Supposed we have an a scaling and b location shift of a variable $y = (y_1, y_2, y_3, \ldots, y_n)$ and its geometrical mirror as $p = (p_1, p_2, p_3, \ldots, p_n)$, where p is the amplified estimate or reference of the associated dataset.

Here is a situation (of scaloc-transform distribution) where only the scaloc-invariant geometrical mirroring is an efficient estimator.

$$KC_{Gprox.}(y, p) = KC_{Gprox.}\{(ya + b) - \mu, p\} = KC_{Gprox.}\{(ya - b) - \mu, p\}$$
$$P_{Gprox.}(y, p) = P_{Gprox.}\{(ya + b) - \mu, p\} = P_{Gprox.}\{(ya - b) - \mu, p\}$$
$$P_{Gdev.}(y, p) = P_{Gdev.}\{(ya + b) - \mu, p\} = P_{Gdev.}\{(ya - b) - \mu, p\}$$

where $y, p, a, b \in \mathbb{R}$; $a \neq 0$; μ is the mean estimate of $(ya \pm b)$; and the resultant effect is a location property.

Here is a situation (of scaloc-transform distribution) where both the scaloc-invariant and scale-invariant geometrical mirroring are efficient estimators.

$$KC_{Gprox.}(y, p) = KC_{Gprox.}\{a(y + b) - \mu, p\} = KC_{Gprox.}\{-a(y + b) - \mu, p\}$$
$$P_{Gprox.}(y, p) = P_{Gprox.}\{a(y + b) - \mu, p\} = P_{Gprox.}\{-a(y + b) - \mu, p\}$$
$$P_{Gdev.}(y, p) = P_{Gdev.}\{a(y + b) - \mu, p\} = P_{Gdev.}\{-a(y + b) - \mu, p\}$$

where $y, p, a, b \in \mathbb{R}$; $a \neq 0$; μ is the mean estimate of $\pm a(y + b)$; and the resultant effect is a scale property.

iii. Geometrical mirroring is variant to pericentral rotation (alternate reflection) except for meanic geometrical mirroring.

Supposed we have a geometrically ordered variable $y = (y_1, y_2, y_3, \ldots, y_n)$ and its statistical mirror as $p = (p_1, p_2, p_3, \ldots, p_n)$, where p is the amplified estimate or reference of the associated dataset.

$$KC_{Gprox.}(\tilde{y})|\vec{p}) \neq KC_{Gprox.}(\vec{y}|\tilde{p})$$
$$P_{Gprox.}(\tilde{y})|\vec{p}) \neq P_{Gprox.}(\vec{y}|\tilde{p})$$
$$P_{Gdev.}(\tilde{y})|\vec{p}) \neq P_{Gdev.}(\vec{y}|\tilde{p})$$

An exception is the case of meanic statistical mirroring

$$KC_{Gmnprox.}(\tilde{y})|\vec{p}) = KC_{Gmnprox.}(\vec{y}|\tilde{p})$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

$$P_{Gmnprox.}(\tilde{y})|\vec{p}) = P_{Gmnprox.}(\vec{y}|\tilde{p})$$
$$P_{Gmndev.}(\tilde{y})|\vec{p}) = P_{Gmndev.}(\vec{y}|\tilde{p})$$

where $y, p \in \mathbb{R}$.

iv. Geometrical mirroring is population-independent and variant to sample size or multiple repeats of a univariate dataset.

Supposed we have a c duplicate of variable $y = (y_1, y_2, y_3, \ldots., y_n)$ and its geometrical mirror as $p = (p_1, p_2, p_3, \ldots., p_n)$, where p is the amplified estimate or reference of the associated dataset.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Gprox.}(y, p) \neq KC_{Gprox.}([y] * c, p)$$
$$P_{Gprox.}(y, p) \neq P_{Gprox.}([y] * c, p)$$
$$P_{Gdev.}(y, p) \neq P_{Gdev.}([y] * c, p)$$

where $y, p \in \mathbb{R}; c \in \mathbb{N}$.

v. Geometrical mirroring is variant to sample size or multiple repeats of multivariate datasets.

Supposed we have a c duplicate of variables $q = (q_1, q_2, q_3, \ldots., q_n)$, $r = (r_1, r_2, r_3, \ldots., r_n)$, $s = (s_1, s_2, s_3, \ldots., s_n)$, and their geometrical mirror as $p = (p_1, p_2, p_3, \ldots., p_n)$, where p is the amplified estimate or reference of the associated dataset.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Gprox.}([[q], [r], [s]], p) \neq KC_{Gprox.}([[q] * c, [r] * c, [s] * c], p)$$
$$P_{Gprox.}([[q], [r], [s]], p) \neq P_{Gprox.}([[q] * c, [r] * c, [s] * c], p)$$
$$P_{Gdev.}([[q], [r], [s]], p) \neq P_{Gdev.}([[q] * c, [r] * c, [s] * c], p)$$

where $p, q, r, s \in \mathbb{R}; c \in \mathbb{N}$.

2.4.6. Python code

Get the python code at:

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/mirroring_estimators.ipynb

Input guide: **mirroring**(*[data, principal_value, centering, ordering, pairing, print]*)

Input options:

- ✓ ***for data:*** *list of numerical values from a set of real numbers.*
- ✓ ***for principal_value:*** *"mean", "median", "mode", "max", "min", "range", or numerical_value,*
- ✓ ***for centering:*** *"allow", "never".*
- ✓ ***for ordering:*** *"ascend", "descend", or "never".*
- ✓ ***for pairing:*** *"H_H", or "T_T".*
- ✓ ***for print:*** *"kc", "pprox", "pdev", "kcalt1", "kcalt2", or "kcalt".*

Example 1:

data = [2,-4,6.12,8,4,0.2,4,5,6,24,12,-2,0,-3,4,-1.05,13.33]

print("Kabirian coefficient =", **mirroring**(*[data, "principal_value:mean", "centering:allow", "ordering:never", "pairing:H_H", "print:kc"])*)

print("Probability of proximity =", **mirroring**(*[data, "principal_value:mean", "centering:allow", "ordering:never", "pairing:H_H", "print:pprox"])*)

print("Probability of deviation =", **mirroring**(*[data, "principal_value:mean", "centering:allow", "ordering:never", "pairing:H_H", "print:pdev"])*)

print("Alt1. Kabirian coefficient =", **mirroring**(*[data, "principal_value:mean", "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt1"])*)

print("Alt2. Kabirian coefficient =", **mirroring**(*[data, "principal_value:mean", "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt2"])*)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

*print("Alt. Kabirian coefficient =", **mirroring**([data, "principal_value:mean", "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt"]))*

Example 2:

*print("Kabirian coefficient =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:never", "pairing:H_H", "print:kc"]))*

*print("Probability of proximity =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:never", "pairing:H_H", "print:pprox"]))*

*print("Probability of deviation =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:never", "pairing:H_H", "print:pdev"]))*

*print("Alt1. Kabirian coefficient =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt1"]))*

*print("Alt2. Kabirian coefficient =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt2"]))*

*print("Alt. Kabirian coefficient =", **mirroring**([data, "principal_value:max", "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt"]))*

Example 3:

*print("Kabirian coefficient =", **mirroring**([data, 0.123, "centering:never", "ordering:never", "pairing:H_H", "print:kc"]))*

*print("Probability of proximity =", **mirroring**([data, 0.123, "centering:never", "ordering:never", "pairing:H_H", "print:pprox"]))*

*print("Probability of deviation =", **mirroring**([data, 0.123, "centering:never", "ordering:never", "pairing:H_H", "print:pdev"]))*

*print("Alt1. Kabirian coefficient =", **mirroring**([data, 0.123, "centering:never", "ordering:never", "pairing:H_H", "print:kcalt1"]))*

*print("Alt2. Kabirian coefficient =", **mirroring**([data, 0.123, "centering:never", "ordering:never", "pairing:H_H", "print:kcalt2"]))*

*print("Alt. Kabirian coefficient =", **mirroring**([data, 0.123, "centering:never", "ordering:never", "pairing:H_H", "print:kcalt"]))*

2.4.7. Drawbacks and limitations of geometrical mirroring

The following are some of the identified drawbacks and limitations of geometrical mirroring:

i. The given random ordering (sequence) of elements of the list of the variable(s) should be preserved, otherwise, a conceptual ordering has to be established.

ii. A suitable and efficient pairing style or alternate reflection has to be chosen and adopted for repeatability and comparison of results. This excludes only statistical meanic mirroring.

iii. The two possible Kabirian bi-coefficients do not function on the same optinalytic scale. For comparison of results, estimates with the mixed Kabirian coefficients should either be translated forward or otherwise uniformed by backward alternate translation.

2.5. Proposal 5: Statistical Pairwise Comparison

2.5.1. Definition

A statistical pairwise comparison between two variables, under Kabirian-based optinalysis, is their isoreflectivity in a statistical order. The statistical pairwise comparison refers to the theoretical ordering, with or without centering and descaling the data and optinalysing the established isoreflective pair for the given variables.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

2.5.2. Computational steps and algorithmic procedure

Suppose we have a set of variables $X = (x_2, x_3, x_1, \ldots\ldots, x_n)$ and $Y = (y_3, y_1, y_2, \ldots\ldots, y_n)$. Let the order of algorithmic transformations t_c, t_o and t_d as centering, ordering, and descaling of the data X respectively. The optinalysis-based statistical pairwise comparison implies the following steps:

First step 1: Centering the data X and Y (i.e., location removal). This is optional, depending on the task.

$$t_c(X) = (x_2, x_3, x_1, \ldots\ldots, x_n)$$
$$t_c(Y) = (y_3, y_1, y_2, \ldots\ldots, y_n)$$

Second step 2: Establish a theoretical order for the $t_c(X)$ and $t_c(Y)$ variables. Note that numerical values are theoretically arranged in ascending or descending order.

$$t_{c,o}(X) = (x_1 \leq, x_2, \leq\ x_3, \leq \ldots\ldots, \leq x_n)$$
$$t_{c,o}(Y) = (y_1 \leq, y_2, \leq\ y_3, \leq \ldots\ldots, \leq y_n)$$

or alternatively

$$t_{c,o}(X) = (x_1 \geq, x_2, \geq x_3, \geq \ldots\ldots, \geq x_n)$$
$$t_{c,o}(Y) = (y_1 \geq, y_2, \geq y_3, \geq \ldots\ldots, \geq y_n)$$

Second step 2: Descaling the theoretically ordered sequence of $t_{c,o}(X)$ and $t_{c,o}(Y)$ variables.

$$t_{c,o,d}(X) = (x_1 \leq, x_2, \leq\ x_3, \leq \ldots\ldots, \leq x_n)$$
$$t_{c,o,d}(Y) = (y_1 \leq, y_2, \leq\ y_3, \leq \ldots\ldots, \leq y_n)$$

or alternatively

$$t_{c,o,d}(X) = (x_1 \geq, x_2, \geq x_3, \geq \ldots\ldots, \geq x_n)$$
$$t_{c,o,d}(Y) = (y_1 \geq, y_2, \geq y_3, \geq \ldots\ldots, \geq y_n)$$

Second step 3: Choose an efficient pairing style (reflection) and establish the isoreflective pair between $t_{c,o,d}(X)$ and $t_{c,o,d}(Y)$ about δ. For instance,

Head-to-head pairing or reflection is given as:

$$\left(t_{c,o,d}(\overleftarrow{X})\middle|t_{c,o,d}(\vec{Y})\right) = (x_1 \leq, x_2, \leq x_3, \leq \ldots\ldots, \leq x_n, \delta, y_n \leq \ldots\ldots, \leq y_3, \leq y_2, \leq y_1)$$
$$\left(t_{c,o,d}(\overleftarrow{X})\middle|t_{c,o,d}(\vec{Y})\right) = (x_1 \geq, x_2, \geq x_3, \geq \ldots\ldots, \geq x_n, \delta, y_n \geq \ldots\ldots, \geq y_3, \geq y_2, \geq y_1)$$

Tail-to-tail pairing or reflection is given as:

$$\left(t_{c,o,d}(\vec{X})\middle|t_{c,o,d}(\overleftarrow{Y})\right) = (x_n, \ldots\ldots \leq, x_3, \leq x_2, \leq x_1, \delta, y_1 \leq, y_2, \leq y_3, \leq \ldots\ldots, \leq y_n)$$
$$\left(t_{c,o,d}(\vec{X})\middle|t_{c,o,d}(\overleftarrow{Y})\right) = (x_n, \ldots\ldots \geq, x_3, \geq x_2, \geq x_1, \delta, y_1 \geq, y_2, \geq y_3, \geq \ldots\ldots, \geq y_n)$$

Third step 4: Optinalyse (by Kabirian-based optinalysis [3]) the isoreflective pair $\left(t_{c,o,d}(\overleftarrow{X})\middle|t_{c,o,d}(\vec{Y})\right)$ or any other suitable isoreflective pair about a mid-point δ.

$$f\colon t_{c,o,d}(\overleftarrow{X}) \overset{\delta}{\twoheadrightarrow} t_{c,o,d}(\vec{Y}) \twoheadrightarrow R$$

$$f\colon \begin{bmatrix} t_{c,o,d}(\overleftarrow{X}) = (x_1 \leq, x_2, \leq x_3, \leq \ldots\ldots, \leq x_n) & \overset{\delta}{\twoheadrightarrow} & t_{c,o,d}(\vec{Y}) = (y_n \leq \ldots\ldots, \leq y_3, \leq y_2, \leq y_1) \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, r_2, r_3, \ldots\ldots, r_n, & r_{n+1}, & r_{n+2}, r_{n+3}, r_{n+4}, \ldots\ldots, r_{2n+1}) \end{bmatrix}$$

Such that $\delta \notin X\ \&\ Y$; $(r_1, r_2, r_3, \ldots\ldots, r_{2n+1}) \in R$; $X, Y\ \&\ R \in \mathbb{R}$; $R_1 \neq 0$; $n \in \mathbb{N}$; and $X\ \&\ Y$ are isoreflective pairs. $\delta = 0$ is by default operation, except under optinalytic normalization.

By Kabirian-based optinalysis [3], the Kabirian coefficient of similarity $(KC_{Ssim.})$ and its derivatives satisfied the Y-rule of Kabirian-based isomorphic optinalysis.

$$\begin{matrix} KC1_{Ssim.}(X,Y) \\ \ldots \\ KC2_{Ssim.}(Y,X) \end{matrix} \rightleftharpoons P_{Ssim.}(X,Y) = P_{Ssim.}(Y,X) \rightleftharpoons P_{Sdsim.}(X,Y) = P_{Sdsim.}(Y,X)$$

where $X, Y \in \mathbb{R}$.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

The two possible Kabirian bi-coefficients ($KC1_{Ssim.}$ & $KC2_{Ssim.}$) function on two different, but inverse optinalytic scales.

2.5.3. Scale and scaloc-invariant statistical pairwise comparison

Statistical pairwise comparison is called scale invariance if the efficient location parameter is not removed from the variable, while it is called scaloc-invariant if the efficient location parameter is removed from the variable.

2.5.4. General properties of statistical pairwise comparison

i. It is based on the entire observations of variables, unlike percentile-based or decile-based statistics. Therefore, extreme maximum and minimum values are not discarded or trimmed.

ii. It applies to variable(s) from the set of real numbers (such as discrete or continuous variables containing either or both negative and positive values),

iii. Statistical pairwise comparison is a scale-invariant (i.e., robust to scale, unitless, and dimensionless) estimator.

Supposed we have an a scaling of variables $x = (x_1, x_2, x_3, \ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots, y_n)$.

$$KC_{Ssim.}(x, y) = KC_{Ssim.}(ax, ay) = KC_{Ssim.}(-ax, -ay)$$
$$P_{Ssim.}(x, y) = P_{Ssim.}(ax, ay) = P_{Ssim.}(-ax, -ay)$$
$$P_{Sdsim.}(x, y) = P_{Sdsim.}(ax, ay) = P_{Sdsim.}(-ax, -ay)$$

where $x, y, a \in \mathbb{R}; a \neq 0$.

Because the variables for statistical pairwise comparison can be descaled, two variables with different scales can be compared.

$$KC_{Ssim.}(x, y) = KC_{Ssim.}(x, ay) = KC_{Ssim.}(-x, -ay)$$
$$KC_{Ssim.}(x, y) = KC_{Ssim.}(ax, y) = KC_{Ssim.}(-ax, -y)$$

$$P_{Ssim.}(x, y) = P_{Ssim.}(x, ay) = P_{Ssim.}(-x, -ay)$$
$$P_{Ssim.}(x, y) = P_{Ssim.}(ax, y) = P_{Ssim.}(-ax, -y)$$
$$P_{Sdsim.}(x, y) = P_{Sdsim.}(x, ay) = P_{Sdsim.}(-x, -ay)$$
$$P_{Sdsim.}(x, y) = P_{Sdsim.}(ax, y) = P_{Sdsim.}(-ax, -y)$$

where $x, y, a \in \mathbb{R}; a \neq 0$.

iv. Statistical pairwise comparison is a location-invariant estimator, only if, the efficient location parameter is removed from the variable. For instance, taking the absolute distances from the mean.

Supposed we have a b location shift of a variable $x = (x_1, x_2, x_3, \ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots, y_n)$, then both were shifted by a b location and returned as $\hat{x} = (x_1 + b, x_2 + b, x_3 + b, \ldots, x_n + b)$, and $\hat{y} = (y_1 + b, y_2 + b, y_3 + b, \ldots, y_n + b)$. The location-invariance implies:

$$KC_{Ssim.}(x, y) = KC_{Ssim.}(\hat{x} - m, \hat{y} - n) = KC_{Ssim.}(\hat{x}, \hat{y})$$
$$P_{Ssim.}(x, y) = P_{Ssim.}(\hat{x} - m, \hat{y} - n) = P_{Ssim.}(\hat{x}, \hat{y})$$
$$P_{Sdsim.}(x, y) = P_{Sdsim.}(\hat{x} - m, \hat{y} - n) = P_{Sdsim.}(\hat{x}, \hat{y})$$

where $x, y, \hat{x}, \hat{y}, b \in \mathbb{R}$; m and n are the mean estimates of $\hat{x}$ and $\hat{y}$ respectively.

v. Where location and scale properties are combined (i.e., scaloc-transform distribution), statistical pairwise is its invariant scaloc-invariant and/or scale-invariant estimator.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

Supposed we have an a scaling and b location shift of variables $x = (x_1, x_2, x_3, \ldots\ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots\ldots, y_n)$.

Here is a situation (of scaloc-transform distribution) where only the scaloc-invariant statistical pairwise comparison is an efficient estimator.

$$KC_{Ssim.}(x,y) = KC_{Ssim.}\{(xa+b)-m, (ya+b)-n\} = KC_{Ssim.}\{(xa-b)-m, (ya-b)-n\}$$
$$P_{Ssim.}(x,y) = P_{Ssim.}\{(xa+b)-m, (ya+b)-n\} = P_{Ssim.}\{(xa-b)-m, (ya-b)-n\}$$
$$P_{Sdsim.}(x,y) = P_{Sdsim.}\{(xa+b)-m, (ya+b)-n\} = P_{Sdsim.}\{(xa-b)-m, (ya-b)-n\}$$

where $x, y, a, b \in \mathbb{R}$; $a \neq 0$; m and n are the mean estimates of $(xa \pm b)$ and $(ya \pm b)$ respectively; and the resultant effect is a location property.

Here is a situation (of scaloc-transform distribution) where both the scaloc-invariant and scale-invariant statistical pairwise comparison are efficient estimators.

$$KC_{Ssim.}(x,y) = KC_{Ssim.}\{a(x+b)-m, a(y+b)-n\} = KC_{Ssim.}\{-a(x+b)-m, -a(y+b)-n\}$$
$$P_{Ssim.}(x,y) = P_{Ssim.}\{a(x+b)-m, a(y+b)-n\} = P_{Ssim.}\{-a(x+b)-m, -a(y+b)-n\}$$
$$P_{Sdsim.}(x,y) = P_{Sdsim.}\{a(x+b)-m, a(y+b)-n\} = P_{Sdsim.}\{-a(x+b)-m, -a(y+b)-n\}$$

where $x, y, a, b \in \mathbb{R}$; $a \neq 0$; m and n are the mean estimates of $\pm a(x+b)$ and $\pm a(y+b)$ respectively; and the resultant effect is a scale property.

vi. Statistical pairwise comparison is a variant of pericentral rotation (alternate reflection).

Supposed we have a statistically ordered variables $x = (x_1, x_2, x_3, \ldots\ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots\ldots, y_n)$.

$$KC_{Ssim.}(\vec{x})|\vec{y}) \neq KC_{Ssim.}(\vec{x}|\vec{y})$$
$$P_{Ssim.}(\vec{x})|\vec{y}) \neq P_{Ssim.}(\vec{x}|\vec{y})$$
$$P_{Sdsim.}(\vec{x})|\vec{y}) \neq P_{Sdsim.}(\vec{x}|\vec{y})$$

where $x, y, \in \mathbb{R}$.

vii. Statistical pairwise comparison is population-independent and variant to sample size or multiple repeats of a univariate dataset.

Supposed we have a c duplicate of variables $x = (x_1, x_2, x_3, \ldots\ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots\ldots, y_n)$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Ssim.}(x,y) \neq KC_{Ssim.}([x]*c, [y]*c)$$
$$P_{Ssim.}(x,y) \neq P_{Ssim.}([x]*c, [y]*c)$$
$$P_{Sdsim.}(x,y) \neq P_{Sdsim.}([x]*c, [y]*c)$$

where $x, y \in \mathbb{R}; c \in \mathbb{N}$.

viii. Statistical pairwise comparison is a variant to sample size or multiple repeats of multivariate datasets.

Supposed we have a c duplicate of variables $A = [[x],[y],[z]]$ and $B = [[q],[r],[s]]$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Ssim.}([[x],[y],[z]],[[q],[r],[s]]) \neq KC_{Ssim.}([[x],[y],[z]]*c, [[q],[r],[s]]*c)$$
$$P_{Ssim.}([[x],[y],[z]],[[q],[r],[s]]) \neq P_{Ssim.}([[x],[y],[z]]*c, [[q],[r],[s]]*c)$$
$$P_{Sdsim.}([[x],[y],[z]],[[q],[r],[s]]) \neq P_{Sdsim.}([[x],[y],[z]]*c, [[q],[r],[s]]*c)$$

where $x, y, z, q, r, s \in \mathbb{R}; c \in \mathbb{N}$.

2.5.5. Python code

Get the python code at:

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/pair-wise_similarity_estimators.ipynb

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

Input guide: **pairwise_similarity**(*[data_x, data_y, centering, ordering, pairing, print])*

Input options:

- ✓ ***for data:*** *list of numerical values from a set of real numbers.*
- ✓ ***for centering:*** *"allow", or "never".*
- ✓ ***for ordering:*** *"ascend", "descend", or "never".*
- ✓ ***for pairing:*** *"H_H", or "T_T".*
- ✓ ***for print:*** *"kc", "psim", "pdsim", "kcalt1", "kcalt2", or "kcalt".*

Example 1:

data_x = [2,-4,6.12,8,4,0.2,4,5,6,24,12,-2,0,-3,4,-1.05,13.33]

data_y = [12,-2,0,-3,4,-1.05,13.33,2,-4,6.12,8,4,0.2,4,5,6,24]

print("Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:T_T", "print:kc"]))*

print("Probability of similarity =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:T_T", "print:psim"]))*

print("Probability of dissimilarity =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:T_T", "print:"]))*

print("Alt1. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:T_T", "print:kcalt1"]))*

print("Alt2. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:T_T", "print:kcalt2"]))*

print("Alt. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:T_T", "print:kcalt"]))*

Example 2:

print("Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "ordering:descend", "pairing:H_H", "print:kc"]))*

print("Probability of similarity =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "ordering:descend", "pairing:H_H", "print:psim"]))*

print("Probability of dissimilarity =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "ordering:descend", "pairing:H_H", "print:pdsim"]))*

print("Alt1. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "ordering:descend", "pairing:H_H", "print:kcalt1"]))*

print("Alt2. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "ordering:descend", "pairing:H_H", "print:kcalt2"]))*

print("Alt. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "ordering:descend", "pairing:H_H", "print:kcalt"]))*

Example 3:

print("Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:H_H", "print:kc"]))*

print("Probability of similarity =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:H_H", "print:psim"]))*

print("Probability of dissimilarity =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:H_H", "print:pdsim"]))*

print("Alt1. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:H_H", "print:kcalt1"]))*

print("Alt2. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:H_H", "print:kcalt2"]))*

print("Alt. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:H_H", "print:kcalt"]))*

2.5.6. Drawback and limitations of statistical pairwise comparison

The following are some of the identified drawbacks and limitations of statistical pairwise comparison:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

i. The given random ordering of elements of the list of variables is not preserved, thus an efficient theoretical ordering (i.e., ascend or descend sorting) has to be adopted or used.
ii. Variables lengths must be the same, otherwise, a suitable method needs to be used to align them.
iii. A suitable and efficient pairing style or alternate reflection has to be chosen and adopted for repeatability and comparison of results.
iv. The two possible Kabirian bi-coefficients do not function on the same optinalytic scale. For comparison of results, estimates with the mixed Kabirian coefficients should either be translated forward or otherwise uniformed by backward alternate translation.

2.6. Proposal 6: Geometrical Pairwise Comparison

2.6.1. Definition

A geometrical pairwise comparison between two variables (i.e., two sequences), under Kabirian-based optinalysis, is their isoreflectivity in geometrical order. The geometrical pairwise comparison refers to the conceptual ordering, with or without centering and descaling the data and optinalysing the established isoreflective pair for the given variables.

2.6.2. Computational steps and algorithmic procedure

Suppose we have a set of variables $X = (x_1, x_3, x_2, \ldots\ldots, x_n)$ and $Y = (y_3, y_1, y_2, \ldots\ldots, y_n)$. Let the order of algorithmic transformations t_c, t_o and t_d as centering, ordering, and descaling of the data X respectively. The optinalysis-based geometrical pairwise comparison implies the following steps:

First step 1: Centering the data X and Y (i.e., location removal). This is optional, depending on the task.

$$t_c(X) = (x_1, x_3, x_2, \ldots\ldots, x_n)$$
$$t_c(Y) = (y_3, y_1, y_2, \ldots\ldots, y_n)$$

Second step 2: Establish a conceptual order of sequence for the $t_c(X)$ and $t_c(Y)$variables.

$$t_{c,o}(X) = (x_1, x_2, x_3, \ldots\ldots, x_n)$$
$$t_{c,o}(Y) = (y_1, y_2, y_3, \ldots\ldots, y_n)$$

Second step 2: Descaling the conceptually ordered sequence of $t_{c,o}(X)$ and $t_{c,o}(Y)$ variables.

$$t_{c,o,d}(X) = (x_1, x_2, x_3, \ldots\ldots, x_n)$$
$$t_{c,o,d}(Y) = (y_1, y_2, y_3, \ldots\ldots, y_n)$$

Third step 3: Choose an efficient pairing style (reflection) and establish the isoreflective pair between $t_{c,o,d}(X)$ and $t_{c,o,d}(Y)$ about δ. For instance,

Head-to-head pairing or reflection is given as:

$$\left(t_{c,o,d}(\vec{X})\middle|t_{c,o,d}(\overleftarrow{Y})\right) = (x_1, x_2, x_3, \ldots\ldots x_n, \delta, y_n \ldots\ldots, y_3, y_2, y_1)$$

Tail-to-tail pairing or reflection is given as:

$$\left(t_{c,o,d}(\vec{X})\middle|t_{c,o,d}(\overleftarrow{Y})\right) = (x_n \ldots\ldots, x_3, x_2, x_1, \delta, y_1, y_2, y_3, \ldots\ldots y_n)$$

Fourth step 4: Optinalyse (by Kabirian-based optinalysis [3]) the isoreflective pair of $\left(t_{c,o,d}(\vec{X})\middle|t_{c,o,d}(\overleftarrow{Y})\right)$ or $\left(t_{c,o,d}(\overleftarrow{X})\middle|t_{c,o,d}(\vec{Y})\right)$ about a mid-point δ.

$$f\colon t_{c,o,d}(X) \underset{\twoheadrightarrow}{\delta} t_{c,o,d}(Y) \twoheadrightarrow R$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

$$f:\begin{bmatrix} t_{c,o,d}(X)=(x_1,x_2,x_3,\ldots.x_n) & \overset{\delta}{\twoheadrightarrow} & t_{c,o,d}(Y)=(y_n.\ldots.,y_3,y_2,y_1) \\ \downarrow & \downarrow & \downarrow \\ R=(r_1,r_2,r_3,\ldots.,r_n, & r_{n+1}, & r_{n+2},r_{n+3},r_{n+4},\ldots.,r_{2n+1}) \end{bmatrix}$$

Such that $\delta \notin X \,\&\, Y$; $(r_1,r_2,r_3,\ldots.,r_{2n+1}) \in R$; $X,Y \,\&\, R \in \mathbb{R}$; $R_1 \neq 0$; $n \in \mathbb{N}$; and $X \,\&\, Y$ are isoreflective pairs. $\delta = 0$ is by default operation, except under optinalytic normalization.

By Kabirian-based optinalysis [3], the Kabirian coefficient of similarity ($KC_{Gsim.}$) and its derivatives satisfied the Y-rule of Kabirian-based isomorphic optinalysis.

$$\begin{matrix} KC1_{Gsim.}(X,Y) \\ \ldots \\ KC2_{Gsim.}(Y,X) \end{matrix} \rightleftharpoons P_{Gsim.}(X,Y)=P_{Gsim.}(Y,X) \rightleftharpoons P_{Gdsim.}(X,Y)=P_{Gdsim.}(Y,X)$$

where $X,Y \in \mathbb{R}$.

The two possible Kabirian bi-coefficients ($KC1_{Gsim.}$ & $KC2_{Gsim.}$) function on two different, but inverse optinalytic scales.

2.6.3. Scale and scaloc-invariant geometrical pairwise comparison

Geometrical pairwise comparison is called scale invariance if the efficient location parameter is not removed from the variable, while it is called scaloc-invariant if the efficient location parameter is removed from the variable.

2.6.4. General properties of geometrical pairwise comparison

i. It is based on the entire observations of variables, unlike percentile-based or decile-based statistics. Therefore, extreme maximum and minimum values are not discarded or trimmed.
ii. It applies to variable(s) from the set of real numbers (such as discrete or continuous variables containing either or both negative and positive values),
iii. Geometrical pairwise comparison is a scale-invariant (i.e., robust to scale, unitless, and dimensionless) estimator.

Supposed we have an a scaling of variables $x=(x_1,x_2,x_3,\ldots.,x_n)$ and $y=(y_1,y_2,y_3,\ldots.,y_n)$.

$$KC_{Gsim.}(x,y)=KC_{Gsim.}(ax,ay)=KC_{Gsim.}(-ax,-ay)$$
$$P_{Gsim.}(x,y)=P_{Gsim.}(ax,ay)=P_{Gsim.}(-ax,-ay)$$
$$P_{Gdsim.}(x,y)=P_{Gdsim.}(ax,ay)=P_{Gdsim.}(-ax,-ay)$$

where $x,y,a \in \mathbb{R}; a \neq 0$.

i. Geometrical pairwise comparison is a location-invariant estimator, only if, the efficient location parameter is removed from the variable. For instance, taking the absolute distances from the mean.

Supposed we have a b location shift of a variable $x=(x_1,x_2,x_3,\ldots.,x_n)$ and $y=(y_1,y_2,y_3,\ldots.,y_n)$, then both were shifted by b location and returned as $\hat{x}=(x_1+b,x_2+b,x_3+b,\ldots.,x_n+b)$, and $\hat{y}=(y_1+b,y_2+b,y_3+b,\ldots.,y_n+b)$. The location-invariance implies:

$$KC_{Gsim.}(x,y)=KC_{Gsim.}(\hat{x}-m,\hat{y}-n)=KC_{Gsim.}(\hat{x},\hat{y})$$
$$P_{Gsim.}(x,y)=P_{Gsim.}(\hat{x}-m,\hat{y}-n)=P_{Gsim.}(\hat{x},\hat{y})$$
$$P_{Gdsim.}(x,y)=P_{Gdsim.}(\hat{x}-m,\hat{y}-n)=P_{Gdsim.}(\hat{x},\hat{y})$$

where $x,y,\hat{x},\hat{y},b \in \mathbb{R}$; m and n are the mean estimates of $\hat{x}$ and $\hat{y}$ respectively.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

ii. Where location and scale properties are combined (i.e., scaloc-transform distribution), geometrical pairwise is its invariant scaloc-invariant and/or scale-invariant estimator.

Supposed we have an a scaling and b location shift of variables $x = (x_1, x_2, x_3, \ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots, y_n)$.

Here is a situation (of scaloc-transform distribution) where only the scaloc-invariant geometrical pairwise comparison is an efficient estimator.

$$KC_{Gsim.}(x, y) = KC_{Gsim.}\{(xa + b) - m, (ya + b) - n\} = KC_{Gsim.}\{(xa - b) - m, (ya - b) - n\}$$
$$P_{Gsim.}(x, y) = P_{Gsim.}\{(xa + b) - m, (ya + b) - n\} = P_{Gsim.}\{(xa - b) - m, (ya - b) - n\}$$
$$P_{Gdsim.}(x, y) = P_{Gdsim.}\{(xa + b) - m, (ya + b) - n\} = P_{Gdsim.}\{(xa - b) - m, (ya - b) - n\}$$

where $x, y, a, b \in \mathbb{R};\ a \neq 0;\ m$ and n are the mean estimates of $(xa \pm b)$ and $(ya \pm b)$ respectively; and the resultant effect is a location property.

Here is a situation (of scaloc-transform distribution) where both the scaloc-invariant and scale-invariant geometrical pairwise comparison are efficient estimators.

$$KC_{Gsim.}(x, y) = KC_{Gsim.}\{a(x + b) - m, a(y + b) - n\} = KC_{Gsim.}\{-a(x + b) - m, -a(y + b) - n\}$$
$$P_{Gsim.}(x, y) = P_{Gsim.}\{a(x + b) - m, a(y + b) - n\} = P_{Gsim.}\{-a(x + b) - m, -a(y + b) - n\}$$
$$P_{Gdsim.}(x, y) = P_{Gdsim.}\{a(x + b) - m, a(y + b) - n\} = P_{Gdsim.}\{-a(x + b) - m, -a(y + b) - n\}$$

where $x, y, a, b \in \mathbb{R};\ a \neq 0;\ m$ and n are the mean estimates of $\pm a(x + b)$ and $\pm a(y + b)$ respectively; and the resultant effect is a scale property.

iii. Geometrical pairwise comparison is variant to pericentral rotation (alternate reflection).

Supposed we have a geometrically ordered variables $x = (x_1, x_2, x_3, \ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots, y_n)$.

$$KC_{Gsim.}(\vec{x})|\vec{y}) \neq KC_{Gsim.}(\vec{x}|\vec{y})$$
$$P_{Gsim.}(\vec{x})|\vec{y}) \neq P_{Gsim.}(\vec{x}|\vec{y})$$
$$P_{Gdsim.}(\vec{x})|\vec{y}) \neq P_{Gdsim.}(\vec{x}|\vec{y})$$

where $x, y, \in \mathbb{R}$.

iv. Geometrical pairwise comparison is a variant to sample size or multiple repeats of a univariate dataset.

Supposed we have a c duplicate of variables $x = (x_1, x_2, x_3, \ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots, y_n)$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Gsim.}(x, y) \neq KC_{Gsim.}([x] * c, [y] * c)$$
$$P_{Gsim.}(x, y) \neq P_{Gsim.}([x] * c, [y] * c)$$
$$P_{Gdsim.}(x, y) \neq P_{Gdsim.}([x] * c, [y] * c)$$

where $x, y \in \mathbb{R}; c \in \mathbb{N}$.

v. Geometrical pairwise comparison is variant to sample size or multiple repeats of multivariate datasets.

Supposed we have a c duplicate of variables $A = [[x], [y], [z]]$ and $B = [[q], [r], [s]]$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{Gsim.}([[x], [y], [z]], [[q], [r], [s]]) \neq KC_{Gsim.}([[x], [y], [z]] * c, [[q], [r], [s]] * c)$$
$$P_{Gsim.}([[x], [y], [z]], [[q], [r], [s]]) \neq P_{Gsim.}([[x], [y], [z]] * c, [[q], [r], [s]] * c)$$
$$P_{Gdsim.}([[x], [y], [z]], [[q], [r], [s]]) \neq P_{Gdsim.}([[x], [y], [z]] * c, [[q], [r], [s]] * c)$$

where $x, y, z, q, r, s \in \mathbb{R}; c \in \mathbb{N}$.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

2.6.5. Python code

Get the python code at:

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/pairwise_similarity_estimators.ipynb

Input guide: **pairwise_similarity**(*[data_x, data_y, centering, descaling , ordering, pairing, print])*

Input options:

- ✓ ***for data:*** *list of numerical values from a set of real numbers.*
- ✓ ***for centering:*** *"allow", or "never".*
- ✓ ***for descaling:*** *"descaling:allow", or "descaling:never".*
- ✓ ***for ordering:*** *"ascend", "descend", or "never".*
- ✓ ***for pairing:*** *"H_H", or "T_T".*
- ✓ ***for print:*** *"kc", "psim", "pdsim", "kcalt1", "kcalt2", or "kcalt".*

Example 1:

data_x = [2,-4,6.12,8,4,0.2,4,5,6,24,12,-2,0,-3,4,-1.05,13.33]

data_y = [12,-2,0,-3,4,-1.05,13.33,2,-4,6.12,8,4,0.2,4,5,6,24]

print("Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:T_T", "print:kc"]))*

print("Probability of similarity =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:T_T", "print:psim"]))*

print("Probability of dissimilarity =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:T_T", "print:pdsim"]))*

print("Alt1. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:T_T", "print:kcalt1"]))*

print("Alt2. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:T_T", "print:kcalt2"]))*

print("Alt. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:T_T", "print:kcalt"]))*

Example 2:

print("Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "descaling:never", "ordering:never", "pairing:H_H", "print:kc"]))*

print("Probability of similarity =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "descaling:never", "ordering:never", "pairing:H_H", "print:psim"]))*

print("Probability of dissimilarity =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "descaling:never", "ordering:never", "pairing:H_H", "print:pdsim"]))*

print("Alt1. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "descaling:never", "ordering:never", "pairing:H_H", "print:kcalt1"]))*

print("Alt2. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "descaling:never", "ordering:never", "pairing:H_H", "print:kcalt2"]))*

print("Alt. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:allow", "descaling:never", "ordering:never", "pairing:H_H", "print:kcalt"]))*

Example 3:

print("Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:H_H", "print:kc"]))*

print("Probability of similarity =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:H_H", "print:psim"]))*

print("Probability of dissimilarity =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:H_H", "print:pdsim"]))*

print("Alt1. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:H_H", "print:kcalt1"]))*

print("Alt2. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:H_H", "print:kcalt2"]))*

print("Alt. Kabirian coefficient =", **pairwise_similarity**(*[data_x, data_y, "centering:never", "descaling:never", "ordering:never", "pairing:H_H", "print:kcalt"]))*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

2.6.6. Drawback and limitations of geometrical pairwise comparison

The following are some of the identified drawbacks and limitations of geometrical pairwise comparison:

i. The given random ordering (sequence) of elements of the list of the variable(s) should be preserved, otherwise, a conceptual ordering has to be established.
ii. Variables lengths must be the same, otherwise, a suitable method needs to be used to align them.
iii. A suitable and efficient pairing style or alternate reflection has to be chosen and adopted for repeatability and comparison of results.
iv. The two possible Kabirian bi-coefficients do not function on the same optinalytic scale. For comparison of results, estimates with the mixed Kabirian coefficients should either be translated forward or otherwise equalized by backward alternate translation.

2.7. Proposal 7: Identity Estimation

2.7.1. Definition

Identity simply refers to the measure of the degree of exactness between a pair of variables. The identity estimation between pair of isoreflective or autoreflective variable(s) under Kabirian-based optinalysis, is the isoreflectivity of its id-strand onto a unified optinalytic mirror (i.e., a mirror with a principal value of 1.00 through a defined length). The id-strand is a pairwise match (score by 1), mismatch (score by 0), and/or gab (score by -1) scoring between the items of isoreflective or autoreflective points of isoreflective or autoreflective pair of variable(s).

The optinalysis-based identity estimation is an extension of symmetry and similarity concepts and is a subject of many combined assumptions such as those that hold the establishment of an order for the variable(s), the isoreflective or autoreflective pair, the id-strand formation, and their optinalytic mirroring.

Identity and similarity or symmetry are related but the former proceeds the latter by the concept. All identical variables are completely similar or symmetrical, but not all completely similar or symmetrical variables are identical. Pericentral rotation invariance must be satisfied for identical variables. Therefore, it can be used to isolate completely identical variables, but that does not account for the degree of the incomplete identity.

2.7.2. Iso-identity (pairwise identity) estimation

Iso-identity (pairwise identity) measures the degree of exactness between a pair of isoreflective variables.

Computational steps and algorithmic procedure

Suppose we have a set of variables $X = (x_1, x_3, x_2, \ldots\ldots, x_n)$ and $Y = (y_3, y_1, y_2, \ldots\ldots, y_n)$. Let the order of algorithmic transformations t_c and t_o as centering and ordering of the data X respectively. The optinalysis-based identity comparison between X and Y implies the following steps:

First step 1: Centering the data X and Y (i.e., location removal). This is optional, depending on the task.

$$t_c(X) = (x_1, x_3, x_2, \ldots\ldots, x_n)$$
$$t_c(Y) = (y_3, y_1, y_2, \ldots\ldots, y_n)$$

Second step 2: Establish a theoretical or conceptual order for the $t_c(X)$ and $t_c(Y)$ variables.

$$t_{c,o}(X) = (x_1, x_2, x_3, \ldots\ldots, x_n)$$
$$t_{c,o}(Y) = (y_1, y_2, y_3, \ldots\ldots, y_n)$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

Third step 3: Choose an efficient pairing style (reflection) and establish the isoreflective pair between $t_{c,o}(X)$ and $t_{c,o}(Y)$ about a mid-point δ. For instance,

Head-to-head pairing or reflection is given as:

$$\left(t_{c,o}(\overleftarrow{X})\middle|t_{c,o}(\vec{Y})\right) = (x_1, x_2, x_3, \ldots . x_n, \delta, y_n \ldots . , y_3, y_2, y_1)$$

Tail-to-tail pairing or reflection is given as:

$$\left(t_{c,o}(\vec{X})\middle|t_{c,o}(\overleftarrow{Y})\right) = (x_n \ldots . , x_3, x_2, x_1, \delta, y_1, y_2, y_3, \ldots . y_n)$$

Fourth step 4: Generate the id-strand $S = (s_1, s_2, s_3, \ldots . , s_n)$ from the established isoreflective pair $\left(t_{c,o}(\overleftarrow{X})\middle|t_{c,o}(\vec{Y})\right)$ or $\left(t_{c,o}(\vec{X})\middle|t_{c,o}(\overleftarrow{Y})\right)$. Note that the id-strand is not an isoreflective with any variable. Here is the rule for an id-strand generation.

If $t_{c,o}(\overleftarrow{X}) \equiv t_{c,o}(\vec{Y})$; then score it as 1; and the $1 \in S$.

If $t_{c,o}(\overleftarrow{X}) \not\equiv t_{c,o}(\vec{Y})$; then score it as 0; and the $0 \in S$.

If $t_{c,o}(\overleftarrow{X}) \emptyset\, t_{c,o}(\vec{Y})$; then score it as -1; and the $-1 \in S$.

Fifth step 5: Design a unified optinalytic mirror. A unified optinalytic mirror P is a mirror with a principal value of 1.00 through a defined length).

$$P = (p_1, p_2, p_3, \ldots . , p_n)$$

Sixth step 6: Choose an efficient pairing style (reflection) and establish the isoreflective pair between the id-strand S and optinalytic mirror P about a mid-point δ. For instance,

Head-to-head pairing or reflection is given as:

$$\left(\overleftarrow{S}\middle|\vec{P}\right) = (s_1, s_2, s_3, \ldots . , s_n, \delta, p_n, \ldots . , p_3, p_2, p_1)$$

Tail-to-tail pairing or reflection is given as:

$$\left(\vec{S}\middle|\overleftarrow{P}\right) = (s_n, \ldots . , s_3, s_2, s_1, \delta, p_1, p_2, p_3, \ldots . , p_n)$$

Seventh step 7: Geometrically Optinalyse (by Kabirian-based optinalysis [3]) the isoreflective pair $\left(\overleftarrow{S}\middle|\vec{P}\right)$ or $\left(\vec{S}\middle|\overleftarrow{P}\right)$ about a mid-point δ. Note, the geometrical optinalysis here does not define the estimation nomenclature, but it relies on the assumption that established the order (theoretical order implies statistical, and conceptual order implies geometrical) for the variable(s).

$$f\colon S \overset{\delta}{\twoheadrightarrow} P \twoheadrightarrow R$$

$$f\colon \begin{bmatrix} S = (s_n, \ldots . , s_3, s_2, s_1) & \overset{\delta}{\twoheadrightarrow} & P = (p_1, p_2, p_3 \ldots p_n) \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, r_2, r_3, \ldots . , r_n, & r_{n+1}, & r_{n+2}, r_{n+3}, r_{n+4}, \ldots . , r_{2n+1}) \end{bmatrix}$$

Such that $\delta \notin S \,\&\, P$; $(r_1, r_2, r_3, \ldots . , r_{2n+1}) \in R$; $S, P \,\&\, R \in \mathbb{R}$; $R_1 \neq 0$; $n \in \mathbb{N}$; and $S \,\&\, P$ are isoreflective pairs. $\delta = 0$ is by default operation, except under optinalytic normalization.

By Kabirian-based optinalysis [3], the Kabirian coefficient of identity $(KC_{id.})$ and its derivatives satisfied the Y-rule of Kabirian-based isomorphic optinalysis.

$$\begin{matrix} KC1_{id.}(S,P) \\ \ldots \\ KC2_{id.}(P,S) \end{matrix} \rightleftharpoons P_{id.}(S,P) = P_{id.}(P,S) \rightleftharpoons P_{uid.}(S,P) = P_{uid.}(P,S)$$

where $S, P \in \mathbb{R}$.

The two possible Kabirian bi-coefficients $(KC1_{id.} \,\&\, KC2_{id.})$ function on two different, but inverse optinalytic scales.

2.7.3. Auto-identity (self-identity) estimation

Auto-identity (self-identity) measures the degree of exactness between a pair of autoreflective variables.

Computational steps and algorithmic procedure

Suppose we have a set of variables $X = (x_3, x_1, x_2, \ldots . , x_n)$. Let the order of algorithmic transformations t_c and t_o as centering and ordering of the data X respectively. The optinalysis-based identity comparison of the shape within X implies the following steps:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

First step 1: Centering the data (i.e., location removal). This is optional, depending on the task. By centering the variable, two distinct sets of positive and negative integers were obtained.

$$\left(t_c(\check{X})\middle|t_c(\check{X}')\right) = (x_3, x_1, x_2, \ldots\ldots, x_n)$$

Second step 2: Establish a theoretical or conceptual order and the autoreflective pair of symmetry (shape) for the X variable. The two distinct separations of the integers into a positive and negative form the basis for the establishment of the autoreflective pair. For the efficiency of the result, absolute estimates of the centered data are used.

$$\left(t_{c,o}(\check{X})\middle|t_{c,o}(\check{X}')\right) = (x_1, x_2, x_3, \ldots\ldots, x_{\frac{n-1}{2}}, \hat{x}_{\frac{n+1}{2}}, x'_{\frac{n-1}{2}}, \ldots\ldots, x'_3, x'_2, x'_1)$$

Third step 3: Generate the id-strand $S = (s_1, s_2, s_3, \ldots\ldots, s_n)$ from the established autoreflective pair $\left(t_{c,o}(\check{X})\middle|t_{c,o}(\check{X}')\right)$. Note that the id-strand is not an autoreflective with any variable. Here is the rule for an id-strand generation.

If $t_{c,o}(\check{X}) \equiv t_{c,o}(\check{X}')$; then score it as 1; and the $1 \in S$.

If $t_{c,o}(\check{X}) \not\equiv t_{c,o}(\check{X}')$; then score it as 0; and the $0 \in S$.

If $t_{c,o}(\check{X})\ \emptyset\ t_{c,o}(\check{X}')$; then score it as -1; and the $-1 \in S$.

Fourth step 4: Design a unified optinalytic mirror. A unified optinalytic mirror P is a mirror with a principal value of 1.00 through a defined length).

$$P = (p_1, p_2, p_3, \ldots\ldots, p_n)$$

Fifth step 5: Choose an efficient pairing style (reflection) and establish the isoreflective pair between the id-strand S and the optinalytic mirror P about a mid-point δ. For instance,

Head-to-head pairing or reflection is given as:

$$\left(\vec{S}\middle|\vec{P}\right) = (s_1, s_2, s_3, \ldots\ldots, s_n, \delta, p_n, \ldots\ldots, p_3, p_2, p_1)$$

Tail-to-tail pairing or reflection is given as:

$$\left(\vec{S}\middle|\overleftarrow{P}\right) = (s_n, \ldots\ldots, s_3, s_2, s_1, \delta, p_1, p_2, p_3, \ldots\ldots, p_n)$$

Sixth step 6: Geometrically Optinalyse (by Kabirian-based optinalysis [3]) the isoreflective pair $\left(\vec{S}\middle|\vec{P}\right)$ or $\left(\vec{S}\middle|\overleftarrow{P}\right)$ about mid-point δ. Note, the geometrical optinalysis here does not define the estimation nomenclature, but it relies on the assumption that established the order (theoretical order implies statistical, and conceptual order implies geometrical) for the variable(s).

$$f\colon S \overset{\delta}{\twoheadrightarrow} P \twoheadrightarrow R$$

$$f\colon \begin{bmatrix} S = (s_1, s_2, s_3, \ldots\ldots s_n) & \overset{\delta}{\twoheadrightarrow} & P = (p_n, \ldots\ldots, p_3, p_2, p_1) \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, r_2, r_3, \ldots\ldots, r_n, & r_{n+1}, & r_{n+2}, r_{n+3}, r_{n+4}, \ldots\ldots, r_{2n+1}) \end{bmatrix}$$

Such that $\delta \notin S\ \&\ P$; $(r_1, r_2, r_3, \ldots\ldots, r_{2n+1}) \in R$; $S, P\ \&\ R \in \mathbb{R}$; $R_1 \neq 0$; $n \in \mathbb{N}$; and $S\ \&\ P$ are autoreflective pairs. $\delta = 0$ is by default operation, except under optinalytic normalization.

By Kabirian-based optinalysis [3], the Kabirian coefficient of identity $(KC_{id.})$ and its derivatives satisfied the Y-rule of Kabirian-based automorphic optinalysis.

$$\begin{matrix} KC1_{id.}(S,P) \\ \ldots \\ KC2_{id.}(P,S) \end{matrix} \rightleftharpoons P_{id.}(S,P) = P_{id.}(P,S) \rightleftharpoons P_{uid.}(S,P) = P_{uid.}(P,S)$$

where $S, P \in \mathbb{R}$.

The two possible Kabirian bi-coefficients $(KC1_{id.}\ \&\ KC2_{id.})$ function on two different, but inverse optinalytic scales.

Optimally, the required sample size to translate the two possible Kabirian bi-efficient is not, in this particular case of identity measure, determined by the variable's sample size or length. A sample size $n = 1$ is generally used in the translation process because both

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

the id-strand and the id-mirror are on a binary numerical expression (i.e., 0 and 1), and this gives an efficient result.

2.7.4. General properties of identity estimation

These properties applied to both the iso-identity and auto-identity. For consistency in the explanation, iso-identity is always preferred here.

i. It is based on the entire observations of variables, unlike percentile-based or decile-based statistics. Therefore, extreme maximum and minimum values are not discarded or trimmed.

ii. It applies to variable(s) from the set of real numbers (such as discrete or continuous variables containing either or both negative and positive values).

iii. The identity estimation is a scale-invariant (i.e., robust to scale, unitless, and dimensionless) estimator.

Supposed we have an a scaling of variables $x = (x_1, x_2, x_3, \ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots, y_n)$.

$$KC_{id.}(x, y) = KC_{id.}(ax, ay) = KC_{id.}(-ax, -ay)$$
$$P_{id.}(x, y) = P_{id.}(ax, ay) = P_{id.}(-ax, -ay)$$
$$P_{uid.}(x, y) = P_{uid.}(ax, ay) = P_{uid.}(-ax, -ay)$$

where $x, y, a \in \mathbb{R}; a \neq 0$.

iv. The identity estimation is a location-invariant estimator, only if, the efficient location parameter is removed from the variable. For instance, taking the absolute distances from the mean.

Supposed we have a b location shift of a variable $x = (x_1, x_2, x_3, \ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots, y_n)$, then both were shifted by b location and returned as $\hat{x} = (x_1 + b, x_2 + b, x_3 + b, \ldots, x_n + b)$, and $\hat{y} = (y_1 + b, y_2 + b, y_3 + b, \ldots, y_n + b)$. The location-invariance implies:

$$KC_{id.}(x, y) = KC_{id.}(\hat{x} - m, \hat{y} - n) = KC_{id.}(\hat{x}, \hat{y})$$
$$P_{id.}(x, y) = P_{id.}(\hat{x} - m, \hat{y} - n) = P_{id.}(\hat{x}, \hat{y})$$
$$P_{uid.}(x, y) = P_{uid.}(\hat{x} - m, \hat{y} - n) = P_{uid.}(\hat{x}, \hat{y})$$

where $x, y, \hat{x}, \hat{y}, b \in \mathbb{R}$; m and n are the mean estimates of $\hat{x}$ and $\hat{y}$ respectively.

v. Where location and scale properties are combined (i.e., scaloc-transform distribution), the identity estimation is its invariant scaloc-invariant and/or scale-invariant estimator.

Supposed we have an a scaling and b location shift of variables $x = (x_1, x_2, x_3, \ldots, x_n)$ and $y = (y_1, y_2, y_3, \ldots, y_n)$.

Here is a situation (of scaloc-transform distribution) where only the scaloc-invariant identity estimator is efficient.

$$KC_{id.}(x, y) = KC_{id.}\{(xa + b) - m, (ya + b) - n\} = KC_{id.}\{(xa - b) - m, (ya - b) - n\}$$
$$P_{id.}(x, y) = P_{id.}\{(xa + b) - m, (ya + b) - n\} = P_{id.}\{(xa - b) - m, (ya - b) - n\}$$
$$P_{uid.}(x, y) = P_{uid.}\{(xa + b) - m, (ya + b) - n\} = P_{uid.}\{(xa - b) - m, (ya - b) - n\}$$

where $x, y, a, b \in \mathbb{R}$; $a \neq 0$; m and n are the mean estimates of $(xa \pm b)$ and $(ya \pm b)$ respectively; and the resultant effect is a location property.

Here is a situation (of scaloc-transform distribution) where both the scaloc-invariant and scale-invariant identity estimators are efficient.

$$KC_{id.}(x, y) = KC_{id.}\{a(x + b) - m, a(y + b) - n\} = KC_{id.}\{-a(x + b) - m, -a(y + b) - n\}$$
$$P_{id.}(x, y) = P_{id.}\{a(x + b) - m, a(y + b) - n\} = P_{id.}\{-a(x + b) - m, -a(y + b) - n\}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

$$P_{uid.}(x,y) = P_{uid.}\{a(x+b)-m, a(y+b)-n\} = P_{uid.}\{-a(x+b)-m, -a(y+b)-n\}$$

where $x, y, a, b \in \mathbb{R}$; $a \neq 0$; m and n are the mean estimates of $\pm a(x+b)$ and $\pm a(y+b)$ respectively; and the resultant effect is a scale property.

vi. The identity estimation is invariant to pericentral rotation (alternate reflection).

Supposed we have suitably ordered variables $x = (x_1, x_2, x_3, \ldots., x_n)$ and $y = (y_1, y_2, y_3, \ldots., y_n)$.

$$KC_{id.}(\vec{x})|\vec{y}) = KC_{id.}(\vec{x}|\vec{y})$$
$$P_{id.}(\vec{x})|\vec{y}) = P_{id.}(\vec{x}|\vec{y})$$
$$P_{uid.}(\vec{x})|\vec{y}) = P_{uid.}(\vec{x}|\vec{y})$$

where $x, y, \in \mathbb{R}$.

vii. The identity estimation is population-independent and variant to sample size or multiple repeats of a univariate dataset.

Supposed we have a c duplicate of variables $x = (x_1, x_2, x_3, \ldots., x_n)$ and $y = (y_1, y_2, y_3, \ldots., y_n)$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{id.}(x,y) \neq KC_{id.}([x]*c, [y]*c)$$
$$P_{id.}(x,y) \neq P_{id.}([x]*c, [y]*c)$$
$$P_{uid.}(x,y) \neq P_{uid.}([x]*c, [y]*c)$$

where $x, y \in \mathbb{R}; c \in \mathbb{N}$.

viii. The identity estimation is variant to sample size or multiple repeats of multivariate datasets.

Supposed we have a c duplicate of variables $A = [[x],[y],[z]]$ and $B = [[q],[r],[s]]$.

By Kabirian-based optinalysis [3], it shows that

$$KC_{id.}([[x],[y],[z]],[[q],[r],[s]]) \neq KC_{id.}([[x],[y],[z]]*c, [[q],[r],[s]]*c)$$
$$P_{id.}([[x],[y],[z]],[[q],[r],[s]]) \neq P_{id.}([[x],[y],[z]]*c, [[q],[r],[s]]*c)$$
$$P_{uid.}([[x],[y],[z]],[[q],[r],[s]]) \neq P_{uid.}([[x],[y],[z]]*c, [[q],[r],[s]]*c)$$

where $x, y, z, q, r, s \in \mathbb{R}; c \in \mathbb{N}$.

2.7.5. Python codes

Get the python codes at:

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/auto-identity_estimators.ipynb

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/iso-identity_estimators.ipynb

Input guide: **iso_identity***([data_x, data_y, centering, ordering, pairing, print])*

auto_identity(*[data, centering, ordering, pairing, print])*

Input options:

- ✓ ***for data:*** *list of numerical values from a set of real numbers.*
- ✓ ***for centering:*** *"allow", or "never".*
- ✓ ***for ordering:*** *"ascend", "descend", or "never".*
- ✓ ***for pairing:*** *"H_H", or "T_T".*
- ✓ ***for print:*** *"kc", "pid", "puid", "kcalt1", "kcalt2", or "kcalt".*

Example 1:

data_x = [2,-4,6.12,8,4,0.2,4,5,6,24,12,-2,0,-3,4,-1.05,13.33]

data_y = [12,-2,0,-3,4,-1.05,13.33,2,-4,6.12,8,4,0.2,4,5,6,24]

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

print("Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:T_T", "print:kc"]*))

print("Probability of identity =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:T_T", "print:pid"]*))

print("Probability of unidentity =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:T_T", "print:puid"]*))

print("Alt1. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:T_T", "print:kcalt1"]*))

print("Alt2. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:T_T", "print:kcalt2"]*))

print("Alt. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:ascend", "pairing:T_T", "print:kcalt"]*))

Example 2:

print("Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:H_H", "print:kc"]*))

print("Probability of identity =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:H_H", "print:pid"]*))

print("Probability of unidentity =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:H_H", "print:puid"]*))

print("Alt1. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:H_H", "print:kcalt1"]*))

print("Alt2. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:H_H", "print:kcalt2"]*))

print("Alt. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:descend", "pairing:H_H", "print:kcalt"]*))

Example 3:

print("Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:never", "pairing:H_H", "print:kc"]*))

print("Probability of identity =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:never", "pairing:H_H", "print:pid"]*))

print("Probability of unidentity =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:never", "pairing:H_H", "print:puid"]*))

print("Alt1. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:never", "pairing:H_H", "print:kcalt1"]*))

print("Alt2. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:never", "pairing:H_H", "print:kcalt2"]*))

print("Alt. Kabirian coefficient =", **iso_identity**(*[data_x, data_y, "centering:never", "ordering:never", "pairing:H_H", "print:kcalt"]*))

Example 4:

print("Kabirian coefficient =", **auto_identity**(*[sorted(data_x), "centering:allow", "ordering:never", "pairing:H_H", "print:kc"]*))

print("Probability of identity =", **auto_identity**(*[sorted(data_x), "centering:allow", "ordering:never", "pairing:H_H", "print:pid"]*))

print("Probability of unidentity =", **auto_identity**(*[sorted(data_x), "centering:allow", "ordering:never", "pairing:H_H", "print:puid"]*))

print("Alt1. Kabirian coefficient =", **auto_identity**(*[sorted(data_x), "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt1"]*))

print("Alt2. Kabirian coefficient =", **auto_identity**(*[sorted(data_x), "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt2"]*))

print("Alt. Kabirian coefficient =", **auto_identity**(*[sorted(data_x), "centering:allow", "ordering:never", "pairing:H_H", "print:kcalt"]*))

Example 5:

print("Kabirian coefficient =", **auto_identity**(*[data_y, "centering:never", "ordering:never", "pairing:T_T", "print:kc"]*))

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

print("Probability of identity =", **auto_identity**(*[data_y, "centering:never", "ordering:never", "pairing:T_T", "print:pid"])*)
print("Probability of unidentity =", **auto_identity**(*[data_y, "centering:never", "ordering:never", "pairing:T_T", "print:puid"])*)
print("Alt1. Kabirian coefficient =", **auto_identity**(*[data_y, "centering:never", "ordering:never", "pairing:T_T", "print:kcalt1"])*)
print("Alt2. Kabirian coefficient =", **auto_identity**(*[data_y, "centering:never", "ordering:never", "pairing:T_T", "print:kcalt2"])*)
print("Alt. Kabirian coefficient =", **auto_identity**(*[data_y, "centering:never", "ordering:never", "pairing:T_T", "print:kcalt"])*)

2.7.6. Drawbacks and limitations of identity estimation

The following are some of the identified drawbacks and limitations of identity estimation:

i. For geometrical auto-identity or iso-identity, the given random ordering (sequence) of elements of the list of the variable(s) should be preserved, otherwise, a conceptual ordering has to be established.
ii. For statistical auto-identity or iso-identity, the given random ordering of elements of the list of the variable(s) is not preserved, thus an efficient theoretical ordering (i.e., ascend or descend sorting) has to be adopted or used.
iii. For geometrical or statistical iso-identity, variable lengths must be the same, otherwise, a suitable method needs to be used to align them.
iv. For geometrical or statistical iso-identity, a suitable and efficient pairing style or alternate reflection has to be chosen and adopted for repeatability and comparison of results.
v. The two possible Kabirian bi-coefficients do not function on the same optinalytic scale. For comparison of results, estimates with the mixed Kabirian coefficients should either be translated forward or otherwise homogenized by backward alternate translation.

2.8. Proposal 8: Feature Transformation

2.8.1. Definition

Based on optinalysis, feature transformation of a given dataset is the isoreflectivity of every item of that dataset to a defined estimate (e.g., an estimate of location, scale, or other efficient parameters) of itself.

2.8.2. Assumption of feature transformation

Suppose we have a variable $X = (x_1, x_2, x_3, \ldots\ldots, x_n)$ and its efficiently defined parameter Q, then their optinalytic construction with an assigned optiscale $R = (r_1, r_2, r_3)$ is expressed as follows:

$$f\colon X \overset{\delta}{\twoheadrightarrow} Q \twoheadrightarrow R = (r_1, r_2, r_3)$$

$$f\colon \begin{bmatrix} X & \delta = 0 & Q \\ \downarrow & \twoheadrightarrow & \downarrow \\ R = (r_1, & r_2, & r_3) \end{bmatrix}$$

Or the optinalytic construction is inversely expressed as:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

$$f: \begin{bmatrix} Q & \delta = 0 & X \\ & \twoheadrightarrow & \\ \downarrow & \downarrow & \downarrow \\ R = (r_1, & r_2, & r_3) \end{bmatrix}$$

Such that $(x_1, x_2, x_3, \ldots\ldots, x_n) \in X$; Q can be an estimate of X; $(r_1, r_2, r_3) \in R$; X, Q & $R \in \mathbb{R}$; $R_1 \neq 0$; and X & Q are isoreflective pairs. $\delta = 0$ is by default operation, except under optinalytic normalization. Q could be the average (mean), median, mode, maximum, range, standard deviation, variance, or others, of variable X.

By Kabirian-based optinalysis [3], the Kabirian coefficient of proximity/similarity ($KC1_{trans.}/KC2_{trans.}$) and its derivatives satisfied the Y-rule of Kabirian-based isomorphic optinalysis.

$$\begin{matrix} KC1_{trans.}(X,Q) \\ \ldots \\ KC2_{trans.}(Q,X) \end{matrix} \rightleftharpoons PS_{trans.}(X,Q) = PS_{trans.}(Q,X) \rightleftharpoons PD_{trans.}(X,Q) = PD_{trans.}(Q,X)$$

where $X, Q \in \mathbb{R}$.

The two possible Kabirian bi-coefficients ($KC1_{trans.}$ & $KC2_{trans.}$) function on two different, but inverse optinalytic scales.

2.8.3. Different methods of optinalytic feature transformation

Let $A = (a_1, a_2, a_3, \ldots\ldots, a_n)$ be a given random variable and $X = (x_1, x_2, x_3, \ldots\ldots, x_n)$ is an estimate of the A variable.

If $KC_{trans.}(X,Q) = X \overset{\delta}{\twoheadrightarrow} Q \twoheadrightarrow R$ expresses the optinalytic function, then we have the following methods:

Scaloc-invariant transformer

Scaler method 1: $X = A - mean(A)$, and $Q = std(A)$.

Scaler method 2: $X = A - mean(A)$, and $Q = ScalocFree\_P_{Smnprox}(A)$

Scaler method 3: $X = A - mean(A)$, and $Q = ScaleFree\_P_{Smnprox}(A)$

Scaler method 4: $X = A - mean(A)$, and $Q = max(A) - \min(A)$

Scaler method 5: $X = A - mean(A)$, and $Q = max(abs(X))$

Scaler method 6: $X = A - mean(A)$, and $Q = mean(abs(X))$

Scale-invariance transformer

Scaler method 7: $X = \frac{A}{\max(A)}$, and $Q = max(X) - \min(X)$

Scaler method 8: $X = \frac{A}{\max(A)}$, and $Q = max(X)$

Scaler method 9: $X = A$ and $Q = max(A)$

2.8.4. Python code

Get the python code at:

https://github.com/Abdullahi-KB/Kabirian-based_optinalysis/blob/main/feature_transformation_estimators.ipynb

Input guide: **feature_transformation**(*[data, method, print, guide])*

Input options:

- ✓ ***for data:*** *list of numerical values from a set of real numbers.*
- ✓ ***for method:*** *"std", "min_max", "maxbyabsmndiff", "mnbyabsmndiff", "scalocSMM", "scaleSMM", or "max".*
- ✓ ***for print:*** *"kc", "psim", "pdsim", "kcalt1", "kcalt2", or "kcalt".*
- ✓ ***for guide:*** *"view", or "never".*

Examples:

data = [2,-4,6.12,8,4,0.2,4,5,6,24,12,-2,0,-3,4,-1.05,13.33]

Examples: 1 *print("Transformed_data =",* **feature_transformation** *([data, "method:std", "print:psim", "guide:never"]))*

Examples: 2 *print("Transformed_data =",* **feature_transformation** *([data, "method:min_max", "print:psim", "guide:never"]))*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 31 October 2022 doi:10.20944/preprints202210.0464.v1

Examples: 3 *print("Transformed_data =",* **feature_transformation** *([data, "method:maxbyabsmndiff", "print:psim", "guide:never"]))*
Examples: 4 *print("Transformed_data =",* **feature_transformation** *([data, "method:mnbyabsmndiff", "print:psim", "guide:never"]))*
Examples: 5 *print("Transformed_data =",* **feature_transformation** *([data, "method:scalocSMM", "print:psim", "guide:never"]))*
Examples: 6 *print("Transformed_data =",* **feature_transformation** *([data, "method:scaleSMM", "print:psim", "guide:never"]))*
Examples: 7 *print("Transformed_data =",* **feature_transformation** *([data, "method:max", "print:psim", "guide:never"]))*

3. Discussion

The estimates produced by Kabirian-based optinalysis [**3**] can be estimated as scale-invariant, location-invariant, and scale-and-location-invariant. These invariance properties of these proposed estimators are good pieces of evidence to prove goodness and robustness for symmetry/asymmetry, similarity/dissimilarity, and identity/unidentity estimations. The commonly used estimators of symmetry/asymmetry, similarity/dissimilarity, and identity/unidentity are either scale-invariant or location-invariant [**1**], but rarely have combined the scale-and-location-invariant property. These properties have solved a very important problem of data analytics that relates to dealing with the issues of scaling and location shift. Comparison of multiple sets of variables (datasets) can be compared irrespective of the effect of scaling or location shift. Invariances and robustness are very important and desirable properties supposed to be found with estimators of dispersion, symmetry, cluster analysis using pairwise comparison approaches, etc.; that are essentially and routinely used in data analytics [**1**], [**2**]. Some of the feature transformation alternatives of this proposal are not at zero mean and restricted boundary, which is unlike the commonly used methods of feature standardization and normalization.

The proposed statistical and geometrical symmetry is complementary and matching alternatives to skewness measure (e.g., Pearson's first and second coefficients of skewness, standardized third central moment, etc) and object symmetry analysis (e.g., Riemannian distance, centroid distances, etc) respectively. Symmetric pairs of structures are exact automorphisms of their pair points. The commonly used estimators of symmetry cannot be proven by the concept of automorphism but automorphism is always assumed. The proposed statistical and geometrical pairwise similarity and identity estimators are alternatives to similarity or distance measures (e.g., cosine similarity, Euclidean distance, etc), and the pairwise sequence similarity or identity estimation respectively. The proposed statistical and geometrical mirroring are alternatives to the repeated approach of pairwise matrix comparisons clustering and also the dispersion estimators around a centre (e.g., standard deviation and coefficient of variation). Similarly, alternatives to the methods of feature transformation were also provided in this proposal to eliminate the possibilities of zero mean and restricted boundary after feature standardization and normalization.

The main drawbacks and limitations of these proposed estimators include: the variables' lengths must be the same (for the case of pairwise comparison), and pairing style or alternate reflection has to be chosen and adopted. All these limitations cannot be considered problematic but can be carefully treated by optimized and standardized alignment approaches and as well as common adoption of a chosen pairing style.

4. Conclusion

Based on the paradigm of Kabirian-based optinalysis, some statistical and geometrical estimators of symmetry/asymmetry, similarity/dissimilarity/distance, and identity/unidentity were proposed. Other estimators for feature transformation are also proposed. These estimators are characterized as invariant (robust) to either or both the scale and location parameters.

Supplementary Materials: The supplementary files are python codes. All the python codes are available at: https://github.com/Abdullahi-KB/Kabirian-based_optinalysis

Funding: This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflicts of Interest: The author has no conflict of interest to declare.

References

[1] K. Takeuchi, "Some Theorems on Invariant Estimators of location. In: Contribution on Theory of Mathematical Statistics", Springer, Tokyo, 2020. https://doi.org/10.1007/978-4-431-55239-0_3

[2] V. Hogg Robert, W. McKean Joseph, & T. Craig Allen, "Introduction to Mathematical Statistics", Eighth Edition, Pearson Education, USA, 2019.

[3] Z. Xiao, J. Wu, "Analysis on Image Symmetry Detection Algorithms", Proceedings - Fourth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2007. 4. 745-750. https://doi.org/10.1109/FSKD.2007.173

[4] A.D. Costea, R. Varga, S. Nedevschi, "Fast boosting based detection using scale invariant multimodal multiresolution filtered features. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR", Honolulu, HI, USA, pp. 993–1002, 2017. https://doi.org/10.1109/CVPR.2017.112

[5] A. S. Shirkhorshidi, S. Aghabozorgi, & T. Wah, "A Comparison Study on Similarity and Dissimilarity Measures in Clustering Continuous Data", PLOS ONE, 10.e0144059, 2015. https://doi.org/10.1371/journal.pone.0144059

[6] X. Fu, S. Shao, "Symmetry and Asymmetry Features for Human Detection. In: Li, B., Yang, M., Yuan, H., Yan, Z. (eds) IoT as a Service. IoTaaS 2018. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering", Cham. Vol 271, 2019, Springer. https://doi.org/10.1007/978-3-030-14657-3_30

[7] K. Wada, "Outliers in official statistics", Japanese Journal of Statistics and Data Science, 3:669-691, 2020. https://doi.org/10.1007/s42081-020-00091-y

[8] K. Bindawa Abdullahi, "Kabirian-based Optinalysis: Its Paradigm, Theorems, and Properties", Preprints, 2020080072, 2022. https://doi.org/10.20944/preprints202008.0072.v3