The Runge–Kutta discontinuous Galerkin method with compact stencils for hyperbolic conservation laws

Qifan Chen [1], Zheng Sun [2], Yulong Xing [1]

[1]The Ohio State University

[2]The University of Alabama

SIAM Central States Section, Oct 1, 2022

Outline

1. Introduction

2. Numerical schemes for compact RKDG

3. Numerical experiments

4. Conclusions and future work

Overview

- Discontinuous Galerkin (DG) methods: a class of finite element methods using discontinuous basis functions
- Time discretization: often use Runge-Kutta schemes
- DG: flexibility, local data structure, parallel implementation...
- Compact Runge-Kutta DG(cRKDG): more compactness, less communication and easier boundary treatments

Introduction

Discontinuous Galerkin methods

- Consider the hyperbolic conservation laws:

$$\partial_t u + \nabla \cdot f(u) = 0$$

- Partition the domain Ω into elements $K \in T_h$
- Define finite element space consisting of piecewise polynomials:

$$V_h^k = \left\{ v : v|_{I_i} \in P^k(K); 1 \leq i \leq N \right\}$$

- Integrate by parts to get semi-discrete DG scheme:

$$\int_K [(u_h)_t]\, v_h \mathrm{d}x - \int_K f_i(u_h) \cdot \nabla v_h \mathrm{d}x + \int_{\partial K} n \cdot \hat{f} v_h \mathrm{d}s = 0 \quad \forall v_h \in V_h.$$

with numerical flux function $\hat{f}$ (Godunov, upwind, Lax-Friedrichs, etc)

Discontinous Galerkin methods

- Define $\nabla^{\mathrm{DG}}\cdot$ such that

$$\int_K \nabla^{\mathrm{DG}} \cdot f(u_h) v_h \mathrm{dx} = -\int_K f(u_h) \cdot \nabla v_h \mathrm{dx} + \int_{\partial K} n \cdot \hat{f} v_h \mathrm{ds}, \quad \forall v_h \in V_h$$

- Rewrite the strong form for semi-discrete DG scheme:

$$\partial_t u_h + \nabla^{\mathrm{DG}} \cdot f(u_h) = 0.$$

$$\int_K \left[(u_h)_t \right] v_h \mathrm{dx} - \int_K f_i \left(u_h \right) \cdot \nabla v_h \mathrm{dx} + \int_{\partial K} n \cdot \hat{f} v_h \mathrm{ds} = 0 \quad \forall v_h \in V_h.$$

Runge-Kutta schemes

- Consider an explicit RK method associated with the Butcher Tableau

$$\begin{array}{c|c} c & A \\ \hline & b \end{array}, \quad A = (a_{ij})_{s\times s}, \quad b = (b_1, \cdots b_s)$$

- We get the corresponding RKDG scheme as

$$\begin{aligned} u_h^{(i)} =& u_h^n - \Delta t \sum_{j=1}^{s} a_{ij} \nabla^{\mathrm{DG}} \cdot f\left(u_h^{(j)}\right), \quad i = 1, 2, \cdots s, \\ u_h^{n+1} =& u_h^n - \Delta t \sum_{i=1}^{s} b_i \nabla^{\mathrm{DG}} \cdot f\left(u_h^{(i)}\right). \end{aligned}$$

Note we have $u_h^{(1)} = u_h^n$

Numerical schemes for compact RKDG

compact RKDG methods

- We define a local operator as projected local derivative

$$\nabla^{\mathrm{loc}} \cdot f\left(u_h\right) = \Pi(\nabla \cdot f\left(u_h\right)),$$

where Π is the L^2 projection to V_h.

- Making use of the local spatial operator $\nabla^{\mathrm{loc}}\cdot$, we rewrite

$$\int_K \nabla^{\mathrm{loc}} \cdot f\left(u_h\right) v_h \mathrm{d}x = -\int_K f\left(u_h\right) \cdot \nabla v_h \mathrm{d}x + \int_{\partial K} n \cdot f\left(u_h\right) v_h \mathrm{d}s, \ \forall v_h \in V_h.$$

$$\int_K \nabla^{\mathrm{DG}} \cdot f(u_h) v_h \mathrm{d}x = -\int_K f(u_h) \cdot \nabla v_h \mathrm{d}x + \int_{\partial K} n \cdot \hat{f} v_h \mathrm{d}s, \quad \forall v_h \in V_h.$$

compact RKDG methods

- Replacing $\nabla^{\mathrm{DG}}\cdot$ in inner stages with $\nabla^{\mathrm{loc}}\cdot$, we can write new scheme as

$$
\begin{aligned}
u_h^{(i)} =& u^n - \Delta t \sum_{j=1}^{i-1} a_{ij} \nabla^{\mathrm{loc}} \cdot f\left(u_h^{(j)}\right), \quad i = 1, 2, \cdots s, \\
u_h^{n+1} =& u_h^n - \Delta t \sum_{i=1}^{s} b_i \nabla^{\mathrm{DG}} \cdot f\left(u_h^{(i)}\right).
\end{aligned}
$$

- Remark:
  - Standard DG operators are needed in the final stage
  - Limiters are only applied at the final stage

compact RKDG methods

Two examples:

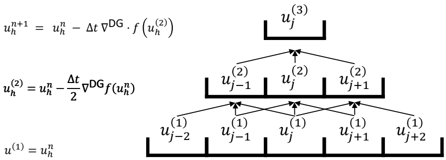
- *Second-order scheme.*

$$\begin{aligned} u_h^{(2)} &= u_h^n - \frac{\Delta t}{2} \nabla^{\mathrm{DG}} \cdot f\left(u_h^n\right), \\ u_h^{n+1} &= u_h^n - \Delta t \nabla^{\mathrm{DG}} \cdot f\left(u_h^{(2)}\right). \end{aligned} \Longrightarrow \begin{aligned} u_h^{(2)} &= u_h^n - \frac{\Delta t}{2} \nabla^{\mathrm{loc}} \cdot f\left(u_h^n\right), \\ u_h^{n+1} &= u_h^n - \Delta t \nabla^{\mathrm{DG}} \cdot f\left(u_h^{(2)}\right). \end{aligned}$$

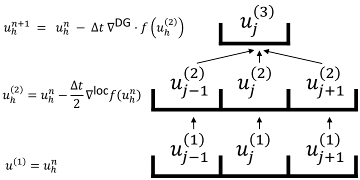
- *Third-order scheme.*

$$\begin{aligned} u_h^{(2)} &= u_h^n - \frac{1}{3} \Delta t \nabla^{\mathrm{DG}} \cdot f\left(u_h^n\right), \\ u_h^{(3)} &= u_h^n - \frac{2}{3} \Delta t \nabla^{\mathrm{DG}} \cdot f\left(u_h^{(2)}\right), \\ u_h^{n+1} &= u_h^n - \Delta t \nabla^{\mathrm{DG}} \cdot \left(\frac{1}{4} f\left(u_h^{(n)}\right) + \frac{3}{4} f\left(u_h^{(3)}\right)\right). \end{aligned} \Longrightarrow \begin{aligned} u_h^{(2)} &= u_h^n - \frac{1}{3} \Delta t \nabla^{\mathrm{loc}} \cdot f\left(u_h^n\right), \\ u_h^{(3)} &= u_h^n - \frac{2}{3} \Delta t \nabla^{\mathrm{loc}} \cdot f\left(u_h^{(2)}\right), \\ u_h^{n+1} &= u_h^n - \Delta t \nabla^{\mathrm{DG}} \cdot \left(\frac{1}{4} f\left(u_h^{(n)}\right) + \frac{3}{4} f\left(u_h^{(3)}\right)\right). \end{aligned}$$

- For example, in one dimensional case with Lax–Friedrichs fluxes, stencil size for s-stage RK method is $2s+1$, while for cRKDG scheme is identically 3



(a) RKDG



(b) cRKDG

Convergence

Theorem (Lax-Wendroff type theorem)

If u_h^n converges boundedly almost everywhere to some function u as $\Delta t, h \to 0$, then u is a weak solution to the conservation law, namely

$$\int_{\mathbb{R}^d} u_0 \phi \mathrm{d}x + \int_{\mathbb{R}^+ \times \mathbb{R}^d} u \phi_t \mathrm{d}x + \int_{\mathbb{R}^+ \times \mathbb{R}^d} f(u) \cdot \nabla \phi \mathrm{d}x = 0,$$

$$\forall \phi \in C_0^\infty(\mathbb{R}^+ \times \mathbb{R}^d), \text{ where } u(\cdot, 0) = u_0 \text{ in } \mathbb{R}$$

Following the argument in [Shi and Shu, 2018], with standard conditions

- f is Lipschitz continuous and f', f'' are uniformly bounded in L^∞
- Numerical flux $\hat{f} \cdot (u_h)$ satisfies consistency and Lipschitz continuity
- $\Delta t / h \leq C$ for some fixed constant C

Boundary error

- For RK schemes with time dependent Dirichlet boundary conditions, imposing exact boundary values for inner stages will reduce the accuracy [Carpenter et al., 1995]
- RKDG also suffers from this reduction of accuracy [Zhang, 2011]
- Our methods use the local solution in inner stages and will automatically achieve the optimal convergence rate

	N	L^2 error	order	L^∞ error	order
periodic $u(0,t)=u(4\pi,t)$ SSP-RK3	40	4.9340e-04	-	5.1206e-04	-
	80	5.9520e-05	3.05	6.5063e-05	2.98
	160	7.3468e-06	3.02	8.1871e-06	2.99
	320	9.1377e-07	3.01	1.0270e-06	2.99
	640	1.1397e-07	3.00	1.2858e-07	3.00
	1280	1.4232e-08	3.00	1.6080e-08	3.00
inflow $u(0,t)=\sin(-t)$ SSP-RK3	40	4.0905e-04	-	4.9561e-04	-
	80	5.1156e-05	3.00	6.2417e-05	2.99
	160	6.4875e-06	2.98	7.8358e-06	2.99
	320	8.7923e-07	2.88	1.5444e-06	2.34
	640	1.1747e-07	2.90	3.3560e-07	2.20
	1280	1.5805e-08	2.89	6.6682e-08	2.33

Table: Errors table in [Zhang, 2011]

Connections with other schemes

- cRKDG is equivalent to Lax-Wendroff DG [Qiu et al., 2005] for linear conservation laws with constant coefficients, namely

$$\partial_t u + \nabla \cdot (Au) = 0, \quad A \text{ is constant}$$

- cRKDG is similar to ADER DG schemes with a local predictor

Numerical experiments

Accuracy tests: linear scalar problem

$$\partial_t u + \partial_x u = 0.$$

with the exact solution $u(x,t) = \sin(x - t)$. We use upwind flux and compute to $T = 0.2$. The method is optimal for the RK$(k+1)$-$\mathcal{P}^k$-DG method with $k = 1, 2, 3$.

k	N	L^1 error	order	L^2 error	order	L^∞ error	order
1	20	1.3370E-02	-	6.9430E-03	-	7.6863E-03	-
	40	3.3393E-03	2.00	1.7429E-03	1.99	1.9443E-03	1.98
	80	8.3379E-04	2.00	4.3597E-04	2.00	4.8794E-04	1.99
	160	2.0816E-04	2.00	1.0898E-04	2.00	1.2219E-04	2.00
	320	5.2004E-05	2.00	2.7240E-05	2.00	3.0567E-05	2.00
2	20	3.6114E-04	-	1.7639E-04	-	1.6517E-04	-
	40	4.9933E-05	2.85	2.4829E-05	2.83	2.1925E-05	2.91
	80	6.0194E-06	3.05	3.0473E-06	3.03	2.6587E-06	3.04
	160	7.5124E-07	3.00	3.7942E-07	3.01	3.3173E-07	3.00
	320	9.4039E-08	3.00	4.7455E-08	3.00	4.1477E-08	3.00
3	20	9.6002E-06	-	5.2699E-06	-	5.1792E-06	-
	40	6.5906E-07	3.86	3.1914E-07	4.05	2.5855E-07	4.32
	80	3.1405E-08	4.39	1.6915E-08	4.24	1.6166E-08	4.00
	160	2.1497E-09	3.87	1.1030E-09	3.94	9.5995E-10	4.07
	320	1.2943E-10	4.05	6.5656E-11	4.07	5.6293E-11	4.09
	640	8.1123E-12	4.00	4.1202E-12	3.99	3.5426E-12	3.99

Table: Linear advection equation. $k = 1, 2, 3$. $\Delta t = 0.1\Delta x$.

Accuracy test: 1d boundary error

	N	L^1 error	order	L^2 error	order	L^∞ error	order
periodic $u(0,t)=u(4\pi,t)$ ssprk3	40	1.4845e-03	-	4.9340e-04	-	5.1206e-04	-
	80	1.7657e-04	3.07	5.9520e-05	3.05	6.5063e-05	2.98
	160	2.1649e-05	3.03	7.3468e-06	3.02	8.1871e-06	2.99
	320	2.6843e-06	3.01	9.1377e-07	3.01	1.0270e-06	2.99
	640	3.3430e-07	3.01	1.1397e-07	3.00	1.2858e-07	3.00
	1280	4.1719e-08	3.00	1.4232e-08	3.00	1.6080e-08	3.00
inflow $u(0,t)=\sin(-t)$ ssprk3	40	1.1079e-03	-	4.0905e-04	-	4.9561e-04	-
	80	1.3651e-04	3.02	5.1156e-05	3.00	6.2417e-05	2.99
	160	1.7018e-05	3.00	6.4875e-06	2.98	7.8358e-06	2.99
	320	2.1467e-06	2.99	8.7923e-07	2.88	1.5444e-06	2.34
	640	2.6788e-07	3.00	1.1747e-07	2.90	3.3560e-07	2.20
	1280	3.3375e-08	3.00	1.5805e-08	2.89	6.6682e-08	2.33
periodic $u(0,t)=u(4\pi,t)$ cRKDG3	40	5.7366e-03	-	1.7975e-03	-	7.4848e-04	-
	80	7.1035e-04	3.01	2.2264e-04	3.01	9.1854e-05	3.03
	160	8.8527e-05	3.00	2.7740e-05	3.00	1.1503e-05	3.00
	320	1.1025e-05	3.01	3.4547e-06	3.01	1.4249e-06	3.01
	640	1.3780e-06	3.00	4.3180e-07	3.00	1.7853e-07	3.00
	1280	1.7235e-07	3.00	5.4006e-08	3.00	2.2448e-08	2.99
inflow $u(0,t)=\sin(-t)$ cRKDG3	40	2.1823e-03	-	7.4246e-04	-	4.6480e-04	-
	80	2.7098e-04	3.01	9.2143e-05	3.01	5.9471e-05	2.97
	160	3.3919e-05	3.00	1.1519e-05	3.00	7.5423e-06	2.98
	320	4.1668e-06	3.03	1.4202e-06	3.02	9.3907e-07	3.01
	640	5.2328e-07	2.99	1.7813e-07	3.00	1.1818e-07	2.99
	1280	6.5867e-08	2.99	2.2384e-08	2.99	1.4867e-08	2.99

Table: $\partial_t u + \partial_x u = 0.$ on $(0, 4\pi)$. k=2. Final time: 20. Time step:$\tau = 0.16h$.

Accuracy tests: nonlinear scalar problem

$$\partial_t u + \partial_x \left(\frac{u^2}{2}\right) = 0, \quad u(x,0) = \sin(x) \tag{1}$$

with periodic condition. We use local Lax-Friedrichs flux with $\alpha = 1$ and compute to $T = 0.2$.

k	N	L^1 error	order	L^2 error	order	L^∞ error	order
	20	2.3909E-02	-	1.2545E-02	-	1.3295E-02	-
	40	5.7654E-03	2.05	3.3145E-03	1.92	4.4218E-03	1.59
	80	1.3593E-03	2.08	7.8527E-04	2.08	1.0588E-03	2.06
1	160	3.3309E-04	2.03	1.9615E-04	2.00	2.7398E-04	1.95
	320	8.3021E-05	2.00	4.9307E-05	1.99	6.9491E-05	1.98
	640	2.0756E-05	2.00	1.2366E-05	2.00	1.7516E-05	1.99
	1280	5.1917E-06	2.00	3.0967E-06	2.00	4.3967E-06	1.99
	20	7.9715E-04	-	4.3367E-04	-	5.2666E-04	-
	40	1.1094E-04	2.85	6.1161E-05	2.83	8.5806E-05	2.62
	80	1.2956E-05	3.10	7.8273E-06	2.97	1.3620E-05	2.66
2	160	1.6167E-06	3.00	9.9359E-07	2.98	1.9932E-06	2.77
	320	2.0012E-07	3.01	1.2521E-07	2.99	2.6826E-07	2.89
	640	2.4636E-08	3.02	1.5706E-08	2.99	3.4294E-08	2.97
	1280	3.0516E-09	3.01	1.9667E-09	3.00	4.3225E-09	2.99

Table: Inviscid Burgers equation. $k = 1, 2$. $\Delta t = 0.1\Delta x$

Accuracy tests: 1D Euler equations

We solve the Euler equations on domain [0,2] with periodic boundary condition

$$\boldsymbol{u}_t + \boldsymbol{f}(\boldsymbol{u})_x = 0,$$

with $\boldsymbol{u} = (\rho, \rho v, E)^{\mathrm{T}}, \boldsymbol{f}(\boldsymbol{u}) = \left(\rho v, \rho v^2 + p, v(E+p)\right)^{\mathrm{T}}, E = \frac{p}{v} + \frac{1}{2}\rho v^2, p = (\gamma - 1)\left(E - \frac{1}{2}\rho v^2\right)$

Exact solution: $\rho(x,t) = 1 + 0.2 sin(\pi(x-t)), v(x,t) = 1, p(x,t) = 1$

Compute up to $t = 2$ with global Lax-Friedrichs flux

	cRKDG						SSP-RKDG					
N	L^1 error	order	L^2 error	order	L^∞ error	order	L^1 error	order	L^2 error	order	L^∞ error	order
20	2.3782e-03	-	2.0199e-03	-	2.6926e-03	-	2.4163E-03	-	2.0531E-03	-	2.7193e-03	-
40	5.6481e-04	2.07	4.8159e-04	2.07	6.5024e-04	2.05	5.6994E-04	2.08	4.8753E-04	2.07	6.5706e-04	2.05
80	1.3775e-04	2.04	1.1747e-04	2.04	1.5920e-04	2.03	1.3856E-04	2.04	1.1870E-04	2.04	1.6094e-04	2.03
160	3.3980e-05	2.02	2.8999e-05	2.02	3.9410e-05	2.01	3.4156E-05	2.02	2.92776E-05	2.02	3.9795e-05	2.02
320	8.4375e-06	2.01	7.2033e-06	2.01	9.8020e-06	2.00	8.4790e-06	2.01	7.2694E-06	2.01	9.8921e-06	2.01
640	2.1022e-06	2.00	1.7950e-06	2.00	2.4441e-06	2.00	2.1122E-06	2.01	1.8111E-06	2.00	2.4660e-06	2.00

Table: $k = 1$, cRKDG compared with SSP-RKDG.

Accuracy tests: 1D Euler equations

k	N	L^1 error	order	L^2 error	order	L^∞ error	order
2	20	5.7751e-05	-	5.1977e-05	-	1.4073e-04	-
	40	7.4092e-06	2.96	6.7552e-06	2.94	1.8496e-05	2.93
	80	9.3034e-07	2.99	8.5295e-07	2.99	2.3429e-06	2.98
	160	1.1628e-07	3.00	1.0686e-07	3.00	2.9363e-07	3.00
	320	1.4525e-08	3.00	1.3363e-08	3.00	3.6716e-08	3.00
	640	1.8147e-09	3.00	1.6704e-09	3.00	4.5899e-09	3.00
3	20	4.7353e-07	-	4.5971e-07	-	1.5495e-06	-
	40	2.9549e-08	4.00	2.8667e-08	4.00	9.4809e-08	4.03
	80	1.8427e-09	4.00	1.7887e-09	4.00	5.9125e-09	4.00
	160	1.1503e-10	4.00	1.1157e-10	4.00	3.6804e-10	4.01
	320	7.1980e-12	4.00	6.9853e-12	4.00	2.3080e-11	4.00
	640	4.6085e-13	3.97	4.4095e-13	3.99	1.4228e-12	4.02

Table: cRKDG for Euler equation. $k = 2, 3$. $\Delta t = 0.1\Delta x$

Accuracy test: clarification on schemes

Wrong implementation

- For some RK schemes, we can not directly applying the local operator $\nabla^{\text{loc}}\cdot$ at inner stages.
- Take two-stage ssp-RK for example

$$u_h^{(1)} = u_h^n - \Delta t \nabla^{\text{loc}} \cdot f\left(u_h^n\right)$$

$$u_h^{n+1} = \frac{1}{2} u_h^n + \frac{1}{2}\left(u_h^{(1)} - \Delta t \nabla^{\text{DG}} \cdot f\left(u_h^{(1)}\right)\right)$$

This can be made clear by substituting into

$$u_h^{n+1} = u_h^n - \Delta t \left(\frac{1}{2}\nabla^{\text{loc}} \cdot f\left(u_h^n\right) + \frac{1}{2}\nabla^{\text{DG}} \cdot f\left(u_h^{(1)}\right)\right).$$

k	N	L^1 error	order	L^2 error	order	L^∞ error	order
1	20	3.2417e-02	-	2.5518e-02	-	3.0467e-02	-
	40	1.6343e-02	0.99	1.2854e-02	0.99	1.4481e-02	1.07
	80	8.1932e-03	1.00	6.4425e-03	1.00	7.0456e-03	1.04
2	20	1.2196e-03	-	9.5949e-04	-	1.1384e-03	-
	40	3.2064e-04	1.93	2.5196e-04	1.93	2.7603e-04	2.04
	80	8.1344e-05	1.98	6.3894e-05	1.98	6.7135e-05	2.04

Table: wrong

Correct implementation

- Rewrite into one-step one-stage method
- Leave the $\nabla^{\text{DG}}\cdot$ in last stage unchanged
- Modified ssp-RK will work well

$$u_h^{(1)} = u_h^n - \Delta t \nabla^{\text{loc}} \cdot f\left(u_h^n\right)$$

$$u_h^{n+1} = u_h^n - \Delta t \left(\frac{1}{2}\nabla^{\text{DG}} \cdot f\left(u_h^n\right) + \frac{1}{2}\nabla^{\text{DG}} \cdot f\left(u_h^{(1)}\right)\right).$$

k	N	L^1 error	order	L^2 error	order	L^∞ error	order
1	20	2.4002e-03	-	2.0401e-03	-	2.6818e-03	-
	40	5.6836e-04	2.08	4.8480e-04	2.07	6.4887e-04	2.05
	80	1.3852e-04	2.04	1.1816e-04	2.04	1.5923e-04	2.03
2	20	5.7697e-05	-	5.2205e-05	-	1.4279e-04	-
	40	7.4069e-06	2.96	6.7983e-06	2.94	1.8810e-05	2.92
	80	9.3095e-07	2.99	8.5910e-07	2.98	2.3839e-06	2.98

Table: correct

Accuracy test: 2D linear advection problem

Consider the 2D linear advection equation

$$\partial_t u + c_x \partial_x u + c_y \partial_y u = 0, \quad \mathbf{x} \in \Omega$$

with exact conditions $u(x, y, t) = \sin(x - t) \sin(y - t)$.

We use upwind flux and periodic boundary condition

k	N	L^1 error	order	L^2 error	order	L^∞ error	order
1	10	9.0377e-03	-0.00	8.9181e-03	-0.00	2.4621e-02	-0.00
	20	2.1765e-03	2.05	2.2284e-03	2.00	6.8345e-03	1.85
	40	5.3246e-04	2.03	5.5630e-04	2.00	1.7717e-03	1.95
	80	1.3214e-04	2.01	1.3902e-04	2.00	4.5041e-04	1.98
	160	3.2939e-05	2.00	3.4751e-05	2.00	1.1335e-04	1.99
2	10	7.9812e-04	-0.00	6.7398e-04	-0.00	2.0400e-03	-0.00
	20	9.3131e-05	3.10	7.8860e-05	3.10	2.3290e-04	3.13
	40	1.1587e-05	3.01	9.8484e-06	3.00	2.9751e-05	2.97
	80	1.4401e-06	3.01	1.2302e-06	3.00	3.8047e-06	2.97
	160	1.7986e-07	3.00	1.5366e-07	3.00	4.7313e-07	3.01

Table: cRKDG

k	N	L^1 error	order	L^2 error	order	L^∞ error	order
1	10	9.3026e-03	-0.00	9.6950e-03	-0.00	2.7892e-02	-0.00
	20	2.1307e-03	2.13	2.4231e-03	2.00	7.6986e-03	1.86
	40	5.0305e-04	2.08	6.0486e-04	2.00	1.9906e-03	1.95
	80	1.2300e-04	2.03	1.5114e-04	2.00	5.0470e-04	1.98
	160	3.0550e-05	2.01	3.7780e-05	2.00	1.2702e-04	1.99
2	10	7.9994e-04	-0.00	7.0980e-04	-0.00	2.3246e-03	-0.00
	20	9.9862e-05	3.00	8.8034e-05	3.01	2.8666e-04	3.02
	40	1.2498e-05	3.00	1.1035e-05	3.00	3.6416e-05	2.98
	80	1.5630e-06	3.00	1.3810e-06	3.00	4.6023e-06	2.98
	160	1.9542e-07	3.00	1.7262e-07	3.00	5.7500e-07	3.00

Table: SSP-RKDG

Accuracy test: boundary error of 2D linear advection

For the same 2D linear advection equation, use the exact solution as the time dependent Dirichlet boundary condition

	N	L^1 error	order	L^2 error	order	L^∞ error	order
Butcher RKDG	10	3.1561e-05	-	3.0616e-05	-	1.1977e-04	-
	20	2.2211e-06	3.83	2.4337e-06	3.65	2.7118e-05	2.14
	40	1.6528e-07	3.75	2.8514e-07	3.09	6.6092e-06	2.04
	80	1.3237e-08	3.64	4.5845e-08	2.64	1.6418e-06	2.01
	160	1.1920e-09	3.47	7.9968e-09	2.52	4.0979e-07	2.00
cRKDG	10	2.8579e-05	-	2.6524e-05	-	7.4158e-05	-
	20	1.8721e-06	3.93	1.7295e-06	3.94	4.9173e-06	3.91
	40	1.1658e-07	4.01	1.0770e-07	4.01	3.0924e-07	3.99
	80	7.2898e-09	4.00	6.7326e-09	4.00	1.9284e-08	4.00
	160	4.5697e-10	4.00	4.2143e-10	4.00	1.2036e-09	4.00

Table: $k = 3$, fourth-order cRKDG and RKDG

Accuracy test: 2D Euler equations

We solve the following nonlinear system

$$\boldsymbol{u}_t + \boldsymbol{f}(\boldsymbol{u})_x + \boldsymbol{g}(\boldsymbol{u})_y = 0$$

$$\boldsymbol{u} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix}, \quad \boldsymbol{f}(\boldsymbol{u}) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ u(E+p) \end{pmatrix}, \quad \mathbf{g}(\boldsymbol{u}) = \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ v(E+p) \end{pmatrix}$$

The exact solution is
$\rho(x, y, t) = 1 + 0.2\sin(\pi(x + y - (u + v)t)), \quad u = 0.7, \quad v = 0.3, \quad p = 1$

We compute the solution up to $t = 2$ with periodic boundary conditions in both directions

Accuracy test: 2D Euler equations

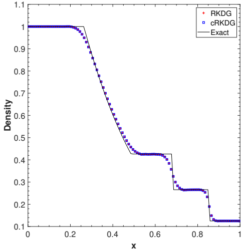
cRKDG							
k	N	L^1 error	order	L^2 error	order	L^∞ error	order
1	20	5.3856e-04	-	5.0705e-04	-	1.3182e-03	-
	40	1.2430e-04	2.12	1.1962e-04	2.08	3.4579e-04	1.93
	80	3.0324e-05	2.04	2.9458e-05	2.02	8.8657e-05	1.96
	160	7.5185e-06	2.01	7.3425e-06	2.00	2.2476e-05	1.98
2	20	4.7666e-05	-	4.8022e-05	-	1.7750e-04	-
	40	5.9380e-06	3.00	6.0850e-06	2.98	2.2876e-05	2.96
	80	7.3929e-07	3.01	7.6323e-07	3.00	2.8739e-06	2.99
	160	9.2235e-08	3.00	9.5466e-08	3.00	3.5939e-07	3.00
SSP-RKDG							
1	20	5.6122e-04	-	5.3013e-04	-	1.3776e-03	-
	40	1.2839e-04	2.13	1.2481e-04	2.09	3.6220e-04	1.93
	80	3.1160e-05	2.04	3.0708e-05	2.02	9.2904e-05	1.96
	160	7.6974e-06	2.02	7.6456e-06	2.01	2.3536e-05	1.98
2	20	4.8266e-05	-	4.8740e-05	-	1.8086e-04	-
	40	6.0219e-06	3.00	6.1806e-06	2.98	2.3315e-05	2.96
	80	7.5015e-07	3.00	7.7523e-07	3.00	2.9291e-06	2.99
	160	9.3631e-08	3.00	9.6981e-08	3.00	3.6633e-07	3.00

Table: cRKDG vs RKDG, $k = 1, 2$

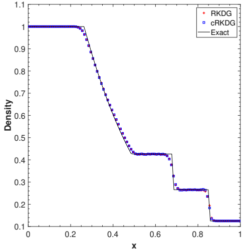
We solve one-dimensional nonlinear system of Euler equation with Sod problem, given by the initial condition as

$$\rho(x,0) = \begin{cases} 1.0, & x < 0.5 \\ 0.125, & x \geq 0.5, \end{cases} \qquad \rho u(x,0) = 0 \quad E(x,0) = \frac{1}{\gamma - 1} \begin{cases} 1, & x < 0.5 \\ 0.1, & x \geq 0.5 \end{cases}$$

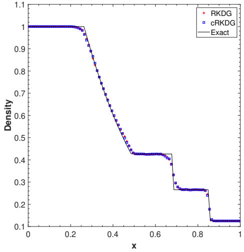
We compute to $T = 0.2$ with K=100 elements. We use WENO limiter with M=1



(a) k=1



(b) $k = 2$



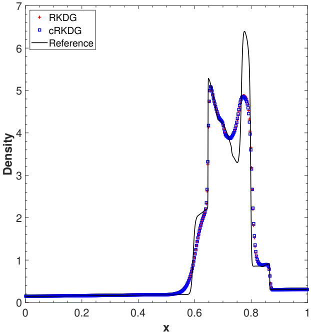
(c) $k = 3$

Test cases with shocks: blast wave

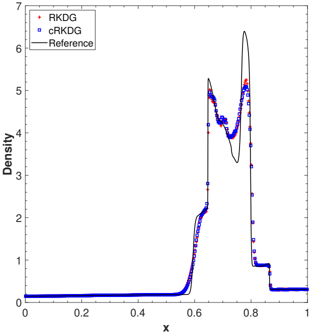
We consider the interaction of blast waves of Euler equation with the initial condition and compute to $t = 0.038$

$$
\begin{aligned}
(\rho, v, p) &= (1, 0, 1000) && \text{for } 0 \leqslant x < 0.1 \\
(\rho, v, p) &= (1, 0, 0.01) && \text{for } 0.1 \leqslant x < 0.9 \\
(\rho, v, p) &= (1, 0, 100) && \text{for } 0.9 \leqslant x
\end{aligned}
$$

We use local Lax-Friedrichs flux and WENO limiters with M=300



(a) $k = 1, N = 400$



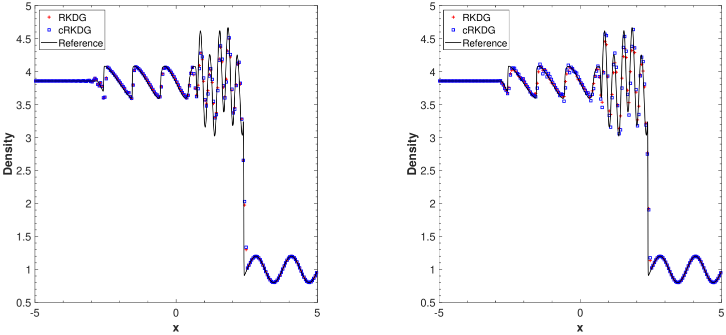
(b) $k = 2, N = 400$

Test cases with shocks: Shu-Osher problem

We consider the shock density wave interaction problem with the initial condition and compute to t = 1.8

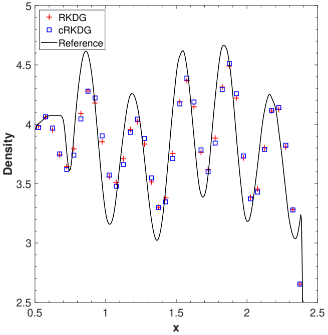
$$\begin{aligned}(\rho_L, v_L, p_L) &= (3.857143, 2.629369, 10.333333), && \text{when } x < -4,\\ (\rho_R, v_R, p_R) &= (1 + 0.2\sin(5x), 0, 1), && \text{when } x \geqslant -4.\end{aligned}$$

We use local Lax-Friedrichs flux and WENO limiters with M=300

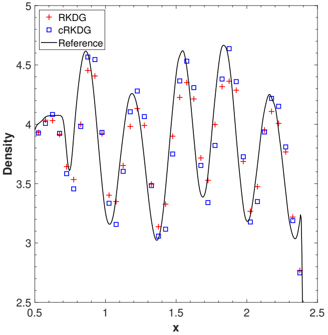


(a) $k = 1$, $N = 200$ (b) $k = 2$, $N = 200$

Test cases with shocks: Shu-Osher problem



(a) $k = 1, N = 200$



(b) $k = 2, N = 200$

- cRKDG has slightly better resolution, probably due to using limiter only once per time step.

Test cases with shocks: 2D Riemann problem

We solve Riemann problem of the 2D Euler equations with shocks on $[0,1] \times [0,1]$ and compute to t=0.2 The initial data consist of four constant states :

$$(\rho, u, v, P)(x, y, 0)$$

$$= \begin{cases} (1.1, 0, 0, 1.1), & x > 0.5, y > 0.5, \\ (0.5065, 0.8939, 0, 0.35), & x < 0.5, y > 0.5, \\ (1.1, 0.8939, 0.8939, 1.1), & x < 0.5, y < 0.5, \\ (0.5065, 0, 0.8939, 0.35), & x > 0.5, y < 0.5, \end{cases}$$

We use TVB limiter with M=50 and $\kappa = 1.5$

We plot density ρ at t=0.2 with 20 equally spaced contour lines from 0.6 to 1.9

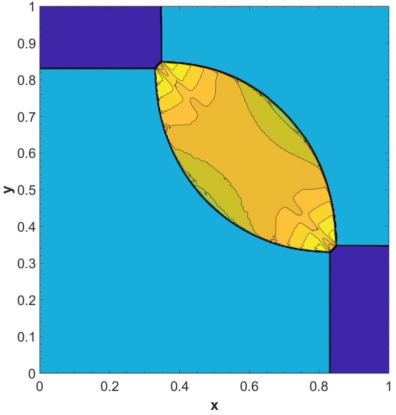


Figure: k=1, 400×400 uniform cells

Conclusions and future work

Future work & Acknowledgement

Conclusions:

- We develop a new class of RKDG methods. They feature improved compactness, local structure and communication efficiency

Future work:

- Oscillation control, implicit time marching, and parallel computing
- Theoretical framework for convergence, stability, and error analysis
- Application on convection-diffusion problems

Acknowledgement: Travel support for this presentation was provided by the SIAM-CSS 2022 local organizing committee

Thanks for your attention!

Reference I

Carpenter, M. H., Gottlieb, D., Abarbanel, S., and Don, W.-S. (1995).

The theoretical accuracy of runge–kutta time discretizations for the initial boundary value problem: a study of the boundary error.

SIAM Journal on Scientific Computing, 16(6):1241–1252.

Qiu, J., Dumbser, M., and Shu, C.-W. (2005).

The discontinuous galerkin method with lax–wendroff type time discretizations.

Computer methods in applied mechanics and engineering, 194(42-44):4528–4543.

Reference II

Shi, C. and Shu, C.-W. (2018).

On local conservation of numerical methods for conservation laws.

Computers & Fluids, 169:3–9.

Zhang, Q. (2011).

Third order explicit runge-kutta discontinuous galerkin method for linear conservation law with inflow boundary condition.

Journal of Scientific Computing, 46(2):294–313.