CHAPTER

3

Spectral element methods for turbulence

Paul F. Fischer[a] *and Ananias G. Tomboulides*[b]

[a]Department of Computer Science, and Department of Mechanical Science and Engineering, University of Illinois, Urbana-Champaign, Urbana, IL, United States, [b]School of Mechanical Engineering, Aristotle University of Thessaloniki, Thessaloniki, Greece

3.1 Introduction

The spectral element method (SEM) is a natural extension of spectral methods in several ways. The principal objective is to retain the rapid convergence of spectral methods, resulting in minimal numerical dispersion and dissipation, while efficiently accommodating more complex domains. The importance of high-order spatial discretizations for turbulent flow simulations was established in Chapter 1. With increasing Reynolds number, turbulent flows exhibit a larger range of scales, where small-scale features of size η must be transported over relatively long distances $\mathcal{L} \gg \eta$. Kreiss and Oliger [1] noted early on that simulation of advection-dominated (i.e., turbulent) flows favor high-order discretizations in order to control dispersion errors that accumulate with increasing scale separation, $\mathcal{L}/\eta$.

The accuracy of the SEM derives from the exponential rate of convergence of polynomial approximations, which is realized for solutions having modest smoothness requirements, and from the use of stable Lagrangian interpolants based on the Gauss–Lobatto–Legendre (GLL) quadrature points. Unlike high-order finite differences, the SEM (and FEM) uses compactly-supported polynomial basis functions to represent the solution on subsets (i.e., elements) of the domain, which avoids the need for special boundary treatment where otherwise wide stencils would

Numerical Methods in Turbulence Simulation
https://doi.org/10.1016/B978-0-32-391144-3.00009-7

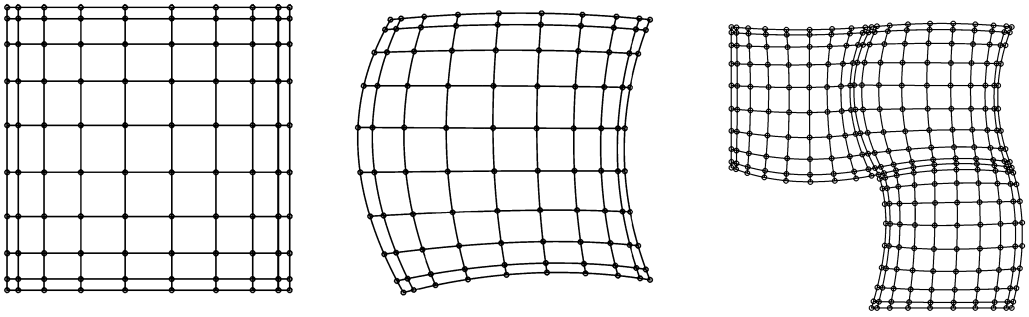

FIGURE 3.1 Illustration of 2D isoparametric mapping from $\hat{\Omega} := [-1, 1]^2$ (left) to deformed domain Ω (center) and to a collection of such subdomains (i.e., as in used in the SEM), $\Omega := \bigcup_{e=1}^{E} \Omega^e$ (right). The grids indicate tensor-products of $(N+1)$ Gauss–Lobatto–Legendre points mapped to respective subdomains according to (3.1) for $N = 9$.

extend outside the domain. Fig. 3.1 illustrates the basic geometric considerations for the SEM, which include a single (i.e., spectral) domain on the left, for which many (constant-coefficient) operators are separable, a deformed domain in the center, and a multidomain (i.e., spectral element) configuration on the right.

Orszag [2] established a key breakthrough in the development of the SEM when he demonstrated that spectral domains of moderate complexity could be effectively implemented through an isoparametric mapping of the form

$$\mathbf{x}(r, s) = \sum_{j=0}^{N} \sum_{i=0}^{N} \mathbf{x}_{ij}\, l_i(r)\, l_j(s), \tag{3.1}$$

which maps $\mathbf{r} := (r, s) \in \hat{\Omega} := [-1, 1]^2$ to a deformed domain Ω, as illustrated in Fig. 3.1 center. Here, $\{l_i(r)\}$ is a suitable basis for $\mathbb{P}_N(\hat{\Omega})$, the space of polynomials of degree $\leq N$ in $\hat{\Omega}$. In the SEM, (3.1) would represent the geometry for a single element, Ω^e, and would be denoted as $\mathbf{x}^e$. The local SEM basis functions are chosen to be Lagrange cardinal polynomials satisfying $l_i(\xi_j) = \delta_{ij}$, where the nodes $\{\xi_j\}_{j=0}^{N}$ are the Gauss–Lobatto–Legendre (GLL) quadrature points on $[-1, 1]$ and δ_{ij} is the Kronecker delta function. The solution on each element is represented in a form similar to (3.1),

$$u(r, s) = \sum_{j=0}^{N} \sum_{i=0}^{N} u_{ij}\, l_i(r)\, l_j(s), \tag{3.2}$$

where $\underline{u} := \{u_{ij}\}$ represents the vector of unknown basis coefficients. With the choice of Lagrangian bases, the u_{ij} also represent nodal values.[1]

[1] Although Orszag's original exposition focused on expansions in Chebyshev polynomials, the Lagrange polynomials based on the GLL points provide equivalent convergence and sta-

Orszag's key observation was that, for the deformed single-domain case, *forward* operator evaluation could be evaluated in only $O(N^4)$ work and $O(N^3)$ storage for problems in 3D, where the number of unknowns is $n \approx N^3$ [2]. This economy derives from tensor-product-sum factorization, the same that is used for standard spectral methods, save that in the latter case one recovers $O(N^3 \log N)$ complexity owing to the use of fast transforms for differentiation or interpolation. To exploit this fast (matrix-free) operator evaluation, one needs to resort to iterative methods to affect the *inverse* of the operators, and these need to be preconditioned. While there are today a multitude of preconditioning strategies, as we describe shortly, Orszag [2] introduced an effective preconditioning approach in which the spectral operator is approximated by a sparse low-order operator, resulting in order-independent condition numbers and thus reducing the solution of a high-order problem to that of solving a low-order one.

General-domain spectral methods were originally based on collocation, using orthogonal polynomial bases in the reference domain (e.g., Fig. 3.1, left). As noted in Chapter 2, expansions based on eigenfunctions of *singular* Sturm–Liouville problems are critical to rapid convergence as such expansions alleviate boundary constraints placed on the solution and data [3]. For example, in the case of the 1D heat equation, $-u_{xx} = f$ with $f(x)$ analytic, expansions in terms of the eigenfunctions (sines and cosines) converge exponentially fast only if u and f are periodic on Ω. If $u(x)$ has Dirichlet conditions, the eigenfunctions are still sines and cosines, but convergence is only algebraic (e.g., $O(N^{-3})$) in the number of basis functions used in the truncated expansion. By contrast, expansions in Legendre or Chebyshev polynomials will yield exponential (i.e., $O(e^{-\sigma N})$, $\sigma > 0$) convergence for this problem.

With support for general (i.e., aperiodic) boundary conditions and geometric deformation, it is also possible to consider unstructured, multidomain spectral methods, as illustrated in Fig. 3.1, right. Early multidomain spectral methods were based on collocation [4]. A significant advance emerged when Patera [5] united the efficient tensor-product forms and rapid convergence of spectral methods with Galerkin-based projection operators common to finite element methods (FEMs). The use of the Galerkin form ensures that the numerical solution to the elliptic subproblems in the Navier–Stokes applications is the *best fit* in the approximation space, regardless of the chosen basis; good bases are still essential for the conditioning of the discrete operators, but the method is much more robust than collocation. Moreover, for second-order symmetric elliptic operators, the discrete operator is guaranteed to be symmetric-positive definite (SPD) for all sets of boundary conditions. In particular, application of Neumann boundary conditions is straightforward in the variational context.

bility properties, with a condition number for the variational-based Laplace operator scaling as $\kappa(A) = O(N^3)$. Lagrange polynomials on *uniformly spaced* points, however, yield a condition number that grows exponentially with N.

As Patera's original paper noted, the spectral element method (SEM) is in fact exactly an FEM. Its strict adherence to local tensor-product forms, however, reduces the complexity for E elements with Nth-order expansions to $O(n)$ storage and $O(nN)$ work, where $n \approx EN^3$ is the number unknown basis coefficients in 3D. This low complexity is qualitatively different from the $O(nN^3)$ work and storage complexities of standard FEM approaches, which are practically limited to $N < 4$, though the use of tensor-product-sum factorization has been gaining traction within the FEM community (e.g., [6–9]).

Tensor-product expansions can yield reduced work for other element types as well. Tetrahedra are accessible through "folded" quadrature rules (e.g., [10–12]) and through the use of matrix-free forms in which the tensor expansion is applied to an $n_L \times E$ array of data, where $n_L \approx (N^3)/6$ is the number of local degrees-of-freedom in an Nth-order element (i.e., tetrahedron). The efficiency in the latter case derives from the fact that the dense interpolation and derivative operators, which are applied in the reference domain, are common to all elements. Although the number of nonzeros in these local operators is $O(N^6)$, the overall storage complexity, which is typically a main concern in scientific computing, is only $O(n)$, provided that $E \gg n_L$. (In a distributed-memory parallel computing context, one requires $E \geq n_L$ on *each processor* because the operator matrices are replicated on each processor.) In a more direct approach, Bernstein polynomial bases have been considered by [13,14] and by Chan and Warburton [15,16], to develop fast *sparse* operators on tetrahedra and pyramids.

3.2 Advection-diffusion in a single deformed element

To introduce the SEM, we consider the 2D advection–diffusion equation,

$$\frac{\partial \tilde{u}}{\partial t} + \mathbf{c} \cdot \nabla \tilde{u} = \nabla \cdot (\mu(\mathbf{x}) \nabla \tilde{u}) + f(\mathbf{x}), \tag{3.3}$$

with initial condition $\tilde{u}(\mathbf{x}, t = 0) = u^0(\mathbf{x})$ and boundary condition $\tilde{u}|_{\partial\Omega} = 0$. We start with the single element case where Ω is a mapping of $\hat{\Omega} = [-1, 1]^2$ (e.g., as in Fig. 3.1, center). We further stipulate positive diffusivity, $\mu(\mathbf{x}) \geq \mu_0 > 0$, and a divergence-free advecting field, $\nabla \cdot \mathbf{c} = 0$. The discrete formulation is based on the weak form, *Find* $u \in X_0^N \subset \mathcal{H}_0^1(\Omega)$ *such that for all* $v \in X_0^N$,

$$\left(v, \frac{\partial u}{\partial t}\right) + a_\mu(v, u) = -c(v, u) + (v, f). \tag{3.4}$$

We define the infinite-dimensional function spaces, $\mathcal{L}^2 := \{v | \int_\Omega v^2 dV < \infty\}$, $\mathcal{H}^1 := \{v \in \mathcal{L}^2 | \int_\Omega \nabla v \cdot \nabla v \, dV < \infty\}$, $\mathcal{H}^1_0 := \{v \in \mathcal{H}^1 | v = 0 \text{ on } \partial\Omega\}$, and the finite-dimensional subspace $X^N \subset \mathcal{H}^1$ spanned by the tensor-product Lagrange polynomials, $l_i(r)l_j(s)$, $i, j \in \{0, \ldots, N\}^2$, introduced in (3.1)–(3.2). An important subset of X^N is the space $X^N_0 := X^N \cap \mathcal{H}^1_0$, which in this example is spanned by $l_i(r)l_j(s)$ for $i, j \in \{1, \ldots, N-1\}^2$. Note that $X^N = \mathbb{P}_N(\hat{\Omega})$; that is, the approximation space is the space of polynomials of degree $\leq N$ on $\hat{\Omega}$. Associated with these spaces we have three bilinear forms,

$$(v, u) := \int_\Omega v\, u \, dV, \quad c(v, u) := \int_\Omega v\, \mathbf{c} \cdot \nabla u \, dV, \quad a_\mu(v, u) := \int_\Omega \mu \, \nabla v \cdot \nabla u \, dV,$$

which will give rise to respective mass matrices (B), convection matrices (C), and diffusion matrices (A). Under the stated conditions, A and B will be symmetric positive definite (SPD), while C will be skew-symmetric. Our task is to compute the integrals in (3.4) using suitable quadrature rules.

Mass matrix. We start with the $\mathcal{L}^2$ inner product,

$$(v, u) := \int_\Omega v\, u \, dV \; = \int_{-1}^{1}\int_{-1}^{1} v\, u \, \mathcal{J}(r, s) \, dr \, ds, \tag{3.5}$$

where the Jacobian is the determinant of the metric terms,

$$\mathcal{J}(r, s) = \left| \frac{\partial x_i}{\partial r_j} \right|, \tag{3.6}$$

associated with the map (3.1). Here and elsewhere, we identify spatial direction using indicial notation, $\mathbf{x} = (x, y) = (x_1, x_2)$ and $\mathbf{r} = (r, s) = (r_1, r_2)$. If (3.5) is to be SPD then we must have $\mathcal{J} > 0$ in $\hat{\Omega}$, which is to say that the map (3.1) should be invertible and preserve right-handedness. In d space dimensions, it is clear from (3.1) that $\mathcal{J}(\mathbf{r}) \in \mathbb{P}_{dN}(\hat{\Omega})$. In many cases the geometry will just be a polynomial of degree $G = 1$ or 2, with the order of $\mathcal{J}$ equal to dG. In general, the integrand in (3.5) will be a polynomial of degree $2N + dG$ in r and s.

All integrals are evaluated using quadrature of appropriate order. For any pair of functions $(v, u) \in X^N$, it is natural to consider GLL-based quadrature,

$$(v, u) \approx (v, u)_N \; := \sum_{k=0}^{N}\sum_{l=0}^{N} v(\xi_k, \xi_l)\rho_k\rho_l\mathcal{J}(\xi_k, \xi_l)u(\xi_k, \xi_l) = \underline{\bar{v}}^T \bar{B} \underline{\bar{u}}. \tag{3.7}$$

Here, the ρ_ks are the GLL quadrature weights, while $\underline{\bar{v}} = [v_{00}\, v_{10} \ldots v_{NN}]^T$ and $\underline{\bar{u}} = [u_{00}\, u_{10} \ldots u_{NN}]^T$ are respective vectors of lexicographically-

ordered nodal values for v and u. We define $\bar{B}$ as the diagonal mass matrix

$$\bar{B} = \begin{bmatrix} \rho_0\rho_0\mathcal{J}_{00} & & & \\ & \rho_1\rho_0\mathcal{J}_{10} & & \\ & & \ddots & \\ & & & \rho_N\rho_N\mathcal{J}_{NN} \end{bmatrix}, \tag{3.8}$$

with $\mathcal{J}_{kl} := \mathcal{J}(\xi_k, \xi_l)$.

We use the "bar" notation (e.g., $\bar{B}$) to indicate that $(v, u) = \underline{\bar{v}}^T \bar{B} \underline{\bar{u}}$ holds for all $u, v \in X^N$ and not just those in X_0^N. Functions that are in X_0^N are represented by $\underline{\bar{u}} = R^T \underline{u}$, where $\underline{u}$ is the vector of *interior* degrees-of-freedom and R^T is an $(N+1)^2 \times (N-1)^2$ Boolean matrix that extends $\underline{u}$ by 0 to the domain boundary. By contrast, $\underline{u} = R\underline{\bar{u}}$ restricts the vector $\underline{\bar{u}}$ by discarding the boundary values. The standard (restricted) mass matrix would thus be $B = R\bar{B}R^T$. We note that this notation holds also in the case of the multidomain SEM and for mixed Neumann–Dirichlet conditions. In these cases, R is simply the $n \times n$ identity matrix associated with the unknowns augmented with zero columns positioned according to the locations of the boundary values in $\underline{\bar{u}}$. We will use R in the sequel to precisely form the system of equations. In practice, one can simply use a *mask*, $\mathcal{M} := R^T R$, to zero out the boundary nodes of any vector $\underline{\bar{u}}$ to which it is applied. Neither R nor $\mathcal{M}$ are explicitly formed.

We remark that, with Nth-order GLL quadrature, the $\mathcal{L}^2$ inner product is exact for any integrand in $\mathbb{P}_{2N-1}$, which means that (3.7) it is not exact even if $\mathcal{J}$ is a constant. For advection-dominated problems, inexact integration for the mass matrix is a potential concern when N is small (e.g., < 4), but less significant for higher order.[2] Given the rapid convergence of the high-order expansions (3.2) and high-order quadrature (3.7), the general approach with the SEM is to accept equal-order errors in the interest of efficiency, rather than to insist upon exact integration. A more precise but expensive alternative to (3.8) is to develop a full mass matrix based on a higher-order Gauss–Legendre (GL) rule. We illustrate this approach here in order to introduce several tools for efficient matrix-free operator-evaluation that are critical to the SEM.

Denote the target GL quadrature points as η_k^M, $k = 0, \ldots, M$, with associated quadrature weights σ_k^M. Each term in the integrand of (3.5) is to be interpolated to (η_k^M, η_l^M). For example, from (3.2), u takes the form

$$u_{kl}^M := u(\eta_k^M, \eta_l^M) = \sum_{j=0}^{N} \sum_{i=0}^{N} l_i(\eta_k^M) l_j(\eta_l^M) u_{ij} \tag{3.9}$$

[2] An extensive discussion of the impact of quadrature for the advection problem is presented in [17,18].

$$= \sum_{j=0}^{N} \hat{J}_{lj} \left(\sum_{i=0}^{N} \hat{J}_{ki} u_{ij} \right) = \sum_{j=0}^{N} \sum_{i=0}^{N} \hat{J}_{ki} u_{ij} \hat{J}_{jk}^{T}, \tag{3.10}$$

where $\hat{J}_{ki} := l_i(\eta_k^M)$ is the 1D interpolation matrix from the $(N+1)$ GLL points to the $(M+1)$ GL points. One can view the entries of the input vectors, u_{ij}, and output vectors, u_{kl}^M, as matrices with the relationship $U^M = \hat{J} U \hat{J}^T$. This (fast) matrix-matrix product amounts to a tensor contraction applied to the rank-2 tensor $\underline{u}$. The vector equivalent of this operation is $\underline{u}^M = J\underline{u}$, where $J := \hat{J} \otimes \hat{J}$ is the Kronecker product of the 1D interpolation operator. Its 3D counterpart is $J = \hat{J} \otimes \hat{J} \otimes \hat{J}$ [19]. Notice that if J is formed explicitly, the storage and operator evaluation cost is $O(N^6)$ in 3D, whereas evaluating it term-by-term requires only $O(N^3)$ storage (for $\underline{u}$) and $O(N^4)$ work. This is Orszag's principal insight to fast generalized spectral methods—never form the operator; rather, exploit the tensor-product forms.

In a similar manner, one can evaluate v and $\mathcal{J}$ at the quadrature points, which gives rise to the "full" inner product, $(v,u) = (v,u)_M := \underline{v}^T (J^T B^M J) \underline{u}$, where B^M is the diagonal mass matrix on the quadrature points,

$$B^M = \begin{bmatrix} \sigma_0^M \sigma_0^M \mathcal{J}_{00}^M & & & \\ & \sigma_1^M \sigma_0^M \mathcal{J}_{10}^M & & \\ & & \ddots & \\ & & & \sigma_M^M \sigma_M^M \mathcal{J}_{MM}^M \end{bmatrix}. \tag{3.11}$$

In this case, the integration will be exact for all integrands of order $\leq 2M+1$ and the mass matrix, $\bar{B} = J^T B^M J$, will be *full*, as will the restricted mass matrix, $B := R\bar{B}R^T$. We reiterate that one would never form B when working with the full mass matrix. If it were necessary to solve $B\underline{u} = \underline{b}$, one would do so iteratively and effect the action of B on a vector by evaluating $B\underline{u} = RJ^T B^M J R^T \underline{u}$ term-by-term. (This system can be effectively preconditioned by the diagonal mass.) Note that the matrix–vector product involving $f(\mathbf{x})$ in (3.4) would be written as $RJ^T B^M \underline{f}^M$ (or $R\bar{B}\bar{\underline{f}}$, if using the diagonal mass matrix), because, unlike $u \in X_0^N$, f does not vanish on the domain boundary.

Advection matrix. We turn next to the first-order advection operator. In this case, it will be critical to use overintegration (or *dealiasing*) by taking $M > N$. Advection requires evaluation of $\mathbf{c} \cdot \nabla u$, with physical-space derivatives computed using the chain rule, $\frac{\partial u}{\partial x_i} = \sum_j \frac{\partial u}{\partial r_j} \frac{\partial r_j}{\partial x_i}$. At each timestep, the advection operator is applied to multiple fields (e.g., each component of the velocity in the momentum equation and to temperature

or other scalar fields). It therefore pays to consolidate several of the operations for performance reasons, which leads to the following fast evaluation of the bilinear form

$$\begin{aligned} c(v,u) &:= \int_\Omega v\,\mathbf{c}\cdot\nabla u\,dV \\ &= \int_{\hat\Omega} v \sum_{i=1}^{d} c_i \left(\sum_{j=1}^{d} \frac{\partial u}{\partial r_j}\frac{\partial r_j}{\partial x_i} \right) \mathcal{J}\,d\mathbf{r} = \int_{\hat\Omega} v \sum_{j=1}^{d} \tilde c_j \frac{\partial u}{\partial r_j}\,d\mathbf{r}, \end{aligned} \tag{3.12}$$

where c_i is the ith component of $\mathbf{c}$ and $\tilde c_j := \mathcal{J}\sum_i \frac{\partial r_j}{\partial x_i} c_i$.

Partial derivatives of u follow directly from (3.2). If the outputs are to be on the nodal points, then

$$\left.\frac{\partial u}{\partial r_1}\right|_{\xi_i,\xi_j} := \left.\frac{\partial u}{\partial r}\right|_{\xi_i,\xi_j} = \sum_{p=0}^{N}\sum_{q=0}^{N} \left.\frac{dl_p}{dr}\right|_{\xi_i} l_q(\xi_j)\,u_{pq} = \sum_{p=0}^{N} \hat D_{ip} u_{pj}, \tag{3.13}$$

where $\hat D_{ip} := \left.\frac{dl_p}{dr}\right|_{\xi_i}$ is the one-dimensional derivative matrix. The output of (3.13) can be expressed as $\underline{u}_r = D_r\underline{u} := (\hat I \otimes \hat D)\underline{u} = \hat D U$, where $\hat I$ is the $(N+1)\times(N+1)$ identity matrix. Similarly, we have $\underline{u}_s = D_s\underline{u} := (\hat D \otimes \hat I)\underline{u} = U\hat D^T$. For output on the $(M+1)$ GL points, we write

$$\begin{aligned} \underline{u}_r^M &= D_r^M\underline{u} := (\hat J \otimes \tilde D)\underline{u}, \\ \underline{u}_s^M &= D_s^M\underline{u} := (\tilde D \otimes \hat J)\underline{u}, \end{aligned} \tag{3.14}$$

with $\tilde D := \hat J\hat D$. In 3D, we have

$$D_r^M = (\hat J\otimes\hat J\otimes\tilde D),\quad D_s^M = (\hat J\otimes\tilde D\otimes\hat J),\quad D_t^M = (\tilde D\otimes\hat J\otimes\hat J). \tag{3.15}$$

If $\hat J$ is square (i.e., it interpolates from $(N+1)$ GLL points to $(N+1)$ GL points), the cost of computing the 3D gradient in $\hat\Omega$ is $16(N+1)^4$ operations when using (3.15). If the quadrature is based on the underlying nodes then $\hat J = \hat I$ and the cost is $6(N+1)^4$ operations. With either approach evaluation of the gradient in physical space requires an additional $15(M+1)^3$ operations to collocate and sum with the metric terms, $\frac{\partial r_i}{\partial x_j}$. By contrast, the contraction on the right of (3.12) requires only $6(M+1)^3$ operations, per advected field, which demonstrates the advantage of consolidating the advection field into $\tilde{\mathbf{c}}$.

The preceding definitions lead to a succinct form for the convection operator,

$$\bar{C} = J^T \left(\sum_{j=1}^{d} C_j^M D_j^M \right), \tag{3.16}$$

which is never formed in practice; only the action of $\bar{C}$ on a vector is required. Here, for $j = 1, \ldots, d$, C_j^M is a $(M+1)^d \times (M+1)^d$ diagonal matrix having entries (for the case $d = 2$), $C_j^M = \mathrm{diag}[\sigma_k^M \sigma_l^M \tilde{c}_j(\eta_k^M, \eta_l^M)]$. As with the mass matrix, we also have the restricted advection operator, $C := R\bar{C}R^T$, such that $\underline{v}^T C \underline{u} = c(v,u)$ for all $v, u \in X_0^N$.

Dealiasing the advection operator. We remark that if $\mathbf{c}(\mathbf{r})$ is a polynomial of degree N in $\hat{\Omega}$ and $\mathbf{x}(\mathbf{r})$ is a polynomial of degree G, then $\tilde{c}_j(\mathbf{r})$ is a polynomial of degree $N + (d-1)G$, as is clear from the following observations [20]. First, the inverse metrics $\frac{\partial r_j}{\partial x_k}$ satisfy $\sum_{j=1}^{d} \frac{\partial x_i}{\partial r_j} \frac{\partial r_j}{\partial x_k} = \delta_{ik}$. That is, the matrix $[\frac{\partial r_i}{\partial x_j}]$ is the inverse of $[\frac{\partial x_i}{\partial r_j}]$, which is a polynomial of degree G. The inverse at each quadrature point involves the cofactors of $[\frac{\partial x_i}{\partial r_j}]$—polynomials of degree $(d-1)G$—divided by the Jacobian, $\mathcal{J}_{kl}^M$, which is exactly canceled by the Jacobian associated with integration. As a result, the integrand for the full bilinear form, $c(v,u)$, is a polynomial of degree at most $3N + (d-1)G$ and the form can therefore be integrated *exactly* by an $(M+1)$-point GL rule with $M \geq (3N + (d-1)G - 1)/2 \approx 3N/2$. The significance of using exact quadrature in this case is that it ensures that $C = -C^T$. As is readily shown through integration-by-parts, $c(v,u) = -c(u,v)$ for all $v, u \in X_0^N$, provided that the domain boundary is closed. If the quadrature is exact, then $c(v,u) \equiv \underline{v}^T C \underline{u}$ and the result follows. Skew symmetry of C implies that it has purely imaginary eigenvalues, which is essential for numerical stability of simulations at high Reynolds or Peclet numbers and in particular for turbulent flow simulations [17].

In Fig. 3.2 we plot the eigenvalues of $B^{-1}C$ for two velocity fields $\mathbf{c}_o = [-y, x]^T$ and $\mathbf{c}_+ = [-x, y]^T$, which satisfy $\mathbf{c} \in \mathbb{P}_1$. Each case yields a skew-symmetric advection matrix, $C = -C^T$, provided that $M \geq N+1$ using either GL or GLL quadrature. It is straightforward to show that the aliased operator (i.e., $M = N$ with GLL quadrature), will retain imaginary eigenvalues in the case of plane rotation, as seen in Fig. 3.2, upper center, while the aliased operator in the straining case, lower center, has positive real eigenvalues and is therefore unstable. Stability is restored by taking $M > N$, as indicated on the right.

Note that one could also consider using $\tilde{C} := \frac{1}{2}(C - C^T)$ to yield a skew-symmetric advection operator without the need to dealias. Operationally, matrix-free application of $\tilde{C}$ would be two times the cost of C alone, which

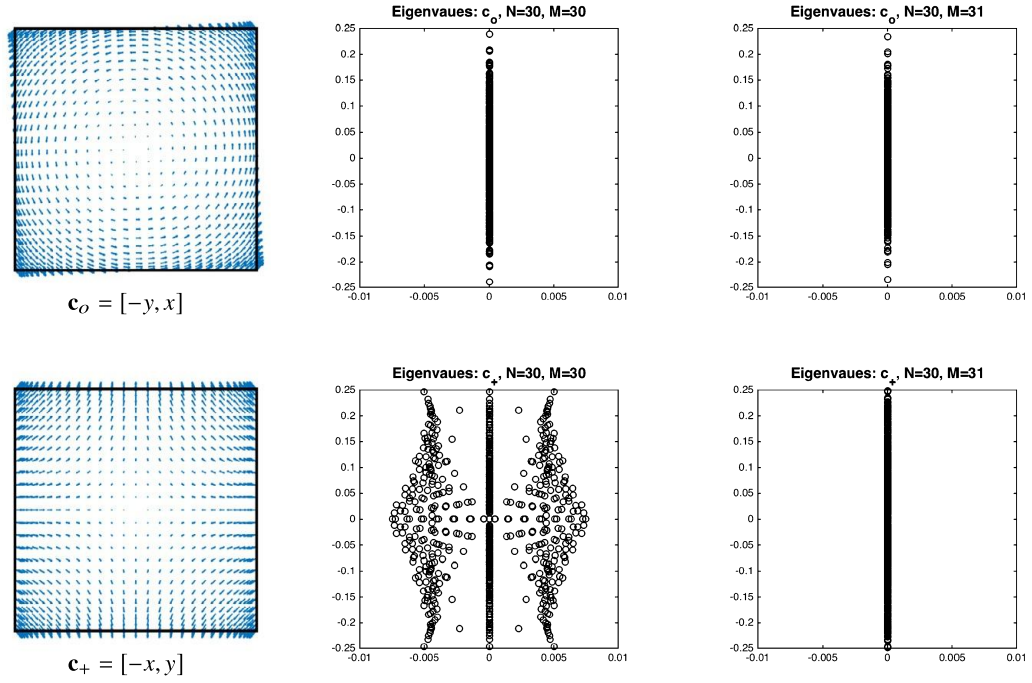

FIGURE 3.2 Effects of aliased (center panels) and dealiased (right panels) evaluation of $c(v,u)$ for the rotational flow, $\mathbf{c}_o$ (upper panels), and straining flow, $\mathbf{c}_+$ (lower panels). In the absence of diffusion, dealiasing is mandatory for the straining case, which otherwise manifests finite-amplitude positive real eigenvalues at any resolution. (The real part of the eigenvalues is $< 10^{-15}$ in the other cases.)

is less than the ≈3–5× cost increase of full dealiasing with $M = \frac{3}{2}N$. In general, however, *C should not* be skew symmetric if $\mathbf{c} \cdot \hat{\mathbf{n}} \neq 0$ on some portion of an outflow (i.e., Neumann) boundary, $\partial\Omega_N$. In practice, it is also often possible to use $M < \frac{3}{2}N$ if full dealiasing is too costly. Dealiasing is required for *stability*—not accuracy.

Diffusion matrix. The highest-order term in the advection–diffusion equation is the Laplace operator, with associated bilinear form $a_\mu(v,u)$. Using the polynomial representations (3.1)–(3.2) leads to [19]

$$a_\mu(v,u) := \int_\Omega \mu(\mathbf{x}) \nabla v \cdot \nabla u \, dV \; := \; \underline{\bar{v}}^T \mathbf{D}^T \mathbf{G} \mathbf{D} \underline{\bar{u}}. \tag{3.17}$$

Here, $\mathbf{D}^T := [D_1^T \; D_2^T] \equiv [D_r^T \; D_s^T]$ is the transpose of the gradient operator in $\hat{\Omega}$ (i.e., as in (3.15)) and $\mathbf{G}$ is the rank 2 tensor having entries G_{ij}, each of which is a diagonal matrix having one entry per nodal point,

$$\left[G_{ij}\right]_{pq} = \rho_p \rho_q \mu(\xi_p, \xi_q) \mathcal{J}(\xi_p, \xi_q) \left(\sum_{k=1}^{d} \frac{\partial r_i}{\partial x_k} \frac{\partial r_j}{\partial x_k} \right)_{\xi_p, \xi_q} . \tag{3.18}$$

k	β_0	β_1	β_2	β_3	α_1	α_2	α_3
1	1	-1	0	0	1	0	0
2	$\frac{3}{2}$	$-\frac{4}{2}$	$\frac{1}{2}$	0	2	-1	0
3	$\frac{11}{6}$	$-\frac{18}{6}$	$\frac{9}{6}$	$-\frac{2}{6}$	3	-3	1

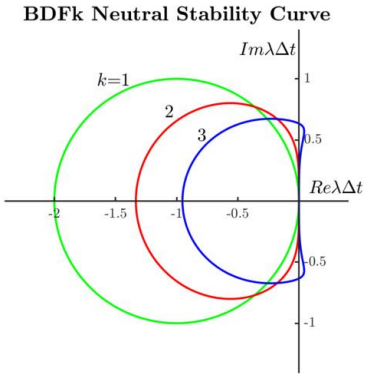

FIGURE 3.3 BDFk/EXTk coefficients and associated stability regions.

For several reasons, the geometric factors G_{ij} are evaluated on the GLL nodes, (ξ_p, ξ_q), rather than with higher-order quadrature. Unless $\mu(\mathbf{r})$ or $\mathbf{x}(\mathbf{r})$ are highly oscillatory, overintegration is not necessary, as noted in [21]. Consequently, the relatively expensive gradient operators become $D_1 = \hat{I} \otimes \hat{D}$ and $D_2 = \hat{I} \otimes \hat{D}$ (or $D_1 = \hat{I} \otimes \hat{I} \otimes \hat{D}$, etc., for $d = 3$), with roughly a d-fold reduction in cost for operator evaluation [22]. From (3.17), we see that we have $\bar{A} = \mathbf{D}^T \mathbf{G} \mathbf{D}$, which we refer to as the *Neumann operator* as it is the matrix that would govern the pure Neumann problem, and $A = R\bar{A}R^T$ as the restricted Dirichlet operator. Note that $\bar{A}$ is semi-positive definite, with null-space corresponding to the constant vector, while A is SPD with condition number $O(N^3)$ [23].

Temporal discretization. The introduction of B, C, and A for the spatial discretization leads to the semidiscrete form of the advection–diffusion equation,

$$B\frac{d\underline{u}}{dt} = -C\underline{u} - A\underline{u} + R\bar{B}\underline{\bar{f}}, \qquad \underline{u}(t=0) = \underline{u}^0. \tag{3.19}$$

Save for the significant interaction of the pressure/divergence-free constraint, this system represents much of the physics of turbulent flow for the case where u is replaced by a vector field $\mathbf{u} = (u_x, u_y, u_z)$ and the advecting field $\mathbf{c}$ is similarly replaced by $\mathbf{u}$. With turbulent incompressible flow in mind, we consider semiimplicit time-advancement for (3.19) based kth-order backward-difference/extrapolation (BDFk/EXTk) with $k = 3$,

$$(\beta_0 B + \Delta t A)\,\underline{u}^n = -\sum_{j=1}^{k}\left[\beta_k \underline{u}^{n-k} + \Delta t \alpha_j \left(C\underline{u}^{n-j} - R\bar{B}\underline{\bar{f}}^{n-j}\right)\right], \tag{3.20}$$

where $\underline{u}^n$ represents $\underline{u}(t)$ at time $t^n = n\Delta t$. The BDFk/EXTk coefficients are given in Fig. 3.3 for the case of uniform Δt, along with the stability regions for explicit BDFk/EXTk applied to the scalar problem $u_t = \lambda u$.

The rationale for the choice (3.20) is the following. First, if one uses a purely explicit formulation, then stability requirements limit Δt to $O(N^{-4})$, which is the inverse of the spectral radius of $B^{-1}A$, whereas explicit treatment of $B^{-1}C$ only requires $\Delta t = O(\Delta x) = O(N^{-2})$. Moreover, since A is SPD, the Helmholtz operator $H = (\beta_0 B + \Delta t A)$ is also SPD and diagonally-preconditioned conjugate gradient (PCG) will generally converge in just a few iterations for small μ (i.e., high Reynolds or Peclet numbers characteristic of turbulent flow). By contrast, C is asymmetric (and nonlinear, in the case of Navier–Stokes), which makes its implicit treatment more challenging. By using $k = 3$, we ensure that the stability region of the timestepper encloses part of the imaginary axis (Fig. 3.3), which is important for the skew-symmetric advection operator. Finally, unlike an implicit RK scheme, (3.19) requires only a single implicit solve per timestep, which is of particular importance for Navier–Stokes where one must also solve a relatively ill-conditioned pressure Poisson problem to enforce incompressibility on each step or substep.

Because the spectral radius of $B^{-1}C$ is $O(\Delta x^{-1})$, the allowable stepsize imposed by the explicit advection treatment is governed by a standard Courant–Friedrichs–Lewy (CFL) condition, modulo a scale factor that relates to the spatial discretization.[3] As with finite differences, we define the Courant number as

$$\mathrm{CFL} := \Delta t \max_i \frac{|\mathbf{c}_i \cdot d\mathbf{x}_i|}{d\mathbf{x}_i \cdot d\mathbf{x}_i}, \tag{3.21}$$

where i ranges over all gridpoints in the domain and $d\mathbf{x}_i \approx (\Delta x_i, \Delta y_i)$ approximates the local grid spacing in each direction at gridpoint $\mathbf{x}_i$. The stability constraint for explicit time advancement of $\underline{u}_t = -B^{-1}C\underline{u}$ is

$$\Delta t \rho(B^{-1}C) \approx S \cdot \mathrm{CFL} \lesssim 0.6339. \tag{3.22}$$

The 0.6339 bound derives from the BDF3/EXT3 stability region of Fig. 3.3. The scaling factor for the SEM is $S \approx 1.5$ for $N = 2$–3 and $S \approx 1.2$–1.16 for $N = 16$–256 ([19], Fig. 3.5.2), from which we find CFL≈ 0.5 as a stability condition for BDF3/EXT3 advancement of (3.19). Note that, for $N = 3$–10, the minimum GLL grid spacing on $\hat{\Omega} = [-1, 1]$ is $\approx 5/N^2$, while the maximum grid spacing is $\sim \frac{\pi}{N}\left(1 - \frac{1}{2N}\right)$. These are to be contrasted with a uniformly-spaced mesh on the same interval for which $\Delta x = 2/N$.

Characteristics timestepping. It is not uncommon in the case of complex geometries to find that the CFL limits are constrained by mesh scales that are significantly smaller than the scales of motion. In these cases, the

[3] For centered 2nd-order finite-differences, $\rho(C)\Delta t = \mathrm{CFL}$, whereas for a Fourier spectral method, $\rho(C)\Delta t = S \cdot \mathrm{CFL}$, with $S = \pi$.

timestep size required for stability might be much smaller than that required for accuracy. Such localized stability constraints can be potentially bypassed by using a characteristics-based approach, introduced originally as a semi-Lagrangian method by Pironneau [24] and further developed in an Eulerian context by Maday et al. [25] and others [26,27]. The idea is to apply implicit BDFk to the *material derivative*, $\frac{Du}{Dt} := \frac{\partial u}{\partial t} + \mathbf{c} \cdot \nabla u$,

$$\left.\frac{Du}{Dt}\right|_{(\mathbf{x}_i, t^n)} = \frac{1}{\Delta t}\left[\beta_0 u_i^n + \sum_{j=1}^{k} \beta_j \tilde{u}_i^{n-j}\right] + O(\Delta t^k). \tag{3.23}$$

Here, $\tilde{u}_i^{n-j}$ is the value of $u^{n-j}(\mathbf{x})$ evaluated at the point $\tilde{\mathbf{x}}_i^{n-j}$ corresponding to the foot of the characteristic that is incident to $\mathbf{x}_i$ at time t^n. Starting at grid point $\mathbf{x}_i$, one can formally find $\tilde{\mathbf{x}}_i^{n-j}$ by following the velocity field backward in time on the interval $[t^n : t^{n-j}]$. Maday et al. [25] note, however, that (3.23) only requires $\tilde{u}_i^{n-j}$, which can alternatively be found by solving an auxiliary hyperbolic problem,

$$\frac{\partial w_j}{\partial s} = -\mathbf{c}(\mathbf{x}, s) \cdot \nabla w_j(\mathbf{x}, s), \quad s \in [t^{n-j}, t^n], \tag{3.24}$$

with respective initial and boundary conditions, $w_j(\mathbf{x}, s = t^{n-j}) := u^{n-j}(\mathbf{x})$ and $w_j(\mathbf{x}, s) = u(\mathbf{x}, s)$ for $x \in \partial\Omega$. At auxiliary time $s = t^n$, the solution satisfies $\underline{w}_j^n = \underline{\tilde{u}}^{n-j}$. That is, the auxiliary problem has advected the required data to the required grid point locations without the need for costly off-grid interpolation. Because the subproblems are linear and the desired result in (3.23) is a linear functional (the sum $\sum_j \beta_j \underline{\tilde{u}}^{n-j}$), one can simultaneously advance the k subproblems using weighted superposition, thereby reducing the overall complexity from $O(k^2)$ to $O(k)$ [27]. Each hyperbolic problem, $\underline{w}_s = -B^{-1}C\underline{w}$, can be efficiently solved with a diagonal mass matrix (B) and dealiased advection operator (C) using multistage (not multistep) methods such as explicit 4th-order Runge–Kutta (RK4). The velocity $\mathbf{c}(\mathbf{x}, s)$ is interpolated/extrapolated on $[t^{n-j}, t^n]$ using known values $\mathbf{c}(\mathbf{x}, t^{n-j'})$, for $j' = 1, \ldots, k$. The subproblems are subject to a Courant condition (based on Δs) of $\mathrm{CFL}_{\Delta s} = \frac{\gamma}{S}$, where $\gamma \approx 2.828$ is the point where the RK4 stability region cuts the imaginary axis and $S \approx 1.4$ is the CFL scaling factor introduced in (3.22). With this approach, one can effectively advance (3.19) with a $\mathrm{CFL}_{\Delta t} \approx 2m$ using m RK4 advective substeps (i.e., $4k$ function evaluations), where $m = 1$ or 2 is most commonly used in practice. The scheme is stable for $k = 1$, 2, and 3, but $k = 2$ is used most frequently as each subproblem can be costly when using a dealiased advection operator, C. Use of the characteristics method typically yields a two- to four-fold reduction in Navier–Stokes solution time, depending on the geometry [27].

3.3 The multielement case

From the basic variational forms and subspaces of the preceding section it is relatively straightforward to develop the multidomain spectral element method. As illustrated in Fig. 3.1, right, a spectral element domain $\Omega = \bigcup_{e=1}^{E} \Omega^e$ comprises E nonoverlapping elements, each of which is a map of $\hat{\Omega}$ with corresponding local representations of the geometry and solution given in the 2D case by

$$\mathbf{x}|_{\Omega^e} := \mathbf{x}^e(r,s) = \sum_{j=0}^{N}\sum_{i=0}^{N} \mathbf{x}_{ij}^e\, l_i(r)\, l_j(s), \tag{3.25}$$

$$u|_{\Omega^e} := u^e(r,s) = \sum_{j=0}^{N}\sum_{i=0}^{N} u_{ij}^e\, l_i(r)\, l_j(s). \tag{3.26}$$

From these expansions we develop local operators, A^e, B^e, and C^e, which are the element-based counterparts to $\bar{A}$, $\bar{B}$, and $\bar{C}$ of Section 3.2. The novel requirements for the multielement case are to account for contributions from each element to the bilinear forms and to ensure that v and u are continuous (i.e., $v,\ u \in X^N \subset \mathcal{H}^1$). We will denote by $\hat{X} \subset \mathcal{L}^2$ the set of piecewise polynomial functions representable by (3.26).

To begin, we derive the discrete diffusion operator A starting, as always, with the bilinear form for $v, u \in X^N \subset \mathcal{H}^1$,

$$\begin{aligned} a_\mu(v,u) &:= \int_\Omega \mu \nabla v \cdot \nabla u \, dV \\ &= \sum_{e=1}^{E} \int_{\Omega^e} \mu \nabla v \cdot \nabla u \, dV \;=\; \sum_{e=1}^{E} (\underline{v}^e)^T A^e \underline{u}^e \;=\; \underline{v}_L^T A_L \underline{u}_L. \end{aligned} \tag{3.27}$$

Here, we have introduced the vectors of local basis coefficients on Ω^e, $\underline{u}^e := [u_{00}^e\, u_{10}^e \ldots u_{NN}^e]^T$, and the collection of these local vectors, $\underline{u}_L$ (illustrated in Fig. 3.4), along with their test-function counterparts, $\underline{v}^e$ and $\underline{v}_L$. The corresponding elemental Laplacian matrices are $A^e = \mathbf{D}^T \mathbf{G}^e \mathbf{D}$, each of which is equivalent to $\mathbf{D}^T\mathbf{G}\mathbf{D}$ of the preceding section, save that $\mathbf{G}^e$ is based on the local geometric expansion $\mathbf{x}^e$ (3.25), rather than $\mathbf{x}$ (3.1). Finally, $A_L = \text{block-diagonal}(A^e)$ is the collection of all the local element matrices. It is clear that application of A_L is fully parallel, since each A^e can be applied independently.[4] Parallel implementations of the SEM therefore use an element-based distribution across P processors, leading to $\approx E/P$ elements per processor [28].

[4]Note that the SEM matrix–vector product is matrix-free in two respects. It is applied element-by-element as a collection of matrices, A^e, each of which is never formed.

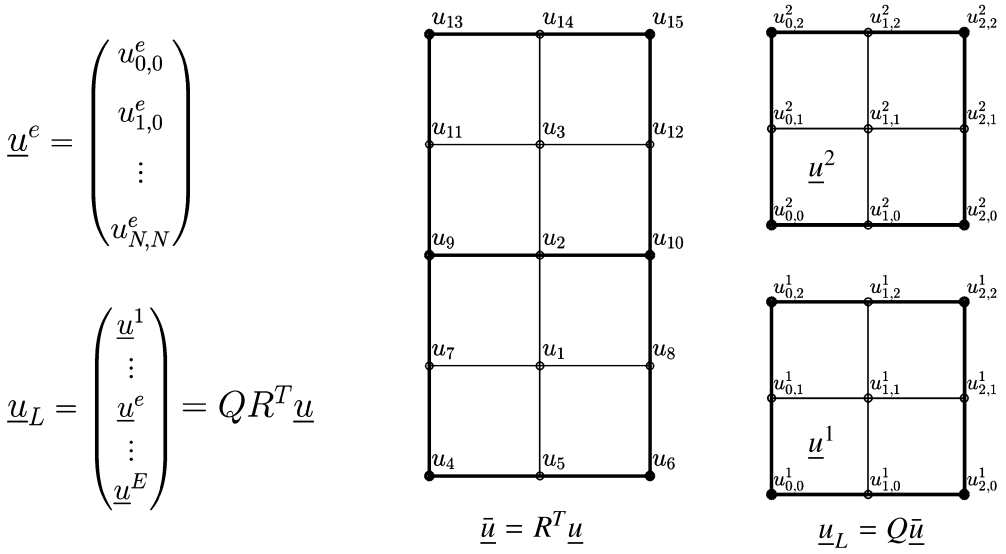

FIGURE 3.4 Illustration of $\underline{u}_L = Q\bar{\underline{u}} = QR^T\underline{u}$ for $(E, N) = (2, 2)$: (center) $\underline{u} = [u_1\ u_2\ u_3]^T$ is the set of active (interior) basis coefficients; $\bar{\underline{u}} = [u_1\ \ldots\ u_{15}]^T = R^T\underline{u}$ is the set of all global basis coefficients, extended by zero to the domain boundary; (right) $\underline{u}_L = Q\bar{\underline{u}}$ is the set of local basis coefficients represented element-by-element (i.e., as $\underline{u}^e$, $e = 1, \ldots, E$). For any $u \in X^N_0 \subset \mathcal{H}^1_0$, all representations contain the same information.

The easiest way to ensure that $u \in X^N \subset \mathcal{H}^1$ is to require that $u(\mathbf{x})$ is continuous. Thus, for any two elements Ω^e and $\Omega^{\hat{e}}$,

$$\mathbf{x}^e_{ij} = \mathbf{x}^{\hat{e}}_{\hat{\imath},\hat{\jmath}} \implies u^e_{ij} = u^{\hat{e}}_{\hat{\imath},\hat{\jmath}}. \tag{3.28}$$

Imposition of continuity for functions in X^N is accomplished by assigning a global numbering, $i_g = g^e_{i,j}$ for each vertex $i, j \in \{0, \ldots, N\}^2$ in each element, $e = 1, \ldots, E$. Vertex pairs (or triplets, etc.) satisfying (3.28) are assigned the same global index, i_g. For example, in Fig. 3.4, we have $i_g = 2$ for $\mathbf{x}^1_{1,2}$ and $\mathbf{x}^2_{1,0}$. That is, the global index is $g^1_{1,2} = g^2_{1,0} = 2$. From the global numbering, one can construct a Boolean matrix Q such that the set of basis coefficients $\underline{u}_L = Q\bar{\underline{u}}$ represents a C^0-continuous function $u(\mathbf{x})$, where $\bar{\underline{u}}$ is the vector of globally- (i.e., uniquely) numbered coefficients, including the boundary values. Assume that there are $\bar{n}$ contiguously-numbered global vertices, $i_g \in \bar{\mathcal{I}} := \{1, \ldots, \bar{n}\}$, and let $k = 1 + i + (N+1)j + (N+1)^2(e-1) \in \{1, \ldots, E(N+1)^2\}$ represent a lexicographical indexing of the local degrees-of-freedom (dofs), $\{u^e_{ij}\}$. Then the kth row of Q will be $\hat{\underline{e}}^T_{i_g}$, where $i_g = g^e_{ij}$ and $\hat{\underline{e}}_{i_g}$ is the i_gth column of the $\bar{n} \times \bar{n}$ identity matrix. Note that the action of Q is to *copy* (scatter) values of a global vector to their local counterparts [19]. (In Fig. 3.4, compare the center panel, $\bar{\underline{u}}$, to the right panel, $\underline{u}_L = Q\bar{\underline{u}}$.) Conversely, application of Q^T *sums* (gathers) local contributions to their global counterparts. Thus, the action of QQ^T is equivalent to an

exchange-and-sum of values among shared dofs (sometimes referred to as direct-stiffness summation [5,29]). This gather–scatter operation is the communication intensive phase of parallel operator evaluation. Note that, in contrast to high-order finite difference stencils, the SEM/FEM stencil depth is unity—one only needs to pass surface data between subdomain interfaces, rather than multiple layers of data. In this sense, SEM operator evaluation is *communication minimal*.

As in the monodomain case, if $u(\mathbf{x}) \in X_0^N$, there exists a rectangular restriction matrix R such that $\underline{\bar{u}} = R^T \underline{u}$ will be zero for all nodes $\mathbf{x}_i \in \partial\Omega_D$. The full Laplacian matrix for the SEM is thus

$$A = R \underbrace{Q^T A_L Q}_{\bar{A}} R^T. \tag{3.29}$$

We refer to A_L as the *unassembled* Laplacian matrix and $\bar{A}$ as the *assembled Neumann operator*. It has a null space corresponding to the constant function. In a similar fashion, the mass matrix and discrete convection operator are

$$B = R \underbrace{Q^T B_L Q}_{\bar{B}} R^T, \qquad C = R \underbrace{Q^T C_L Q}_{\bar{C}} R^T, \tag{3.30}$$

where $B_L = \text{block-diagonal}(B^e)$ and $C_L = \text{block-diagonal}(C^e)$, with B^e and C^e the element-based counterparts to (3.8) and (3.16), respectively. We remark that the factor QR^T on the right of each of (3.29)–(3.30) results from $u \in X^N$ (Q) and $u \in X_0^N \subset X^N$ (R^T), whereas the prefactor RQ^T results from the test functions v being in the same space. The governing mathematics/physics dictates the contents of the block-diagonal matrices, A_L, B_L, and C_L, which are sandwiched between these operators. It is thus relatively easy to construct new operators provided that one understands the constraints on spaces for the test (v) and trial (u) functions.

Before moving onto discretization of the Navier–Stokes equations, we summarize several properties of the SEM by way of the following Poisson example. Consider, in d space dimensions, $-\nabla^2 \tilde{u} = f(\mathbf{x})$, with mixed homogeneous Dirichlet/Neumann boundary conditions, $\tilde{u} = 0$ on $\partial\Omega_D$ and $\nabla\tilde{u} \cdot \hat{\mathbf{n}} = 0$ on $\partial\Omega_N$. The SEM formulation reads

$$A\underline{u} = RQ^T B_L \underline{f}_L. \tag{3.31}$$

The data, $f(\mathbf{x}) \in \hat{X}^N \subset \mathcal{L}^2$, may be piecewise discontinuous, which is why the right-hand side is multiplied by the local mass matrix, B_L. The presence of Q^T and R on the right respectively indicate that the *test functions* are continuous and vanish on $\partial\Omega_D$. If we simply had $B\underline{f}$ on the right of (3.31), it would imply that f is continuous and vanishes on the boundary,

which is overly restrictive. We further remark that if $\partial\Omega_D \neq 0$ then A is symmetric positive definite (SPD) and therefore solvable. Moreover, the variational foundation of the SEM implies that $u(\mathbf{x})$ is the *best fit* in the energy norm, $\|u\|_{\mathcal{E}} := \left[\int_\Omega \nabla u \cdot \nabla u \, dV\right]^{\frac{1}{2}}$, which means that it is the closest element of X_0^N to the analytical solution, $\tilde{u}$, in that norm. The power of the SEM derives from the good approximation properties of high-order polynomials, coupled with stable nodal bases based on the GLL points, which implies that the convergence to $\tilde{u}$ is typically quite rapid (i.e., exponential), once the solution has been resolved.

3.4 $\mathbb{P}_N - \mathbb{P}_N$ formulation for incompressible flows

We turn now to the main topic of this chapter, which is the solution of the incompressible Navier–Stokes equations,

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \nu \nabla^2 \mathbf{u} - \nabla p, \qquad \nabla \cdot \mathbf{u} = 0. \tag{3.32}$$

For conciseness we have absorbed the constant density into the pressure and are using ν rather than the standard inverse Reynolds number, Re^{-1}, as the coefficient in front of the diffusion operator. Two approaches to discretizing (3.32) are common in the spectral element literature. There is the well-known $\mathbb{P}_N - \mathbb{P}_{N-2}$ method [30–32] in which the velocity is in $X_0^N := \mathbb{P}_N(\hat{\Omega}) \cap \mathcal{H}_0^1$ and the pressure is in $Y_0^N := \mathbb{P}_{N-2}(\hat{\Omega}) \cap \mathcal{L}^2$. The idea behind this choice is to use a lower-order discontinuous representation of the pressure to ensure stability and avoid spurious pressure modes, as is commonly done in finite elements. This method yields exponential convergence and can be used for turbulence simulations. An alternative is the $\mathbb{P}_N - \mathbb{P}_N$ method, which has equal-order velocity and pressure spaces. This approach is attractive for marginally-resolved turbulent flows (e.g., as in LES) because additional pressure modes are available to represent the pressure. In the $\mathbb{P}_N - \mathbb{P}_N$ formulation, the pressure and velocity are discretized in the same polynomial space. Even though the development in this section is cast in terms of continuous in space - discrete in time, it should be mentioned that it is specific to $\mathbb{P}_N - \mathbb{P}_N$. The reason is that in the $\mathbb{P}_N - \mathbb{P}_N$ case, pressure is defined on element boundaries and not just inside the elements (as in the $\mathbb{P}_N - \mathbb{P}_{N-2}$ case), making pressure boundary conditions an essential part of this approach. Most of the material presented in this section is based on references [33–35].

The Navier–Stokes equations can be discretized in time using a semi-implicit approach based on numerical integration, such as Adams–Bashforth/Adams–Moulton, or alternatively based on numerical differentiation such as the BDFk/ EXTk method described in the preceding section.

Regarding pressure, the main difference between these two approaches is that in the former the pressure appearing in the semidiscrete equations is an average pressure in the time interval from t^{n-1} to t^n, whereas in the latter the pressure is defined at the new time level t^n. Here, we will limit our discussion to BDF-type methods that are based on a multistep discretization of the velocity time derivative at the new time level and in which the right-hand side is also evaluated at the same time level t^n. In the following we will also focus only on the semidiscrete equations (i.e., discretized in time and not in space) as all important aspects about spatial discretization using spectral elements have been discussed in earlier sections.

Using a splitting approach based on BDFk/EXTk, the semidiscrete form of (3.32) becomes

$$\frac{\beta_0 \mathbf{u}^n}{\Delta t} + \sum_{j=1}^{k} \frac{\beta_j \mathbf{u}^{n-j}}{\Delta t} = -\sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right) - \nabla p^n + \nu \mathbf{L}\left(\mathbf{u}^n\right), \tag{3.33}$$

where ($\mathbf{N} = \mathbf{u} \cdot \nabla \mathbf{u}$) are the nonlinear terms and $\mathbf{L} = \nabla^2 \mathbf{u}$ is the linear viscous operator. The overall approach above can be divided in the following three substeps:

$$\frac{\hat{\mathbf{u}} + \sum_{j=1}^{k} \beta_j \mathbf{u}^{n-j}}{\Delta t} = -\sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right), \tag{3.34}$$

$$\frac{\beta_0 \mathbf{u}^n - \hat{\mathbf{u}}}{\Delta t} = -\nabla p^n + \nu \mathbf{L}\left(\mathbf{u}^n\right). \tag{3.35}$$

A pressure Poisson equation has to solved after the first step (3.34) to obtain the pressure used in the second step (3.35). The pressure equation is obtained by taking the divergence of Eq. (3.33) as follows:

$$-\nabla^2 p^n = \frac{\beta_0 Q^n}{\Delta t} + \sum_{j=1}^{k} \frac{\beta_j Q^{n-j}}{\Delta t} + \nabla \cdot \sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right) - \nu \nabla^2 Q^n, \tag{3.36}$$

where $Q := \nabla \cdot \mathbf{u}$. Since the velocity $\mathbf{u}$ is incompressible, the analytical form of the pressure equation reduces to

$$-\nabla^2 p = \nabla \cdot (\mathbf{u} \cdot \nabla \mathbf{u}). \tag{3.37}$$

However, the way terms are evaluated in (3.36) is instrumental for controlling splitting errors. This control is affected by rewriting Eq. (3.36) as

$$-\nabla^2 p^n - \nabla \cdot \sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right) - \sum_{j=1}^{k} \frac{\beta_j Q^{n-j}}{\Delta t} = \frac{\beta_0 Q^n}{\Delta t} - \nu \nabla^2 Q^n, \tag{3.38}$$

and by setting both sides equal to zero. In this way the pressure Poisson equation to be solved at each time step becomes

$$-\nabla^2 p^n = \sum_{j=1}^{k} \frac{\beta_j Q^{n-j}}{\Delta t} + \nabla \cdot \sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right) = -\frac{1}{\Delta t}\nabla \cdot \hat{\mathbf{u}}. \tag{3.39}$$

Consequently, from (3.38) the divergence Q^n effectively satisfies

$$\frac{\beta_0 Q^n}{\Delta t} - \nu \nabla^2 Q^n = 0, \tag{3.40}$$

which is an elliptic equation governed by the maximum principle, implying that divergence errors at every timestep are maximum at domain boundaries and can be controlled by an appropriate choice of pressure boundary conditions. It is clear that based on (3.40) there exists a numerical boundary layer of thickness $l = \sqrt{\nu \Delta t/\beta_0}$. If instead of (3.40) the first term of (3.39) is not included, then the differential equation governing the evolution of divergence would become

$$\frac{\beta_0 Q^n}{\Delta t} + \sum_{j=1}^{k} \frac{\beta_j Q^{n-j}}{\Delta t} + \nu \nabla^2 Q^n \approx \left(\frac{\partial Q}{\partial t}\right)^n + \nu \nabla^2 Q^n = 0. \tag{3.41}$$

Eq. (3.41) is a parabolic equation and the divergence errors at every time step would depend not only on boundary but also on initial conditions. To avoid this, the pressure Poisson equation solved is (3.39). The pressure boundary condition used at the parts of the domain boundary where the velocity is specified (Dirichlet) is obtained by taking the normal component of (3.33),

$$\mathbf{n} \cdot \nabla p^n = \frac{\partial p^n}{\partial n} = \mathbf{n} \cdot \left(-\mathbf{a}^n - \sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right) + \nu \nabla^2 \mathbf{u}^n \right), \tag{3.42}$$

where $(\mathbf{n} \cdot \mathbf{a})^n$ is the normal component of the acceleration at t^n which is prescribed at Dirichlet velocity boundaries. In the case of impermeable walls (3.42) reduces to

$$\frac{\partial p^n}{\partial n} = -\mathbf{n} \cdot \left(\nu \nabla^2 \mathbf{u}^n\right). \tag{3.43}$$

Thus, the solution of the pressure equation (3.39) is coupled to the solution of the last velocity equation in step (3.35) due to the fact that both p and $\mathbf{u}$ in (3.43) are at the same, new time level t^n. To decouple the solution procedure for p^n and $\mathbf{u}^n$, one option is to neglect the viscous term in (3.43)

and simply use the inviscid boundary condition,

$$\frac{\partial p^n}{\partial n} = 0. \tag{3.44}$$

However, since the right-hand side of Eq. (3.43) is not zero and is independent of the discretization parameter Δt, the use of (3.44) may induce $O(1)$ error.

An alternative is to approximate the linear viscous terms at the boundary using an explicit extrapolation, similar to the nonlinear terms:

$$\frac{\partial p^n}{\partial n} = \mathbf{n} \cdot \left(-\mathbf{a}^n - \sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right) + \nu \sum_{j=1}^{k} \alpha_j \nabla^2 \mathbf{u}^{n-j} \right). \tag{3.45}$$

It was demonstrated in [33] that boundary condition (3.45) can lead to numerical instabilities. A stable scheme can instead be constructed by rewriting the linear viscous operator $\mathbf{L}(\mathbf{u}) = \nabla^2 \mathbf{u}$, using the identity

$$\mathbf{L}(\mathbf{u}) \equiv \nabla^2 \mathbf{u} = \nabla(\nabla \cdot \mathbf{u}) - \nabla \times (\nabla \times \mathbf{u}) \tag{3.46}$$

and by splitting the velocity $\mathbf{u}^n$ into a solenoidal part, $\mathbf{u}_S^n$, approximated by an explicit extrapolation, and an irrotational part, $\mathbf{u}_I^n$, which is treated implicitly as follows:

$$\mathbf{u}^n = \mathbf{u}_S^n + \mathbf{u}_I^n, \tag{3.47}$$

$$\nabla \cdot \mathbf{u}^n = \nabla \cdot \mathbf{u}_I^n = Q^n \approx 0, \tag{3.48}$$

$$\nabla \times \mathbf{u}^n = \nabla \times \mathbf{u}_S^n = \boldsymbol{\omega}^n = \sum_{j=1}^{k} \alpha_j \boldsymbol{\omega}^{n-j} + O(\Delta t^k). \tag{3.49}$$

With this decomposition, the pressure boundary condition becomes

$$\frac{\partial p^n}{\partial n} = \mathbf{n} \cdot \left(-\mathbf{a}^n - \sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right) + \nu \nabla Q^n - \nu \sum_{j=1}^{k} \alpha_j \nabla \times \boldsymbol{\omega}^{n-j} \right). \tag{3.50}$$

In the latter equation the term ∇Q^n is dropped since it is required that $Q^n = 0$ to honor the incompressibility constraint. Defining $\omega_s = \mathbf{n} \cdot (\nabla \times \boldsymbol{\omega})$, the boundary condition (3.50) at impermeable walls can be expressed as

$$\frac{\partial p^n}{\partial n} = -\nu \sum_{j=1}^{k} \alpha_j \omega_s^{n-j}, \tag{3.51}$$

which for first-order extrapolation ($k = 1$) reduces to

$$\frac{\partial p^n}{\partial n} = -\nu \omega_s^{n-1}. \tag{3.52}$$

On the other hand, the exact pressure boundary condition at walls is

$$\frac{\partial p^n}{\partial n} = -\nu \omega_s^n \tag{3.53}$$

and, using Taylor expansion around t^{n-1},

$$\omega_s^n = \omega_s^{n-1} + \Delta t \left(\frac{\partial \omega_s}{\partial t} \right)^{n-1} + \cdots, \tag{3.54}$$

it can be shown that the effective boundary condition for Q^n in the divergence equation (3.40) is

$$\frac{\partial Q^n}{\partial n} = \Delta t \left(\frac{\partial \omega_s}{\partial t} \right)^{n-1} \tag{3.55}$$

and where for a kth-order extrapolation Δt becomes Δt^k. Thus, the solution of Eq. (3.39) using boundary condition (3.50) minimizes splitting errors by reducing the level of divergence at domain boundaries to $O(\Delta t^k)$. Based on (3.40), it is expected that the divergence close to the boundary will behave as $Q(s) = Q_w e^{-s/l}$, where s is a coordinate normal to the boundary. The boundary divergence Q_w will be $Q_w = -l(\partial Q/\partial n)_w$ and since $Q_w = O(\partial \mathrm{v}/\partial n)$, similar order-of-magnitude analysis for the normal-to-the-boundary velocity component v gives

$$\mathrm{v} \propto Q_w l \propto \left(\frac{\partial Q}{\partial n} \right)_w \frac{\nu \Delta t}{\beta_0}.$$

This relation demonstrates that the temporal error of the velocity field is one order smaller in Δt than the corresponding error in the boundary divergence. This result agrees with the classical splitting scheme of first-order corresponding to the inviscid-type boundary condition (3.44). In particular, the order $O(1)$ errors in $(\partial Q/\partial n)_w$ result in first-order $O(\Delta t)$ errors in the velocity field. Similarly, a first (or kth) order time treatment of the pressure boundary condition as explained above, should be expected to produce second (or $(k+1)$) order results in the velocity field. The above argument clearly demonstrates that the time accuracy of the global solution is directly dependent on the boundary values of the divergence (i.e., $\partial Q/\partial n$) and, therefore on the treatment of pressure boundary conditions.

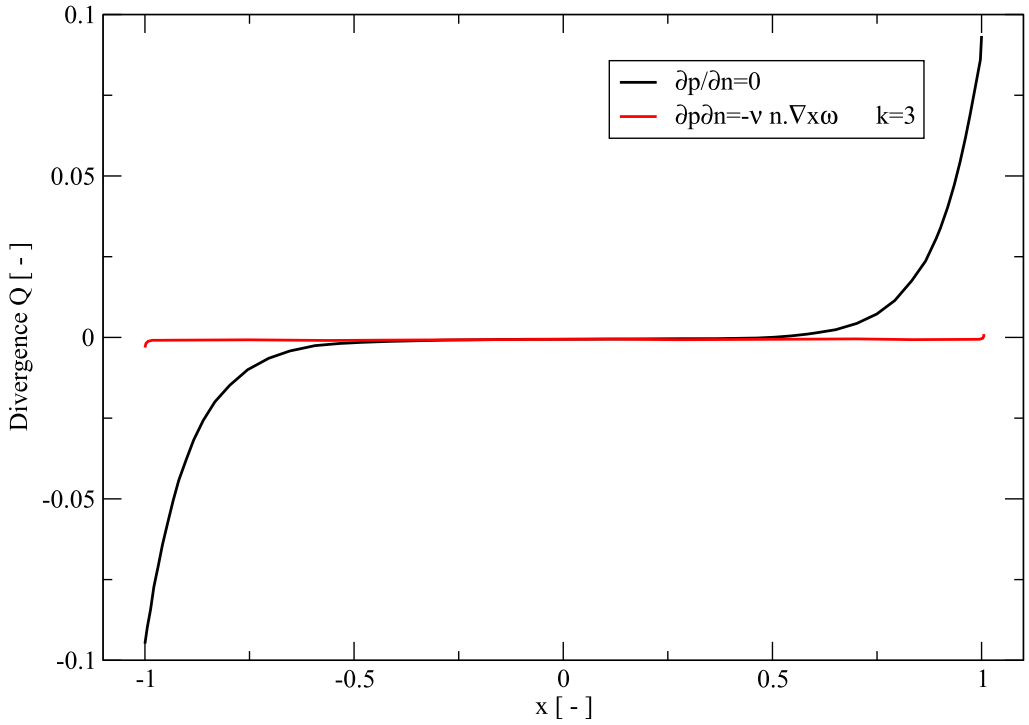

FIGURE 3.5 Distribution of divergence in incompressible channel model problem for the inviscid and rotational pressure boundary conditions.

An example demonstrating the reduction of divergence errors using the pressure boundary condition (3.50) compared to the inviscid boundary condition (3.44) is shown in Fig. 3.5 for a two-dimensional problem with one periodic direction as described in [35].

The $\mathbb{P}_N - \mathbb{P}_N$ based splitting approach presented in this section allows the decoupling of the solution procedure for pressure and velocity. The solution at the end of a timestep satisfies the no-slip solution exactly and the divergence-free condition approximately, with an error which is of high order in time, consistent with and not larger than the formal order of accuracy of the time-stepping scheme employed. This is mainly due to pressure boundary condition (3.50) which controls divergence errors at Dirichlet velocity boundaries as described earlier. It is also important to mention that accurate representation of second order velocity derivatives is necessary at Dirichlet velocity boundaries for the proper implementation of the pressure boundary condition, making this approach specific to high-order methods. Finally, using equal order pressure and velocity approximation spaces allows efficient use of the same matrix operators for the one Poisson and three Helmholtz solvers performed in each timestep.

Fully discrete implementation. Here, we describe the weak-form implementation of the $\mathbb{P}_N - \mathbb{P}_N$ substeps. We assume we have an underlying spectral element mesh with Dirichlet boundary conditions for the velocity,

which translates into Neumann conditions for the pressure. As in Section 3.3, the discrete operators A, B, C operate on functions in X_0^N; $\bar{A}$, $\bar{B}$, $\bar{C}$ operate on functions in X^N; and A_L, B_L, C_L operate on (discontinuous) functions in $\hat{X}^N$. In the $\mathbb{P}_N - \mathbb{P}_N$ formulation, the pressure is in X^N, so we may use the same operators for both velocity and pressure, mindful of the differences in the boundary conditions.

Starting with (3.34), we evaluate a tentative velocity,

$$\hat{\mathbf{u}} := -\sum_{j=1}^{k}\left[\beta_j \mathbf{u}^{n-j} + \Delta t \alpha_j \left(\mathbf{u}^{n-j}\cdot\nabla\mathbf{u}^{n-j} - \mathbf{f}^{n-j}\right)\right], \tag{3.56}$$

where we have added a prescribed body force $\mathbf{f}$ to account for Boussinesq, Coriolis, or other body forces. In (3.56), which is essentially just construction of the right-hand side for the subsequent steps, boundary conditions and interelement continuity are not enforced. The only critical feature is to ensure that the advection terms are dealiased, which means that they should be evaluated locally in weak-form as $C_L^{n-j}\underline{\mathbf{u}}_L^{n-j}$ (i.e., as introduced in (3.30), with advecting velocity $\mathbf{c}_L := \underline{\mathbf{u}}_L^{n-j}$). The discrete form of (3.56) is

$$\underline{\hat{\mathbf{u}}}_L := -\sum_{j=1}^{k}\left[\beta_j \underline{\mathbf{u}}_L^{n-j} + \Delta t \alpha_j \left(B_L^{-1}C_L^{n-j}\underline{\mathbf{u}}_L^{n-j} - \underline{\mathbf{f}}_L^{n-j}\right)\right]. \tag{3.57}$$

Since $\underline{\hat{\mathbf{u}}}_L$ is later multiplied by the mass matrix, one needs to factor it out of the advection term, which already has the (higher-order) quadrature weights factored in, per (3.30) and (3.16).

The pressure Poisson problem is cast in weak form as: *Find* $p \in X_p^N \subset X^N$ *such that for all* $q \in X_p^N$,

$$\int_\Omega \nabla q\cdot\nabla p\, dV = \frac{1}{\Delta t}\int_\Omega \nabla q\cdot\tilde{\mathbf{u}}\, dV + \frac{1}{\Delta t}\int_{\partial\Omega} q\beta_0\mathbf{u}^n\cdot\hat{\mathbf{n}}\, dS. \tag{3.58}$$

If the flow domain is closed, then (3.58) is a Neumann problem and we take X_p^N to be the set of functions in X^N that have zero mean on Ω. If the domain is open, we enforce functions in X_p^N to be zero wherever outflow conditions are to be applied.[5] Here we have introduced

$$\tilde{\mathbf{u}} := \hat{\mathbf{u}} - \Delta t\nu\sum_{j=1}^{k}\alpha_j(\nabla\times\omega^{n-j}), \tag{3.59}$$

[5] If there is more than one outlet, one typically has to set different outlet pressures or prescribe some type of flow distribution [36].

which leads to (3.50) being satisfied weakly, as can be seen through formal integration-by-parts of the left- and right-hand sides of (3.58).

The final substep is the viscous update for each velocity component $\underline{u}_i$,

$$[\beta_0 B + \Delta t \nu A]\,\underline{u}_i^n = R Q^T B_L \left[\underline{\hat{u}}_{i,L} - \Delta t\, \partial_i \underline{p}_L\right], \tag{3.60}$$

where $\partial_i \underline{p}_L$ is understood to be ith component of the pressure gradient, evaluated at the GLL nodal points in each element. Nontrivial velocity boundary conditions, $\mathbf{u} = \mathbf{u}_b$ on $\partial\Omega_D$, are readily treated by solving

$$[\beta_0 B + \Delta t \nu A]\,\underline{u}_{i,0}^n = R Q^T B_L \left[\underline{\hat{u}}_{i,L} - \Delta t\, \partial_i \underline{p}_L\right] + R\bar{A}\underline{\bar{u}}_{b,i}, \tag{3.61}$$

and setting $\underline{\bar{u}}_i^n = R^T \underline{u}_{0,i}^n + \underline{\bar{u}}_{b,i}$, where $\underline{\bar{u}}_{b,i}$ is the set of grid point values, including those on the boundary, for any $\mathbf{u}_b \in X^N \subset \mathcal{H}^1$ that satisfies the prescribed boundary conditions. (Any discontinuity in $\mathbf{u}_b$ will be inherited by $\mathbf{u}^n$ so it is imperative that $\mathbf{u}_b \in X^N$.) Note that $\mathbf{u}_b$ is the value of $\mathbf{u}^n$ that is prescribed in the right-most term of the pressure Poisson equation, (3.58).

3.5 $\mathbb{P}_N - \mathbb{P}_N$ formulation for low-Mach-number flows

Many engineering systems feature flows where compressibility is not negligible. In combustion applications, for example, in internal combustion engines, thermal dilation and especially compression from the piston motion result in significant density variations. In this section, we address a low-Mach-number formulation of the SEM to support turbulent reactive flows in open and closed domains [37,38].

As described in Chapter 13, many combustion applications can be accurately described using the low-Mach-number approximation, in which acoustic waves are decoupled from the governing equations using regular perturbation [39–41]. In most low-Mach approaches, the continuity equation is replaced by an expression for the divergence of the velocity field, derived from continuity, energy and species mass fraction equations, that can be considered as a constraint imposed on the velocity field by "thermochemistry." In addition, when the domain volume also changes in time, the temporal variation of the background "thermodynamic" pressure is nonzero and its rate of change requires the solution of an additional ODE in time, obtained by integrating the energy equation over the domain [27]. The divergence constraint is enforced on the velocity field by the action of the so-called "hydrodynamic" pressure that acts as a Lagrange multiplier similar to incompressible flow.

Here we focus on a splitting scheme for low-Mach compressible flows that extends the formulation presented in Section 3.4. It is a generalized

framework that can account for finite divergence in low speed flows, generated by thermochemical or other processes, and which decouples the solution of pressure and velocity without relying on an iterative procedure within the timestep. Moreover, it smoothly goes to the limit of zero divergence, i.e., incompressible flow, with a high-order accuracy in time and space. The spatial discretization of all equations, including (3.62)–(3.63), is based on the weighted residual formulation of the preceding section.

For brevity, we will assume that the integration of the thermochemistry problem, the energy and mass fractions equations, has been performed within the timestep using a solver that can accurately handle the stiffness due to the chemical reactions involved, as explained in Chapter 13. Thus for the presentation below, it is assumed that the temperature and mass fractions have been updated to their new timestep values and will focus on the splitting approach used for the integration of the Navier–Stokes equations. The form of the low-Mach form of the Navier–Stokes equations is:

$$(\textit{Continuity}) \qquad \nabla \cdot \mathbf{u} = Q_T, \tag{3.62}$$

$$(\textit{Momentum}) \qquad \rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p_1 + \nabla\cdot\left[2\mu\mathsf{S} - \frac{2}{3}(\nabla\cdot\mathbf{u})\,\mathsf{I}\right], \tag{3.63}$$

where Q_T is the thermal divergence constraint imposed by thermochemistry, p_1 is the "hydrodynamic" pressure and S is the strain-rate tensor

$$\mathsf{S} = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right). \tag{3.64}$$

The density is evaluated using the equation of state. Having obtained the density, transport coefficients, temperature, species mass fractions, and thermodynamic pressure (for closed systems) at the new time level t^n, the thermal divergence Q_T^n can also be evaluated. Subsequently, the solution of the hydrodynamic subsystem (3.62)–(3.63) is based on a projection-type velocity correction scheme [35], designed to enforce the nonzero divergence constraint Q_T^n on the velocity field and is an extension of the formulation presented in Section 3.4 As a first step, the velocities are updated with the nonlinear terms and a pressure Poisson equation is solved by using boundary conditions based on high-order extrapolation of viscous contribution of the velocity, as discussed in the preceding sections. Once the hydrodynamic pressure, p_1^n, is known, the velocity is corrected in a the implicit viscous correction step based on standard Helmholtz equations (3.60). Similar to the $\mathbb{P}_N - \mathbb{P}_N$ formulation in Section 3.4, the steps of the splitting scheme are:

$$\frac{\hat{\mathbf{u}} - \sum_{j=1}^{k}\beta_j\mathbf{u}^{n-j}}{\Delta t} = -\sum_{j=1}^{k}\alpha_j\mathbf{N}\left(\mathbf{u}^{n-j}\right), \tag{3.65}$$

$$-\nabla \cdot 2\mu^n \mathsf{S}^n + \beta_0 \rho^n \frac{\mathbf{u}^n}{\Delta t} = -\rho^n \frac{\hat{\mathbf{u}}}{\Delta t} - \nabla p_1^n - \frac{2}{3}\nabla\left(\mu^n Q_T^n\right). \tag{3.66}$$

The variable-coefficient pressure Poisson equation derived to enforce the divergence constraint Q_T^n is

$$\nabla \cdot \left(\frac{1}{\rho^n}\nabla p_1^n\right) = \frac{\nabla \cdot \hat{\mathbf{u}} - \beta_0 Q_T^n}{\Delta t} + \nabla \cdot \left[\nu^n \left(\frac{4}{3}\nabla Q_T^n - \sum_{j=0}^{k} \alpha_j \nabla \times \omega^{n-j}\right) + \frac{\nabla \mu^n}{\rho^n} \cdot \sum_{j=0}^{k} \alpha_j \mathbf{S}^{n-j}\right] \tag{3.67}$$

with boundary conditions at Dirichlet velocity boundaries

$$\frac{1}{\rho^n}\frac{\partial p_1^n}{\partial n} = -\mathbf{n} \cdot \left(\mathbf{a}^n + \sum_{j=1}^{k} \alpha_j \mathbf{N}\left(\mathbf{u}^{n-j}\right)\right) + \mathbf{n} \cdot \left[\nu^n \left(\frac{4}{3}\nabla Q_T^n - \sum_{j=0}^{k} \alpha_j \nabla \times \omega^{n-j}\right) + \frac{\nabla \mu^n}{\rho^n} \cdot \sum_{j=0}^{k} \alpha_j \mathbf{S}^{n-j}\right]. \tag{3.68}$$

As analyzed in detail in [37] and [38], the splitting error in mass conservation, defined as

$$\phi = Q^n - Q_T^n,$$

where $Q^n = \nabla \cdot \mathbf{u}^n$ is the divergence of the velocity field at the end of the timestep, is governed by the following elliptic equation:

$$\left(\frac{\beta_0}{\nu \Delta t}\right)\phi - \nabla \cdot \nu^n \nabla \phi = -\nabla \cdot \nu^n \left(\nabla \times \boldsymbol{\omega}^n - \sum_{j=0}^{k} \alpha_j \nabla \times \omega^{n-j}\right)$$

with boundary conditions

$$\frac{4}{3}\frac{\partial \phi}{\partial n} = \mathbf{n} \cdot \left(\nabla \times \boldsymbol{\omega}^n - \sum_{j=0}^{k} \alpha_j \nabla \times \boldsymbol{\omega}^{n-j}\right).$$

The low-Mach-number formulation yields kth-order accuracy in time (typically, $k = 3$) for all hydrodynamic variables in combination with minimal splitting errors. Specifically, it was shown in [38] and [33] using asymptotic analysis that the splitting errors due to the effect of the pressure boundary condition is indeed of order $O(\nu^{1/2}\Delta t^{k+1/2})$ in a boundary layer of

thickness $l = \sqrt{\beta_0 \nu \Delta t}$ close to the boundaries, exactly in the same way it behaves in incompressible flows. On the other hand, the overall error in the interior of the domain, due to the variation of viscosity and density (e.g., inside a flame) is of order $O(\nu \Delta t^{k+1})$. Therefore, the error in mass conservation due to the splitting scheme is of higher order than the formal order of temporal accuracy of the time-stepping used.

As is the case for the incompressible $\mathbb{P}_N - \mathbb{P}_N$ method, the projection scheme described here amounts to solving a variable coefficient pressure Poisson equation and three coupled Helmoltz equations for the velocity components at each timestep, with boundary conditions being applied at time t^n. It is important to note that this approach does not require any outer iterations to enforce the non-zero divergence constraint.

3.6 Computing the pressure

For incompressible flows, the pressure acts as a constraint (in fact, the Lagrange multiplier) that enforces a divergence-free velocity. This model is of course an approximation. If one pushes liquid into one end of a pipe it does not immediately come out the other end; rather, there is a delay due to the finite acoustic speed in the medium. The incompressible model results when this acoustic speed is taken to infinity. It is therefore not surprising that the pressure solve is the stiffest substep in time advancement of incompressible flows. Nonlinear advection, which features first-order derivatives in space, is relatively slow and can be treated explicitly, resulting in time step sizes scaling as $O(\Delta x/U)$. The nonlinear update does not require imposition of boundary conditions or interelement continuity because $\mathbf{u} \cdot \nabla \mathbf{u} \notin \mathcal{H}_0^1$. Diffusion, which features second-order derivatives, is generally treated implicitly to avoid time steps scaling as $O(\Delta x^2)$ and because it is relatively easy to do so. The viscous update requires solution of discrete elliptic subproblems, $H \underline{u}_i = \underline{b}_i$, $i = 1, \ldots, d$, where the SPD matrix $H = \beta_0 B + \nu \Delta t A$ is readily treated by diagonally-preconditioned conjugate gradient iteration because B is diagonal for the SEM and (in nondimensional units), $\nu \Delta t \ll 1$. By contrast, the pressure-Poisson problem, $A \underline{p}^n = \underline{b}^n$, is relatively ill-conditioned for all values of Δt.

For unstructured turbulent flow problems, the majority of the computational effort for each time step is spent in the pressure solve. When using standard BDFk/EXTk splitting, computing the pressure requires about 80%–90% of the solution time. If a 2nd-order characteristics approach is used, one finds that 30%–50% of the time is spent in the more expensive advection update (3.23), leaving 40%–60% of the time in the pressure solve. In the following, we describe several pressure preconditioning strategies and note that no single strategy is best across all computational domains. Some geometries are particularly challenging (e.g., those with high aspect-

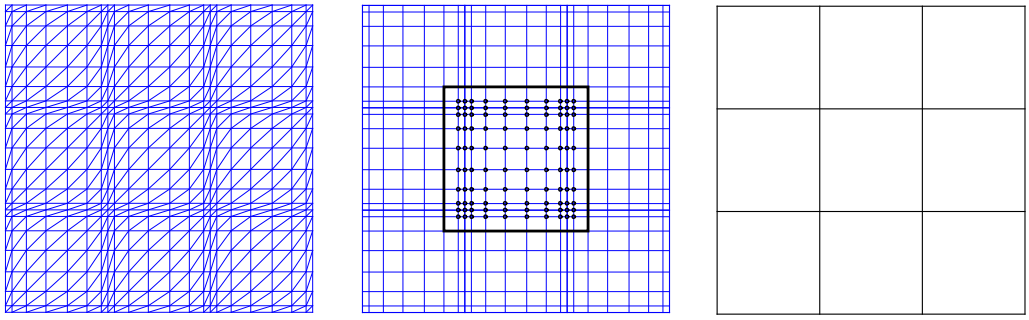

FIGURE 3.6 Meshes for SEM preconditioners: (left) example triangulation of GLL points for low-order finite-element preconditioner, A_f; (center) Schwarz overlapping domain for a single spectral element—each vertex inside the box is a dof for $\underline{z}_s$ (3.71) with homogeneous Dirichlet conditions on the box boundary; (right) triangulation for the Schwarz coarse-grid operator, A_0.

ratio elements) and call for robust but relatively expensive preconditioning approaches.

Low-order preconditioners. Orszag's seminal paper on spectral methods for complex geometries [2] pointed out the remarkable fact that, if A derives from a nodal-based spectral discretization of a 1D Poisson problem with Dirichlet conditions, then its finite-difference counterpart, A_f, formed using *the same nodes*, is spectrally equivalent to A. Specifically, he showed that the condition number of the preconditioned operator satisfies, $\kappa(A_f^{-1}A) \sim \frac{\pi^2}{4}$. Deville and Mund [42] and Canuto and Quarteroni [43] extended the finite-difference idea to base A_f on low-order finite-elements whose nodes were coincident with the spectral nodal points, as illustrated in Fig. 3.6, left. More recent articles [44,45] provide a comprehensive review regarding the analysis of this approach and discuss important considerations in constructing A_f to ensure that $\kappa(A_f^{-1}A)$ is small, especially for the nontrivial three-dimensional case.[6] The key point is that A_f is *sparse* and can therefore be treated with general sparse solvers such as algebraic multigrid (AMG). The A_f systems are, however, difficult to solve because the tensor-product GLL grids lead to high aspect-ratio cells, which are notoriously challenging for iterative methods. Nonetheless, production-level open-source AMG codes can be effective for the systems in A_f (one typically needs only a single V-cycle, and not a full solve) and, when combined with the low condition number of $A_f^{-1}A$, can yield a robust approach that is appropriate when other preconditioners fail.

Overlapping Schwarz methods. The Schwarz overlapping method and its variants gained considerable attention in the late 1980s with the ad-

[6]Note that $\kappa(A_f^{-1}A) > \frac{\pi^2}{4}$ for deformed geometries and for Neumann problems that typify the pressure Poisson problem [45].

vent of parallel computing and has retained its importance under the general heading of domain-decomposition methods. One of the key developments was the additive Schwarz method introduced by Dryja and Widlund [46], which leads to an SPD preconditioner that is not a contractor but is nonetheless suitable for conjugate gradient (CG) iteration.

The idea behind Schwarz methods is to define a set of Boolean restriction matrices R_s associated with overlapping subdomains, $\overline{\Omega}^s$, $s = 1, \ldots, S$ that extract from the set of all dofs those that are *interior* to $\overline{\Omega}^s$. If $M\underline{z} = \underline{r}$ is the preconditioning step in CG (with $M \approx A$), then the additive Schwarz preconditioner M_{AS} is defined by the output,

$$\underline{z} := M_{AS}^{-1}\underline{r} = \sum_s R_s^T A_s^{-1} R_s \underline{r}, \tag{3.69}$$

where $A_s := R_s A R_s^T$ implies that $\underline{z}_s := R_s^T A_s^{-1} R_s \underline{r} = R_s^T A_s^{-1} R_s A \underline{z}$ is the projection (in the A-norm) of $\underline{z}$ onto the column space of R_s^T. If the number of subdomains is large, one must add to (3.69) a coarse-grid correction, $\underline{z}_0 = R_0^T A_0^{-1} R_0 \underline{r}$, where R_0^T is an interpolator from a coarse mesh (usually of mesh size $O(H)$, where H is approximately the diameter of the subdomains) to the originating fine-mesh nodes. An important extension to the additive Schwarz method (ASM) is the *restricted additive Schwarz method*, introduced by Cai and Sarkis [47]. In this approach, the prolongation operator R_s^T is replaced by $\tilde{R}_s^T$, which does *not* add the solutions in the overlap regions after solving the local systems in A_s. RAS has less communication than ASM and is often faster, but it does preclude the use of CG because of lack of symmetry.

Dryja and Widlund [48,49] established that the condition number of $M_{AS}^{-1}A$ with a coarse grid is $O(1 + \frac{H}{\delta})$ for the finite-element case where the subdomains are *shape-regular* (i.e., have reasonable aspect ratios), the amount of overlap is δ, and the coarse-grid scale is roughly the same as the subdomain size, $O(H)$. Consequently, under these conditions the ASM has bounded iteration counts in 2D and 3D, independent of the number of subdomains. The burden of scalability, however, is shifted to solving the coarse grid problem, $A_0\underline{z}_0 = \underline{r}_0$, which can be challenging if the number of subdomains or processors is large [50].

Pahl [51], Fischer [52], and Lottes and Fischer [53] developed Schwarz-based preconditioners (and smoothers, in a multigrid context) for the SEM in which each element is a subdomain, as illustrated in Fig. 3.6, center, and the coarse grid is taken to be the spectral element mesh with $N = 1$ shown on the right. In this example, each domain shares spectral element interface values, plus one strip of interior values, with each of its neighboring elements.[7] Application of the restriction matrix, R_s is equivalent to impos-

[7] In practice, one only collects data from elements that share a face, since the number of elements that share a corner or edge (in 3D) may be highly variable.

ing Dirichlet conditions on the box indicated in black and δ is defined by the distance from the edge of spectral element to this boundary, which corresponds to an overlap of 2δ. While it is possible to increase the overlap by including more dofs from neighboring subdomains, common practice for both SEM and FEM applications is to use minimal overlap as this approach tends to be the most efficient.

For tensor product configurations such as illustrated in Fig. 3.6, the local subdomain problems can be solved using fast diagonalization [54]. For a rectilinear $L_x^s \times L_y^s$ domain, the constant-coefficient Poisson solver can be expressed as

$$A_s = B_y^s \otimes A_x^s + A_y^s \otimes B_x^s, \tag{3.70}$$

where (A_x^s, B_x^s) and (A_y^s, B_y^s) are respective (Laplacian, mass)-matrix pairs in each direction. The x and y operators possibly differ because of the subdomain dimensions and/or boundary conditions. For $* = x$ or y, let $[S_*^s, \Lambda_*^s]$ satisfy the generalized eigenvalue problem $A_*^s S_*^s = B_*^s S_*^s \Lambda_*^s$ with normalization $(S_*^s)^T B_*^s S_*^s = I$. Then the solution to the local subdomain problem, $A_s \underline{z}_s = \underline{r}_s$, is

$$\underline{z}_s = (S_y^s \otimes S_x^s) \left[I \otimes \Lambda_x^s + \Lambda_y^s \otimes I \right]^{-1} (S_y^s \otimes S_x^s)^T R_s \underline{r}. \tag{3.71}$$

In d space dimensions, the equivalent of (3.71) requires only $O(N^{d+1})$ operations and $O(N^d)$ memory references, which is particularly efficient in the 3D case. Schwarz preconditioning does not require exact local solves, so (3.71) can be used even if $\overline{\Omega}^s$ is deformed by simply selecting L_x^s and L_y^s to approximate the shape of $\overline{\Omega}^s$.

***p*-Multigrid.** The polynomial spaces of the SEM and p-type finite elements lend themselves to a natural hierarchy for multigrid in which coarser spaces are nested within the original approximation space, $X^{N'} \subset X^N$, $N' < N$. Starting with $\underline{x}_0 = 0$, $\underline{r} := \underline{b} - A\underline{x}_0$, the basic p-multigrid (pMG) strategy to solve $A\underline{x} = \underline{r}$ is based on damped iteration plus a coarse grid correction,

$$\text{For } k = 1, \ldots, \nu_1: \quad \underline{x}_k = \underline{x}_{k-1} + \sigma_k S^{-1}(\underline{r} - A\underline{x}_{k-1}), \tag{3.72}$$

$$\underline{x}_0' = \underline{x}_k + \underline{e}_c, \qquad \underline{e}_c := J_c A_c^{-1} J_c (\underline{b} - A\underline{x}_k), \tag{3.73}$$

$$\text{For } k = 1, \ldots, \nu_2: \quad \underline{x}_k' = \underline{x}_{k-1}' + \sigma_k S^{-1}(\underline{r} - A\underline{x}_{k-1}'). \tag{3.74}$$

Here (3.72) and (3.74) are (pre- and post-) *smoothing steps*, designed to suppress the error components that are not removed by the coarse-grid correction (3.73). Here S is a *smoother*, and $0 < \sigma_k < 1$ is a damping factor. The matrix J_c is an interpolator from a coarse space to the SEM mesh and

$A_c = J_c^T A J_c$ is the coarse-grid matrix. In pMG, $J_c = I_E \otimes J$ is block diagonal (one block per element) with $J = \hat{J} \otimes \cdots \otimes \hat{J}$ being the tensor-product of 1D interpolation operators from $(N'+1)$ GLL points to the $(N+1)$ GLL nodes of the base mesh. Thus, restriction (J_c^T) and prolongation (J_c) are element-local operations. Possible choices for S include diag(A) or the local part (i.e., excluding A_0) of M_{AS} introduced above.[8] The damping factor σ_k can either be a constant (e.g., as considered in [53]) or preferably a sequence of values designed to suppress the oscillatory part of the error, such as generated by Chebyshev smoothing [7,55–57].

The sequence (3.72)–(3.74) represents a two-level multigrid V-cycle. It becomes a full V-cycle by using another multigrid V-cycle to solve the coarse-grid problem, $A_c^{-1} J_c(\underline{b} - A\underline{x}_k)$. One can iterate with the V-cycle or employ the single V-cycle (3.72)–(3.74) as a preconditioner in CG or GMRES.

Solution projection. Turbulent flows contain a significant range of evolving scales. At each time step, the pressure and velocity are computed by iteratively solving a linear system to a tolerance ϵ. For the pressure, one iterates for $k = 1, \ldots,$ until

$$\|\underline{r}_k\| := \|\underline{b}^n - A\underline{p}_k^n\| < \epsilon, \tag{3.75}$$

where ϵ is a fixed tolerance determined by the physics.[9] It is generally useful to solve only for the *change* in the solution, $\delta\underline{p}^n := \underline{p}^n - \underline{p}^{n-1}$, which has the effect of not having to recompute slowly evolving large-scale structures. For any initial guess $\bar{\underline{p}} \approx \underline{p}^n$ we can rewrite the pressure Poisson problem as follows:

$$\textit{Solve} \quad A\,\delta\underline{p} = \underline{b}^n - A\bar{\underline{p}} =: \underline{r}_0 \quad \text{to tolerance } \epsilon, \tag{3.76}$$

$$\textit{Set} \quad \underline{p}^n = \bar{\underline{p}} + \delta\underline{p}. \tag{3.77}$$

The choice $\bar{\underline{p}} := \underline{p}^{n-1} = \underline{p}^n + \Delta t \frac{\partial p}{\partial t} + O(\Delta t^2)$ implies that the initial iteration residual satisfies $\|\underline{r}_0\| = O(\Delta t)$. In cases where the solution is approaching a steady state, one would have $\|\underline{r}_0\| \longrightarrow 0$ such that (3.76) converges in 0 or 1 iterations.

It is possible to significantly improve on the initial $O(\Delta t)$ error by using information from more than just the preceding timestep. While high-order extrapolation is generally unstable, an optimal, stable, approximation $\bar{\underline{p}} \approx \underline{p}^n$ can be found by projection without knowing $\underline{p}^n$ [58]. Let $X_l := [\tilde{\underline{x}}_1 \ldots \tilde{\underline{x}}_l]$

[8] As noted in [19], (4.4.13)–(4.4.15), the diagonal of the SEM Laplacian matrix can be computed directly, without resorting to formation of the local Laplacian matrices, A^e.

[9] If the equations are nondimensionalized with respect to convective timescales then $\epsilon \approx 10^{-4}$ generally suffices for most turbulence problems.

be a matrix of approximation vectors satisfying $\tilde{\underline{x}}_i^T A \tilde{\underline{x}}_j = \delta_{ij}$. Then

$$\bar{\underline{p}} = \sum_{j=1}^{l} \alpha_j \tilde{\underline{x}}_j, \qquad \alpha_j := \tilde{\underline{x}}_j^T \underline{b}^n \tag{3.78}$$

is the best-fit approximation to $\underline{p}^n$ in $\text{span}\{\tilde{\underline{x}}_1, \ldots, \tilde{\underline{x}}_l\} := \mathcal{R}(X_l)$ in the $\|\cdot\|_A$ norm. If $\mathcal{R}(X_l) = \text{span}\{\underline{p}^{n-1} \ldots \underline{p}^{n-l}\}$ is the space spanned by the preceding l solution vectors, then $\|\bar{\underline{p}} - \underline{p}^n\| = O(\Delta t^l) + O(\epsilon)$, where ϵ is the iteration tolerance for (3.76) [59].

Following [58], there are multiple ways to construct the A-orthogonal approximation basis, X_l. A low-storage variant for up to $l_{\max}$ saved solutions proceeds as follows. If $l = 0$ or $l = l_{\max}$, set $l = 1$ and $X_1 = [\tilde{\underline{x}}_1] = [\underline{p}^n / \|\underline{p}^n\|_A]$. Else, after computing $\delta\underline{p}$ (3.76), set

$$\underline{w} = A\delta\underline{p}, \tag{3.79}$$

$$\tilde{\underline{x}}_* = \delta\underline{p} - \sum_{j=1}^{l} \beta_j \tilde{\underline{x}}_j, \qquad \beta_j := \tilde{\underline{x}}_j^T \underline{w}, \tag{3.80}$$

$$\tilde{\underline{x}}_{l+1} = \tilde{\underline{x}}_* / \|\tilde{\underline{x}}_*\|_A, \tag{3.81}$$

$$l = l + 1. \tag{3.82}$$

This algorithm corresponds to a stable two-pass Gram–Schmidt orthogonalization because $\delta\underline{p}$ is A-orthogonal to $\mathcal{R}(X_l)$ to tolerance ϵ.

The same algorithm can be applied to the velocity (and temperature or other scalar fields). For many applications, projection (3.76)–(3.78) yields a two-fold reduction in Navier–Stokes solution time.

3.7 Practical aspects

Here we discuss some practical issues and problems related to the DNS and LES of turbulent flows using high-order methods such as the SEM, and several techniques that have been proven useful for their stabilization. Two main topics are discussed below, one related to stabilization of marginally resolved simulations and the other to instabilities that may arise at outflow boundaries.

3.7.1 Stabilization techniques

For marginally resolved fields, such as encountered in coarse direct and/or large eddy simulations, the SEM suffers dispersion errors for features that are prominent (or oscillatory) at the grid scale, particularly for the transport of scalars that do not benefit from regularization induced by

the pressure/divergence-free constraint imposed on the momentum in the incompressible Navier–Stokes equations. These errors manifest as over- and undershoots in the transported quantities. Based on the work of Godunov [60], any monotone (i.e., nonoscillatory) linear scheme can be only first-order accurate in the grid spacing. Thus, to realize high-order convergence, one must either resolve the solution down to unit grid-Reynolds (or Peclet) number, or resort to nonlinear stabilization. Below, we describe stabilization approaches that can be used for spectral-element based simulations of turbulent flow.

Explicit polynomial filtering. For marginally resolved cases, one can exploit the local polynomial expansions in the SEM to develop low-pass filters capable of mitigating the effects of high-wavenumber energy pile-up. The key idea in explicit filtering is to suppress the spurious modes at the end of each time step. In the method introduced in [61], instead of adding a dissipation term directly to the Navier–Stokes equation, a low-pass filter function built in the modal space is used. Following [62], a 1D solution $u(r)$ can be expanded for $r \in \hat{\Omega} = [-1, 1]$ using a modal bases as

$$u(r) = \sum_{k=0}^{N} \hat{u}_k \phi_k(r), \tag{3.83}$$

where $\phi_0(r) = \frac{1-r}{2}$, $\phi_1(r) = \frac{1+r}{2}$, and $\phi_k(r) = P_k(r) - P_{k-2}(r)$ for $k > 1$, where P_k is the Legendre polynomial of degree k. For $k > 1$, the ϕ_ks are the so-called bubble functions that vanish at $r = \pm 1$. The basis functions become increasingly oscillatory with k and admit construction of a low- or high-pass filter as follows:

$$\bar{u} = \hat{F}(u) = \sum_{k=0}^{N} \sigma_k \hat{u}_k \phi_k(r), \tag{3.84}$$

where the filter-transfer function is defined by the coefficients $\sigma_k \leq 1$.

For a cut-off integer, $k_c \in [2, N-1]$, one can construct a low-pass filter by setting

$$\sigma_k = \begin{cases} 1 & \text{for } k \leq k_c, \\ 1 - \alpha \left(\frac{k-k_c}{N-k_c} \right)^2 & \text{for } k > k_c, \end{cases} \tag{3.85}$$

where $(1-\alpha) \in [0, 1]$ is the amplitude of the Nth component in the filtered function, $\sigma_N \hat{u}_N$. Note that for $k_c > 1$ (3.84)–(3.85) will preserve the boundary values at $r = \pm 1$, which implies that the filtering is localized to the interior of each element in the spectral element case, for any number of space dimensions d. If $\alpha = 1$, the filter exhibits a sharp cut-off in modal (wave) space, which tends to produce oscillatory and inaccurate reconstructions; a filter with a more gentle roll-off is generally preferred [61,63].

The filter (3.85) also has the property that the transfer function is unity for any $k \leq k_c$, so that low (polynomial) modes are untouched, which is another requirement for accurate reconstruction [63]. While there are many filters that meet these requirements, (3.85) has proven to be effective for the relatively low polynomial orders typical of SEM applications. (See [64] for further considerations.)

High-pass filtering. An alternative filter-based stabilization approach can be affected through a high-pass filter (HPF). In this approach, originally developed by Stolz and Schlatter [65,66] to provide subgrid-scale dissipation in LES, the right-hand sides of the Navier–Stokes and scalar equations are modified with the addition of a forcing term which has the form

$$f_{HPF} = -\chi H_N * u, \tag{3.86}$$

where H_N is a high-pass filter that is applied element-by-element and χ is an appropriately-chosen scale factor.

Similar to (3.83)–(3.84), if $u(x) = \sum_{k=0}^{N} \hat{u}_k P_k(x)$ is any polynomial of degree N on $[-1, 1]$, then $\bar{u}(x) = \sum_{k=0}^{N} \sigma_k \hat{u}_k P_k(x)$ will be a low-pass-filtered variant of u if $\sigma_k = 1$ for $k \leq k_c \leq N$ and $\sigma_k < 1$ for $k > k_c$. If F is the matrix operation that applies this one-dimensional low-pass filter, then the three-dimensional low-pass filter on $\hat{\Omega}$ is given by the Kronecker product, $G = F \otimes F \otimes F$, and the high-pass filter is

$$H_N * u = u - G * u = u - (F \otimes F \otimes F)u\,. \tag{3.87}$$

The relaxation term $-\chi H_N * u$ provides the necessary drain of energy out of a coarsely discretized system [67].

For turbulent flows, the high-wavenumber relaxation of the HPF model is similar to the approximate deconvolution (ADM) approach to LES [68]. It is attractive in that it can be tailored to act directly on marginally resolved modes at the grid scale. We remark that the right-hand side in the Navier–Stokes and advection-diffusion equations do not need to be continuous. The transfer functions can therefore be applied directly in Legendre space (i.e., on P_k), rather than in the modal space associated with the bubble functions (ϕ_k), which is required for the explicit filter approach. Moreover, because the HPF is applied to the right-hand side of the Navier–Stokes equations, the solution $\mathbf{u^n}$ remains divergence-free, which is not necessarily the case with explicit filtering.

Artificial viscosity method. A common approach to stabilizing high-order methods is to add either high-order dissipation such as hyperviscosity [69] or some other spatially- or spectrally-localized dissipation mechanism such as spectral vanishing viscosity [70]. If the approach is nonlinear (e.g., the strength of the dissipation is triggered by the solution), then one can retain high-order accuracy with some hope of monotonicity. Steps in this

direction follow the ideas of entropy and residual-based viscosity introduced in [71–73]. In the context of the SEM, an artificial viscosity method (AVM) following these ideas was developed in [74].

The idea behind the AVM is to develop robust and practical adaptive dissipation that controls grid-scale oscillations rather than simply advect them with the resolved solution throughout the domain. The stabilization is performed through a locally enhanced dissipation term, which is triggered by spatial error indicators only so that global temporal errors (which can be smooth) do not lead to increased dissipation. The method preserves high-order accuracy in space and time away from sharp interfaces while minimizing grid-scale oscillations throughout the domain. It has been effective in several applications including recent studies on the effects of Schmidt-number on Boussinesq and low-Mach models of buoyancy-driven transport [75].

3.7.2 Outflow boundary conditions

Turbulent flows can generate vortices of sufficient strength to create a (locally) negative flux at an outflow boundary. For instance, if a vortex travels across the outflow boundary, the solution may blow up since half the vortex has negative flux. The Neumann boundary condition typically used for velocity at the outflow does not specify flow characteristics and a negative flux at the outflow can rapidly lead to numerical instability. Two approaches typically used for the stabilization of negative fluxes at an outflow boundary in the context of SEM are discussed.

Imposing positive divergence. One way to eliminate incoming characteristics is to add a mean streamwise component to the velocity field by imposing a positive divergence to the flow field near the exit, mimicking a supersonic nozzle. The advantage of using a nozzle is that it ensures that the characteristics at the exit point outward under a wide range of flow conditions [36]. In contrast, schemes based on viscous buffer zones require knowledge of the anticipated space and time scales to ensure that vortical structures are adequately damped as they pass through the buffer zone.

The imposition of this boundary condition is performed without change to the mesh geometry. Typically in simulations, a layer of elements adjacent to the outflow is identified and a positive divergence $D(x)$ is imposed that ramps from zero at the upstream end of the layer to a fixed positive value at the exit. Specifically,

$$D(\mathbf{x}) = C\left[1 - (x_\perp / L_\perp)^2\right], \tag{3.88}$$

where $x_\perp$ is the distance normal to the boundary and $L_\perp$ is maximum thickness of the last layer of elements. By integrating the expression for D

from $x_\perp/L_\perp = 1$ to 0, one obtains the net gain in mean velocity over the extent of the layer. The choice of the constant C is made such that the gain is equal to the mean velocity prior to the correction. However, this gain can be increased if stronger fluctuations are encountered.

Stabilized stress-free outflow condition. Another approach that can be used to stabilize a simulation when sufficiently strong negative fluxes occur at an outflow boundary is the approach of [76], which uses a stabilized stress-free condition ensuring that in case of incoming flow/characteristics (i.e., negative velocity at the outflow), a pressure gradient is applied to locally reverse the direction of the flow to provide outflow rather than inflow.

This approach is based on adding the condition

$$\mathbf{u}\cdot\left(-p+\frac{1}{Re}\nabla\mathbf{u}-\frac{1}{2}|\mathbf{u}|^2\right)\cdot\mathbf{n}=0. \tag{3.89}$$

In this way, if an energy influx on an outflow part of the boundary Γ_O is present, the energy growth as $Re\to\infty$ is eliminated. To remove the discontinuity that appears when fluxes turn from negative to positive from one time step to another, a smooth step function is used

$$\Theta(\mathbf{n},\mathbf{u}):=\frac{1}{2}\left[1-\tanh\left(\frac{\mathbf{n}\cdot\mathbf{u}}{U\delta}\right)\right], \tag{3.90}$$

where U is the characteristic velocity scale, and $\delta>0$ is a chosen nondimensional positive constant that controls the amount of smoothness, i.e., as $\delta\to 0$, $\Theta(\mathbf{n},\mathbf{u})$ approaches a step function. The stabilized outflow boundary condition becomes

$$-p\mathbf{n}+\frac{1}{Re}(\mathbf{n}\cdot\nabla\mathbf{u})-\left[\frac{1}{2}|\mathbf{u}|^2\Theta(\mathbf{n},\mathbf{u})\right]\mathbf{n}=\mathbf{0}\quad\text{on }\Gamma_O. \tag{3.91}$$

3.8 Example applications

We present several examples that illustrate the performance considerations discussed in the preceding sections.

Pebble-bed simulations. The first examples correspond to turbulent flow through a cylindrical packed-bed with $S=146$ and $S=1568$ spherical pebbles, as pictured in Fig. 3.7 [57,77,78]. The Reynolds number is $Re_D\approx 2500$, based on sphere diameter, D, and mean flow velocity in the bed, U. The polynomial order for each case is $N=7$, with $(E,n)=(62{,}132,21\,\text{M})$ for $S=146$ and $(524{,}386,180\,\text{M})$ for $S=1568$, where $n=EN^3$ is the approximate number of grid points in each mesh. Time advancement is based on a two-stage 2nd-order characteristics timestepper with CFL $=4$

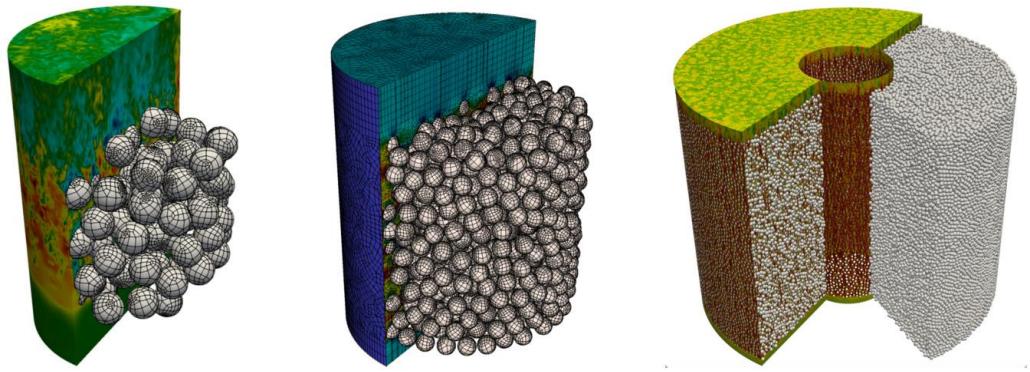

FIGURE 3.7 Pebble-bed configurations with (left to right) $S = 146$, 1568, and 352,625 spherical pebbles and, respectively, $E = 62{,}132$, 524,386, and 98,782,067 elements.

TABLE 3.1 Time-per-step (seconds) for the small pebble-bed cases of Fig. 3.7.

	$S = 146$: $E = 62{,}132$, $N = 7$					$S = 1568$: $E = 524{,}386$, $N = 7$				
P	it	3	6	12	18	it	24	48	72	96
n/P		7.1 M	3.6 M	1.8 M	1.2 M		7.5 M	3.7 M	2.5 M	1.9 M
SEMFEM	11	–	0.367	0.282	0.226	8	–	0.326	–	0.205
Cheby-Jac(2), (7,5,3)	18	0.986	0.492	0.377	0.320	17	0.925	0.526	0.400	0.360
Cheby-ASM(1), (7,3)	8	0.578	0.327	0.239	0.173	9	0.580	0.339	0.294	0.257
Cheby-ASM(2), (7,3)	6	0.556	0.316	0.201	0.190	6	–	0.334	0.262	0.241
Cheby-ASM(3), (7,3)	5	0.560	0.323	0.208	0.172	6	0.559	0.354	0.297	0.255
Cheby-ASM(2), (7,5,3)	5	–	0.308	0.200	0.197	5	–	0.352	0.258	0.220
Cheby-RAS(1), (7,3)	9	0.630	0.320	0.196	0.170	13	0.688	0.407	0.300	0.255
Cheby-RAS(2), (7,3)	7	0.538	0.304	0.191	0.174	9	–	0.405	0.317	0.298
Cheby-RAS(3), (7,3)	6	0.528	0.300	0.189	0.162	7	0.587	0.388	0.276	0.259
Cheby-RAS(2), (7,5,3)	5	–	0.286	0.184	0.155	6	–	0.330	0.270	0.235

($\Delta t = 1 \times 10^{-3}$ for $S = 146$ and $\Delta t = 2.5 \times 10^{-4}$ for $S = 1568$). At each step, the solution is projected onto a space of up to $l = 10$ prior solution vectors using (3.76)–(3.78) to generate a high-quality initial guess for velocity and pressure. An absolute tolerance of 10^{-4} is used for the pressure and velocity solvers. Because the optimal Schwarz preconditioners are non-symmetric, the pressure Poisson problem is solved with GMRES rather than CG [53]. With projection and high-quality preconditioners, the iteration count is low enough that restarted GMRES is not required.

In Table 3.1 we present Navier–Stokes timing data (time-per-step, in seconds) and pressure iteration counts (it) for several preconditioners and processor counts, P, for the $S = 146$ and 1568 cases. The simulations are run on P NVIDIA V100 GPUs on the OLCF Summit supercomputer at Oak Ridge National Laboratory using the spectral element code NekRS [79]. The first preconditioner considered is the low-order preconditioner (SEMFEM), in which A is preconditioned by a sparse finite element operator, A_f, which is approximately inverted by a single algebraic multigrid

(AMG) V-cycle. The remaining preconditioners are based on p-multigrid (pMG), with jth-order Chebyshev smoothing at each level, where j is indicated in the first set of parentheses. The polynomial-order schedule for the smoothing levels, starting with $N = 7$, is indicated in the second set. At the lowest level ($N = 1$), the pMG coarse-grid operator is solved approximately with a single AMG V-cycle. Three smoothing operators are considered: diagonal (Jac), additive Schwarz (ASM), and restricted additive Schwarz (RAS) [47]. All of the preconditioners execute a single V-cycle per outer GMRES iteration.

Because iteration counts can be sensitive to the amount of structure in the flow, the simulations are initialized with a turbulent field (via a restart) prior to collecting timings. In all cases, results are averaged over 2000 timesteps. For a given algorithm/architecture coupling, the critical parameter governing parallel efficiency is the number of grid points per processor, n/P, which is given in Row 2 of the table. The timing breakdown is not given in the table but we remark that the pressure solve consumes about 80% of the solution time so the choice of preconditioner can have a significant impact on the overall performance.

Several features are evident from Table 3.1. First, the relative performance of the preconditioners is a function of the geometry and of the number of processors used. For $S = 146$, Cheby-RAS(2), (7,5,3), is the fastest for all values of P. However, SEMFEM is fastest for $S = 1568$. The next fastest for this case is Cheby-ASM(2), (7,5,3) for $P = 96$, and Cheby-RAS(2), (7,5,3) for $P = 48$. By contrast, SEMFEM is second *slowest* for $S = 146$, $P = 18$. Second, from this table one can estimate parallel efficiency, $\eta := (P_m\, t_m)/(P\, t)$, where P_m is the minimum number of processors that can be used (due to memory constraints) and (t_m, t) is the time-per-step for processor counts (P_m, P). Taking t_m and t to be the minimum time in their corresponding columns on finds a parallel efficiency of ≈ 0.72 when $n/P = 1.8\,\mathrm{M}$ for $S = 148$ and $n/P = 2.5\,\mathrm{M}$ for $S = 1568$. From Table 3.1 one can therefore conclude that users would realize a time-per-step of 0.184 s for $S = 146$ and 0.25 s for $P = 72$. These strong-scale-limit results are similar to those reported in [79,80], where it was found that one needs $n/P \approx 2.1$–$2.8\,\mathrm{M}$ grid points per V100 to yield a parallel efficiency of $\eta \approx 0.8$. Summit has $P = 27{,}648$ V100s, implying one needs $\approx 70\,\mathrm{B}$ grid points to be near performance-saturation when using the entire machine. Indeed, the $S = 352\,\mathrm{K}$ configuration of Fig. 3.7, which comprises 51 B grid points, realizes 76% parallel efficiency on all of Summit [78]. At 0.36 s per timestep, this case requires about six hours of wall-clock time to simulate a single flow-through time, H/U, where H is the domain height.

DNS of a compression stroke. Fig. 3.8 shows the results for DNS of the compression stroke during motored operation of the optically transparent, single-cylinder pentroof engine of TU Darmstadt [81] at an engine speed of 800 rpm with valve operation (intake pressure of 0.4 bar). The aim of

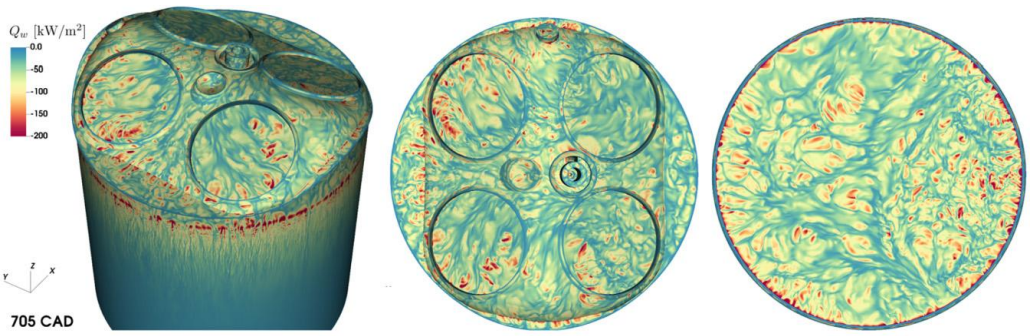

FIGURE 3.8 DNS of compression in an optical engine. Isocontours of heat flux along the cylinder walls at 15° bTDC, left-to-right: bird's eye view, cylinder head, piston.

this ETH Zurich-based study is to investigate the evolution of the momentum and thermal boundary layers [82]. This simulation was performed using the low-Mach approach described in Section 3.5. Because of volume variation due to piston motion during compression an additional ODE was solved for the thermodynamic pressure which varies in time. Fig. 3.8 depicts isocontours of the wall heat flux on the cylinder liner, head and piston surfaces towards the end of compression. A strong correlation between the flow and heat flux structures is observed with finer structures generated as the Reynolds number increases during compression. In agreement with previous observations [83], higher heat fluxes were noted in regions where the flow is predominantly impinging/stagnating, as evidenced by the higher values at the right part of the piston surface compared to the left (Fig. 3.8, right), where the tumble vortex impinges on the piston. For the same reason, the heat flux is higher on the left part of the cylinder head. Significant heat flux values are seen at the entrance of the large crevice volume of the optical engine (Fig. 3.8, left) as a result of the intense, hollow jet flow that is formed in this region. The study of crevice flows is of interest also with respect to unburned fuel and pollutant emissions.

These spectral element simulations were performed using Nek5000 [84] on 802 1.3 GHz Intel KNL nodes (51,328 cores, one MPI rank per core) of the ALCF supercomputer, Theta, at Argonne National Laboratory ($n/P =$ 13.8 K). Second-order, single-stage, characteristics timestepping was used with a target CFL of 2.5. Roughly 40 pressure iterations per step were required when using the Nek5000 default additive Schwarz preconditioner described in [85]. The time-per-step is 2.25 seconds.

References

[1] H. Kreiss, J. Oliger, Comparison of accurate methods for the integration of hyperbolic problems, Tellus 24 (1972) 199–215.

[2] S. Orszag, Spectral methods for problems in complex geometry, J. Comput. Phys. 37 (1980) 70–92.

[3] D. Gottlieb, S. Orszag, Numerical Analysis of Spectral Methods: Theory and Applications, SIAM-CBMS, Philadelphia, 1977.
[4] B. Metivet, Y. Morchoisne, in: Proc. of the 4th GAMM Conf. on Numerical Methods in Fluid Mechanics, Paris, 1981.
[5] A. Patera, A spectral element method for fluid dynamics: laminar flow in a channel expansion, J. Comput. Phys. 54 (1984) 468–488.
[6] R. Anderson, J. Andrej, A. Barker, J. Bramwell, J.-S. Camier, J.C.V. Dobrev, Y. Dudouit, A. Fisher, T. Kolev, W. Pazner, M. Stowell, V. Tomov, I. Akkerman, J. Dahm, D. Medina, S. Zampini, MFEM: a modular finite element library, Comput. Math. Appl. (2020).
[7] M. Kronbichler, K. Ljungkvist, Multigrid for matrix-free high-order finite element computations on graphics processors, ACM Trans. on Par. Comp. 6 (1) (2019) 1–32.
[8] I. Huismann, J. Stiller, J. Fröhlich, Scaling to the stars – a linearly scaling elliptic solver for p-multigrid, J. Comput. Phys. 398 (2019) 108868.
[9] D. Arndt, N. Fehn, G. Kanschat, K. Kormann, M. Kronbichler, P. Munch, W. Wall, J. Witte, ExaDG: high-order discontinuous Galerkin for the exa-scale, in: H. Bungartz, S. Reiz, B. Uekermann, P. Neumann, W. Nagel (Eds.), Software for Exascale Computing – SPPEXA 2016–2019, Springer, 2020, pp. 189–224.
[10] M. Dubiner, Spectral methods on triangles and other domains, J. Sci. Comput. 6 (1991) 345–390.
[11] S. Sherwin, G. Karniadakis, Tetrahedral hp finite elements: algorithms and flow simulations, J. Comput. Phys. 124 (1996) 14–45.
[12] D. Moxey, R. Amici, M. Kirby, Efficient matrix-free high-order finite element evaluation for simplicial elements, SIAM J. Sci. Comput. 42 (3) (2020) C97–C123.
[13] M. Ainsworth, G. Andriamaro, O. Davydov, Bernstein–Bézier finite elements of arbitrary order and optimal assembly procedures, SIAM J. Sci. Comput. 33 (6) (2011) 3087–3109.
[14] M. Ainsworth, S. Jiang, Preconditioning the mass matrix for high order finite element approximation on tetrahedra, SIAM J. Numer. Anal. 43 (1) (2021) A384–A414.
[15] J. Chan, T. Warburton, A short note on a Bernstein-Bezier basis for the pyramid, 2015.
[16] J. Chan, T. Warburton, GPU-accelerated Bernstein–Bézier discontinuous Galerkin methods for wave problems, SIAM J. Sci. Comput. 39 (2) (2017) A628–A654.
[17] J. Malm, P. Schlatter, P. Fischer, D. Henningson, Stabilization of the spectral-element method in convection dominated flows by recovery of skew symmetry, J. Sci. Comput. 57 (2013) 254–277.
[18] M. Ainsworth, Dispersive behaviour of high order finite element schemes for the one-way wave equation, J. Comput. Phys. 259 (2014) 1–10.
[19] M. Deville, P. Fischer, E. Mund, High-Order Methods for Incompressible Fluid Flow, Cambridge University Press, Cambridge, 2002.
[20] J. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, Texts in Appl. Math., vol. 54, Springer, 2008.
[21] Y. Maday, E. Rønquist, Optimal error analysis of spectral methods with emphasis on non-constant coefficients and deformed geometries, Comput. Methods Appl. Mech. Eng. 80 (1990) 91–115.
[22] P. Fischer, M. Min, T. Rathnayake, S. Dutta, T. Kolev, V. Dobrev, J.-S. Camier, M. Kronbichler, T. Warburton, K. Swirydowicz, J. Brown, Scalability of high-performance PDE solvers, Int. J. High Perform. Comput. Appl. 34 (5) (2020) 562–586.
[23] C. Bernardi, Y. Maday, Approximations spectrales de problèmes aux limites elliptiques, Springer, Paris, 1992.
[24] O. Pironneau, On the transport-diffusion algorithm and its applications to the Navier–Stokes equations, Numer. Math. 38 (1982) 309–332.
[25] Y. Maday, A. Patera, E. Rønquist, An operator-integration-factor splitting method for time-dependent problems: application to incompressible fluid flow, J. Sci. Comput. 5 (1990) 263–292.

[26] P. Gresho, S. Chan, R. Lee, C. Upson, A modified finite element method for solving the time-dependent, incompressible Navier–Stokes equations, Int. J. Numer. Methods Fluids 4 (1984) 557–598.
[27] S. Patel, P. Fischer, M. Min, A. Tomboulides, A characteristic-based, spectral element method for moving-domain problems, J. Sci. Comput. 79 (2019) 564–592.
[28] P. Fischer, E. Rønquist, D. Dewey, A. Patera, Spectral element methods: algorithms and architectures, in: Proc. of the First Int. Conf. on Domain Decomposition Methods for Partial Differential Equations, SIAM, 1988, pp. 173–197.
[29] G. Strang, G. Fix, An Analysis of the Finite Element Method, Prentice-Hall Series in Automatic Computation, Prentice-Hall, Englewood Cliffs, NJ, 1973.
[30] C. Bernardi, Y. Maday, A collocation method over staggered grids for the Stokes problem, Int. J. Numer. Methods Fluids 8 (1988) 537–557.
[31] Y. Maday, A. Patera, Spectral element methods for the Navier–Stokes equations, in: A. Noor, J. Oden (Eds.), State-of-the-Art Surveys in Computational Mechanics, ASME, New York, 1989, pp. 71–143.
[32] Y. Maday, A. Patera, E. Rønquist, The $\mathbb{P}_N \times \mathbb{P}_{N-2}$ method for the approximation of the Stokes problem, Tech. Rep. 92009, Department of Mechanical Engineering, MIT, Cambridge, MA, 1992.
[33] S. Orszag, M. Israeli, M. Deville, Boundary conditions for incompressible flows, J. Sci. Comput. 1 (1986) 75–111.
[34] G. Karniadakis, M. Israeli, S. Orszag, High-order splitting methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 97 (1991) 414–443.
[35] A. Tomboulides, M. Israeli, G. Karniadakis, Efficient removal of boundary-divergence errors in time-splitting methods, J. Sci. Comput. 4 (1989) 291–308.
[36] P. Fischer, F. Loth, S. Lee, D. Smith, H. Bassiouny, Simulation of high Reynolds number vascular flows, Comput. Methods Appl. Mech. Eng. 196 (2007) 3049–3060.
[37] A. Tomboulides, J. Lee, S. Orszag, Numerical simulation of low Mach number reactive flows, J. Sci. Comput. 12 (June 1997) 139–167.
[38] A. Tomboulides, S. Orszag, A quasi-two-dimensional benchmark problem for low Mach number compressible codes, J. Comput. Phys. 146 (1998) 691–706.
[39] B.T. Chu, X. Kovasznay, Non-linear interactions in a viscous heat conducting compressible gas, J. Fluid Mech. 3 (5) (1958) 494–514.
[40] R.G. Rehm, H.R. Baum, Equations of motion for thermally driven, buoyant flows, J. Res. Natl. Bur. Stand. 83 (3) (1978) 97–308.
[41] A. Majda, J. Sethian, The derivation and numerical solution of the equations for zero Mach number combustion, Combust. Sci. Technol. 42 (3–4) (1985) 185–205.
[42] M. Deville, E. Mund, Chebyshev pseudospectral solution of second-order elliptic equations with finite element preconditioning, J. Comput. Phys. 60 (1985) 517–533.
[43] C. Canuto, A. Quarteroni, Preconditioned minimal residual methods for Chebyshev spectral calculations, J. Comput. Phys. 60 (1985) 315–337.
[44] C. Canuto, P. Gervasio, A. Quarteroni, Finite-element preconditioning of G-NI spectral methods, SIAM J. Sci. Comput. 31 (2010) 4422–44251.
[45] P. Bello-Maldonado, P. Fischer, Scalable low-order finite element preconditioners for high-order spectral element Poisson solvers, SIAM J. Sci. Comput. 41 (2019) S2–S18.
[46] M. Dryja, O. Widlund, An additive variant of the Schwarz alternating method for the case of many subregions, Tech. Rep. TR 339, Courant Inst., NYU, Dept. Comp. Sci., 1987.
[47] X. chuan Cai, M. Sarkis, A restricted additive Schwarz preconditioner for general sparse linear systems, SIAM J. Sci. Comput. 21 (1999) 792–797.
[48] M. Dryja, O.B. Widlund, An additive Schwarz algorithm for two- and three-dimensional finite element elliptic problems, in: T.F. Chan, R. Glowinski, J. Périaux, O.B. Widlund (Eds.), Domain Decomposition Methods, SIAM, 1989.
[49] M. Dryja, O.B. Widlund, Domain decomposition algorithms with small overlap, SIAM J. Sci. Comput. 15 (1994) 604–620.

[50] H. Tufo, P. Fischer, Fast parallel direct solvers for coarse-grid problems, J. Parallel Distrib. Comput. 61 (2001) 151–177.
[51] S. Pahl, Schwarz type domain decomposition methods for spectral element discretizations, Master's thesis, Univ. of Witwatersrand, Johannesburg, South Africa, Dept. of Computational and Applied Math., 1993.
[52] P. Fischer, An overlapping Schwarz method for spectral element solution of the incompressible Navier-Stokes equations, J. Comput. Phys. 133 (1997) 84–101.
[53] J.W. Lottes, P.F. Fischer, Hybrid multigrid/Schwarz algorithms for the spectral element method, J. Sci. Comput. 24 (2005) 45–78.
[54] R. Lynch, J. Rice, D. Thomas, Direct solution of partial difference equations by tensor product methods, Numer. Math. 6 (1964) 185–199.
[55] H. Sundar, G. Stadler, G. Biros, Comparison of multigrid algorithms for high-order continuous finite element discretizations, Numer. Linear Algebra Appl. 22 (4) (2015) 664–680.
[56] M. Adams, M. Brezina, J. Hu, R. Tuminaro, Parallel multigrid smoothing: polynomial versus Gauss Seidel, J. Comput. Phys. 188 (2) (2003) 593–610.
[57] M. Phillips, S. Kerkemeier, P. Fischer, Tuning spectral element preconditioners for parallel scalability on GPUs, in: Proc. of the 2022 SIAM Conf. on Par. Proc. for Sci. Comp., SIAM, 2022, pp. 37–48.
[58] P. Fischer, Projection techniques for iterative solution of $A\underline{x} = \underline{b}$ with successive right-hand sides, Tech. Rep. 93–90, ICASE, Hampton,VA, 1993.
[59] T. Chan, W. Wan, Analysis of projection methods for solving linear systems with multiple right-hand sides, SIAM J. Sci. Comput. 18 (6) (1997) 1698–1721.
[60] S. Godunov, A difference scheme for numerical solution of discontinuous solution of hydrodynamic equations, Math. Sb. 47 (1959) 271–306.
[61] P. Fischer, J. Mullen, Filter-based stabilization of spectral element methods, C. R. Acad. Sci., Sér. I: Anal. Numér. 332 (2001) 265–270.
[62] J. Boyd, Two comments on filtering for Chebyshev and Legendre spectral and spectral element methods, J. Comput. Phys. 143 (1998) 283–288.
[63] J. Hesthaven, S. Gottlieb, D. Gottlieb, Spectral Methods for Time-Dependent Problems, Cambridge University Press, Cambridge, UK, 2007.
[64] J. Hesthaven, R. Kirby, Filtering in Legendre spectral methods, Math. Comput. 77 (2008) 1425–1452.
[65] S. Stolz, P. Schlatter, L. Kleiser, High-pass filtered eddy-viscosity models for large-eddy simulations of transitional and turbulent flow, Phys. Fluids 17 (6) (2005) 065103.
[66] P. Schlatter, S. Stolz, L. Kleiser, Analysis of the sgs energy budget for deconvolution- and relaxation-based models in channel flow, in: Direct and Large-Eddy Simulation VI, Springer, Dordrecht, 2006, pp. 135–142.
[67] P. Schlatter, S. Stolz, L. Kleiser, Les of transitional flows using the approximate deconvolution model, in: Turbulence and Shear Flow Phenomena (TSFP-3), Int. J. Heat Fluid Flow 25 (3) (2004) 549–558.
[68] S. Stolz, N.A. Adams, L. Kleiser, An approximate deconvolution model for large-eddy simulation with application to incompressible wall-bounded flows, Phys. Fluids 13 (4) (2001) 997–1015.
[69] V. Borue, S. Orszag, Self-similar decay of three-dimensional homogeneous turbulence with hyperviscosity, Phys. Rev. E 51 (2) (1995) R856–R859.
[70] B. Guo, H. Ma, E. Tadmor, Spectral vanishing viscosity method for nonlinear conservation laws, SIAM J. Numer. Anal. 39 (4) (2001) 1254–1268.
[71] J.-L. Guermond, R. Pasquetti, B. Popov, Entropy viscosity for conservation equations, in: J.C.F. Pereira, A. Sequeira (Eds.), Proceedings of ECCOMAS CFD 2010 Conference, Lisbon, Portugal, 2010.
[72] M. Nazarov, Convergence of a residual based artificial viscosity finite element method, Comput. Math. Appl. 65 (4) (2013) 616–626.

[73] M. Nazarov, A. Larcher, Numerical investigation of a viscous regularization of the Euler equations by entropy viscosity, Comput. Methods Appl. Mech. Eng. 317 (2017) 128–152.
[74] L. Lu, M. Nazarov, P. Fischer, Nonlinear artificial viscosity for spectral element methods, C. R. Acad. Sci., Sér. I: Anal. Numér. 357 (2019) 646–654.
[75] J. Lai, E. Merzari, Y. Hassan, Sensitivity analyses in a buoyancy-driven closed system with high resolution CFD using Boussinesq approximation and variable density models, Int. J. Heat Fluid Flow 75 (2019) 1–13.
[76] S. Dong, G. Karniadakis, C. Chryssostomidis, A robust and accurate outflow boundary condition for incompressible flow simulations on severely truncated unbounded domains, J. Comput. Phys. 261 (2014) 83–105.
[77] Y.-H. Lan, P. Fischer, E. Merzari, M. Min, All-hex meshing strategies for densely packed spheres, in: Proceedings of the 29th International Meshing Roundtable, 2021, pp. 293–305.
[78] P. Fischer, E. Merzari, M. Min, S. Kerkemeier, Y. Lan, M. Phillips, T. Rathnayake, A. Novak, D. Gaston, N. Chalmers, T. Warburton, Highly optimized full-core reactor simulations on Summit, arXiv preprint, arXiv:2110.01716, 2021.
[79] P. Fischer, S. Kerkemeier, M. Min, Y. Lan, M. Phillips, T. Rathnayake, E. Merzari, A. Tomboulides, A. Karakus, N. Chalmers, T. Warburton, NekRS, a GPU-accelerated spectral element Navier–Stokes solver, arXiv preprint, arXiv:2104.05829, 2021.
[80] A. Abdelfattah, V. Barra, N. Beams, R. Bleile, J. Brown, J.-S. Camier, R. Carson, N. Chalmers, V. Dobrev, Y. Dudouit, P. Fischer, A. Karakus, S. Kerkemeier, T. Kolev, Y.-H. Lan, E. Merzari, M. Min, M. Phillips, T. Rathnayake, R. Rieben, T. Stitt, A. Tomboulides, S. Tomov, V. Tomov, A. Vargas, T. Warburton, K. Weiss, GPU algorithms for efficient exascale discretizations, Parallel Comput. 108 (2021) 102841.
[81] E. Baum, B. Peterson, B. Böhm, A. Dreizler, On the validation of LES applied to internal combustion engine flows: Part 1: comprehensive experimental database, Flow Turbul. Combust. 92 (2) (2014) 269–297.
[82] G.K. Giannakopoulos, K. Keskinen, J. Koch, M. Bolla, C.E. Frouzakis, Y.M. Wright, K. Boulouchos, M. Schmidt, B. Böhm, A. Dreizler, Characterizing the evolution of boundary layers in IC engines by combined laser-optical diagnostics, direct numerical and large-eddy simulations, Flow Turbul. Combust. (2022), submitted for publication.
[83] M. Schmitt, C. Frouzakis, Y. Wright, A. Tomboulides, K. Boulouchos, Direct numerical simulation of the compression stroke under engine relevant conditions: local wall heat flux distribution, Int. J. Heat Mass Transf. 91 (2015) 948–960.
[84] P. Fischer, J. Lottes, S. Kerkemeier, Nek5000: open source spectral element CFD solver, http://nek5000.mcs.anl.gov, https://github.com/nek5000/nek5000, 2008.
[85] P. Fischer, J. Lottes, Hybrid Schwarz-multigrid methods for the spectral element method: extensions to Navier–Stokes, in: R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund, J. Xu (Eds.), Domain Decomposition Methods in Science and Engineering Series, Springer, Berlin, 2004.