Geostatistics - Geostatistical simulation: sequential indicator simulation

Julián M. Ortiz - https://julianmortiz.com/

September 28, 2024

Summary

Sequential indicator simulation is an extension of multiple indicator kriging and can also be adapted to handle categorical variables. The principle is simple: build the conditional distribution by discretizing it using a set of thresholds, in the case of continuous variables, or by inferring the probability of prevalence of each category in the case of categorical variables.

The sequential simulation framework is used in the same manner as in the case of sequential Gaussian simulation: from the sample data, the conditional distribution is inferred at an unsampled location. Monte Carlo simulation is used to draw a simulated value from that distribution, which is used to condition all subsequent nodes. This is repeated until the simulation grid is complete and can be repeated to create

different realizations. We discuss some of the practical aspects of sequential indicator simulation.

1 Sequential indicator simulation

1.1 Theoretical foundation

Unlike sequential Gaussian simulation, the sequential indicator approach is not based on a distributional assumption. On the contrary, the indicator approach seeks to determine the shape of the conditional probability density function by discretizing and estimating it for different thresholds, or by computing the probability mass function, inferring the probability of prevalence of each category. The sequential principle, however, is exactly the same: simulated nodes are conditioned to sample values (transformed to indicators in this case), and on previously simulated values. This imposes the spatial correlation between simulation locations. K independent kriging runs are preformed (where K is the number of thresholds or the number of categories). K indicator variograms are therefore needed.

Sequential indicator simulation proceeds in a similar fashion as indicator kriging. Data are coded as probabilities and treated accordingly. For continuous variables, the indicator coding represents the probability of not exceeding a threshold, while for categorical variables, it just represents the probability of prevalence of a category at a location.

$$i(\boldsymbol{u_\alpha}; z_k) = Prob\{z(\boldsymbol{u_\alpha}) \leq z_k\} \quad \forall k = 1, \ldots, K$$

$$i(\boldsymbol{u_\alpha}; s_k) = Prob\{s(\boldsymbol{u_\alpha}) = s_k\} \quad \forall k = 1, \ldots, K$$

The resulting estimates of the indicator variables need to comply with the rules of probability distribution (a probability density or a probability mass function). This means that in the case of continuous variables, the indicator kriging estimates must lie in the interval $[0, 1]$ and as thresholds increase, estimated probabilities (that is, the indicator kriging estimates) must not decrease (monotonic non-decreasing function). Now, the method does not guarantee this, therefore, any departures known as order relation deviations, must be corrected *a posteriori*. In the case of categorical variables, again, the estimated probabilities cannot lie outside the interval $[0, 1]$ and must sum to 1, given that the K categories are considered exhaustive and mutually exclusive. In case the sum of estimated probabilities does not sum to 1, then, the probabilities are scaled to ensure closure of the probability mass function.

1.2 Steps

The steps for sequential indicator simulation are the same as those for indicator kriging, discussed before. Changes occur due to the sequential nature of the implementation of the simulation and also by the fact that instead of keeping the conditional distribution at each location, a simulated value is drawn by Monte Carlo simulation.

The steps to implement sequential indicator simulation in the **continuous case** are:

1. **Representative distribution**: decluster the data to

obtain a representative distribution.

2. **Define thresholds for indicator coding**: from the declustered distribution define a set of K thresholds to discretize evenly the distribution and characterize important values or the tails of the distribution with enough detail.
3. **Indicator coding**: transform the sample data to indicators according to the formula:

$$i(\boldsymbol{u_\alpha}; z_k) = \begin{cases} 1, & \text{if } z(\boldsymbol{u_\alpha}) \leq z_k \\ 0, & \text{otherwise} \end{cases} \quad k = 1, \ldots, K$$

4. **Indicator variograms**: compute the experimental indicator variogram for each threshold and fit a model. Make sure the models change smoothly from one category to the next.
5. **Perform the simulation**:

   (a) **Visit a node**: following the random path visit a node and check that it is not informed by a sample or has been previously simulated.

   (b) **Search for data and previously simulated nodes in neighborhood**: find the nearby sample data and any previously simulated nodes in the search neighborhood.

   (c) **Perform simple indicator kriging for each threshold**: perform K simple indicator kriging estimations. In each case, keep the kriging estimate. The indicator kriging variance is not used.

   (d) **Correct order relation deviations**: verify and correct order relation deviations to ensure the collection of K indicator kriging estimates comply with

the conditions of a cumulative distribution function.

(e) **Complete the conditional distribution by interpolation between thresholds and extrapolation beyond the first and last thresholds**: select interpolation and extrapolation functions to complete the conditional distribution discretized by the simple indicator kriging estimates for each threshold.

(f) **Simulate a value from the conditional distribution**: Using the completed conditional distribution, simulate a value by Monte Carlo simulation. This entails drawing a uniform random value in the interval $(0, 1)$ and numerically computing the corresponding quantile.

(g) **Go back to 5a**: if there are remaining nodes to be visited in the random path, otherwise, stop or start a new realization.

The steps to implement sequential indicator simulation in the **categorical case** are:

1. **Representative distribution**: decluster the data to obtain a representative distribution of the proportions of each category.
2. **Indicator coding**: transform the sample data to categorical indicators according to the formula:

$$i(\boldsymbol{u_\alpha}; s_k) = \begin{cases} 1, & \text{if } s(\boldsymbol{u_\alpha}) = s_k \\ 0, & \text{otherwise} \end{cases} \quad k = 1, \ldots, K$$

3. **Indicator variograms**: compute the experimental indicator variogram for each category and fit a model.

Notice that in the binary case, a single model is required as the variograms for both categories are identical. In general, when K categories are available there are $K-1$ degrees of freedom in the modeling of the variograms.

4. **Perform the simulation**:

   (a) **Visit a node**: following the random path visit a node and check that it is not informed by a sample or has been previously simulated.

   (b) **Search for data and previously simulated nodes in neighborhood**: find the nearby sample data and any previously simulated nodes in the search neighborhood.

   (c) **Perform simple indicator kriging for each category**: perform K simple indicator kriging estimations. In each case, the kriging estimate represents the probability of prevalence of the category. The indicator kriging variance is not used.

   (d) **Correct distribution inconsistencies**: verify and correct the collection of K indicator kriging estimates so they comply with the conditions of a probability mass function.

   (e) **Simulate a value from the conditional distribution**: Using a fixed order of the categories, build a cumulative distribution function and simulate a value by Monte Carlo simulation.

   (f) **Go back to 4a**: if there are remaining nodes to be visited in the random path, otherwise, stop or start a new realization.

As usual, the result of the process is a set of multiple realizations, reflecting the variability expected in space and the uncertainty at every location.

Post-processing and other implementation details are similar to those in sequential Gaussian simulation.

Index