Prediction of missing values. Comparison of approaches

Alexander Litvinenko¹,*, Ronald Kriemann²,***, and Vladimir Berikov³,4,***
¹ RWTH Aachen, Aachen, Germany
² Max Planck Institute for Mathematics in the Sciences, Inselstr. 22, Leipzig
³ Sobolev Institute of Mathematics, Novosibirsk, Russia
⁴ Novosibirsk State University, Pirigova 1, Novosibirsk, Russia

We compare four numerical methods for the prediction of missing values in four different datasets [1]. These methods are 1) the hierarchical maximum likelihood estimation (H-MLE), and three machine learning (ML) methods, which include 2) k-nearest neighbors (kNN), 3) random forest, and 4) Deep Neural Network (DNN). From the ML methods, the best results (for considered datasets) were obtained by the kNN method with three (or seven) neighbors. On one dataset, the MLE method showed a smaller error than the kNN method, whereas, on another, the kNN method was better. The ML method requires a lot of linear algebra computations and works fine on almost all datasets. Its result can be improved by taking a smaller threshold and more accurate hierarchical matrix arithmetics. To our surprise, the well-known kNN method produces similar results as H-MLE and worked much faster.

1 Problem settings, H-MLE and Machine Learning methods

Let \((s_1, \ldots, s_n)\) be a set of spatial locations. Let \(I_1 := (s_1, \ldots, s_{n_1})\) be locations with known values \(Z_1 = (Z(s_1), \ldots, Z(s_{n_1}))^\top\), and \(I_2 = (s_{n_1+1}, \ldots, s_n)\) be another locations with unknown values \(Z_2 = (Z(s_{n_1+1}), \ldots, Z(s_n))^\top\). The values \(Z_2\) should be predicted.

H-MLE method and parameter inference. We model the set of measurements as a realization from a stationary Gaussian spatial random field. Specifically, we let \(Z = (Z(s_1), \ldots, Z(s_n))^\top\), where \(Z(s)\) is a Gaussian random field indexed by a spatial location \(s \in \mathbb{R}^d, d = 2, 3\). Then, we assume that \(Z\) has zero mean and a stationary parametric covariance function \(C(h; \theta) = \text{cov}(Z(s), Z(s + h))\), where \(h \in \mathbb{R}^d\) is a spatial lag vector and \(\theta \in \mathbb{R}^\nu\) the unknown parameter vector of interest. Statistical inferences about \(\theta\) are often based on the Gaussian log-likelihood function:

\[
L(\theta) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log |C(\theta)| - \frac{1}{2} Z^\top C(\theta)^{-1} Z,
\]

where the covariance matrix \(C(\theta)\) has entries \(C(s_i - s_j; \theta), i, j = 1, \ldots, n\). The maximum likelihood estimator of \(\theta\) is the value \(\hat{\theta}\) that maximizes (1). When the sample size \(n\) is large, the evaluation of (1) becomes challenging. Indeed, the storage of the \(n\times n\) covariance matrix \(C\) requires \(O(n^2)\) units of memory. Computation of the inverse and log-determinant of \(C(\theta)\) cost \(O(n^3)\) FLOPs. Hence, parallel and scalable methods that can reduce this high cost are needed.

We consider the Matérn covariance, which depends only on the distance \(h := ||s - s'||\), where \(s\) and \(s'\) are any two spatial locations:

\[
C(h; \theta) = \frac{\sigma^2}{2^{\nu-1} \Gamma(\nu)} \left( \frac{h}{\ell} \right)^{\nu} K_\nu \left( \frac{h}{\ell} \right) + \tau^2 I,
\]

with parameters \(\theta = (\sigma, \ell, \nu, \tau)^\top\). Here \(\sigma^2\) is the variance, \(\tau^2\) the nugget, \(\nu > 0\) controls the smoothness of the random field, with larger values of \(\nu\) corresponding to smoother fields, and \(\ell > 0\) the spatial range parameter that measures how quickly the correlation of the random field decays with distance. \(K_\nu\) denotes the modified Bessel function of the second kind of order \(\nu\), and \(I\) the identity matrix.

Prediction: Estimating the unknown parameters \(\theta\) is only an intermediate step. Once it is done, the estimation \(\hat{\theta} \approx \theta\) is used for the prediction at new locations. \(Z(s)\) is a Gaussian random field indexed by spatial locations with indices from the index set \((I_1, I_2)\). After discretisation we can get a \(2 \times 2\) block covariance matrix, consisting of 4 sub-blocks \(C_{11}; C_{12}; C_{21}; C_{22}\), where \(C_{11} \in \mathbb{R}^{n_1 \times n_1}\), and \(C_{22} \in \mathbb{R}^{(n-n_1) \times (n-n_1)}\). Now, the unknown vector \(Z_2\) can be computed by the following formula \(Z_2 = C_{21} C_{11}^{-1} Z_1\). We can also say that \(Z_2\) has the conditional distribution with the mean value \(C_{21} C_{11}^{-1} Z_1\) and the covariance matrix \(C_{22} - C_{21} C_{11}^{-1} C_{22}\).

The classical maximum likelihood method is very time-consuming for large datasets. To make computations faster, we approximated the joint Gaussian log-likelihood function in the hierarchical (H-) matrix format [2,3]. The H-matrix technique drastically reduces the required memory and computing time, making it possible to work with larger sets of observations obtained on unstructured meshes.

* Corresponding author: e-mail litvinenko@uq.rwth-aachen.de,
** e-mail rok@mis.mpg.de
*** e-mail berikov@math.nsc.ru

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.
1.1 Machine Learning methods

$k$-nearest neighbours (kNN) is a non-parametric statistical machine learning method, which follows a simple idea: for each data point $x$ for which one needs to predict its output $\hat{y}$, find its $k$ nearest neighbors $x_1, \ldots, x_k$ with respect to some metrics, and set $\hat{y} = \frac{1}{k} \sum_{i=1}^{k} y_i$, where $y_i$ is the observed value at $x_i$. The value of $k$ should be determined in the best way, for example, by a cross-validation procedure.

Random Forest (RF) is another popular machine learning method in which a large number of decision (or regression) trees are generated independently on random sub-samples of data. The final decision for $x$ is calculated over the ensemble of trees by averaging the predicted outcomes. The method shows promising results in many practical tasks, especially in the presence of multiple irrelevant features describing the observed data.

Deep Neural Network (DNN) methods are based on the artificial neural network paradigm, which models the functioning of neurons in the brain. We use a fully connected neural network (FCNN). FCNN training procedure consists of finding neurons’ connection weights for which the quality metric takes the best value.

By minimizing the RMSE error using a validation sample repeatedly obtained by random sub-sampling of data in the proportion 1:9, we found optimal values of $k$ for kNN, the number of trees for RF, and the number of layers and hidden units for FCNN. We considered values $k \in \{1, \ldots, 20\}$, the number of trees in the set $\{100, \ldots, 150\}$ and examined some variants of FCNN architecture for different number of hidden layers in the interval $[3, 10]$ (having 50-100 neurons in each hidden layer).

1.2 Numerical tests

We considered four datasets, taken from [4, 5]: two in Test 2a, and two in Test 2b. All datasets in Tests 2a contain 90,000 locations for training and 10,000 for testing (prediction). Both datasets in Test 2b contained 900,000 locations for training and 100,000 for prediction.

\(H\)-MLE can require up to 8 hours on a modern parallel node for the dataset with 900,000 locations. A possible remedy is to precompute the initial guess. It will significantly reduce the required number of iterations to minimize the log-likelihood and the total computing time. A good initial guess can be found, for instance, on a smaller subset of observations. Another drawback of \(H\)-MLE is that the \(H\)-matrix approximation of $\Sigma$ and its inverse (we compute it via Cholesky factorisation) were recomputed entirely on every iteration for each new value of $\tau$ and $\sigma$. It would be a lot cheaper just to update already existing matrices by adding a new diagonal or by scaling them.

Results. Among all implemented ML methods (kNN, random forest, deep neural network), the best results (for given datasets) were obtained by the kNN method with three or seven neighbors depending on the dataset [5].

Trying different numbers of trees in the RF method, we defined that an ensemble of 120 regression trees is optimal. Further, we have designed the FCNN architecture with 7 hidden layers with 100 neurons in each layer. The number of training epochs is 500, and the batch size equals 10000. We used tanh activation function and Adam optimizer. The average computing time of kNN was 0.07 sec. RF required 12 sec., and FCNN finished after 173 sec.

The results computed with the \(H\)-MLE method were compared with the results obtained by the kNN method. For Test 2a, the \(H\)-MLE method showed a smaller RMSE error than the kNN method, whereas, for Test 2b, the kNN method was better. To conclude, it is not surprising that our \(H\)-MLE method worked fine on most datasets. What surprised us is that the well-known and straightforward kNN method performed very well and very fast. Since we did not make any theoretical comparison and compared \(H\)-MLE and ML methods only numerically on given datasets, we cannot, in general, conclude which method is better. We also remind that we used kNN only for the prediction, without the identification of unknown parameters.

More details about the \(H\)-MLE method are given in [2, 3], about numerical experiments and comparison in [1,4, 5].

Software: All \(H\)-MLE numerical results are reproducible https://github.com/litvinen/large_random_fields.git.

Acknowledgements. V. Berikov was supported by the Sobolev Institute of Mathematics (project no 0314-2019-0015) and by RFBR grant 19-29-01175. A. Litvinenko was supported by funding from the AvH Foundation. Open access funding enabled and organized by Projekt DEAL.

References

[1] A. Litvinenko, R. Kriemann, and V. Berikov, Proceedings UNCECOMP 2021 Conf., arXiv 2104.07146, (Greece (online), 2021).
[2] A. Litvinenko, Y. Sun, M. G. Genton, and D. E. Keyes, Computational Statistics & Data Analysis 137, 115–132 (2019).
[3] A. Litvinenko, R. Kriemann, M. G. Genton, Y. Sun, and D. E. Keyes, MethodsX 7, 100600 (2020).
[4] H. Huang, S. Abdulah, M. G. Genton, Y. Sun, H. Ltaief, and D. E. Keyes, 2021 KAUST Competition on Spatial Statistics for Large Datasets, 2020, https://cemse.kaust.edu.sa/stsds/2021-kaust-competition-spatial-statistics-large-datasets.
[5] H. Huang, S. Abdulah, Y. Sun, H. Ltaief, D. E. Keyes, and M. G. Genton, accepted to JABES (2021).