Iterative Randomized Algorithms for Low Rank Approximation of Tera-scale Matrices with Small Spectral Gaps

Chander Iyer*, Alex Gittens*, Christopher Carothers*, and Petros Drineas†
*Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180, USA
†Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA

***Abstract*—Randomized approaches for low rank matrix approximations have become popular in recent years and often offer significant advantages over classical algorithms because of their scalability and numerical robustness on distributed memory platforms. We present a distributed implementation of randomized block iterative methods to compute low rank matrix approximations for dense tera-scale matrices. We are particularly interested in the behavior of randomized block iterative methods on matrices with small spectral gaps. Our distributed implementation is based on four iterative algorithms: block subspace iteration, the block Lanczos method, the block Lanczos method with explicit restarts, and the thick-restarted block Lanczos method. We analyze the scalability and numerical stability of the four block iterative methods and demonstrate the performance of these methods for various choices of the spectral gap. Performance studies demonstrate superior runtimes of the block Lanczos algorithms over the subspace power iteration approach on (up to) 16,384 cores of AMOS, Rensselaer's IBM Blue Gene/Q supercomputer.**

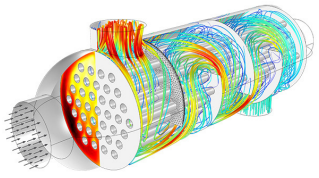
***Index Terms*—Block Lanczos methods, high performance computing, low-rank approximations, tera-scale matrices.**

I. Introduction

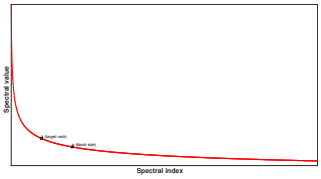
Low-rank matrix approximations are among the most popular tools in many machine learning [1], statistical [2] and scientific computing applications [3]. Given the explosive growth of data in multiple domains, computing the low rank approximation efficiently and accurately poses significant challenges for the analysis of such large scale data sets. There have been a number of approaches to compute the low rank approximation in classical numerical analysis, including the truncated singular value decomposition [4], the (column) pivoted QR factorization, and the rank-revealing QR factorization [5]. However, classical direct approaches are quite inefficient when computing low rank approximations of massive matrices due to the large number of data accesses between memory hierarchies; such issues make these algorithms prohibitively slow in distributed environments. Another challenge of such factorization techniques is due to the computational inefficiency of computing low rank approximations of matrices whose singular spectra decay slowly. Golub and Kahan [6] proposed to alleviate the spectral decay problem by computing powers of the matrix to iteratively form a Krylov subspace and then obtain a low rank approximation from this subspace.

Randomized algorithms are able to exploit modern architectures due to their inherent parallelism and often require fewer passes over the input matrix compared to traditional approaches. There has been a spate of work to approximately compute low rank approximation via randomized algorithms over the past 20 years. Halko, Martinsson and Tropp [7] gave a comprehensive overview of the historical context of randomized algorithms arising from several research disciplines and also covered related work in the numerical linear algebra literature. Their work was among the first to propose a randomized framework for computing low rank matrix approximations by projecting the matrix onto a low-dimensional subspace. Then, a low-rank approximation of the projected matrix is computed using iterative methods. Subsequently Halko et. al. [8] proposed a randomized iterative method using a block Lanczos approach. Another significant development involving randomized methods in a distributed environment was the advent of communication avoidance algorithms for SVD [9]. Recent work in block iterative methods for sparse matrices [10], [11] from real-world applications combined block Lanczos algorithms and subspace iteration with communication avoiding approaches.

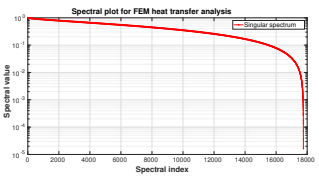
(a) Image dataset sample



(b) FEM analysis of heat transfer



(c) Rapid spectral decay plot of sample image



(d) Slow spectral decay plot for FEM analysis of heat transfer

Fig. 1. Spectral decay characteristics from two application domains: Images and Heat transfer

Recent analyses of randomized methods for low-rank approximations resulted in both gap-dependent and gap-independent error bounds [12] for the spectral norm approximation error; more recently, a per-vector approximation error was presented in [13]. These bounds are more useful for low-rank approximations of matrices with "low-rank plus shift" distributions of their singular values. Such matrices exhibit a considerable gap between the top singular values and the unwanted (shift) singular values as shown in Figures 1a and 1c. Several large-scale datasets from application domains ranging from image classification and high-particle physics to text analysis and astrophysics exhibit such spectral characteristics. However, other application domains (e.g., finite element analysis of heat transfer equations, circuit simulations, etc.) tend to generate matrices which have singular value distributions with small spectral gaps and extremely slow decay (see Figures 1b and 1d). Low-rank approximations for such matrices using randomized subspace iteration or block Lanczos methods tend to converge very slowly. This phenomenon is compounded by the presence of large singular value clusters. For such matrices, restarted block Lanczos methods have been designed and are the methods of choice in practice. However, no gap-dependent or gap-independent bounds exist for restarted block Lanczos methods for matrices with small spectral gaps. In our work, we experimentally demonstrate the efficiency of these restarted methods over standard block iterative methods.

a) Our contributions: Our goal in this paper is to address the challenges of building a framework for randomized, restarted, block iterative methods for computing low rank approximations of tera-scale sized matrices in the Blue Gene/Q environment. More specifically:

- We experimentally evaluate the impact of spectral gaps on the performance and the scalability of randomized, restarted block iterative methods. Our work is inspired by the recent analyses of Musco & Musco [13], Wang et. al. [14] and Drineas et. al. [15].
- We also wish to understand the impact of block sizes on the convergence of the various randomized iterative methods for varying spectral gaps.
- From a systems perspective, our randomized block iterative framework serves as a proxy for a truncated SVD implementation in Elemental [16], a state-of-the-art software for large-scale matrix computations.

II. Background

A. Notation

Let $\mathbf{A}, \mathbf{B}, \ldots$ denote matrices and $\mathbf{a}, \mathbf{b}, \ldots$ denote column vectors. Given a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, let $\|\mathbf{A}\|_2 = \max_{\mathbf{x} \neq 0} \frac{\|\mathbf{A}\mathbf{x}\|_2}{\|\mathbf{x}\|_2}$ be its spectral norm and $\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}^2|}$ be its Frobenius norm. Let the SVD of $\mathbf{A}$ be $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$ and let $\mathbf{U}_k$ and $\mathbf{V}_k$ be the top k left and right singular vectors respectively. Then the best rank-k approximation to $\mathbf{A}$ is given by $\mathbf{A}_k = \mathbf{U}_k \Sigma_k \mathbf{V}_k^T$, where Σ_k is a diagonal matrix containing the top k singular values of $\mathbf{A}$. Similarly, let a rank-k approximation to $\mathbf{A}$ computed by a randomized, approximate SVD algorithm be given by $\hat{\mathbf{A}_k} = \hat{\mathbf{U}}_k \hat{\Sigma}_k \hat{\mathbf{V}}_k^T$ where $\hat{\mathbf{U}}_k$, $\hat{\Sigma}_k$, and $\hat{\mathbf{V}}_k$ are approximations to the top k left singular vectors, the top k singular values, and the top k right singular vectors respectively. Let δ be a tolerance parameter indicating the error between successive iterations of randomized iterative block methods [1]; when the error drops below δ the method has converged. Let γ be the spectral gap difference between the k-th singular value σ_k and the $k+1$-st singular value σ_{k+1}, given by $\gamma = \frac{\sigma_k}{\sigma_{k+1}} - 1$. Finally, let q be the total number of iterations for the block iterative methods and let c be the number of restarts for the restarted block Lanczos methods. Also, let q' be the number of iterations before restarting the restarted methods; this number is equal to $q' = \lfloor q/c \rfloor$. Table I below summarizes the notations used in this paper; we note that parameters l (the oversampling parameter for matrix sketching) and p (the block size) will be discussed in Sections II-C0a and II-C0b respectively.

Notation	Description
$m \times n$	Dimensions of the input matrix
k	Target rank for the low rank approximation
q	Number of iterations of iterative methods
c	Number of restarts for the restarted block Lanczos methods
q'	Number of iterations before restart for restarted methods
l	Oversampling parameter for random projection
p	Block size for block methods
δ	Tolerance value guiding convergence
γ	Spectral gap

TABLE I
NOTATIONS USED IN THE PAPER.

B. Spectral gap assumptions in prior work

The past 20 years have seen a flurry of activity on randomized algorithms for low-rank approximation (see [8], [17], [18] for reviews). We focus our discussion here on randomized block iterative methods only. The earliest randomized block iterative method [19] computes an approximation to the top k left singular vectors of the input matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$. These approximate left singular vectors $\hat{\mathbf{U}}_k$ satisfy

$$\left\| \mathbf{A} - \hat{\mathbf{U}}_k \hat{\mathbf{U}}_k^T \mathbf{A} \right\|_2 \leq (1+\epsilon) \left\| \mathbf{A} - \mathbf{A}_k \right\|_2 .$$

To achieve the above error bound, $q = O\left(\frac{\log(n/\epsilon)}{\sqrt{\gamma}}\right)$ iterations are needed, where γ is the spectral gap as defined in the previous section. However, for a number of application domains involving learning and data analysis, $\gamma \gg \epsilon$ and typically a much looser bound is preferable. Subsequently, Woodruff [20] presented similar approximation guarantees for subspace iteration after $q = O(\log(n/\epsilon))$ iterations; this was the first gap-independent bound for randomized subspace iteration with respect to spectral norm error. Musco & Musco [13] provided the first approximation error bounds that were gap-independent in terms of the per-vector error

[1] We use the maximum residual norm error of the top-k approximate singular vectors defined subsequently as our error metric to compute δ.

using $q = O(\log n/\epsilon)$ iterations for subspace iteration and $O(\log n/\sqrt{\epsilon})$ iterations for block Krylov methods, whose the block size (denoted by p) is equal to the target rank k. Typically, in distributed environments, larger block sizes are preferred: larger block sizes limit the costs of I/O operations. [13] as well as [14] provided tighter gap-dependent bounds for block Krylov (and corresponding bounds for subspace iteration) after $q = O\left(\frac{\log(n/\epsilon)}{\sqrt{\gamma_p}}\right)$ iterations for block sizes $p \geq k$, where $\gamma_p = \frac{\sigma_k}{\sigma_{p+1}} - 1$ (an improved spectral gap between the k-th and the $(p+1)$-st singular value). Table II gives an overview for the various gap-dependent and gap-independent bounds of different randomized block iterative methods. However, as

Block iterative methods	**Number of iterations (gap-dependent)** ($\gamma_p \gg \epsilon$)	**Number of iterations (gap-independent)**
Subspace iteration	$\Theta\left(\frac{\log(n/\epsilon)}{\gamma_p}\right)$	$\Theta(\log n/\epsilon)$
Block Lanczos	$\Theta\left(\frac{\log(n/\epsilon)}{\sqrt{\gamma_p}}\right)$	$\Theta\left(\log n/\sqrt{\epsilon}\right)$
Restarted Block Lanczos	?	?

TABLE II
SPECTRAL GAP CONVERGENCE BOUNDS FOR BLOCK ITERATIVE METHODS.

mentioned previously, both subspace iteration methods as well as block Krylov methods converge slowly for singular value distributions that exhibit slow spectral decay or have singular values that are clustered together [21]. Another limitation of block Lanczos methods when computing a large number of target vectors is that the storage and computational costs increase as the number of iterations increase. Restarted block Lanczos methods serve as an alternative approach for matrices with such spectral characteristics. To the best of our knowledge, no gap-dependent or gap-independent analysis exists for any restarted block Lanczos method in the current literature. In our work here, we experimentally evaluate the behavior of restarted block Lanczos methods for matrices with small spectral gaps and contrast them with vanilla block Lanczos and subspace iteration methods.

For matrices with slow spectral decay, block iterative methods converge much faster (at least in practice) than the theoretical bounds mentioned above suggest. Further, as the number of iterations increases, the number of computations increases prohibitively. To avoid this, we introduce an early stopping criterion to check for convergence based on the residual norm error between two consecutive iterations given by

$$\delta = \left\{\max_{i|\forall i\in[1..k]} \left\|\mathbf{r}_i^{j+1}\right\|_2\right\} - \left\{\max_{i|\forall i\in[1..k]} \left\|\mathbf{r}_i^{j}\right\|_2\right\}$$

where

$$\|\mathbf{r}_i\|_2 = \left(\|\mathbf{A}\hat{\mathbf{v}}_i - \hat{\sigma}_i\hat{\mathbf{u}}_i\|_2^2 + \left\|\mathbf{A}^T\hat{\mathbf{u}}_i - \hat{\sigma}_i\hat{\mathbf{v}}_i\right\|_2^2\right)^{1/2}.$$

In words, the tolerance value δ is a tunable parameter that controls the tradeoff between the target accuracy and the number of iterations required.

C. Randomized Block Iterative Methods

We provide a brief overview of the randomized block iterative methods that approximate the top singular vectors and singular values of dense matrices; we focus on methods that we will evaluate in our work.

a) Randomized Subspace Iteration: This is the most commonly used approach and generates an orthonormal basis that approximates the top left/right singular vectors of a target matrix using power iterations [22]. The subspace iteration is initialized using a (low dimensional) sketch of $\mathbf{A}$ given by $\mathbf{A}\Pi$ where $\Pi \in \mathbb{R}^{n\times l}$ is a suitable random matrix. The subspace iteration generates the matrix $\mathbf{K} = (\mathbf{A}\mathbf{A}^T)^q\mathbf{A}\Pi$ at the end of q iterations, but intermediate matrices are orthonormalized at every iteration to improve numerical stability, especially for ill-conditioned matrices. Once an orthonormal basis $\mathbf{K}$ is generated, a low rank approximation is constructed by forming the matrix $\mathbf{T} = \mathbf{K}^T\mathbf{A}$, computing the SVD of $\mathbf{T}$, $\mathbf{T} = \tilde{\mathbf{V}}\tilde{\Sigma}\tilde{\mathbf{W}}^T$ and generating the top-k approximate left singular vectors $\hat{\mathbf{U}}$ given by $\hat{\mathbf{U}} = \mathbf{K}\tilde{\mathbf{V}}_k$.

b) Randomized Block Lanczos: The randomized block Lanczos algorithm is a block Krylov subspace approach to approximate a low-rank matrix decomposition of large-scale matrices, especially when the singular spectrum of the matrix has a heavy tail. For nonsymmetric matrices, the block Krylov subspace is given by

$$\mathbf{K} = [\mathbf{A}\Pi \;\; (\mathbf{A}\mathbf{A}^T)\mathbf{A}\Pi \;\; (\mathbf{A}\mathbf{A}^T)^2\mathbf{A}\Pi \;\; \ldots \;\; (\mathbf{A}\mathbf{A}^T)^q\mathbf{A}\Pi].$$

where $\Pi \in \mathbb{R}^{n\times p}$ is a random projection matrix that computes a (low-dimensional) sketch of $\mathbf{A}$. Here p is the block size of the Krylov subspace. Similar to the randomized subspace iteration approach, the block Krylov subspace $\mathbf{K}$ is orthonormalized after every iteration to avoid stability issues for ill-conditioned Krylov blocks. A low rank approximation is then constructed from the orthonormal basis $\mathbf{K}$ similar to subspace iteration.

c) Randomized Block Lanczos with explicit restart: One of the problems associated with the randomized block Lanczos approach is the computational cost involved in orthonormalizing $\mathbf{K}$ and computing the SVD of $\mathbf{T}$, along with the storage costs of $\mathbf{K}$ and $\mathbf{T}$ as the number of iterations q increase. Golub, Luk and Overton [23] proposed an approach to explicitly restart the block Lanczos iterations by setting the initial matrix Π to the left singular vectors computed at the end of q' iterations.

d) Bidiagonal Block Lanczos with thick-restart: A disadvantage of the explicit restart is that it discards the residual basis vectors after each restart. An effective way of restarting the bidiagonal block Lanczos method was proposed by Wu and Simon [24] and is called the *thick restarted* Lanczos algorithm. This algorithm keeps multiple approximate singular vectors (the so-called Ritz vectors) at restart. Subsequently, Baglama and Reichel [25] augmented the Ritz vectors associated with the largest Ritz values with the Lanczos block computed by the bidiagonalization algorithm to generate a low rank approximation. Typically, the number of augmented Ritz vectors is equal

to the target rank k that we seek to approximate. The residual vectors $\tilde{\mathbf{r}_j}^T$ in the bidiagonalization matrix at the end of each restart are given by $\tilde{\mathbf{r}_j}^T = \mathbf{u}_j{}^T \mathbf{A}\mathbf{P}_{q'+1}$ for $j = 1 \ldots k$. These residual vectors along with the top-k approximate singular values $\tilde{\sigma_i}$ for all $i = 1 \ldots k$ form a basis for the subsequent restart cycle.

III. Implementing our Algorithms on the Blue Gene/Q

Distribution formats for a 2-D process grid	$\sim$	$[M_C, M_R]$ & $[M_R, M_C]$ $[V_R, \star]$ & $[\star, V_R]$ $[V_C, \star]$ & $[\star, V_C]$ $[\star, \star]$
Distribution order within each grid dimension	M_C	Matrix column
	M_R	Matrix row
	V_C	Vector in column major order
	V_R	Vector in row major order
	$\star$	Stored on every process
Description	$[X, Y]$	Distribute [columns, rows] with scheme [X, Y]
	$[M_C, M_R]$	Distribute [columns, rows] equally among processes
	V_C/V_R	Distribute over processes in column/row major wrapping

TABLE III
Elemental data distribution overview.

The randomized block iterative methods were implemented on top of the Elemental library [16]. Given a distributed environment over p processes, any dense matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is partitioned in Elemental into rectangular grids whose sizes are $r \times c$ in a 2D cyclic distribution, such that $p = r \times c$ and both r and c are $O(\sqrt{p})$. We use the standard distribution $[M_C, M_R]$ listed in Table III for dense input matrices in order to take advantage of operations that are communication intensive. [16] has a comprehensive discussion and provides insights of Elemental notation, describing different data distributions and the communication costs involved in redistribution.

A number of design issues arise when implementing randomized, block iterative algorithms for tera-scale *dense* matrices. Several recent approaches [10] that advocate *communication-avoidance(CA)* are ineffective for partitioning and distributing dense matrices, as CA-based approaches are mostly useful when the sparsity structure can be efficiently exploited. This is not the case for our dense matrices. We address such design issues for our implementations below.

A. Orthogonalization

The first step in our block iterative algorithms is the choice of an initial column-orthogonal matrix Π that acts as a sketching matrix for the input matrix $\mathbf{A}$. There are several choices for Π: they are typically chosen to be matrices whose entries are independent random variables (say normal random variable, or Rademacher random variables, etc.). We refer the reader to Iyer et. al. [26] for a discussion of such choices in the context of large-scale implementations of randomized numerical linear algebra algorithms. In our work here, we set Π to be a gaussian random matrix, whose entries are independent Gaussian random variables of zero mean and variance $1/(n)$.

Another significant consideration for our implementation is the choice of orthogonalization algorithms for our block iterative methods. At each iteration, to avoid numerical instability and accuracy loss, we orthogonalize the Krylov subspace that has been formed thus far. We chose to employ the Householder QR (HouseQR) algorithm without any column pivoting for such orthogonalizations instead of a Cholesky based QR (CholQR) factorization. We observed that HouseQR was more numerically stable than CholQR, especially for ill-conditioned matrices.

Finally, we also explored one-sided reorthogonalization (when multiplying by $\mathbf{A}^T$) vs. full reorthogonalization (multiplying by both $\mathbf{A}$ and $\mathbf{A}^T$). We observed in our evaluations that HouseQR did a reasonable job of orthogonalizing basis vectors in each iteration and that one-sided reorthogonalization sufficed for all our evaluations.

IV. Evaluation

We evaluate our randomized block iterative methods on dense synthetic matrices with different spectral distributions and varying spectral gaps. Each synthetic matrix is generated in the form of $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$ where $\mathbf{U}$ and $\mathbf{V}$ are random orthogonal matrices generated by orthogonalizing the columns of a random Gaussian matrix. The matrix Σ represents the spectral characteristics of the matrix and its structure is shown in Figure 2. Let the initial singular value of synthetic matrix

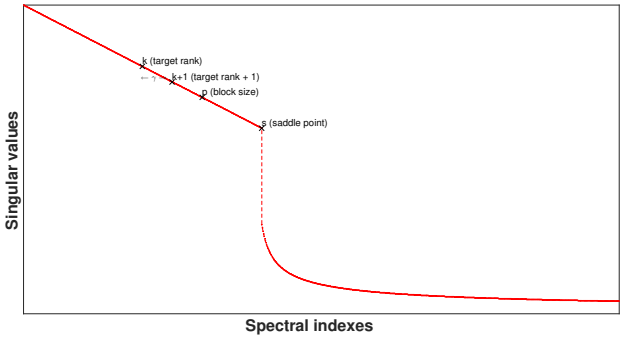


Fig. 2. Spectral characteristics of matrices with small spectral gaps

be denoted by σ_0. The remaining singular values of the matrix are then divided in two parts:

- the top singular values which exhibit a (uniform) linear decay up to a saddle point denoted by s; each of the top singular values is equal to $\sigma_i = \sigma_0 - i\gamma$ for $0 \leq i \leq s$.
- the bottom singular values which exhibit a nonlinear decay starting with the saddle point s all the way to the rank of the matrix which is equal to $\min\{m, n\}$. We focus on matrices with $m \geq n$ and we chose two forms of nonlinear decays for our evaluations: a power decay such that $\sigma_i = i^{-1}$ and an exponential decay such that $\sigma_i = 10^{-10i/n}$ for all $s < i \leq n$.

We formed a dense synthetic matrix $\mathbf{A} \in \mathbb{R}^{100,000 \times 30,000}$ and varied $s \in [600, 1800, 2400]$ and $\gamma \in [10^{-2}, 10^{-3}, 10^{-4}]$ in

Spectral characteristics	Spectral gap $\gamma = 10^{-2}$	Spectral gap $\gamma = 10^{-3}$	Spectral gap $\gamma = 10^{-4}$
	Power Spectral decay		
σ_0	25.0		
$\sigma_i \quad [0 \leq i \leq s]$ $\sigma_i \quad [s < i \leq n]$	$\sigma_0 - i\gamma$ i^{-1}		
σ_k	22.0	24.7	24.97
σ_{k+1}	21.99	24.699	24.9699
σ_p	19.0	24.4	24.94
σ_s	13.0	23.8	24.88
$\kappa(\mathbf{A})$	$.75 * 10^6$		

TABLE IV
SPECTRAL DISTRIBUTION CHARACTERISTICS OF Σ WITH POWER DECAY FOR THE DENSE SYNTHETIC MATRIX $\mathbf{A}$ WHERE $p = 600$ AND $s = 1, 200$.

Spectral characteristics	Spectral gap $\gamma = 10^{-2}$	Spectral gap $\gamma = 10^{-3}$	Spectral gap $\gamma = 10^{-4}$
	Exponential Spectral decay		
σ_0	25.0		
$\sigma_i \quad [0 \leq i \leq s]$ $\sigma_i \quad [s < i \leq n]$	$\sigma_0 - i\gamma$ $10^{-10i/n}$		
σ_k	22.0	24.7	24.97
σ_{k+1}	21.99	24.699	24.9699
σ_p	19.0	24.4	24.94
σ_s	13.0	23.8	24.88
$\kappa(\mathbf{A})$	$2.5 * 10^{11}$		

TABLE V
SPECTRAL DISTRIBUTION CHARACTERISTICS OF Σ WITH EXPONENTIAL DECAY FOR THE DENSE SYNTHETIC MATRIX $\mathbf{A}$ WHERE $p = 600$ AND $s = 1200$.

our evaluations. Further, we set the target rank k to 300, the tolerance value δ to 10^{-8} and the oversampling factor l to be equal to the block size p in our evaluations. We set the maximum number of iterations q to 50 for both the subspace iteration and the block Lanczos methods. For the restarted block Lanczos methods, we set the maximum number of iterations q to 5 and the maximum number of restarts c to 10. We outline the spectral characteristics of the matrix $\mathbf{A}$ for specific combinations of p and s in Tables IV and V.

We evaluated our randomized block iterative methods using the following two metrics. First, the maximum residual norm error difference:

$$max_{i|\forall i \in [1..k]} \left(\left\| \mathbf{A}\hat{\mathbf{v}}_i - \hat{\sigma}_i \hat{\mathbf{u}}_i \right\|_2^2 + \left\| \mathbf{A}^T \hat{\mathbf{u}}_i - \hat{\sigma}_i \hat{\mathbf{v}}_i \right\|_2^2 \right)^{1/2}, \tag{1}$$

and, second, the Frobenius norm error,

$$\frac{\|\mathbf{A}\|_F^2 - \left\| \hat{\mathbf{U}_k^T} \mathbf{A} \right\|_F^2}{\|\mathbf{A}\|_F^2}. \tag{2}$$

A. BG/Q Environment Setup

Our distributed implementation of the randomized block iterative methods run on AMOS [2], the high-performance Blue Gene/Q supercomputer system at RPI. AMOS supports a hybrid communication framework that uses the MPI (Message Passing Interface) [27] standard for distributed communication and multithreading using OpenMP [28] which our distributed implementation seeks to exploit. We implemented our distributed block iterative methods within the libskylark [3] framework that uses a LLVM/bgclang [4] build for our evaluations. As mentioned earlier, Elemental lacks a truncated SVD implementation. Existing truncated SVD implementations based on ARPACK++ failed to scale for our chosen experimental setup using the LLVM/bgclang build. Hence, our evaluations do not measure the performance of our randomized methods relative to a baseline truncated SVD implementation, since no such implementation exists for data of our size.

B. The effect of the spectral gap γ

Our goal in this work is to provide a thorough evaluation of our implementation of block iterative methods (with or without restarting) and the impact of the spectral gap γ on these methods. We seek to analyze the impact of γ from three primary viewpoints: scalability, performance, and numerical stability. We focus on dense random matrices generated with power decay spectral characteristics for our scalability and performance evaluations.

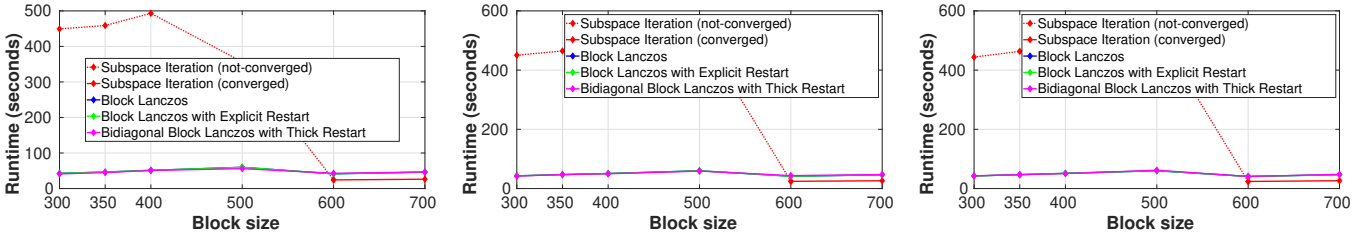
1) Scalability evaluation: A key goal in evaluating our distributed implementation of the various randomized block iterative methods is to analyze the effect of the block size on their convergence in the presence of decreasing spectral gaps. Figure 3 highlights the convergence behavior in terms of speedup for our distributed implementations with increasing values of s (the location of the saddle point) and decreasing values of the gap γ. As s increases, the number of singular values that are clustered increases; a decreasing value of γ also clusters the singular values together. Figures 3a, 3b and 3c show the impact of block sizes on the convergence of our block iterative methods with decreasing spectral gaps when the saddle point is set at $s = 600$. We see that in each of these figures, the subspace iteration approach fails to converge for increasing block sizes until the block size reaches the value $p = 600$. All three variants of block Lanczos converge quickly for all block sizes when $s = 600$. However, the speedups steadily decrease for increasing block sizes up to $p = 600$. When $p > 600$, this results in $\sigma_k > (1 + \epsilon)\sigma_{p+1}$ (a larger gap), which leads to improved convergence. We also observe that the bidiagonal block Lanczos approach with thick restart outperforms the other block Lanczos variants at $s = 600$ regardless of the spectral gap.

As the location of the saddle point s increases, the speedups for the block methods decrease continuously. The subspace iteration fails to converge for all block sizes when the saddle point s exceeds 600. For smaller block sizes, the block Lanczos methods fail to converge as $\sigma_k < (1 + \epsilon)\sigma_{p+1}$. This is especially true for the restart-based methods. However, as the block sizes increase, they satisfy the gap dependent convergence criteria of [13] (Theorem 13) achieving significant speedups. Eventually, the restarted methods dominate

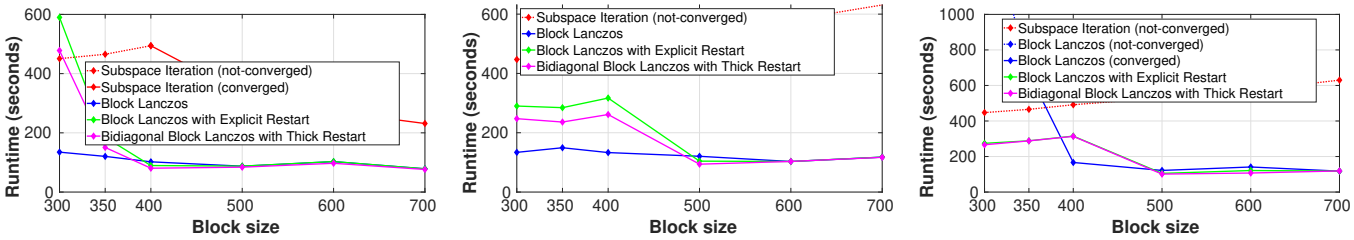
[2] https://secure.cci.rpi.edu/wiki/index.php/Blue_Gene/Q

[3] https://xdata-skylark.github.io/libskylark/

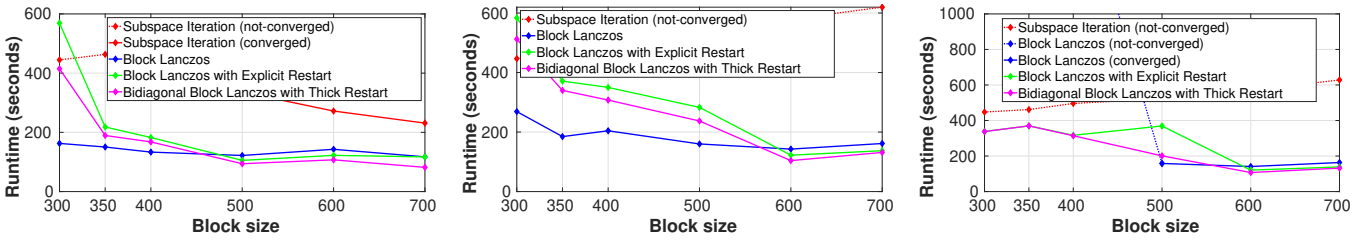
[4] https://trac.alcf.anl.gov/projects/llvm-bgq



(a) Spectral gap 10^{-2} with saddle point at 600 (b) Spectral gap 10^{-3} with saddle point at 600 (c) Spectral gap 10^{-4} with saddle point at 600



(d) Spectral gap 10^{-2} with saddle point at 1800 (e) Spectral gap 10^{-3} with saddle point at 1800 (f) Spectral gap 10^{-4} with saddle point at 1800



(g) Spectral gap 10^{-2} with saddle point at 2400 (h) Spectral gap 10^{-3} with saddle point at 2400 (i) Spectral gap 10^{-4} with saddle point at 2400

Fig. 3. Analysis of runtime (seconds) as a function of the block size p for varying spectral gaps γ and saddle points s for 128 BG/Q nodes.

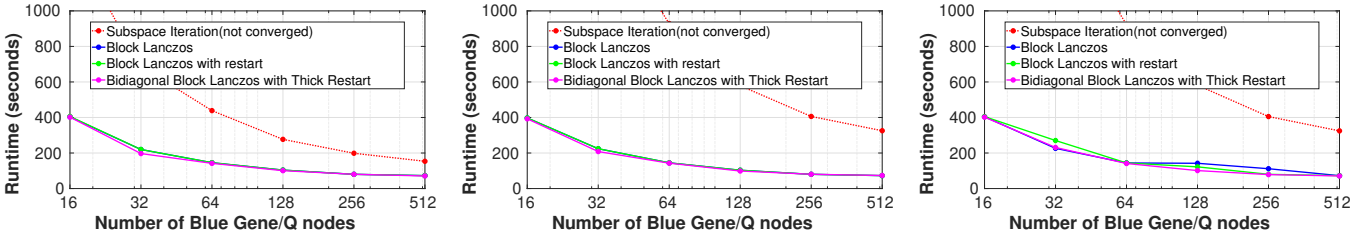
the standard block Lanczos implementation at larger block sizes when the saddle point index increases. This can be observed by comparing the speedups across the different block Lanczos methods between Figures 3d, 3e, and 3f. The standard block Lanczos method fails to converge for small block sizes when the spectral gap is the set to the smallest value, namely $\gamma = 10^{-4}$. Another significant observation is that the maximum speedup values keep decreasing as the spectral gap increases. Finally, at $s = 2,400$, the vanilla block Lanczos method initially dominates the restarted methods for smaller block sizes. However, as the block size increases, the restarted methods outperform the block Lanczos method.

To summarize: **(i)** the number of singular values in a cluster increases for as the location of the saddle point increasing; decreasing spectral gaps also cluster the singular values together; **(ii)** the subspace iteration method fails to converge for all block sizes p where the index of the saddle point s exceeds p; **(iii)** the block Lanczos methods fail to converge for small block sizes as s increases since $\sigma_k < (1+\epsilon)\sigma_{p+1}$. **(iv)** the bidiagonal block Lanczos with thick-restart converges faster than the other block Lanczos methods for small spectral gaps, especially when the block size increases; and **(v)** the maximum speedup achieved for all block Lanczos approaches decreases continuously for decreasing spectral gaps.

2) Performance evaluation: We evaluated the strong and weak scaling performance of our distributed block iterative implementations as a function of the (increasing) number of Blue Gene/Q (BG/Q) nodes. We chose to fix the block size p to 600 and we set the index of the saddle point s to 1800 in the evaluations of this section. We chose $\mathbf{A} \in \mathbb{R}^{100,000 \times 30,000}$ as our base matrix for the strong scaling experiments across all BG/Q nodes. For our weak scaling experiments, we chose $\mathbf{A} \in \mathbb{R}^{100,000 \times 30,000}$ as the base matrix when 16 nodes were used and we doubled the number of rows as we increased the number of BG/Q nodes.

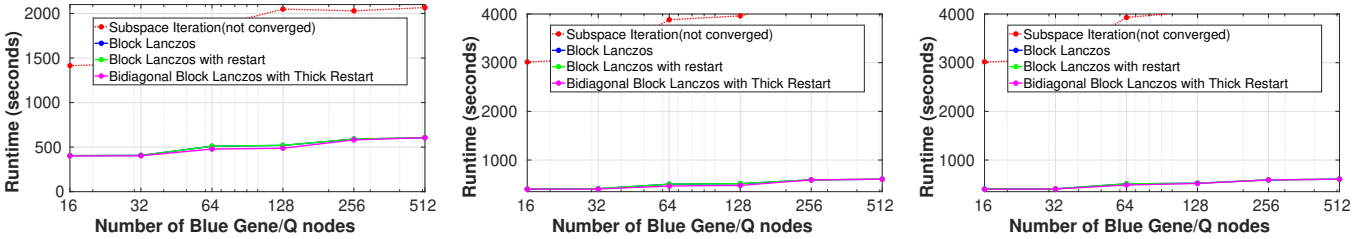
a) Strong scaling: Figures 4a, 4b, and 4c demonstrate the strong scaling performance for decreasing spectral gap values. In all scenarios, the subspace iteration method performs significantly worse than the block Lanczos implementations. All the block Lanczos variants perform almost equally, with the bidiagonal block Lanzos with thick-restart demonstrating the best performance. This is because the restarted Lanczos variants converge even before the first restart occurs (note that the restart occurs after every 5 restart blocks i.e. after every 5 iterations). Secondly, the strong scaling performance of the block Lanczos variants suffers when spectral gaps decrease.

b) Weak scaling: The weak scaling performance in terms of runtime is shown in Figures 5a, 5b, and 5c for decreasing spectral gaps.The running time for all randomized block iter-



(a) Spectral gap 10^{-2} (b) Spectral gap 10^{-3} (c) Spectral gap 10^{-4}

Fig. 4. Strong scaling in terms of runtime (seconds) as a function of increasing Blue Gene/Q nodes for different spectral gap values



(a) Spectral gap 10^{-2} (b) Spectral gap 10^{-3} (c) Spectral gap 10^{-4}

Fig. 5. Weak scaling in terms of runtime (seconds) as a function of increasing Blue Gene/Q nodes for different spectral gap values

ative methods, including subspace iteration, is fairly constant for increasing matrix sizes as the number of BG/Q nodes increases proportionately. The subspace iteration method does not converge for all spectral gaps when the saddle point index s is set to $1,800$ and hence exhibits significantly larger running times when compared to the block Lanczos variants.

3) Numerical stability evaluation: We evaluated our randomized block iterative methods to demonstrate the impact of spectral gaps on convergence as captured by the per vector error (eqn. 1) and the Frobenius norm error (eqn. 2). As mentioned earlier in Section IV-B2, we chose block size $p = 600$ and set the index of the saddle point s to 1800 to generate the random matrix $\mathbf{A}$ for our numerical stability evaluations. We evaluated numerical stability for $\mathbf{A} \in \mathbb{R}^{100,000 \times 30,000}$, generated using power as well as exponential power decays for singular values below the saddle point. Figures 6a, 6b, and 6c show that the block Lanczos approaches converge quickly as compared to the subspace iteration approach when the spectrum of the matrix decays following a power law. The subspace iteration method fails to converge for spectral gap values equal to 10^{-3} and 10^{-4}. Also, the restarted block Lanczos methods converge somewhat faster than the vanilla block Lanczos method for $\gamma = 10^{-4}$. This highlights the efficiency of the restarted block Lanczos methods in the presence of large clusters of singular values with small spectral gaps. Similarly, Figures 7a, 7b, and 7c show rapid convergence of the block Lanczos approaches as compared to the subspace iteration approaches for the exponential spectral decay.

V. Conclusions and Future Work

In this paper we implemented and evaluated highly scalable randomized block iterative methods that approximate, in a scalable manner, low rank approximations in the presence of clustered spectra with very small gaps. The methods are based on the block Lanczos algorithm and they outperform the block subspace iteration implementation in terms of running time performance while exhibiting high degrees of accuracy.

A topic of interest for future research would be to establish gap-independent bounds for approximation errors of restarted block Lanczos methods. Another research topic would be to explore the performance of our block methods on other multicore architectures, based on, say, GPUs.

Acknowledgments

We would like to thank Jack Poulson for numerous helpful discussions and his constant and timely support with the Elemental library.

References

[1] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, vol. 41, no. 6, pp. 391–407, 1990.

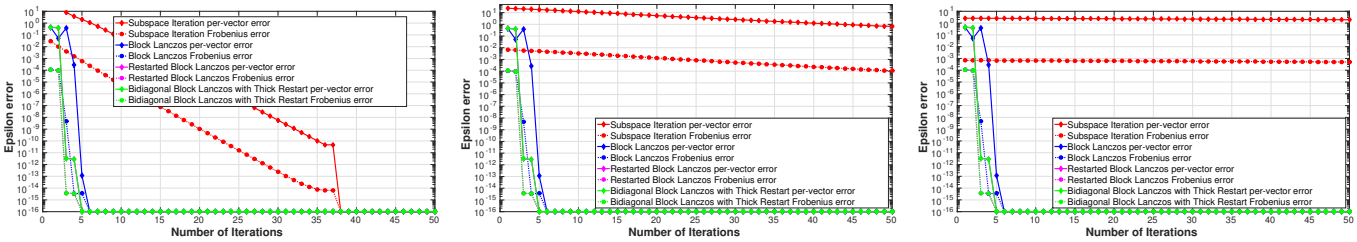
[2] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, ser. Algorithms and architectures for advanced scientific computing. Manchester University Press, 1992. [Online]. Available: https://books.google.com/books?id=FAkNAQAAIAAJ

[3] H. Cheng, Z. Gimbutas, P. G. Martinsson, and V. Rokhlin, "On the compression of low rank matrices," *SIAM J. Sci. Comput.*, vol. 26, no. 4, pp. 1389–1404, apr 2005. [Online]. Available: http://dx.doi.org/10.1137/030602678

[4] G. H. Golub and C. F. Van Loan, *Matrix Computations (3rd Ed.)*. Baltimore, MD, USA: Johns Hopkins University Press, 1996.

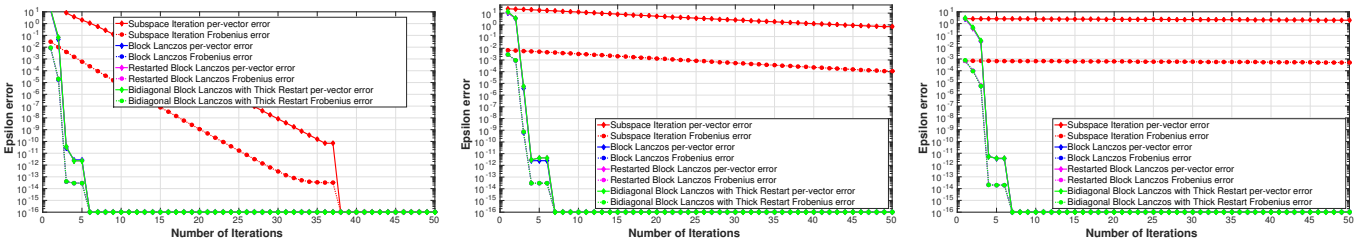
[5] T. F. Chan and P. C. Hansen, "Some applications of the rank revealing qr factorization," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, no. 3, pp. 727–741, 1992. [Online]. Available: http://dx.doi.org/10.1137/0913043

[6] G. Golub and W. Kahan, "Calculating the singular values and pseudo-inverse of a matrix," *Journal of the Society for Industrial and Applied Mathematics Series B Numerical Analysis*, vol. 2, no. 2, pp. 205–224, 1965. [Online]. Available: http://dx.doi.org/10.1137/0702016



(a) Spectral gap 10^{-2} (b) Spectral gap 10^{-3} (c) Spectral gap 10^{-4}

Fig. 6. Numerical stability analysis for the power spectral decay as a function of the number of iterations for different spectral gap values



(a) Spectral gap 10^{-2} (b) Spectral gap 10^{-3} (c) Spectral gap 10^{-4}

Fig. 7. Numerical stability analysis for the exponential spectral decay as a function of number of iterations for different spectral gap values

[7] N. Halko, P. G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, May 2011. [Online]. Available: http://dx.doi.org/10.1137/090771806

[8] N. Halko, P.-G. Martinsson, Y. Shkolnisky, and M. Tygert, "An algorithm for the principal component analysis of large data sets," *SIAM J. Sci. Comput.*, vol. 33, no. 5, pp. 2580–2594, Oct. 2011. [Online]. Available: http://dx.doi.org/10.1137/100804139

[9] G. Ballard, J. Demmel, and I. Dumitriu, "Minimizing communication for eigenproblems and the singular value decomposition," *CoRR*, vol. abs/1011.3077, 2010. [Online]. Available: http://arxiv.org/abs/1011.3077

[10] I. Yamazaki, T. Mary, J. Kurzak, S. Tomov, and J. Dongarra, "Access-averse framework for computing low-rank matrix approximations," in *Big Data (Big Data), 2014 IEEE International Conference on*, Oct 2014, pp. 70–77.

[11] I. Yamazaki, J. Kurzak, P. Luszczek, and J. Dongarra, "Random sampling to update partial singular value decomposition on a hybrid CPU/GPU cluster," in *International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM/IEEE, 2015.

[12] A. Szlam, Y. Kluger, and M. Tygert, "An implementation of a randomized algorithm for principal component analysis," *ArXiv e-prints*, Dec. 2014.

[13] C. Musco and C. Musco, "Stronger approximate singular value decomposition via the block lanczos and power methods," *CoRR*, vol. abs/1504.05477, 2015. [Online]. Available: http://arxiv.org/abs/1504.05477

[14] S. Wang, Z. Zhang, and T. Zhang, "Improved analyses of the randomized power method and block lanczos method," *CoRR*, vol. abs/1508.06429, 2015. [Online]. Available: http://arxiv.org/abs/1508.06429

[15] P. Drineas, I. Ipsen, E. Kontopoulou, and M. Magdon-Ismail, "Structural convergence results for low-rank approximations from block krylov spaces," *CoRR*, vol. abs/1609.00671, 2016. [Online]. Available: http://arxiv.org/abs/1609.00671

[16] J. Poulson, B. Marker, R. A. van de Geijn, J. R. Hammond, and N. A. Romero, "Elemental: A New Framework for Distributed Memory Dense Matrix Computations," *ACM Trans. Math. Softw.*, vol. 39, no. 2, pp. 13:1–13:24, Feb. 2013. [Online]. Available: http://doi.acm.org/10.1145/2427023.2427030

[17] P. Drineas, R. Kannan, and M. W. Mahoney, "Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix," *SIAM J. Comput.*, vol. 36, no. 1, pp. 158–183, Jul. 2006. [Online]. Available: http://dx.doi.org/10.1137/S0097539704442696

[18] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, "Relative-error CUR matrix decompositions," *SIAM J. Matrix Analysis Applications*, vol. 30, no. 2, pp. 844–881, 2008. [Online]. Available: http://dx.doi.org/10.1137/07070471X

[19] V. Rokhlin, A. Szlam, and M. Tygert, "A randomized algorithm for principal component analysis," *SIAM J. Matrix Anal. Appl.*, vol. 31, no. 3, pp. 1100–1124, Aug. 2009. [Online]. Available: http://dx.doi.org/10.1137/080736417

[20] D. P. Woodruff, "Sketching as a tool for numerical linear algebra," *Found. Trends Theor. Comput. Sci.*, vol. 10, no. 1–2, pp. 1–157, Oct. 2014. [Online]. Available: http://dx.doi.org/10.1561/0400000060

[21] R.-C. Li and L.-H. Zhang, "Convergence of the block lanczos method for eigenvalue clusters," *Numer. Math.*, vol. 131, no. 1, pp. 83–113, Sep. 2015. [Online]. Available: http://dx.doi.org/10.1007/s00211-014-0681-6

[22] H. Rutishauser, "Simultaneous iteration method for symmetric matrices," *Numerische Mathematik*, vol. 16, no. 3, pp. 205–223. [Online]. Available: http://dx.doi.org/10.1007/BF02219773

[23] G. H. Golub, F. T. Luk, and M. L. Overton, "A block lanczos method for computing the singular values and corresponding singular vectors of a matrix," *ACM Trans. Math. Softw.*, vol. 7, no. 2, pp. 149–169, Jun. 1981. [Online]. Available: http://doi.acm.org/10.1145/355945.355946

[24] K. Wu and H. Simon, "Thick-restart lanczos method for large symmetric eigenvalue problems," *SIAM J. Matrix Anal. Appl.*, vol. 22, no. 2, pp. 602–616, May 2000. [Online]. Available: http://dx.doi.org/10.1137/S0895479898334605

[25] J. Baglama and L. Reichel, "Augmented implicitly restarted lanczos bidiagonalization methods," *SIAM Journal on Scientific Computing*, vol. 27, no. 1, pp. 19–42, 2005. [Online]. Available: http://dx.doi.org/10.1137/04060593X

[26] C. Iyer, H. Avron, G. Kollias, Y. Ineichen, C. Carothers, and P. Drineas, "A randomized least squares solver for terabyte-sized dense overdetermined systems," *Journal of Computational Science*, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877750316301508

[27] M. P. Forum, "MPI: A message-passing interface standard," Knoxville, TN, USA, Tech. Rep., 1994.

[28] L. Dagum and R. Menon, "OpenMP: An industry-standard API for shared-memory programming," *IEEE Comput. Sci. Eng.*, vol. 5, no. 1, pp. 46–55, Jan. 1998. [Online]. Available: http://dx.doi.org/10.1109/99.660313