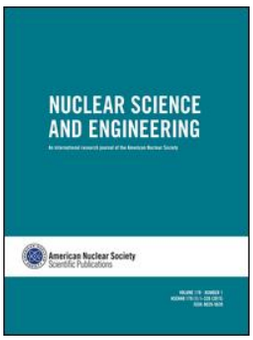

Nuclear Science and Engineering

ISSN: 0029-5639 (Print) 1943-748X (Online) Journal homepage: http://www.tandfonline.com/loi/unse20

A Cellwise Block-Gauss-Seidel Iterative Method for Multigroup S_N Transport on a Hybrid Parallel Computer Architecture

Massimiliano Rosa, James S. Warsa & Michael Perks

To cite this article: Massimiliano Rosa, James S. Warsa & Michael Perks (2013) A Cellwise Block-Gauss-Seidel Iterative Method for Multigroup S_N Transport on a Hybrid Parallel Computer Architecture, Nuclear Science and Engineering, 174:3, 209-226, DOI: 10.13182/NSE12-57

To link to this article: https://doi.org/10.13182/NSE12-57

Published online: 12 May 2017.

NUCLEAR SCIENCE AND ENGINEERING: **174**, 209–226 (2013)

A Cellwise Block–Gauss-Seidel Iterative Method for Multigroup S_N Transport on a Hybrid Parallel Computer Architecture

Massimiliano Rosa* and James S. Warsa

Los Alamos National Laboratory
Computational Physics and Methods Group
Los Alamos, New Mexico 87545

and

Michael Perks

IBM Corporation, Austin, Texas 78758

Received July 19, 2012
Accepted November 6, 2012

Abstract—*A Fourier analysis is conducted in two-dimensional (2-D) geometry for the discrete ordinates (S_N) approximation of the neutron transport problem solved with Richardson iteration (source iteration) using the cellwise block-Jacobi (bJ) and block–Gauss-Seidel (bGS) algorithms. The results of the Fourier analysis show that convergence of bJ can degrade, leading to a spectral radius ρ equal to 1, in problems containing optically thin cells. For problems containing cells that are optically thick, instead, ρ tends to 0. Hence, in the optically-thick-cell regime, bJ is rapidly convergent even for scattering-dominated problems. Similar conclusions hold for bGS, except bGS approaches the asymptotic, thick-cell regime at convergence rates higher than bJ. Hence, we have implemented the bGS algorithm on the Roadrunner hybrid, parallel computer architecture. A compute node of this massively parallel machine comprises AMD Opteron cores that are linked to a Cell Broadband Engine (Cell/B.E.). LAPACK routines have been ported to the Cell/B.E. in order to make use of its parallel synergistic processing elements (SPEs). The bGS algorithm is based on the LU factorization and solution of a linear system that couples the fluxes for all S_N angles and energy groups on a mesh cell. For every cell of a mesh that has been parallel decomposed on the higher-level Opteron processors, a linear system is transferred to the Cell/B.E. and the parallel LAPACK routines are used to compute a solution, which is then transferred back to the Opteron, where the rest of the S_N transport computations take place. Compared to standard parallel machines, a one-hundred-fold speedup of the bGS was observed on Roadrunner. Numerical experiments with strong and weak parallel scaling demonstrate that the bGS method is viable and compares favorably to full parallel transport sweeps (FPS) on 2-D unstructured meshes when it is applied to optically thick, multimaterial problems. Specifically, the strong parallel efficiency of accelerated bGS on Roadrunner can achieve 73% at 512 processors, compared with 32 processors, while efficiency is 34% for the (Opteron-only) implementation of FPS. The weak parallel efficiency of bGS is 58% while it reaches 10% for FPS. As expected, however, bGS is not as efficient as FPS in optically thin problems.*

I. INTRODUCTION

Computational methods for particle transport are very demanding, consuming the bulk of computational resources in a range of multiphysics simulations, including astrophysics calculations and nuclear reactor design. Current transport methods have evolved incrementally in response to advances in hardware technology. This picture, though, may soon change dramatically. It is understood by the scientific computing community that parallel, advanced hybrid computing architectures may become the machines of choice for large-scale

*E-mail: maxrosa@lanl.gov

computing in the future. The new machines have different processors and communication layers at different levels in their architecture, with different processing characteristics and capabilities at each level. Hybrid computing platforms, like the Roadrunner[1] class of hybrid parallel computers, several of which have been assembled at Los Alamos National Laboratory[2] (LANL), require the development of new transport methods, or reconsideration of previously discarded methods, in order to make full use of their capabilities.

In this paper we conduct a study of the stability and convergence properties of the cellwise block-Jacobi (bJ) and block–Gauss-Seidel (bGS) algorithms. Fourier analysis is traditionally used to study transport iteration schemes in a homogeneous infinite medium. In fact, it is a valuable tool to understand the behavior of the iteration error modes of various iterative schemes. Therefore, we conduct a Fourier analysis for the S_N approximation of the steady-state one-group transport problem solved with Richardson iteration using cellwise bJ and bGS. The spatial discretization is a discontinuous finite element method[3] (DFEM), specialized to triangular cells, and the scattering is assumed to be isotropic. The analysis is verified with results from a two-dimensional (2-D) transport code that implements cellwise bJ and bGS on parallel computing platforms.

One can view the cellwise bJ and bGS schemes as an evolution of the inexact parallel bJ (IPBJ) algorithm,[4] in which the IPBJ operator-split approach is applied to each mesh cell on a parallel subdomain. A review of the analogies and differences between bJ and IPBJ can be found in Table I of Ref. 5. Similar to IPBJ, the degradation in the spectral properties of bJ and bGS in the optically-thin-cell limit is a consequence of the "lagging" of the incoming angular fluxes at a mesh cell's boundaries, as the cells are made thinner and become more "strictly coupled."[6] In the opposite limit of optically thick cells, bJ and bGS display superior spectral properties with respect to IPBJ. In fact, while IPBJ's spectral radius approaches the scattering ratio c of the medium considered in the Fourier analysis, both bJ and bGS are characterized by a vanishing spectral radius independent of c, even for scattering-dominated problems. As we will show in this paper, the latter behavior is a consequence of the different splitting philosophy behind bJ and bGS, in which the contribution from the scattering source on a mesh cell is not lagged, as opposed to the "inexact" splitting characteristic of IPBJ where the scattering contribution is also lagged.

It is well known that for transport problems containing "diffusive" spatial regions that are optically thick and scattering dominated the traditional source iteration (SI) algorithm converges slowly.[7] Since many important applied problems do contain diffusive regions, it has long been desired to accelerate the iterative convergence of SI. The cellwise bJ and bGS algorithms can overcome such limitations at the higher computational cost resulting from their numerical implementation. While such elevated computational cost made bJ and bGS unappealing for implementation on traditional parallel computing architectures, the arithmetic-intensive bJ and bGS appear suitable algorithms for S_N transport computations on the parallel, advanced hybrid computing architectures that are gradually becoming available to the computational transport community.

In view of its superior spectral properties, we have implemented the multigroup bGS iteration on the Roadrunner machines at LANL (Ref. 8) with the expectation that the combination of fast factorization algorithms available on the Cell Broadband Engine™ (Cell/B.E.)[a] and rapid convergence rate associated with the cellwise bGS algorithm will make the bGS iteration competitive with the full parallel transport sweeps (FPS) that is typically used in parallel S_N transport applications.[9,10] Specifically, we investigate the range of problems for which the cellwise bGS algorithm is more efficient than FPS on unstructured meshes.

The emphasis on the bGS iteration, as opposed to traditional FPS, sets this work apart from other deterministic transport efforts within the emerging and rapidly developing interdisciplinary area of computational codesign, focused on the simultaneous consideration and design of algorithms, or mathematical methods, and computing technology. Other recent efforts have been dedicated to speeding up the execution time of the Koch-Baker-Alcouffe parallel S_N sweep algorithm for three-dimensional Cartesian meshes[11,12] either on Roadrunner[13] or on hybrid architectures where the accelerators are general-purpose graphics processing units[14] (GPGPUs). Finally, the use of GPGPUs to speed up S_N sweeps on unstructured grids is investigated in Ref. 15.

The main contributions of this paper are the study of the spectral properties of bJ and bGS and the numerical implementation of bGS on Roadrunner. Specifically, the Fourier analysis of the cellwise algorithms is a novel contribution and was conducted in order to ascertain their stability and to predict theoretically the convergence behavior, as a function of the optical thickness of the computational cells, observed from their numerical implementation. The spectral study described in the first half of the paper was also instrumental in identifying the class of optically thick problems that are used, in the second half, to showcase the superior strong and weak scaling properties of bGS compared to FPS on unstructured meshes. Finally, to the authors' knowledge, the implementation on the Cell/B.E. architecture is an original contribution toward adapting the multigroup bGS iteration on advanced hybrid computing architectures.

The remainder of the paper is organized as follows. The Fourier analysis of one-group cellwise bJ is

[a]Cell Broadband Engine™ and Cell/B.E. are trademarks of Sony Computer Entertainment, Inc.

performed in Sec. II, while the spectral properties of one-group cellwise bGS are investigated in Sec. III. In Sec. IV we discuss how bGS is applied to the multigroup S_N equations. Single- and dual-threaded strategies for the numerical implementation of multigroup bGS on Roadrunner are illustrated in Sec. V. Strong and weak scaling results are discussed in Sec. VI. Finally, in Sec. VII we present a summary of our main findings and conclusions.

II. FOURIER ANALYSIS OF ONE-GROUP CELLWISE bJ

The cellwise bJ algorithm for the steady-state, one-group, nonmultiplying transport equation with isotropic scattering is based on an operator splitting of

$$(L - SD)\psi = q \ , \tag{1}$$

where

L = streaming-plus-total interaction operator

S = scattering operator

D = discrete-to-moment operator (which represents integration over all S_N angles)

ψ = angular flux

q = fixed source.

The cellwise bJ splitting of the L operator is $L = L_c + L_b$, where L_c acts on the angular fluxes within a mesh cell and L_b acts on all the angular fluxes whose directions are oriented such that they are incoming relative to the faces of neighboring mesh cells. Within this splitting, Eq. (1) is written in the form

$$[I + (L_c - SD)^{-1}L_b]\psi = (L_c - SD)^{-1}q \ , \tag{2}$$

where I is the identity operator. This corresponds to a linear system for all S_N angles and all spatial degrees of freedom (we use a DFEM on triangles). Equation (2) is solved iteratively with restarted generalized minimum residual[16] (GMRES) such that, for each iteration, the action of the $(L_c - SD)^{-1}$ operator is computed using a LU decomposition for any mesh cell on a parallel subdomain. Specifically, all angular fluxes for all the spatial degrees of freedom on a given mesh cell are computed simultaneously, one mesh cell at a time. The ordering in which this "bJ sweep" takes place depends only on the numbering of the mesh. In the cellwise bJ algorithm, the term involving L_b is constructed for a given mesh cell using angular fluxes that are lagged from the previous iteration:

$$\psi^{(\ell+1)} = (L_c - SD)^{-1}(-L_b\psi^{(\ell)} + q) \ , \tag{3}$$

where $(\ell) > 0$ is the iteration index. Hence, not only are the angular fluxes lagged at the interface between two adjacent parallel subdomains, as for IPBJ (Ref. 4), but they are also lagged at the boundaries that the mesh cells share in the interior of a subdomain. The convergence rate of bJ is the spectral radius of the operator:

$$T_{\text{bJ}} = -(L_c - SD)^{-1}L_b \ . \tag{4}$$

To perform Fourier analysis of cellwise bJ, the equations representing the fixed-source-free 2-D DFEM spatial discretization of the S_N approximation to Eq. (3) are written for the system of four triangular cells over a Cartesian element sketched in Fig. 1. The equations for the projections of the discrete ordinates angular fluxes onto the linear basis functions of the finite element method, in the four neighboring cells, are grouped together for the discrete ordinates in the four quadrants, respectively. An infinite homogeneous medium, in which the four-cell system is periodically repeated, is considered in the Fourier analysis. Therefore, a suitable ansatz must be introduced at the boundaries of the Cartesian element. To fix ideas, the following Fourier ansatz is formulated for the discrete ordinates with cosines $\mu_m < 0$ and $\eta_m > 0$. Analogous expressions are introduced for the discrete ordinates in the remaining quadrants.

Fourier ansatz: ($\mu_m < 0$, $\eta_m > 0$)

$$\psi_{1,m}^{i(1)} = \psi_{3,m}^{k(\ell)}\exp(-j\lambda_y\sigma\mathbf{dy}) \ ,$$
$$i = 1,2; k = 2,1 \tag{5}$$

and

$$\psi_{2,m}^{i(1)} = \psi_{4,m}^{k(\ell)}\exp(+j\lambda_x\sigma\mathbf{dx}) \ ,$$
$$i = 1,2; k = 2,1 \ , \tag{6}$$

where

$j = \sqrt{-1}$ = imaginary unit

σ = macroscopic total cross section

$\mathbf{dx}$ ($\mathbf{dy}$) = width of the Cartesian element in the x (y) direction

λ_x (λ_y) = wave number of the Fourier modes in the x (y) direction.

The Fourier ansatz accounts for the lagging of the incoming angular fluxes on face (1) of both cells 1 and 2 from the previous iteration. Similar lagged boundary conditions are written for the angular fluxes that are incoming on faces that the triangles share in the interior of the Cartesian element.

Substitution of the boundary conditions and of the Fourier ansatz into the original DFEM equations produces, after considerable algebra, the iteration matrix

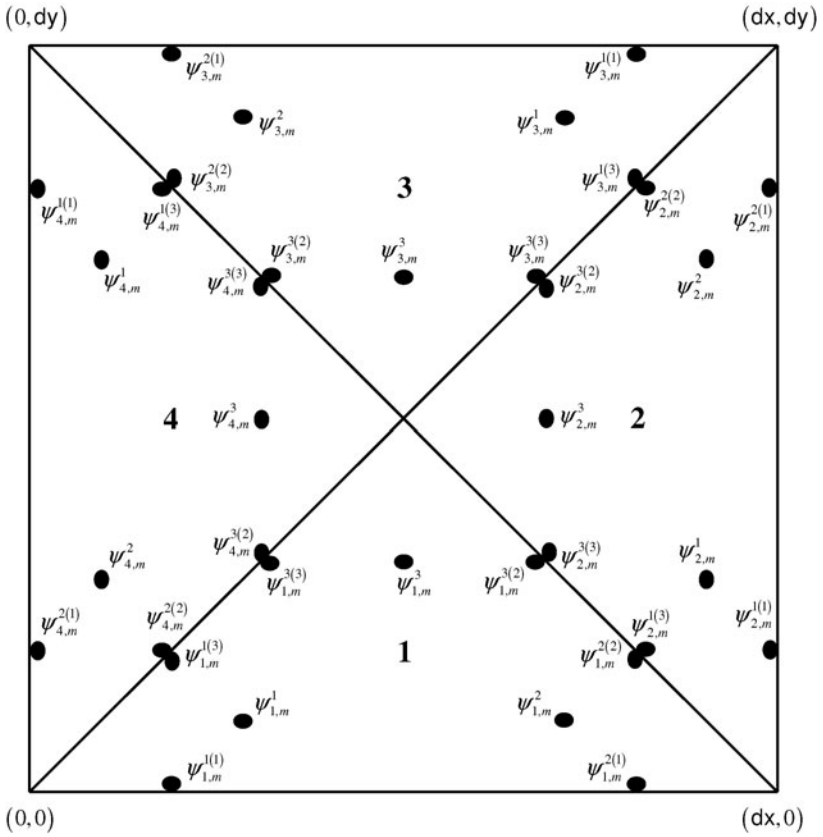

Fig. 1. Four-cell system for the Fourier analysis of cellwise bJ and bGS.

$\mathbf{T}_{\mathrm{bJ}}$ for the cellwise bJ algorithm. Given a certain quadrature order, and values for σ and the scattering ratio c, the spectral radius ρ of $\mathbf{T}_{\mathrm{bJ}}$ is finally obtained as a function of **dx** and **dy** as follows:

$$\rho(\mathbf{dx},\mathbf{dy}) = \max_{\lambda_x,\lambda_y} |\mathrm{Eig}(\mathbf{T}_{\mathrm{bJ}}(\mathbf{dx},\mathbf{dy},\lambda_x,\lambda_y))| \quad . \qquad (7)$$

Figure 2 is a plot of the spectral radius as a function of Cartesian element widths obtained for a level-symmetric S_4 quadrature with equal weights, assuming $\sigma = 1$ and $c = 0.5$. The results in Fig. 2 point to the fact that convergence of cellwise bJ can degrade for problems containing optically thin cells, even for values of the scattering ratio c less than 1. In fact, the spectral radius tends to 1 independent of the value of c, as the Cartesian element widths are decreased. This is similar to the behavior of the IPBJ algorithm that is analyzed in Ref. 4. Preconditioning of IPBJ with transport synthetic acceleration proved to be effective in improving the spectral properties of IPBJ, especially for optically thin problems, and was suggested as a possible remedy for cellwise bJ (Ref. 5). As the computational cells become optically thicker, memory of the information exchanged at the boundaries is lost, and the spectral properties of cellwise bJ are dominated by the fact that the "full" transport operator is locally inverted on each computational cell. This explains why, as evident in Fig. 2, the spectral radius tends to 0 in the optically-thick-cell regime. Hence, for sufficiently optically thick problems, the cellwise bJ algorithm is rapidly convergent even for diffusive problems that are scattering dominated with c close to 1; see Fig. 3.

The predictions of the Fourier analysis for cellwise bJ have been compared with the numerical results obtained from the implementation of the bJ sweep in a 2-D transport code. The results for ρ obtained for a level-symmetric S_4 quadrature with equal weights, a unit macroscopic total cross section, and scattering ratios of 0.5 and 0.99 are compared with the Fourier analysis in Tables I and II, respectively. To reproduce the conditions of the Fourier analysis, the 2-D transport code was used to solve a sequence of Cartesian meshes, with vacuum boundary conditions, characterized by increasing mesh size. Every Cartesian element in the mesh is subdivided into four triangular cells. Hence, the 1×1 mesh corresponds to the four-cell system depicted in Fig. 1. The results shown in Tables I and II can be understood in view of the fact that, while the theoretical spectrum from the Fourier analysis is obtained for an infinite medium, the actual spectrum incorporates the effect of particle leakage at the

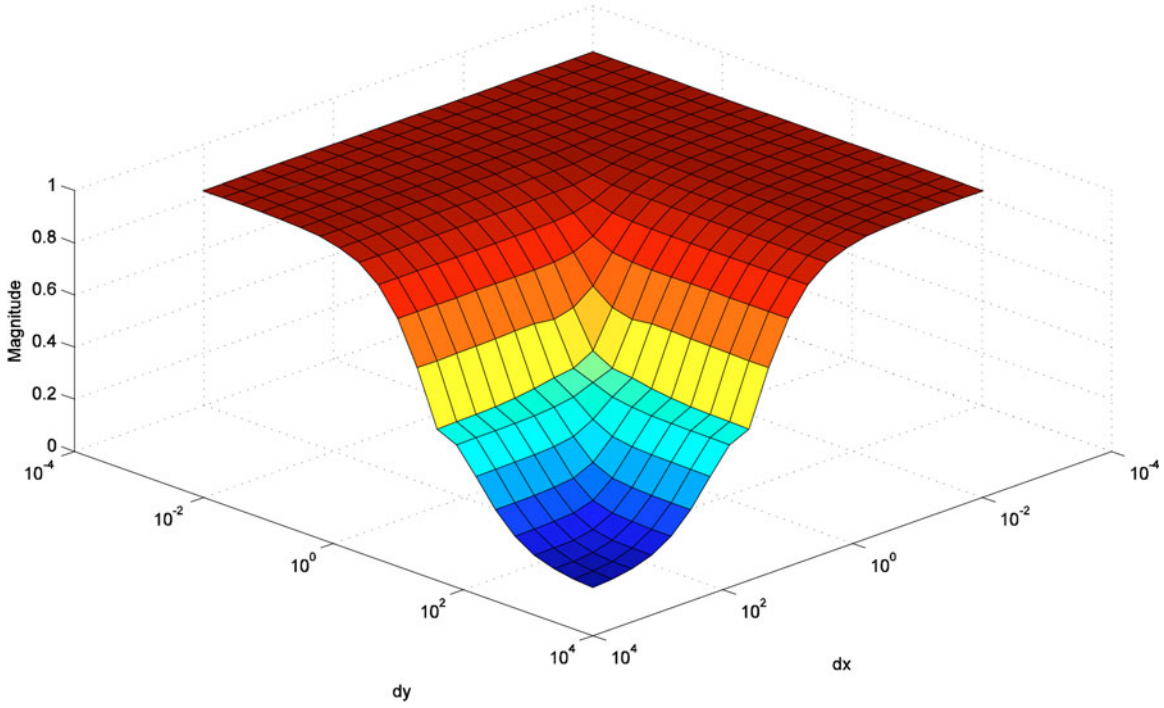

Fig. 2. Fourier analysis of one-group cellwise bJ: ρ.

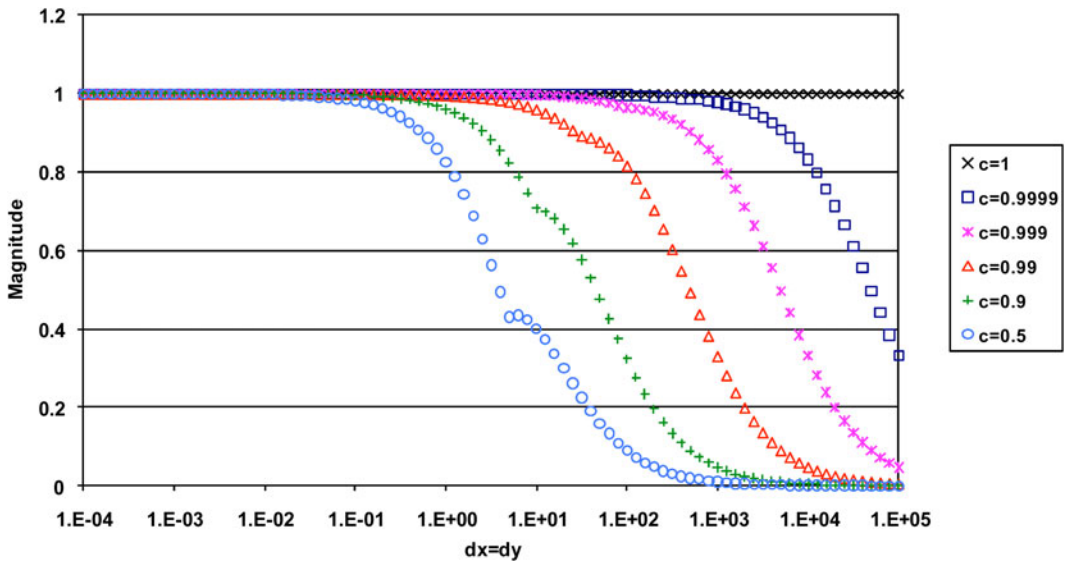

Fig. 3. Spectral radius of one-group cellwise bJ for various scattering ratios.

boundaries of the finite 2-D domain. As the mesh size is increased, for a given element width, the effect of leakage is less and less important. Therefore, the values obtained from the code are coincident with one another and with the theoretically predicted ρ in the optically thick regime. As the element width is decreased, the effect of leakage becomes dominant and the estimates of the spectral radius obtained with the code depart from the infinite-medium value. As expected, though, the actual spectral radius approaches the theoretical value as the number of cells in the mesh is increased, making the overall domain thicker. The theoretical values would ideally be obtained in the limit as the number of cells in the mesh goes to infinity.

TABLE I

Theoretical and Computed ρ of One-Group Cellwise bJ for $c = 0.5$

dx = dy	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{+1}	10^{+2}	10^{+3}	10^{+4}
1×1	0.005	0.017	0.057	0.181	0.194	0.039	0.004	4×10^{-4}
2×2	0.162	0.251	0.390	0.523	0.341	0.073	0.008	8×10^{-4}
4×4	0.475	0.571	0.679	0.701	0.381	0.083	0.009	9×10^{-4}
8×8	0.724	0.785	0.842	0.781	0.396	0.088	0.010	0.001
16×16	0.864	0.897	0.923	0.812	0.400	0.089	0.010	0.001
32×32	0.936	0.952	0.958	0.822	0.401	0.089	0.010	0.001
64×64	0.970	0.978	0.973	0.825	0.402	0.089	0.010	0.001
128×128	0.986	0.990	0.978	0.826	0.402	0.089	0.010	0.001
Fourier	0.999	0.998	0.981	0.827	0.402	0.089	0.010	0.001

TABLE II

Theoretical and Computed ρ of One-Group Cellwise bJ for $c = 0.99$

dx = dy	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{+1}	10^{+2}	10^{+3}	10^{+4}
1×1	0.007	0.024	0.085	0.327	0.759	0.718	0.237	0.032
2×2	0.184	0.288	0.462	0.725	0.910	0.789	0.308	0.043
4×4	0.502	0.607	0.740	0.897	0.947	0.805	0.323	0.046
8×8	0.742	0.808	0.889	0.965	0.957	0.813	0.328	0.047
16×16	0.875	0.911	0.957	0.987	0.960	0.816	0.330	0.047
32×32	0.941	0.961	0.985	0.994	0.960	0.816	0.330	0.047
64×64	0.973	0.984	0.995	0.995	0.960	0.816	0.330	0.047
128×128	0.988	0.994	0.998	0.996	0.960	0.816	0.330	0.047
Fourier	1.000	1.000	0.999	0.996	0.961	0.817	0.330	0.047

III. FOURIER ANALYSIS OF ONE-GROUP CELLWISE bGS

The results discussed in Sec. II indicate that convergence of cellwise bJ can degrade in problems containing optically thin cells. An alternative approach, resulting in a faster transition to the optically-thick-cell regime, is resorting to cellwise bGS in which the most up-to-date information available on the incoming angular fluxes is used at a cell's boundary. In the following we discuss the spectral properties of cellwise bGS, considering initially the most favorable case of a single processor, $Np = 1$.

For cellwise bJ the angular fluxes incoming on the boundaries of a mesh cell are always lagged from the previous iteration. Therefore, whether or not a cell boundary is coincident with the interface between two different processors, in the parallel implementation of the bJ sweep, has no consequence on the spectral properties of cellwise bJ. Hence, no mention on the number of processors was made either in devising the Fourier analysis for cellwise bJ or in discussing the results contained in Tables I and II. For the same reason, the order in which the four triangular cells over the Cartesian element depicted in Fig. 1 are swept in the bJ sweep has no impact on the spectral properties of cellwise bJ. Since for cellwise bGS the most up-to-date information available on the incoming angular fluxes is used at a mesh cell's boundaries, in the interior of a parallel subdomain, the above circumstances no longer hold true. Under the assumption of a single processor, a Fourier analysis for cellwise bGS can be devised by referring to the system of four triangular cells over a Cartesian element sketched in Fig. 1. We also assume the sweeping sequence 4-1-2-3 for the triangular mesh cells. This implies, for example, that the information on the angular fluxes incoming on cell 1 from cell 4 is already available at $(\ell + 1)$ while that from cell 2 is only available at (ℓ). To further fix ideas, while Eq. (5) for the Fourier ansatz previously formulated for cellwise bJ remains the same for cellwise bGS, Eq. (6) needs to be replaced with

$$\psi_{2,m}^{i(1)} = \psi_{4,m}^{k(\ell+1)} \exp(+j\lambda_x \sigma \mathbf{dx}) \quad , \quad i = 1,2; k = 2,1 \quad . \tag{8}$$

Therefore, the cellwise bGS algorithm is based on splitting the L operator into two contributions $\bar{L}_c$ and $\bar{L}_b$,

which are different from L_c and L_b, respectively, for cellwise bJ; see Eqs. (2) and (3). Since part of the contributions originally lagged from the previous iteration for cellwise bJ via L_b are attributed to $\bar{L}_c$, it is expected that the iteration operator T_{bGS} for cellwise bGS,

$$T_{\text{bGS}} = -(\bar{L}_c - SD)^{-1}\bar{L}_b \ , \tag{9}$$

should display improved spectral properties with respect to T_{bJ}. The latter expectation is verified by the comparison of the spectral radii of cellwise bJ and bGS presented in Fig. 4 for different values of the scattering ratio. For all the values of c attempted, the curve for cellwise bGS lies below the corresponding curve for cellwise bJ, except for sufficiently optically thin problems, where both curves approach a value of 1. In particular, cellwise bGS enters the optically-thick-cell regime, characterized by a value of ρ vanishing to 0, at a faster rate than cellwise bJ.

We implemented the "bGS sweep" in the 2-D transport code and used the code to verify the predictions from the Fourier analysis for cellwise bGS. To reproduce the conditions of the Fourier analysis, the code ran on a single processor to solve the sequence of Cartesian meshes, with vacuum boundary conditions, first introduced in Sec. II. The values for ρ predicted by the Fourier analysis for the level-symmetric S_4 quadrature, a unit macroscopic total cross section, and scattering ratios of 0.5 and 0.99 are verified by the estimates of ρ obtained from the code in Tables III and IV, respectively.

The dependence of the spectral properties of cellwise bGS on Np is because the information on the angular fluxes incoming on an interface between two adjacent processors, available in the present iteration, was communicated at the end of the previous iteration. As a simple illustration of the impact of an interface on the Fourier analysis for cellwise bGS, imagine that face (1) of cell 2, in Fig. 1, lies at the interface between two different processors. Under this assumption, for example, Eq. (8) is no longer valid and must be replaced with Eq. (6). Part of the cell's interior contribution exerted via $\bar{L}_c$ is therefore reattributed back to $\bar{L}_b$. This leads in turn to a new splitting of the L operator, $L = \bar{L}'_c + \bar{L}'_b$, and to a different iteration operator, $T_{\text{bGS1}Int}$, whose spectral properties are expected to be intermediate between those of T_{bGS} and T_{bJ}, respectively. This reasoning helps understanding the results presented in Fig. 5 for $c = 0.5$. The bGS and bJ curves are the same as in Fig. 4. The curve for two interfaces was obtained for the case in which both face (1) of cell 2 and face (1) of cell 3 coincide with an interface between different processors.

Further research is needed in order to possibly tie the parameter Np into the Fourier analysis for cellwise bGS. It is expected, though, that the curves obtained from this more refined analysis for a given c would have to lie somewhere in between the bGS curve and the bJ curve; see Fig. 5. The former curve is the lower envelope obtained for the most favorable case of $Np = 1$. The latter is the upper envelope and is equivalent to cellwise bGS for the worst case of $Np = Nc$, where Nc is the number of cells in the mesh.

It is noted that the fact that the spectral properties of the cellwise bGS algorithm are dependent on the particulars of the parallel decomposition stems from the

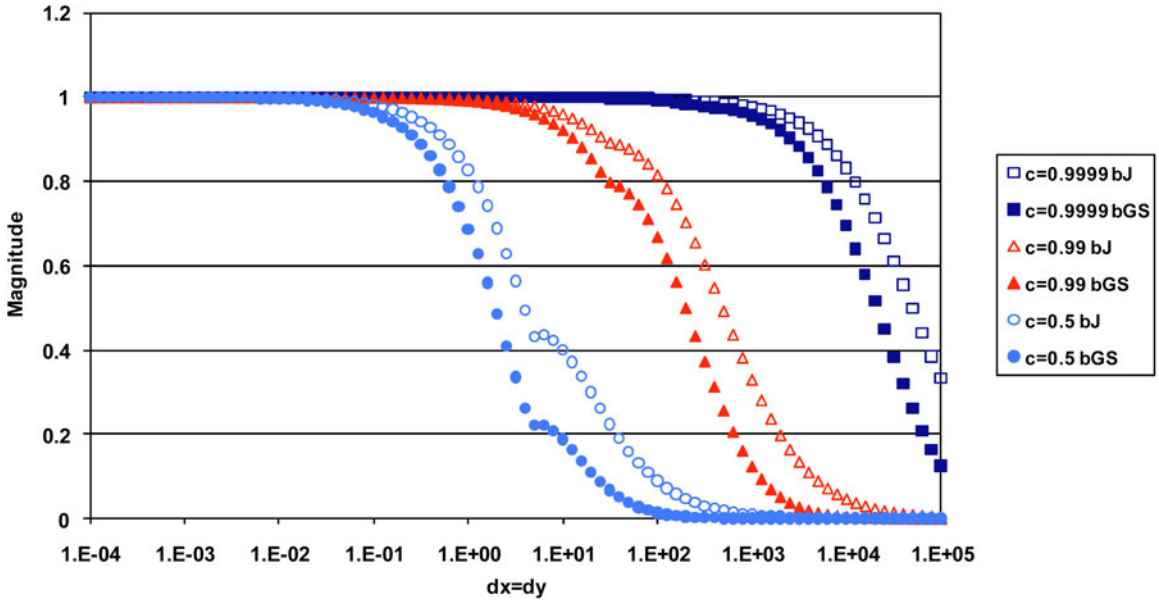

Fig. 4. Spectral radii of one-group cellwise bJ and bGS for various scattering ratios.

TABLE III

Theoretical and Computed ρ of One-Group Cellwise bGS for $c = 0.5$

dx = dy	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{+1}	10^{+2}	10^{+3}	10^{+4}
1×1	5×10^{-4}	0.003	0.013	0.062	0.065	0.007	3×10^{-4}	2×10^{-5}
2×2	0.037	0.078	0.172	0.295	0.131	0.010	4×10^{-4}	2×10^{-5}
4×4	0.231	0.333	0.469	0.503	0.168	0.013	5×10^{-4}	2×10^{-5}
8×8	0.524	0.618	0.712	0.618	0.182	0.014	5×10^{-4}	2×10^{-5}
16×16	0.747	0.806	0.852	0.665	0.186	0.015	5×10^{-4}	2×10^{-5}
32×32	0.875	0.907	0.919	0.681	0.187	0.015	5×10^{-4}	2×10^{-5}
64×64	0.941	0.957	0.947	0.686	0.188	0.015	5×10^{-4}	2×10^{-5}
128×128	0.973	0.980	0.957	0.688	0.188	0.015	5×10^{-4}	2×10^{-5}
Fourier	0.999	0.996	0.962	0.688	0.188	0.015	5×10^{-4}	2×10^{-5}

TABLE IV

Theoretical and Computed ρ of One-Group Cellwise bGS for $c = 0.99$

dx = dy	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{+1}	10^{+2}	10^{+3}	10^{+4}
1×1	8×10^{-4}	0.004	0.022	0.146	0.587	0.528	0.080	0.004
2×2	0.046	0.100	0.231	0.535	0.830	0.626	0.108	0.005
4×4	0.258	0.375	0.555	0.807	0.896	0.650	0.119	0.005
8×8	0.552	0.655	0.791	0.931	0.916	0.664	0.123	0.005
16×16	0.766	0.831	0.916	0.974	0.921	0.668	0.124	0.005
32×32	0.886	0.924	0.970	0.987	0.922	0.669	0.125	0.005
64×64	0.947	0.970	0.990	0.991	0.923	0.670	0.125	0.005
128×128	0.976	0.988	0.996	0.992	0.923	0.670	0.125	0.005
Fourier	1.000	0.999	0.999	0.992	0.923	0.670	0.125	0.005

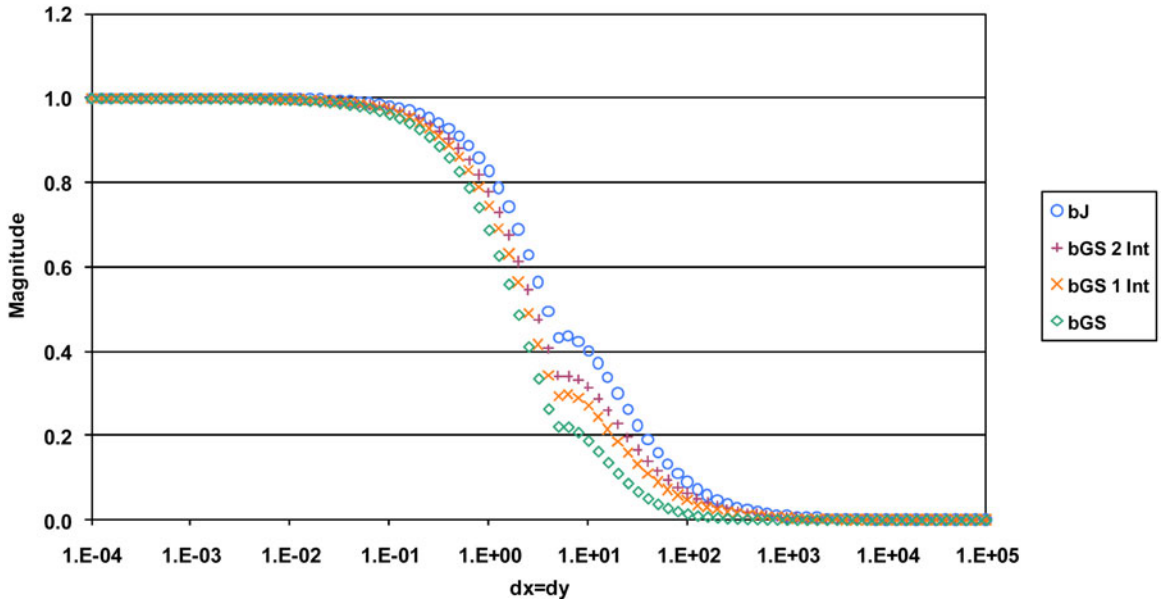

Fig. 5. Effect of processor interface on the spectral properties of cellwise bGS.

implementation of bGS used in this work, in which the information on the angular fluxes incoming on an interface between two adjacent processors, available in the present iteration, was communicated at the end of the previous iteration. The latter approach was followed in order to avoid the complications and potential inefficiencies associated with conducting FPS on unstructured meshes. It may be possible to combine parallel sweep technologies with the iterative splitting associated with a Gauss-Seidel iterative scheme, therefore using the most up-to-date information at an interface between two adjacent processors. Nonetheless, we will not explore here the endless possibilities associated with such hybrid approaches, which lie between the two extremes of FPS on one hand and cellwise bJ on the other. In this respect, our bGS iteration is only one of many such possible implementations.

IV. MULTIGROUP CELLWISE bGS ALGORITHM

We consider the steady-state, multigroup transport problem using two energy groups to illustrate the concepts exposed during this discussion. Later, results with more groups will be presented. The two-group transport problem is, in operator notation,

$$\begin{cases} \hat{\Omega}\cdot\nabla\psi_1 + \sigma_1\psi_1 = \sigma_{S11}\phi_1 + \sigma_{S12}\phi_2 + q_1 \\ \hat{\Omega}\cdot\nabla\psi_2 + \sigma_2\psi_2 = \sigma_{S21}\phi_1 + \sigma_{S22}\phi_2 + q_2 \end{cases} . \tag{10}$$

The terms on the left side in Eq. (10) are the streaming-plus-total-interaction operators for each group while the scalar fluxes on the right side of Eq. (10) are the integral of the group angular fluxes over all angles, performed via the discrete-to-moment operator D. The $\sigma_{Sgg'}$ cross sections represent isotropic scattering from group g' into group g, where $g, g' = 1, 2$.

The coupled system in Eq. (10) is often solved with SI, which is written for iteration (ℓ) as

$$\begin{bmatrix} \psi_1^{(\ell+1)} \\ \psi_2^{(\ell+1)} \end{bmatrix} = \begin{bmatrix} L_1 & 0 \\ 0 & L_2 \end{bmatrix}^{-1} \left(\begin{bmatrix} \sigma_{S11} & \sigma_{S12} \\ \sigma_{S21} & \sigma_{S22} \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix} \times \begin{bmatrix} \psi_1^{(\ell)} \\ \psi_2^{(\ell)} \end{bmatrix} + \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \right) . \tag{11}$$

On a parallel computer, it is the block inverses on the right side of Eq. (11) that are computed with FPS. The FPS forms the bulk of the computational effort in an S_N transport calculation. Efficiency of the FPS degrades as the number of processors increases. Note that this version of SI does not use nested, within-group iterations typically used in transport codes.

The two-group bGS algorithm employs the same spatial splitting of the group streaming-plus-total-interaction operators, as described in Sec. III, for each of the two groups. In this case, at iteration (ℓ), the cellwise bGS algorithm is

$$\begin{bmatrix} \psi_1^{(\ell+1)} \\ \psi_2^{(\ell+1)} \end{bmatrix} = \left(\begin{bmatrix} \bar{L}_{c1} & 0 \\ 0 & \bar{L}_{c2} \end{bmatrix} - \begin{bmatrix} \sigma_{S11} & \sigma_{S12} \\ \sigma_{S21} & \sigma_{S22} \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix} \right)^{-1} \times \left(- \begin{bmatrix} \bar{L}_{b1} & 0 \\ 0 & \bar{L}_{b2} \end{bmatrix} \begin{bmatrix} \psi_1^{(\ell)} \\ \psi_2^{(\ell)} \end{bmatrix} + \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \right) . \tag{12}$$

Again, the bGS variation on bJ uses a mix of updated fluxes $\psi_g^{(\ell+1)}$ and lagged fluxes $\psi_g^{(\ell)}$ operated on by the $\bar{L}_{bg}$ operators, which depends on the numbering of the mesh cells. The combination of operations on the right side of Eq. (12) is the bGS sweep.

The convergence rate of SI is the spectral radius of the matrix operator:

$$\mathbf{T}_{\mathrm{SI2}g} = \begin{bmatrix} L_1 & 0 \\ 0 & L_2 \end{bmatrix}^{-1} \begin{bmatrix} \sigma_{S11} & \sigma_{S12} \\ \sigma_{S21} & \sigma_{S22} \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix} . \tag{13}$$

Similarly, the convergence rate of cellwise bGS is given by the spectral radius of the matrix operator:

$$\mathbf{T}_{\mathrm{bGS2}g} = -\left(\begin{bmatrix} \bar{L}_{c1} & 0 \\ 0 & \bar{L}_{c2} \end{bmatrix} - \begin{bmatrix} \sigma_{S11} & \sigma_{S12} \\ \sigma_{S21} & \sigma_{S22} \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix} \right)^{-1} \times \begin{bmatrix} \bar{L}_{b1} & 0 \\ 0 & \bar{L}_{b2} \end{bmatrix} . \tag{14}$$

Fourier analysis indicates that the flat error mode is the dominant eigenvalue for SI. Because that error mode is constant in space, particle streaming can be ignored and we can compute the spectral radius of the iteration matrix on the right side of

$$\begin{bmatrix} \phi_1^{(\ell+1)} \\ \phi_2^{(\ell+1)} \end{bmatrix} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}^{-1} \begin{bmatrix} \sigma_{S11} & \sigma_{S12} \\ \sigma_{S21} & \sigma_{S22} \end{bmatrix} \begin{bmatrix} \phi_1^{(\ell)} \\ \phi_2^{(\ell)} \end{bmatrix} \tag{15}$$

to investigate the asymptotic convergence rate of SI. That is, if we let

$$\mathbf{T}_{\mathrm{SI2}g} = \mathbf{\Sigma}_T^{-1}\mathbf{\Sigma}_S = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}^{-1} \begin{bmatrix} \sigma_{S11} & \sigma_{S12} \\ \sigma_{S21} & \sigma_{S22} \end{bmatrix} , \tag{16}$$

the two-group spectral radius is

$$\rho_{\mathrm{SI2}g} = \max|\mathrm{Eig}(\mathbf{T}_{\mathrm{SI2}g})| . \tag{17}$$

The scattering cross sections in Eq. (16) must satisfy the inequalities

$$\begin{cases} \sigma_{S11} + \sigma_{S21} \le \sigma_1 \\ \sigma_{S12} + \sigma_{S22} \le \sigma_2 \end{cases} . \tag{18}$$

If equality holds in Eq. (18), then the spectral radius of the $\mathbf{T}_{\mathrm{SI2}g}$ matrix is exactly equal to 1, regardless of the

numerical values of the cross sections. This result applies for any number of energy groups. For example, we devise a problem for which ρ for two-group SI is 0.9999 by enforcing the constraint

$$\begin{cases} \sigma_{S11} + \sigma_{S21} = 0.9999\sigma_1 \\ \sigma_{S12} + \sigma_{S22} = 0.9999\sigma_2 \end{cases} \Rightarrow \rho_{\mathrm{SI}2g} = 0.9999 \ . \qquad (19)$$

In this case, SI would converge slowly. On the other hand, the convergence of GMRES depends on the shape of the eigenvalue spectrum and not just the size of the dominant eigenvalue.[16] Random scattering matrices can be used to generate an iteration matrix whose spectrum is widely distributed in the complex plane, with eigenvalues that are small relative to the size of the spectrum, a situation in which GMRES would converge slowly. Therefore, we will devise problems for which both SI and GMRES converge slowly by generating a dense, random scattering matrix and compute the total cross sections from the constraints that fix $\rho \simeq 1$.

We begin constructing such a multigroup problem with G energy groups by creating a random $G \times G$ matrix $\mathbf{R}$, with entries $r_{gg'}$ in the range [0,1], and renormalize the elements on each row of matrix $\mathbf{R}$ so that their sum equals 1. We then use the renormalized matrix elements to construct the scattering cross section matrix

$$\sigma_{Sgg'} = r_{gg'} \left(\sum_{g'=1}^{G} r_{gg'} \right)^{-1} \ , \quad g, g' = 1, \ldots, G \ . \qquad (20)$$

Note that renormalization is not strictly necessary to obtain a desired ρ for multigroup SI. We do so only to ensure that the total cross sections obtained in each group are not far from unity. If we take the total cross sections to be proportional to the sums of the entries on the columns of the scattering matrix

$$\sigma_{g'} = (1 + \alpha_{g'}) \sum_{g=1}^{G} \sigma_{Sgg'} \ , \quad g' = 1, \ldots, G \ , \qquad (21)$$

then coefficients $\alpha_{g'}$ are obtained as follows:

$$\alpha_{g'} = l + (u - l) r_{g'} \ , \quad g' = 1, \ldots, G \ , \qquad (22)$$

where $r_{g'}$ is a random value on [0,1], such that the α coefficients have random values on $[l, u]$. For example, by selecting $l = 10^{-5}$ and $u = 10^{-4}$, we obtain ρ for multigroup SI comprised between 0.9999 and 0.99999. If $l = u = 10^{-4}$, then ρ is exactly equal to 0.9999. Physically, this means that each α coefficient introduces a small neutron absorption in its corresponding energy group.

To verify this procedure for multigroup SI, we compare the value of ρ predicted via Eq. (17) with Fourier analysis for two-group SI, using cross sections obtained by selecting $l = 10^{-5}$ and $u = 10^{-4}$. This leads to scattering cross sections $\sigma_{S11} = 6.1789 \times 10^{-1}$, $\sigma_{S12} = 3.8211 \times 10^{-1}$, $\sigma_{S21} = 9.2747 \times 10^{-1}$, and $\sigma_{S22} = 7.2534 \times 10^{-2}$ and total cross sections $\sigma_1 = 1.5454$ and $\sigma_2 = 4.5468 \times 10^{-1}$. Using these values in Eq. (17) leads to $\rho_{\mathrm{SI}2g} = 0.99995$ for this model problem. Fourier analysis on a four-triangle, 1×1 square, as depicted in Fig. 1, gives the same result. We have verified, as well, that the maximum eigenvalue is found at the Fourier wave numbers $\lambda_x = \lambda_y = 0$; namely, it is the flat error mode that is most slowly attenuated by multigroup SI.

Fourier analysis for two-group SI was also compared with numerical results obtained from a multigroup FPS implementation in 2-D Cartesian coordinates. Measured estimates for ρ obtained for the model problem on an $I \times J$ domain, where I (J) is the number of mesh cells along x (y), are compared with the Fourier analysis in Table V for a sequence of meshes of squares with sides of length **dx** = **dy**. Every Cartesian element in the mesh is subdivided into four triangular cells. The difference in the results presented in Table V is explained because the Fourier analysis is over an infinite medium whereas the actual spectrum incorporates the effect of particle leakage at the boundaries of the problem, for which vacuum boundary conditions were specified. As the mesh

TABLE V

Theoretical and Computed ρ of Two-Group SI

dx = **dy**	10^{-1}	10^{0}	10^{+1}	10^{+2}	10^{+3}	10^{+4}	10^{+5}	10^{+6}
1×1	0.06336	0.41485	0.91905	0.99831	0.99993	0.99995	0.99995	0.99995
2×2	0.11910	0.60015	0.97183	0.99954	0.99994	0.99995	0.99995	0.99995
4×4	0.21469	0.77284	0.99186	0.99986	0.99995	0.99995	0.99995	0.99995
8×8	0.35891	0.89521	0.99778	0.99992	0.99995	0.99995	0.99995	0.99995
16×16	0.54046	0.96118	0.99939	0.99994	0.99995	0.99995	0.99995	0.99995
32×32	0.72195	0.98798	0.99981	0.99995	0.99995	0.99995	0.99995	0.99995
64×64	0.86236	0.99664	0.99991	0.99995	0.99995	0.99995	0.99995	0.99995
Fourier	0.99995	0.99995	0.99995	0.99995	0.99995	0.99995	0.99995	0.99995

TABLE VI

Theoretical and Computed ρ of Two-Group Cellwise bGS

dx = dy	10^{-1}	10^{0}	10^{+1}	10^{+2}	10^{+3}	10^{+4}	10^{+5}	10^{+6}
1×1	0.02209	0.13220	0.52027	0.90705	0.96422	0.78241	0.22617	0.01563
2×2	0.22834	0.49514	0.83448	0.97496	0.97528	0.84453	0.30419	0.01916
4×4	0.54655	0.77117	0.94952	0.99128	0.97780	0.85753	0.32751	0.02111
8×8	0.78134	0.91477	0.98602	0.99553	0.97889	0.86289	0.33598	0.02178
16×16	0.90729	0.97268	0.99614	0.99660	0.97915	0.86437	0.33831	0.02197
32×32	0.96410	0.99211	0.99879	0.99687	0.97918	0.86477	0.33891	0.02203
64×64	0.98728	0.99785	0.99947	0.99694	0.97927	0.86487	0.33911	0.02204
Fourier	0.99999	0.99997	0.99970	0.99696	0.97932	0.86487	0.33911	0.02204

size is increased, for a given element width, the effect of leakage becomes less significant. Overall, the spectral properties of multigroup SI appear to be similar to the well-known spectral properties of one-group SI (Ref. 7).

Table VI compares Fourier analysis of two-group bGS to a numerical implementation for the same problem as in Table V. Again, the results for multigroup bGS are similar to those for one group presented in Sec. III; that is, multigroup bGS is rapidly convergent in problems with optically thick cells, even for the highly scattering model problem, while convergence degrades for problems containing optically thin cells. This is the reverse of SI, which converges more quickly when cells are optically thin and more slowly when cells are thick.

V. IMPLEMENTATION OF MULTIGROUP bGS ON ROADRUNNER

The LANL Roadrunner machine has a massively parallel, hybrid computing architecture separated into several layers of communication pathways and processor types. While this idea is not completely new, Roadrunner was the first supercomputer to break the "petaflop barrier."[1] The smallest logical building block is commonly referred to as a TriBlade, which consists of an IBM LS21 blade connected to two QS22 blades using a special expansion card for the quad PCIe 8x interconnect; see Fig. 6. Each of the cores in the LS21 two-dual-core Opterons is connected to a Cell/B.E. CPU using a PCIe 8x bus. Each of the two QS22 blades has two IBM PowerXCell 8i Cell/B.E. CPUs. Each PowerXCell 8i CPU consists of a dual-threaded 64-bit PowerPC core [power processing element (PPE)] connected to eight SPEs. The SPEs are designed for running compute-intensive applications but depend on the PPE to run the operating system. In turn, the PPE depends on the SPEs to provide the bulk of the compute power.

Each TriBlade can be configured either as four message passing interface (MPI) nodes consisting of one Opteron core connected to a PPE and eight SPEs or as two MPI nodes consisting of two Opteron cores connected to a single PPE and sixteen SPEs. Overall, each TriBlade has >400 Gflops double precision, 8 Gbytes Opteron memory, and 8 Gbytes Cell/B.E. memory.

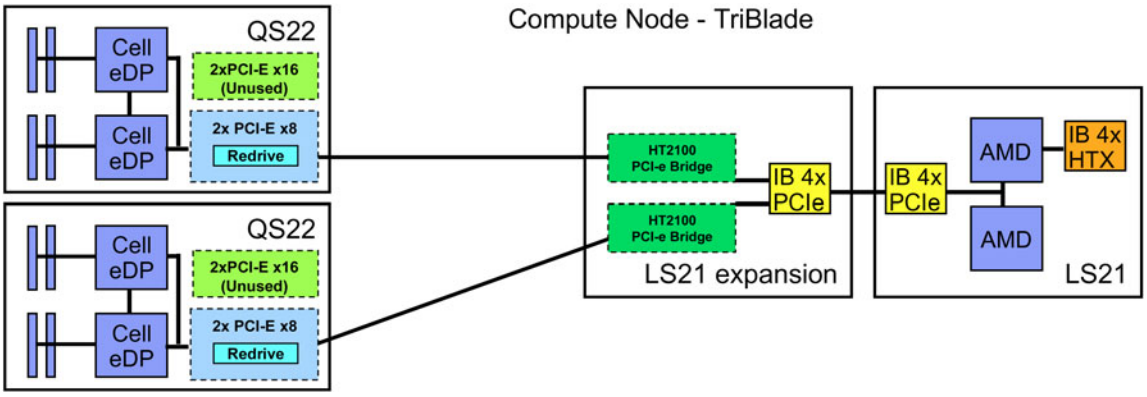

Fig. 6. Compute node of Roadrunner hybrid architecture.

The TriBlades can be clustered together using first-level infiniband (IB) switches into connected units (CUs), each of which has up to 180 TriBlades (or up to 720 MPI ranks). Redundant second-level IB switches are used to connect the CUs together. Several machine configurations are available at LANL with double-precision computation rates of up to 1 petaflop.

V.A. Single-Threaded Hybrid Implementation of bGS

The bGS sweep can be executed on the cores on the Opteron blades, without taking advantage of the Cell/B.E. processors. Numerical experiments shown later illustrate that the cellwise bGS algorithm is not viable in this case. This is because the times for the LU factorization and solution on which the algorithm is based increase cubically with the size of the linear system. For multigroup S_N transport, this means the algorithm can only be applied for a very small number of energy groups and S_N quadratures. The Cell/B.E. implementation of the LU decomposition in LAPACK (Ref. 17), which exploits up to 16 SPEs, scales much better with the size of the linear system. A hybrid implementation of the cellwise bGS algorithm restores the viability of the method because of the lower computational cost of the LU factorizations for typical numbers of energy groups and S_N quadrature orders.

The basic bGS sweep, conducted on the spatial mesh subdomain assigned to one MPI rank, is written in pseudocode in Fig. 7. For a given mesh cell, the size **N** is computed, which is the product of the number of S_N angles, the spatial degrees of freedom on the mesh cell (three for DFEM on triangles), and the number of coupled energy groups. The matrix **A** and right side **b** are constructed and a LU factorization and solution of matrix **A** computed. Finally, we store the cell angular fluxes in the global solution vector of angular fluxes before moving to the next cell. The rest of the S_N transport computations take place outside the bGS sweep code.

The most straightforward way to implement the above procedure on the Roadrunner architecture, and the one that requires the least modification of the original code, is to move the LU factorization, circled in Fig. 7, to the PPE as shown in Fig. 8. In this case, the Opteron still computes **N** and builds **A** and **b** as before. The Opteron signals the PPE that a linear system is ready to be solved at that point by writing **N** into the PPE mailbox. The mailbox is a communication mechanism whereby an unsigned integer can rapidly be transferred from the Opteron to the PPE. Since **A** and **b** can be large, the most efficient way for the Opteron to make them available to the PPE is to "share" memory for **A** and **b** with the PPE. This gives the PPE unidirectional access to the data so that the PPE then gets the data for **A** and **b** and proceeds with the LU factorization and solution via the Cell/B.E. implementation of the LAPACK routines.

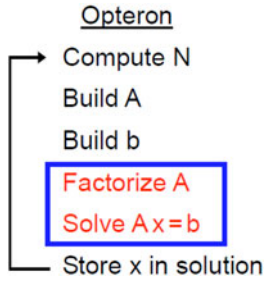

Fig. 7. A bGS computational kernel: traditional Opteron-only implementation.

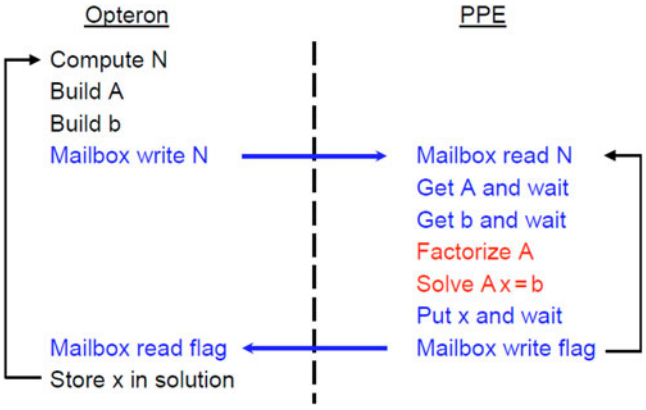

Fig. 8. A bGS computational kernel: single-threaded hybrid implementation.

It is noted that the calls to the LAPACK routines on the PPE take care both of the data movement between the PPE and the SPEs and of the partition of work on the data between the various SPE threads. Hence, by hiding the details of the SPE implementation, the LAPACK routines offer the hybrid programmer a certain level of abstraction in harnessing the computational power of the accelerators.

The PPE puts the solution into the Opteron memory space and sends a mailbox signal to the Opteron indicating that the solution for this computational cell is available for retrieval, at which point it is stored in the global solution vector of angular fluxes.

V.B. Dual-Threaded Hybrid Implementation of bGS

The single-threaded hybrid implementation discussed in Sec. V.A is straightforward, but it has a drawback; namely, the Opteron is idle while the PPE is working, and vice versa. Computational efficiency of the bGS sweep is improved by overlapping communication and computation using multithreading techniques,[18] as shown in Fig. 9. Communication threads (Comm

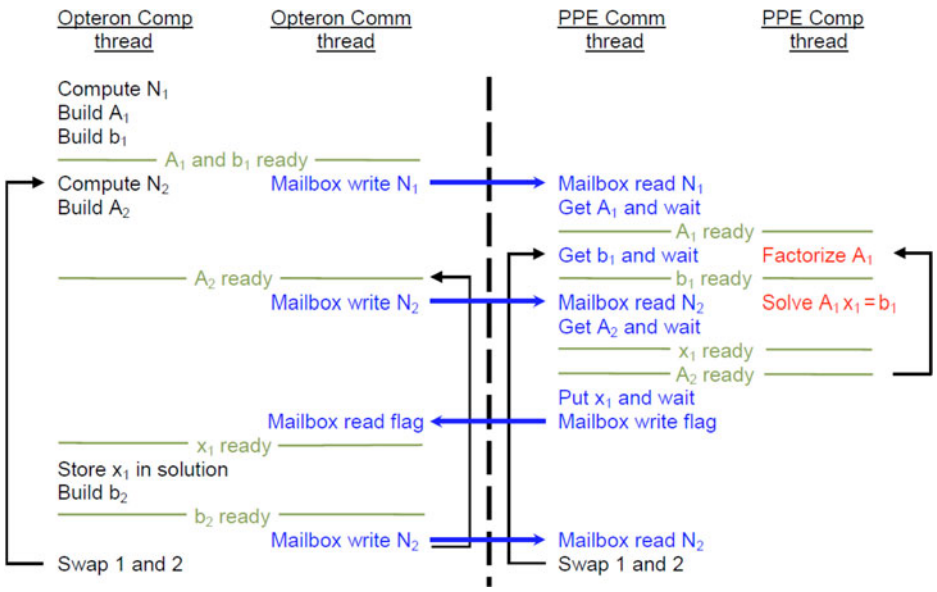

Fig. 9. A bGS computational kernel: dual-threaded hybrid implementation.

threads) and computation threads (Comp threads) take place simultaneously on both the Opteron and the PPE, for each MPI rank, using two buffers denoted with the subscripts 1 and 2 in Fig. 9. A Comp thread on the Opteron starts by computing $\mathbf{N_1}$ and building $\mathbf{A_1}$ and $\mathbf{b_1}$. The Opteron Comm thread then writes $\mathbf{N_1}$ in the PPE mailbox, but the Comp thread is not idle; it starts computing $\mathbf{N_2}$ and building the next matrix $\mathbf{A_2}$. Concurrently, the PPE Comm thread receives $\mathbf{N_1}$ and is ready to get $\mathbf{A_1}$. As soon as $\mathbf{A_1}$ is received, the PPE Comp thread performs a LU factorization. Once $\mathbf{b_1}$ is available, it computes the solution $\mathbf{x_1}$. Meanwhile, the PPE Comm thread receives $\mathbf{N_2}$ and then $\mathbf{A_2}$, and so on. Since $\mathbf{b_2}$ depends on $\mathbf{x_1}$ for bGS, the Opteron builds $\mathbf{b_2}$ after storing $\mathbf{x_1}$ and reuses $\mathbf{N_2}$ to flag the PPE that $\mathbf{b_2}$ is ready. Overlapping of the threads takes place by swapping the buffers, illustrated by the loops in Fig. 9. Race conditions between threads are avoided using semaphores indicated by the "ready" labeled horizontal lines spanning the Comp threads and Comm threads. Appropriate conditions (not shown) ensure that the loops are exited once a bGS sweep is completed.

V.C. Speedup of the Hybrid Implementation of bGS

To validate the speedup provided by the single- and dual-threaded implementations of cellwise bGS, we used an 8×8 grid of squares, each of which is 10^{+4} cm on a side. The grid was further subdivided into four triangular cells for every square—a total of 256 mesh cells. Scattering ratio $c = 0.99$, $\sigma = 1$ cm^{-1}, a uniform fixed source of 1 m^{-3} s^{-1}, and vacuum boundary conditions were specified. Square Chebyshev-Legendre quadratures vary from order $n = 2$ to 40, such that the linear systems sizes vary as $\mathbf{N} = 3n^2$.

Figure 10 shows the execution times for the GMRES(20) solution, to a relative convergence tolerance of 10^{-5} using different numbers of processors ($Np =$ 4, 8, and 16) for the same problem. The Opteron-only implementation of bGS is labeled GSOP, while GSST and GSDT refer to the single- and dual-threaded hybrid implementations of bGS, respectively, using 16 SPEs for every process.

The dual-threaded, double-buffered version of bGS is almost twice as fast as the single-threaded version because the Comm thread and Comp thread are completely overlapped. When $\mathbf{N}$ is less than ~300, the hybrid implementations are slower than the Opteron-only implementation, because of the overhead associated with moving data between the Opteron and the PPE. When the linear system size increases, the LU decomposition time on the Opteron-only implementation begins to dominate, while the hybrid solution times increase at a much slower rate, such that the hybrid implementations are up to 100 times faster.

VI. STRONG AND WEAK SCALING RESULTS

We now compare our bGS sweep implementation to FPS for a more complex problem that comprises two square regions, an outer region 5 m wide surrounding an inner region 2.5 m wide. The outer region contains material 1 (Mat 1) and the inner region material 2 (Mat 2); see Fig. 11. Table VII details the configurations that we used to construct four problems of increasing material

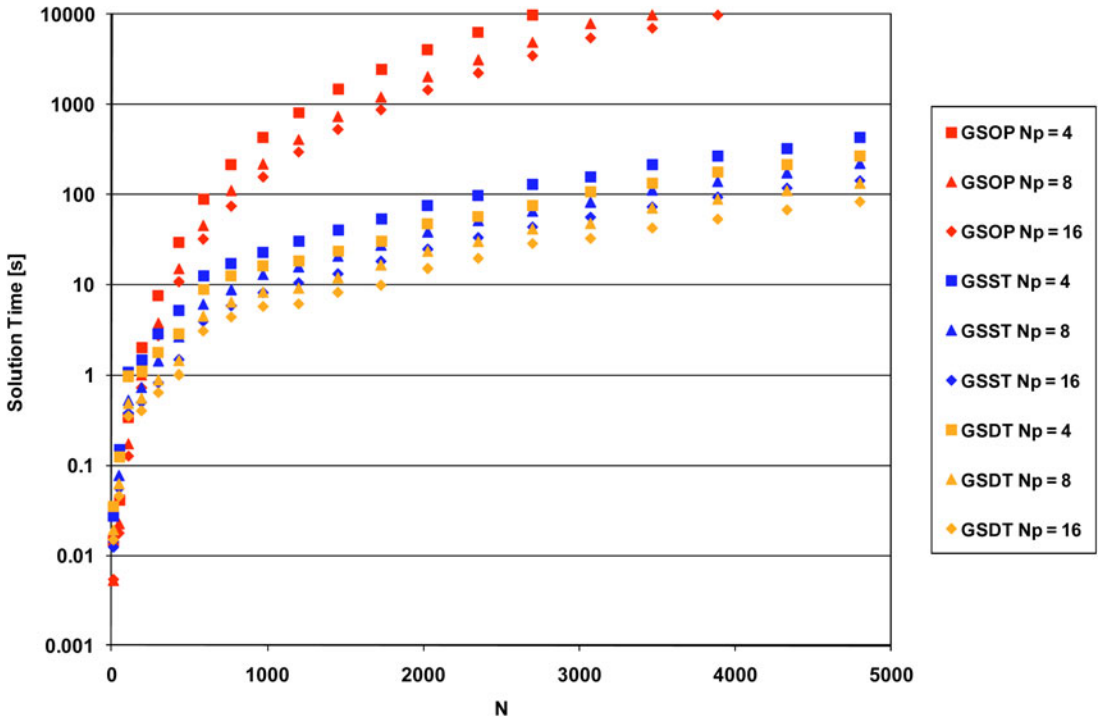

Fig. 10. Speedup of bGS on the hybrid architecture.

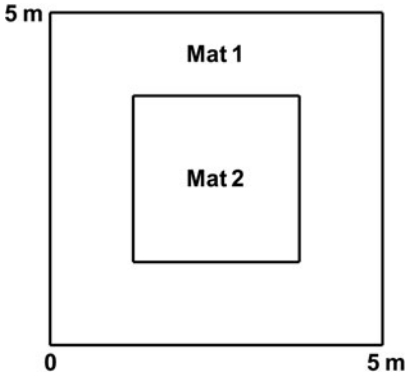

Fig. 11. Two-region configuration.

heterogeneity. Because cellwise bGS performs well in optically thick problems, Mat 1 is chosen to be optically thick and characterized by a scattering ratio of 0.9999. The properties of Mat 2 are then varied in such a way that the convergence rate of GMRES for the FPS implementation is adversely affected. In Table VII and in Figs. 12 and 13, the label Hom stands for a homogeneous problem while the label Het identifies the heterogeneous problems. The first digit in the homogeneous problem is used to denote the exponent in the total cross section, while the first and second digits of the heterogeneous problems indicate the exponents in the cross sections for the heterogeneous problems in Mat 1 and Mat 2, respectively.

TABLE VII

One-Group Problems of Increasing Material Heterogeneity for the Two-Region Configuration

Problem	σ_1 (m^{-1})	σ_{S1} (m^{-1})	σ_2 (m^{-1})	σ_{S2} (m^{-1})
Hom66	$1.0001 \times 10^{+6}$	$1. \times 10^{+6}$	$1.0001 \times 10^{+6}$	$1. \times 10^{+6}$
Het66	$1.0001 \times 10^{+6}$	$1. \times 10^{+6}$	$1. \times 10^{+6}$	$0.9999 \times 10^{+6}$
Het65	$1.0001 \times 10^{+6}$	$1. \times 10^{+6}$	$1. \times 10^{+5}$	$0.999 \times 10^{+5}$
Het64	$1.0001 \times 10^{+6}$	$1. \times 10^{+6}$	$1. \times 10^{+4}$	$0.99 \times 10^{+4}$

As long as the inner region is not too optically thin, we expect that the convergence of bGS will be fairly insensitive to material discontinuities. The spatially discretized problem is solved on an unstructured triangular mesh comprising 10 454 mesh cells, and we vary the number of energy groups used in each of the four problems: 1, 2, 10, 20, and 30. We use an S_4 level-symmetric angular quadrature, which has 12 discrete ordinates. Thus, the dimension of linear systems increases with increasing number of groups. For the multigroup extensions of the four one-group problems described in Table VII, we construct the scattering and total cross sections for a material as described in Sec. IV, namely, in such a way as to ensure that the eigenvalue of maximum magnitude for the multigroup SI iteration is equal to the scattering ratio selected for the material in the one-group cases. We assume vacuum boundary conditions and a fixed uniform source of strength 1 cm^{-3} s^{-1} throughout both material regions. Solutions are computed using GMRES(50) to a relative convergence tolerance of 10^{-5}.

VI.A. Strong Scaling of bGS and FPS

Strong scaling is measured by increasing the number of processors employed in the parallel computation, from 32 to 512 in powers of 2. The results of the strong scaling are reported in Fig. 12, showing the solution times and number of iterations versus the number of processors, plotted separately for each of the various numbers of energy groups. Closed markers connected with solid lines refer to the dual-threaded implementation of cellwise bGS (GSDT), and open markers with dashed lines refer to the (Opteron-only) implementation of FPS (PSWP).

For $g = 1$, solution times with the bGS implementation are fairly constant for the four types of problems, while the FPS implementation solution times increase with increasing material heterogeneity. For a given problem and number of groups, the number of iterations for FPS is constant with respect to Np. The slight increase in the number of iterations for bGS, as Np is increased, is a consequence of the effect of processor interfaces on the spectral properties of bGS because the terms involving $\bar{L}_b$ on the parallel-decomposed mesh boundaries are always lagged; see Sec. III. The bGS implementation also displays better scaling properties than the FPS implementation, for which the dependencies between processors result in a sweep schedule that scales poorly as Np becomes large.[9,10] This becomes evident on more than 256 processors.

Similar results are observed for the $g = 2$ and 10 calculations. For $g \geq 20$, the bGS implementation slows down relative to the FPS implementation due to the increasing size of the linear systems for a larger number of groups. Even though the execution times are slower, the scaling of the bGS implementation is better than the FPS implementation because bGS requires fewer iterations compared to FPS. Even for a homogeneous configuration, convergence of the multigroup problems slows as a consequence of the eigenvalue distribution arising out of the randomly generated multigroup scattering operator; this effect is more pronounced for the FPS implementation than for the bGS implementation.

Finally, the strong parallel efficiency of bGS on Roadrunner can achieve 73% at 512 processors, compared with 32 processors, while efficiency is 34% for FPS. The above results were computed for the Het64 problem solved with 20 energy groups, using the solver time data reported on the ordinate axis of Fig. 12d.

VI.B. Weak Scaling of bGS and FPS

We compared weak scaling by considering the strong scaling unstructured triangular mesh, comprising 10 454 cells, that was used for the initial number of processors, $Np = 32$. We generated a sequence of meshes such that the average number of cells per processor was roughly the same as for the initial mesh, that is, 326 mesh cells per processor. In other words, as the number of processors is increased for the fixed-size problem domain, the mesh is increasingly refined and the spatial discretization becomes increasingly resolved.

Results in Fig. 13 show that the bGS implementation on the hybrid architecture displays better weak scaling than the FPS implementation and can be competitive with FPS, once the number of processors becomes large enough. This is again due to scheduling dependencies among parallel subdomains associated with the FPS implementation and to the resilience of the GMRES solution using the bGS splitting in the presence of material discontinuities. The increase in GMRES iterations for bGS is due to the triangles becoming progressively optically thinner as the meshes are refined and the fact that bGS spectral properties degrade as the cells become thinner.

Finally, it is noted that the weak parallel efficiency of bGS on Roadrunner is 58% at 512 processors, compared with 32 processors, while it is 10% for FPS. The latter results were also computed for the Het64 problem solved with 20 energy groups, using the solver time data reported on the ordinate axis of Fig. 13d.

VII. CONCLUSIONS

We have performed a Fourier analysis for the one-group, cellwise bJ and bGS iterative algorithms for S_N transport. The results of the Fourier analysis show that convergence of both algorithms can degrade in problems containing optically thin cells, for which the spectral radii of both cellwise bJ and bGS tend to a value of 1, independent of the scattering ratio c. In the opposite limit of optically thick cells both cellwise bJ and bGS are rapidly convergent, with a spectral radius vanishing to 0.

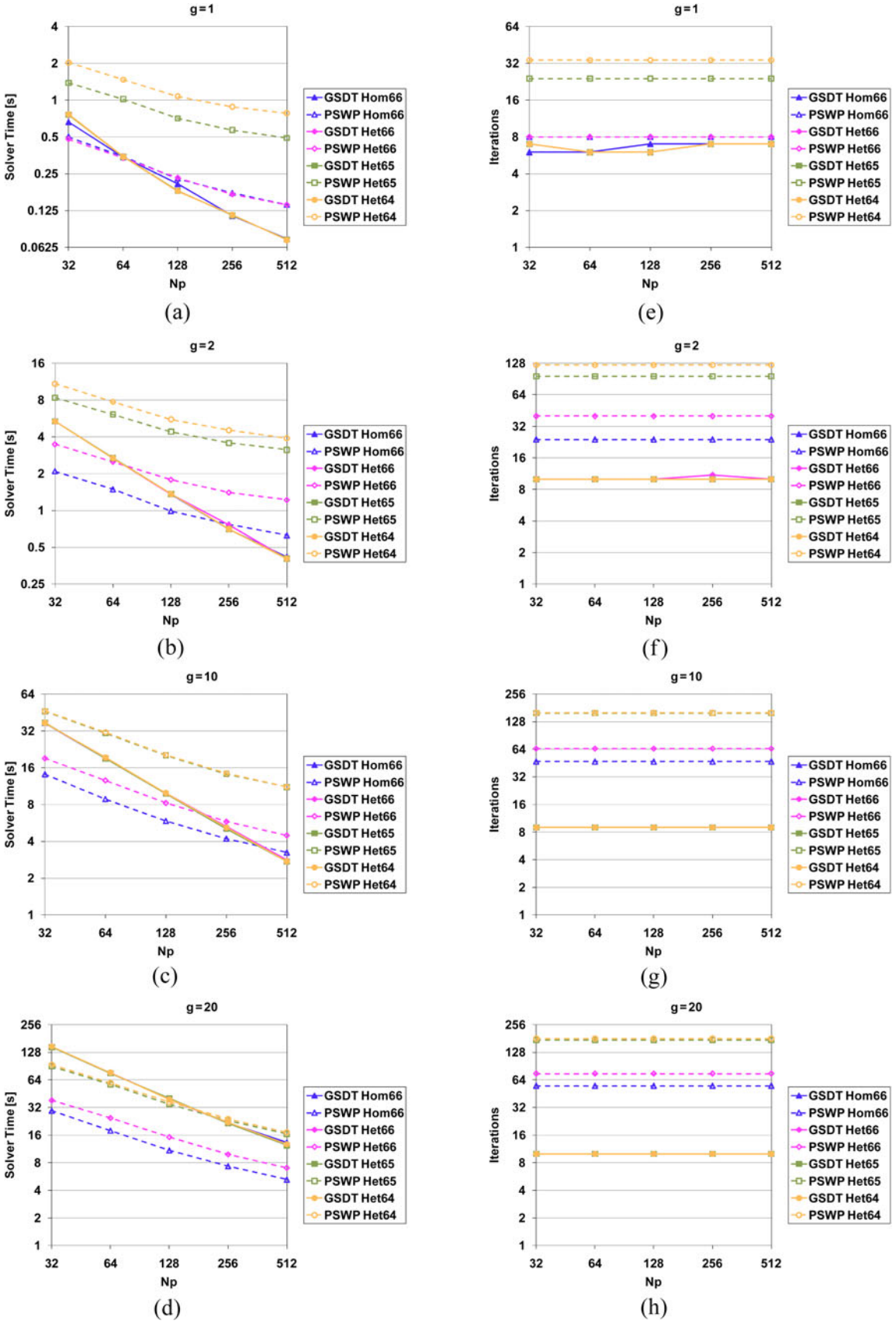

Fig. 12. Strong scaling of bGS compared to FPS.

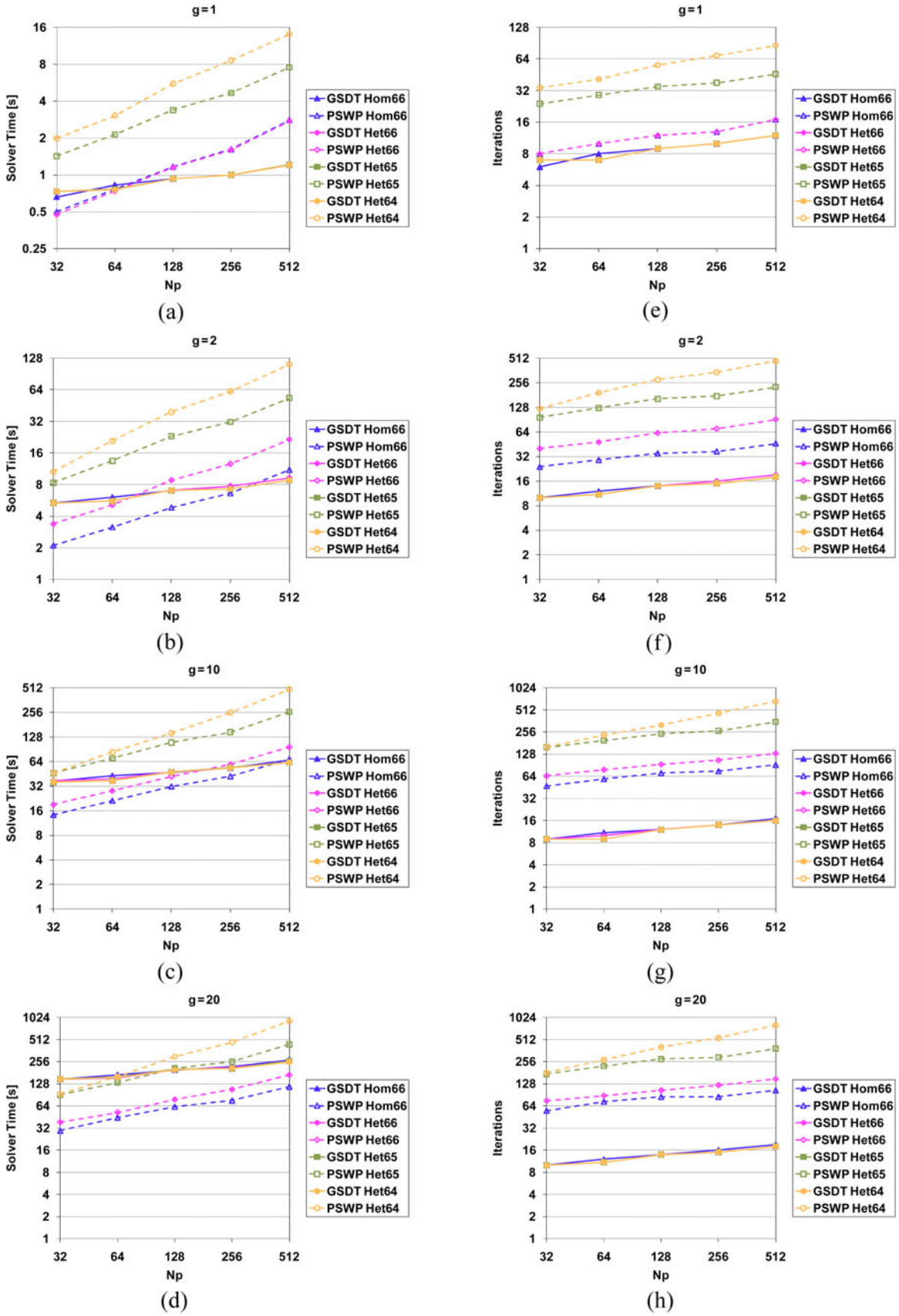

Fig. 13. Weak scaling of bGS compared to FPS.

Interestingly, this feature also holds for scattering-dominated problems while bGS displays superior spectral properties by approaching the asymptotic, thick-cell regime at convergence rates higher than bJ.

By virtue of its superior spectral properties, we have implemented a multigroup, dual-threaded, double-buffered version of the bGS sweep on the Roadrunner hybrid computer architecture. When combined with the Cell/B.E. implementation of a LU decomposition and solution, we have shown that the bGS splitting is a viable solution algorithm, when solved with GMRES on the Opteron level of this architecture.

We observed that the bGS implementation outperformed the traditional FPS implementation of SI on large numbers of processors by virtue of superior strong and weak scaling. While bGS holds the potential of being an efficient algorithm for massively parallel transport calculations in the model problems we devised for our numerical experiments, further refinements to the algorithm involving energy-splitting strategies will be investigated in order to solve problems for applications with realistic data and large numbers of energy groups.

The speedup and scalings we observed, while impressive, are not sufficient to make bGS competitive with the FPS algorithms under all circumstances. Other hybrid architectures using even faster accelerators, such as those based on GPGPUs or field-programmable gate arrays hold the potential to further speed up a LU factorization on which the bGS implementation is based. The overlapped dual-threaded approach to the bGS sweep could be applied to other hybrid architectures.

ACKNOWLEDGMENTS

One author (M. R.) would like to thank T. M. Kelley and B. K. Bergen, in the Applied Computer Science Group at LANL, for helpful discussions concerning hybrid programming techniques and for granting access to the Cerrillos TriBlade cluster under project C09_hybridca.

This information has been authored by an employee or employees of Los Alamos National Security, operator of LANL, under contract DE-AC52-06NA25396 with the U.S. Department of Energy.

REFERENCES

1. D. GRICE et al., "Breaking the Petaflops Barrier," *IBM J. Res. Dev.*, **53**, *5*, 1:1 (2009).

2. C. H. CRAWFORD et al., "Accelerating Computing with the Cell Broadband Engine Processor," *Proc. Conf. Computing Frontiers*, Ischia, Italy, May 5–7, 2008, p. 3.

3. J. S. WARSA, "A Continuous Finite Element–Based, Discontinuous Finite Element Method for S_N Transport," *Nucl. Sci. Eng.*, **160**, 385 (2008).

4. M. ROSA, J. S. WARSA, and J. H. CHANG, "Fourier Analysis of Inexact Parallel Block-Jacobi Splitting with Transport Synthetic Acceleration," *Nucl. Sci. Eng.*, **164**, 248 (2010).

5. M. ROSA and J. S. WARSA, "Extension of a Transport Synthetic Acceleration Scheme to the Cell-Wise Block-Jacobi and Gauss-Seidel Algorithms," *Trans. Am. Nucl. Soc.*, **101**, 1103 (2009).

6. M. ROSA, J. S. WARSA, and T. M. KELLEY, "Fourier Analysis of Cell-Wise Block-Jacobi Splitting in Two-Dimensional Geometry," *Proc. Int. Conf. Mathematics, Computational Methods & Reactor Physics*, Saratoga Springs, New York, May 3–7, 2009, American Nuclear Society (2009).

7. M. L. ADAMS and E. W. LARSEN, "Fast Iterative Methods for Discrete-Ordinates Particle Transport Calculations," *Prog. Nucl. Energy*, **40**, *1*, 3 (2002).

8. M. ROSA, J. S. WARSA, and M. PERKS, "Implementation of a Cell-Wise Block-Gauss-Seidel Iterative Method for S_N Transport on a Hybrid Parallel Computer Architecture," *Proc. Int. Conf. Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Rio de Janeiro, Brazil, May 8–12, 2011.

9. S. D. PAUTZ, "An Algorithm for Parallel S_n Sweeps on Unstructured Meshes," *Nucl. Sci. Eng.*, **140**, 111 (2002).

10. S. J. PLIMPTON et al., "Parallel S_n Sweeps on Unstructured Grids: Algorithms for Prioritization, Grid Partitioning, and Cycle Detection," *Nucl. Sci. Eng.*, **150**, 267 (2005).

11. K. R. KOCH, R. S. BAKER, and R. E. ALCOUFFE, "Solution of the First-Order Form of Three-Dimensional Discrete Ordinates Equations on a Massively Parallel Machine," *Trans. Am. Nucl. Soc.*, **65**, 198 (1992).

12. R. S. BAKER and K. R. KOCH, "An S_n Algorithm for the Massively Parallel CM-200 Computer," *Nucl. Sci. Eng.*, **128**, 312 (1998).

13. R. S. BAKER et al., "Solution of the First-Order Form of the Multi-Dimensional Discrete Ordinates Equations on a Two-Level Heterogeneous Processing System," *Trans. Am. Nucl. Soc.*, **105**, 510 (2011).

14. C. GONG et al., "GPU Accelerated Simulations of 3D Deterministic Particle Transport Using Discrete Ordinates Method," *J. Comput. Phys.*, **230**, 6010 (2011).

15. C. GONG et al., "Particle Transport with Unstructured Grid on GPU," *Comput. Phys. Comm.*, **183**, 588 (2012).

16. Y. SAAD, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, Massachusetts (1996).

17. E. ANDERSON et al., *LAPACK Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania (1999).

18. B. LEWIS and D. J. BERG, *Multithreaded Programming with Pthreads*, Sun Microsystems Press, Upper Saddle River, New Jersey (1998).