The Journal of Open Source Software

# Pyinterpolate: Spatial interpolation in Python for point measurements and aggregated datasets

**Szymon Moliński**[1]

**1** Independent Researcher

**DOI:** 10.21105/joss.02869

**Software**

- Review
- Repository
- Archive

**Editor:** Hugo Ledoux

**Reviewers:**

- @chrisbrunsdon
- @kenohori
- @sdesabbata

**Submitted:** 26 October 2020
**Published:** 23 February 2022



## Summary

We use spatial interpolation techniques to interpolate values at unknown locations or filter and smooth existing data sources. Those methods work for point observations and areal aggregates. The basic idea behind the spatial interpolation algorithms is that every point in space can be described as a function of its neighbors' values weighted by the relative distance from the analyzed point. It is known as Tobler's First Law of Geography, which states: *everything is related to everything else, but near things are more related than distant things* (Tobler, 1970).

The Kriging technique, originally designed for mining applications, exploits this statement formally, and nowadays, it has gained a lot of attention outside the initial area of interest. Today Kriging is a set of methods applied to problems from multiple fields: environmental science, hydrogeology, natural resources monitoring, remote sensing, epidemiology and ecology, and even computer science (Chilès & Desassis, 2018). Commonly, Kriging is used to interpolate values from point measurements or regular block units. However, the real-world datasets are often different. Especially challenging is data that represents aggregated values over polygons, for example, the administrative units (Goovaerts, 2007).

Pyinterpolate transforms areas of irregular shapes and sizes with Area-to-Area and Area-to-Point Poisson Kriging functions. Those algorithms make Pyinterpolate beneficial for social, environmental, and public health scientists because they usually deal with areal counts instead of point measurements. Moreover, the package offers basic point Kriging and Inverse Distance Weighting techniques. Those algorithms are used in every field of research where geostatistical (distance) analysis gives meaningful results. Pyinterpolate merges basic Kriging techniques with more sophisticated Area-to-Area and Area-to-Point Poisson Kriging methods.

## Statement of need

Pyinterpolate is a Python package for spatial interpolation. It performs predictions from point measurements and areal aggregates of different sizes and shapes. Pyinterpolate automates Kriging interpolation, and semivariogram regularization. The package helps with data exploration, data preprocessing, and semivariogram analysis. A researcher with geostatistical background has control over the basic modeling parameters: semivariogram models, nugget, sill, range, and the number of neighbors included in the interpolation and Kriging type. The thing that makes Pyinterpolate different from other spatial interpolation packages is the ability to perform Kriging on areas of different shapes and sizes. This type of operation is essential in social, medical and ecological sciences (Goovaerts, 2007; Goovaerts & Gebreab, 2008; Kerry et al., 2013).

### Importance of areal (block) Kriging

There are many applications where researchers need to model areal data with irregular shapes and sizes. A good example is the public health sector, where data is aggregated over administrative units for patient protection and policy-making purposes. Unfortunately, this transformation makes data analysis and modeling more complex for researchers. There are different techniques to deal with this kind of data. We can work directly with areal aggregates or fall the irregular polygons into their centroids or, finally, transform dataset into a regular grid of smaller blocks if the point-support model is available. The latter is not a way to *get back* original observations but rather a form of lossy semivariogram transformation to the point-support scale. There are reasons to do it:

1. The presence of extremely unreliable rates that typically occur for sparsely populated areas and rare events. Consider the examples with the number of leukemia cases (numerator) per population size in a given county (denominator) or the number of whales observed in a given area (numerator) per time of observation (denominator). In those cases, extreme values may be related to the fact that variance for a given area of interest is high (low number of samples) and not to the fact that the chance of the event is exceptionally high for this region.
2. The visual bias. People tend to give more importance to large blocks in contrary to the small regions.
3. The mismatch of spatial supports for aggregated data and other variables. Data for spatial modeling should have harmonized spatial scale and the same extent. The aggregated datasets are not an exception. It may lead to the trade-off where we must aggregate other variables to build a model. Unfortunately, we lost a lot of information in this case. The other problem is that administrative regions are artificial constructs and aggregation of variables may remove spatial trends from data. A downscaling of areal data into filtered population blocks may be better suited to risk estimation along with remote-sensed data or in-situ observations of correlated variables (Goovaerts, 2006).

In this context, Area-to-Area Poisson Kriging serves as the noise-filtering algorithm or areal interpolation model, and Area-to-Point Poisson Kriging interpolates and transforms values and preserves the prediction coherence (where the disaggregated estimates sum is equal to the baseline area value) (Goovaerts & Gebreab, 2008). The chained-pipelines may utilize Area-to-Point Poisson Kriging, especially if scientist needs to change the support of variables. The author created a model of this type, the machine-learning pipeline with a model based on the remote-sensing data was merged with the geostatistical population-at-risk model derived from the Area-to-Point Poisson Kriging (the research outcomes are not published yet).

Alternatively to the Area-to-Area and Area-to-Point Poisson Kriging, researchers may use centroids and perform point kriging over a prepared regular point grid. However, this method has its pitfalls. Different sizes and shapes of the baseline units lead to the imbalanced number of variogram point pairs per lag. The centroid-based approach misses spatial variability of the linked variable, for example, population density over an area in the context of infection rates.

## Methodology

Pyinterpolate performs six types of spatial interpolation; inverse distance weighting and five types of Kriging:

1. **Ordinary Kriging** is a universal method for point interpolation.
2. **Simple Kriging** is a special case of point interpolation when the mean of the spatial process is known and does not vary spatially in a systematic way.
3. **Centroid-based Poisson Kriging** is used for areal interpolation and filtering. We assume that each block can collapse into its centroid. It is much faster than Area-to-Area and Area-to-Point Poisson Kriging but introduces bias related to the area's transformation into single points.

4. **Area-to-Area Poisson Kriging** is used for areal interpolation and filtering. The point-support allows the algorithm to filter unreliable rates and makes final areal representation of rates smoother.
5. **Area-to-Point Poisson Kriging** where areal support is deconvoluted in regards to the point support. Output map has a spatial resolution of the point support while coherence of analysis is preserved (sum of rates is equal to the output of Area-to-Area Poisson Kriging). It is used for point-support interpolation and data filtering.

The theory of Kriging is described in supplementary materials in the paper repository or in more detail in Armstrong (1998). Oliver & Webster (2015) point to the practical aspects of Kriging. The procedure of the interpolation with Poisson Kriging is presented in Goovaerts (2006) and the semivariogram regularization process is described in Goovaerts (2007).

The comparison to existing software is presented in the supplementary document here, Ordinary Kriging outcomes are compared for *gstat* and Pyinterpolate.

## Interpolation steps

The user starts with semivariogram exploration and modeling. Next, the researcher, or automatically with an algorithm, chooses the theoretical model which best fits the semivariogram. If this is done automatically, the algorithm tests linear, spherical and exponential models with different sills and ranges and the constant nugget against the experimental curve. Model performance is measured by the root mean squared error between the tested theoretical model with the experimental semivariance.

Areal data interpolation, especially transformation from areal aggregates into point support maps, requires deconvolution of areal semivariogram. Users may do it without prior knowledge of kriging and spatial statistics because an operation is automated. The iterative procedure of the semivariogram regularization is described in detail in Goovaerts (2007). The last step of analysis is a solution of linear Kriging equations.

Predicted data is stored as a `DataFrame` known from the *Pandas* and *GeoPandas* Python packages. Pyinterpolate allows the user to transform the point data into a regular Numpy array grid for further processing and analysis. Use case with the whole scenario is available in the paper package repository.

The package can automatically perform the semivariogram fitting step with a derivation of the theoretical semivariogram from the experimental curve. The semivariogram regularization is entirely automated. The process is described in Goovaerts (2007). Users can change the derived theoretical model only by directly overwriting the derived semivariogram model parameters (nugget, sill, range, model type).

The initial field of study (epidemiology) was the reason behind the automation of the tasks related to semivariogram modeling. Pyinterpolate was initially developed for the epidemiological research, where areal aggregates of infections were transformed to point support population-at-risk maps. It is assumed that users without a broad geostatistical background may use Pyinterpolate for spatial data modeling and analysis, especially users observing processes related to the human population.

The Figure 1 is an example of a full-scale process of the semivariogram regularization and Area-to-Point Poisson Kriging.

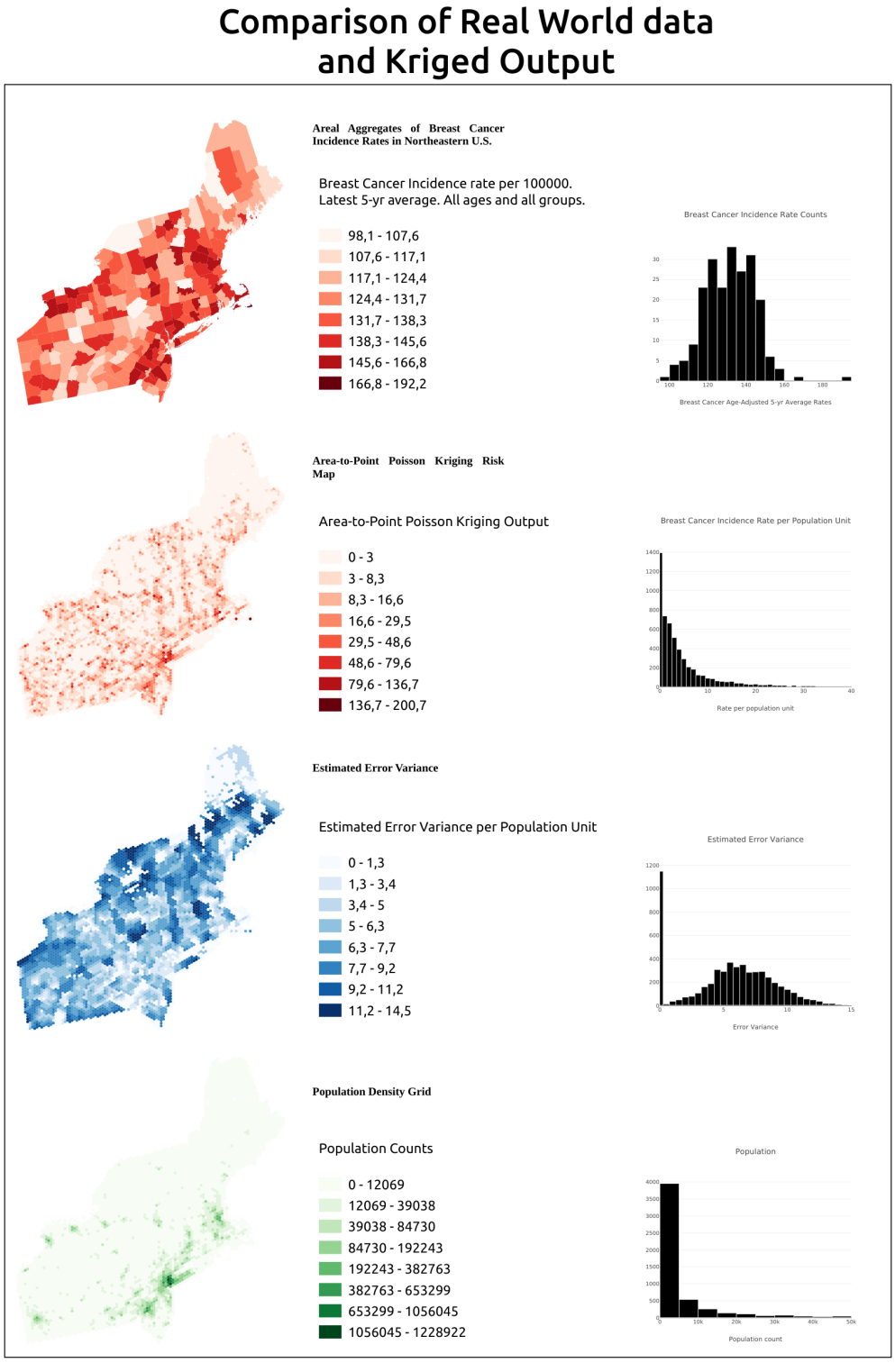


**Figure 1:** Example use case of Pyinterpolate for the derivation of the population-at-risk map for a cancer development from the areal aggregates and the population blocks.

The repository here presents an example of Poisson Kriging of cancer rates in North-Eastern U.S. step-by-step, with semivariogram regularization and Point Poisson Kriging functions. This

repository contains three documents with additional information:

1. *IPython* notebook with code.
2. Document with detailed description of methodology.
3. Document that describes the areal data transformation process. This procedure follows (Goovaerts, 2007).

### Modules

Pyinterpolate has seven modules covering all operations needed to perform spatial interpolation: input/output operations, data processing, transformation, semivariogram fitting, Kriging interpolation. Figure 2 shows the internal package structure.

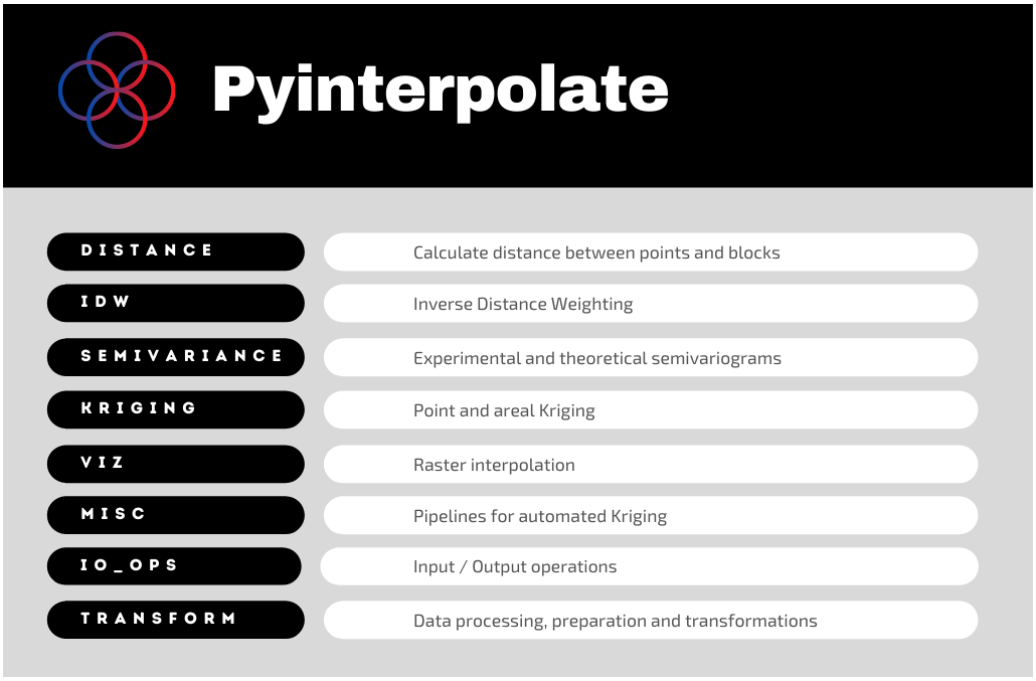


**Figure 2:** Structure of Pyinterpolate package.

## Comparison to Existing Software

The main difference between Pyinterpolate and other packages is that it focuses on areal deconvolution methods and Poisson Kriging techniques useful for ecology, social science and public health studies. Potential users may choose other packages if they can perform their research with the point data interpolation.

The most similar and significant package from the Python environment is *PyKrige* (Murphy et al., 2020). PyKrige is designed especially for point kriging. PyKrige supports 2D and 3D ordinary and universal Kriging. User can incorporate their own semivariogram models and use external functions (as an example from *scikit-learn* package (Pedregosa et al., 2011)) to model drift in universal Kriging. The package is actively maintained.

*GRASS GIS* (GRASS Development Team, 2020) is a well-established software for vector and raster data processing and analysis. GRASS contains multiple modules and a user may access them in numerous ways: GUI, command line, C API, Python APU, Jupyter Notebooks, web, QGIS or R. GRASS has three functions for spatial interpolation:

- `r.surf.idw` and `v.surf.idw`: both use Inverse Distance Weighting technique, first interpolates raster data and second vectors (points).
- `v.surf.rst` that performs surface interpolation from vector points map by splines. Spline interpolation and Kriging are compared in (Dubrule, 1984).

*PySAL* is the next GIS / geospatial package that is used for spatial interpolation. However, PySAL is built upon the spatial graph analysis algorithms. The areal analysis is performed with the sub-module *tobler* (Knaap et al., 2020). Moreover, the package has functions for multisource regression, where raster data is used as auxiliary information to enhance interpolation results.

The R *gstat* package is another option for spatial interpolation and spatial modeling (Pebesma, 2004). The package is designed for variogram modeling, simple, ordinary and universal point or block kriging (with drift), spatio-temporal kriging and sequential Gaussian (co)simulation. Gstat is a solid Kriging and spatial interpolation package and has the largest number of methods to perform spatial modeling. The main difference between gstat and Pyinterpolate is the availability of area-to-point Poisson Kriging in the latter and the difference between baseline programming languages (Goovaerts, 2007). The functional comparison to gstat is available in the paper repository.

## References


Armstrong, M. (1998). *Basic linear geostatistics*. Springer. https://doi.org/10.1007/978-3-642-58727-6

Chilès, J.-P., & Desassis, N. (2018). Fifty years of kriging. In B. S. Daya Sagar, Q. Cheng, & F. Agterberg (Eds.), *Handbook of mathematical geosciences: Fifty years of IAMG* (pp. 589–612). Springer International Publishing. https://doi.org/10.1007/978-3-319-78999-6_29

Dubrule, O. (1984). Comparing splines and kriging. *Computers & Geosciences*, *10*(2), 327–338. https://doi.org/10.1016/0098-3004(84)90030-X

Goovaerts, P. (2006). Geostatistical analysis of disease data: Accounting for spatial support and population density in the isopleth mapping of cancer mortality risk using area-to-point poisson kriging. *International Journal of Health Geographics*, *5*. https://doi.org/10.1186/1476-072X-5-52

Goovaerts, P. (2007). Kriging and semivariogram deconvolution in the presence of irregular geographical units. *Mathematical Geosciences*, *40*, 101–128. https://doi.org/10.1007/s11004-007-9129-1

Goovaerts, P., & Gebreab, S. (2008). How does poisson kriging compare to the popular BYM model for mapping disease risks? *International Journal of Health Geographics*, *7*, 6. https://doi.org/10.1186/1476-072x-7-6

GRASS Development Team. (2020). *Geographic resources analysis support system (GRASS GIS) software*. Open Source Geospatial Foundation. https://grass.osgeo.org

Kerry, R., Goovaerts, P., Smit, I. P. J., & Ingram, B. R. (2013). A comparison of multiple indicator kriging and area-to-point poisson kriging for mapping patterns of herbivore species abundance in kruger national park, south africa. *International Journal of Geographical Information Science*, *27*, 47–67. https://doi.org/10.1080/13658816.2012.663917

Knaap, E., Cortes, R. X., Rey, S., Gaboardi, J., & Frontiera, P. (2020). *Pysal/tobler: Release v0.5.4* (Version v0.5.4) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.4385980

Murphy, B., Müller, S., & Yurchak, R. (2020). *GeoStat-framework/PyKrige v1.5.1* (Version v1.5.1) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.3991907

Oliver, M., & Webster, R. (2015). *Basic steps in geostatistics: The variogram and kriging*. Springer. https://doi.org/10.1007/978-3-319-15865-5

Pebesma, E. J. (2004). Multivariable geostatistics in s: The gstat package. *Computers & Geosciences*, *30*(7), 683–691. https://doi.org/10.1016/j.cageo.2004.03.012

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Tobler, W. R. (1970). A computer movie simulating urban growth in the detroit region. *Economic Geography*, *46*, 234. https://doi.org/10.2307/143141