Abstract

The growth of big data in domains such as Earth Sciences, Social Networks, Physical Sciences, etc. has brought to an immense need for efficient and scalable linear algebraic operations. One of the examples is the creation of environmental and climate models that are used for weather prediction and require evenly spaced meteorological datasets at a very high spatial and temporal resolution to facilitate the analysis of recent climatic changes. However, due to the small number of weather stations available, often the data collected from them are scattered and inadequate for such model creation. For this reason a very high resolution gridded meteorological surface is developed by interpolating the available scattered data points to fulfill the need for various ecological and climatic applications. Among various interpolation techniques, Ordinary Kriging (OK) is one of the most popular and widely used gridding methodologies with a sound statistical basis providing a possibility to obtain highly accurate results. However, OK interpolation on large unevenly spaced data points is computationally demanding and has a computational cost that scales as the cube of the number of data points as it involves multiplication and inversion of matrices of large cardinalities infeasible for computation on a single node.

Meanwhile, Apache Spark has emerged as a large scale data processing engine with a fault-tolerant data-structure (Resilient Distributed Datasets (RDDs)) and a dedicated machine learning library (MLLib) for processing large matrices and therefore can be used for large scale kriging analysis with considerable time. In this thesis, we explore the distributed scalable block-recursive matrix computation approaches which are particularly suitable for data distributed parallel processing engines like Spark. We consider two important matrix operations — multiplication and inversion which are two essential operations in distributed OK implementation.

In the first work, we present a fast and highly scalable distributed matrix multiplication algorithm, called Stark, based on Strassen's matrix multiplication algorithm. Stark preserves Strassen's seven multiplication scheme in a distributed environment and thus achieves asymptotically faster execution time. It creates a distributed recursion tree of computation where each level of the tree corresponds to division and combination of distributed matrix blocks stored in the form of Resilient Distributed Datasets (RDDs). It processes each divide and combine step in parallel and memorizes the sub-matrices by intelligently tagging matrix blocks in it. We report a detailed complexity analysis for the proposed algorithm, taking into account computation and communication costs. Experimental results suggest that Stark outperforms existing distributed matrix multiplication implementations on Spark – Marlin and MLLib, for high matrix sizes (16384×16384). Our experiments reveal optimal block sizes for each matrix size, which is also shown in theoretical analysis. We also show that the experimental and theoretical running times for Stark match closely. It is also shown experimentally that Stark exhibits strong scalability with an increasing number of executors.

Existing methods for efficient and distributed matrix inversion using big data platforms rely on LU decomposition-based block-recursive algorithms. However, these algorithms are complex and require a lot of side calculations, e.g. matrix multiplication, at various levels of recursion. In the second work, we propose a different scheme based on Strassen's matrix inversion algorithm (mentioned in Strassen's original paper in 1969), which uses far fewer operations at each level of recursion. We implement the proposed algorithm, and through extensive experimentation, show that it is more efficient than the state of the art methods. Furthermore, we provide a detailed theoretical analysis of the proposed algorithm and derive theoretical running times that match closely with the empirically observed wall clock running times, thus explaining the U-shaped behavior w.r.t. Block-sizes.

Finally, we present a fast distributed Ordinary Kriging algorithm and provide an efficient and simple distributed matrix inversion scheme to accelerate the execution of a distributed OK algorithm. We employ Strassen's direct method for matrix inversion and the acceleration is achieved by exploiting the symmetry nature of the variance-covariance matrix of the OK equation to invert the matrix. We show experimentally that our distributed inversion scheme enables us to invert a 16000 × 16000 matrix with 51%, and 38% less wall-clock time than distributed Spark-based LU and Strassen's based inversion schemes.

Keywords: Numerical Linear Algebra, Ordinary Kriging, Distributed Matrix Computation, Strassen's Algorithm, Apache Spark.