Multivariate Standard Normal Transformation

Clayton V. Deutsch

Transforming K *regionalized variables with complex multivariate relationships to* K *independent multivariate standard normal variables is an important problem. Univariate transformation is straightforward and multivariate transformation of non-standard variables is straightforward. The transformation becomes more difficult when there are complex non-Gaussian features such as non-linearity, heteroscedatsticity or constraints. The stepwise conditional transform is useful, but requires many data and there are always some minor artifacts due to binning and the reduced data in the transform of high dimensional distributions. A new technique is developed in this paper. Consider transforming* N *observations of K variables from an arbitrary multivariate distribution to* N *observations of* K *variables from a multivariate standard normal distribution. The first step is to generate* N *observations from a* K *variate multivariate standard normal distribution using a modified latin hypercube sampling to have values as standard normal as possible. Then, each of the original* N *data are matched to a standard normal observation. The key is to match values in a way that preserves the structure of the variables.*

Introduction

Multivariate transformation is a longstanding problem in multivariate geostatistics. A variety of techniques are used, but most are for variables that are already Gaussian (principal components, minimax autocorrelation factors,…) or are for a limited number of variables (stepwise, kernel estimation,…). The guidebook by Ryan Barnett would be a good place to review the range of available techniques with software.

The stepwise transform is a powerful non-parametric transformation technique introduced to geostatistics by Leuangthong a number of years ago, see her thesis and many CCG papers. This transformation is difficult to apply when the number of variables becomes large. Practically, it is only possible to transform three variables simultaneously. In presence of more variables, one could specify a hierarchy and transform subsets, but this is somewhat unsatisfying. Moreover, even with three variables there is always the potential to introduce binning artifacts.

Kernel density estimation as proposed by numerous authors including Sahyun Hong and John Manchuk of CCG has an important place in multivariate geostatistics. The challenge is that storing and manipulating a large multivariate distribution becomes difficult. Consider a K=10 variate distribution discretized by 100 bins representing the univariate percentiles. There would be 100^{10} numbers required to store the distribution explicitly. Some computational tricks could be used to reduce the values stored, but the full distribution would be required to implement the required marginalization and calculation of conditional distributions.

The idea proposed here is a direct transformation with no binning and no calculation of conditional distributions. Each multivariate data observation is directly mapped to a point in a standard normal distribution of the same dimension. The standard normal values are independent. Determining this mapping is an optimization problem. Estimation and simulation proceeds in the standard normal space independently for each variable, then values are mapped back to the original space. The idea is closely related to the normal score transform and the stepwise conditional transform in the sense that each data observation in original units is mapped to one observation in a standard normal distribution. The difference is that multivariate observations are mapped directly; normal scores and stepwise consider only one variable at a time.

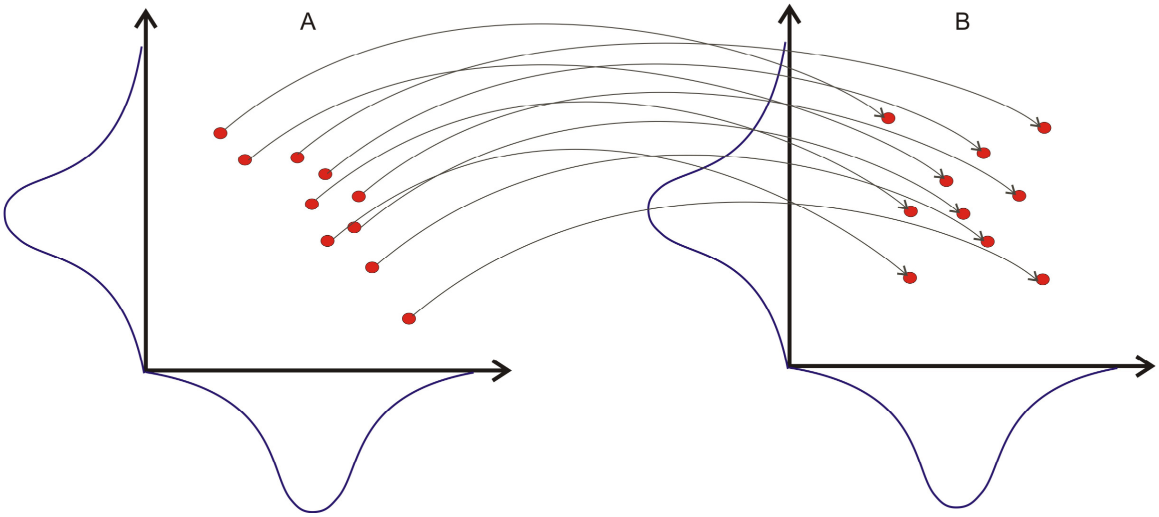
The key idea is documented below with prototype code (the standard GSLIB-like programs). There are a number of implementation challenges to overcome, but the idea is simple and demonstrates great potential as another tool in our growing arsenal of multivariate geostatistical tools.

Proposed Methodology

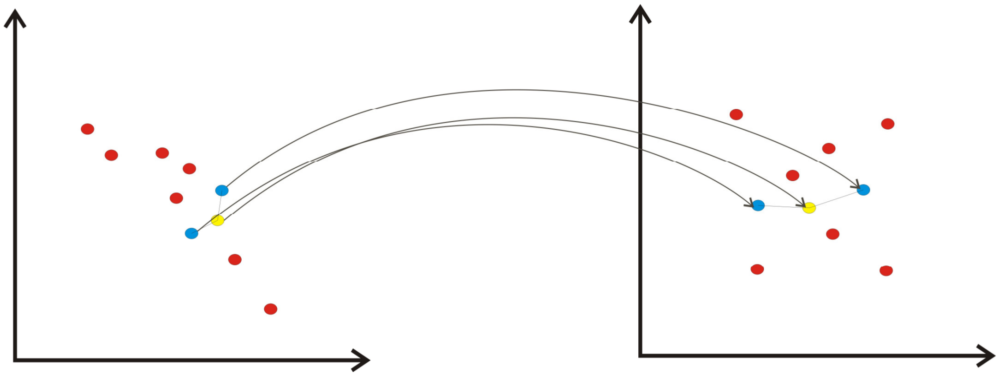
The methodology will be explained by describing the steps and showing some figures. The starting point is N observations of K variables in some arbitrary unit system. It may be convenient to independently transform each of the *K* variables to be standard normal to avoid concerns about drastically different units. The data variables could also simply be standardized. This prior standardization is not required, but could be applied without loss of generality.

The same number of samples *N* are drawn from a standard normal distribution of *K* variables. The *K*-variate standard normal distribution is independent. Like the stepwise approach, this will permit quick and efficient simulation because each of the transformed variables can be treated independently. The *N* observations should uniformly sample the *K* dimensional in a reasonable manner. We could consider generating the samples by Monte Carlo Simulation, but it would be more efficient to implement a type of directed sampling like Latin hypercube sampling (LHS), perhaps with multidimensional uniformity, see paper 2009-125. The current program (see below) has a simple type of LHS implemented whereby each univariate distribution is exactly the same (standard normal to the extent that the number *N* would allow) and avoiding the same values in the same bins of the distribution.

The central step is to map each of the *N* original data to one of the *N* generated multivariate standard normal observations. This is shown as a bivariate cartoon below with 10 observations. This picture does not capture the fact that we will be working in a *K*-dimensional space with *K* as large as 100. This picture also does not capture the fact that we would normally have 100s of data points to start with. The criteria for the best mapping and a procedure to get that mapping are key.



A number of criteria could be considered. Any mapping would achieve a transform to values that are independent multivariate standard normal, but we want a mapping with some specific features (1) preservation of the important spatial features of the input variables, (2) minimal distortion of multivariate features so that they would be preserved in back transformation, and (3) a transform that facilitates back transformation of simulated points that may not coincide with the starting N points in Gaussian units. These criteria are summarized by the notion that each data point should be close to its neighbors in transformed space. Consider the yellow data point in original units (below left) and its mapped point to the right. The mapping of the two nearest neighbors in original units is identified in blue on the left and on the right. A good mapping would keep the proximity of the original neighbors.



Each of the original *N* data points would have to be considered and some number *M* of nearest neighbors for each. The code documented below permits an arbitrary number of neighbors, but a limited number of,say, 2 or 3 seems to work best. Considering too many neighbors appears to reduce the precision/resolution of the transformation. Note that the actual distance in original units is only used to identify neighbor points. A standardization of the original units is still recommended because the neighbors are computed in the K-dimensional space of the original units and if one dimension has large units, then it would overly influence the selection of neighbors.

The original data are ordered 1 through *N*. The nearest *M* data to each of the original data is identified once and is not changed. The "mapping" of the input data space to the target multidimensional normal distribution is a list of numbers *o(i),i=1,…,N*. Each of the *o(i)* values is an integer between 1 to *N* and there are no repeats. This provides a unique mapping. The notation below is a little loose because it is easier to drop the mapping function for simplicity.

The objective function to be minimized is the sum of squared distances to the neighbors to each point in the mapped space. It could be written in the following form:

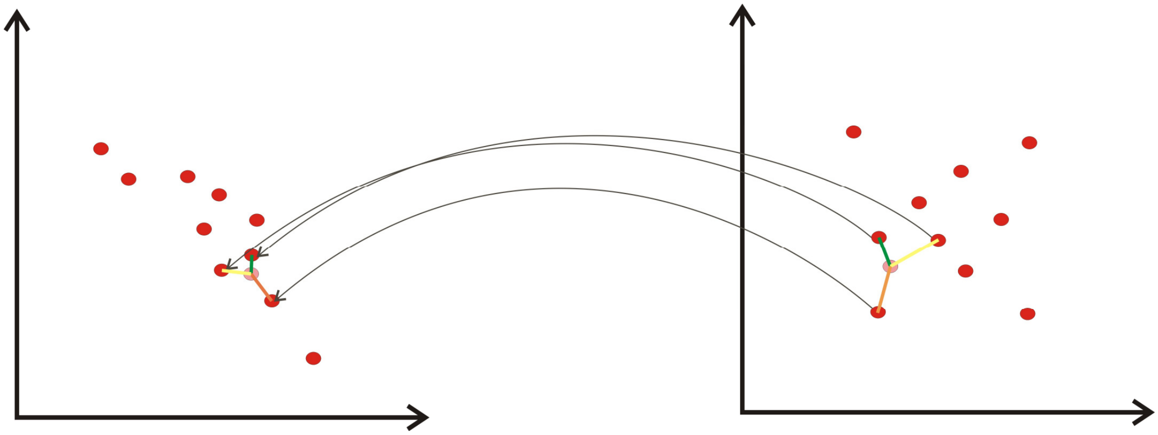
$$O = \sum_{i=1}^{N} \sum_{m=1}^{M} d_{o(i)o(m)}^{2}$$

Where $d_{i(i)o(m)}$ is the Euclidean distance in the *K*-dimensional multivariate normal space between the mapped location of data *i* and the mapped location of its m^{th} neighbor. In most cases, the *N*x*N* distance matrix could be computed once, stored and queried when needed to compute the objective function. This may become intractable if there is more than *N*=10000 data points, but that is uncommon. The geomodeler would likely subdivide their domain if there were that many data points.

The initial mapping is random, that is, the *o(i)* vector is a randomly sorted set of integers between 1 and *N*. The initial objective function is computed as above. Then, the vector is perturbed pairwise. This swapping of ordering is convenient because it preserves the constraint that the mapping is unique and one-to-one. The objective function can be recalculated after a perturbation. The first idea was to update the objective function instead of recalculating, but the logic is complex because the nearby mapped points may also be changing because there are two points changing. The algorithm is fast enough that a simple recalculation was kept.

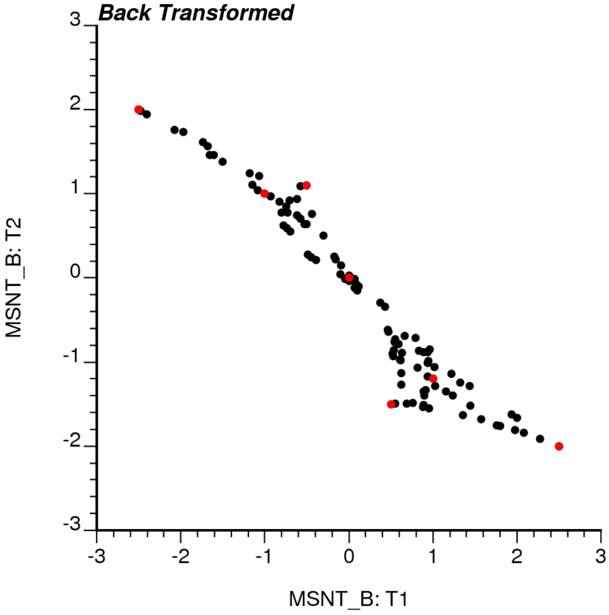
A full simulated annealing framework is implemented in the program, but the decision rule for changes could be greedy, that is, accept changes where the objective function decreases and reject changes where the objective function increases. Convergence is rapid and the program is hard coded to attempt far more changes than necessary to achieve stable results.

Back transformation of the data points back to original values is unique because of the unique forward mapping; however, the back transformation of values that do not coincide with the data values requires some attention. The idea is shown on the sketch below – the close values in transformed normal values are found and the back transformed values consider the corresponding original values and the distances to each of them. An inverse distance weighting is considered. Data exactitude is enforced because the weight to a collocated value in Gaussian units would be one and the weight to all others would be zero.



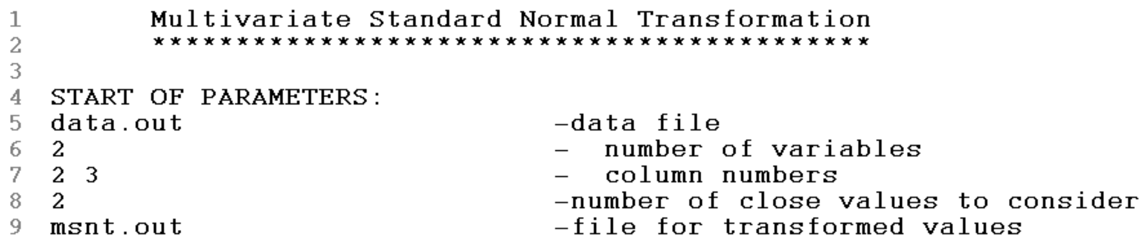
This back transformation scheme works well, but there some additional variability is added because nearby transformed normal values would be transformed to different values once the nearest neighbor changes. Only a few close values should be considered to avoid back transformation to a region of the input multivariate space that is not sampled by the input data.

The following example considers three values, which is consistent with each value being transformed considering the distance to the two neighbors. The red dots are seven original data and the black dots are 100 simulated values.



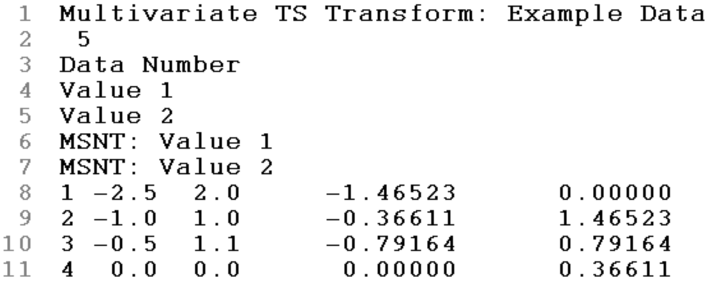
Programs

The parameters and source code for the proposed algorithm are relatively straightforward. The parameters for the `msnt` transformation are shown below.



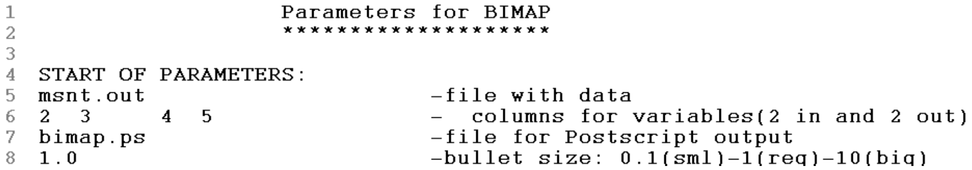
```
        Multivariate Standard Normal Transformation
        *******************************************

START OF PARAMETERS:
data.out                        -data file
2                               -  number of variables
2 3                             -  column numbers
2                               -number of close values to consider
msnt.out                        -file for transformed values
```

The input data file is in standard GSLIB format with all of the input variables. The data can be in any units, but it is best if they are in Gaussian units so that different univariate distributions do not cause any artifacts. The number of close values is the number of neighbors (M) described above. A random number seed is also required, but any value could be chosen. The output file contains all of the input values plus K standard normal values that accompany each of the input values, see a few lines of one output file shown below.

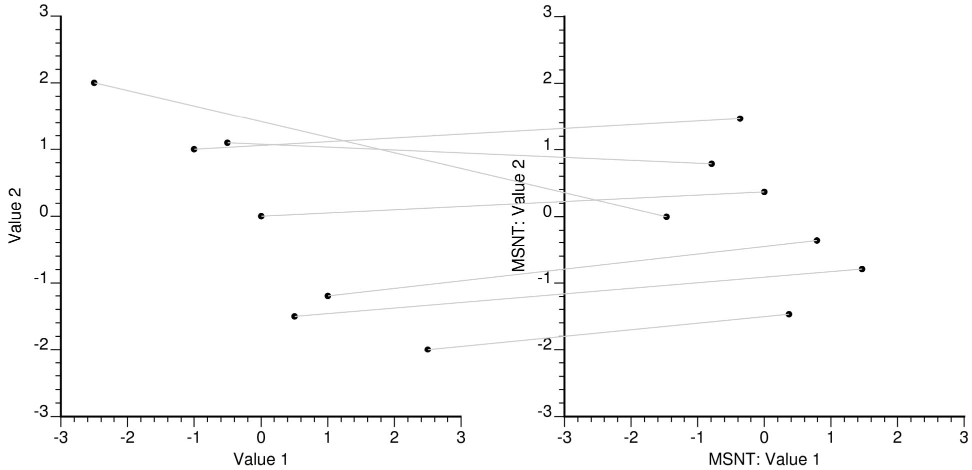


```
Multivariate TS Transform: Example Data
 5
Data Number
Value 1
Value 2
MSNT: Value 1
MSNT: Value 2
1 -2.5  2.0     -1.46523      0.00000
2 -1.0  1.0     -0.36611      1.46523
3 -0.5  1.1     -0.79164      0.79164
4  0.0  0.0      0.00000      0.36611
```

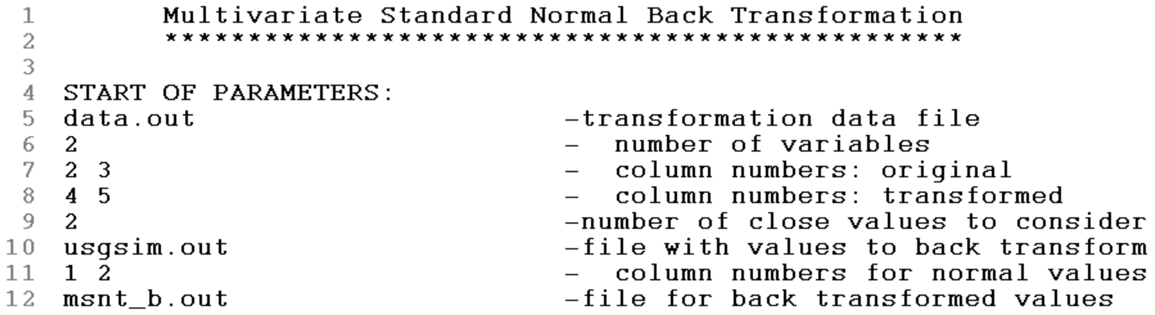
The bimap program will plot how pairs of the input variables are mapped to pairs of the output variables. This program helps check the results. The parameters and an example output file are given below. This program assumes that the input variables are standard normal.



```
                    Parameters for BIMAP
                    ********************

START OF PARAMETERS:
msnt.out                     -file with data
2  3    4  5                 -  columns for variables(2 in and 2 out)
bimap.ps                     -file for Postscript output
1.0                          -bullet size: 0.1(sml)-1(reg)-10(big)
```



The msnt_b program does the back transformation. The parameters are shown below. A different number of close values (or neighbors) could be considered; however, it would be common to choose the same number as the transformation. Choosing more could lead to artifacts because the distance to the far points could be too large and the final values would be too random.



```
      Multivariate Standard Normal Back Transformation
      ************************************************

START OF PARAMETERS:
data.out                      -transformation data file
2                             -  number of variables
2 3                           -  column numbers: original
4 5                           -  column numbers: transformed
2                             -number of close values to consider
usgsim.out                    -file with values to back transform
1 2                           -  column numbers for normal values
msnt_b.out                    -file for back transformed values
```

Discussion

There are some important challenges that must be overcome before the widespread use of this idea. Firstly, the use of declustering weights and representative distributions of the original variables is not straightforward. There is an implicit assumption that the input distributions are representative. Also, this transformation could only be applied if there are enough data to span the input multivariate space in a reasonable fashion. If there are few points, then the neighbors are far apart in both the original space and in the transformed space; the interpolation between points becomes important.

An important challenge of many multivariate techniques is the requirement for equal sampling, that is, all variables at all locations. This is a common problem, but it must be addressed.

Conclusion

This is an exciting new technique that has great potential in multivariate geostatistics. It has the promise to overcome some of the limitations of the stepwise and kernel techniques. Additional development is required, but the prototype code accompanying this paper provides a good starting point for additional research.

References

Barnett, R.M., 2011. Conditional Standardization: A Multivariate Transformation for the Removal of Non-linear and Heteroscedastic Features, CCG Paper 2011-310.

Deutsch, CV, Journel, AG. 1998. GSLIB: Geostatistical Software Library and User’s Guide: 2nd edition. New York: Oxford University Press.

Deutsch J. L. and C. V. Deutsch, 2009, "Latin Hypercube Sampling with Multidimensional Uniformity", CCG Paper 2009-125, Centre for Computational Geostatistics, University of Alberta, Edmonton, Canada

Johnson, RA, Wichern, DW. 1988. Applied Multivariate Statistical Analysis. New Jersey: Prentice Hall. p. 340 – 370.

Leuangthong, O. 2003. Stepwise Conditional Transformation for Multivariate Geostatistical Simulation, PhD Thesis, University of Alberta.

Leuangthong, O, Deutsch, CV. 2003. Stepwise Conditional Transformation for Simulation of Multiple Variables. Mathematical Geology. Vol.35, No.2: p155-172.