



CORRUPTION RISK COMPOSITE INDICATOR (CI)

Structure of data dimensionality

Deliverable WP 3.4

Michela Gnaldi

Niccolò Salvini

Simone Del Sarto

Grant Agreement number: 101038790 — CO.R.E — ISFP-2020-AG-CORRUPT

The content of this document represents the views of the author only and is his/her sole responsibility.

The European Commission does not accept any responsibility for use that may be made of the information it contains.

INDEX OF CONTENTS

Summary.....	3
1. Walkthrough: how to read the ‘coresoi’ R Package	4
1.1 Premises.....	4
1.2 Package Documentation.....	5
1.3 coresoi.....	5
1.4 Get Started.....	6
1.5 Reference	7
1.6 Articles (documentation).....	9
1.7 Changelog	11
2. Assessment of the structure of data dimensionality of the elementary indicators of corruption risk in emergency in the coresoi R Package	12



SUMMARY

This document contains the CO.R.E. deliverable 3.4. It is composed of two main sections:

1. Walkthrough: how to read the *coresoi* R Package
2. Assessment of the structure of data dimensionality of the elementary indicators of corruption risk in emergency in the *coresoi* R Package

The Walkthrough is an introductory section intended at supporting the users in the process of reading the documentation for the *coresoi* package, available at <https://core-forge.github.io/coresoi/>.

A Package Documentation is typically organised into different sections that provide information about the R package, such as the purpose of the package, the functions and datasets it provides, and how to use the package effectively.

In the case of *coresoi*, the documentation is made of five sections: “*coresoi*”, “Getting Started”, “Compute corruption risk in emergencies”, “Documentation” and “Changelog”. In the “Walkthrough” section, the internal architecture of the five sections will be detailed to ease users’ package navigation and practical use. As the “Walkthrough” section provides preliminary support to the users, it is recalled in the first part of the deliverables 3.3, 3.4 and 3.5.

The subsequent section 2 devoted to the “Assessment of the structure of data dimensionality of the elementary indicators of corruption risk in emergency in the *coresoi* R Package” includes the specific content of the deliverable 3.4. Specifically, it includes details on the implementation, documentation and exemplification of the dimensionality assessment of the red flags.



1. WALKTHROUGH: HOW TO READ THE 'CORESOI' R PACKAGE

1.1 Premises

R packages are essential tools for data analysts and statisticians who work with the R programming language. These packages consist of a set of organised functions, data, and documentation that are designed to solve specific data analysis problems.

To use an R package effectively, it is essential to understand its documentation. The documentation contains all the relevant information for a package, including the functions it contains, how to use them, and any other important details.

The *coresoi* package is a valuable tool for quantifying risk in emergency scenarios (such as COVID-19 or earthquakes) by exploiting relevant red flags in open contracting. It provides support to any users interested in computing corruption risk in public procurement over emergencies through analytical codes, users' guides and practical examples. *coresoi* contains a toy dataset based on the Italian Anti-Corruption Authority (ANAC) open data – the BDNCP (*Banca Dati Nazionale dei Contratti Pubblici*) – which researchers can use to calculate our elementary indicators of corruption risk (i.e., red flags) under emergencies and the composite indicator (CI) of corruption risk, resulting from the aggregation of the red flags.

The main purpose of R packages like *coresoi* is to provide users with a set of functions, data, and documentation aimed at addressing specific analytical problems. The package's functions and datasets are meant to be directly accessed by users within their R code. The included documentation offers guidance on its usage. Additionally, R packages like *coresoi* are essential as they are guaranteed to be easily installable across different systems. With regular building, installation, and testing, users can rely on the package's stability and conformity to industry standards.

In this document, we will focus on the process of reading the documentation for the *coresoi* package. The documentation for this package is available at <https://core-forge.github.io/coresoi/>, and we will explore it in detail in the following sections to show how to use the package to compute the risk of corruption in public procurement data under different emergency scenarios.



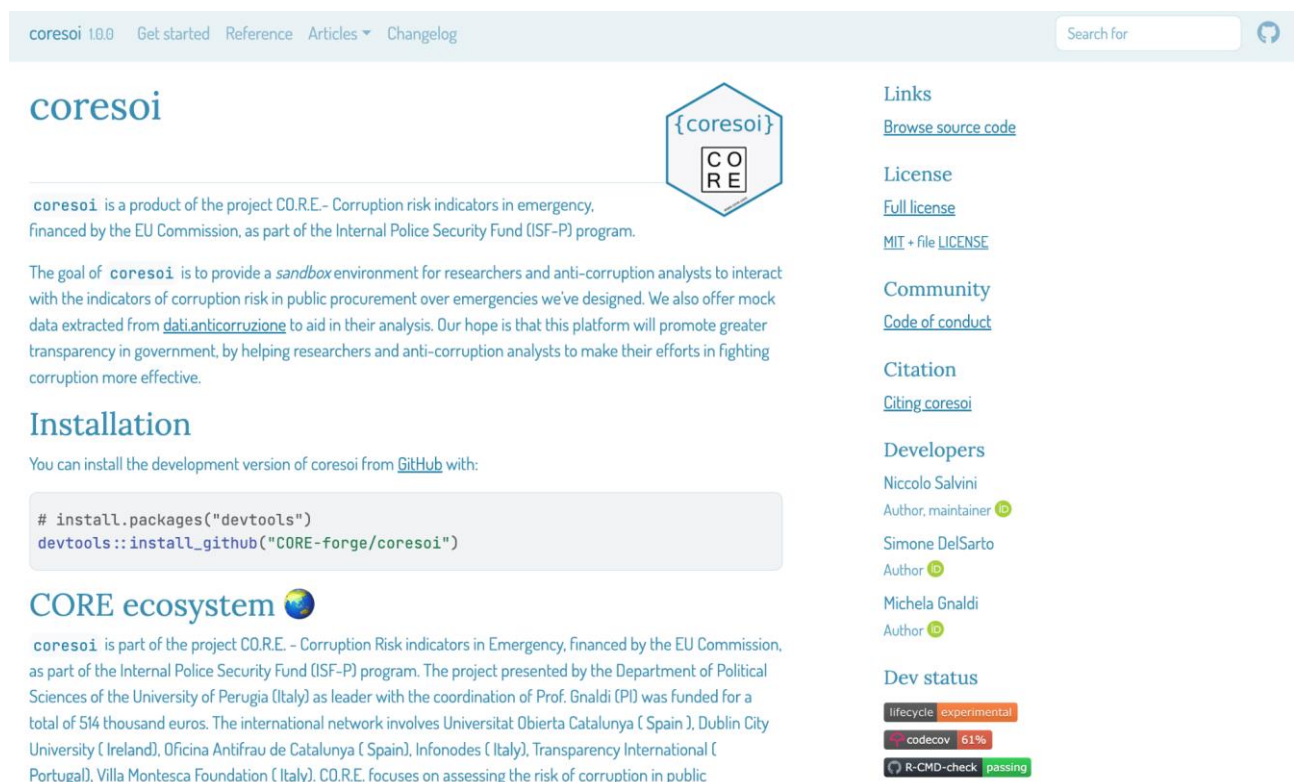
1.2 Package Documentation

The Package Documentation is typically organised into different sections that provide information about the R package, such as the purpose of the package, the functions and datasets it provides, and how to use the package effectively. Generally, the documentation has an introductory section that explains the motivation behind the package, the scope of the package, and provides an overview of its features. This section is usually easy to read and provides a general introduction to the package.

In the case of *coresoi*, the documentation is made of five sections: “coresoi”, “Get Started”, “Reference”, where the functions to compute red flags of corruption risk in emergencies are displayed, “Articles”, where users’ cases and further documentation are located, and “Changelog”.

The internal architecture of the five sections will be detailed in the following, to ease users’ package navigation and practical use.

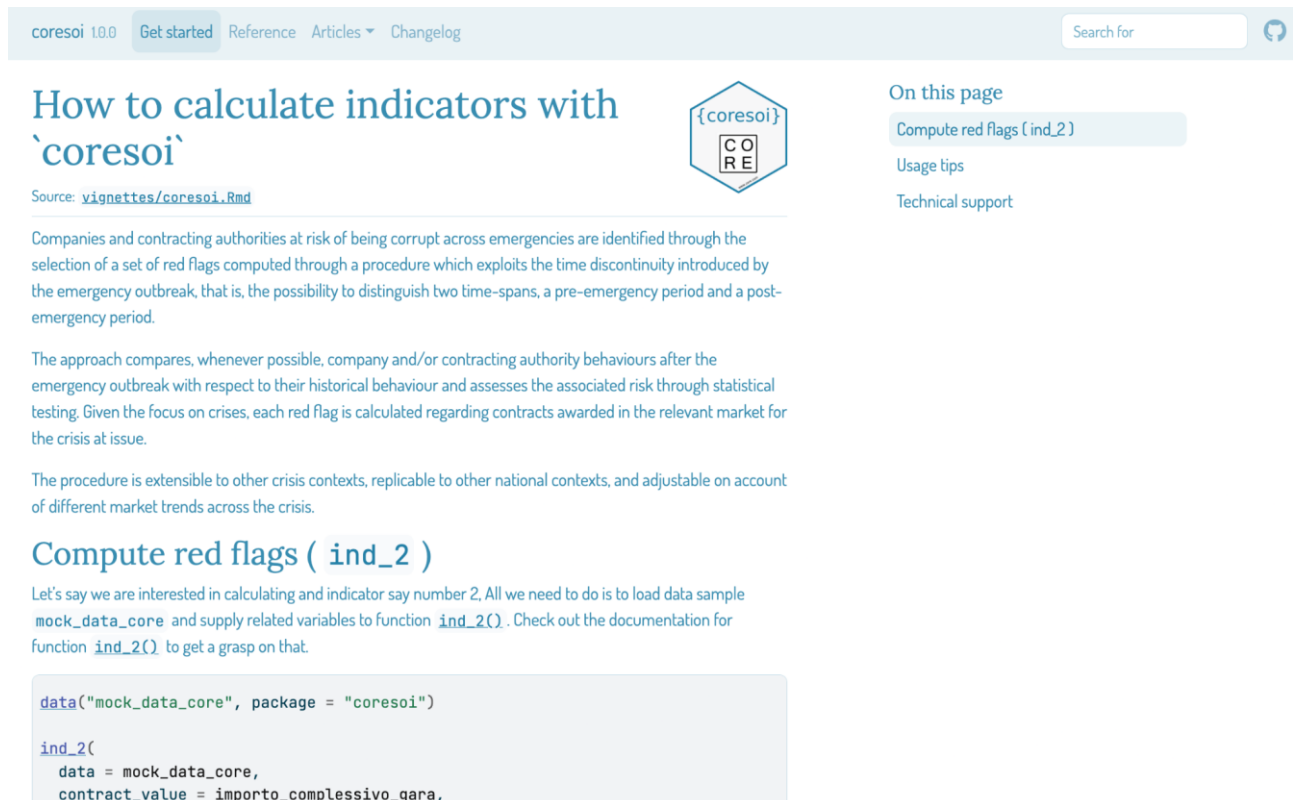
1.3 coresoi





The “coresoi” section is an introductory section providing synthetic information of both the *coresoi* package and the CO.R.E. project structure, rationale and objectives.

1.4 Get Started



The screenshot shows the documentation page for the 'coresoi' R package. The page title is 'How to calculate indicators with `coresoi`'. The source is cited as 'vignettes/coresoi.Rmd'. The main text describes how companies and contracting authorities at risk of being corrupt are identified through a selection of red flags. It explains that the approach compares company and/or contracting authority behaviours after an emergency outbreak with their historical behaviour. The procedure is extensible to other crisis contexts. A section titled 'Compute red flags (ind_2)' provides an example of how to load data and use the 'ind_2()' function. The code snippet is as follows:

```
data("mock_data_core", package = "coresoi")

ind_2(
  data = mock_data_core,
  contract_value = importo_compressivo_gara,
```

On the right side of the page, there is a 'On this page' section with links to 'Compute red flags (ind_2)', 'Usage tips', and 'Technical support'.

The “Get started” section of an R package documentation is an important part of the documentation. It provides new users with an introduction to the package and helps them to start interacting with the package quickly.

Here are some key elements of an effective “Getting started” section:

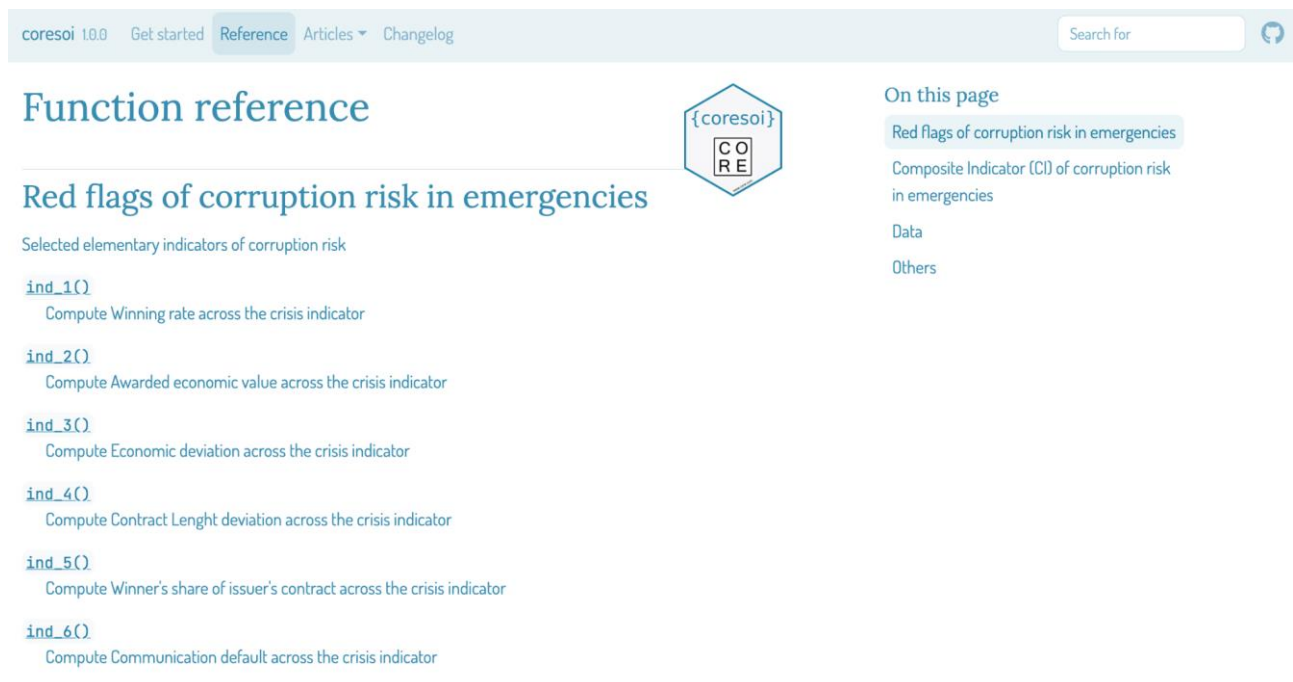
1. How to calculate indicators with ‘coresoi’. It provides a tutorial on how to use the package. The tutorial should be seen as an introductory guide that walks the user through the different types of analysis the package can perform.
2. Compute red flags. It includes example codes on how to use the functions (in this case, that for computing indicator 2).
3. Usage tips. It includes tips on how to install, load and use the package effectively. These tips can help new users to avoid common mistakes and make better use of the package.



4. Technical support. It provides information on how users can get help if they encounter problems with the package. This could include links to a help forum or support email address, or instructions on how to report bugs or issues with the package.

In our case, we display a brief dive into the calculation of an elementary indicator (`ind_2()` in this case) in the context of the COVID 19 emergency, whose analysis targets are Italian provinces. Data used for this showcase is still offered by the package under the name of `mock_data_core`. Results show the first 20 rows of the resulting dataset for `ind_2()`.

1.5 Reference



coresoi 1.0.0 Get started Reference Articles Changelog Search for

Function reference

Red flags of corruption risk in emergencies

Selected elementary indicators of corruption risk

- [ind_1\(\)](#)
Compute Winning rate across the crisis indicator
- [ind_2\(\)](#)
Compute Awarded economic value across the crisis indicator
- [ind_3\(\)](#)
Compute Economic deviation across the crisis indicator
- [ind_4\(\)](#)
Compute Contract Length deviation across the crisis indicator
- [ind_5\(\)](#)
Compute Winner's share of issuer's contract across the crisis indicator
- [ind_6\(\)](#)
Compute Communication default across the crisis indicator

On this page

- Red flags of corruption risk in emergencies
- Composite Indicator (CI) of corruption risk in emergencies
- Data
- Others

The “Reference” is the key section of the *coresoi* package, providing a list of all the functions provided by the package (including also sample data and utility functions), along with a description of each function, the arguments it requires and examples on how to use the functions with sample codes and data at hand.

The “Reference” section is structured into two main sub-sections: the “Red flags of corruption risk in emergencies” sub-section and the “Composite Indicator (CI) of corruption risk in emergencies” sub-section. The former is devoted to the computation, documentation and exemplification of the elementary indicators of corruption risk. The latter is devoted to the computation, documentation and exemplification of the synthetic indicator of corruption risk by company, contracting authority, region, province and municipality. Moreover, two further sub-sections are included: “Data” and “Other”. The former gathers information on



the sample data we used to showcase and validate (for internal testing purposes) indicator functions. The latter collects together all the internal functions that constitute the backbone of our software.

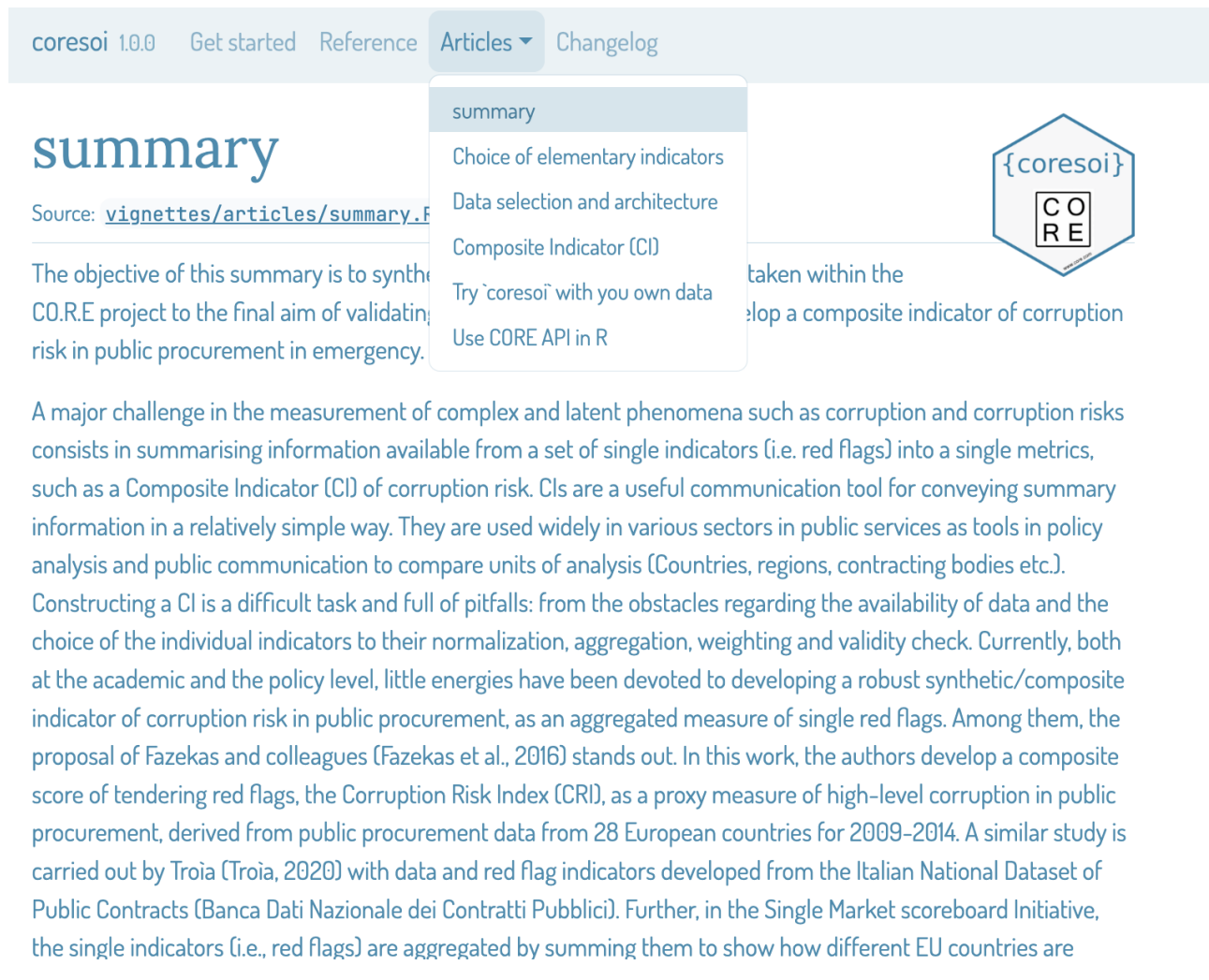
The section provides a technical reference and serves as the primary resource for using the package. For example, in sub-section “Red flags of corruption risk in emergencies”, the function 'ind_1()', computes the first elementary indicator of corruption risk in emergency (or red flag), labelled “Winning rate across the crisis”. Similarly, 'ind_2()' computes the second red flag, labelled “Awarded economic value across the crisis”.

Once a function has been selected, users can access the documentation (or guide). The users’ documentation typically provides information about the function, including its meaning and usage, the arguments it accepts, the default values for those arguments, along with additional information on how to use the function effectively and any special caveats or considerations that users should be aware of when using that function.

In paragraph 2 of the present document, the function in *coresoi* to implement the assessment of the dimensionality data structure of the elementary indicators of corruption risk in emergency (together with the users’ documentation or guide) will be described.



1.6 Articles (documentation)



coresoi 1.0.0 Get started Reference **Articles** Changelog

summary

Source: [vignettes/articles/summary.f](#)

The objective of this summary is to synthesize the CO.R.E project to the final aim of validating the use of composite indicators to measure corruption risk in public procurement in emergency.

Choice of elementary indicators
Data selection and architecture
Composite Indicator (CI)
Try `coresoi` with you own data
Use CORE API in R

{coresoi}

taken within the
develop a composite indicator of corruption

A major challenge in the measurement of complex and latent phenomena such as corruption and corruption risks consists in summarising information available from a set of single indicators (i.e. red flags) into a single metrics, such as a Composite Indicator (CI) of corruption risk. CIs are a useful communication tool for conveying summary information in a relatively simple way. They are used widely in various sectors in public services as tools in policy analysis and public communication to compare units of analysis (Countries, regions, contracting bodies etc.). Constructing a CI is a difficult task and full of pitfalls: from the obstacles regarding the availability of data and the choice of the individual indicators to their normalization, aggregation, weighting and validity check. Currently, both at the academic and the policy level, little energies have been devoted to developing a robust synthetic/composite indicator of corruption risk in public procurement, as an aggregated measure of single red flags. Among them, the proposal of Fazekas and colleagues (Fazekas et al., 2016) stands out. In this work, the authors develop a composite score of tendering red flags, the Corruption Risk Index (CRI), as a proxy measure of high-level corruption in public procurement, derived from public procurement data from 28 European countries for 2009–2014. A similar study is carried out by Troia (Troia, 2020) with data and red flag indicators developed from the Italian National Dataset of Public Contracts (Banca Dati Nazionale dei Contratti Pubblici). Further, in the Single Market scoreboard Initiative, the single indicators (i.e., red flags) are aggregated by summing them to show how different EU countries are

The “Article” section is where we provide additional information and resources related to the package. This section goes beyond the technical aspects of the package, exploring how the package can be used in various fields, the challenges it addresses, and other topics of interest to the community of users and developers.

The “Article” section usually contains blog posts, papers and other resources that are either authored by the package creators themselves or by the broader community using the package. Here are some examples of the type of content that might be included in the Documentation section:

1. Technical blog posts that provide in-depth technical detail on how to use the package, troubleshoot common issues, or that highlight specific features of the package.
2. Case studies that demonstrate how the package has been used in real-world scenarios.
3. Tutorials and educational materials, workshops, and presentations that teach users about the package or its underlying concepts.



4. Published articles, preprints, or manuscripts that describe research that either uses the package or was used to develop the package.
5. Community contributions from the broader community of developers or users (including blog posts, tutorials, videos or other materials), which help disseminate knowledge about the package and demonstrate how it is being used in various contexts.

In the *coresoi* package, the “Article” section contains an adapted documentation taken from the related CO.R.E documentation (i.e., project deliverables). Specifically, the section includes the following four sub-sections:

- Summary
- Choice of elementary indicators
- Data selection and architecture
- Composite Indicator (CI)
- Try `coresoi` with you own data
- Use CORE API in R

1.7 Changelog



The screenshot shows the Changelog page for the coresoi package. The page header includes navigation links for coresoi 1.0.0, Get started, Reference, Articles, and Changelog, along with a search bar and a GitHub icon. The main content area is titled "Changelog" and lists several versions with their respective changes:

- coresoi 1.0.0**
 - add *composite* main function
 - complete documentation for *composite*
- coresoi 0.2.0**
 - updates pkg documentation
 - release first version for composite indicator
- coresoi 0.1.1**
 - realesed all the feasible red flags: from `ind_1` to `ind_9`
- coresoi 0.1.0**

On the right side, there is a "On this page" section with a list of version numbers: 1.0.0, 0.2.0, 0.1.1, and 0.1.0. A small icon representing the package structure is also visible.

The “Changelog” provides a record of changes to the package between different versions, including new features added, bugs fixed, and any other updates made to the package. This helps users who are upgrading to a new version of the package understand what has changed and what new functionality is available. It also helps developers understand the evolution of the package and the motivations behind the changes made over time.

The section includes:

- the version number
- the date of release so that users can keep track of when updates were made
- changes and updates made to the package since the last version
- Impact and migration. For some changes, it may be helpful to include information on how they will impact users and how to migrate to the new functionality. Providing this information can help users better understand the changes and adapt to any modifications.
- Branch-specific changes. Since we managed the whole software development project on an open source online versioning tool (i.e. Github), we are able to trace back any changes applied to our codes. As a result, changes will only be relevant to a particular branch (please if you are unfamiliar with terminology refer to [gitflow](#) and this resource we explicitly develop to collaborate through [Github](#) and [RStudio](#)) of the package (e.g. only certain platforms or only certain packages). In these cases, it should be made clear in the changelog which changes apply to which branches.

2. ASSESSMENT OF THE STRUCTURE OF DATA DIMENSIONALITY OF THE ELEMENTARY INDICATORS OF CORRUPTION RISK IN EMERGENCY IN THE *CORESUIR* PACKAGE

coresui 1.0.0 [Get started](#) [Reference](#) [Articles](#) [Changelog](#) Search for

Dimensionality Check

Source: [R/dimensionality.R](#)

`dimensionality_check` performs a dimensionality assessment of a set of elementary indicators using either the Item Response Theory (IRT) framework or the Factor Analysis (FA).

Usage

```
dimensionality_check(
  indicator_list,
  dim_method = "IRT",
  cutoff = 0.95,
  missing = 0,
  max_ndim = length(indicator_list),
  nrep = 5,
  ...
)
```

Arguments

indicator_list
list of outputs about each indicator computable for the target unit (e.g., company or contracting authority), as returned by, for example, `ind_1()`, `ind_2()`, etc.

dim_method
method for the dimensionality assessment, to be chosen between "IRT" and "FA". If the former is selected, the dimensionality of elementary indicators is evaluated in the IRT framework using `mirt::mirt()` function. On the other hand, exploratory factor analysis is used by means of function `psych::fa()`. See Details.

On this page

- Usage
- Arguments**
- Value
- Details
- See also
- Examples

When building a CI of corruption risk, a set of red flags is transformed into a unique measure of the underlying phenomenon (corruption risk). In doing this, however, the phenomenon is considered as if it were unidimensional, as it is assumed that each single red flag reflects the same underlying construct. On the other hand, it might be reasonable to suppose the existence of several dimensions or groups of red flag indicators, which measure, from different perspectives, several risk types. For this reason, a dimensionality assessment is crucial in the process of construction of every CI, with the aim of uncovering whether the phenomenon at issue can be considered unidimensional or multidimensional. To this end, multivariate statistical techniques are available for acknowledging the dimensionality of a latent phenomenon, especially in terms of number and internal composition of the sub-dimensions to be accounted for.



In the following, it will be described the content of the *coresoi* package devoted to the implementation, documentation and exemplification of the dimensionality assessment of the elementary indicators of corruption risk in emergency. The “Composite indicator (CI) of corruption risk in emergencies” is the relevant sub-section under “Function reference” of the *coresoi* package for this task.

The function that performs the dimensionality assessment of the composite indicator is ‘dimensionality_check()’. It performs a complete dimensionality assessment of the elementary indicators proposed in the project, in order to evaluate whether the pool of these red flags can be considered as a unique set of indicators that measure a single construct, or, conversely, different subgroups of indicators can be detected, each measuring a different sub-dimension of the general phenomenon (corruption risk in emergency in our case) and identifying a specific type of risk.

For this function, the users’ documentation/guide includes:

- Usage. It provides an overview of how to use the function, including the arguments it requires.
- Arguments. It includes additional details on each of the function's arguments. In the case of the function ‘[dimensionality_check\(\)](#)’, it includes: i. ‘indicator_list’, the list of outputs about each indicator computable for the target unit (e.g., company or contracting authority), as returned by ‘ind_all()’; ii. ‘dim_method’, method for the dimensionality assessment, to be chosen between "IRT" and "FA"; iii. ‘cutoff’, threshold for dichotomising the indicators; iv. ‘missing’, method for imputing missing values; v. ‘max_ndim’, maximum number of dimensions to check in the IRT framework (not greater than the number of elementary indicators); vi. ‘nrep’, number of replicates for random initialisation of the algorithm for fitting IRT models; vii. optional arguments for ‘mirt::mirt()’ (e.g., estimation algorithm, convergence threshold, etc.) or ‘psych::fa()’ (e.g., method for factor extraction, rotation method, etc.).
- Value. It describes the output of the function. In this case, it returns different objects according to selected method of dimensionality check. If ‘dim_method’ = "IRT", it gets a list of IRT models (as returned by ‘mirt::mirt()’) for each possible dimensional solution, from one to ‘max_ndim’ dimensions; if ‘dim_method’ = "FA", it gives the best factorial solution (as returned by ‘psych::fa()’).
- Details. It clarifies how the dimensionality assessment of the elementary indicators is run. Firstly, it deals with dichotomised indicators (as those proposed in CO.R.E.), without missing values. Consequently, before carrying out the dimensionality assessment, the user has to provide the list of indicators (see argument ‘indicator_list’) together with two further arguments for their dichotomisation (‘cutoff’) and missing management (‘missing’).

Then, the dimensionality check is performed according to the chosen method (‘dim_method’).



If `'dim_method' = "IRT"`, the IRT framework is considered (by means of `'mirt::mirt()'` function). In this case, as first step, the function evaluates the model fitting of the Rasch model against the 2PL model (two-parameter logistic), two of the most widely used IRT models for binary data. It is evaluated under the unidimensional setting, in order to understand which type of model has a better fit on the data at hand (using common penalised likelihood metrics, such as AIC, SABIC, BIC, etc.).

As second step, multidimensional models are estimated by incrementing the number of dimensions each time, from two onwards (until `'max_ndim'`). For a given number of dimensions, say d , several estimates of the IRT model (i.e., Rasch or 2PL, according to step 1) are obtained on the data at hand according to the different initialisations of the estimation algorithm:

- a first initialisation is deterministic, based on observed `'data'`;
- the others (according to `'nrep'`) are random, in order to completely explore the likelihood function to maximise.

Finally, $1 + 'nrep'$ estimates of the IRT model with d dimensions are obtained and that with the largest value of maximised likelihood is saved in the list to be returned.

Step 2 is repeated starting from $d = 2$ until $d = 'max_ndim'$. As the function ends, a list of `'max_ndim'` IRT models is returned, one for each potential number of dimensions. Moreover, a summary of the dimensionality check is displayed, showing, for each d , the model fitting metrics of the best model with d dimensions. This summary helps in selecting the most suitable dimensional solution.

If `'dim_method' = "FA"`, exploratory factor analysis is considered (using `'psych::fa()'` function). In particular, in order to find the suitable number of factors to extract, this function computes the eigenvalues of the correlation matrix among the elementary indicators (computed using the tetrachoric correlation, given the binary nature of our indicators). Then, given the "eigenvalues > 1" rule, the suitable number of factors is retained and used in calling `'psych::fa()'`. The plot of the eigenvalues against the number of factors is also displayed.

- Examples. They provide some sample code that demonstrates how to practically use the function, which can be useful for users who are new to the package and are learning how to use it. In the case of the `'dimensionality_check()'` function, the example code firstly shows the computation of the set of elementary indicators by companies (using `'ind_all()'` wrapper function). This set is given in input to `'dimensionality_check()'`, specifying the cutoff value for dichotomising them (`'cutoff'`), together with the method for imputing missing values (`'missing'`). Moreover, other arguments are passed to the function in order to customise the analysis:

- the method for evaluating the dimensionality structure is IRT (`'dim_method' = "IRT"`);
- four is the maximum number of dimensions to check (`'max_ndim' = 4`);

- three is the number of replicates for the random initialisations of IRT fitting algorithm ('nrep' = 3);
- only for exemplificative purposes, the tolerance threshold of the IRT fitting algorithm is set to 0.1 to get the algorithm to converge faster ('TOL' = 0.1);
- 'verbose' = TRUE and 'method' = "QMCEM" are further specific arguments of 'mirt::mirt()' function.

Here is an example on what the function displays as a summary of the dimensionality assessment:

```
Step 1: Rasch vs. 2PL
      AIC      SABIC      HQ      BIC      logLik      X2 df p
rasch 85424.38 85467.13 85446.04 85492.56 -42704.19
2pl   74993.91 75068.72 75031.81 75113.22 -37482.96 10442.472 6 0

Step 2: multidimensional models
-----
Fitting 2PL model with 2 dimensions
-----
Fitting 2PL model with 3 dimensions
-----
Fitting 2PL model with 4 dimensions
-----
Summary of the dimensionality check
dim      AIC      SABIC      HQ      BIC      logLik
1  74993.91 75068.72 75031.81 75113.22 -37482.95
2  74371.52 74478.4 74425.66 74541.96 -37165.76
3  74318.72 74452.31 74386.39 74531.76 -37134.36
4  74194.27 74349.24 74272.77 74441.4 -37068.13
```