CORA and LOGIGRAM: A Duo of Python Packages for Combinational Regularity Analysis (CORA)

Zuzana Sebechlebská [1*], **Lusine Mkrtchyan** [1*], **and Alrik Thiem** [1*¶]

1 University of Lucerne, Switzerland ¶ Corresponding author * These authors contributed equally.

DOI: 10.21105/joss.05019

Software

- Review
- Repository
- Archive

Editor: Vissarion Fisikopoulos

Reviewers:

- @Naeemkh
- @Arrondissement5etDemi

Submitted: 05 December 2022
Published: 08 May 2023

Summary

Combinational Regularity Analysis (CORA) (Thiem et al., 2022) is a new member of the family of configurational comparative methods (CCMs). The most sophisticated CCMs infer INUS structures from configurational data (Thiem, 2022b). CORA generalizes the capabilities of existing CCMs. By means of Boolean optimization algorithms for multi-output data imported from electrical engineering, CORA can infer INUS structures that include complex conjunctive effects. The CORA package is an open-source Python tool, hosted on GitHub and deployed on the Google Colab platform, with which such structures can be identified. The visualization of INUS structures in CORA is achieved by the LOGIGRAM package, which provides functionality to CORA for producing logic diagrams.

Statement of need

Modern CCMs can infer regularity-theoretic causal structures from categorical data. The most sophisticated of these structures are so-called "INUS" structures. "INUS" refers to an **i**nsufficient but **n**on-redundant part of a condition that is **u**nnecessary but **s**ufficient for some outcome (Mackie, 1965). INUS structures are syntactically codified as two-level Boolean functions, such as $a \cdot b' + c' \Leftrightarrow z$, where a, b, c and z are Boolean literals, " $\cdot$ " represents the Boolean multiplication operator, " $+$ " the Boolean addition operator, " $'$ " the Boolean negation operator, and " $\Leftrightarrow$ " the Boolean implication operator. Branches of Boolean algebra include, most importantly, set theory, propositional logic and switching circuit theory (Lewin & Protheroe, 1992).

So far, the two most sophisticated CCMs have been Qualitative Comparative Analysis (QCA) (Ragin, 1987) and Coincidence Analysis (CNA) (Baumgartner, 2009; Whitaker et al., 2020). For QCA, several software packages exist, each with different functionality (Cronqvist, 2019; Dusa, 2022; Ragin & Davey, 2019; Thiem, 2018). However, since there is no harmonization of procedures in QCA, different software packages often generate very different results despite identical data input and parameter settings (Thiem & Duşa, 2013). For CNA, the cna package is available (Ambuehl & Baumgartner, 2022).

QCA and CNA can analyze single outputs only. Although CNA has been built for analyses with multiple outputs, its algorithm has to process each output one after another. Many scientific research problems, however, require the simultaneous analysis of multiple effects, such as analyses of multimorbidity (Suls & Green, 2019). CORA offers a solution. It generalizes QCA and CNA by allowing the simultaneous analyses of multiple outputs, even for multi-value data (Mkrtchyan et al., 2023). In this way, CORA can discover more complex INUS structures that include conjunctive effects. To this end, CORA implements enhanced adaptations of optimization algorithms that have originally been developed in electrical engineering for the design of multi-output switching circuits (Tison, 1967). In addition, CORA offers the possibility to mine configurational data via a tuple enlargement procedure for input factors. Last, but not least, with CORA, researchers can draw on logic diagrams to communicate their results. The Python package duo of `CORA` and `LOGIGRAM`, which have been developed to work in tandem, implement all aforementioned features.

Functionality and design

Figure 1 shows the internal structure of `CORA`. Two Python packages constitute its foundation. While the `CORA` package is responsible for all processes related to algorithmic optimization, the `LOGIGRAM` package provides graphical functionality for the production of logic diagrams. On the middle layer, these two Python packages are combined in CORA's Colab notebook, which delivers the visual interface to the user. This end-user interface represents the top level of `CORA` on which all interactions between the user and software take place. The input which the user has to provide consists of a set of `CORA`-compatible data and the specification of all required modelling parameters. Following the process of algorithmic optimization, a solution is generated. In `CORA`, such solutions take the form of a set of irredundant systems of two-level Boolean functions. These systems, or any stand-alone function in disjunctive normal form provided independently of `CORA`, can then also be rendered through `LOGIGRAM` as a logic diagram.

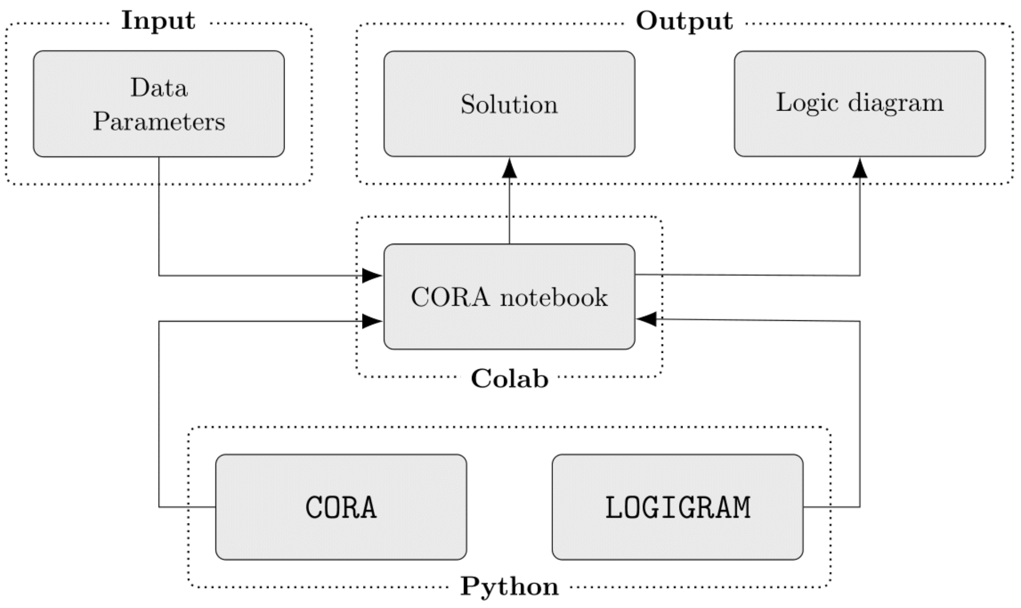
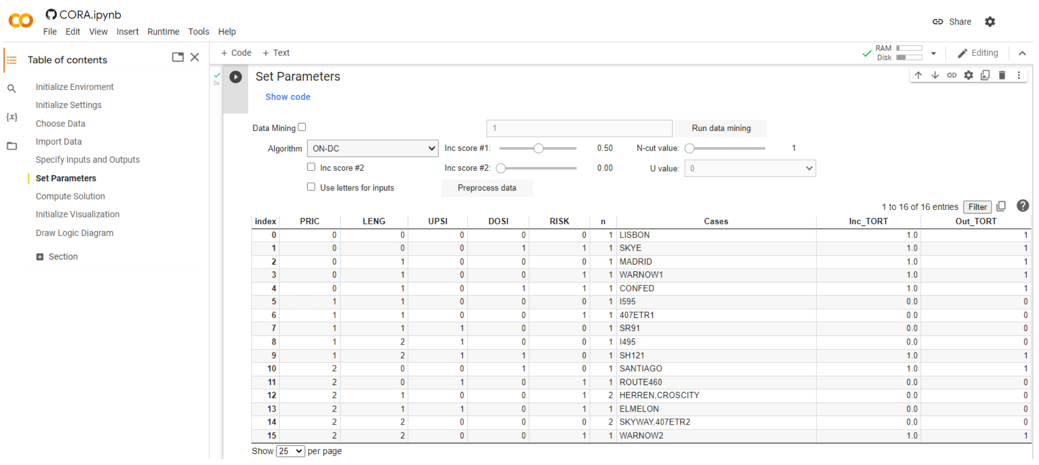


Figure 1: The structure of CORA.

The user interface of CORA's Colab notebook is shown in Figure 2. It guides users through the analysis in nine steps, the last two of which are optional: (1) the initialization of the framework and (2) default settings, the (3) choice and (4) import of data, the (5) specification of the inputs and outputs, the (6) setting of search parameters and thresholds for data fit statistics, (7) the computation of the solution, (8) the initialization of `CORA`'s visualization module and, finally, (9) the drawing and export of logic diagrams. In QCA and CNA, the existence of multiple models that fit some set of data equally well is called model ambiguity (Baumgartner & Thiem, 2017; Thiem et al., 2020). The analogue of model ambiguitiy in `CORA` is called "system ambiguity". A further commonality with CNA, but difference to QCA, is that `CORA` does not permit the manipulation of don't care terms. So-called "conservative" or "intermediate" solutions, which produce high rates of false positives (Baumgartner & Thiem, 2020; Thiem, 2022a), are not offered in `CORA`. In contrast to both QCA and CNA, `CORA` does not allow the use of fuzzy variables because of the numerous inferential problems associated with such variables. However, CORA comes with full support for multivalent inputs and multivalent outputs, which allow for more fine-grained analyses than bivalent crisp or fuzzy variables do (Thiem, 2014).



Set Parameters

Data Mining ☐ 1

Algorithm: ON-DC

Inc score #1: 0.50

N-cut value: 1

☐ Inc score #2

Inc score #2: 0.00

U value: 0

☐ Use letters for inputs

1 to 16 of 16 entries

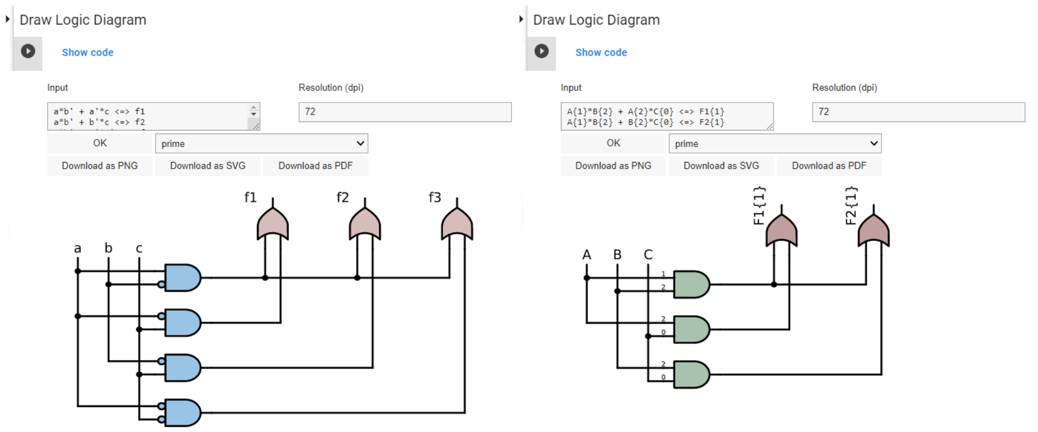
index	PRIC	LENG	UPSI	DOSI	RISK	n	Cases	Inc_TORT	Out_TORT
0	0	0	0	0	0	1	LISBON	1.0	1
1	0	0	0	1	1	1	SKYE	1.0	1
2	0	1	0	0	0	1	MADRID	1.0	1
3	0	1	0	0	1	1	WARNOW1	1.0	1
4	0	1	0	1	1	1	CONFED	1.0	1
5	1	1	0	0	0	1	I595	0.0	0
6	1	1	0	0	1	1	407ETR1	0.0	0
7	1	1	1	0	0	1	SR91	0.0	0
8	1	2	1	0	0	1	I495	0.0	0
9	1	2	1	1	0	1	SH121	1.0	1
10	2	0	0	1	0	1	SANTIAGO	1.0	1
11	2	0	1	0	1	1	ROUTE460	0.0	0
12	2	1	0	0	1	2	HERREN,CROSCITY	0.0	0
13	2	1	1	0	1	1	ELMELON	0.0	0
14	2	2	0	0	0	2	SKYWAY,407ETR2	0.0	0
15	2	2	0	0	1	1	WARNOW2	1.0	1

Show 25 per page

Figure 2: Interface of CORA's Colab notebook.

Graphics

Logic diagrams are graphical representations of Boolean-algebraic functions that have so far almost exclusively been used in electrical engineering (Thiem et al., 2023). Over the last 10 years, however, scientists from other disciplines have begun to discover the utility of logic diagrams (Lorenzo & Schmidt, 2018; Pearl, 2009; Thiem et al., 2020; Wang & Buck, 2012). The LOGIGRAM package in CORA is an interactive tool for the standardized production of two-level logic diagrams, called "logigrams" in CORA. Two examples of a logigram, including the interface through which they are produced, are shown in Figure 3 for a binary (left) and a multi-value system (right), respectively.



```
a*b' + a'*c <=> f1
a*b' + b'*c <=> f2
```

Resolution (dpi): 72

```
A{1}*B{2} + A{2}*C{0} <=> F1{1}
A{1}*B{2} + B{2}*C{0} <=> F2{1}
```

Resolution (dpi): 72

Figure 3: Logigrams for system of binary (left panel) and multi-valued functions (right panel).

Future work

Future work on CORA aims to implement several improvements and advancements. We list them in the order of priority:

- So far, CORA offers two exact optimization algorithms. At least one heuristic algorithm will be added. Heuristic algorithms are not guaranteed to find a global optimum, but they can handle larger data sets.
- The LOGIGRAM package will receive further functionality.
- Method evaluation tools will be added.

Requirements

CORA and LOGIGRAM are implemented in Python 3.7. For users who would like to directly run these in a Python environment, Python 3.7 or higher versions are required. For running CORA on Google Colab, a Google account is required.

Funding

The Swiss National Science Foundation has generously funded this research under grant award number PP00P1_202676 to AT. The funding body had no role in the design, writing or publication of this manuscript.

References

Ambuehl, M., & Baumgartner, M. (2022). *cna: Causal modeling with Coincidence Analysis, version 3.4.0*. http://cran.r-project.org/package=cna

Baumgartner, M. (2009). Inferring causal complexity. *Sociological Methods & Research*, *38*(1), 71–101. https://doi.org/10.1177/0049124109339369

Baumgartner, M., & Thiem, A. (2017). Model ambiguities in configurational comparative research. *Sociological Methods & Research*, *46*(4), 954–987. https://doi.org/10.1177/0049124115610351

Baumgartner, M., & Thiem, A. (2020). Often trusted but never (properly) tested: Evaluating Qualitative Comparative Analysis. *Sociological Methods & Research*, *49*(2), 279–311. https://doi.org/10.1177/0049124117701487

Cronqvist, L. (2019). *Tosmana: Tool for small-n analysis, version 1.61 [computer program]*. University of Trier. https://www.tosmana.net

Dusa, A. (2022). *QCA: Qualitative Comparative Analysis, R package version 3.17*. https://cran.r-project.org/package=QCA

Lewin, D., & Protheroe, D. (1992). *Design of logic systems* (2nd ed.). Chapman & Hall.

Lorenzo, V. de, & Schmidt, M. (2018). Biological standards for the knowledge-based bioeconomy: What is at stake. *New Biotechnology*, *40*, 170–180. https://doi.org/10.1016/j.nbt.2017.05.001

Mackie, J. L. (1965). Causes and conditions. *American Philosophical Quarterly*, *2*(4), 245–264.

Mkrtchyan, L., Thiem, A., & Sebechlebská, Z. (2023). Re-establishing a lost connection: Multi-value logic in causal data analysis in social science disciplines. *IEEE Access*, *11*, 10471–10482. https://doi.org/10.1109/ACCESS.2023.3240094

Pearl, J. (2009). *Causality: Models, reasoning, and inference* (2nd ed.). Cambridge University Press.

Ragin, C. C. (1987). *The comparative method: Moving beyond qualitative and quantitative strategies*. University of California Press.

Ragin, C. C., & Davey, S. (2019). *fs/QCA: fuzzy-set/Qualitative Comparative Analysis, version 3.1b [computer program]*. http://www.socsci.uci.edu/~cragin/fsQCA/software.shtml

Suls, J., & Green, P. A. (2019). Multimorbidity in health psychology and behavioral medicine. *Health Psychology*, *38*(9), 769–771. https://doi.org/10.1037/hea0000783

Thiem, A. (2014). Unifying configurational comparative methods: Generalized-set qualitative comparative analysis. *Sociological Methods & Research*, *43*(2), 313–337. https://doi.org/10.1177/0049124113500481

Thiem, A. (2018). *QCApro: Advanced functionality for performing and evaluating Qualitative Comparative Analysis, R package version 1.1-2*. http://cran.r-project.org/package=QCApro

Thiem, A. (2022a). Beyond the facts: Limited empirical diversity and causal inference in Qualitative Comparative Analysis. *Sociological Methods & Research*, *51*(2), 527–540. https://doi.org/10.1177/0049124119882463

Thiem, A. (2022b). Qualitative Comparative Analysis (QCA). In R. J. Huddleston, T. Jamieson, & P. James (Eds.), *Handbook of research methods in International Relations* (pp. 607–628). Edward Elgar. https://doi.org/10.4337/9781839101014.00044

Thiem, A., & Dușa, A. (2013). Boolean minimization in social science research: A review of current software for Qualitative Comparative Analysis (QCA). *Social Science Computer Review*, *31*(4), 505–521. https://doi.org/10.1177/0894439313478999

Thiem, A., Mkrtchyan, L., Haesebrouck, T., & Sanchez, D. (2020). Algorithmic bias in social research: A meta-analysis. *PLOS ONE*, *15*(6), e0233625. https://doi.org/10.1371/journal.pone.0233625

Thiem, A., Mkrtchyan, L., & Sebechlebská, Z. (2022). Combinational regularity analysis (CORA) — a new method for uncovering complex causation in medical and health research. *BMC Medical Research Methodology*, *22*(1), 333. https://doi.org/10.1186/s12874-022-01800-9

Thiem, A., Sebechlebská, Z., & Mkrtchyan, L. (2023). Logic diagrams: A visual tool with untapped potential. *Nature Reviews Methods Primers*, *3*(1), 21. https://doi.org/10.1038/s43586-023-00210-5

Tison, P. (1967). Generalization of consensus theory and application to the minimization of boolean functions. *IEEE Transactions on Electronic Computers*, *16*(4), 446–456. https://doi.org/10.1109/PGEC.1967.264648

Wang, B., & Buck, M. (2012). Customizing cell signaling using engineered genetic logic circuits. *Trends in Microbiology*, *20*(8), 376–384. https://doi.org/10.1016/j.tim.2012.05.001

Whitaker, R. G., Sperber, N., Baumgartner, M., Thiem, A., Cragun, D., Damschroder, L., Miech, E. J., Slade, A., & Birken, S. (2020). Coincidence analysis: A new method for causal inference in implementation science. *Implementation Science*, *15*(1), 108. https://doi.org/10.1186/s13012-020-01070-3