Parallel Computing 000 (2016) 1–18

Contents lists available at ScienceDirect

Parallel Computing

journal homepage: www.elsevier.com/locate/parco

Parallel iterative refinement linear least squares solvers based on all-reduce operations

Karl E. Prikopa, Wilfried N. Gansterer*, Elias Wimmer

University of Vienna, Faculty of Computer Science, Vienna, Austria

ARTICLE INFO

Article history:
Received 6 November 2014
Revised 17 May 2016
Accepted 24 May 2016
Available online xxx

Keywords:
Parallel least squares solver
Semi-normal equations
Normal equations
Iterative refinement
Mixed precision
Tall and skinny matrices
All-reduce

ABSTRACT

We present the novel parallel linear least squares solvers ARPLS-IR and ARPLS-MPIR for dense overdetermined linear systems. All internode communication of our ARPLS solvers arises in the context of all-reduce operations across the parallel system and therefore they benefit directly from efficient implementations of such operations. Our approach is based on the semi-normal equations, which are in general not backward stable. However, the method is stabilised by using iterative refinement. We show that performing iterative refinement in mixed precision also increases the parallel performance of the algorithm. We consider different variants of the ARPLS algorithm depending on the conditioning of the problem and in this context also evaluate the method of normal equations with iterative refinement. For ill-conditioned systems, we demonstrate that the semi-normal equations with standard iterative refinement achieve the best accuracy compared to other parallel solvers.

We discuss the conceptual advantages of ARPLS-IR and ARPLS-MPIR over alternative parallel approaches based on QR factorisation or the normal equations. Moreover, we analytically compare the communication cost to an approach based on communication-avoiding QR factorisation. Numerical experiments on a high performance cluster illustrate speed-ups up to 3820 on 2048 cores for ill-conditioned tall and skinny matrices over state-of-the-art solvers from DPLASMA or ScaLAPACK.

1. Introduction

In scientific applications, a typical problem is fitting the parameters of a mathematical model to observations which are subject to errors. Performing linear regression analysis on these observations requires efficient linear least squares (LLS) solvers. We consider the problem of solving the dense LLS problem

$$\min_x \|b - Ax\|_2 \tag{1}$$

in parallel, where $A \in \mathbb{R}^{n\times m}$ with $n \geq m$ and $b \in \mathbb{R}^n$. Of special interest are strongly over-determined LLS problems where the matrix A has many more rows than columns ($n \gg m$), also known as *tall and skinny* LLS problems. Many big data applications naturally exhibit such a strongly rectangular structure, having billions of data points with only a few hundred descriptors. For example, monitoring seismological activity generates massive amount of data. In [1], a wireless sensor network with only three nodes was deployed around a volcano and returned millions of rows of sensed data. Tall and skinny problems

* Corresponding author.
E-mail addresses: karl.prikopa@univie.ac.at (K.E. Prikopa), wilfried.gansterer@univie.ac.at (W.N. Gansterer), elias.wimmer@univie.ac.at (E. Wimmer).

http://dx.doi.org/10.1016/j.parco.2016.05.014
0167-8191/

also arise in partial differential equations [2]. Another field of application where high dimensional regression is required is genetics [3]. Single nucleotide polymorphisms (SNPs) can help exhibit a human's susceptibility to different diseases. Millions of SNPs are known today, but the number of subjects for a study of a certain disease is often very low, often limited to a few thousand due to high costs.

We present and evaluate the novel *all-reduce parallel least squares* solvers ARPLS-IR and ARPLS-MPIR for solving problem (1) which are based on the method of semi-normal (SNE) or normal equations (NE) with (mixed precision) iterative refinement (IR). A and b are distributed row-wise across all N processes and the solution $x \in \mathbb{R}^m$ is replicated across the processes. All internode communication in the ARPLS algorithms is contained in all-to-all reduction operations across the participating processes. We consider different variants of the ARPLS algorithm depending on the conditioning of the problem. We show that the application of *mixed precision* iterative refinement (MPIR) in the context of parallel LLS solvers not only reduces the amount of computation but also the communication costs. To the best of our knowledge, the combination of MPIR with LLS solvers has not been studied so far. The mixed precision SNE approach is limited to systems with a condition number up to $\kappa(A) \approx 10^7$ due to single precision being used throughout the majority of the algorithm. In the case of NE, the mixed precision approach is further limited because the normal equations square the condition number of A. Therefore, mixed precision NE does not work for ill-conditioned systems. However, we demonstrate that the accuracy of the standard precision IR method is comparable to existing methods also for higher condition numbers of A. For ill-conditioned systems, unlike some other methods, the approach using SNE with IR (ARPLS-SNE-IR) can still solve the LLS problem and in all cases returns the highest accuracy among the compared algorithms. Moreover, we provide an analysis and comparison of the communication cost of different parallel LLS solvers. Like many standard parallel solvers for dense LLS problems, many ARPLS variants require a parallel QR factorisation algorithm. We thoroughly compare an all-reduce-based parallel version of modified Gram-Schmidt with Tall Skinny QR [4] in terms of computation and communication cost and show how to optimise the communication cost of the all-reduce-based QR factorisation.

The paper is organised as follows. Section 2 summarises related work on parallel and distributed LLS solvers. Section 3 describes the mathematical basis for our approach. Section 4 introduces and discusses different variants of the ARPLS method and a parallel QR factorisation method based on modified Gram-Schmidt. Section 5 provides an analysis of the communication cost and a comparison with an LLS solver based on the communication-avoiding QR (CAQR) algorithm [4] which achieves the theoretical minimum in terms of communication cost. In Section 6 we summarise numerical experiments conducted on a large scale cluster for comparing the performance of ARPLS methods and the LLS solvers from DPLASMA and ScaLAPACK. Section 7 concludes our paper.

2. Related work

Many parallel algorithms for solving LLS problems have been studied in the literature and are available in high-performance libraries like ScaLAPACK [5], aimed at distributed memory parallel computers, PLASMA [6], designed for shared-memory multi-core machines, MAGMA [6], considering heterogeneous and hybrid architectures with multi-core and GPU systems, or DPLASMA [7], which extends PLASMA to distributed heterogeneous systems. PLASMA and DPLASMA use specialised dynamic scheduling systems (QUARK and PaRSEC, respectively) based on building a direct acyclic graph of parallel tasks and considering the data dependencies of these tasks.

The basic building block of many LLS solvers is a QR factorisation. In PLASMA this is implemented using the tiled QR factorisation algorithm [8], which divides the matrix into small square tiles instead of using rectangular panels seen in block algorithms. The finer granularity achieved by the square tiles is better suited for multi-core architectures [9]. Demmel et al. [4] introduced communication-avoiding QR factorisation (CAQR) and proved that its communication cost is optimal up to poly-logarithmic factors. The CAQR algorithm factorises block-columns of A, called panels, in parallel using the TSQR algorithm, which is designed for tall and skinny matrices. The panels are divided into block-rows, called domains, which are factorised independently and then merged using a binary tree strategy. In [10], a systolic QR factorisation algorithm is implemented for a distributed memory machine using the PaRSEC parallel scheduler. The authors target a 3D torus topology and limit the communication of the algorithm to neighbouring nodes, aiming to minimise the amount of communication in the reduction trees.

An example for a parallel LLS solver is the parallel multi-splitting method by Renaut [11], which uses the well-known fixed-point iteration methods Jacobi, Gauss-Seidel and successive over-relaxation to solve the LLS problem by forming the normal equations. The matrix A is distributed column-wise over the network nodes and weighting matrices are used to recombine the local problems, which are independent problems resulting from the linear multi-splitting of A. In each iteration a vector of size n has to be broadcast to all other nodes in the network.

A slightly different emphasis is pursued in *distributed* LLS solvers, which limit their communication to the immediate neighbourhood and are therefore of particular interest for loosely coupled networks. Sayed et al. [12] have proposed a diffusion-based least mean square estimator (diffLMS) using normal equations and steepest-descent iterations. Diffusion strategies are seen as an alternative to consensus strategies for distributed optimisation problems, both aiming at limiting the communication to the neighbourhood. The data A and b are both distributed row-wise. The distributed least mean squares method (D-LMS) described in [13] also only uses neighbourhood communication. The method is based on Lagrange multipliers and uses the least squares residual and the difference between the estimates of x from the neighbourhood in a correction step to compute the LLS solution iteratively. The data distribution of A and b is again row-wise. At each step an

Please cite this article as: K.E. Prikopa et al., Parallel iterative refinement linear least squares solvers based on all-reduce operations, Parallel Computing (2016), http://dx.doi.org/10.1016/j.parco.2016.05.014