Journal of Computational Physics 229 (2010) 3989–4016

Contents lists available at ScienceDirect

Journal of Computational Physics

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

A first-order system approach for diffusion equation. II: Unification of advection and diffusion

Hiroaki Nishikawa [*]

National Institute of Aerospace, 100 Exploration Way, Hampton, VA 23666, USA

ARTICLE INFO

Article history:
Received 16 July 2009
Received in revised form 20 October 2009
Accepted 26 October 2009
Available online 1 November 2009

Keywords:
Advection–diffusion
Hyperbolic system
Fast convergence
$O(h)$ time step
Cell Reynolds number
Boundary layer
Unstructured grids

ABSTRACT

In this paper, we *unify advection and diffusion into a single hyperbolic system* by extending the first-order system approach introduced for the diffusion equation [J. Comput. Phys., 227 (2007) 315–352] to the advection–diffusion equation. Specifically, we construct a unified hyperbolic advection–diffusion system by expressing the diffusion term as a first-order hyperbolic system and simply adding the advection term to it. Naturally then, we develop upwind schemes for this *entire* system; there is thus no need to develop two different schemes, i.e., advection and diffusion schemes. We show that numerical schemes constructed in this way can be automatically uniformly accurate, allow $O(h)$ time step, and compute the solution gradients (viscous stresses/heat fluxes for the Navier–Stokes equations) simultaneously to the same order of accuracy as the main variable, for *all Reynolds numbers*. We present numerical results for boundary-layer type problems on non-uniform grids in one dimension and irregular triangular grids in two dimensions to demonstrate various remarkable advantages of the proposed approach. In particular, we show that the schemes solving the first-order advection–diffusion system give a tremendous speed-up in CPU time over traditional scalar schemes despite the additional cost of carrying extra variables and solving equations for them. We conclude the paper with discussions on further developments to come.

1. Introduction

In this paper, we unify advection and diffusion into a single hyperbolic system by extending the first-order system approach introduced for the diffusion equation in [1] to the advection–diffusion equation. We show that advection and diffusion terms can be very naturally integrated into a single hyperbolic system, and that numerical schemes constructed for the hyperbolic system will have remarkable advantages, including $O(h)$ time step, uniform accuracy, accurate solution gradients for *all Reynolds numbers*. There is no need to develop two different schemes, i.e., advection and diffusion schemes, for the advection–diffusion equation.

1.1. First-order system approach for diffusion

In the first-order system approach [1], we compute a steady state solution of the diffusion equation,

$$u_t = \nu u_{xx}, \tag{1.1}$$

[☆] DOI of original article: 10.1016/j.jcp.2007.07.029
[*] Tel.: +1 757 864 7244.
E-mail address: hiro@nianet.org

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.10.040

where $\nu > 0$, by integrating in time the following first-order *hyperbolic* diffusion system:

$$\begin{aligned} u_t &= \nu p_x, \\ p_t &= (u_x - p)/T_r, \end{aligned} \tag{1.2}$$

where p is a variable that approaches the solution gradient at the time scale of $T_r(> 0)$. This system is known as a system of hyperbolic heat equations [2–4]; it is equivalent to the diffusion equation (1.1) in the limit, $T_r \to 0$ (so that p relaxes to u_x rapidly and $p \to u_x$ at any instant of time). T_r is often called the relaxation time. There have been many attempts to develop numerical methods for such relaxation systems [2,5–8], often with a particular focus on the stiff source term: an explicit time step, $\Delta t = O(T_r) \to 0$, is prohibitively restricted due to the extremely small relaxation time; an implicit treatment of the stiff source term could degrade the solution accuracy [9]. Our first-order system approach is different from these relaxation methods in that we use the first-order diffusion system specifically for computing a steady state solution of the diffusion equation (1.1). The key idea is that the first-order hyperbolic diffusion system (1.2) is equivalent to the original diffusion equation in the steady state for *arbitrary* T_r. Hence, T_r does not have to be small; the stiffness is not an issue for steady state computations. The system is hyperbolic, having the eigenvalues,

$$-\sqrt{\frac{\nu}{T_r}}, \quad \sqrt{\frac{\nu}{T_r}}, \tag{1.3}$$

which are real for any positive T_r. Hence, we simply apply an advection scheme and march in time until the solution stops changing, with T_r chosen specifically for accelerating the convergence towards the steady state. We have shown in [1] that numerical schemes derived from this approach allow $O(h)$ time step (instead of $O(h^2)$ time step which is typical to diffusion schemes) and converge very rapidly towards a steady state, simultaneously computing the solution gradient to the same order of accuracy as the main variable. We now extend the first-order system approach to the advection–diffusion equation.

1.2. *Unification of advection and diffusion*

Consider the advection–diffusion equation,

$$u_t + a u_x = \nu u_{xx}, \tag{1.4}$$

where $a > 0$ and $\nu > 0$. This equation is typically viewed as a sum of advection and diffusion, and numerical schemes are generally constructed by adding a diffusion scheme to an advection scheme. However, in some cases, such a simple construction is known to destroy the formal accuracy of the two schemes, resulting in a lower order scheme; it requires a very careful tuning of the balance between the two schemes of different nature [10–13]. In this paper, we avoid this problem by solving the following first-order advection–diffusion system:

$$\begin{aligned} u_t + a u_x &= \nu p_x, \\ p_t &= (u_x - p)/T_r, \end{aligned} \tag{1.5}$$

which is obtained simply by adding the advection term to the hyperbolic diffusion system (1.2). The system, of course, remains hyperbolic; it has now the following eigenvalues,

$$\frac{1}{2}\left[a - \sqrt{a^2 + \frac{4\nu}{T_r}}\right], \quad \frac{1}{2}\left[a + \sqrt{a^2 + \frac{4\nu}{T_r}}\right]. \tag{1.6}$$

It is striking that the balance between advection and diffusion is automatically embedded in a single hyperbolic system as it manifests itself in the expression of the eigenvalues. We have just unified advection and diffusion, *in the differential level*, into a single hyperbolic system. Naturally, we then simply consider developing upwind schemes for the entire advection–diffusion system; it is no longer necessary to develop advection and diffusion schemes separately and carefully combine them. Numerical schemes constructed in this way will have, for example, the following advantages over traditional schemes:

- Rapid convergence towards a steady state with $O(h)$ time step for *all Reynolds numbers*.
- Uniform accuracy over *all Reynolds numbers*.
- Solution gradients can be computed simultaneously to the *equal* order of accuracy as the main variable.

We emphasize that these are direct consequences of solving the first-order hyperbolic differential system, not specific to a particular discretization method. Any numerical schemes which discretize the first-order advection–diffusion system consistently, accurately, and stably are expected to have these advantages. In this paper, we demonstrate these features in one and two dimensions by a representative discretization method on non-uniform grids.

1.3. Outline

In the next section, we begin by analyzing the one-dimensional first-order advection–diffusion system. In particular, we show that the time scale, T_r, and the associated length scale, L_r, can be determined in the differential level by optimizing the system for a fast convergence towards a steady state. Having defined the first-order advection–diffusion system completely in the differential level, we discuss the $O(h)$ time step property and its implication on the number of iterations and the CPU time to reach a steady state. Next, we construct a compact second-order upwind scheme for the first-order advection–diffusion system. The accuracy of the scheme is discussed in relation to the positivity of the scheme in the steady state. We also show that the scheme can be implemented in the form of a finite-volume scheme. In Section 3, we extend the one-dimensional analysis to two dimensions, and develop a multidimensional upwind scheme for unstructured triangular grids. In Section 4, we present numerical results to demonstrate remarkable advantages of the proposed method, for both one-dimensional and two-dimensional boundary-layer type problems, including fully irregular triangular grids. Finally, in Section 5, we discuss further developments to come.

2. One dimension

2.1. First-order advection–diffusion system

Consider the one-dimensional advection–diffusion problem,

$$u_t + au_x = \nu u_{xx} \quad \text{in } \Omega = (0,1), \tag{2.1}$$

where $u(0)$ and $u(1)$ are given as boundary conditions, a is a positive advection speed, and ν is a positive diffusion coefficient. To compute the steady state solution to this problem, we propose to solve instead the following first-order system,

$$\mathbf{U}_t + \mathbf{A}\mathbf{U}_x = \mathbf{Q}, \tag{2.2}$$

where

$$\mathbf{U} = \begin{bmatrix} u \\ p \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} a & -\nu \\ -1/T_r & 0 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} 0 \\ -p/T_r \end{bmatrix}, \tag{2.3}$$

where T_r is a free parameter. This system is hyperbolic since $\mathbf{A}$ has real eigenvalues,

$$\lambda_1 = \frac{a}{2}\left[1 - \sqrt{1 + \frac{4\nu}{a^2 T_r}}\right], \quad \lambda_2 = \frac{a}{2}\left[1 + \sqrt{1 + \frac{4\nu}{a^2 T_r}}\right], \tag{2.4}$$

and linearly independent right-eigenvectors which are written in the matrix form as

$$\mathbf{R} = \begin{bmatrix} -\lambda_1 T_r & -\lambda_2 T_r \\ 1 & 1 \end{bmatrix}. \tag{2.5}$$

It is easy to show (simply by setting the time derivatives to be zero) that this system is equivalent to the advection–diffusion equation (2.1) in the steady state for *arbitrary* T_r. In [1], for pure diffusion problems, we defined T_r such that the entire right hand side of the system proportional to ν, thereby making the transient behavior of the solution independent of ν. Here, we take a more general approach. We define T_r as the ratio of a length scale, denoted by L_r, to the characteristic wave speed of the system, thus equalizing the relaxation time scale and the characteristic time scale to enhance the convergence towards the steady state. That is, we set

$$T_r = \frac{L_r}{\lambda_2} = \frac{L_r}{\frac{a}{2}\left[1 + \sqrt{1 + \frac{4\nu}{a^2 T_r}}\right]}, \tag{2.6}$$

where λ_2 has been chosen (instead of λ_1) to keep the speed positive in both advection and diffusion limits. Solving this equation for T_r, we obtain

$$T_r = \frac{L_r}{a + \nu/L_r}. \tag{2.7}$$

In the diffusion limit ($a \to 0$), this reduces precisely to the form of T_r defined in [1]: $T_r = L_r^2/\nu$. It follows from this that the characteristic speed in the diffusion limit, denoted by a_d, can be expressed as

$$a_d \equiv \pm\sqrt{\frac{\nu}{T_r}} = \pm\frac{\nu}{L_r}. \tag{2.8}$$

We now substitute (2.7) back into the eigenvalues (2.4) to find

$$\lambda_1 = -\frac{a}{Re_{L_r}}, \quad \lambda_2 = a\left(1 + \frac{1}{Re_{L_r}}\right), \tag{2.9}$$

where we have introduced the Reynolds number associated with L_r,

$$Re_{L_r} \equiv \frac{aL_r}{\nu}. \tag{2.10}$$

Observe that as $Re_{L_r} \to 0$, the eigenvalues reduce to the diffusion characteristic speeds (2.8), while as $Re_{L_r} \to \infty$, they approach 0 and a, implying scalar advection. In fact, the Reynolds number, Re_{L_r}, is exactly the ratio of the pure advection speed to the diffusion speed:

$$Re_{L_r} = \frac{aL_r}{\nu} = \frac{a}{\nu/L_r} = \frac{a}{|a_d|}. \tag{2.11}$$

This is the key dimensionless parameter in the first-order advection–diffusion system that describes the balance between advection and diffusion. The right-eigenvector matrix (2.5) can now be simplified and written in terms of Re_{L_r}:

$$\mathbf{R} = \begin{bmatrix} \dfrac{L_r}{Re_{L_r}+1} & -L_r \\ 1 & 1 \end{bmatrix}. \tag{2.12}$$

The corresponding left-eigenvector matrix is given by

$$\mathbf{L} = \mathbf{R}^{-1} = \frac{1}{L_r}\frac{Re_{L_r}+1}{Re_{L_r}+2}\begin{bmatrix} 1 & L_r \\ -1 & \dfrac{L_r}{Re_{L_r}+1} \end{bmatrix}. \tag{2.13}$$

It is insightful to look at a wave structure in a Riemann problem; a typical wave structure is shown in Fig. 1. Generally, two waves are created at the interface: left-moving and right-moving waves corresponding to the wave speeds given by (2.9). In the advection limit, the left-moving wave approaches the t-axis while the right-moving wave becomes the pure advection wave (the dotted line). In the diffusion limit, the left- and right-moving waves form a symmetric wave structure with the same wave speed of opposite sign. Note that right-moving wave is always faster than the pure advection wave while the left-moving wave is always slower than the pure advection wave.

We gain further insight by considering the decomposition of $\mathbf{A}$:

$$\mathbf{A} = \mathbf{R}\mathbf{\Lambda}\mathbf{L} = \lambda_1\mathbf{\Pi}_1 + \lambda_2\mathbf{\Pi}_2, \tag{2.14}$$

where

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}, \tag{2.15}$$

$$\mathbf{\Pi}_1 = \mathbf{r}_1\ell_1 = \begin{bmatrix} \dfrac{1}{Re_{L_r}+2} & \dfrac{L_r}{Re_{L_r}+2} \\ \dfrac{Re_{L_r}+1}{Re_{L_r}+2}\dfrac{1}{L_r} & \dfrac{Re_{L_r}+1}{Re_{L_r}+2} \end{bmatrix}, \tag{2.16}$$

$$\mathbf{\Pi}_2 = \mathbf{r}_2\ell_2 = \begin{bmatrix} \dfrac{Re_{L_r}+1}{Re_{L_r}+2} & \dfrac{-L_r}{Re_{L_r}+2} \\ -\dfrac{Re_{L_r}+1}{Re_{L_r}+2}\dfrac{1}{L_r} & \dfrac{1}{Re_{L_r}+2} \end{bmatrix}, \tag{2.17}$$

and $\mathbf{r}_k$ and $\boldsymbol{\ell_k}$ are the k-th column of $\mathbf{R}$ (k-th right-eigenvector) and the k-th row of $\mathbf{L}$ (k-th left-eigenvector), respectively. The matrices, $\boldsymbol{\Pi}_1$ and $\boldsymbol{\Pi}_2$, are the projection matrices which project the system (or a solution change) onto the corresponding subspaces: the left-running and right-running waves, respectively. Naturally, they have the following properties:

$$\mathbf{\Pi}_1\mathbf{\Pi}_1 = \mathbf{\Pi}_1, \quad \mathbf{\Pi}_2\mathbf{\Pi}_2 = \mathbf{\Pi}_2, \quad \mathbf{\Pi}_1\mathbf{\Pi}_2 = \mathbf{0}. \tag{2.18}$$

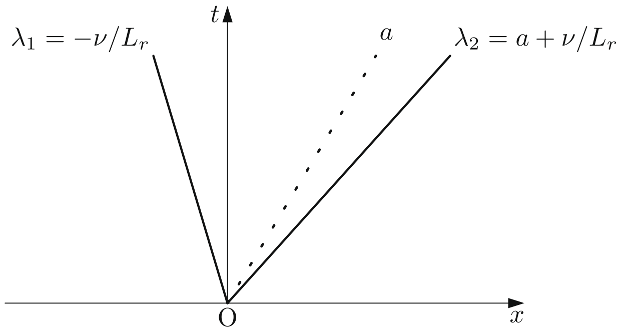


Fig. 1. A typical wave structure for the hyperbolic advection–diffusion system in a Riemann problem. The dotted line indicates a reference pure advection wave (the advection limit of the right-moving wave).

It is interesting that these projection matrices can be expressed as, for both $i = 1$ and 2,

$$\mathbf{\Pi}_i = \frac{Re_{L_r}}{Re_{L_r}+2}\mathbf{\Pi}_i^A + \frac{2}{Re_{L_r}+2}\mathbf{\Pi}_i^D, \tag{2.19}$$

where $\mathbf{\Pi}_i^A$ and $\mathbf{\Pi}_i^D$ are the projection matrices in the advection and diffusion limits:

$$\mathbf{\Pi}_i^A = \lim_{Re_{L_r}\to\infty} \mathbf{\Pi}_i, \quad \mathbf{\Pi}_i^D = \lim_{Re_{L_r}\to 0} \mathbf{\Pi}_i. \tag{2.20}$$

This means that each subspace is a linear combination of its own limits: pure advection and diffusion. This suggests that we may construct an advection–diffusion scheme by combining a pure advection scheme and a pure diffusion scheme by using the weights as in (2.19). This type of construction may be useful in applications to more complex equations, and will be investigated in future. Here, we do not consider such a construction. We rather consider developing a scheme for the entire advection–diffusion system (2.2); such a linear combination emerges as a result.

The time scale T_r has now been clearly determined; it is the length scale L_r that is a free parameter. We determine L_r in the next section such that the first-order advection–diffusion system is made further suited for steady state computations.

2.2. Length scale L_r

In the previous study [1], for pure diffusion problems, the length scale L_r was chosen to optimize a given numerical scheme in terms of error damping or propagation. Here, we consider determining L_r in the differential level, i.e., solely based on the character of the first-order advection–diffusion system. Specifically, we shall choose L_r to minimize a measure of the stiffness of the system, thereby reaching the steady state as quickly as possible. Consider a Fourier mode of phase angle (or nondimensional wave number) $\beta \in [0,\pi]$,

$$\mathbf{U}^\beta = e^{i\beta x/h}\mathbf{U}_0, \tag{2.21}$$

where $\mathbf{U}^\beta = (u^\beta, p^\beta), i = \sqrt{-1}, \ \mathbf{U}_0 = (u_0, p_0)$, and h may be considered as a mesh size of a computational grid, so that the Fourier mode can be taken as a discrete mode on the computational grid with the smoothest mode given by $\beta = \pi h$. Inserting this into the advection–diffusion system (2.2), we obtain

$$\frac{d\mathbf{U}^\beta}{dt} = \mathbf{M}\mathbf{U}^\beta, \tag{2.22}$$

where

$$\mathbf{M} = \begin{bmatrix} -\frac{ia\beta}{h} & \frac{i\nu\beta}{h} \\ \frac{i\beta}{hT_r} & -\frac{1}{T_r} \end{bmatrix}. \tag{2.23}$$

The eigenvalues of this matrix are given by

$$\lambda_{1,2}^M = -\frac{1}{2}\left[\left(\frac{1}{T_r}+\frac{ia\beta}{h}\right) \pm \sqrt{\left(\frac{1}{T_r}-\frac{ia\beta}{h}\right)^2 - \frac{4\nu\beta^2}{h^2T_r}}\right]. \tag{2.24}$$

If these are complex conjugate, the system will be perfectly conditioned because they will have the same propagation speed (magnitude of the imaginary part) and damping factor (the real part). For the diffusion system, this is possible, but not for the advection–diffusion system since we have in the advection limit $(\nu \to 0)$

$$\lambda_{1,2}^M \to -\frac{ia\beta}{h}, \quad -\frac{1}{T_r}, \tag{2.25}$$

which are pure imaginary and pure real. To deal with this mixed case, borrowing the idea from local preconditioning techniques [14,15], we consider equalizing the magnitude of the eigenvalues:

$$\frac{|\lambda_1^M|}{|\lambda_2^M|} = 1, \tag{2.26}$$

i.e., equalize the combined effect of propagation and damping. Solving this for T_r, we find two solutions: negative and positive. The positive solution is given by

$$T_r = -\frac{\nu}{a^2} + \frac{1}{a}\sqrt{\left(\frac{\nu}{a}\right)^2 + \left(\frac{h}{\beta}\right)^2}. \tag{2.27}$$

We then set $\beta = \pi h$ to enforce the condition (2.26) for the most persistent (smoothest) error mode,

$$T_r = -\frac{\nu}{a^2} + \frac{1}{a}\sqrt{\left(\frac{\nu}{a}\right)^2 + \frac{1}{\pi^2}}. \tag{2.28}$$

Now, equating this with (2.7) and solving for L_r, we again find two solutions:

$$L_r = \frac{1}{2}\left[-\left(\frac{\nu}{a}\right) + \sqrt{\left(\frac{\nu}{a}\right)^2 + \left(\frac{1}{\pi}\right)^2} \pm \sqrt{-2\left(\frac{\nu}{a}\right)^2 + \left(\frac{1}{\pi}\right)^2 + 2\left(\frac{\nu}{a}\right)\sqrt{\left(\frac{\nu}{a}\right)^2 + \left(\frac{1}{\pi}\right)^2}}\right]. \tag{2.29}$$

The positive solution can be written as

$$L_r = \frac{1}{2\pi}\left[\frac{Re_\pi}{\sqrt{1+Re_\pi^2}+1} + \sqrt{1+\frac{2}{\sqrt{1+Re_\pi^2}+1}}\right], \tag{2.30}$$

where

$$Re_\pi \equiv \frac{a(1/\pi)}{\nu}. \tag{2.31}$$

This is the length scale that yields the perfect conditioning for the entire range of the Reynolds number, Re_π. Fig. 2 shows a plot of the optimal L_r versus Re_π. We observe that the variation of L_r is confined within a narrow region, approximately $0.01 \leqslant Re_\pi \leqslant 10$. Fig. 3 shows the condition number, $K = |\lambda_1^M|/|\lambda_2^M|$, for the optimal formula (2.30) in comparison with a constant value, $L_r = \frac{1}{\pi}$, which is the optimal value in the advection limit. Clearly, the optimal formula yields the perfect conditioning of $K = 1$ while the other choice introduces non-optimal conditioning over the intermediate region. But we also see that this non-optimal L_r results in the perfect conditioning in the diffusion limit (as well as in the advection limit). This is because the optimal L_r is not unique in the diffusion limit: the eigenvalues become complex conjugate for small Re_π and $L_r > \frac{1}{2\pi}$ (which includes both choices above), and thus we have $|\lambda_1^M| = |\lambda_2^M|$.

The first-order advection–diffusion system has now completely defined in the differential level, *independently of discretization methods*.

2.3. $O(h)$ Time step

Simply because the first-order advection–diffusion system is hyperbolic, the time step for any explicit scheme is restricted based on the CFL condition:

$$\Delta t = \mathrm{CFL}\frac{h_{min}}{a + \nu/L_r}, \tag{2.32}$$

where CFL is the CFL number ($\leqslant 1$, typically), h_{min} is the minimum mesh size for a given mesh, and $a + \nu/L_r$ is the maximum wave speed of the first-order advection–diffusion system (2.2). This shows, since $L_r = O(1)$ as in (2.30), that the time step is proportional to the mesh size (not squared) for *all Reynolds numbers*. Note that this $O(h)$ time step is a direct consequence of solving the hyperbolic advection–diffusion system, not a property of a particular numerical scheme. Any explicit scheme,

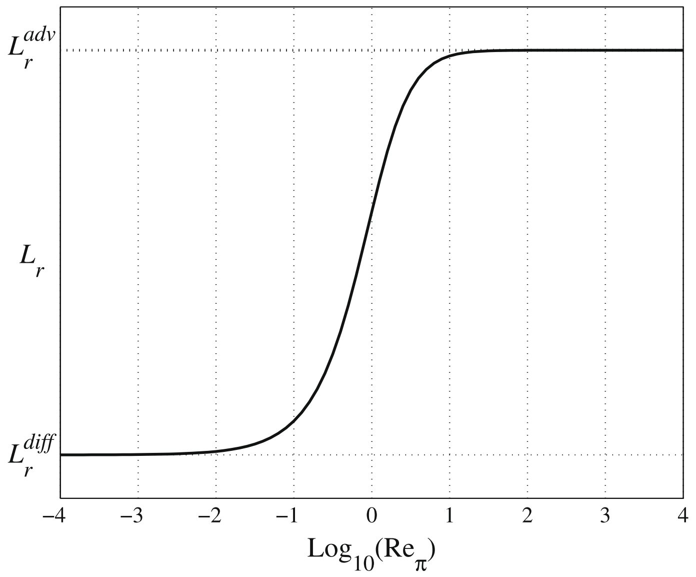


Fig. 2. Plot of the optimal L_r (2.30). $L_r^{adv} = \frac{1}{\pi}$ and $L_r^{diff} = \frac{1}{\sqrt{2}\pi}$ are the limiting values.

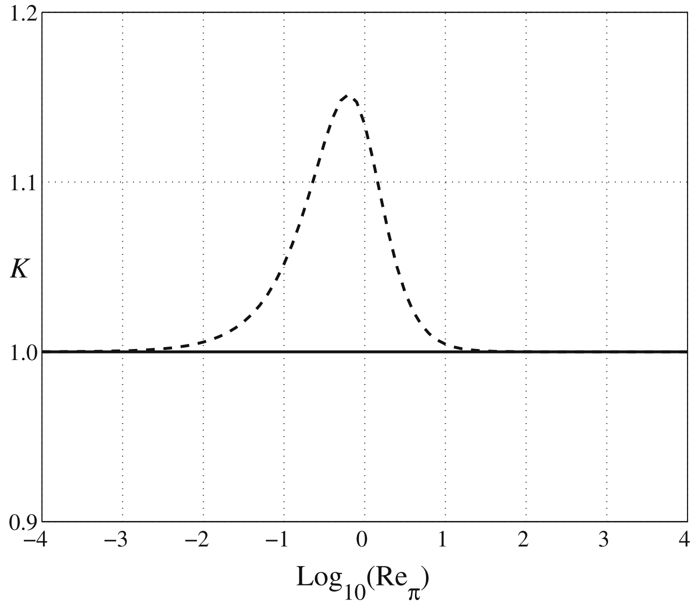


Fig. 3. The condition number, $K = |\lambda_1^M|/|\lambda_2^M|$, versus $log_{10}Re_\pi$: solid – optimal L_r (2.30), dashed – $L_r = \frac{1}{\pi}$.

e.g., finite-volume or finite-element schemes, developed for the hyperbolic advection–diffusion system will allow this remarkably large time step. Compare this with the well-known time step restriction for common scalar schemes (e.g., central or upwind schemes (2.62) or (2.66), augmented with the forward Euler time-stepping):

$$\Delta t = \text{CFL}\frac{h_{min}}{a + 2\nu/h_{min}}. \tag{2.33}$$

This is $O(h^2)$ in general, unless advection dominates everywhere in the domain (i.e., it depends on Reynolds numbers). This is a very severe restriction; it is one of the motivations for employing implicit schemes. For example, when $h = 10^{-7}$ which may be required to resolve a boundary layer, $O(h^2)$ time step gives a time step of $O(10^{-14})$, which is almost machine zero. $O(h)$ time step, on the other hand, gives a time step of $O(10^{-7})$, which is substantially larger.

To see an impact of the size of time steps on the number of iterations (total time steps) to reach a steady state, suppose that the steady state is reached at $t = t_f$ with n_f iterations:

$$t_f = n_f \Delta t, \tag{2.34}$$

and thus

$$n_f = \frac{t_f}{\Delta t}. \tag{2.35}$$

The time t_f may somewhat depend on the equations solved: the first-order system or the diffusion equation (the solution follows different transient physics). However, in each case, it is constant for a given problem and initial solution, and more importantly it is independent of the time step and the grid size. Hence, we write

$$n_f = O\left(\frac{1}{\Delta t}\right). \tag{2.36}$$

Now, since $h \propto 1/N$ where N is the number of unknowns, we obtain

$$n_f = \begin{cases} O(N) & \text{for } \Delta t = O(h), \\ O(N^2) & \text{for } \Delta t = O(h^2). \end{cases} \tag{2.37}$$

Therefore, the number of iterations is proportional to the number of unknowns (not squared) with $O(h)$ time step. This means that $O(h)$ time step gives $O(N)$ times faster convergence than $O(h^2)$ time step; the factor grows substantially with the problem size. This is a tremendous advantage of $O(h)$ time step over $O(h^2)$ time step. We point out that this type of convergence with $\Delta t = O(h)$ is not observed in general by stationary iterative methods (i.e., those which use only the solution or the residual at the previous iteration) for traditional diffusion schemes, such as the Jacobi, the Gauss–Seidel, or the successive over-relaxation (SOR). These well-known methods all correspond to $O(h^2)$ time-step schemes.

The above argument can be immediately translated into the CPU time. Each iteration requires $O(N)$ operations: residual computations and solution updates. Therefore, the CPU time required to reach the steady state, denoted by CPU, is estimated by

$$\mathrm{CPU} = n_f \times O(N) = \begin{cases} O(N^2) & \text{for } \Delta t = O(h), \\ O(N^3) & \text{for } \Delta t = O(h^2). \end{cases} \tag{2.38}$$

Hence, $O(h)$ time step gives $O(N)$ speed-up also in actual computing time. Note again that the speed-up factor is not a constant but grows with the problem size: the finer the grid, the faster the convergence in both the iteration number and the CPU time. We point out here that compared with traditional scalar schemes for the advection–diffusion equation, the first-order system approach requires system schemes which involve more operations per iteration. However, it does not affect the above estimates because it only introduces a constant factor. Even if an $O(h)$ time-step scheme requires 10 times more operations per degree of freedom, it will be $O(N/10)$ faster in one dimension than traditional schemes, which can be quite substantial for large scale problems, $N \gg 10$. In effect, the extra cost of carrying gradient variables and their equations in the first-order system approach is overwhelmed by the speed-up factor for large N.

Note that $O(h)$ time step extends straightforwardly to two and three dimensions simply because the first-order advection–diffusion system is hyperbolic for all dimensions. Noting that $h \propto 1/N^{\frac{1}{2}}$ in two dimensions and $h \propto 1/N^{\frac{1}{3}}$ in three dimensions, we obtain similar results as summarized in Table 1. We see from Table 1 that, as we would naturally expect, $O(h)$ time step gives $O(1/h)$ times faster convergence over $O(h^2)$ time step in both the iteration number and the CPU time in all dimensions: $N, \; N^{1/2}, \; N^{1/3}$ times faster in one, two, and three dimensions, respectively.

It is possible to translate the above estimates further into relations between the solution error and the CPU time. Assume that a scheme is p-th order accurate ($p \geqslant 1$), so that the solution error, $\mathcal{E}$, measured in some norm of interest is given by

$$\mathcal{E} = O(h^p). \tag{2.39}$$

In one dimension, we have $O(h^p) = O(N^{-p})$, and therefore it follows from (2.38) that

$$\mathcal{E} = O(N^{-p}) = \begin{cases} O(\mathrm{CPU}^{-p/2}) & \text{for } \Delta t = O(h), \\ O(\mathrm{CPU}^{-p/3}) & \text{for } \Delta t = O(h^2). \end{cases} \tag{2.40}$$

Note that the exponent to CPU is smaller (i.e., larger in magnitude) for $O(h)$ time-step. It means that schemes with $O(h)$ time-step give a smaller solution error for a fixed CPU time. We can also express (2.40) conversely as

$$\mathrm{CPU} = \begin{cases} O(\mathcal{E}^{-2/p}) & \text{for } \Delta t = O(h), \\ O(\mathcal{E}^{-3/p}) & \text{for } \Delta t = O(h^2), \end{cases} \tag{2.41}$$

meaning that it takes less CPU time for $O(h)$ time-step schemes to produce a solution at a specified error level. Including results for two and three dimensions, these estimates are summarized in Table 2.

If implicit time-stepping schemes are employed to drive the solution to the steady state, the CFL number can be infinitely large in principle for both $O(h)$ and $O(h^2)$ time steps. The steady solution is then obtained by solving a linear system arising from the linearization of the residual, i.e., inversion of a global $N \times N$ Jacobian matrix. The size of the Jacobian matrix is larger for $O(h)$ time-step schemes based on the first-order advection–diffusion system by the number of the gradient variables (1 in 1D, 2 in 2D, 3 in 3D). However, the benefit of $O(h)$ time-step comes in the condition number of the Jacobian matrix: $O(N)$ for $O(h)$ time-step schemes against $O(N^2)$ for $O(h^2)$ time-step schemes; iterative methods will converge much faster for $O(h)$ time-step schemes. Another benefit is expected in the construction of the Jacobian matrix. Especially for unstructured grids, the exact linearization may not be practical because of an extended stencil required for discretizing the second derivative of diffusion. Often, an approximate Jacobian matrix is employed instead, and consequently the CFL number cannot be infinite although could still be larger than 1. On the other hand, there are no second derivatives in the first-order advection–diffusion system. Hence, it can be discretized within a compact stencil (as shown in this paper), and the construction of the exact Jacobian matrix can be made easier. It should be noted also that $O(h)$ time-step schemes can produce accurate solution gradients simultaneously. Detailed study on implicit schemes is a subject of future work.

Table 1
Complexity comparisons of $O(h)$ and $O(h^2)$ time steps. In all dimensions, $O(h)$ time step gives faster convergence in both the iteration number and the CPU time: $O(1/h)$ times faster. The factor grows with the problem size.

	One dimension		Two dimensions		Three dimensions	
Time step:	$O(h)$	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$	$O(h^2)$
n_f	$O(N)$	$O(N^2)$	$O\left(N^{1/2}\right)$	$O(N)$	$O\left(N^{1/3}\right)$	$O\left(N^{2/3}\right)$
CPU	$O(N^2)$	$O(N^3)$	$O\left(N^{3/2}\right)$	$O(N^2)$	$O\left(N^{4/3}\right)$	$O\left(N^{5/3}\right)$

Table 2
Relations between CPU time and solution error.

	One dimension		Two dimensions		Three dimensions	
Time step:	$O(h)$	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$	$O(h^2)$
CPU	$O(\mathcal{E}^{-2/p})$	$O(\mathcal{E}^{-3/p})$	$O(\mathcal{E}^{-3/p})$	$O(\mathcal{E}^{-4/p})$	$O(\mathcal{E}^{-4/p})$	$O(\mathcal{E}^{-5/p})$
$\mathcal{E}$	$O(\text{CPU}^{-p/2})$	$O(\text{CPU}^{-p/3})$	$O(\text{CPU}^{-p/3})$	$O(\text{CPU}^{-p/4})$	$O(\text{CPU}^{-p/4})$	$O(\text{CPU}^{-p/5})$

2.4. Discretization

To discretize the first-order advection–diffusion system, we employ the residual-distribution method: nodal solutions and cell-residuals. The method is known to achieve a design accuracy in the steady state with a compact stencil (involving the neighbor nodes only) on irregular meshes for equations with source terms [1,16,17]. These properties are particularly attractive for our purpose because we are interested only in the steady state; the first-order advection–diffusion system has a source term; non-uniform meshes are required to efficiently resolve boundary layers. It is, of course, possible to employ other methods such as finite-volume or finite-element methods. Many properties of the scheme we are going to develop here are direct consequences of solving the first-order advection–diffusion system, and therefore, can be shared by other methods. In fact, the scheme we construct here can be viewed, as will be shown later, as a finite-volume scheme. In any case, it will be a compact *three-point* difference scheme, involving only the nearest neighbors at every data point.

We begin by generating a set of nodes, $\{J\}$, with coordinates, x_j, distributed arbitrarily over the domain of interest. With the solution stored at each node, $(u_j, p_j),\ j \in \{J\}$, and two boundary conditions given for u, the task is to compute the steady state solution: $\{u_j\}$ at the interior nodes and $\{p_j\}$ at all nodes. Note, as in the pure diffusion case [1] that the number of unknowns will be exactly equal to the number of cell-residuals: all the cell-residuals can be driven to zero exactly in the steady state, thus implying the existence of a unique solution set. We remark that this is not true for scalar residual-distribution schemes (or any cell-vertex type schemes) that directly solve (2.1), resulting in a discrete problem overdetermined by one extra cell-residual.

To begin residual-distribution, we first define the cell-residual, $\boldsymbol{\Phi}^C$, as an integral value of the spatial part of the system over the cell, $C = [x_j, x_{j+1}]$,

$$\begin{aligned}\boldsymbol{\Phi}^C = \begin{bmatrix}\Phi_1^C\\ \Phi_2^C\end{bmatrix} &= \int_{x_j}^{x_{j+1}} (-\mathbf{A}\mathbf{U}_x + \mathbf{Q})dx = -\mathbf{A}\Delta\mathbf{U}^C + \overline{\mathbf{Q}}^C h^C = -\mathbf{A}(\mathbf{U}_{j+1} - \mathbf{U}_j) + \left(w_j^C \mathbf{Q}_j + w_{j+1}^C \mathbf{Q}_{j+1}\right)h^C\\ &= \begin{bmatrix} -a(u_{j+1} - u_j) + \nu(p_{j+1} - p_j) \\ (u_{j+1} - u_j)/T_r - (w_{j+1}^C p_{j+1} + w_j^C p_j)h^C/T_r \end{bmatrix},\end{aligned} \tag{2.42}$$

where $h^C = x_{j+1} - x_j$, and $\left(w_j^C, w_{j+1}^C\right)$ is a set of quadrature weights that satisfy, within the cell,

$$w_j^C + w_{j+1}^C = 1. \tag{2.43}$$

The choice of the weights is left open at this point; it will be discussed in the next subsection. Second, we distribute the cell-residual to the nodes, j and $j+1$, by using the upwind distribution matrix (see [1]):

$$\mathcal{B}_j^C = \frac{1}{2}\mathbf{R}\begin{bmatrix}\left(1 - \frac{\lambda_1}{|\lambda_1|}\right) & 0 \\ 0 & \left(1 - \frac{\lambda_2}{|\lambda_2|}\right)\end{bmatrix}\mathbf{R}^{-1}, \tag{2.44}$$

$$\mathcal{B}_{j+1}^C = \frac{1}{2}\mathbf{R}\begin{bmatrix}\left(1 + \frac{\lambda_1}{|\lambda_1|}\right) & 0 \\ 0 & \left(1 + \frac{\lambda_2}{|\lambda_2|}\right)\end{bmatrix}\mathbf{R}^{-1}. \tag{2.45}$$

Each matrix projects the residual onto characteristic subspaces, and distributes the projected residuals to the left or the right according to the sign of the characteristic speed. Under the assumption that $a > 0$, it immediately follows from (2.9) that $\lambda_1 < 0$ and $\lambda_2 > 0$. Then, the distribution matrices can be simplified to

$$\mathcal{B}_j^C = \frac{1}{Re_{L_r} + 2}\begin{bmatrix} 1 & L_r \\ \frac{Re_{L_r} + 1}{L_r} & 1 + Re_{L_r}\end{bmatrix}, \tag{2.46}$$

$$\mathcal{B}_{j+1}^C = \frac{1}{Re_{L_r} + 2}\begin{bmatrix} 1 + Re_{L_r} & -L_r \\ -\frac{Re_{L_r} + 1}{L_r} & 1\end{bmatrix}, \tag{2.47}$$

After completing the distribution step within all elements, we arrive at the following semi-discrete equation,

$$\tilde{h}_j \frac{d\mathbf{U}_j}{dt} = \left[\mathcal{B}_j^L \boldsymbol{\Phi}^L + \mathcal{B}_j^R \boldsymbol{\Phi}^R\right], \tag{2.48}$$

where L and R denote the cells on the left and right of the node j, respectively, and $\tilde{h}_j$ is the measure of the dual control volume around the node j defined by

$$\tilde{h}_j = \frac{h^L + h^R}{2} \tag{2.49}$$

(see Fig. 4). Finally, we integrate the semi-discrete equation in time towards the steady state. In this study, we employ the forward Euler time-stepping:

$$\tilde{h}_j \frac{\mathbf{U}_j^{n+1} - \mathbf{U}_j^n}{\Delta t} = \left[\mathcal{B}_j^L \boldsymbol{\Phi}^L + \mathcal{B}_j^R \boldsymbol{\Phi}^R\right], \tag{2.50}$$

where the right hand side is evaluated at the time level n. The time step is restricted by the CFL condition (2.32); it is $O(h)$ for all Reynolds numbers. This is a fully-discrete *explicit* upwind residual-distribution scheme for the first-order advection–diffusion system. Note that this scheme is a three-point compact difference scheme because it involves only the two neighbor cells (hence two neighbor nodes).

It is instructive to expand (2.50) and look at the limiting behaviors of each component:

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{\tilde{h}_j}\left[\frac{(Re_{L_r}+1)\Phi_1^L + \Phi_1^R}{Re_{L_r}+2} + \frac{L_r}{Re_{L_r}+2}\left(\Phi_2^R - \Phi_2^L\right)\right], \tag{2.51}$$

$$p_j^{n+1} = p_j^n + \frac{\Delta t}{\tilde{h}_j}\left[\frac{\Phi_2^L + (Re_{L_r}+1)\Phi_2^R}{Re_{L_r}+2} + \frac{Re_{L_r}+1}{L_r(Re_{L_r}+2)}\left(\Phi_1^R - \Phi_1^L\right)\right]. \tag{2.52}$$

In the advection limit, $Re_{L_r} \to \infty$ ($\nu \to 0$), these reduce to

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{\tilde{h}_j}\Phi_1^L = u_j^n - \frac{a\Delta t}{\tilde{h}_j}\left(u_j^n - u_{j-1}^n\right), \tag{2.53}$$

$$p_j^{n+1} = p_j^n + \frac{\Delta t}{\tilde{h}_j}\left[\Phi_2^R + \frac{1}{L_r}\left(\Phi_1^R - \Phi_1^L\right)\right] = p_j^n + \frac{a\Delta t}{\tilde{h}_j L_r}\left[\left(u_j^n - u_{j-1}^n\right) - \left(w_j^R p_j^n + w_{j+1}^R p_{j+1}^n\right)h^R\right]. \tag{2.54}$$

Naturally, we have a scalar upwind scheme for u_j. For p_j, the scheme will be fully upwind by taking $w_j^R = 1$ and $w_{j+1}^R = 0$. We discuss these quadrature weights in the next subsection in relation to numerical oscillations. On the other hand, in the diffusion limit, $Re_{L_r} \to 0$ ($a \to 0$), we have

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{\tilde{h}_j}\left[\frac{1}{2}\left(\Phi_1^L + \Phi_1^R\right) + \frac{L_r}{2}\left(\Phi_2^R - \Phi_2^L\right)\right], \tag{2.55}$$

$$p_j^{n+1} = p_j^n + \frac{\Delta t}{\tilde{h}_j}\left[\frac{1}{2}\left(\Phi_2^L + \Phi_2^R\right) + \frac{1}{2L_r}\left(\Phi_1^R - \Phi_1^L\right)\right]. \tag{2.56}$$

These are central schemes (with appropriate dissipation terms which vanish in the steady state) suitable for diffusion problems. Note that these central schemes have emerged as a result of applying upwind schemes for the two wave-like components traveling in the opposite directions at the same speed [1]. It is remarkable that the scalar upwind scheme for advection and the central scheme for diffusion have been integrated automatically, simply by applying a single upwind scheme for the entire first-order advection–diffusion system. In particular, no considerations on any Reynolds number effects was necessary. The scheme adjusts itself to respond to the balance between advection and diffusion. Again, this is due to the unification of advection and diffusion in the differential level. Any schemes developed for the entire first-order advection–diffusion system will have a similar property.

To see how the upwind and central schemes are combined into one, note first that the distribution matrices (2.46) and (2.47) are the projection matrices, (2.16) and (2.17):

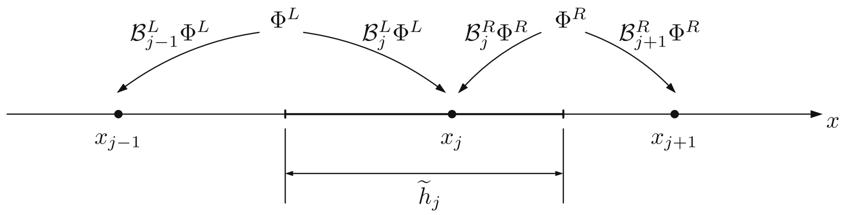


Fig. 4. Distribution of cell-residuals in one dimension.

$$\mathcal{B}_j^C = \mathbf{\Pi}_1, \quad \mathcal{B}_{j+1}^C = \mathbf{\Pi}_2, \tag{2.57}$$

and recall that each projection matrix can be written as a linear combination of its advection and diffusion limits, i.e., (2.19). Then, we can express the distribution matrices as

$$\mathcal{B}_j^C = \mathbf{\Pi}_1 = \frac{Re_{L_r}}{Re_{L_r}+2}\mathbf{\Pi}_1^A + \frac{2}{Re_{L_r}+2}\mathbf{\Pi}_1^D = \frac{Re_{L_r}}{Re_{L_r}+2}\begin{bmatrix} 0 & 0 \\ \frac{1}{L_r} & 1 \end{bmatrix} + \frac{2}{Re_{L_r}+2}\begin{bmatrix} \frac{1}{2} & \frac{L_r}{2} \\ \frac{1}{2L_r} & \frac{1}{2} \end{bmatrix}, \tag{2.58}$$

$$\mathcal{B}_{j+1}^C = \mathbf{\Pi}_2 = \frac{Re_{L_r}}{Re_{L_r}+2}\mathbf{\Pi}_2^A + \frac{2}{Re_{L_r}+2}\mathbf{\Pi}_2^D = \frac{Re_{L_r}}{Re_{L_r}+2}\begin{bmatrix} 1 & 0 \\ -\frac{1}{L_r} & 0 \end{bmatrix} + \frac{2}{Re_{L_r}+2}\begin{bmatrix} \frac{1}{2} & \frac{-L_r}{2} \\ -\frac{1}{2L_r} & \frac{1}{2} \end{bmatrix}. \tag{2.59}$$

This shows that each of the upwind distribution matrices, $\mathcal{B}_j^C$ and $\mathcal{B}_{j+1}^C$, can be thought of as a weighted average of an upwind distribution matrix for advection and another upwind distribution matrix for diffusion.

2.5. Source term discretization and cell Reynolds number

It is important to note that the upwind distribution does not guarantee monotone solutions in the steady state. To see this, suppose we employ the trapezoidal rule to discretize the source term in (2.42):

$$w_j^C = w_{j+1}^C = \frac{1}{2}, \tag{2.60}$$

which ensures second-order accuracy of the cell-residual, $\boldsymbol{\Phi}^C$. Also, recall that all cell-residuals vanish in the steady state, i.e.,

$$\boldsymbol{\Phi}^L = \boldsymbol{\Phi}^R = \mathbf{0}, \tag{2.61}$$

for the left and right cells of all $j \in \{J\}$. Then, we find from this pair of vanishing cell-residuals with $h = h^L = h^R$ that the steady state solution satisfies

$$a\frac{u_{j+1}-u_{j-1}}{2h} = \nu\frac{u_{j+1}-2u_j+u_{j-1}}{h^2}, \tag{2.62}$$

and the same for p_j. This can be written as

$$u_j = \frac{1}{2}\left(1-\frac{Re_h}{2}\right)u_{j+1} + \frac{1}{2}\left(1+\frac{Re_h}{2}\right)u_{j-1}, \tag{2.63}$$

where Re_h is the cell Reynolds number defined by

$$Re_h \equiv \frac{ah}{\nu}. \tag{2.64}$$

This is nothing but the classical central-difference approximation to the steady advection–diffusion equation; it is prone to spurious oscillations because the coefficient for u_{j+1} goes negative when $Re_h > 2$. Note that this is derived from the cell-residuals only, and has nothing to do with the distribution matrix. To avoid oscillations, therefore, we must modify the cell-residuals such that they correspond to an upwind discretization in the steady state. This is possible through the source term, and in fact, a one-sided evaluation of the source term,

$$w_j^C = 1, \quad w_{j+1}^C = 0, \tag{2.65}$$

leads to the classical upwind discretization of the steady equation:

$$a\frac{u_j-u_{j-1}}{h} = \nu\frac{u_{j+1}-2u_j+u_{j-1}}{h^2}, \tag{2.66}$$

i.e.,

$$u_j = \frac{Re_h+1}{Re_h+2}u_{j+1} + \frac{1}{Re_h+2}u_{j-1}, \tag{2.67}$$

and the same for p_j. Observe that all coefficients are now positive for all Re_h. The accuracy of the cell-residual, however, deteriorates to first-order for this choice.

To achieve second-order accuracy without oscillations, we must construct a grid such that $Re_h \leqslant 2$ is satisfied. This requires extremely fine grids for advection-dominated flows; it can be too restrictive. In regions where a solution is nearly uniform, the condition may be violated without introducing serious oscillations. In practice, therefore, it generally suffices to satisfy $Re_h \leqslant 2$ in high-gradient regions, e.g., boundary layers. We may also employ the above quadratures locally in an adaptive manner: one-sided for $Re_h > 2$; the trapezoidal rule for $Re_h \leqslant 2$, depending on the local mesh size. In order to suppress oscillations completely while keeping the second-order accuracy, we need to incorporate non-oscillatory schemes. This is a subject of future work.

2.6. Accuracy

Expand smooth functions u and p around a node j, and substitute them into the semi-discrete equation (2.48) to get

$$\frac{d\mathbf{U}_j}{dt} = \frac{1}{\tilde{h}_j}\left(\mathcal{B}_j^R h^R + \mathcal{B}_j^L h^L\right)(-\mathbf{A}\mathbf{U}_x + \mathbf{Q}) + \frac{1}{2\tilde{h}_j}\left\{\mathcal{B}_j^R (h^R)^2 - \mathcal{B}_j^L (h^L)^2\right\}(-\mathbf{A}\mathbf{U}_x + \mathbf{Q})_x$$
$$+ \frac{1}{\tilde{h}_j}\left[\mathcal{B}_j^R\left(w_j^R - \frac{1}{2}\right)(h^R)^2 - \mathcal{B}_j^L\left(w_j^L - \frac{1}{2}\right)(h^L)^2\right]\mathbf{Q}_x + O(h^2). \tag{2.68}$$

If the smooth functions are exact steady solutions of the first-order system, the time derivative as well as the spatial terms on the right vanish, and we are left with the terms on the second line, which is the local truncation error, $\mathcal{TE}$,

$$\mathcal{TE} = \frac{1}{\tilde{h}_j}\left[\mathcal{B}_j^R\left(w_j^R - \frac{1}{2}\right)(h^R)^2 - \mathcal{B}_j^L\left(w_j^L - \frac{1}{2}\right)(h^L)^2\right]\mathbf{Q}_x + O(h^2). \tag{2.69}$$

If the trapezoidal rule is used to discretize the source term on both cells, the leading term will vanish and the second-order accuracy is obtained in the steady state. But if the one-sided quadrature is used in either cell, it remains finite and the accuracy reduces to first-order. Alternatively, we may deduce the accuracy of the scheme from the residual property: second-order accurate with the trapezoidal rule because the residual (2.42) then vanishes for exact linear solutions; first-order accurate with the one-sided quadrature because the residual (2.42) vanishes only for exact constant solutions. This is a general result that is true for arbitrary grids.

2.7. Finite-volume form

It is well known that an upwind residual-distribution scheme is equivalent to a flux-difference splitting finite-volume scheme in one dimension (see [18] for example). Our scheme is not an exception. We now show that our scheme can be written as a finite-volume scheme. First, we express the products of the distribution matrix and $\mathbf{A}$ as

$$\mathcal{B}_j^C\mathbf{A} = \mathbf{\Pi}_1\mathbf{A} = \mathbf{\Pi}_1(\lambda_1\mathbf{\Pi}_1 + \lambda_2\mathbf{\Pi}_2) = \lambda_1\mathbf{\Pi}_1 = \frac{1}{2}(\mathbf{A} - |\mathbf{A}|), \tag{2.70}$$

$$\mathcal{B}_{j+1}^C\mathbf{A} = \mathbf{\Pi}_2\mathbf{A} = \mathbf{\Pi}_2(\lambda_1\mathbf{\Pi}_1 + \lambda_2\mathbf{\Pi}_2) = \lambda_2\mathbf{\Pi}_2 = \frac{1}{2}(\mathbf{A} + |\mathbf{A}|), \tag{2.71}$$

where

$$|\mathbf{A}| \equiv \mathbf{R}|\mathbf{\Lambda}|\mathbf{L} = |\lambda_1|\mathbf{\Pi}_1 + |\lambda_2|\mathbf{\Pi}_2. \tag{2.72}$$

Using these relations, we can expand and rewrite the semi-discrete scheme (2.48) as

$$\tilde{h}_j\frac{d\mathbf{U}_j}{dt} = \mathcal{B}_j^L\boldsymbol{\Phi}^L + \mathcal{B}_j^R\boldsymbol{\Phi}^R = -\frac{1}{2}(\mathbf{A} + |\mathbf{A}|)\Delta\mathbf{U}^L + \mathbf{\Pi}_2\overline{\mathbf{Q}}^L h^L - \frac{1}{2}(\mathbf{A} - |\mathbf{A}|)\Delta\mathbf{U}^R + \mathbf{\Pi}_1\overline{\mathbf{Q}}^R h^R$$
$$= -\frac{1}{2}(\mathbf{f}_{j+1} - \mathbf{f}_j - |\mathbf{A}|\Delta\mathbf{U}^R) + \mathbf{\Pi}_1\overline{\mathbf{Q}}^R h^R + \frac{1}{2}(-\mathbf{f}_j + \mathbf{f}_{j-1} - |\mathbf{A}|\Delta\mathbf{U}^L) + \mathbf{\Pi}_2\overline{\mathbf{Q}}^L h^L$$
$$= -\frac{1}{2}(\mathbf{f}_j + \mathbf{f}_{j+1} - |\mathbf{A}|\Delta\mathbf{U}^R) + \frac{1}{2}(\mathbf{f}_j + \mathbf{f}_{j-1} - |\mathbf{A}|\Delta\mathbf{U}^L) + \mathbf{\Pi}_2\overline{\mathbf{Q}}^L h^L + \mathbf{\Pi}_1\overline{\mathbf{Q}}^R h^R, \tag{2.73}$$

where

$$\mathbf{f}_j = \mathbf{A}\mathbf{U}_j = \begin{bmatrix} au_j - \nu p_j \\ -u_j/T_r \end{bmatrix}. \tag{2.74}$$

Therefore, our scheme is a finite-volume scheme:

$$\tilde{h}_j\frac{d\mathbf{U}_j}{dt} = -[\mathbf{F}_{j+1/2} - \mathbf{F}_{j-1/2}] + \tilde{\mathbf{Q}}_j, \tag{2.75}$$

where

$$\mathbf{F}_{j+1/2} = \frac{1}{2}(\mathbf{f}_j + \mathbf{f}_{j+1} - |\mathbf{A}|\Delta\mathbf{U}^R), \tag{2.76}$$

$$\mathbf{F}_{j-1/2} = \frac{1}{2}(\mathbf{f}_{j-1} + \mathbf{f}_j - |\mathbf{A}|\Delta\mathbf{U}^L), \tag{2.77}$$

$$\widetilde{\mathbf{Q}}_j = \mathbf{\Pi}_1\overline{\mathbf{Q}}^R h^R + \mathbf{\Pi}_2\overline{\mathbf{Q}}^L h^L. \tag{2.78}$$

This apparently first-order finite-volume scheme is second-order accurate in the steady state provided the trapezoidal rule is used within each cell for $\overline{\mathbf{Q}}^L$ and $\overline{\mathbf{Q}}^R$, as shown in Section 2.6. In effect, the particular source term discretization (2.78) makes it possible to achieve second-order accuracy in the steady state without reconstructing the solution. It is important to note, however, that the above argument is valid only for interior nodes, and appropriate boundary fluxes must be supplied on the boundary nodes if it is implemented as a finite-volume scheme. On the other hand, no special boundary treatment is necessary for the residual-distribution scheme because the boundary flux is already incorporated in the cell-residual over the cell adjacent to the boundary. We simply ignore in (2.50) the left cell-residual at the left boundary point and the right cell-residual at the right boundary point.

3. Two dimensions

We now consider two-dimensional problems. As we shall see, many remarkable properties of the one-dimensional numerical scheme will directly carry over to two dimensions, and again they directly come from solving the first-order advection–diffusion system, i.e., independently of discretization methods. Here, we construct a compact multidimensional upwind scheme by using the residual-distribution method on fully irregular triangular grids.

For structured grids, the one-dimensional scheme can be applied as a finite-difference scheme or a finite-volume scheme by decomposing the two-dimensional equation into dimension by dimension one-dimensional equations. This can be done in a straightforward manner (see [17,19,20] for example). We point out also that a finite-volume scheme can be developed in a similar manner for unstructured grids by applying a one-dimensional flux function normal to each cell face. Applications to other discretization methods will be undertaken in future, in relation to extensions to more complicated problems.

3.1. First-order advection–diffusion system

We consider the two-dimensional advection–diffusion problem,

$$u_t + au_x + bu_y = \nu(u_{xx} + u_{yy}) \quad \text{in } \Omega = (0,1) \times (0,1), \tag{3.1}$$

where u is given on the boundary, a and b are constants (not necessarily positive), and $\nu > 0$. To compute the steady state solution to this problem, we solve the following equivalent first-order system:

$$\begin{aligned} &u_t + au_x + bu_y = \nu(p_x + q_y), \\ &p_t = (u_x - p)/T_r, \\ &q_t = (u_y - q)/T_r, \end{aligned} \tag{3.2}$$

where p and q are the gradient variables which will be equivalent to the solution gradients, u_x and u_y, respectively, in the steady state. As in one dimension, this system is equivalent to the original advection–diffusion equation only in the steady state, and it is again hyperbolic. In the vector form, the system (3.2) is written as

$$\mathbf{U}_t + \mathbf{A}\mathbf{U}_x + \mathbf{B}\mathbf{U}_y = \mathbf{Q}, \tag{3.3}$$

where

$$\mathbf{U} = \begin{bmatrix} u \\ p \\ q \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} a & -\nu & 0 \\ -1/T_r & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} b & 0 & -\nu \\ 0 & 0 & 0 \\ -1/T_r & 0 & 0 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} 0 \\ -p/T_r \\ -q/T_r \end{bmatrix}. \tag{3.4}$$

Consider the Jacobian matrix $\mathbf{A}_n$ for an arbitrary vector $\mathbf{n} = (n_x, n_y)$,

$$\mathbf{A}_n = \mathbf{A}n_x + \mathbf{B}n_y = \begin{bmatrix} a_n & -\nu n_x & -\nu n_y \\ -n_x/T_r & 0 & 0 \\ -n_y/T_r & 0 & 0 \end{bmatrix}, \tag{3.5}$$

where a_n is the advection velocity projected onto $\mathbf{n}$:

$$a_n = an_x + bn_y. \tag{3.6}$$

It is easy to show that it has a set of real eigenvalues,

$$\lambda_1 = \frac{1}{2}\left[a_n - \sqrt{a_n^2 + \frac{4\nu}{T_r}}\right], \quad \lambda_2 = \frac{1}{2}\left[a_n + \sqrt{a_n^2 + \frac{4\nu}{T_r}}\right], \quad \lambda_3 = 0, \tag{3.7}$$

with linearly independent right-eigenvectors,

$$\mathbf{R}_n = \begin{bmatrix} -\lambda_1 T_r & -\lambda_2 T_r & 0 \\ n_x & n_x & -n_y \\ n_y & n_y & n_x \end{bmatrix}. \tag{3.8}$$

The system is therefore hyperbolic. The first two eigenvalues are essentially the same as the one-dimensional counterparts except that they are now based on the projected velocity, a_n. The third eigenvalue, λ_3, is associated with the inconsistency damping mode which damps out any inconsistency (nonzero $q_x - p_y$) that may be contained in an initial solution but must vanish at a steady state. See [1] for details.

On the boundary, since u is given, we can specify the gradient variables in the direction along the boundary: p at $y = 0$ or $y = 1$, and q at $x = 0$ or $x = 1$. Elsewhere, the gradient variables (the normal gradients) will be computed by a numerical scheme. For problems with the Neumann boundary condition, the gradient variables can be directly specified on the boundary. In effect, the Neumann condition turns into the Dirichlet condition in the first-order system approach.

To determine the time scale, T_r, we proceed as in one dimension: we set

$$T_r = \frac{L_r}{\lambda_2}, \tag{3.9}$$

and solve it for T_r to get

$$T_r = \frac{L_r}{|a_n| + \nu/L_r}, \tag{3.10}$$

where we have replaced a_n by $|a_n|$ to keep T_r positive. We then substitute this back into the eigenvalues and find

$$\lambda_1 = a_n^-\left(1 - \frac{1}{Re_{L_r}^-}\right), \quad \lambda_2 = a_n^+\left(1 + \frac{1}{Re_{L_r}^+}\right), \quad \lambda_3 = 0, \tag{3.11}$$

where

$$Re_{L_r}^- = \frac{a_n^- L_r}{\nu}, \quad Re_{L_r}^+ = \frac{a_n^+ L_r}{\nu}, \tag{3.12}$$

$$a_n^+ = \max(0, a_n), \quad a_n^- = \min(0, a_n). \tag{3.13}$$

The right-eigenvector matrix (3.8) can now be written as

$$\mathbf{R}_n = \begin{bmatrix} \dfrac{L_r}{Re_{L_r}^+ + 1} & \dfrac{L_r}{Re_{L_r}^- - 1} & 0 \\ n_x & n_x & -n_y \\ n_y & n_y & n_x \end{bmatrix}. \tag{3.14}$$

The corresponding left-eigenvector matrix is given by

$$\mathbf{L}_n = \mathbf{R}_n^{-1} = \frac{1}{|Re_{L_r}| + 2} \begin{bmatrix} \dfrac{|Re_{L_r}| + 1}{L_r} & (1 + Re_{L_r}^+)n_x & (1 + Re_{L_r}^+)n_y \\ -\dfrac{|Re_{L_r}| + 1}{L_r} & (1 - Re_{L_r}^-)n_x & (1 - Re_{L_r}^-)n_y \\ 0 & -n_y & n_x \end{bmatrix}, \tag{3.15}$$

where

$$|Re_{L_r}| = \frac{|a_n| L_r}{\nu}. \tag{3.16}$$

As in one dimension, the Jacobian matrix, $\mathbf{A}_n$, can be decomposed as follows:

$$\mathbf{A}_n = \lambda_1 \mathbf{\Pi}_{1,n} + \lambda_2 \mathbf{\Pi}_{2,n}, \tag{3.17}$$

where the projection matrices, $\mathbf{\Pi}_{1,n}$ and $\mathbf{\Pi}_{2,n}$ are given by

$$\mathbf{\Pi}_{1,n} = \frac{1}{|Re_{L_r}| + 2} \begin{bmatrix} 1 - Re_{L_r}^- & L_r n_x & L_r n_y \\ \dfrac{|Re_{L_r}| + 1}{L_r} n_x & (1 + Re_{L_r}^+)n_x^2 & (1 + Re_{L_r}^+)n_x n_y \\ \dfrac{|Re_{L_r}| + 1}{L_r} n_y & (1 + Re_{L_r}^+)n_x n_y & (1 + Re_{L_r}^+)n_y^2 \end{bmatrix}, \tag{3.18}$$

$$\mathbf{\Pi}_{2,n} = \frac{1}{|Re_{L_r}| + 2} \begin{bmatrix} 1 + Re_{L_r}^+ & -L_r n_x & -L_r n_y \\ -\dfrac{|Re_{L_r}| + 1}{L_r} n_x & (1 - Re_{L_r}^-)n_x^2 & (1 - Re_{L_r}^-)n_x n_y \\ -\dfrac{|Re_{L_r}| + 1}{L_r} n_y & (1 - Re_{L_r}^-)n_x n_y & (1 - Re_{L_r}^-)n_y^2 \end{bmatrix}. \tag{3.19}$$

It can be easily verified that these projection matrices can be written as a linear combination of its advection and diffusion limits, exactly as in the one-dimensional case, in the form of (2.19).

3.2. Length scale L_r

To determine the length scale, L_r, we again proceed as in one dimension and attempt to optimize the condition number of the system. In so doing, since the third component, i.e., the inconsistency damping mode, has no wave-like character (the source term is essential to describe its behavior), we focus on the two wave-like components. Substitute the Fourier mode, of phase angle $\beta = (\beta_x, \beta_y)$ with $\beta_x,\ \beta_y \in [0, \pi]$:

$$\mathbf{U}^\beta = e^{i(\beta_x x/h + \beta_y y/h)}\mathbf{U}_0. \tag{3.20}$$

Inserting this into the advection–diffusion system (3.2), we get

$$\frac{d\mathbf{U}^\beta}{dt} = \mathbf{M}\mathbf{U}^\beta, \tag{3.21}$$

where

$$\mathbf{M} = \begin{bmatrix} i\dfrac{a\beta_x + b\beta_y}{h} & \nu\dfrac{i\beta_x}{h} & \nu\dfrac{i\beta_y}{h} \\ \dfrac{i\beta_x}{hT_r} & -\dfrac{1}{T_r} & 0 \\ \dfrac{i\beta_y}{hT_r} & 0 & -\dfrac{1}{T_r} \end{bmatrix}. \tag{3.22}$$

The eigenvalues of this matrix are given by

$$\lambda_{1,2}^M = -\frac{1}{2}\left[\left(\frac{1}{T_r} + \frac{i\beta a_\beta}{h}\right) \pm \sqrt{\left(\frac{1}{T_r} - \frac{i\beta a_\beta}{h}\right)^2 - \frac{4\nu\beta^2}{h^2 T_r}}\right], \quad \lambda_3^M = -1/T_r, \tag{3.23}$$

where

$$a_\beta = \frac{a\beta_x + b\beta_y}{\beta}, \tag{3.24}$$

$$\beta = \sqrt{\beta_x^2 + \beta_y^2}. \tag{3.25}$$

The first two eigenvalues, which we focus on, are essentially the same as the one-dimensional eigenvalues (2.24). Therefore, the analysis in Section 2.2 directly applies to these two eigenvalues, and the optimal formula (2.30) derived for the one-dimensional system can be considered as optimal also in two dimensions, with

$$Re_\pi = \frac{\sqrt{a^2 + b^2}(1/\pi)}{\nu}. \tag{3.26}$$

The two-dimensional advection–diffusion system (3.2) has now been completely defined. We are ready to discretize the system. As discussed in Section 2.3, we expect to have $O(h)$ time step for any discretization methods also in two dimensions.

3.3. Discretization on unstructured triangular grids

We consider discretizing the two-dimensional advection–diffusion system (3.3) on unstructured triangular grids. We begin by dividing the domain into a set of triangles $\{T\}$ and a set of nodes $\{J\}$, and store the solution at each node, $(u_j, p_j),\ j \in \{J\}$. Now, the task is to compute the steady state solution: $\{u_j\}$ at the interior nodes and $\{p_j, q_j\}$ at all nodes except for the boundary nodes on which they can be computed from u given on the boundary.

To discretize the first-order advection–diffusion system on the triangular grid, we employ the residual-distribution method. We first define the cell-residual over a cell T (see Fig. 5) as

$$\boldsymbol{\Phi}^T = \begin{bmatrix} \Phi_u^T \\ \Phi_p^T \\ \Phi_q^T \end{bmatrix} = \int_T (-\mathbf{A}\mathbf{U}_x - \mathbf{B}\mathbf{U}_y + \mathbf{Q})\,dxdy. \tag{3.27}$$

Assuming a piecewise-linear variation of $\mathbf{U}$ over the cell, we obtain

$$\boldsymbol{\Phi}^T = -\sum_{i=1}^{3}\mathbf{K}_i\mathbf{U}_i + \overline{\mathbf{Q}}_T S_T, \tag{3.28}$$

where

$$\mathbf{K}_i = \frac{1}{2}(\mathbf{A}, \mathbf{B}) \cdot \mathbf{n}_i, \quad \overline{\mathbf{Q}}_T = \frac{\mathbf{Q}_1 + \mathbf{Q}_2 + \mathbf{Q}_3}{3}, \tag{3.29}$$

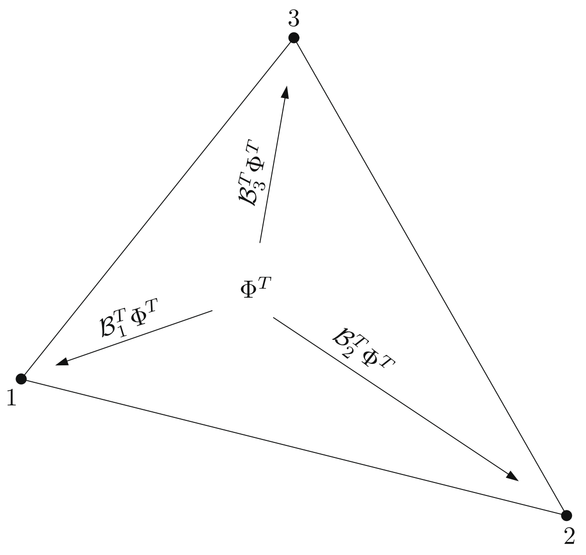


Fig. 5. Distribution of a cell-residual to the set of vertices $\{i_T\} = \{1, 2, 3\}$. Each contribution is determined by multiplying the cell-residual by the distribution matrix, $\mathcal{B}_i^T$, where $i \in \{i_T\}$.

$\mathbf{n}_i = (n_{i_x}, n_{i_y})$ is the scaled inward normal (see Fig. 6), and S_T is the cell area. The source term has been discretized to ensure the exactness for linear functions, and the derivatives, $\mathbf{U}_x$ and $\mathbf{U}_y$, are evaluated by the Green–Gauss integration over the cell which is also exact for linear functions. We remark here that in the definition of the cell-residual above, we set

$$T_r = \frac{L_r}{|\mathbf{a}| + \nu/L_r}, \tag{3.30}$$

where $|\mathbf{a}| = \sqrt{a^2 + b^2}$, so that T_r is constant within the cell. This is to ensure the residual property on the equations, $u_x - p$ and $u_y - q$; no updates will be sent to the nodal solutions if these equations are satisfied exactly over the cell in the integral sense. We now distribute the cell-residual to the nodes by a distribution matrix, $\mathcal{B}_i^T$:

$$\mathcal{B}_i^T \boldsymbol{\Phi}^T \tag{3.31}$$

(see Fig. 5) where the distribution matrix is required to satisfy

$$\sum_{i=1}^{3} \mathcal{B}_i^T = \mathbf{I}, \tag{3.32}$$

for conservation. In this work, we employ the matrix LDA scheme [21,22], which is an upwind scheme defined by

$$\mathcal{B}_i^T = \mathbf{K}_i^+ \left(\sum_{i=1}^{3} \mathbf{K}_i^+ \right)^{-1}, \tag{3.33}$$

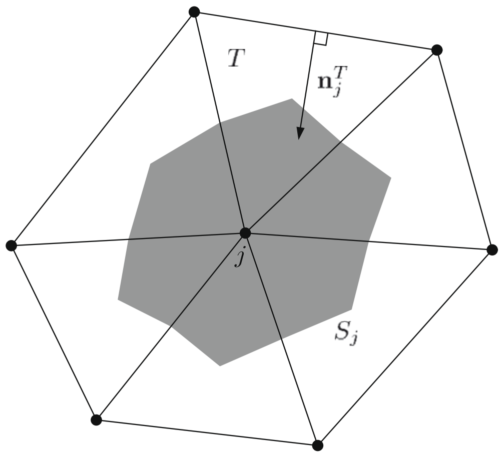


Fig. 6. Median dual cell around a node j over the set of surrounding triangles, $\{T_j\}$. S_j is the dual cell area. $\mathbf{n}_j^T$ is the scaled inward normal (not drawn to scale) associated with a triangle $T \in \{T_j\}$.

where

$$\mathbf{K}_i^+ = \frac{1}{2}\mathbf{R}_{n_i}\mathbf{\Lambda}_{n_i}^+\mathbf{R}_{n_i}^{-1} = \frac{1}{2}\mathbf{R}_{n_i}\begin{bmatrix} 0 & 0 & 0 \\ 0 & a_{n_i}^+(1+1/Re_{L_r}^+)\mid \mathbf{n}_i \mid & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathbf{R}_{n_i}^{-1} = \frac{1}{2}a_{n_i}^+\left(1+\frac{1}{Re_{L_r}^+}\right)\mid \mathbf{n}_i \mid \mathbf{\Pi}_{2,n_i}, \tag{3.34}$$

and all quantities with the subscript, n_i, are evaluated by the unit vector $\mathbf{n}_i/|\mathbf{n}_i|$. Note that we do not use (3.30) in computing the distribution matrix to properly account for the characteristics of the hyperbolic system, and that $Re_{L_r}^+$ is not a cell-wise constant but depends on the scaled inward normal vector $\mathbf{n}_i$. After performing the distribution step all over the cells, we have the following semi-discrete equation at each node:

$$\frac{d\mathbf{U}_j}{dt} = \frac{1}{S_j}\sum_{T\in\{T_j\}}\mathcal{B}_j^T\mathbf{\Phi}^T, \tag{3.35}$$

where $\{T_j\}$ denotes a set of triangles that share the node j and S_j is the median dual cell area (see Fig. 6). This is then integrated in time by the forward Euler time-stepping to reach the steady state. The time step is defined by

$$\Delta t = \text{CFL}\frac{2S_j}{\sum_{T\in\{T_j\}}\max_{i\in\{i_T\}} a_{n_i}^+(1+1/Re_{L_r}^+)|\mathbf{n}_i|}, \tag{3.36}$$

where CFL $\leqslant 1$, which is $O(h)$ since $S_j = O(h^2)$ and $|\mathbf{n}_i| = O(h)$.

Accuracy of residual-distribution schemes is obtained in the steady state, and generally determined by the exactness of the cell-residuals: p-th order accurate if the cell-residual is exact for polynomials of degree $p-1$ (see [16,23]). For the LDA scheme above, the solution is expected to be second-order accurate since the cell-residuals are designed to vanish for linear exact solutions. However, the accuracy of the gradient variables is not generally second-order. As demonstrated for diffusion problems in previous studies, it is second-order for smooth grids [1] but first-order on irregular grids [24]. This is because the cell-residuals are not designed to be exact for linear gradients, i.e., quadratic solutions. Consider the cell-residuals for the equations for p and q:

$$\Phi_p^T = \frac{1}{T_r}\int_T (u_x - p)dxdy = \frac{1}{T_r}\left(\frac{1}{2}\sum_{i=1}^{3}u_i n_{i_x} - \bar{p}_T S_T\right), \tag{3.37}$$

$$\Phi_q^T = \frac{1}{T_r}\int_T (u_y - q)dxdy = \frac{1}{T_r}\left(\frac{1}{2}\sum_{i=1}^{3}u_i n_{i_y} - \bar{q}_T S_T\right). \tag{3.38}$$

The first term in each residual is the Green–Gauss evaluation of the solution derivative; this is exact only for linear solutions, i.e., constant gradients. Therefore, although the source term has been designed to be exact for linear gradients, the whole cell-residuals cannot be exact for linear gradients (quadratic solutions). Consequently, the scheme is expected to be only first-order accurate for p and q in general although it recovers second-order accuracy for smooth grids [1]. To achieve second-order accuracy for the gradient variables on arbitrary grids, we need to improve the accuracy of the Green–Gauss term such that it will be exact for quadratic solutions. This can be achieved by the high-order curvature correction approach [24–29]:

$$\Phi_p^T = \frac{1}{T_r}\left(\frac{1}{2}\sum_{i=1}^{3}(u_i+\delta_i)n_{i_x} - \bar{p}_T S_T\right), \tag{3.39}$$

$$\Phi_q^T = \frac{1}{T_r}\left(\frac{1}{2}\sum_{i=1}^{3}(u_i+\delta_i)n_{i_y} - \bar{q}_T S_T\right), \tag{3.40}$$

where δ_i is the high-order curvature correction term given by

$$\delta_i = -\frac{1}{6}(\Delta p_i \Delta x_i + \Delta q_i \Delta y_i), \tag{3.41}$$

where Δp_i denotes the difference of the nodal values of p taken counterclockwise along the edge opposite to the node i (e.g., $\Delta p_1 = p_2 - p_3$), and similarly for others. This corresponds to using, instead of the Green–Gauss integration, the Simpson's rule along each edge with midpoint values reconstructed by the Hermite interpolation (see [24]). These 'corrected' residuals are now exact for quadratic solution, u, and thus exact for linear gradients, p and q, ensuring the second-order accuracy of the gradient variables. Note that the correction term does not require any explicit gradient reconstruction (which was required in [24–29]) since we now carry the gradients as unknowns and they are directly available at nodes. The scheme thus remains compact; this is a great advantage of the first-order system approach.

We remark that it is possible to make the same high-order correction also to the advection term to devise a third-order scheme for u. However, we do not consider such a scheme here because it is special for scalar equations and does not extend in general to systems of equations. For general systems, p and q are diffusive fluxes, not necessarily the solution derivatives.

We also point out that we expect third-order accuracy in the solution, u, in the diffusion limit. This is because the cell-residual for the advection–diffusion equation will be dominated by the diffusion term which is exact for linear gradients, meaning exact for quadratic solution.

The source term quadrature in (3.29) has been chosen to ensure second-order accuracy, but it may produce oscillatory solutions for high-Reynolds-number cases. An upwind quadrature is required for monotonicity. One possible formula is the following:

$$\bar{p}_T = w_1^T p_1 + w_2^T p_2 + w_3^T p_3, \tag{3.42}$$

$$\bar{q}_T = w_1^T q_1 + w_2^T q_2 + w_3^T q_3, \tag{3.43}$$

where

$$w_i^T = \frac{k_i^-}{\sum_{m=1}^{3} k_m^-}, \quad i = 1, 2, 3, \tag{3.44}$$

$$k_i^- = \min(0, k_i), \quad k_i = \frac{1}{2}(a, b) \cdot \mathbf{n}_i, \tag{3.45}$$

and we set $b = 0$ for $\bar{p}_T$ while $a = 0$ for $\bar{q}_T$. It is easy to see that we have $k_i^- = 1$ if the node i is the only upwind node. If there are two upwind nodes, k_i^- gives a fraction of the triangle defined by the other two nodes and the intersection point of the line along (a, b) passing through the downwind node to the area of the triangle, T (see Fig. 7). In either case, the quadrature weights for the nodes in the downwind side will be effectively set to be zero. This gives monotone solutions, but the accuracy reduces to first-order. Again, to suppress oscillations completely while retaining second-order accuracy, we need to incorporate non-oscillatory schemes. This will be explored in future.

4. Results

4.1. One-dimensional problem

We consider the following problem:

$$u_t + a u_x = \nu u_{xx} + q(x) \quad \text{in } \Omega = (0, 1), \tag{4.1}$$

with $u(0) = 0$ and $u(1) = 1$, where

$$q(x) = \frac{\pi}{Re}[a \cos(\pi x) + \pi \nu \sin(\pi x)], \tag{4.2}$$

and $Re = a/\nu$. The source term has been introduced to make the steady state solution non-trivial in the diffusion limit. The exact steady state solution is given by

$$u_{exact}(x) = \frac{\exp(-Re) - \exp(xRe - Re)}{\exp(-Re) - 1} + \frac{1}{Re}\sin(\pi x). \tag{4.3}$$

This is a smooth sine curve in the diffusion limit, but develops a narrow boundary layer near $x = 1$ when advection dominates (see Fig. 8).

We compute the steady state solution to this problem, by solving the equivalent first-order system:

$$\begin{aligned} u_t + a u_x &= \nu p_x + q(x), \\ p_t &= (u_x - p)/T_r, \end{aligned} \tag{4.4}$$

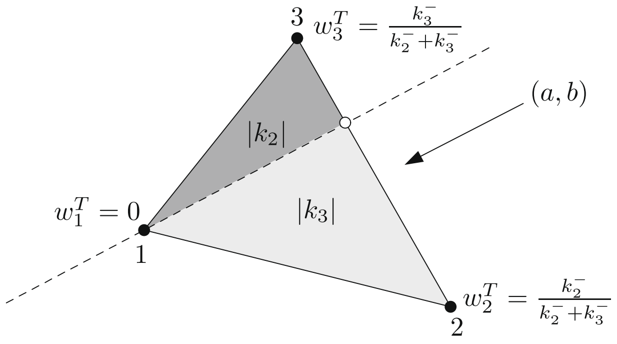

3
$w_3^T = \frac{k_3^-}{k_2^- + k_3^-}$
(a, b)
$|k_2|$
$|k_3|$
$w_1^T = 0$
1
$w_2^T = \frac{k_2^-}{k_2^- + k_3^-}$
2

Fig. 7. Quadrature weights given by (3.44) in the case of two upwind nodes. In actual implementation, we set $b = 0$ for the integration of p while $a = 0$ for the integration of q.

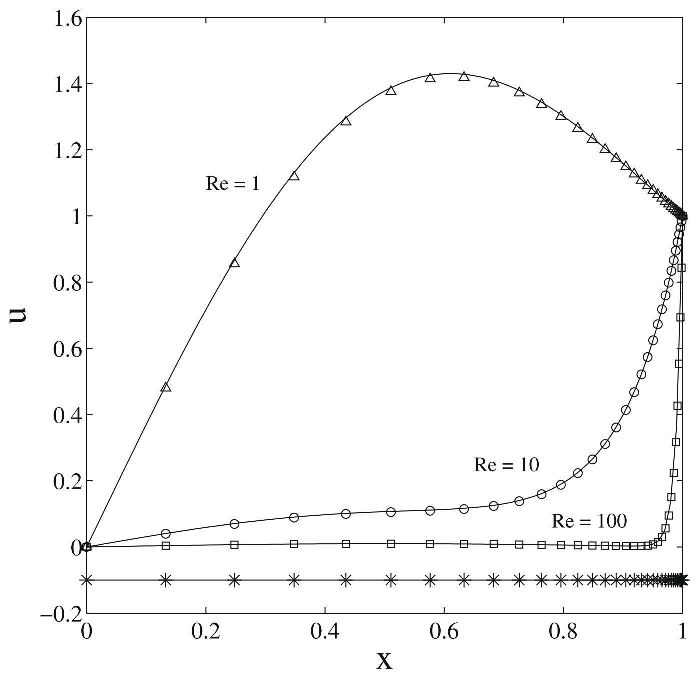


Fig. 8. Exact solutions (solid curves) and numerical solutions obtained by our advection–diffusion scheme (symbols: triangles, circles, squares) for $Re = 1, 10, 10^2$. A computational grid (33 nodes) used in the numerical tests is shown by stars in the bottom.

with the upwind advection–diffusion system scheme developed in Section 2.4. For the length scale, L_r, we use the optimal length scale in (2.30) and also $L_r = 1/\pi$ for comparison. The source term in the first equation does not affect the monotonicity of the steady solution (unlike the one in the second equation as discussed in Section 2.5), and therefore it is evaluated by the trapezoidal rule over each cell to ensure the second-order accuracy and added to the cell-residual. We start from the initial solution, $(u, p) = (x^2, 2x)$, and integrate in time until convergence by the forward Euler method. The method is taken to be converged when the nodal residuals are reduced five orders of magnitude in the L_1 norm. The time step is taken as global with CFL = 0.99. We conducted numerical experiments with non-uniform grids with the number of nodes, $N = 33$, 65, 129, 257. Each grid was generated from a uniform grid by the following mapping:

$$x_i = \frac{1 - \exp(-\alpha\xi_i)}{1 - \exp(-\alpha)}, \tag{4.5}$$

where $\xi_i = (i-1)/(N-1),\ i = 1, 2, 3, \ldots, N$, and $\alpha = 4.5$ for all grids (see Fig. 8 for an example). In this study, we set $a = 1$ and determine ν for a given Reynolds number. Results were obtained for a wide range of the Reynolds numbers: $Re = 10^k$, where $k = -3, -2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 3$.

Table 3 shows the iteration numbers obtained with the optimal length scale (2.30). Remarkably, the number of iterations to reduce the residuals by five orders of magnitude is nearly *independent of the Reynolds number*. This is considered due to $O(h)$ time step for all Reynolds numbers and also to the perfect preconditioning of the first-order system by the optimal L_r.

Table 3
The number of iterations for the upwind advection–diffusion scheme with the optimal L_r (2.30) in one dimension.

$\log_{10} Re$	Number of iterations			
	33 nodes	65 nodes	129 nodes	257 nodes
−3.0	2976	7368	14,685	29,170
−2.0	2979	7376	14,700	29,199
−1.5	2986	7393	14,735	29,270
−1.0	3010	7449	14,847	29,497
−0.5	3086	7629	15,218	30,244
0.0	3349	8186	16,491	32,869
0.5	3175	7926	17,277	38,081
1.0	3999	7735	15,428	35,747
1.5	3062	7180	15,389	32,277
2.0	3214	6458	13,962	29,518
3.0	3286	6877	14,355	29,893

Table 4
The number of iterations for the upwind advection–diffusion scheme with $L_r = 1/\pi$ in one dimension.

$\log_{10}Re$	Number of iterations			
	33 nodes	65 nodes	129 nodes	257 nodes
−3.0	3680	8825	17,697	41,274
−2.0	3688	8842	17,730	41,351
−1.5	3705	8882	17,810	41,536
−1.0	3761	9008	20,978	42,120
−0.5	3935	9401	21,836	43,947
0.0	4373	9000	21,356	43,014
0.5	3791	7713	18,945	38,766
1.0	4006	7574	16,570	36,420
1.5	3071	7197	15,422	32,344
2.0	3215	6460	13,967	29,528
3.0	3286	6877	14,355	29,893

Table 4 shows the results obtained for the non-optimal value, $L_r = 1/\pi$. Comparing with Table 3, we observe, as expected, that the number of iterations is generally larger, especially in the intermediate region where the first-order system is not perfectly conditioned, but it is nearly optimal in the advection limit. In the diffusion limit, the non-optimal length scale gives a perfect preconditioning of the system as shown in Fig. 3, but it gives a slightly slower convergence. This is because the perfect conditioning for a differential system does not necessarily imply the perfect conditioning of the numerical scheme. As shown in the previous study [1], the optimal length scale for our upwind scheme has a leading term, $0.5/\pi$, in the diffusion limit. The diffusion limit of the optimal formula (2.30) is $L_r = \frac{1}{\sqrt{2}\pi} \approx 0.707/\pi$; it is much closer to this leading term than the other one, $L_r = 1/\pi$, thus giving a faster convergence with the optimal formula (2.30). We also point out that the convergence in the diffusion limit can be much faster in practice because the grid stretching is not required and a uniform grid can be safely employed (a much larger minimum mesh spacing than stretched grids).

To demonstrate the impact of $O(h)$ time step on the number of iterations, we computed the same steady state solution by integrating the scalar advection–diffusion equation in time, again until the residual is reduced by five orders of magnitude, with a spatially second-order Galerkin scheme derived with a continuous piecewise-linear basis function over a non-uniform grid:

$$u_j^{n+1} = u_j^n + \frac{2\Delta t}{x_{j+1} - x_{j-1}}\left[-a\frac{u_{j+1}^n - u_{j-1}^n}{2} + \nu\left(\frac{u_{j+1}^n - u_j^n}{x_{j+1} - x_j} - \frac{u_j^n - u_{j-1}^n}{x_j - x_{j-1}}\right)\right]. \tag{4.6}$$

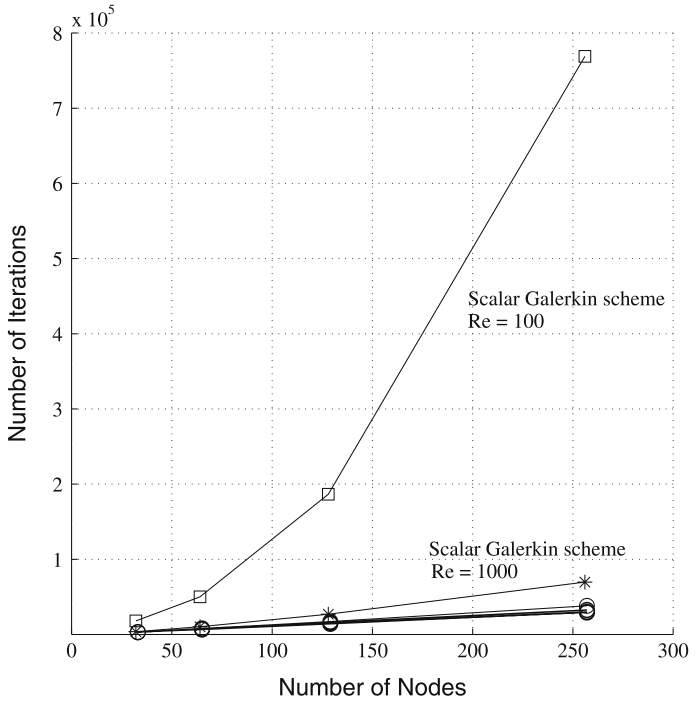


Fig. 9. The number of iterations to reduce the nodal residuals by five orders of magnitude for one-dimensional schemes. Circles: our upwind advection–diffusion system scheme for all values of Re in Table 3. Squares and stars: the scalar Galerkin scheme for $Re = 10^2$ and 10^3, respectively.

For this scalar scheme, the time step is restricted by (2.33) which is $O(h^2)$ in general. Fig. 9 shows the iteration number versus the mesh size, for our system scheme with the optimal L_r and for the scalar Galerkin scheme. It is clearly seen that the number of iterations for the scalar scheme increases quadratically with the mesh size even for a fairly advection-dominated case, $Re = 10^3$. On the other hand, for our system scheme, the number of iterations grows linearly with the mesh size for all Reynolds numbers. This is a natural consequence of solving the system that is *hyperbolic* for all Reynolds numbers.

Fig. 10(a) and 10(b) show L_∞ error convergence results: Fig. 10(a) for the main variable, u, Fig. 10(b) for the gradient variable, p. Note that the solution errors are independent of the choice of L_r; we obtained exactly the same results for both the optimal and non-optimal length scales because the discrete steady state solution is unique. Here, for a better visibility, we shifted the results with respect to those for $Re = 10^{-3}$ so that it can be read from the top to the bottom for increasing Reynolds numbers (only the errors for $Re = 10^{-3}$ can be correctly read off from the numbers indicated along the vertical axis). These figures show clearly that our system scheme is uniformly second-order accurate for *all Reynolds numbers and all variables, including the gradient variable on the boundary*. These results demonstrate that the scheme maintains its accuracy through the boundary.

Fig. 11 shows the CPU time versus the number of unknowns for the scalar Galerkin scheme and our system scheme. It clearly shows that our system scheme is orders of magnitude faster than the scalar Galerkin scheme for a comparable number of unknowns. These results also confirm the estimates in Table 1: CPU $= O(N^2)$ for $O(h)$ time step, and CPU $= O(N^3)$ for $O(h^2)$ time step. Fig. 12 shows the solution error versus CPU time. We observe that the error levels are comparable for each case, but the CPU time is orders of magnitude larger for the scalar Galerkin scheme. The rates of decrease in the solution error are in good agreement with those predicted in Table 2 for second-order accuracy $(p = 2) : \mathcal{E} = O(\mathrm{CPU}^{-1})$ for $O(h)$ time step, and $\mathcal{E} = O(\mathrm{CPU}^{-2/3})$ for $O(h^2)$ time step. These results demonstrate that the system scheme based on the first-order system be much more efficient than the scalar Galerkin scheme in spite of the additional cost of computing an extra variable. In fact, the system scheme was found to be about 3 times more expensive per iteration than the scalar Galerkin scheme. But the $O(N)$ speed-up in iterations did overwhelm this additional cost as predicted in Section 2.3.

In this numerical experiment, we employed the trapezoidal rule in the source term discretization for all cases. Although the condition, $Re_h \leqslant 2$, is violated in some region, since it is almost always satisfied near the narrow layer, we do not observe any serious oscillations (see Fig. 8).

4.2. Two-dimensional problems

We now consider the two-dimensional advection–diffusion problem:

$$u_t + au_x + bu_y = \nu(u_{xx} + u_{yy}) \quad \text{in } \Omega = (0,1) \times (0,1), \tag{4.7}$$

with the solution specified on the boundary by the following boundary-layer type exact steady solution [30],

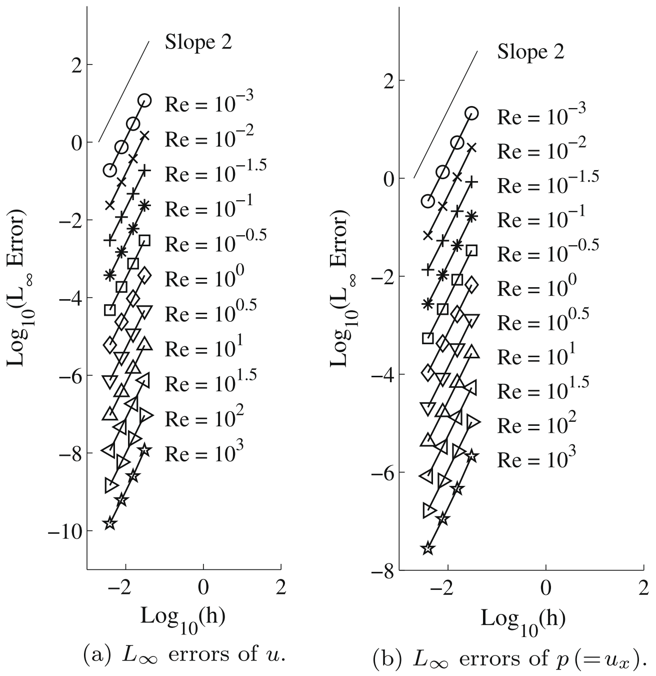


Fig. 10. L_∞ errors obtained by our upwind advection–diffusion system scheme for the one-dimensional problem. Second-order accuracy is confirmed for all Reynolds numbers.

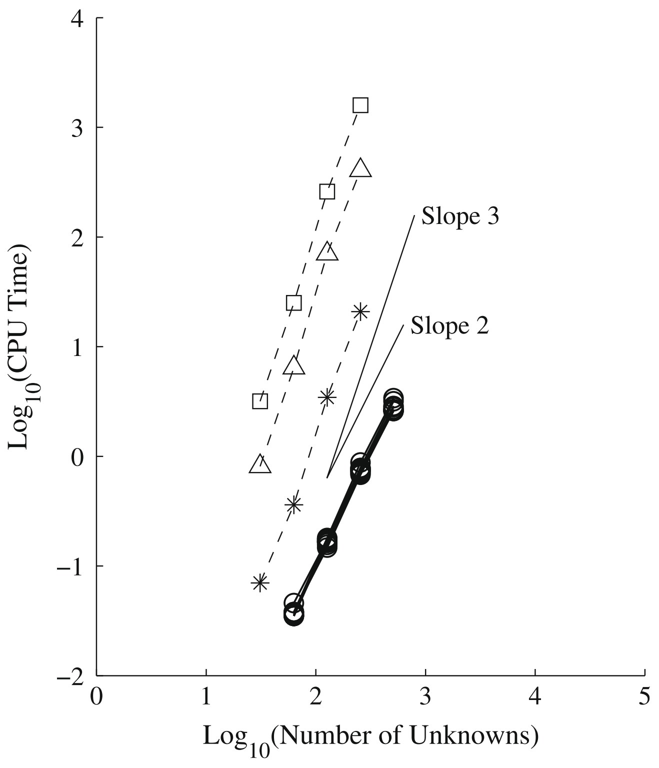


Fig. 11. CPU time versus the number of unknowns for one-dimensional schemes. CPU time is measured in seconds. Circles are used for our upwind advection–diffusion system scheme for all the Reynolds numbers. Symbols with dashed lines are used for the scalar Galerkin scheme: squares for $Re = 0$, triangles for $Re = 10$, and stars for $Re = 10^2$.

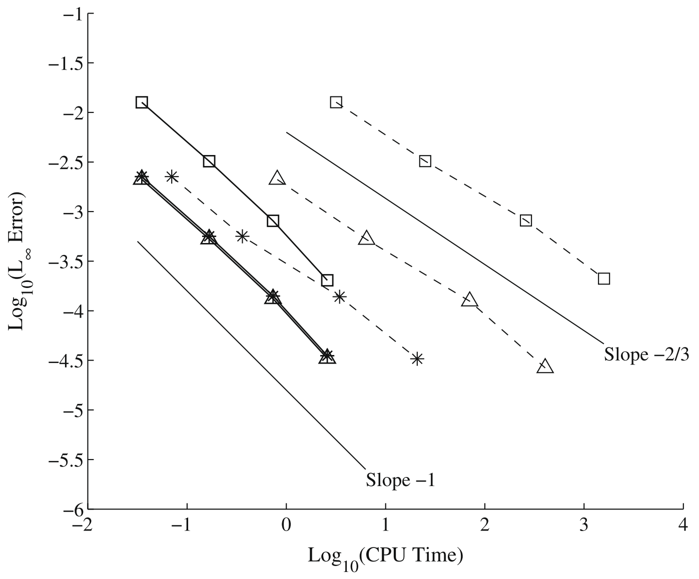


Fig. 12. L_∞ error of u versus CPU time for one-dimensional schemes. CPU time is measured in seconds. Solid lines are for our upwind advection–diffusion system scheme and dashed lines are for the scalar Galerkin scheme: squares for $Re = 0$, triangles for $Re = 10$, and stars for $Re = 10^2$.

$$u(x,y) = \frac{\left[1 - \exp\left((x-1)\frac{a}{\nu}\right)\right]\left[1 - \exp\left((y-1)\frac{b}{\nu}\right)\right]}{\left[1 - \exp\left(-\frac{a}{\nu}\right)\right]\left[1 - \exp\left(-\frac{b}{\nu}\right)\right]}. \tag{4.8}$$

To compute the steady state solution numerically, we integrate the first-order system in time:

$$\begin{aligned} &u_t + au_x + bu_y = \nu(p_x + q_y), \\ &p_t = (u_x - p)/T_r, \\ &q_t = (u_y - q)/T_r, \end{aligned} \tag{4.9}$$

until we reach the steady state. On the boundary, in addition to the solution value, we specify also the tangential gradients along on the boundary (simply because they can be computed from the solution on the boundary). The advection velocity is fixed as $(a,b) = (1.0, 0.8)$ for this study. The viscosity coefficient, ν, is determined for a given global Reynolds number, $Re = \sqrt{a^2 + b^2}/\nu$. For the length scale L_r we use the optimal formula (2.30) for all cases. The initial solution is set to be zero for all variables at all nodes except where the boundary condition is given. The time step is taken as global with CFL = 0.99 for all cases.

We present results for the multidimensional upwind scheme developed for triangular grids in Section 3.3 on a series of irregular triangular grids: each grid generated from a structured grid with random diagonal splittings and nodal perturbations (see Fig. 13). In the cell-residuals for the gradient variables, we employed the high-order curvature correction as described in Section 3.3 to ensure second-order accuracy for all variables. There are seven irregular grids, generated independently: 17×17, 24×24, 33×33, 41×41, 49×49, 57×57, and 66×66 grids. To reach the steady state, we march in time until the residual is reduced by 10 orders of magnitude, with the second-order accurate source term quadrature. Results were obtained for a set of Reynolds numbers: $Re = 10^k$, $k = -2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2$.

L_1 error convergence results are shown in Fig. 14 for both the solution and the gradient variables. As can be seen clearly, the scheme is second-order accurate for all variables and for all Reynolds numbers. Note that the error of u shows a third-order behavior in the diffusion dominated cases, thanks to the high-order curvature correction as discussed in Section 3.3. L_∞ errors (not shown) show some irregularity but generally go down at second-order accuracy. Fig. 15 shows the number of iterations to reach the steady state versus the square root of the number of nodes. Circles indicate the iteration numbers for our system scheme; squares indicate the iteration numbers for a scalar scheme with $O(h^2)$ time step restriction. The scalar scheme is constructed by adding the standard (continuous piecewise-linear) Galerkin discretization for diffusion to the scalar LDA scheme for advection, incorporating a blending function proposed in [12] to make the scheme uniformly accurate for all Reynolds numbers. The blending function is necessary because simply adding the scalar Galerkin scheme and the LDA scheme does not guarantee uniform accuracy as pointed out in [11]. It is observed in Fig. 15 that the number of iterations is proportional to square root of the grid size for our system scheme, whereas it increases quadratically for the scalar scheme. Also, observe that the number of iterations is almost independent of the Reynolds number for our system scheme.

Fig. 16 shows the CPU time versus the number of unknowns for our system scheme and the scalar LDA–Galerkin scheme. Here, the system scheme was found to be about 7 times more expensive per iteration than the scalar LDA–Galerkin scheme. However, despite the additional cost, again, the system scheme converged faster in the CPU time than the scalar LDA–Galerkin scheme in most cases as seen in Fig. 16. Again, the results are consistent with the estimates in Table 1: CPU = $O(N^{3/2})$ for $O(h)$ time step, and CPU = $O(N^2)$ for $O(h^2)$ time step. In a high-Reynolds number case ($Re = 10^2$), the scalar scheme actually

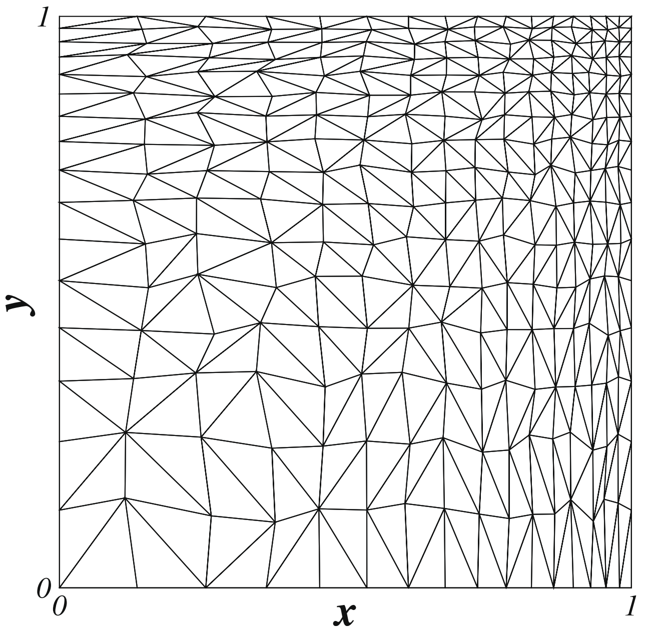
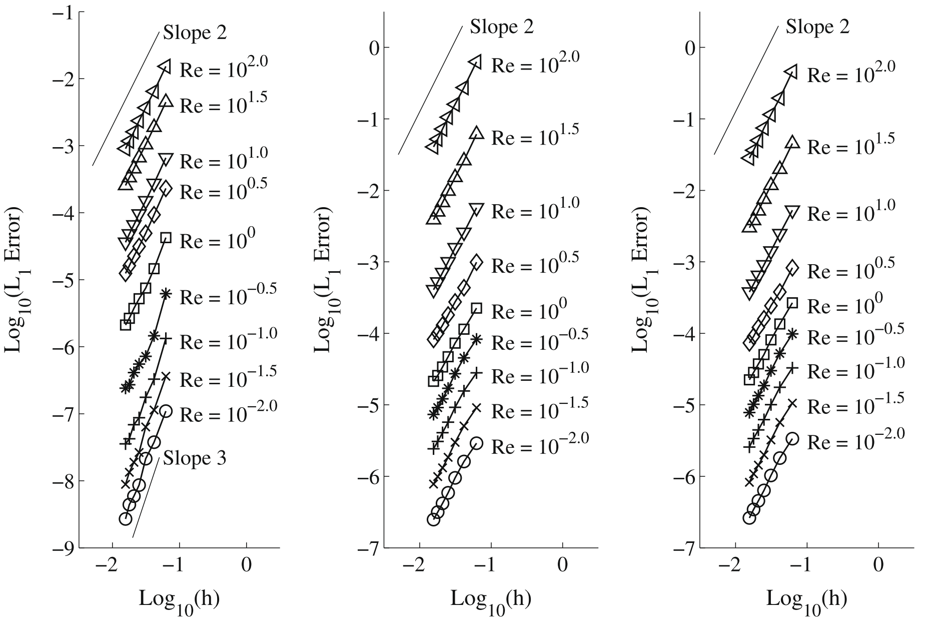


Fig. 13. Irregular triangular grid generated from a structured quadrilateral grid by random diagonal splittings and nodal perturbations.



(a) L_1 errors of u. (b) L_1 errors of $p\,(=u_x)$. (c) L_1 errors of $q\,(=u_y)$.

Fig. 14. L_1 errors for our LDA advection–diffusion system scheme on unstructured triangular grids.

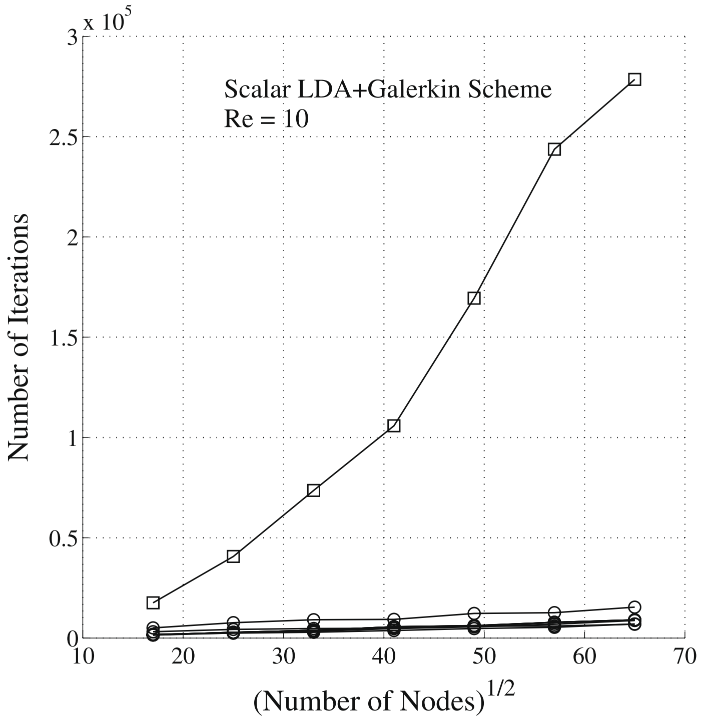


Fig. 15. The number of iterations to reduce the nodal residuals by 10 orders of magnitude for the unstructured grid schemes. Circles: our LDA advection–diffusion system scheme for $Re = 10^k$, $k = -2, -1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2$. Squares: the scalar LDA + Galerkin scheme for $Re = 10$.

converged faster than the system scheme, but it is found to be less accurate than the system scheme. Fig. 17(a) and 17(b) show the solution errors versus the CPU time. We observe in Fig. 17(a) that the scalar scheme exhibits first-order behavior in the first two grids for $Re = 10^2$. The slope of $-1/4$ corresponds to first-order accuracy ($p = 1$) as predicted in Table 2. Solution errors are consistently larger than those obtained by the system scheme (compare Fig. 17(a) and (b)). The case $Re = 10^2$ is actually a somewhat difficult case since coarse grids are not sufficiently resolving the boundary layer; the system scheme

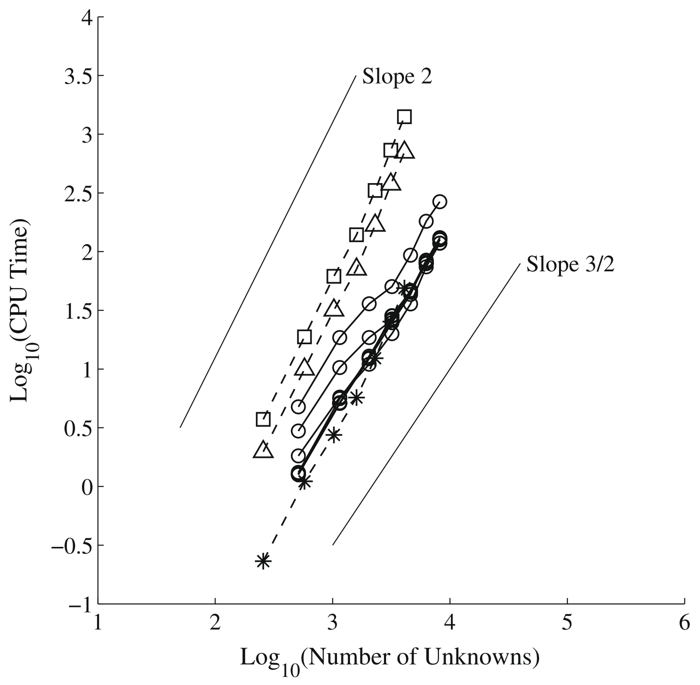
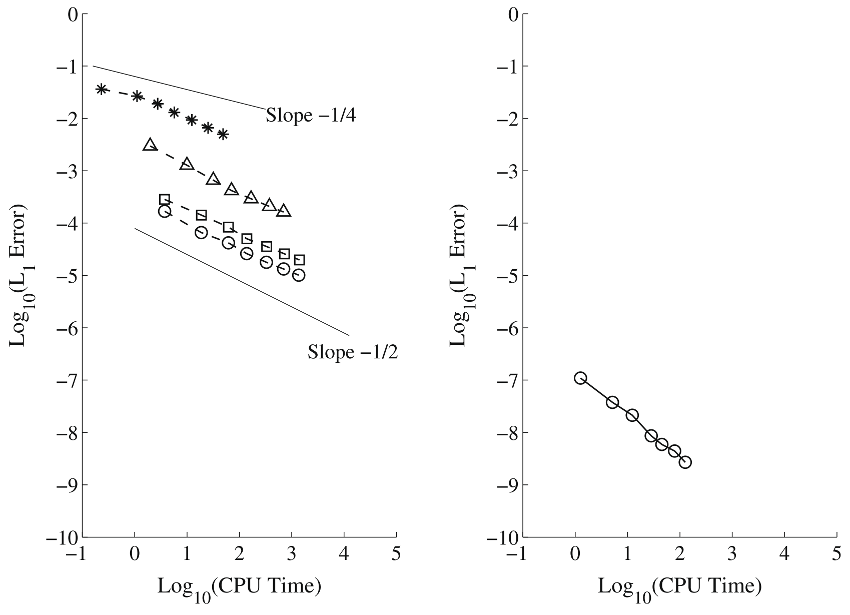
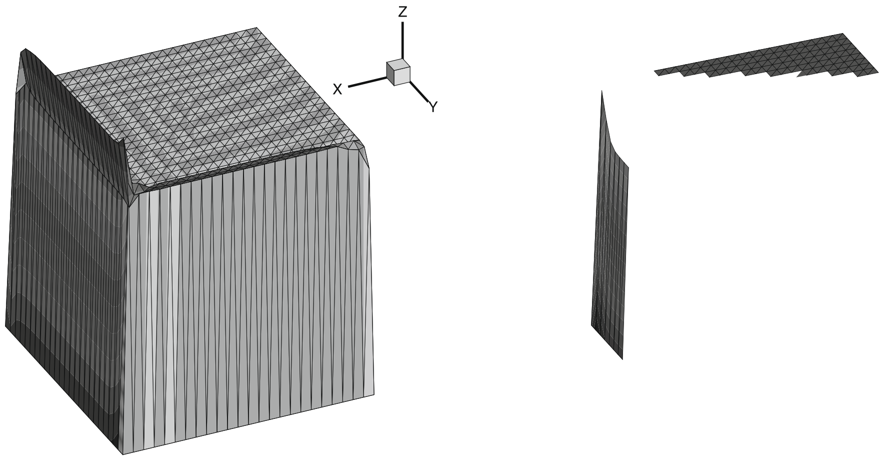


Fig. 16. CPU time versus the number of unknowns for two-dimensional unstructured grid schemes. CPU time is measured in seconds. Circles are used for our LDA advection–diffusion system scheme for all the Reynolds numbers. Symbols with dashed lines are used for the LDA–Galerkin scheme: squares for $Re = 0$, triangles for $Re = 10$, and stars for $Re = 10^2$.



(a) Scalar LDA-Galerkin scheme

slows down also although the design accuracy is achieved. In other cases, error levels are comparable, but the CPU time to reach a given error level is about an order of magnitude smaller for the system scheme. These results confirm, again, the estimates in Table 2 for second-order accuracy $(p = 2)$: $\mathcal{E} = O(\text{CPU}^{-2/3})$ for $O(h)$ time step, and $\mathcal{E} = O(\text{CPU}^{-1/2})$ for $O(h^2)$ time



(a) Oscillatory solution with the trapezoidal rule.

step. Note for the case $Re = 10^{-2}$ that the error level is orders of magnitude lower for the system scheme. This is because the system scheme becomes third-order accurate in the diffusion limit; the rate of decrease in the solution error approaches the third-order limit: $\mathcal{E} = O(\text{CPU}^{-1})$.

Finally, the upwind quadrature in Section 3.3 was tested for a case, $Re = 10^2$ on a regular uniform triangular grid with 17×17 nodes. As can be seen in Fig. 18, the scheme produced a monotone solution with the upwind quadrature.

5. Concluding remarks

In this paper, we have established the unification of advection and diffusion in the differential level into a single hyperbolic system, based on the first-order system approach proposed in the previous paper [1]. The proposed first-order advection–diffusion system was shown to be a *unified* representation of advection and diffusion; it is hyperbolic for all Reynolds numbers. The relaxation time, T_r, was determined in the differential level to enhance the convergence towards the steady state; an optimal formula for the associated length scale, L_r, was also derived. Simply by applying upwind schemes for the whole first-order system, we demonstrated that schemes constructed this way can be uniformly accurate and stably integrated in time by $O(h)$ time step with accurate solution gradients simultaneously computed *for all Reynolds numbers*. A remarkable implication is that we do not need to develop two different schemes (advection and diffusion schemes); we just need one scheme for solving the advection–diffusion equation. In one dimension, we developed an upwind residual-distribution scheme which can also be implemented as a finite-volume scheme. Monotonicity of the numerical solution was discovered to be closely related to the source term discretization: monotone by an upwind quadrature and oscillatory (for $Re_h > 2$) by the trapezoidal rule. In two dimensions, we developed a compact multidimensional upwind residual-distribution scheme for unstructured triangular grids. All schemes were shown to converge rapidly with $O(h)$ time step, giving second-order accurate solution and gradients through the boundary in the steady state for all Reynolds numbers. The property of $O(h)$ time step is particularly a decisive advantage of the first-order system approach. From comparisons with traditional scalar schemes, we demonstrated that schemes based on the first-order system give a tremendous speed-up in the CPU time for computing steady state solutions despite a higher cost per iteration associated with extra variables. We emphasize that the first-order system approach is not specific to a particular numerical method. It is a general approach applicable to various numerical methods: residual-distribution, finite-difference, finite-element, finite-volume, spectral-difference, *et cetera*.

The differential level unification of advection and diffusion would have a dramatic impact on future developments of numerical methods for advection–diffusion type problems. Most of all, we only need advection schemes for solving the advection–diffusion equation. To develop non-oscillatory schemes, we can directly apply techniques developed for advection schemes. To develop high-order schemes, again, we can directly employ high-order methods developed for advection schemes. The resulting scheme will be uniformly high-order accurate for all Reynolds numbers. For high-Reynolds-number applications, the minimum mesh size can be so small that explicit time-stepping schemes with $O(h)$ time step (although significantly larger than $O(h^2)$) may not be efficient enough for practical purposes. But then we can always employ implicit time-stepping as discussed in Section 2.3. Advantages still come in simpler Jacobian matrix construction and/or fast iterative linear solvers. For time-accurate computations, we may employ the dual-time-stepping technique [29,31–33], and the proposed approach gives a fast iterative solver for the inner iteration: integrate the following system,

$$
\begin{aligned}
u_\tau &= -a u_x + \nu p_x - u_t, \\
p_\tau &= (u_x - p)/T_r,
\end{aligned}
\tag{5.1}
$$

towards the steady state in the pseudo-time, τ, by the type of schemes presented in this paper. For turbulence model equations, which are typically in the form of the advection–diffusion equation with source terms, the schemes developed in this paper can be directly applied with an added advantage that source terms which depend on the solution gradient can be accurately evaluated by using the gradient variables that will be computed simultaneously. All the above-mentioned developments will be undertaken in future.

Yet, the proposed approach can also be used to derive time-accurate scalar advection–diffusion schemes by ignoring updates for the gradient variables and evaluating them instead by explicit gradient reconstruction (or by high-order moments if available, e.g., in the discontinuous Galerkin method). This idea has been presented in the previous paper [1] for the residual-distribution method for pure diffusion problems. But the idea is actually quite general; it is applicable to various discretization methods. Focusing on pure diffusion problems, we have already derived various diffusion schemes for node/cell-centered finite-volume, discontinuous Galerkin, and residual-distribution methods. Distinguished features of the diffusion schemes derived this way are twofold: *automatically* introduced 'dissipation' terms (similar to the so-called 'edge-term' in finite-volume schemes [34,35] or the 'penalty term' in the discontinuous Galerkin finite-element schemes [36,37]) that make the scheme *h*-elliptic [38] (i.e., damp spurious modes); the same implementation structure as advection schemes (e.g., interface flux evaluated by two discontinuous states in finite-volume schemes). The latter makes them extremely simple to integrate a diffusion scheme with an advection scheme. These diffusion schemes are, however, subject to the traditional $O(h^2)$ time-step restriction. Details on this derived approach will be reported elsewhere.

Among various future works, our immediate target are extensions to nonlinear systems, the Navier–Stokes equations in particular. To this end, however, the eigen-structure of the Navier–Stokes equations cast in the form of a first-order system may seem too difficult to analyze or too complicated to be practical. Successful applications to the Navier–Stokes equations may, therefore, come in some form of simplification or approximation. This will be the theme of the subsequent paper.

Acknowledgments

The author would like to thank Dr. Pieter Buning (NASA Langley) for an illuminating discussion, Dr. Yoshifumi Suzuki (Desktop Aeronautics) and the reviewer of the paper for their constructive comments. The quality of the paper was greatly improved by their contributions.

References

[1] H. Nishikawa, A first-order system approach for diffusion equation. I: Second-order residual-distribution schemes, Journal of Computational Physics 227 (2007) 315–352.
[2] P.L. Roe, M. Arora, Characteristic-based schemes for dispersive waves. I: The method of characteristics for smooth solutions, Numerical Methods for Partial Differential Equations 9 (1993) 459–505.
[3] C. Cattaneo, A form of heat-conduction equations which eliminates the paradox of instantaneous propagation, Comptes Rendus Académie Sciences 247 (1958) 431–433.
[4] G.B. Nagy, O.E. Ortiz, O.A. Reula, The behavior of hyperbolic heat equations' solutions near their parabolic limits, Journal of Mathematical Physics 35 (1994) 4334–4356.
[5] H. Gómez, I. Colominas, F. Navarrina, M. Casteleiro, A discontinuous Galerkin method for a hyperbolic model for convection–diffusion problems in CFD, International Journal for Numerical Methods in Engineering 71 (2007) 1342–1364.
[6] R.B. Lowrie, J.E. Morel, Methods for hyperbolic systems with stiff relaxation, International Journal for Numerical Methods in Fluids 40 (2002) 413–423.
[7] S.F. Liotta, V. Romano, G. Russo, Central schemes for balance laws of relaxation type, SIAM Journal on Numerical Analysis 38 (2000) 1337–1356.
[8] S. Jin, C.D. Levermore, Numerical schemes for hyperbolic conservation laws with stiff relaxation terms, Journal of Computational Physics 126 (1996) 449–467.
[9] M. Arora, Explicit characteristic-based high-resolution algorithms for hyperbolic conservation laws with stiff source terms, Ph.D. Thesis, University of Michigan, Ann Arbor, MI (1996).
[10] L.P. Franca, S.L. Frey, T.J.R. Hughes, Stabilized finite element methods: I. Application to the advective–diffusive model, Computer Methods in Applied Mechanics and Engineering 95 (1992) 253–276.
[11] H. Nishikawa, P.L. Roe, On high-order fluctuation-splitting schemes for Navier–Stokes equations, in: C. Groth, D.W. Zingg (Eds.), Computational Fluid Dynamics 2004, Springer-Verlag, 2004, pp. 799–804.
[12] M. Ricchiuto, N. Villedieu, R. Abgrall, H. Deconinck, On uniformly high-order accurate residual distribution schemes for advection–diffusion, Journal of Computational and Applied Mathematics 215 (2008) 547–556.
[13] C.-S. Chou, C.-W. Shu, High order residual distribution conservative finite difference WENO schemes for convection–diffusion steady state problems on non-smooth meshes, Journal of Computational Physics 224 (2007) 992–1020.
[14] C. Depcik, B. van Leer, In search of an optimal local Navier–Stokes preconditioner, in: Proceedings of the 16th AIAA Computational Fluid Dynamics Conference, AIAA Paper 2003-3703, 2003.
[15] S. Venkateswaran, C.L. Merkle, Analysis of time-derivative preconditioning for the Navier–Stokes equations, in: Proceedings of the Fifth International Symposium on Computational Fluid Dynamics, 1995.
[16] M. Ricchiuto, R. Abgrall, H. Deconinck, Application of conservative residual distribution schemes to the solution of the shallow water equations on unstructured meshes, Journal of Computational Physics 222 (2007) 287–331.
[17] C.-S. Chou, C.-W. Shu, High order residual distribution conservative finite difference WENO schemes for steady state problems on non-smooth meshes, Journal of Computational Physics 214 (2006) 698–724.
[18] H. Deconinck, P.L. Roe, R. Struijs, A multi-dimensional generalization of Roe's flux difference splitter for the Euler equations, Computers and Fluids 22 (1993) 215–222.
[19] H. Nishikawa, Towards future Navier–Stokes schemes: uniform accuracy, $O(h)$ time step, and accurate viscous/heat fluxes, in: Proceedings of the 19th AIAA Computational Fluid Dynamics Conference, AIAA Paper 2009-3648, San Antonio, 2009.
[20] R.-H. Ni, A multiple-grid scheme for solving the Euler equations, AIAA Journal 20 (11) (1981) 1565–1571.

[21] E. van der Weide, H. Deconinck, Positive matrix distribution schemes for hyperbolic systems, with application to the Euler equations, in: Computational Fluid Dynamics 1996, Wiley, New York, 1996, pp. 747–753.
[22] E. van der Weide, H. Deconinck, E. Issman, G. Degrez, A parallel, implicit, multi-dimensional upwind, residual distribution method for the Navier–Stokes equations on unstructured grids, Computational Mechanics 23 (1999) 199–208.
[23] R. Abgrall, Toward the ultimate conservative scheme: following the quest, Journal of Computational Physics 167 (2001) 277–315.
[24] H. Nishikawa, Higher-order discretization of diffusion terms in residual-distribution methods, in: Proceedings of the 34th VKI CFD Lecture Series Very-High Order Discretization Methods, VKI Lecture Series, 2005.
[25] D. Caraeni, L. Fuchs, Compact third-order multidimensional upwind scheme for Navier–Stokes simulations, Theoretical and Computational Fluid Dynamics 15 (2002) 373–401.
[26] H. Nishikawa, On grids and solutions from residual minimization, Ph.D. Thesis, University of Michigan, Ann Arbor, MI (August 2001).
[27] H. Nishikawa, P.L. Roe, High-order fluctuation-splitting schemes for advection–diffusion equations, in: H. Deconinck, E. Dick (Eds.), Computational Fluid Dynamics 2006, Springer-Verlag, 2006, pp. 77–82.
[28] H. Nishikawa, M. Rad, P. Roe, A third-order fluctuation-splitting scheme that preserves potential flow, in: Proceedings of the 15th AIAA Computational Fluid Dynamics Conference, AIAA Paper 01-2595, Anaheim, 2001.
[29] G. Rossiello, P. De Palma, G. Pascazio, M. Napolitano, Third-order-accurate fluctuation-splitting schemes for unsteady hyperbolic problems, Journal of Computational Physics 222 (2007) 332–352.
[30] J.C. Tannehill, D.A. Anderson, R.H. Pletcher, Computational Fluid Mechanics and Heat Transfer, second ed., Taylor & Francis, 1997.
[31] R. Payret, T. Taylor, Computational Methods for Fluid Flows, Springer, New York, 1983.
[32] A. Jameson, Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings, AIAA Paper 91-1596, 1991.
[33] D. Caraeni, L. Fuchs, Compact third-order multidimensional upwind discretization for steady and unsteady flow simulations, Computers and Fluids 34 (2005) 419–441.
[34] A. Haselbacher, J. Blazek, Accurate and efficient discretization of Navier–Stokes equations on mixed grids, AIAA Journal 38 (11) (2000) 2094–2102.
[35] B. Diskin, J.L. Thomas, E.J. Nielsen, H. Nishikawa, J.A. White, Comparison of node-centered and cell-centered unstructured finite-volume discretizations. Part I: Viscous fluxes, in: Proceedings of the 47th AIAA Aerospace Sciences Meeting, AIAA Paper 2009-597, 2009.
[36] D.N. Arnold, An interior penalty finite element method with discontinuous elements, SIAM Journal on Numerical Analysis 19 (4) (1982) 742–760.
[37] D.N. Arnold, F. Brezzi, B. Cockburn, L.D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM Journal on Numerical Analysis 39 (5) (2002) 1749–1779.
[38] U. Trottenberg, C.W. Oosterlee, A. Schüller, Multigrid, Academic Press, 2001.