On Hyperbolic DG Discretization

Hiroaki Nishikawa

January 14, 2016

1 Conventional DG Discretization for Hyperbolic Diffusion

Consider the hyperbolic diffusion system:

$$\frac{\partial \mathbf{u}}{\partial \tau} + \mathrm{div}\mathbf{F} = \mathbf{S}, \tag{1}$$

where

$$\mathbf{u} = \begin{bmatrix} u \\ p \\ q \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} -\nu p & -\nu q \\ -u/T_r & 0 \\ 0 & -u/T_r \end{bmatrix}, \quad \mathbf{s} = \begin{bmatrix} 0 \\ -p/T_r \\ -q/T_r \end{bmatrix}. \tag{2}$$

Numerical solution $\mathbf{u}_h$ is defined within an element T_j as a polynomial of degree p:

$$\mathbf{u}_h = \sum_k \mathbf{u}_j^{(k)} \phi_j^{(k)}(x, y), \tag{3}$$

where $\mathbf{u}_j^{(k)}$ is the k-th vector of unknown coefficients, and $\phi_j^{(k)}(x, y)$ is the k-th polynomial basis: e.g., in the P_1 case,

$$\mathbf{u}_j^{(0)} = \begin{bmatrix} \overline{u} \\ \overline{p} \\ \overline{q} \end{bmatrix}, \mathbf{u}_j^{(1)} = \begin{bmatrix} u_x \\ p_x \\ q_x \end{bmatrix}, \mathbf{u}_j^{(2)} = \begin{bmatrix} u_y \\ p_y \\ q_y \end{bmatrix}. \tag{4}$$

For our purpose, it is convenient to express the numerical solution in the form:

$$\mathbf{u}_h = \mathbf{B}_1 \mathbf{U}_1, \tag{5}$$

where, again, in the P_1 case,

$$\mathbf{B}_1 = \begin{bmatrix} 1 & 0 & 0 & \phi_j^{(1)} & 0 & 0 & \phi_j^{(2)} & 0 & 0 \\ 0 & 1 & 0 & 0 & \phi_j^{(1)} & 0 & 0 & \phi_j^{(2)} & 0 \\ 0 & 0 & 1 & 0 & 0 & \phi_j^{(1)} & 0 & 0 & \phi_j^{(2)} \end{bmatrix}, \quad \mathbf{U}_1 = [\overline{u}, \overline{p}, \overline{q}, u_x, p_x, q_x, u_y, p_y, q_y]^T. \tag{6}$$

In the general case of P_k approximation, we have

$$\mathbf{u}_h = \mathbf{B}_k \mathbf{U}_k, \tag{7}$$

where

$$\mathbf{B}_k = \left[\mathbf{I} \mid \phi_j^{(1)} \mathbf{I} \mid \phi_j^{(2)} \mathbf{I} \mid \cdots \mid \phi_j^{(n)} \mathbf{I} \right], \mathbf{U}_k = \left[(\mathbf{u}_j^{(0)})^T, (\mathbf{u}_j^{(1)})^T, (\mathbf{u}_j^{(2)})^T, \cdots, (\mathbf{u}_j^{(n)})^T \right], n = \frac{k(k+3)}{2}. \tag{8}$$

The DG discretization is obtained by

$$\int_{T_j} \mathbf{B}_1^T \frac{\partial \mathbf{u}_h}{\partial \tau} \, dV + \int_{T_j} \mathbf{B}_1^T \mathrm{div} \mathbf{F} \, dV = \int_{T_j} \mathbf{B}_1^T \mathbf{S} \, dV, \tag{9}$$

which becomes by integration by parts

$$\mathbf{M}_1 \frac{\partial \mathbf{U}_1}{\partial \tau} + \oint_{\partial T_j} \mathbf{B}_1^T \mathbf{F}_n \, dS - \int_{T_j} (\mathrm{grad} \, \mathbf{B}_1^T) : \mathbf{F} \, dV = \int_{T_j} \mathbf{B}_1^T \mathbf{S} \, dV. \tag{10}$$

where $\mathbf{M}_1$ is the mass matrix,

$$\mathbf{M}_1 = \int_{T_j} \mathbf{B}_1^T \mathbf{B}_1 \, dV. \tag{11}$$

This is the pseudo-time evolution equations for the vector of unknown coefficients $\mathbf{U}_1$ in the element T_j. It is straightforward to extend it to higher-order polynomials. For the P_k polynomial approximation,

$$\mathbf{u}_h = \mathbf{B}_k \mathbf{U}_k, \tag{12}$$

the Galerkin discretization is obtained as

$$\mathbf{M}_k \frac{\partial \mathbf{U}_k}{\partial \tau} + \oint_{\partial T_j} \mathbf{B}_k^T \mathbf{F}_n \, dS - \int_{T_j} (\mathrm{grad} \, \mathbf{B}_k^T) : \mathbf{F} \, dV = \int_{T_j} \mathbf{B}_k^T \mathbf{S} \, dV. \tag{13}$$

where $\mathbf{M}_k$ is the mass matrix,

$$\mathbf{M}_k = \int_{T_j} \mathbf{B}_k^T \mathbf{B}_k \, dV. \tag{14}$$

2 Reduced DG Discretization for Hyperbolic Diffusion

To construct a DG method with reduced number of coefficients, we begin by defining a reduced polynomial approximation:

$$\mathbf{u}_h = \mathbf{C}_k \mathbf{V}_k, \tag{15}$$

where $\mathbf{C}_k$ is a modified basis matrix and $\mathbf{V}_k$ is a vector of unknown coefficients such that

$$m < n, \quad \mathbf{V}_k \in R^m, \quad \mathbf{U}_k \in R^n. \tag{16}$$

Once we define $\mathbf{C}_k\mathbf{V}_k$, the Galekrin discretization is obtained straightforwardly by multiplying the target equation by the modified basis functions $\mathbf{C}_k$ and integrating by parts:

$$\mathbf{M}_k\frac{\partial \mathbf{V}_k}{\partial \tau} + \oint_{\partial T_j} \mathbf{C}_k^T \mathbf{F}_n \, dS - \int_{T_j} (\mathrm{grad}\, \mathbf{C}_k^T) : \mathbf{F} \, dV = \int_{T_j} \mathbf{C}_k^T \mathbf{S} \, dV. \tag{17}$$

where $\mathbf{M}_k$ is the mass mattix,

$$\mathbf{M}_k = \int_{T_j} \mathbf{C}_k^T \mathbf{C}_k \, dV. \tag{18}$$

This is the pseudo-time evolution equations for $\mathbf{V}_k$ in the element T_j.

2.1 P_1 Case

In the P_1 case, u_x and u_y can be replaced by $\overline{p}$ and $\overline{q}$, respectively. Also, we can introduce a single coefficient v_{xy} to represent both p_y and q_x. That is, instead of

$$u_h = \overline{u} + u_x\phi_x + u_y\phi_y, \tag{19}$$
$$p_h = \overline{p} + p_x\phi_x + p_y\phi_y, \tag{20}$$
$$q_h = \overline{q} + q_x\phi_x + q_y\phi_y, \tag{21}$$

where $\phi_x = \phi_j^{(1)}$ and $\phi_y = \phi_j^{(2)}$, we define

$$u_h = \overline{u} + \overline{p}\phi_x + \overline{q}\phi_y, \tag{22}$$
$$p_h = \overline{p} + p_x\phi_x + v_{xy}\phi_y, \tag{23}$$
$$q_h = \overline{q} + v_{xy}\phi_x + q_y\phi_y, \tag{24}$$

which can be written (by replacing $\mathbf{U}_1$ by $\tilde{\mathbf{U}}_1$ in Equation (5)) as

$$\mathbf{u}_h = \mathbf{B}_1\tilde{\mathbf{U}}_1 = \mathbf{B}_1\mathbf{Z}_1\mathbf{V}_1, \tag{25}$$

where

$$\tilde{\mathbf{U}}_1 = [\overline{u}, \overline{p}, \overline{q}, \overline{p}, p_x, v_{xy}, \overline{q}, v_{xy}, q_y]^T, \tag{26}$$
$$\mathbf{V}_1 = [\overline{u}, \overline{p}, \overline{q}, p_x, v_{xy}, q_y]^T, \tag{27}$$

$$\mathbf{Z}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{28}$$

Finally, the polynomial u_h can be upgraded to quadratic because the second derivatives are available as p_x, v_{xy}, q_y:

$$u_h = \overline{u} + \overline{p}\phi_x + \overline{q}\phi_y + p_x\phi_{xx} + v_{xy}\phi_{xy} + q_y\phi_{yy}, \tag{29}$$

where $\phi_{xx} = \phi_j^{(3)}$, $\phi_{xy} = \phi_j^{(4)}$, and $\phi_{yy} = \phi_j^{(5)}$. The extra quadratic terms are added as follows:

$$\mathbf{u}_h = \mathbf{B}_1\tilde{\mathbf{U}}_1 + \mathbf{r}_u\mathbf{c}_1^T\mathbf{V}_1 = (\mathbf{B}_1\mathbf{Z}_1 + \mathbf{r}_u\mathbf{c}_1^T)\mathbf{V}_1 = \mathbf{C}_1\mathbf{V}_1, \tag{30}$$

where $\mathbf{r}_u$ is the vector indicating which variable is given the extra terms and $\mathbf{c}_1^T$ is the vector of extra basis functions:

$$\mathbf{r}_u = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{c}_1^T = [0, 0, 0, \phi_{xx}, \phi_{xy}, \phi_{yy}]. \tag{31}$$

The advective part will then be upgraded to third-order accurate provided quadrature formulas are also upgraded. These extra quadratic terms are optional in P_1, but extra high-order terms in u_h will be a requirement in the P_2 case and beyond for consistency as we will discuss later.

The Galekrin discretization is obtained by multiplying the target equation by the basis functions $\mathbf{C}_1$ and integrating by parts:

$$\mathbf{M}_1\frac{\partial \mathbf{V}_1}{\partial \tau} + \oint_{\partial T_j} \mathbf{C}_1^T\mathbf{F}_n\, dS - \int_{T_j} (\text{grad}\, \mathbf{C}_1^T) : \mathbf{F}\, dV = \int_{T_j} \mathbf{C}_1^T\mathbf{S}\, dV. \tag{32}$$

where $\mathbf{M}$ is the mass mattix,

$$\mathbf{M}_1 = \int_{T_j} \mathbf{C}_1^T\mathbf{C}_1\, dV. \tag{33}$$

This is the pseudo-time evolution equations for the vector of unknown coefficients in the element T_j.

2.2 P_2 Case

In the P_2 case, we begin with the standard quadratic polynomials:

$$u_h = \overline{u} + u_x\phi_x + u_y\phi_y + u_{xx}\phi_{xx} + u_{xy}\phi_{xy} + u_{yy}\phi_{yy}, \tag{34}$$

$$p_h = \overline{p} + p_x\phi_x + p_y\phi_y + p_{xx}\phi_{xx} + p_{xy}\phi_{xy} + p_{yy}\phi_{yy}, \tag{35}$$

$$q_h = \overline{q} + q_x\phi_x + q_y\phi_y + q_{xx}\phi_{xx} + q_{xy}\phi_{xy} + q_{yy}\phi_{yy}. \tag{36}$$

The coefficients u_x and u_y are the point values at the centroid of the gradients, and so can be replaced as

$$u_x \leftarrow p_h(x_c, y_c) = \overline{p} + p_{xx}\phi_{xx}^c + p_{xy}\phi_{xy}^c + p_{yy}\phi_{yy}^c, \tag{37}$$

$$u_y \leftarrow q_h(x_c, y_c) = \overline{q} + q_{xx}\phi_{xx}^c + q_{xy}\phi_{xy}^c + q_{yy}\phi_{yy}^c, \tag{38}$$

where the superscript c denotes the evaluation of the basis function at the centroid. Furthermore, some of the coefficients can be unified as

$$v_{xy} = p_y = q_x, \tag{39}$$

$$v_{xxy} = p_{xy} = q_{xx}, \tag{40}$$

$$v_{xyy} = p_{yy} = q_{xy}. \tag{41}$$

Therefore, we obtain the reduced polynomial approximations as

$$
\begin{aligned}
u_h &= \overline{u} + (\overline{p} + p_{xx}\phi^c_{xx} + v_{xxy}\phi^c_{xy} + v_{xyy}\phi^c_{yy})\phi_x + (\overline{q} + v_{xxy}\phi^c_{xx} + v_{xyy}\phi^c_{xy} + q_{yy}\phi^c_{yy})\phi_y \\
&+ p_x\phi_{xx} + v_{xy}\phi_{xy} + q_y\phi_{yy}, \qquad (42) \\
p_h &= \overline{p} + p_x\phi_x + v_{xy}\phi_y + p_{xx}\phi_{xx} + v_{xxy}\phi_{xy} + v_{xyy}\phi_{yy}, \qquad (43) \\
q_h &= \overline{q} + v_{xy}\phi_x + q_y\phi_y + v_{xxy}\phi_{xx} + v_{xyy}\phi_{xy} + q_{yy}\phi_{yy}. \qquad (44)
\end{aligned}
$$

At this point, it is important to note that u_h is not a valid polynomial approximation:

$$
\begin{aligned}
u_h &= \overline{u} + \overline{p}\phi_x + \overline{q}\phi_y + p_x\phi_{xx} + v_{xy}\phi_{xy} + q_y\phi_{yy} \\
&+ p_{xx}\phi^c_{xx}\phi_x + v_{xxy}\phi^c_{xy}\phi_x + v_{xyy}\phi^c_{yy}\phi_x + v_{xxy}\phi^c_{xx}\phi_y + v_{xyy}\phi^c_{xy}\phi_y + q_{yy}\phi^c_{yy}\phi_y, \qquad (45)
\end{aligned}
$$

which shows that the basis functions

$$
1, \phi_x, \phi_y, \phi_{xx}, \phi_{xy}, \phi_{yy}, \phi^c_{xx}\phi_x, \phi^c_{xy}\phi_x, \phi^c_{yy}\phi_x, \phi^c_{xx}\phi_y, \phi^c_{xy}\phi_y, \phi^c_{yy}\phi_y, \qquad (46)
$$

are not independent, e.g., ϕ_x and $\phi^c_{xx}\phi_x$ are not independent. This leads to an underdetermined problem. For example, the Galerkin discretization of the advection equation $\partial_x u = 0$ with a polynomial $u_h = a + b\phi_x + c\phi_x$ will result in two independent discrete equations for three coefficients (a, b, c). Consequently, the coefficients cannot be determined uniquely, thus leading to inconsistency. Some coefficients could be obtained accurately with others having no accuracy: e.g., a and b are accurate but c is not accurate at all. In the P_2 hyperbolic DG case, $\overline{p}$ and $\overline{q}$ could be accurate and p_{xx}, v_{xxy}, v_{xyy}, and q_{yy} have no accuracy. Of course, there is no guarantee that this is the case. In the worst case, all coefficients have no accuracy. To resolve the issue, we add a cubic term in u_h, which is possible because p_{xx}, v_{xxy}, v_{xyy}, and q_{yy} correspond to the third derivatives of u:

$$
\begin{aligned}
u_h &= \overline{u} + \overline{p}\phi_x + \overline{q}\phi_y + p_x\phi_{xx} + v_{xy}\phi_{xy} + q_y\phi_{yy} \\
&+ p_{xx}\phi^c_{xx}\phi_x + v_{xxy}\phi^c_{xy}\phi_x + v_{xyy}\phi^c_{yy}\phi_x + v_{xxy}\phi^c_{xx}\phi_y + v_{xyy}\phi^c_{xy}\phi_y + q_{yy}\phi^c_{yy}\phi_y, \\
&+ p_{xx}\phi_{xxx} + v_{xxy}\phi_{xxy} + v_{xyy}\phi_{xyy} + q_{yy}\phi_{yyy}, \qquad (47)
\end{aligned}
$$

where $\phi_{xxx} = \phi^{(6)}_j$, $\phi_{xxy} = \phi^{(7)}_j$, $\phi_{xyy} = \phi^{(8)}_j$, and $\phi_{yyy} = \phi^{(9)}_j$, which becomes

$$
\begin{aligned}
u_h &= \overline{u} + \overline{p}\phi_x + \overline{q}\phi_y + p_x\phi_{xx} + v_{xy}\phi_{xy} + q_y\phi_{yy} + p_{xx}(\phi^c_{xx}\phi_x + \phi_{xxx}) + v_{xxy}(\phi^c_{xy}\phi_x + \phi^c_{xx}\phi_y + \phi_{xxy}) \\
&+ v_{xyy}(\phi^c_{yy}\phi_x + \phi^c_{xy}\phi_y + \phi_{xyy}) + q_{yy}(\phi^c_{yy}\phi_y + \phi_{yyy}). \qquad (48)
\end{aligned}
$$

The basis functions are now independent:

$$
1, \phi_x, \phi_y, \phi_{xx}, \phi_{xy}, \phi_{yy}, (\phi^c_{xx}\phi_x + \phi_{xxx}), (\phi^c_{xy}\phi_x + \phi^c_{xx}\phi_y + \phi_{xxy}), (\phi^c_{yy}\phi_x + \phi^c_{xy}\phi_y + \phi_{xyy}), (\phi^c_{yy}\phi_y + \phi_{yyy}). \quad (49)
$$

In this sense, the one-order-higher polynomial in u_h is not an option but a requirement in higher-order schemes. In summary, we have the following polynomial approximation in the P_2 hyperbolic DG:

$$
\begin{aligned}
u_h &= \overline{u} + \overline{p}\phi_x + \overline{q}\phi_y + p_x\phi_{xx} + v_{xy}\phi_{xy} + q_y\phi_{yy} + p_{xx}(\phi^c_{xx}\phi_x + \phi_{xxx}) + v_{xxy}(\phi^c_{xy}\phi_x + \phi^c_{xx}\phi_y + \phi_{xxy}) \\
&+ v_{xyy}(\phi^c_{yy}\phi_x + \phi^c_{xy}\phi_y + \phi_{xyy}) + q_{yy}(\phi^c_{yy}\phi_y + \phi_{yyy}), \qquad (50) \\
p_h &= \overline{p} + p_x\phi_x + v_{xy}\phi_y + p_{xx}\phi_{xx} + v_{xxy}\phi_{xy} + v_{xyy}\phi_{yy}, \qquad (51) \\
q_h &= \overline{q} + v_{xy}\phi_x + q_y\phi_y + v_{xxy}\phi_{xx} + v_{xyy}\phi_{xy} + q_{yy}\phi_{yy}. \qquad (52)
\end{aligned}
$$

which is written as

$$\mathbf{u}_h = \mathbf{B}_2\tilde{\mathbf{U}}_2 + \mathbf{r}_u\mathbf{c}_2^T\mathbf{V}_2 = (\mathbf{B}_2\mathbf{Z}_2 + \mathbf{r}_u\mathbf{c}_2^T)\mathbf{V}_2 = \mathbf{C}_2\mathbf{V}_2, \tag{53}$$

where

$$\mathbf{r}_u = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{c}_2^T = [0,0,0,0,0,0,\phi_{xxx},\phi_{xxy},\phi_{xyy},\phi_{yyy}]. \tag{54}$$

$$\tilde{\mathbf{U}}_2 = \begin{bmatrix} \overline{u} \\ \overline{p} \\ \overline{q} \\ \overline{p} + p_{xx}\phi^c_{xx} + v_{xxy}\phi^c_{xy} + v_{xyy}\phi^c_{yy} \\ p_x \\ v_{xy} \\ \overline{q} + v_{xxy}\phi^c_{xx} + v_{xyy}\phi^c_{xy} + q_{yy}\phi^c_{yy} \\ v_{xy} \\ q_y \\ p_x \\ p_{xx} \\ v_{xxy} \\ v_{xy} \\ v_{xxy} \\ v_{xyy} \\ q_y \\ v_{xyy} \\ q_{yy} \end{bmatrix}, \quad \mathbf{V}_2 = \begin{bmatrix} \overline{u} \\ \overline{p} \\ \overline{q} \\ p_x \\ v_{xy} \\ q_y \\ p_{xx} \\ v_{xxy} \\ v_{xyy} \\ q_{yy} \end{bmatrix}, \tag{55}$$

$$
\mathbf{Z}_2 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & \phi^c_{xx} & \phi^c_{xy} & \phi^c_{yy} & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & \phi^c_{xx} & \phi^c_{xy} & \phi^c_{yy} \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}. \tag{56}
$$

The Galekrin discretization is obtained by multiplying the target equation by the basis functions $\mathbf{C}_2$ and integrating by parts:

$$
\mathbf{M}_2 \frac{\partial \mathbf{V}_2}{\partial \tau} + \oint_{\partial T_j} \mathbf{C}_2^T \mathbf{F}_n \, dS - \int_{T_j} (\mathrm{grad}\, \mathbf{C}_2^T) : \mathbf{F} \, dV = \int_{T_j} \mathbf{C}_2^T \mathbf{S} \, dV. \tag{57}
$$

where $\mathbf{M}_2$ is the mass mattix,

$$
\mathbf{M}_2 = \int_{T_j} \mathbf{C}_2^T \mathbf{C}_2 \, dV. \tag{58}
$$

This is the pseudo-time evolution equations for the vector of unknown coefficients in the element T_j.

2.3 P_k Case

In the general case, the hyperbolic DG method can be constructed as follows. First, we define a reduced polynomial approximation as

$$
u_h \in \tilde{P}_{k+1}, \quad p_h \in P_k, \quad q_h \in P_k, \tag{59}
$$

where P_k denotes a vector space of polynomials spanned by the Taylor basis functions of degree k, and $\tilde{P}_{k+1}$ denotes a vector space of polynomials spanned by a modified Taylor basis functions of degree $k+1$ (see (49) for $\tilde{P}_3$). Express the numerical solution in terms of a polynomial approximation in the form:

$$\mathbf{u}_h = \mathbf{C}_k \mathbf{V}_k, \tag{60}$$

where the basis function matrix $\mathbf{C}_k$ consists of two parts: the reduction of the unknown coefficients and the addition of extra high-order terms, which are characterized by $\mathbf{B}_k \mathbf{Z}_k$ and $\mathbf{r}\mathbf{c}_k^T$, respectively,

$$\mathbf{C}_k = \mathbf{B}_k \mathbf{Z}_k + \mathbf{r}\mathbf{c}_k^T. \tag{61}$$

Note that $\mathbf{B}_k$ is the baseline DG polynomial basis function matrix of degree k and it is $\mathbf{Z}_k$ that characterizes the reduction operation. Note also that $\mathbf{r}$ specifies the variables that are given extra high-order terms and $\mathbf{c}_k^T$ is the vector of the extra high-order Taylor basis functions of degree $k+1$. Given the polynomial approximation, we perform the Galerkin discretization:

$$\mathbf{M}_k \frac{\partial \mathbf{V}_k}{\partial \tau} + \oint_{\partial T_j} \mathbf{C}_k^T \mathbf{F}_n \, dS - \int_{T_j} (\operatorname{grad} \mathbf{C}_k^T) : \mathbf{F} \, dV = \int_{T_j} \mathbf{C}_k^T \mathbf{S} \, dV. \tag{62}$$

where $\mathbf{M}_k$ is the mass mattix,

$$\mathbf{M}_k = \int_{T_j} \mathbf{C}_k^T \mathbf{C}_k \, dV. \tag{63}$$

A Note on Hyperbolic Method and DG/RDG Methods

Hiroaki Nishikawa

August 18, 2015

1 DG, RDG, and Hyperbolic-RDG

Consider the advection-diffusion equation:

$$\partial_\tau u + \partial_x f = \nu \partial_{xx} u, \tag{1}$$

where τ is a pseudo time, f is a convective flux, and ν is a positive constant diffusion coefficient. The time derivative term has been defined with a pseudo time because we are interested to develop a steady solver, which serves as a nonlinear solver required in implicit time-stepping schemes for unsteady problems.

A P_2 discontinuous Galerkin method based on a polynomial defined in each computational cell would require a piecewise quadratic polynomial to represent the solution u:

$$u_j(x) = \overline{u}_j + (\overline{u_x})_j \phi_x + (\overline{u_{xx}})_j \phi_{xx}, \tag{2}$$

where ϕ_x and ϕ_{xx} are linear and quadratic basis functions, respectively, $\overline{u}_j$, $(\overline{u_x})_j$, and $(\overline{u_{xx}})_j$ are the degrees of freedom. Therefore, the P_2 DG method requires three degrees of freedom per cell. The method is formally third-order accurate. On the other hand, in the reconstructed DG method (RDG), we reconstruct $(\overline{u_{xx}})_j$ from $(\overline{u_x})_j$,

$$u_j(x) = \overline{u}_j + (\overline{u_x})_j \phi_x + (\widetilde{u_{xx}})_j \phi_{xx}, \tag{3}$$

where $(\widetilde{u_{xx}})_j$ denotes a reconstructed value, and thus we need only two degrees of freedom, $\overline{u}_j$ and $(\overline{u_x})_j$, to achieve third-order of accuracy.

In the hyperbolic method, we reformulate the diffusion term as a first-order system that is hyperbolic in the pseudo time:

$$\partial_\tau u + \partial_x f = \nu \partial_x p, \tag{4}$$

$$\partial_\tau p = \frac{1}{T_r} \left(\partial_x u - p \right), \tag{5}$$

where $T_r = L_r^2/\nu$ and $L_r = 1/(2\pi)$; the system has been designed to recover the advection-diffusion equation (1) in the pseudo steady state. Consider a P_0 discontinuous Galerkin method (i.e., a cell-centered finite-volume method), which requires one degree of freedom for each variable:

$$u_j(x) = \overline{u}_j, \tag{6}$$

$$p_j(x) = \overline{p}_j. \tag{7}$$

This leads to a first-order scheme for both the advective and diffusive terms. Note, then, that as we will have $p = \partial_x u$ in the pseudo steady state, we can upgrade $u_j(x)$ to a linear polynomial:

$$u_j(x) = \overline{u}_j + \overline{p}_j \phi_x, \tag{8}$$

$$p_j(x) = \overline{p}_j. \tag{9}$$

This gives second-order accuracy for the advective term while the diffusive-term approximation remains first order accurate. Yet, employing the RDG method, we linearly reconstruct $(\widetilde{p_x})_j$ from $\overline{p}_j$ and upgrade both polynomials by one order:

$$u_j(x) = \overline{u}_j + \overline{p}_j \phi_x + (\widetilde{p_x})_j \phi_{xx}, \tag{10}$$

$$p_j(x) = \overline{p}_j + (\widetilde{p_x})_j \phi_x, \tag{11}$$

which results in third- and second-order accuracy for the advective and diffusive terms, respectively. This method is called the hyperbolic-RDG-L method. Ultimately, we may quadratically reconstruct $(\widetilde{p_x})_j$ from $\overline{p}_j$, so that we will have $(\widetilde{p_{xx}})_j$ as well, and set

$$u_j(x) = \overline{u}_j + \overline{p}_j \phi_x + (\widetilde{p_x})_j \phi_{xx} + (\widetilde{p_{xx}})_j \phi_{xxx}, \tag{12}$$

$$p_j(x) = \overline{p}_j + (\widetilde{p_x})_j \phi_x + (\widetilde{p_{xx}})_j \phi_{xx}. \tag{13}$$

This leads to fourth- and third-order accuracy for the advective and diffusive terms, respectively. This method is called the hyperbolic-RDG-Q method. Note that the discretization is performed purely as a P_0 discontinuous Galerkin method in all cases, and thus the only basis function used in the weak formulation is 1. The number of degrees of freedom is thus equal to that in the RDG method.

Extending the discussion, we have a fourth-order DG method with

$$u_j(x) = \overline{u}_j + (\overline{u_x})_j \phi_x + (\overline{u_{xx}})_j \phi_{xx} + (\overline{u_{xxx}})_j \phi_{xxxx}, , \tag{14}$$

where ϕ_{xxx} is a cubic basis function, and $(\overline{u_{xxx}})_j$ is the corresponding degree of freedom, and a fourth-order RDG method with

$$u_j(x) = \overline{u}_j + (\overline{u_x})_j \phi_x + (\overline{u_{xx}})_j \phi_{xx} + (\widetilde{u_{xxx}})_j \phi_{xxxx}, , \tag{15}$$

where $(\widetilde{u_{xxx}})_j$ denotes a reconstructed value. In the hyperbolic method, we have a P_1 DG method with

$$u_j(x) = \overline{u}_j + (\overline{u_x})_j \phi_x, \tag{16}$$

$$p_j(x) = \overline{p}_j + (\overline{p_x})_j \phi_x. \tag{17}$$

This leads to a second-order scheme. But again we can replace $(\overline{u_x})_j$ by $(\overline{p})_j$ and upgrade the polynomial of u:

$$u_j(x) = \overline{u}_j + (\overline{p})_j \phi_x + (\overline{p_x})_j \phi_{xx}, \tag{18}$$

$$p_j(x) = \overline{p}_j + (\overline{p_x})_j \phi_x, \tag{19}$$

to obtain third-order accuracy in the advective term (and second-order accuracy in the diffusive term). Using the RDG method with a linear reconstruction (Hyperbolic-RDG-L), we obtain

$$u_j(x) = \overline{u}_j + (\overline{p})_j \phi_x + (\overline{p_x})_j \phi_{xx} + (\widetilde{p_{xx}})_j \phi_{xxx}, \tag{20}$$

$$p_j(x) = \overline{p}_j + (\overline{p_x})_j \phi_x + (\widetilde{p_{xx}})_j \phi_{xx}, \tag{21}$$

which gives fourth- and third-order accuracy for the advective and diffusive terms, respectively. Finally, applying a quadratic reconstruction (Hyperbolic-RDG-Q), we obtain

$$u_j(x) = \overline{u}_j + (\overline{p})_j \phi_x + (\overline{p_x})_j \phi_{xx} + (\widetilde{p_{xx}})_j \phi_{xxx} + +(\widetilde{p_{xxx}})_j \phi_{xxxx}, \tag{22}$$

$$p_j(x) = \overline{p}_j + (\overline{p_x})_j \phi_x + (\widetilde{p_{xx}})_j \phi_{xx} + (\widetilde{p_{xxx}})_j \phi_{xxx}, \tag{23}$$

which leads to 5th- and 4th-order accuracy for the advective and diffusive terms, respectively.

Potential advantages of the hyperbolic DG and RDG methods are

1. Simplicity in the discretization, upwind fluxes for all terms.
2. Simple and systematic construction of efficient iterative solvers.
3. The numerical stiffness associated with the second derivative is completely eliminated, implying convergence acceleration for problems where diffusion is important.
4. The quality of the solution gradient on highly irregular stretched grids can be greatly improved in the hyperbolic method while conventional methods are known to generate oscillations.

Accuracy on irregular grids is very important for grid adaptation, which can easily result in 'bad' grids especially for viscous problems. Potential applications can be found in any viscous flow problem for both the incompressible and compressible Navier-Stokes equations. A suitable hyperbolic viscous system is already available.

2 Comparison of Degrees of Freedom

In two dimensions, again for a scalar equation, we consider the advection-diffusion equation:

$$\partial_\tau u + \partial_x f + \partial_y g = \nu(\partial_{xx} u + \partial_{yy} u), \tag{24}$$

where g is a flux in y-direction. The P_2 DG method requires a quadratic polynomial in a computational cell:

$$u_j(x, y) = \overline{u}_j + (\overline{u_x})_j \phi_x + (\overline{u_y})_j \phi_y + (\overline{u_{xx}})_j \phi_{xx} + (\overline{u_{xy}})_j \phi_{xy} + (\overline{u_{yy}})_j \phi_{yy}, \tag{25}$$

where ϕ_y and ϕ_{yy} are linear and quadratic basis functions in y, and ϕ_{xy} is a basis function of a cross term. The polynomial involves six degrees of freedom. The method is formally third-order accurate. On the other hand, the RDG method requires only a linear polynomial, with reconstructed quadratic terms, to achieve third-order accuracy:

$$u_j(x, y) = \overline{u}_j + (\overline{u_x})_j \phi_x + (\overline{u_y})_j \phi_y + (\widetilde{u_{xx}})_j \phi_{xx} + (\widetilde{u_{xy}})_j \phi_{xy} + (\widetilde{u_{yy}})_j \phi_{yy}, \tag{26}$$

and thus only three degrees of freedom are required. In the hyperbolic method, we discretize the following system:

$$\partial_\tau u + \partial_x f + \partial_y g = \nu(\partial_x p + \partial_y q), \tag{27}$$

$$\partial_\tau p = \frac{1}{T_r}(\partial_x u - p), \tag{28}$$

$$\partial_\tau q = \frac{1}{T_r}(\partial_y u - q). \tag{29}$$

As in the 1D case, it suffices to have a piecewise constant representation for each variable:

$$u_j(x, y) = \overline{u}_j, \tag{30}$$

$$p_j(x, y) = \overline{p}_j, \tag{31}$$

$$q_j(x, y) = \overline{q}_j, \tag{32}$$

to achieve third-order accuracy in the advective term because we can upgrade the polynomials, by reconstruction, as follows:

$$u_j(x, y) = \overline{u}_j + \overline{p}_j \phi_x + \overline{q}_j \phi_y + (\widetilde{p_x})_j \phi_{xx} + (\widetilde{p_y})_j \phi_{xy} + (\widetilde{q_y})_j \phi_{yy}, \tag{33}$$

$$p_j(x, y) = \overline{p}_j + (\widetilde{p_x})_j \phi_x + (\widetilde{p_y})_j \phi_y, \tag{34}$$

$$q_j(x, y) = \overline{q}_j + (\widetilde{q_x})_j \phi_x + (\widetilde{q_y})_j \phi_y. \tag{35}$$

This is a two-dimensional version of Hyperbolic-RDG-L: 3rd-order advective term and 2nd-order diffusive term. In the case of Hyperbolic-RDG-Q, we have

$$u_j(x, y) = \overline{u}_j + \overline{p}_j \phi_x + \overline{q}_j \phi_y + (\widetilde{p_x})_j \phi_{xx} + (\widetilde{p_y})_j \phi_{xy} + (\widetilde{q_y})_j \phi_{yy} \tag{36}$$

$$+ (\widetilde{p_{xx}})_j \phi_{xxx} + (\widetilde{p_{xy}})_j \phi_{xxy} + (\widetilde{p_{yy}})_j \phi_{xyy} + (\widetilde{q_{yy}})_j \phi_{yyy}, \tag{37}$$

$$p_j(x, y) = \overline{p}_j + (\widetilde{p_x})_j \phi_x + (\widetilde{p_y})_j \phi_y + (\widetilde{p_{xx}})_j \phi_{xx} + (\widetilde{p_{xy}})_j \phi_{xy} + (\widetilde{p_{yy}})_j \phi_{yy}, \tag{38}$$

$$q_j(x, y) = \overline{q}_j + (\widetilde{q_x})_j \phi_x + (\widetilde{q_y})_j \phi_y + (\widetilde{q_{xx}})_j \phi_{xx} + (\widetilde{q_{xy}})_j \phi_{xy} + (\widetilde{q_{yy}})_j \phi_{yy}, \tag{39}$$

which gives 4th-order advective term and 3rd-order diffusive term. Therefore, the hyperbolic-RDG(P_0)-Q method is a P_0P_3 method for advection and a P_0P_2 method for diffusion. In all case, the total number of degrees of freedom is three, which is the same, again, as that of the RDG method. Similarly, in 3D also, the hyperbolic-RDG method and the RDG method require the same number of degrees of freedom for third-order accuracy in the advective term, at least. See Table 1.

Extending the discussion to fourth-order accuracy, we obtain the results shown in Table 2. As shown in the table, the hyperbolic-RDG requires slightly more degrees of freedom in 2D and 3D. This is due to the fact,

Dimension	DG(P_2) 3rd/3rd	RDG(P_1) 3rd/3rd	Hyp-DG(P_0) 2nd/1st	Hyp-RDG(P_0)-L 3rd/2nd	Hyp-RDG(P_0)-Q 4th/3rd
1D	$3: \overline{u}_j, (\overline{u_x})_j, (\overline{u_{xx}})_j$	$2: \overline{u}_j, (\overline{u_x})_j$	$2: \overline{u}_j, (\overline{p})_j$	$2: \overline{u}_j, (\overline{p})_j$	$2: \overline{u}_j, (\overline{p})_j$
2D	6	3	3	3	3
3D	10	4	4	4	4

Table 1: Degrees of freedom per cell required for 3rd-order accuracy for a scalar equation. Orders of accuracy are indicated below each method name for advection and diffusion terms.

Dimension	DG(P_3) 4th/4th	RDG(P_2) 4th/4th	Hyp-DG(P_1) 3rd/2nd	Hyp-RDG(P_1)-L 4th/3rd	Hyp-RDG(P_1)-Q 5th/4th
1D	$4: \overline{u}_j, (\overline{u_x})_j, (\overline{u_{xx}})_j, (\overline{u_{xxx}})_j$	$3: \overline{u}_j, (\overline{u_x})_j, (\overline{u_{xx}})_j$	$3: \overline{u}_j, (\overline{p})_j, (\overline{p_x})_j$	$3: \overline{u}_j, (\overline{p})_j, (\overline{p_x})_j$	$3: \overline{u}_j, (\overline{p})_j, (\overline{p_x})_j$
2D	10	6	7 or 6	7 or 6	7 or 6
3D	20	10	13 or 10	13 or 10	13 or 10

Table 2: Degrees of freedom per cell required for 4th-order accuracy for a scalar equation. The numbers in red are possible if the high-order moments corresponding to cross derivatives are unified. Orders of accuracy are indicated below each method name for advection and diffusion terms.

Variable	DG(P_2)	RDG(P_1)	Hyp-DG(P_0)	Hyp-RDG(P_0)-L	Hyp-RDG(P_0)-Q
Primal, u	$O(h^3)$	$O(h^3)$	$O(h^2)$	$O(h^2)$ - $O(h^3)$	$O(h^4)$ - $O(h^3)$
Gradients	$O(h^2)$	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h^3)$

Table 3: Order of accuracy expected for the primal solution variable u and the gradients compared with third-order DG methods.

Variable	DG(P_3)	RDG(P_2)	Hyp-DG(P_1)	Hyp-RDG(P_1)-L	Hyp-RDG(P_1)-Q
Primal, u	$O(h^4)$	$O(h^4)$	$O(h^3)$	$O(h^3)$ - $O(h^4)$	$O(h^4)$ - $O(h^5)$
Gradients	$O(h^3)$	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^4)$

Table 4: Order of accuracy expected for the primal solution variable u and the gradients compared with fourth-order DG methods.

for example, that the variables $\overline{p_x}_j$ and $\overline{q_y}_j$ both represent the same derivative $\partial_{xy} u$. That is, the same quantity is represented by more than one variable in the hyperbolic-RDG method. It is possible to match the number of degrees of freedom between the RDG and the hyperbolic-RDG by replacing $\overline{q_y}_j$ by $\overline{p_x}_j$, for example: Instead of

$$\begin{aligned} u_j(x,y) &= \overline{u}_j + \overline{p}_j\phi_x + \overline{q}_j\phi_y + (\overline{p_x})_j\phi_{xx} + (\overline{p_y})_j\phi_{xy} + (\overline{q_y})_j\phi_{yy} \\ &+ (\widetilde{p_{xx}})_j\phi_{xxx} + (\widetilde{p_{xy}})_j\phi_{xxy} + (\widetilde{q_{xy}})_j\phi_{xxy} + (\widetilde{q_{yy}})_j\phi_{yyy} \end{aligned} \tag{40}$$

$$p_j(x,y) = \overline{p}_j + (\overline{p_x})_j\phi_x + (\overline{p_y})_j\phi_y + (\widetilde{p_{xx}})_j\phi_{xx} + (\widetilde{p_{xy}})_j\phi_{xy} + (\widetilde{p_{yy}})_j\phi_{yy}, \tag{41}$$

$$q_j(x,y) = \overline{q}_j + (\overline{q_x})_j\phi_x + (\overline{q_y})_j\phi_y + (\widetilde{q_{xx}})_j\phi_{xx} + (\widetilde{q_{xy}})_j\phi_{xy} + (\widetilde{q_{yy}})_j\phi_{yy}, \tag{42}$$

where ϕ_{xxx}, ϕ_{xxy}, ϕ_{xyy}, and ϕ_{yyy} are cubic basis functions, we replace $(\overline{p_y})_j$ and $(\overline{q_x})_j$ by a common degree of freedom denoted by $(\overline{u_{xy}})_j$:

$$\begin{aligned} u_j(x,y) &= \overline{u}_j + \overline{p}_j\phi_x + \overline{q}_j\phi_y + (\overline{p_x})_j\phi_x + (\overline{u_{xy}})_j\phi_{xy} + (\overline{q_y})_j\phi_{yy} \\ &+ (\widetilde{p_{xx}})_j\phi_{xxx} + (\widetilde{p_{xy}})_j\phi_{xxy} + (\widetilde{q_{xy}})_j\phi_{xyy} + (\widetilde{q_{yy}})_j\phi_{yyy} \end{aligned} \tag{43}$$

$$p_j(x,y) = \overline{p}_j + (\overline{p_x})_j\phi_x + (\overline{u_{xy}})_j\phi_y + (\widetilde{p_{xx}})_j\phi_{xx} + (\widetilde{p_{xy}})_j\phi_{xy} + (\widetilde{p_{yy}})_j\phi_{yy}, \tag{44}$$

$$q_j(x,y) = \overline{q}_j + (\overline{u_{xy}})_j\phi_x + (\overline{q_y})_j\phi_y + (\widetilde{q_{xx}})_j\phi_{xx} + (\widetilde{q_{xy}})_j\phi_{xy} + (\widetilde{q_{yy}})_j\phi_{yy}. \tag{45}$$

If this strategy works, then we match the degrees of freedom between the RDG and the hyperbolic RDG for fourth-order accuracy in all dimensions.

In the RDG method, the reconstructed polynomials are used only in the flux and source computations. It means that only the evolution equation for $\overline{u}_j$ is derived from the equation for u. Then, the pseudo-time evolution equation of the common degree of freedom $(\overline{u_{xy}})_j$ needs to be derived either from the equation for p or the equation for q, or a combination of them. How we can actually do so remains to be determined.

3 Remarks

1. The Hyp-RDG-Q method may encounter issues in practical problems as it requires many neighbors for a quadratic reconstruction. But this is the best method, which potentially outperforms the RDG method as it achieves one order higher order of accuracy for the advective part for the same number of degrees of freedom.

2. It is still not clear how to unify the degrees of freedom corresponding to cross derivatives. What equation needs to be solved? I think this is an interesting point to consider in extending the methodology to arbitrarily high-order accuracy. Of course, we can proceed with a full set of degrees of freedom; we just need to carry more degrees of freedom (e.g., 65 instead of 50 in the case of the 3D compressible Navier-Stokes system).

3. The Hyp-RDG-L method is the one comparable to the DG and RDG methods in terms of accuracy for (almost) the same number of degrees of freedom. The approximation order to the diffusive term is one order lower, but it does produce the gradient to the same order of accuracy because the order of accuracy for the gradient is one order lower in the DG and RDG methods. The difference lies in the order of accuracy for u, which can be the same only for advection dominated cases; otherwise one order lower (but still with the same order of accuracy in the gradients).

4. If we compare the DG and hyperbolic-DG methods, we see that Hyp-DG(P_1) is comparable with DG(P_2). In general, Hyp-DG(P_n) is comparable with DG(P_{n+1}).

5. I'm interested in higher-order accuracy, i.e., higher than third-order.