

# Metasyn: Transparent Generation of Synthetic Tabular Data with Privacy Guarantees

Raoul Schram<sup>1\*</sup>, Thom Volker<sup>1,3</sup>, Ayoub Bagheri<sup>1</sup>, Ron Scholten<sup>1</sup>, Samuel Spithorst<sup>1</sup>, and Erik-Jan van Kesteren<sup>1,2\*</sup>

<sup>1</sup> Utrecht University, The Netherlands <sup>2</sup> ODISSEI: Open Data Infrastructure for Social Science and Economic Innovations, The Netherlands <sup>3</sup> Statistics Netherlands ¶ Corresponding author \* These authors contributed equally.

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Editor: [Open Journals](#) ↗

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## Summary

Synthetic data is a promising tool for improving the accessibility of datasets that are otherwise too sensitive to be shared publicly. To this end, we introduce metasyn, a Python package for generating synthetic data from tabular datasets. Unlike existing synthetic data generation software, metasyn is built on a simple generative model with a “naive” marginal independence assumption — an explicit choice that lowers the multivariate precision of the synthetic data in order to maintain transparency and auditability, to keep information leakage to a minimum, and even to enable privacy or disclosure risk guarantees through a plug-in system. While the analytical validity of the generated data is thus intentionally limited, its potential uses are broad, including exploratory analyses, code development and testing, and external communication and teaching. Metasyn is flexible, scalable, and easily extended to meet diverse privacy needs.



Figure 1: Logo of the metasyn project.

## Statement of need

Metasyn is a python package for generating synthetic data with a focus on privacy and disclosure control. It is aimed at owners of sensitive datasets such as public organisations, research groups, and individual researchers who want to improve the accessibility of their data for research and reproducibility by others. The goal of metasyn is to make it easy for data owners to share the structure and and approximation of the content of their data with others without any privacy concerns.

With this goal in mind, metasyn distinguishes itself from existing software for generating synthetic data (e.g., [Nowok et al., 2016](#); [Ping et al., 2017](#); [Templ et al., 2017](#)) by restricting itself to the “augmented plausible” category of synthetic data ([Bates et al., 2019](#)). This choice enables the software to generate synthetic data with **privacy and disclosure guarantees** through a plug-in system. Moreover, our system provides an **auditable and editable intermediate**

31 **representation** in the form of a human- and machine-readable .json metadata file from which  
32 new data can be synthesized.

33 Through our focus on privacy and transparency, metasyn explicitly avoids generating synthetic  
34 data with high analytical validity. The data generated by our system is realistic in terms of  
35 data structure and plausible in terms of values for each variable, but any multivariate relations  
36 or conditional patterns are excluded. This has implications for how this synthetic data can be  
37 used: not for statistical analysis and inference, but rather for initial exploration, analysis script  
38 development, and communication outside the data owner's institution. In the intended use  
39 case, an external researcher can make use of the synthetic data to assess the feasibility of their  
40 intended research before making the (often time-consuming) step of requesting access to the  
41 sensitive source data for the final analysis.

42 As mentioned before, the privacy capacities of metasyn are extensible through a plug-in system,  
43 recognizing that different data owners have different needs and definitions of privacy. A  
44 data owner can define under which conditions they would accept open distribution of their  
45 synthetic data — be it based on differential privacy (Dwork, 2006), statistical disclosure control  
46 (Wolf, 2012), k-anonymity (Sweeney, 2002), or another specific definition of privacy. As  
47 part of the initial release of metasyn, we publish two proof-of-concept plugins: one following  
48 the disclosure control guidelines from Eurostat (Bond et al., 2015), and one based on the  
49 sample-and-aggregate technique for differential privacy (Dwork & Smith, 2010, p. 142).

## 50 Software features

51 At its core, metasyn is designed for three functions, which are briefly described in this section:

- 52 1. **Estimation:** Automatically select univariate distributions and fit them to a well-defined  
53 tabular dataset, possibly with privacy guarantees.
- 54 2. **(De)serialization:** Create an intermediate representation of the fitted model for auditing,  
55 editing, and exporting.
- 56 3. **Generation:** Generate new synthetic datasets based on the fitted model or its serialized  
57 representation.

### 58 Estimation

59 The generative model for multivariate datasets in metasyn makes the simplifying assumption  
60 of marginal independence: each column is considered separately, just as is done in e.g., naïve  
61 Bayes classifiers (Hastie et al., 2009). Formally, this leads to the following generative model for  
62 the  $K$ -variate data  $\mathbf{x}$ :

$$p(\mathbf{x}) = \prod_{k=1}^K p(x_k) \quad (1)$$

63 There are many advantages to this naïve approach when compared to more advanced generative  
64 models: it is transparent and explainable, it is able to flexibly handle data of mixed types, and  
65 it is computationally scalable to high-dimensional datasets. As mentioned before, the tradeoff  
66 is the limited analytical validity when the independence assumption does not hold: in the  
67 synthetic data, the expected value of correlations, regression parameters, and other measures  
68 of association is 0.

69 Model estimation starts with an appropriately pre-processed data frame. For metasyn, this  
70 means the data frame is tidy (Wickham, 2014), each column has the correct data type, and  
71 missing data are represented by a missing value. Internally, our software uses the polars data  
72 frame library (Vink et al., 2023), as it is performant, has consistent data types, and native  
73 support for missing data (null). A simple example source table could look like this (note that  
74 categorical data has the appropriate cat data type, not str):

ID	fruits	B	cars	optional
---	---	---	---	---
i64	cat	i64	cat	i64
1	banana	5	beetle	28
2	banana	4	audi	300
3	apple	3	beetle	null
4	apple	2	beetle	2
5	banana	1	beetle	-30

For each data type supported by metasyn, there is a set of candidate distributions that can be fitted to that data type (see Table Table 1). To estimate the generative model of Equation Equation 1, for each variable the software fits all compatible candidate distributions — by default with maximum likelihood estimation — and then selects the one with the lowest AIC (Akaike, 1973).

**Table 1:** Candidate distributions associated with data types in the core metasyn package.

Variable type	Data type	example	candidate distributions
continuous	float	1.0, 2.1, ...	UniformDistribution, NormalDistribution, ...
discrete	int	1, 2, ...	DiscreteUniformDistribution
categorical	pl.Categorical	gender, country	MultinoulliDistribution
structured string	str	Room number A108, C122	RegexDistribution
unstructured string	str	Names, open answers	FakerDistribution, LLMDistribution
temporal	Date, Datetime	2021-01-13, 01:40:12	DateUniformDistribution

Generative model estimation done in code as follows:

```
from metasyn import MetaFrame

# "ID" column is the primary key;
# ensure that it has unique values
# when data is synthesized later
spec_dict = {
    "ID": {"unique": True}
}

# create metaframe
mf = MetaFrame.from_dataframe(df, spec=spec_dict)
```

## Serialization and deserialization

After a fitted model object is created, metasyn allows it to be transparently stored in a human- and machine-readable .json file. This file can be considered as metadata: it contains dataset-level descriptive information as well as variable-level information. The metadata format has a

specific structure, which we call generative metadata format, or gmf. The header contains the following dataset-level information:

```
"n_rows": 5,
"n_columns": 5,
"provenance": {
  "created by": {
    "name": "metasyn",
    "version": "0.6.0",
    "privacy": null
  },
  "creation time": "2023-09-27T13:54:16.686166"
}
```

Then, for each variable the gmf file contains information the name, the data type, the proportion of missing values, and the distribution fitted on the data. For example, a table column containing different types of fruits could result in the following .json:

```
{
  "name": "fruits",
  "type": "categorical",
  "dtype": "<class 'polars.datatypes.Categorical'>",
  "prop_missing": 0.0,
  "distribution": {
    "name": "MultinoulliDistribution",
    "parameters": {
      "labels": ["apple", "banana"],
      "probs": [0.4, 0.6]
    }
  }
}
```

There are several advantages to creating such a serialized representation. First, it can be audited: the data owner can see exactly what information from the real data is made public through exporting the synthetic data, namely, the parameters of the distribution. Second, the file can be edited. For example, if a data owner thinks some of the labels of the “fruit” column contain sensitive information, these can simply be pseudonymized in the metadata file. Third, after exporting this file, an unlimited number of synthetic records can be created without incurring additional privacy risks, because the original data is no longer part of the synthetization process.

## Data generation

After creating either the fitted model object from the original data or by deserializing a model object from a .json file, new data can be generated by the object. For each variable in the model object, the software randomly samples from the fitted distribution to create a synthetic version of the data.

```
from metasyn import MetaFrame

# load json into a metadataset object
mf = MetaFrame.from_json("metasyn_example.json")

# create a fake dataset
mf.synthesize(10)
```

This may result in the following polars data frame. Note that missing values in the optional column are appropriately reproduced as well, courtesy of the “prop\_missing” entry in the

116 metadata format.

117 shape: (10, 5)

118

ID	fruits	B	cars	optional
---	---	---	---	---
i64	cat	i64	cat	i64
1	banana	4	beetle	null
2	banana	3	audi	null
3	banana	1	beetle	223
4	banana	0	beetle	258
...	...	...	...	...
7	banana	3	beetle	298
8	banana	2	beetle	67
9	banana	4	beetle	-30
10	banana	2	beetle	172

132

## 133 Plug-ins and automatic privacy

134 In addition to the core features described above, the metasyn package allows for plug-ins: add-  
135 on packages that alter the behaviour of the parameter estimation. Through this system, privacy  
136 guarantees can be built into metasyn. For example, a package called [metasyn-disclosure-](#)  
137 [control](#) implements the disclosure control output guidelines from Eurostat ([Bond et al., 2015](#))  
138 by re-implementing the `fit()` method of the candidate distributions shown in Table [Table 1](#)  
139 to include a micro-aggregation step. In this way, information transfer from the sensitive real  
140 data to the synthetic public data can be further reduced.

141 This plug-in system is user-friendly: the user only needs to `pip install` the package and then  
142 metasyn can automatically find it to make the methods accessible:

```
from metasyn import MetaDataset
from metasyncontrib.disclosure import DisclosurePrivacy

mds = MetaFrame.from_dataframe(df, privacy=DisclosurePrivacy())
```

## 143 Conclusion

144 Synthetic data is a valuable tool for communicating about sensitive datasets. In this work, we  
145 have presented the software metasyn, which allows data owners to generate a synthetic version  
146 of their sensitive tabular data with a focus on privacy and transparency. Unlike existing tools  
147 for generating synthetic data, we choose to aim for low analytic validity to enable high privacy  
148 guarantees: the underlying model makes a simplifying independence assumption, resulting in  
149 few parameters and thus a very small information transfer. This approach additionally allows  
150 for disclosure guarantees through a plug-in system.

151 Further documentation and examples can be found on [metasyn.readthedocs.io](#).

## 152 Acknowledgements

153 This research was conducted in whole or in part using ODISSEI, the Open Data Infrastructure  
154 for Social Science and Economic Innovations (<https://ror.org/03m8v6t10>)

The metasyn project is supported by the FAIR Research IT Innovation Fund of Utrecht University (as of March 2023)

## References

- Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. *2nd International Symposium on Information Theory*, 267–281.
- Bates, A., Spakulová, I., Dove, I., & Meador, A. (2019). *ONS methodology working paper series number 16—synthetic data pilot*.
- Bond, S., Brandt, M., & Wolf, P. de. (2015). *Guidelines for output checking*. eurostat.
- Dwork, C. (2006). Differential privacy. *International Colloquium on Automata, Languages, and Programming*, 1–12.
- Dwork, C., & Smith, A. (2010). Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2).
- Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (Vol. 2). Springer.
- Nowok, B., Raab, G. M., & Dibben, C. (2016). Synthpop: Bespoke creation of synthetic data in r. *Journal of Statistical Software*, 74, 1–26.
- Ping, H., Stoyanovich, J., & Howe, B. (2017). Datasynthesizer: Privacy-preserving synthetic datasets. *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, 1–5.
- Sweeney, L. (2002). K-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05), 557–570.
- Templ, M., Meindl, B., Kowarik, A., & Dupriez, O. (2017). Simulation of synthetic complex data: The r package simPop. *Journal of Statistical Software*, 79(10), 1–38.
- Vink, R., Gooijer, S. de, Beedie, A., Gorelli, M. E., Zundert, J. van, Hulselmans, G., Grinstead, C., Santamaria, M., Heres, D., ibENPC, Magarick, J., Leitao, J., Marshall, Wilksch, M., Heerden, M. van, Borchert, O., Jermain, C., Peek, J., Russell, R., ... Robert. (2023). *Pola-rs/polars: Python polars 0.18.8 (py-0.18.8)*. Zenodo. <https://doi.org/10.5281/zenodo.8167449>
- Wickham, H. (2014). Tidy data. *Journal of Statistical Software*, 59(10), 1–23. <https://doi.org/10.18637/jss.v059.i10>
- Wolf, P.-P. de. (2012). *Statistical disclosure control*. Wiley & Sons, Chichester.