AIAA Journal
Vol. 47, No. 10, October 2009

Massively Parallel Solution of the BiGlobal Eigenvalue Problem Using Dense Linear Algebra

Daniel Rodríguez* and Vassilis Theofilis†
Universidad Politécnica de Madrid, E-28040 Madrid, Spain

DOI: 10.2514/1.42714

Linear instability of complex flows may be analyzed by numerical solutions of partial-derivative-based eigenvalue problems; the concepts are, respectively, referred to as BiGlobal or TriGlobal instability, depending on whether two or three spatial directions are resolved simultaneously. Numerical solutions of the BiGlobal eigenvalue problems in flows of engineering significance, such as the laminar separation bubble in which global eigenmodes have been identified, reveal that recovery of (two-dimensional) amplitude functions of globally stable but convectively unstable flows (i.e., flows which sustain spatially amplifying disturbances in a local instability analysis context) requires resolutions well beyond the capabilities of serial, in-core solutions of the BiGlobal eigenvalue problems. The present contribution presents a methodology capable of overcoming this bottleneck via massive parallel solution of the problem at hand; the approach discussed is especially useful when a large window of the eigenspectrum is sought. Two separated flow applications, one in the boundary-layer on a flat plate and one in the wake of a stalled airfoil, are briefly discussed as demonstrators of the class of problems in which the present enabling technology permits the study of global instability in an accurate manner.

Nomenclature

$\mathcal{D}_x$, $\mathcal{D}_y$	=	$\partial/\partial x$, $\partial/\partial y$
mem_d	=	memory required for one floating-point complex number
$\mathrm{mem}_{\mathrm{proc}}$	=	memory available for computations on each processor
N_{arrays}	=	number of arrays to be stored
N_{var}	=	number of variables and equations
N_x, N_y	=	Chebyshev–Gauss–Lobatto collocation points in the x and y directions
p	=	number of processors
Re	=	Reynolds number
t	=	time
$t_{\mathrm{AR,it}}$	=	time required for one Arnoldi iteration
t_{EIG}	=	time required for the eigenvalues and eigenvectors calculation
t_{EVP}	=	time required for the eigenvalue problems creation
t_{LU}	=	time required for the matrix shifting and lower–upper decomposition
$\bar{u}$, $\bar{v}$, $\bar{w}$	=	basic flow streamwise, wall-normal, and spanwise velocity components
x, y, z	=	streamwise, wall-normal, and spanwise spatial coordinates

I. Introduction

Linear instability analysis of flows has been a growing discipline during the last century [1,2]. This theory permits determination of the conditions under which a given flow amplifies small perturbations, thus evolving into a different (nonlinear) state; one of the key aims of linear theory is the prediction of laminar-turbulence transition via solution of conceptually simple eigenvalue problems (EVPs). In practice, the solution of the linear EVP resulting from superposing three-dimensional small-amplitude perturbations upon a three-dimensional basic flow (i.e., flow which is inhomogeneous in all three spatial dimensions) presents a numerically daunting task. Simplifications on the form of the basic flow, the stability of which is analyzed, are called for, the strongest of which is that of a "parallel" basic flow, i.e., a steady one- or two-component, one-dimensional velocity profile. The numerical solution of the corresponding EVP, of the Orr–Sommerfeld class, is currently straightforward and may be obtained with almost no restrictions. However, considering inhomogeneous basic flows in two or three spatial directions results in partial-derivative EVPs, on occasion requiring state-of-the-art algorithms and hardware for their solution. The present contribution discusses one such methodology for the solution of the linear BiGlobal EVP.

Concretely, in incompressible flow, the problem to be solved is obtained by assuming modal perturbations and homogeneity in one spatial direction, say the spanwise direction z. Eigenmodes are introduced into the linearized Navier–Stokes and continuity equations according to

$$(\hat{\mathbf{q}}^*, \hat{p}^*) = (\hat{\mathbf{q}}(x, y), \hat{p}(x, y))e^{+i(\beta z-\omega t)} \tag{1}$$

where $\hat{\mathbf{q}}^* = (\hat{u}^*, \hat{v}^*, \hat{w}^*)^T$ and $\hat{p}^*$ are, respectively, the vector of amplitude functions of linear velocity and pressure perturbations, superimposed upon steady two-dimensional, two- ($\bar{w} \equiv 0$) or three-component, $\bar{\mathbf{q}} = (\bar{u}, \bar{v}, \bar{w})^T$, steady basic states. The spanwise wave number β is associated with the spanwise periodicity length L_z through $L_z = 2\pi/\beta$. Substitution of Eq. (1) into the linearized equations of motion results in the complex BiGlobal eigenvalue problem [3]

$$\hat{u}_x + \hat{v}_y + i\beta\hat{w} = 0 \tag{2}$$

$$(\mathcal{L} - \bar{u}_x + i\omega)\hat{u} - \bar{u}_y\hat{v} - \hat{p}_x = 0 \tag{3}$$

$$-\bar{v}_x\hat{u} + (\mathcal{L} - \bar{v}_y + i\omega)\hat{v} - \hat{p}_y = 0 \tag{4}$$

$$-\bar{w}_x\hat{u} - \bar{w}_y\hat{v} + (\mathcal{L} + i\omega)\hat{w} - i\beta\hat{p} = 0 \tag{5}$$

where

Presented as Paper 0594 at the 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 7–10 January 2008; received 12 December 2008; revision received 14 April 2009; accepted for publication 30 April 2009.

*School of Aeronautics, Plaza Cardenal Cisneros 3. Member AIAA.
†School of Aeronautics, Plaza Cardenal Cisneros 3; vassilis@aero.upm.es. Member AIAA.

$$\mathcal{L} = \frac{1}{Re}\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} - \beta^2\right) - \bar{u}\frac{\partial}{\partial x} - \bar{v}\frac{\partial}{\partial y} - i\beta\bar{w} \tag{6}$$

In the compressible case, an Ansatz analogous to Eq. (1) may be substituted into the compressible linearized equations of motion. Although the form of the BiGlobal EVP is more involved [4,5], in this case too, a system of (five) coupled equations for the disturbance amplitude functions is obtained.

The spatial discretization of either compressible or incompressible BiGlobal EVPs results in large matrices which, when stored in core, confine the resolution employed to low Reynolds numbers. On the other hand, flows of industrial interest usually involve complex geometries at high Reynolds numbers, requiring resolutions which cannot be handled by the currently available top-end serial machines.

In this paper, two problems representative of the computational difficulties associated with the BiGlobal approach in open systems are considered. The first problem is the instability of a laminar boundary layer on a flat plate with an embedded separation bubble. Unlike earlier works [6], where homogeneous Dirichlet boundary conditions have been used to close the system and to permit the entrance of wavelike disturbances of the Tollmien–Schlichting class into the integration domain, a Robin boundary condition is used at the inflow boundary. This boundary condition imposes a relation between the wave number and the frequency using information from local analysis. Numerically, this boundary condition prohibits reducing the EVP to one with real coefficients. In addition, the computational domain must include several periods of the most unstable/least stable wavelike eigenmodes in the streamwise direction to adequately recover the physics. Furthermore, the spatial resolution must be adequate to recover the fine structure of the instability wave, especially in the surroundings of the separation bubble. Finally, here, three-dimensional disturbances are recovered by solution of a system of four coupled partial differential equations; this is in contrast to analogous earlier studies [7,8] which solved a system of three coupled partial differential equations (PDEs), thereby focusing on two-dimensional Tollmien–Schlichting waves alone. The second problem considered is the instability of the flow around a stalled NACA0015 airfoil. This problem is physically related to the previous one, but is more interesting from an industrial point of view. A coordinate transformation implemented for its solution [9] to represent this relatively complex geometry introduces new nonzero terms in the matrix discretizing the EVP, as will be discussed later. In this application too, a large domain is necessary to reduce the influence of the nonphysical boundary conditions imposed at the far-field boundary and, together with the nonlocalized structures appearing in the eigenmodes, extremely large resolutions are required.

Typically, these problems are addressed by using a time-stepping method for the numerical solution of the BiGlobal EVP [10–12]. Time-stepping approaches were devised at a time when storage of large matrices was impractical and work optimally for the recovery of a small number of leading eigenmodes. If additional modes are necessary (as, for example, in the case of transient growth analysis), a new time-stepping iteration, which excludes the modes already identified, is necessary.

Here, an alternative methodology is presented, which forms the discretized matrix and stores it over several processors of a computing cluster; using distributed-memory parallel computers, the maximum dimension of the problem which may be solved is thus determined by the number of processors available. The combination of distributed matrix formation and storage in conjunction with dense linear-algebra software is employed to the problem at hand for the first time.

The proposed solution relaxes both the memory restrictions associated with the matrix formation and the CPU time limitations imposed by a serial solution of the EVP. Linear-algebra operations are performed by the ScaLAPACK, BLACS, and PBLAS parallel, dense linear-algebra libraries [13]. These libraries are outgrowths of the well-known, well-tested LAPACK project and have been documented to work in a satisfactory manner with dense matrices of leading dimension $\mathcal{O}(10^6)$ achieving ~100 teraflops on ~20,000 processors [14]. Supercomputers with several thousand processors featuring the proposed linear-algebra libraries, such as Magerit[‡] or Mare Nostrum,[§] are becoming widely available and are increasingly deployed for the solution of large-scale scientific problems [15].

The paper is organized as follows: in Sec. II, the proposed methodology is presented, including information on the spatial discretization and the matrix distribution. Section III presents results obtained, both from a numerical and a physical point of view. Validation and verification work is presented in Sec. III.A, where the capabilities of parallelization are demonstrated. The main body of the scalability studies is presented in Sec. III.B, exclusively devoted to massive parallelization of the eigenvalue problem. The two physical instability problems which gave rise to devising of the present solution methodology are presented in Sec. III.C; in both problems monitored, the large resolutions employed have been instrumental for the success of the analysis. Conclusions and some discussion of alternatives to the methodology presented are discussed in Sec. IV.

II. Massively Parallel Eigenvalue Problem Solution Methodology

BiGlobal EVPs involve square matrices of large leading dimension resulting from the spatial discretization of four (five for the compressible case) coupled partial differential equations. Numerical solution of such problems is facilitated by numerical methods of a formal accuracy as high as possible, capable of minimizing the number of discretization nodes and thus keeping the memory requirements as low as possible. Spectral methods have such characteristics, although they come at the price of dense matrices, which make implementation of sparse solution techniques not straightforward. On the other hand, the coupled discretization of two spatial dimensions results in matrices with a certain degree of sparsity, even when using spectral methods. Here, only the treatment of the BiGlobal EVP problem using dense linear-algebra operations is considered. Experience with spectral methods for the solution of the BiGlobal EVP suggests that this numerical discretization methodology requires matrices of leading dimension $\mathcal{O}(10^4 \sim 10^5)$ for the coupled discretization of the two spatial directions. On the other hand, experience with both spectral and finite difference methods for one-dimensional (ordinary-differential-equation-based) stability problems [16,17] has delivered a rule of thumb for the number of nodes required by a spectral and a finite difference numerical method to obtain results of the same accuracy. This rule of thumb depends on the order of the finite difference discretization; use of a sixth-order compact finite difference scheme requires a factor four higher number of nodes compared with a spectral method of equivalent accuracy [18]. Extrapolation of such results to the (two-dimensional, partial-differential-equation-based) BiGlobal EVP suggests that a high-order finite difference approach would require discretized arrays the leading dimension of which would be at least 1 order of magnitude higher than that quoted previously; such arrays would only be able to be treated by sparse-matrix techniques. By contrast, spectral methods and dense-matrix algebra has been used presently, as follows.

A. Spatial Discretization

Spatial discretization is accomplished by Chebyshev–Gauss–Lobatto (CGL) points

$$\xi = \cos\frac{i\pi}{N}, \qquad i = 0, \ldots, N \tag{7}$$

and the correspondent derivative matrices $\mathcal{D} = \partial/\partial\xi$, $\mathcal{D}^{(2)} = \mathcal{D}\cdot\mathcal{D}, \ldots, \mathcal{D}^{(n)}$ [19]. The CGL points are mapped onto the domain of interest by coordinate transformations, which permit clustering of nodes in specific regions of the domain, like in boundary layers. Tensor products are used to form the derivative matrices for the present PDE-based problem. If $N_x + 1$ and $N_y + 1$ collocation

[‡]Data available at www.cesvima.upm.es.

[§]Data available at www.bsc.es.

points are used for the discretization of the x- and y-spatial directions, respectively, the discrete resulting differentiation matrices have a leading dimension of $(N_x + 1) \times (N_y + 1)$ and are obtained by applying the Kronecker product

$$\mathcal{D}_x = \mathcal{D} \otimes I, \qquad \mathcal{D}_y = I \otimes \mathcal{D} \tag{8}$$

where I is the identity matrix. Applying this spectral discretization to the linear problem in Eqs. (2–5) results in a nearly block-diagonal discretized matrix problem, shown in Fig. 1, with matrices of (global) leading dimension $\mathrm{GLD} = N_{\mathrm{var}} \times (N_x + 1) \times (N_y + 1)$, the factor N_{var} being equal to four in the incompressible case, arising from the four coupled equations, and five in the compressible case. The structure of the matrices, in which many elements are equal to zero, makes it possible to implement sparse techniques to drastically reduce the memory requirements. This possibility was tested in a previous research [20] using the parallel version of the library SuperLU, but the scalability was shown to be unsatisfactory. Such sparse techniques are not used here; instead, the problem is treated as one of dense matrices, which permits greater flexibility in introducing variations to the linear operators (i.e., different types of boundary conditions) without altering the data storing and solution algorithms. The linear operators describing the physics and the numerics and parallelization used to solve the problem are independent. In this manner, it is possible to solve both compressible and incompressible problems with an arbitrary combination of boundary conditions by making only minor changes in the core solution algorithm. The structure of the left-hand-side (LHS) matrices corresponding to both incompressible and compressible problems are shown in Fig. 1.

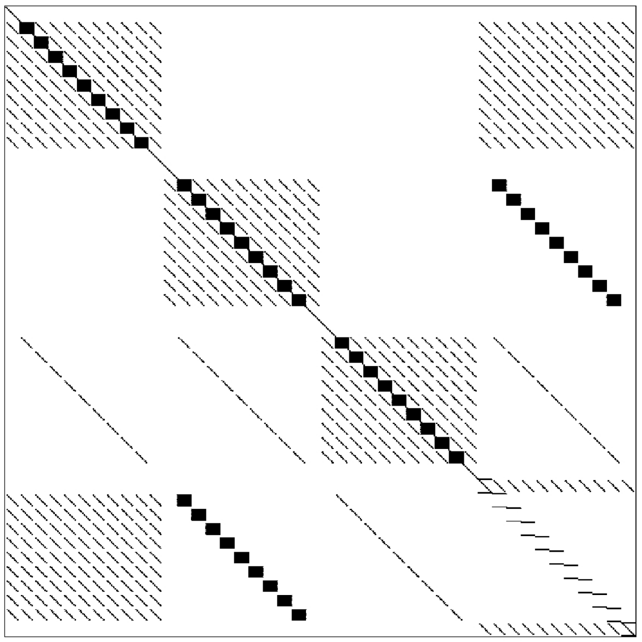
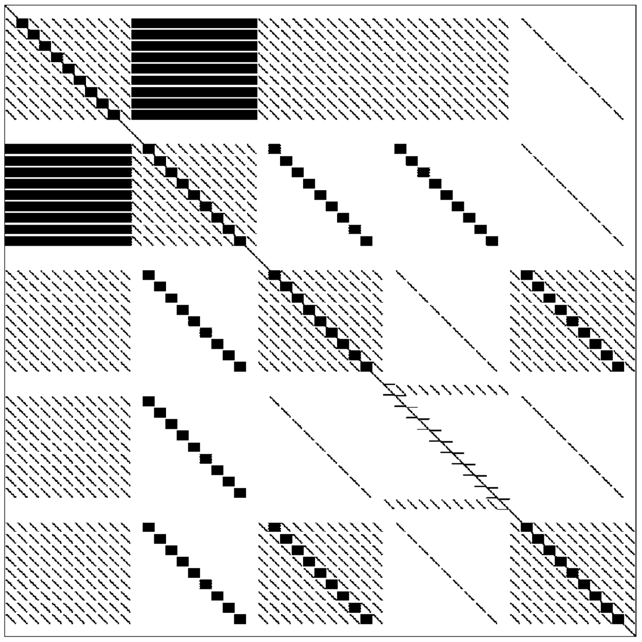



Fig. 1 Block-diagonal structure of the incompressible BiGlobal EVP left-hand side matrix in Eqs. (2–5) (top) and the equivalent compressible problem (bottom), when discretized using spectral collocation methods. For these matrices, $N_x = N_y = 10$ Chebyshev–Gauss–Lobatto points were used.

B. Arnoldi Algorithm

The large leading dimension of the complex matrices resulting from the discretization of the problem Eqs. (2–5) makes the application of the QZ algorithm (generalized Schur decomposition) [21] impossible. In contrast, Krylov subspace-based algorithms are used to recover efficiently the most interesting part of the eigenspectrum. A shift-and-invert variation of the Arnoldi algorithm [22] is employed here to transform the large EVP into a several orders-of-magnitude smaller problem (having a leading dimension equal to the Krylov subspace dimension, $m \sim 200$–2000). The QZ algorithm is used then to obtain the solution of the latter problem.

An Arnoldi algorithm is used; its two main tasks are a lower–upper (LU) decomposition of the large left-hand-side matrix in Eqs. (2–5) and a certain number of back substitutions equal to the dimension of the Krylov subspace generated. The first task accounts for most of the CPU time required in the serial solution, the latter methodology having been employed in the past in a satisfactory manner. However, resolution of physical phenomena involving steep gradients and/or several structures (e.g., Tollmien–Schlichting waves as amplitude functions of a BiGlobal eigenvector) give rise to the need for a step-change improvement. Such a borderline case has been encountered in the problem of instability of the laminar separation bubble at moderate Reynolds number [23]; using a number of discretizing points of $N_x = N_y = 60$ translates into a LHS matrix leading dimension $\mathcal{O}(15,000)$, requiring $\mathcal{O}(3.5)$ GB of in-core memory, and taking nearly 2 h of CPU time in a fast, shared-memory computer.

C. Data Distribution

Using ScaLAPACK, the parallelization is understood as a two-step approach. First, a virtual, rectangular processor grid is formed using all the processors available. Second, the arrays that store both matrices and vectors are distributed amongst the processor's grid following the block-cyclic algorithm [13]. In this second step, each array is divided into blocks, that is, small pieces of the arrays with a number of rows and columns given by an user-defined parameter called blocking factor (BF). ScaLAPACK permits using a different blocking factor for rows and columns, but according to the square matrices to be operated on, an unique value of BF was used for both. The distribution algorithm works independently on rows and columns. To illustrate the distribution, suppose an array of length GLD (with entries numbered from 1 to GLD) to be stored on p processors (numbered 0 through $p - 1$). The array is divided into blocks of size BF, except the last block which will contain GLD mod BF elements in the most general case. These blocks are numbered starting from zero and are distributed amongst the processors, so that the kth block is assigned to the processor of coordinate k mod p. The algorithm results in that the element IG (which is the index of the element on the global matrix) from the original global array maps to the element IL (which is the index of the element on the local matrix) of the local matrix assigned to the processor IP (which is the index of the processor on the processor's grid), where IL and IP are defined by

$$\mathrm{IL} = \mathrm{BF} \times \left(\frac{\mathrm{IG}-1}{\mathrm{BF} \times p}\right) + (IG - 1 \bmod \mathrm{BF}) + 1 \tag{9}$$

and

$$\mathrm{IP} = \left(\frac{\mathrm{IG}-1}{\mathrm{BF}}\right) \bmod p \tag{10}$$

More details on the distribution algorithm can be found in ScaLAPACK's documentation [13]. The arrays are stored in a balanced manner using the memory of all the processors available. The difference in the number of elements contained in the local arrays is, at most, equal to the blocking factor, which should be chosen to be orders-of-magnitude smaller than the global leading dimensions, as will be shown later. The main memory constraint in the solution of the BiGlobal EVP is the storage of one large matrix for the incompressible case (two for the compressible case), of leading dimension GLD. Other arrays also need to be stored, both in distributed or nondistributed manners, but its size is negligible compared to the large matrices. The minimum number of processors required for the solution of the problem is then estimated as

$$p \approx N_{\mathrm{arrays}} \times \mathrm{GLD}^2 \times \frac{\mathrm{mem}_d}{\mathrm{mem}_{\mathrm{proc}}} \tag{11}$$

where N_{arrays} is the number matrices to store, mem_d is the memory required for the storage in floating-point format of one complex matrix element (i.e., 16 B using double precision), and $\mathrm{mem}_{\mathrm{proc}}$ is the memory available for the computations per processor. Because of the almost diagonal structure of the right-hand-side matrix in the incompressible problem, only the storage of the left-hand-side matrix is required and $N_{\mathrm{arrays}} = 1$. Conversely, in the compressible case, the right-hand-side matrix is more involved, and the two matrices need to be stored ($N_{\mathrm{arrays}} = 2$). A small amount of memory is required for the other variables, the relative amount becoming smaller with increasing GLD. In any case, as will be discussed later, better performances are attained when a higher number of processors than the minimum is used.

III. Results

A. Verification and Validation

The problem of stability in a constant pressure-gradient driven rectangular duct [24,25] has been solved as a validation test. This problem has been chosen on the basis of two characteristics:

1) The required basic flow is recovered as the solution of the related Poisson problem

$$\nabla^2_{2D}\bar{w} = -2 \tag{12}$$

subject to homogeneous Dirichlet boundary conditions. The analytical solution of the problem is known and can be used both to check the convergence of the discretization and the performance of the parallel LU decomposition.

2) There is no transformation which permits converting the complex BiGlobal EVP into an equivalent EVP with real coefficients.

The physical instability in the rectangular duct flow is encountered as a consequence of increasing the aspect ratio Æ or increasing the Reynolds number at low finite values of Æ $\neq 1$[¶]; in either case, the resolution requirements increase beyond the maximum memory available on a typical serial machine. A case in which high resolution is required, and as such could be used to implement the distributed-memory techniques discussed herein, is the critical point of an aspect ratio Æ $= 5$ duct, $Re = 10,400$, $\beta = 0.91$ [25]. The convergence history is shown in Table 1.

In terms of the theme of the present paper, when the number of collocation points is less than 40×40, it is impossible to determine the value of the critical eigenvalue, due to the proximity of many unresolved eigenvalues. The convergence of the third decimal place

[¶]Flow is linearly stable in a square-duct configuration [24].

Table 1 Convergence history for the critical eigenvalue corresponding to the rectangular duct with Æ = 5 at $Re = 10,400$ and $\beta = 0.91$; reference frequency result $\omega_r = 0.21167$ [24,25] (leading dimension and required memory corresponding to the stored matrix are also shown)

N_x	N_y	GLD	Memory, GB	ω_r	ω_i
40	40	6724	0.67	0.20846870	1.229E–002
50	50	10,404	1.61	0.20983043	1.789E–003
60	60	14,884	3.30	0.21091032	1.299E–004
70	70	20,164	6.05	0.21138506	−2.221E–005
80	80	26,244	10.26	0.21146319	−2.687E–005
90	90	33,124	16.35	0.21147609	−2.037E–005
100	100	40,804	24.81	0.21147678	−1.982E–005
110	110	49,284	36.19	0.21147683	−1.972E–005

in ω_r is attained for a resolution of 70×70; this translates in 6 GB of in-core memory, more than that available on most serial computers. The convergence of the sixth decimal place is attained at a resolution of 110×110 nodes per amplitude function. This requires in excess of 36 GB and is clearly impossible to be handled by a typical serial machine.

B. Scalability and Massive Parallelization

The main objective of this work is to break the barrier in resolution imposed by the limited memory available on even the most powerful shared-memory machines. Distributed-memory parallelization makes it possible to store and compute with matrices whose size is only a function of the number of processors available, while drastically reducing the CPU time required for calculations. Three computing clusters have been used for the present work; their characteristics are summarized in Table 2. Aeolos is an own-local distributed-memory machine formed by 128 Myrinet interconnected xeon microprocessors, with own-compiled versions of BLACS and ScaLAPACK. At the other end, Mare Nostrum has been used; this machine is currently the number 13, top 500 supercomputer (at the time testing commenced, was at number 5), and is situated at the Barcelona Supercomputing Center and comprises 10240 IBM 970MP processors, interconnected by Myrinet and Gigabit Ethernet networks. Mare Nostrun features the IBM optimized version of ScaLAPACK, PESSL. Between the two cluster extremes, another local facility, Magerit, has been used. Magerit comprises 1200 eServer BladeJS20, each one with two power-PC microprocessors, interconnected by the Myrinet network, and also features PESSL, a machine-optimized version of ScaLAPACK.

A first scalability test was performed using the solution of Eq. (12), as it involves only the construction of a distributed matrix and the parallel LU decomposition. The local cluster Aeolos was used for these computations comprising different values of the blocking factor and matrix leading dimension. Two results are of significance here, one that the suggested [13] value of the blocking factor BF = 64 for both directions corresponds to the lowest wall time at all processor-grid configurations examined; as such, in subsequent computations, this parameter has been kept at its fixed optimal value. Secondly, and probably most significant, a near-perfect linear scaling is observed when the number of processors is increased, at all blocking-factor values. The latter result has been confirmed with solutions of both smaller and larger leading dimension matrices. The results of this first test are summarized in Fig. 2. The parallel solution using 16 processors reduces the computing time to less than an hour and the use of 64 processors reduces the time required for the serial solution by a factor of 36.

A second scalability test used Aeolos and Mare Nostrum, by solving the BiGlobal EVP of the instability of rectangular duct flow at $Re = 100$ and $\beta = 1$. The results obtained using different resolutions and number of processors are shown in the left part of Fig. 3. When the low (though perfectly adequate for convergence of the eigenmode) resolution 40×40 is used, the wall-time reduction with the number of processors is not significant, staying in the same order of magnitude of the serial solution. As resolution is increased, the theoretically constant CPU-time/number of processors ratio

Table 2 Characteristics of the distributed-memory machines used

Cluster	No. processors	Processors	Network	Library
Aeolos	128	Intel Xeon	Myrinet	Own-compiled ScaLAPACK
Magerit	2400	IBM PPC	Myrinet and Gigabit	PESSL
Mare Nostrum	10,240	IBM 970 MP	Myrinet and Gigabit	PESSL

$$T_p = \frac{t_{\text{CPU}}}{p} \tag{13}$$

is visible in the results: the time required to solve a 60×60 domain on four processors is almost 25 min; it is reduced to 12 min for eight processors and to 7 min on 16 processors. As mentioned earlier, solution of this problem was found to require almost 2 h CPU time on a serial machine. The same validation test was also solved on Mare Nostrum, using a number of processors from 64 to 1024. The results are shown in the right part of Fig. 3. The large, maximum number of processors used, 1024, makes it possible to increase resolution up to 256×256. The CPU time required when the 80×80 domain is solved always stays under 10 min, but scales poorly with the number of processors; wall-clock time is even found to increase when more than 512 processors are employed. When the resolution is increased, in this case to 128×128, the constant T_p scaling is recovered.

Conclusions drawn from this part of the work are as follows. As mentioned in the ScaLAPACK documentation, a workload balance is required for the code to scale satisfactorily, that is, so that the theoretically constant T_p is attained. If the local matrix, that is, the submatrix stored in each processor is too small, most of the wall time is spent on communication between processors; in that situation, the matrix is said to be overdistributed. On the other hand, if the local matrix is too large, most of the calculations take place inside each computer and better performance can be achieved by using a larger number of processors. A rule of thumb states that the dimension of the local matrices should be of size $\sim 1000 \times 1000$. The existence of an optimal number of processors to solve a given problem is evident, and this optimal value should be studied for each resolution. Typical academic BiGlobal EVPs require resolutions corresponding to a number of processors below 512; however, as this theory enters the realm of industrial applications, for which the Reynolds numbers involved are orders-of-magnitude higher than those of academic problems, resolutions comprising hundreds of points for each direction are required. A last conclusion concerns the wall-clock time versus number of processors, assuming a correct workload balance. The ideal constant value for the T_p is nearly accomplished when the

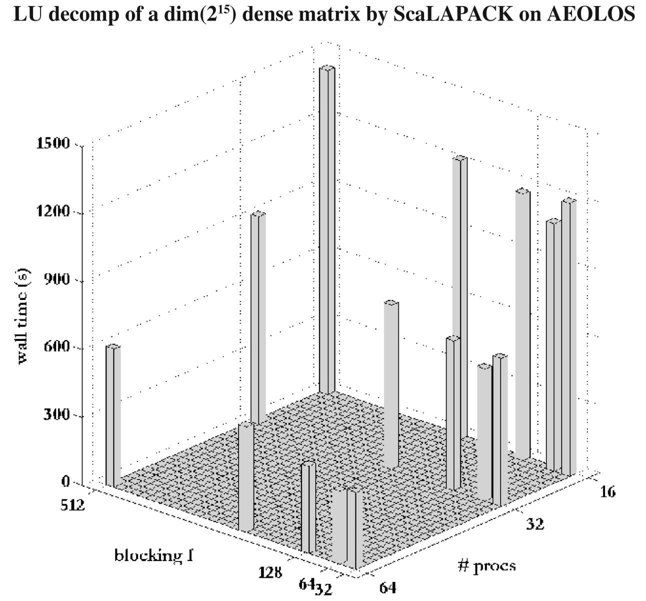


Fig. 2 Wall time for the parallel solution of the Poisson model problem as a function of the number of processors and the blocking factors used.

LU factorization occupies around half of total wall-clock time, the matrix generation costs about 20% of the total time, and all other tasks together account for the last 30%. However, when the number of processors is increased, effects other than the main parallel task, the LU decomposition, become increasingly more relevant.

It is therefore appropriate to turn attention to quantifying scalability under these conditions next, focusing on problems for which the dimensions are more representative of the current requirements of BiGlobal EVPs. In what follows, four aspects are studied: Sec. III.B.1 monitors the time required for the (manual) creation of the left-hand-side matrix in a distributed manner; Sec. III.B.2 deals with the LU decomposition of the EVP matrix, using ScaLAPACK; Sec. III.B.3 studies (in an average manner) the time devoted to the Arnoldi iteration, and Sec. III.B.4 is dedicated to the QZ subroutine of the Hessenberg matrix and the calculation of the eigenvectors.

1. *Eigenvalue Problem Generation*

In the previous section, the creation, in a distributed manner, of the large leading-dimension matrices describing the eigenvalue problem (2–5) was found to consume a considerable amount of time ($\sim$20% of the total). This is in contrast to the serial solution of the same problem, where this fraction of time is negligible. The origins of this result are to be found in the fact that the dimension of the matrices grows with the square of the resolution used. Concretely, the following operations are needed to generate the matrices: 1) double loop over the global matrix dimension: $t \sim (\text{GLD})^2$; 2) value assignment to the correspondent element and processor: $t \sim (\text{GLD})^2/p$; 3) certain number of loops over the matrix dimension: $t \sim \text{GLD}$.

When the leading dimension of the global matrix GLD is large, the time required for the EVP generation scales as

$$t_{\text{EVP}} \sim \left(1 + \frac{K}{p}\right) \cdot (\text{GLD})^2 \tag{14}$$

K being a constant that depends on the processor and communications speeds. Studies were conducted to evaluate how well this theoretical scaling is attained. The test problem was solved using a constant number of collocation points in the y direction ($N_y = 40$), a variable number of points in the x direction, and a different number of processors. The CPU time required for the creation of the global matrix was computed and the results are shown in the left part of Fig. 4. Thicker, dashed lines belong to Aeolos, whereas solid lines belong to Magerit. With minor deviations, the behavior of the CPU time is that described by Eq. (14). To isolate the first-order behavior, the same time is scaled with GLD and p,

$$\frac{t_{\text{EVP}} \cdot p}{\text{GLD}^2} \sim (p + K) + \frac{K_2 \cdot p}{\text{GLD}} \tag{15}$$

taking into account second-order effects, whose relative contribution is unknown a priori. This new variable is plotted against N_x and p in the right part of Fig. 4. The coincidence of the scaled data in the lower figure indicates that the constant K_2 is small enough to neglect its effect. This constant is related to the third operation stated before. The time consumed for a given resolution and number of processors is higher in Magerit that in Aeolos. As no library subroutine is used and there is no communication between processors in this task, this difference is attributed to a higher speed of the processors of Aeolos.

2. *Lower–Upper Decomposition of the Global Matrix*

The LU decomposition is the most time-consuming task in both the serial and parallel versions of the code; serially, it accounts for over $\sim$90% of the total CPU time, whereas this percentage drops to

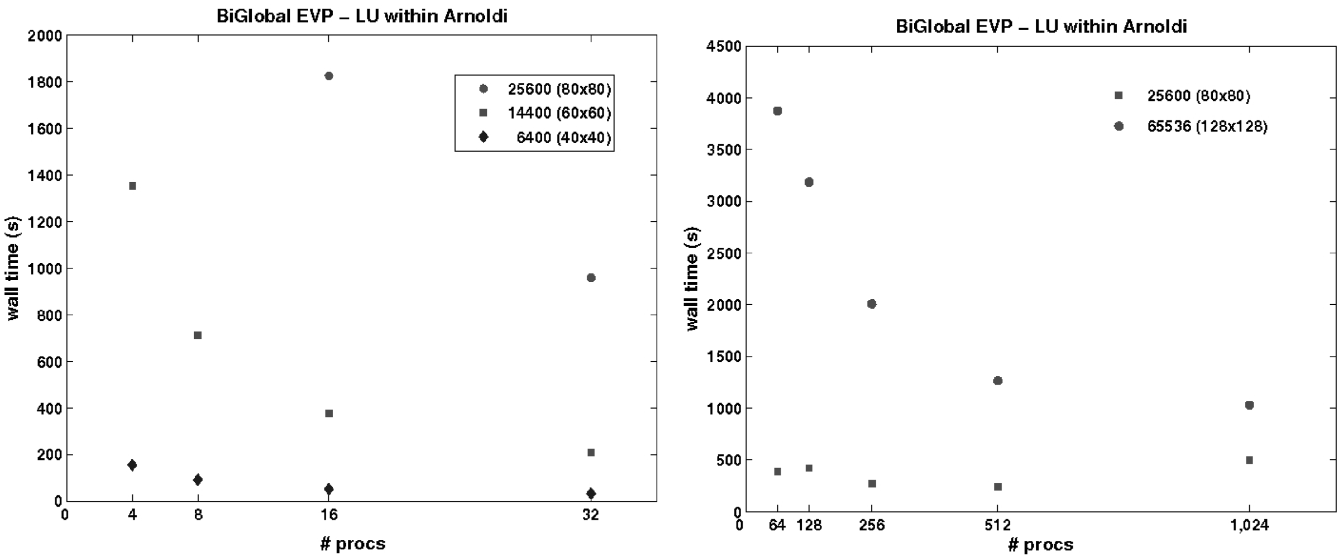


Fig. 3 Scalability tests performed on Aeolos (left) and Mare Nostrum (right). The CPU time required for the solution of the eigenvalue problem (LU factorization and Arnoldi algorithm) is plotted against the number of processors (procs) for various resolutions (in parenthesis). In brackets, the leading dimension of the LHS matrix is shown.

~50% in a typical parallel solution. Although efficiency of the code in this task depends entirely on the library software, fine tuning the discretization parameters in the parallel version of the code is essential to obtain optimal performances. The time required for shifting the matrix is also computed in this section but, as will be shown, this time is negligible compared to the time devoted to the LU decomposition. The operations required in this context are 1) loop over the matrix dimension: $t \sim \mathrm{GLD}$; 2) value assignment to the correspondent element and processor: $t \sim \mathrm{GLD}/p$; 3) LU decomposition, by call to the subroutine: $t \sim (\mathrm{GLD})^3/p$.

When the leading dimension of the matrix is large, the theoretical prediction for the scaling is

$$t_{\mathrm{LU}} \sim \frac{(\mathrm{GLD})^3}{p} \tag{16}$$

The same scalability tests as in the previous section were conducted, and results are shown in the left part of Fig. 5. The trends predicted from the first-order scaling Eq. (16) are reproduced in all cases. To recover second-order effects, the CPU time is scaled with GLD and p:

$$\frac{t_{\mathrm{LU}} \cdot p}{\mathrm{GLD}^3} \sim K + \frac{K_2 \cdot p + K_3}{\mathrm{GLD}^2} \tag{17}$$

where K, K_2, and K_3 are constants different to the ones defined earlier, but related to the same issues. The scaled times are plotted against GLD and p on the right side of Fig. 5. The relative importance of the LU decomposition is even increased over the shifting time when the dimension of the matrix grows, as the scaled time is reduced with increasing resolutions. The benefits of using the optimized library and massive parallelization are evident from the figures, because the scaled time decreases substantially from Aeolos data to Magerit, the latter platform affording substantially higher resolutions due to the larger number of processors available. The importance of the correct data parallelization may be observed on the lower-right plot. The documentation of ScaLAPACK suggests that the processor grid be as square as possible for performance to

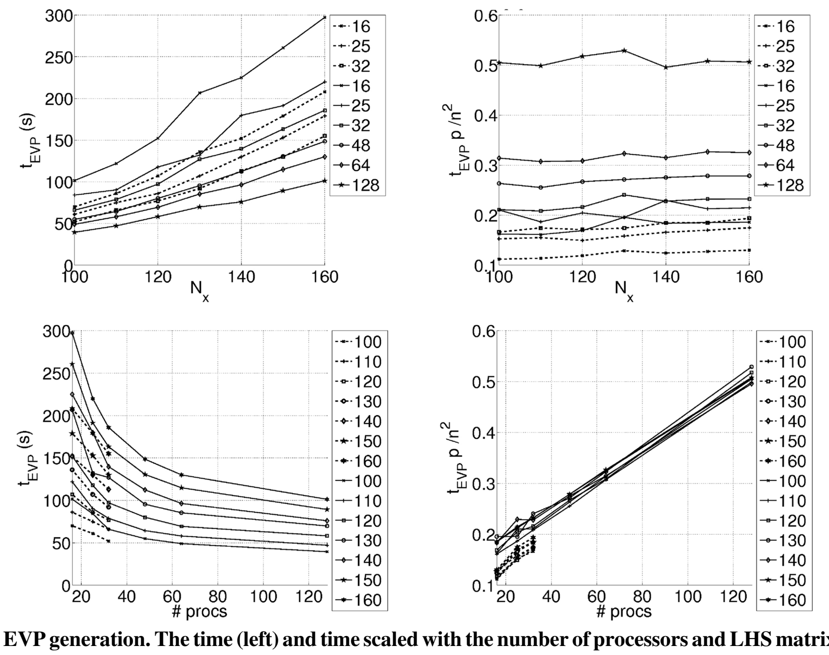


Fig. 4 CPU time scaling for EVP generation. The time (left) and time scaled with the number of processors and LHS matrix size (right) is plotted against the resolution (top) and number of processors (bottom). Thicker, dashed lines belong to Aeolos; solid lines belong to Magerit.

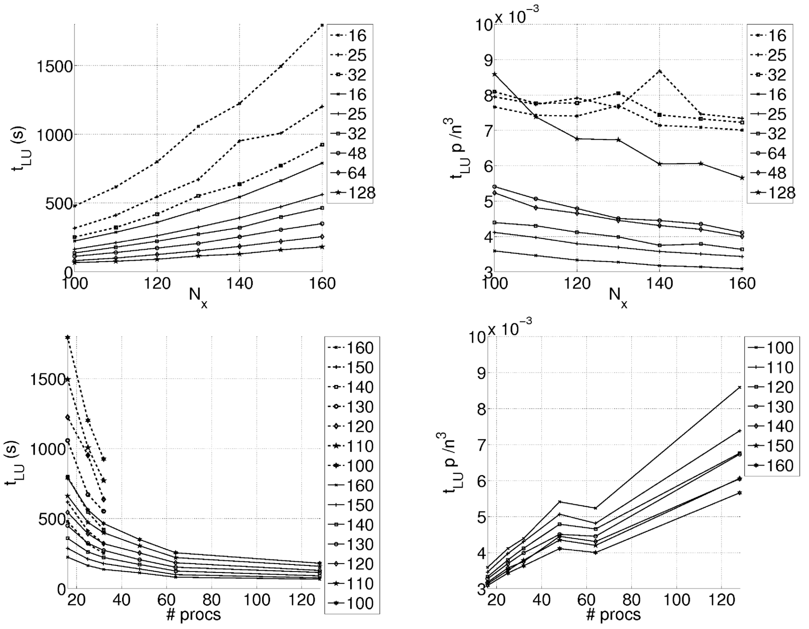


Fig. 5 CPU time scaling for shift and invert and LU decomposition. The time (left) and time scaled with the number of processors and LHS matrix size (right) is plotted against the resolution (top) and number of processors (bottom). Thicker, dashed lines belong to Aeolos; solid lines belong to Magerit.

be optimized. The processors' grids defined in this figure are as follows: 4×4 $(p = 16)$, 5×5 $(p = 25)$, 4×8 $(p = 32)$, 6×8 $(p = 48)$, 8×8 $(p = 64)$, and 8×16 $(p = 128)$. In the results shown in the lower-right part of Fig. 5, it can be seen that performance at the square processor grid 8×8 is much better than that on any other processor distribution.

3. *Arnoldi Iteration*

The Arnoldi algorithm generates a Krylov subspace whose dimension m is orders-of-magnitude smaller than GLD. Nevertheless, m must increase as the resolution increases, in line with the increase of the number of processors used. The time required for each Krylov iteration is generally small but, if the Krylov subspace dimension is high, the cumulative time cannot be neglected. Here, the time required for each iteration is averaged over $m = 200$ iterations. The different calculations performed within each Krylov iteration require time consumption in several places, of which only the more relevant are taken into account next: 1) loop over the global matrix dimension: $t \sim \mathrm{GLD}$; 2) value assignment to the correspondent element and processor: $t \sim \mathrm{GLD}/p$; 3) backsubstitution on the LU decomposed matrix: $t \sim (\mathrm{GLD})^2/p$.

When the leading dimension of the matrices is large, the leading-order effect on time is

$$t_{\mathrm{Ar,it}} \sim \frac{\mathrm{GLD}^2}{p} \tag{18}$$

The results of the scalability tests are shown in the left part of Fig. 6. The time consumed increases with increasing resolution, but almost linearly rather than the expected quadratic trend. The

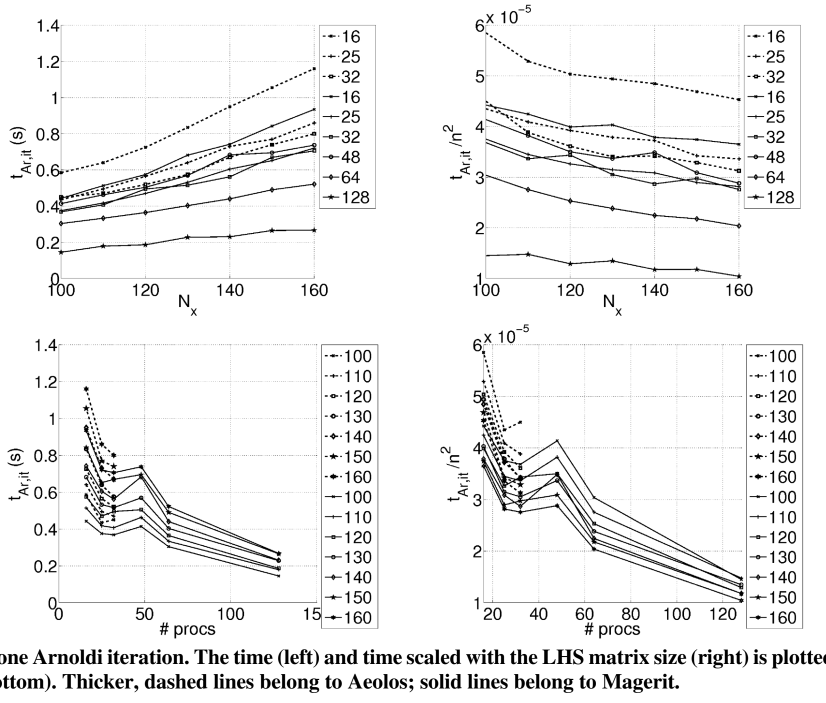


Fig. 6 CPU time scaling for one Arnoldi iteration. The time (left) and time scaled with the LHS matrix size (right) is plotted against the resolution (top) and number of processors (bottom). Thicker, dashed lines belong to Aeolos; solid lines belong to Magerit.

variation with the number of processors also shows the predicted trend, but has an important deviation for a given (i.e., $p = 48$) processor grid. The conclusion drawn is that what was assumed to be second-order effects are more significant than implied by the scaling Eq. (18). The alternative time scaling is then constructed:

$$\frac{t_{\mathrm{Ar,it}}}{\mathrm{GLD}^2} \sim \frac{K}{p} + \left(K_2 + \frac{K_3}{p}\right) \cdot \frac{1}{\mathrm{GLD}} \tag{19}$$

Here, the time required for tasks 1 and 2, supposed negligible in Eq. (18), has a more pronounced effect and increases the efficiency as the resolution becomes higher, as can be seen in upper-right part of Fig. 6. The lower-right part of the same figure shows that, although the scaled time versus number of processors is correctly described by Eq. (18), the relation to the shape of the processor's grid is not fully understood. The most plausible explanation relates the increase in time to one or more tasks, severely affected when the processor's grid is not square. The ScaLAPACK suggestion about the shape of the processor's grid has been taken into account here, but the exact cause in the performance degradation in the backsubstitution task is unclear. The fact that better performance is obtained in Magerit suggests use of the optimized library version as a possible explanation.

4. *Eigenvalues and Eigenvectors Calculation*

Once the Krylov subspace and Hessenberg matrix have been generated, most of the tasks are performed in each processor in a serial manner, with no communication between processors. Each processor stores a copy of the Hessenberg matrix and calls the QZ subroutine independently; although this approach generates large redundancy, it is faster than computing the eigenvalues in one processor only and communicating the data between a large number of processors. On the other hand, once the eigenvectors from the Hessenberg are computed, they must be distributed to compute the Ritz vectors by forming their product with the Krylov subspace base. This combination of serial (Hessenberg matrix eigensystem computation) and parallel (matrix–matrix product) tasks makes it difficult to estimate scalings, but much of the workload is distributing data over the processors, and so it is expected that increasing the number of processors will increase the CPU time. The major time-consuming tasks are as follows: 1) double loop over the Hessenberg matrix dimension in each processor: $t \sim (m)^2 \cdot p$; 2) product of distributed matrices: $t \sim (m)^2 \cdot \mathrm{GLD}/p$; 3) loop over the global matrix dimension in each processor: $t \sim \mathrm{GLD} \cdot p$.

Little can be said about the relative importance of each contribution at this point. As the resolution and Krylov subspace dimension should grow together for approximately the same fraction of the eigenspectrum to be computed, an increment in resolution will result in an increment of CPU time. On the other hand, and contrary to what happens in other tasks, the required time grows linearly with the number of processors if p is high enough. This theoretical behavior has been recovered in the numerical experimentation performed, as is shown in the left part of Fig. 7. Time was scaled with the global matrix leading dimension, as the dimension of the Hessenberg matrix was kept constant in all tests:

$$\frac{t_{\mathrm{EIG}}}{\mathrm{GLD}} \sim K + p \cdot \left(K_2 + \frac{K_3}{\mathrm{GLD}}\right) \tag{20}$$

The dependence of this scaled time with the resolution and number of processors is plotted in the right part of Fig. 7. There is no variation of the scaled time with the resolution, and so the term multiplied by K_3 is neglected. The number of processors has an important effect on the efficiency of the code, as an increase in p drastically increases the time t_{EIG}. In this respect Aeolos was found to be much more efficient than Magerit in performing this task, the time being almost independent of the number of processors, probably due to better internal communications between the processors in the former, as opposed to the latter platform. A summary of our findings on the relative time required for each one of the main parallelization tasks is shown in Fig. 8. When the number of processors is fixed, the relative importance of the LU-decomposition time increases with the resolution, reducing the corresponding contribution of Arnoldi iterations and the eigenvalues and eigenvectors computation. When the resolution is fixed, the time required for the computation of eigenvalues and eigenvectors grows notably, being the factor which imposes an optimal number of processors. The time required for the EVP generation is around one-fifth of the total CPU time.

C. Applications to Separated Flow Instability

The present methodology enables use of high resolutions as permitted by the number of processors available; the nearly linear scaling demonstrated implies that the CPU time necessary for the recovery of a given window of eigenvalues is an inverse linear function of the number of processors used. This approach has permitted the study of problems out of reach of previously available methodologies of the same class. Two examples are shortly exposed, one representative of open problems in which the convective nature

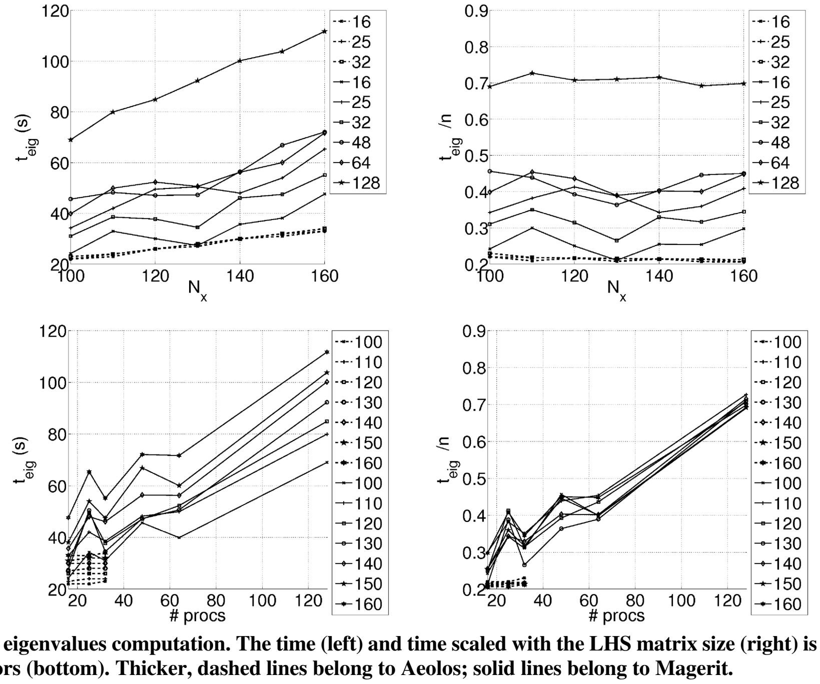


Fig. 7 CPU time scaling for eigenvalues computation. The time (left) and time scaled with the LHS matrix size (right) is plotted against the resolution (top) and number of processors (bottom). Thicker, dashed lines belong to Aeolos; solid lines belong to Magerit.

of the dominant instability dictates use of large resolutions, and a second application, closer to industrial interests, in which, in addition to the previous considerations, a relatively complex geometry has to be dealt with.

1. *Instability of Laminar Separation Bubble in a Flat-Plate Boundary Layer*

This application has been discussed in detail elsewhere [26]. The basic flow corresponds to a flat-plate boundary layer with Reynolds number based on the displacement thickness at the inflow equal to 450 at inflow and 700 at outflow, and a separation bubble with peak reversed flow $\sim 1\%$ of the far-field velocity. This two-dimensional flow is analyzed with respect to its three-dimensional global instability within the entire range of spanwise wave numbers $\beta \in [0, \infty)$; in practice, a large number of eigenvalue problems, each corresponding to a given discrete value of β, are analyzed. The entrance of Tollmien–Schlichting waves through the inflow is permitted by imposing the Robin boundary condition

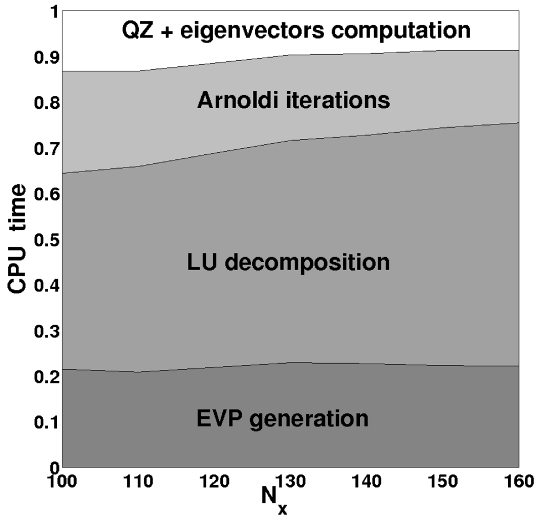
$$\frac{\partial \hat{\mathbf{q}}}{\partial x} = \pm i\alpha \hat{\mathbf{q}}, \quad \text{where } \alpha \approx \alpha_0 + c_0(\omega - \omega_0) \tag{21}$$

The dispersion relation $\mathcal{D}(\alpha, \omega, \beta) = 0$ obtained from local analysis is used to evaluate the wave number α_0 and the group velocity $c_0 = \partial\alpha/\partial\omega$ at some fixed frequency ω_0. The two signs permitted in Eq. (21) avoid the symmetry of the problem with respect to ω. A wave with $\alpha > 0$ propagates downstream when $\omega > 0$, and upstream when $\omega < 0$. Leaving apart the physical meaning of an upstream moving convective wavelike disturbance, the eigenspectrum is no longer symmetric, as is shown in Fig. 9. Furthermore, this lack of symmetry prohibits the reduction of the problem to an EVP with real coefficients, which in turn would result in saving half of the computational memory.

A large window of eigenvalues, that is, a large Krylov subspace ~ 2000, is required to recover accurately the discretized branches of eigenvalues corresponding to the wavelike disturbances. To converge the window of eigenspectrum corresponding to the most unstable/least stable branch of wavelike eigenmodes, a resolution of $N_x \times N_y = 360 \times 64$ collocation points has been used; this required approximately 120 min of wall time on 144 processors, that is, a total of 300 CPU hours on Mare Nostrum. The reconstruction of the flowfield composed of the linear superposition of the dominant wavelike eigenmode at $\beta = 0.15$ upon the basic flow is also shown in Fig. 9.

2. *Massive Separation on a NACA0015 Airfoil*

This application has also been discussed extensively elsewhere [9]; in this reference, Kitsios et al. provide details on the analytical coordinate transformation used. The instability of the flow around a stalled NACA0015 airfoil has been monitored, which is a problem physically related to that of the laminar separation bubble on a flat plate, but with a higher industrial interest. Here, a large domain is necessary to reduce the influence of the artificial boundary conditions on the analysis results. Convergence of the solution obtained at each of the present BiGlobal eigenvalue problems at chord-based Reynolds number 200 and angle of attack equal to 18 deg required $N_x \times N_y = 250 \times 250$ collocation points. The memory requirements of this stored matrix approached 1 TeraB of memory, distributed over 1024 processors; the computation of the leading part of the spectrum required 22 h of wall time on Mare Nostrum. This is, to the authors best knowledge, the largest EVP solved to date using the present methodology.



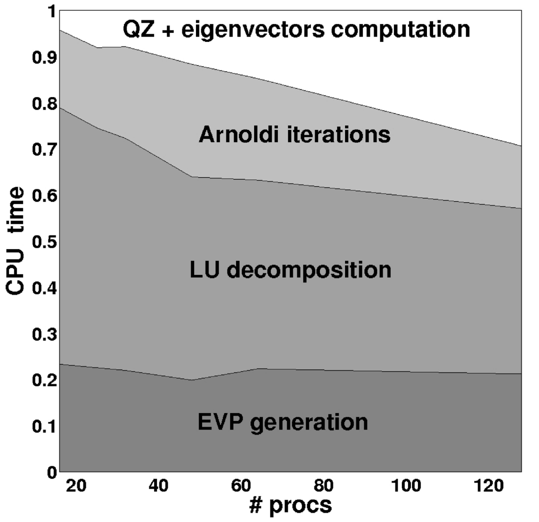


Fig. 8 Relative contribution to total CPU time from each main task, against resolution (top) and number of processors (bottom). The values are averaged in number of processors and resolution, respectively. The test case solves the complex BiGlobal EVP by constructing a Krylov subspace of dimension $m = 200$, in which 65 eigenvalues and Ritz vectors are computed.

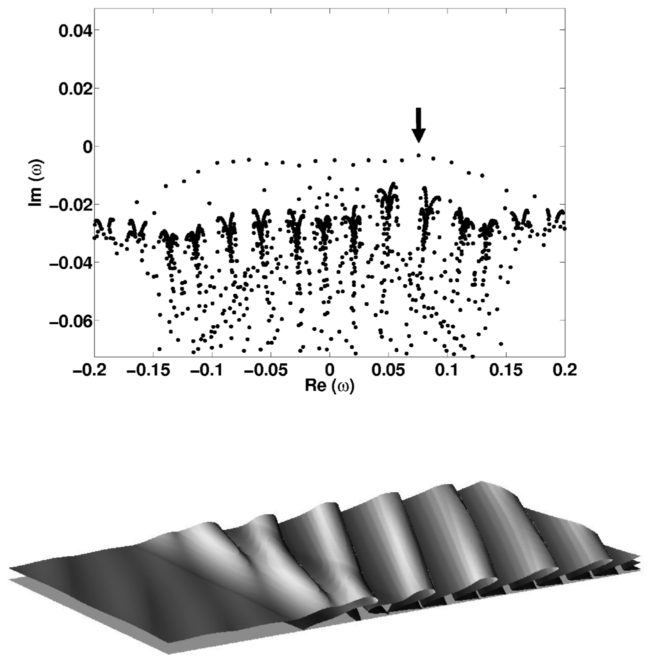
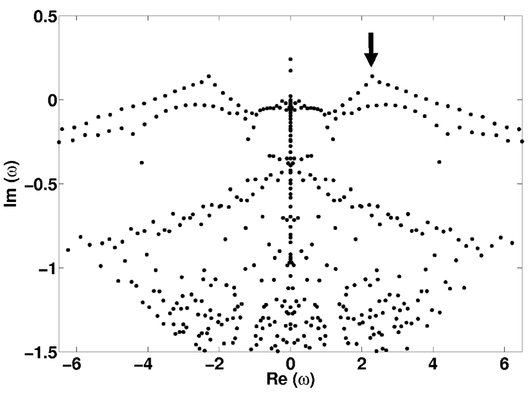


Fig. 9 Eigenspectum corresponding to a laminar separation bubble model on a flat-plate boundary layer at $Re_{\delta*} = 450$ and $\beta = 0.15$ (top). Three-dimensional flow reconstruction superposing linearly the least stable Tollmien–Schlichting eigenmode to the basic flow (bottom).



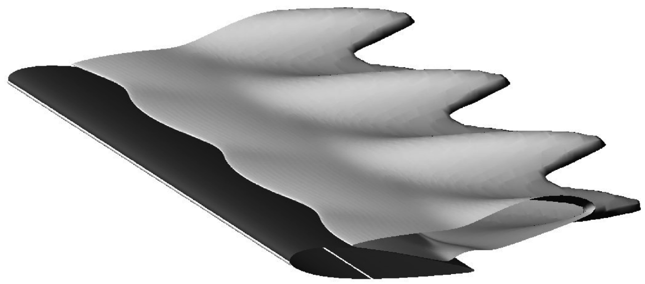


Fig. 10 Eigenspectum corresponding to NACA0015 airfoil at $Re = 200$ and $\beta = 1$ (top). Three-dimensional flow reconstruction superposing linearly the least stable eigenmode to the basic flow (bottom).

A representative eigenspectrum and a flow reconstruction using the dominant wavelike eigenmode at $\beta = 1$ is shown in Fig. 10. Note that, in contrast with the previous problem, the eigenspectrum is now symmetric, in accordance with the fact that the boundary conditions imposed do not alter the symmetry with respect to ω. The instability properties of the two-dimensional basic flow were shown to be responsible for the three-dimensionalization of the flowfield. In the snapshot on the figure, the resultant three-dimensional reversed-flow region is apparent.

IV. Conclusions

A parallel code has been developed for the solution of large partial-differential-equation eigenvalue problems resulting from the BiGlobal instability theory. This code employs the dense, parallel linear-algebra library ScaLAPACK, and distributed-memory machines to circumvent the restrictions in memory and CPU time imposed by solution of this problem on serial shared-memory platforms. A hand-coded parallel version of the Arnoldi algorithm has been developed, allowing for flexibility, and a detailed study of the different parallelization aspects has been performed. Although parallelization of the matrix creation and its LU decomposition has been straightforward, the parallel implementation of the Arnoldi algorithm has been rather challenging, especially in terms of scalability. A solution based on a combination of distributed global and nondistributed (and then repeated on each processor) arrays has been proposed. Verification and validation of the algorithm was provided at low resolutions, by reference to well-studied model problems. Convergence of results was obtained at high resolutions, unattainable on serial machines.

Massive parallelization studies have been carried out, using up to 1024 processors, on three different platforms. The comparison between the wall-clock times indicates that the performance obtained by the optimized versions of ScaLAPACK is an order of magnitude better than that offered by the free versions of this library. The existence of an optimal number of processors at each resolution has been documented and the effect of using an appropriately constructed processor grid has been demonstrated. At the high end of the number of processors used, the detrimental effect of increasing the number of processors beyond this optimal has been shown, with an increase of wall-clock time resulting from overdistribution of matrices. At the other extreme of the number of processors, if the minimum number of processors is used to store the global matrix, the resulting wall time is prohibitive for practical applications. A systematic study of the scaling of time with resolution and number of processors has been completed, providing qualitative and quantitative predictions for the wall time required; this is expected to be useful in future studies devoted to physical aspects of BiGlobal flow instability.

The present methodology has been employed to analyze physical problems in which large resolutions are instrumental for the success of the analysis. Concretely, instability results of three-dimensional wavelike disturbances of reversed-flow configurations, namely a laminar separation bubble on a flat plate and massive separation on a NACA0015 airfoil, were presented as examples of the applications, the global instability of which may be addressed successfully by the proposed enabling technology. Nevertheless, the rather large computing resources required for the analysis of flow instability in realistic geometries (especially when a large part of the eigenspectrum must be recovered) points at the need for investigation into alternative approaches for the eigenspectrum computation; the results of such efforts will be presented elsewhere.

Acknowledgments

The material is based on research sponsored by the U.S. Air Force Office of Scientific Research, Air Force Material Command, under agreement number FA8655-06-1-3066 to Nu modelling s.l., entitled *Global Instabilities in Laminar Separation Bubbles*. The grant is monitored by Douglas Smith (originally by Rhett Jefferies) of the U.S. Air Force Office of Scientific Research and S. Surampudi of the European Office of Aerospace Research and Development. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Air Force Office of Scientific Research or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Computations have been performed on the CeSViMa and MareNostrum facilities.

References

[1] Drazin, P. G., and Reid, W. H., *Hydrodynamic Stability*, Cambridge Univ. Press, Cambridge, England, U.K., 1981.

[2] Schmid, P., and Henningson, D. S., *Stability and Transition in Shear Flows*, Springer, New York, 2001.

[3] Theofilis, V., “Advances in Global Linear Instability Analysis of Nonparallel and Three-Dimensional Flows,” *Progress in Aerospace Sciences*, Vol. 39, No. 4, 2003, pp. 249–315. doi:10.1016/S0376-0421(02)00030-1

[4] Theofilis, V., and Colonius, T., “Three-Dimensional Instabilities of Compressible Flow over Open Cavities: Direct Solution of the BiGlobal Eigenvalue Problem,” AIAA Paper 2004-2544, 2004.

[5] Robinet, J.-C., “Bifurcations in Shock-Wave/Laminar-Boundary-Layer Interaction: Global Instability Approach,” *Journal of Fluid Mechanics*, Vol. 579, May 2007, pp. 85–112. doi:10.1017/S0022112007005095

[6] Theofilis, V., Hein, S., and Dallmann, U., “On the Origins of Unsteadiness and Three-Dimensionality in a Laminar Separation Bubble,” *Philosophical Transactions of the Royal Society of London, Series A: Mathematical and Physical Sciences*, Vol. 358, No. 1777, 2000, pp. 3229–3324. doi:10.1098/rsta.2000.0706

[7] Ehrenstein, U., and Gallaire, F., “On Two-Dimensional Temporal Modes in Spatially Evolving Open Flows: The Flat-Plate Boundary Layer,” *Journal of Fluid Mechanics*, Vol. 536, Aug. 2005, pp. 209–218. doi:10.1017/S0022112005005112

[8] Åkervik, E., Ehrenstein, U., Gallaire, F., and Henningson, D., “Global Two-Dimensional Stability Measures of the Flat-Plate Boundary-Layer Flow,” *European Journal of Mechanics, B: Fluids*, Vol. 27, No. 5, 2008, pp. 501–513. doi:10.1016/j.euromechflu.2007.09.004

[9] Kitsios, V., Rodríguez, D., Theofilis, V., Ooi, A., and Soria, J.,

"BiGlobal Instability Analysis of Turbulent Flow over an Airfoil at an Angle of Attack," AIAA Paper 2008–4384, 2008.

[10] Edwards, W. S., Tuckerman, L. S., Friesner, R. A., and Sorensen, D. C., "Krylov Methods for the Incompressible Navier–Stokes Equations," *Journal of Computational Physics*, Vol. 110, No. 1, 1994, pp. 82–102. doi:10.1006/jcph.1994.1007

[11] Barkley, D., Gomes, M. G. M., and Henderson, R. D., "Three-Dimensional Instability in a Flow over a Backward-Facing Step," *Journal of Fluid Mechanics*, Vol. 473, Dec. 2002, pp. 167–190.

[12] Bagheri, S., Åkervik, E., Brandt, L., and Henningson, D. S., "Matrix-Free Methods for the Stability and Control of Boundary Layers," *AIAA Journal*, Vol. 47, No. 5, 2009, pp. 1057–1068. doi:10.2514/1.41365

[13] Blackford, L. S., Choi, J., Cleary, A., E. D'Azeuedo, J. Demmel, and I. Dhillon et al., ScaLAPACK User's Guide, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1997, ISBN 0-89871-397-8. Petitet, A., Whaley, R. C., Demmel, J., Dhillon, I., Stanley, K., Dongarra, J., Hammarling, S., Henry, G., and Walker, D., "ScaLAPACK: A Portable Linear Algebra Library for Distributed Memory Computers: Design Issues and Performance," 1996.

[14] Borrill, J., "MADCAP: The Microwave Anositropy Dataset Computational Analysis Package," *Proceedings of the 5th European SGI/Cray MPP Workshop*, 1999, http://arxiv.org/abs/astro-ph/9911389v1.

[15] Bonoli, P., Batchelor, D., Berry, L., Choi, M., D'Ippolito, D. A., Harvey, R. W., Jaeger, E. F., Myra, J. R., Phillips, C. K., Smithe, D. N., Tang, V., Valeo, E., Wright, J. C., Brambilla, M., Bilato, R., Lancellotti, V., and Maggiora, R., "Evolution of Nonthermal Particle Distributions in Radio Frequency Heating of Fusion Plasmas," *Journal of Physics: Conference Series*, Vol. 78, 2007, p. 012006. doi:10.1088/1742-6596/78/1/012006

[16] Macaraeg, M., Streett, C., and Hussaini, M., A Spectral Collocation Solution to the Compressible Stability Eigenvalue Problem, NASA, TR TP-2858, 1988.

[17] Malik, M. R., "Numerical Methods for Hypersonic Boundary Layer Stability," *Journal of Computational Physics*, Vol. 86, No. 2, 1990, pp. 376–413. doi:10.1016/0021-9991(90)90106-B

[18] Theofilis, V., *Journal of Engineering Mathematics*, Vol. 34, Nos. 1–21998, pp. 111–129. doi:10.1023/A:1004366529352

[19] Canuto, C., Hussaini, M. Y., Quarteroni, A., and Zang, T. A., *Spectral Methods: Fundamentals in Single Domains*, Springer, New York, 2006.

[20] de Vicente, J., Valero, E., and Theofilis, V., "Numerical Considerations in Spectral Multidomain Methods for BiGlobal Instability Analysis of Open Cavity Configurations," *Progress in Industrial Mathematics at ECMI 2006*, Springer, New York, 2006, ISBN 9783540719922.

[21] Golub, G. H., and Van Loan, C. F., *Matrix Computations*, 2nd ed., John Hopkins Univ. Press, Baltimore, MD, 1989, ISBN 0-8018-3739-1.

[22] Saad, Y., "Variations of Arnoldi's Method for Computing Eigenelements of Large Unsymmetric Matrices," *Linear Algebra and Its Applications*, Vol. 34, No. 1, 1980, pp. 269–295. doi:10.1016/0024-3795(80)90169-X

[23] Theofilis, V., "On Instability Properties of Incompressible Laminar Separation Bubbles on a Flat Plate Prior to Shedding," AIAA Paper 2007-0540, 2007.

[24] Tatsumi, T., and Yoshimura, T., "Stability of the Laminar Flow in a Rectangular Duct," *Journal of Fluid Mechanics*, Vol. 212, No. 1, 1990, pp. 437–449. doi:10.1017/S002211209000204X

[25] Theofilis, V., Duck, P. W., and Owen, J., "Viscous Linear Stability Analysis of Rectangular Duct and Cavity Flows," *Journal of Fluid Mechanics*, Vol. 505, April 2004, pp. 249–286. doi:10.1017/S002211200400850X

[26] Rodríguez, D., and Theofilis, V., "On Instability and Structural Sensitivity of Incompressible Laminar Separation Bubbles in a Flat-Plate Boundary Layer," AIAA Paper 2008–4148, 2008.

A. Tumin
Associate Editor