Journal of Computational Physics 231 (2012) 6622–6642

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

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

# Multi-moment ADER-Taylor methods for systems of conservation laws with source terms in one dimension

M.R. Norman [a,*], H. Finkel [b]

[a] Oak Ridge National Laboratory, P.O. Box 2008, MS6016, Oak Ridge, TN 37831, United States
[b] Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439, United States

ARTICLE INFO

*Article history:*
Received 15 December 2011
Received in revised form 29 March 2012
Accepted 4 May 2012
Available online 27 June 2012

*Keywords:*
Multi-moment
Hermite
WENO
ADER
Conservation laws
Fully-discrete

ABSTRACT

A new integration method combining the ADER time discretization with a multi-moment finite-volume framework is introduced. ADER runtime is reduced by performing only one Cauchy–Kowalewski (C–K) procedure per cell per time step and by using the Differential Transform Method for high-order derivatives. Three methods are implemented: (1) single-moment WENO (WENO), (2) two-moment Hermite WENO (HWENO), and (3) entirely local multi-moment (MM-Loc). MM-Loc evolves all moments, sharing the locality of Galerkin methods yet with a constant time step during *p*-refinement.

Five 1-D experiments validate the methods: (1) linear advection, (2) Burger's equation shock, (3) transient shallow-water (SW), (4) steady-state SW simulation, and (5) SW shock. WENO and HWENO methods showed expected polynomial *h*-refinement convergence and successfully limited oscillations for shock experiments. MM-Loc showed expected polynomial *h*-refinement and exponential *p*-refinement convergence for linear advection and showed sub-exponential (yet super-polynomial) convergence with *p*-refinement in the SW case.

HWENO accuracy was generally equal to or better than a five-moment MM-Loc scheme. MM-Loc was less accurate than RKDG at lower refinements, but with greater *h*- and *p*-convergence, RKDG accuracy is eventually surpassed. The ADER time integrator of MM-Loc also proved more accurate with *p*-refinement at a CFL of unity than a semi-discrete RK analog of MM-Loc. Being faster in serial and requiring less frequent inter-node communication than Galerkin methods, the ADER-based MM-Loc and HWENO schemes can be spatially refined and have the same runtime, making them a competitive option for further investigation.



## 1. Introduction

Numerical simulation has become a so-called third leg of science, and often it is in the form of numerically integrating a system of conservation laws. Crucial to the success of simulation is efficiently addressing the problem of interest on modern computers. Here, we restrict attention to the class of time-explicit integration methods for problems with low or limited stiffness such that the computational efficiency afforded by only local spatial dependence outweighs the time step relaxation afforded by time-implicit methods. Our immediate application of interest is that of a climate model dynamical core, the inviscid Euler equations for a stratified, dry fluid on a rotating sphere. However, we see no reason to restrict these methods to only that application or even to conservation laws. They apply to the more general class of time-dependent PDE systems as well.

* Corresponding author.
*E-mail addresses:* normanmr@ornl.gov (M.R. Norman), hfinkel@anl.gov (H. Finkel).

0021-9991/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2012.05.029

One design choice we make for efficient parallel integration is reducing the frequency and distance (hops between nodes) of inter-process parallel communication. Toward this, we adopt the ADER methodology [1–3] for time integration because it is fully-discrete (i.e., one-step, one-stage) and achieves high-order accuracy in time. This, compared to a multi-stage integrator, reduces the frequency of inter-process communication over a given simulation time, improving the computation-to-communication ratio and thus parallel efficiency. Also, for nearest-neighbor data dependence (neighboring cells, not necessarily compute nodes), we choose to implement multi-moment methods and Hermite WENO interpolation [4–6].

By "moment," we mean any temporally and spatially discrete piece of information describing the fluid state including the fluid state at a point in space and time, some time or space averaging of the state, any spatial or temporal derivative of the state (point-wise or averaged), or a set of spatial and/or temporal basis functions onto which the solution is projected. "Multi-moment" is intended to convey that multiple moments of the fluid state are stored in each cell and evolved directly by the scheme. The computational domains herein are assumed to be uniquely spanned by equal-sized "cells." The moments evolved by the schemes presented herein are cell-averaged spatial derivatives valid at a common point in time. Multi-moment simulation also improves accuracy because for smooth flows, $p$-refinement typically shows exponential convergence as compared to polynomial $h$-refinement convergence. Hermite methods use not only values of the fluid state but spatial derivatives as well to construct an interpolating function. Hermite WENO methods use the same WENO philosophy except that the candidate polynomials are constrained by both values and derivatives.

There already exist multi-moment ADER methods implemented as ADER-Galerkin methods [7]. We choose not to adopt these due to superlinear reduction in maximum stable time step (MSTS) with increasing moments. This reduction in MSTS is related to the clustering of quadrature grid points near cell edges, but it also depends on the temporal discretization and type of spatial quadrature used. We investigate an alternative finite-volume (FV) implementation of multiple moments using the ADER time discretization, which avoids this MSTS reduction and keeps a constant MSTS when increasing the number of moments (i.e., $p$-refinement). It will also be demonstrated that ADER time integration, which couples the terms of the PDE at a high temporal order of accuracy, exhibits accuracy superior to a multi-stage counterpart as the spatial order of the scheme increases via $p$-refinement and the CFL number is held at unity. We attribute this to the fact that temporal order of accuracy increases with $p$-refinement in ADER methods. Due to decreased communication frequency, we believe a multi-moment, FV ADER method may permit more $h$- and/or $p$-refinement than existing methods for a given simulation throughput due to enhanced parallel efficiency. This refinement may in turn lead to a more accurate solution for a given throughput.

The spectral element (SE) method, a continuous Galerkin method with only nearest neighbor data dependence (neighboring elements, not necessarily nodes), is the current state-of-the-art method for the Community Atmosphere Model (CAM) [8]. It is locally mass-conserving and makes feasible grid spacings never before reached for the required global climate simulation throughput of five Simulated Years Per wall clock Day (SYPD). The success of SE demonstrates that improving parallel efficiency can greatly improve simulation resolution by using present-day supercomputers more effectively. This is even despite efficiency losses due to reduction in MSTS inherent in all Galerkin methods for non-linear PDEs.

Another approach to time-explicit efficiency is the semi-Lagrangian method, particularly characteristics-based variants [9–11], which perform the semi-Lagrangian treatment on Riemann invariants allowing larger time steps than traditional FV methods. With characteristics-based semi-Lagrangian methods, the ceiling of the maximum CFL (Courant–Friedrichs–Lewy) number is the halo (or distance in number of cells) of information that must be communicated per-step. Therefore, there are trade-offs among communication volume, frequency, and locality. One difficulty with characteristics-based, semi-Lagrangian methods is that a genuinely multi-dimensional extension is complex to formulate and implement at high-order accuracy. Also, characteristics-based, semi-Lagrangian methods typically linearize the characteristics over a time step whereas ADER methods retain more of the non-linear interaction over a time step. Given the success of SE in climate modeling and the relative simplicity of an Eulerian, multi-moment approach in general, we investigate arbitrarily high-order, multi-moment FV methods using ADER techniques.

In Section 2, we introduce the fully-discrete FV framework and the ADER methodology as well as a modification to reduce the cost of ADER schemes which we call "ADER-Taylor". In Section 3, we introduce the fully-discrete FV evolution of higher-order spatial moments and the multi-moment ADER-Taylor method. In that section, we describe the three methods implemented in this study. Section 4 details the numerical experiments and provides results and discussion. Concluding remarks and future direction are given in Section 5.

## 2. ADER-Taylor methods

To accommodate a high-order-accurate (i.e. greater than second-order) and fully-discrete framework, the time evolution of state variables must be explicitly taken into account. This is most best accomplished by some translation of spatial information into temporal information because spatial information is readily available. For instance, characteristics-based methods (e.g., Roe-type schemes) diagonalize flux Jacobians to determine trajectories for characteristic variables (Riemann invariants). The trajectories translate from the spatial domain to the temporal. ADER methods also translate spatial information into temporal, but they do so in a more direct way and without the use of characteristic decompositions. To our knowledge, time-explicit ADER methods cannot be directly extended to CFL values larger than unity as can characteristics-base, semi-Lagrangian methods, but they are nonetheless fully-discrete. Before describing the ADER methodology, we give the fully-discrete FV framework in which we are working.

The CFL number is defined as $CFL = c\Delta t/\Delta x$ where $c$ is the fastest wave speed, $\Delta x$ is the grid spacing, and $\Delta t$ is the time step. Regarding Galerkin methods, $\Delta x$ is assumed to mean the element grid spacing and not the spacing between GLL quadrature points. While the notion of CFL number by itself has no relation to a particular scheme, the maximum stable CFL number (MSCFL) specifies the largest CFL value a given scheme can use without amplifying a harmonic input to a linear system over an arbitrary time step (by the von Neumann approach). In the context of a constant maximum wave speed and grid spacing, the CFL number acts as a normalization of the time step, and the MSCFL is a normalization of the MSTS.

### 2.1. Fully-discrete FV framework

Consider the following 1-D generic system of conservation laws with a source term:

$$\frac{\partial \mathbf{U}(x,t)}{\partial t} + \frac{\partial \mathbf{F}(\mathbf{U},x,t)}{\partial x} = \mathbf{S}(\mathbf{U},x,t)$$

where $\mathbf{U}$ is a vector of conserved variables, $\mathbf{F}$ is the vector of fluxes, and $\mathbf{S}$ is a vector of source terms. We integrate the equation set over a space–time domain $[x_i - \Delta x/2, x_i + \Delta x/2] \times [t_n, t_n + \Delta t]$ and apply the Gauss divergence theorem to obtain:

$$\overline{\mathbf{U}}_{i,n+1} = \overline{\mathbf{U}}_{i,n} - \frac{\Delta t}{\Delta x}\left(\widehat{\mathbf{F}}_{i+\frac{1}{2}} - \widehat{\mathbf{F}}_{i-\frac{1}{2}}\right) + \Delta t \widehat{\widehat{\mathbf{S}}}_i \tag{1}$$

where

$$\overline{\mathbf{U}}_{i,n} = \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} \mathbf{U}(\xi, t_n)\, d\xi, \tag{2}$$

$$\widehat{\mathbf{F}}_{i\pm\frac{1}{2}} = \frac{1}{\Delta t}\int_{t_n}^{t_n+\Delta t} \mathbf{F}(\mathbf{U}(x_{i\pm\frac{1}{2}}, \tau), x_{i\pm\frac{1}{2}}, \tau)\, d\tau, \tag{3}$$

$$\widehat{\widehat{\mathbf{S}}}_i = \frac{1}{\Delta x \Delta t}\int_{t_n}^{t_n+\Delta t}\int_{x_{i-1/2}}^{x_{i+1/2}} \mathbf{S}(\mathbf{U}(\xi,\tau), \xi, \tau)\, d\xi d\tau, \tag{4}$$

where $x_{i\pm\frac{1}{2}} = x_i \pm \Delta x/2$ and $t_{n+1} = t_n + \Delta t$. In practice, (2) is for definition and initialization purposes only and is not computed during the time stepping. Eqs. (3) and (4) form all that is needed in order to update $\overline{\mathbf{U}}_i$ for the next time step.

### 2.2. Cauchy–Kowalewski procedure for computing time derivatives

ADER methods use the definition of the PDE system itself via repeated differentiation to determine time derivatives, a process referred to in ADER literature as the Cauchy–Kowalewski (hereafter, C-K) procedure. The first time derivative of $\mathbf{U}$ is given in terms the first spatial derivative of $\mathbf{U}$ by the definition of the PDE itself, and the second-order space–time derivatives are computed by spatiotemporal differentiation:

$$\frac{\partial \mathbf{U}}{\partial t} = -\frac{\partial \mathbf{F}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial x} + \mathbf{S}$$

$$\frac{\partial^2 \mathbf{U}}{\partial x \partial t} = -\frac{\partial \mathbf{F}}{\partial \mathbf{U}}\frac{\partial^2 \mathbf{U}}{\partial x^2} - \frac{\partial^2 \mathbf{F}}{\partial \mathbf{U}^2}\left(\frac{\partial \mathbf{U}}{\partial x}\right)^2 + \frac{\partial \mathbf{S}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial x}$$

$$\frac{\partial^2 \mathbf{U}}{\partial t^2} = -\frac{\partial \mathbf{F}}{\partial \mathbf{U}}\frac{\partial^2 \mathbf{U}}{\partial x \partial t} - \frac{\partial^2 \mathbf{F}}{\partial \mathbf{U}^2}\frac{\partial \mathbf{U}}{\partial t}\frac{\partial \mathbf{U}}{\partial x} + \frac{\partial \mathbf{S}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial t}$$

Higher-order time derivatives are obtained by repeated differentiation with respect to space and time, and all derivatives of $\mathbf{U}$ are eventually functions of only spatial derivatives of $\mathbf{U}$ (by substitution). In practice, we do this component-by-component with symbolic mathematical software. In the end, we have time and mixed space–time derivatives defined at a single space–time point using only spatial derivatives at that point. The spatial derivatives are obtained from a reconstruction procedure, that is, a projection of the discrete moments onto a functional model to provide (usually continuous) variation of the fluid state within a cell or across a stencil of cells. Typically, the functional model is a polynomial, and the projection is a simple Vandermonde-type matrix inversion that can be pre-computed.

The total number of time and space–time derivatives needed by the Taylor polynomial for an $m$th-order accurate scheme is $[(m)(m+1)/2!]$ in 1-D. In 2-D, it would be $[(m)(m+1)(m+2)/3!]$. Although this is not a global memory requirement (because time derivatives need not be saved), this will likely be a source of memory pressure for accelerator devices such as Graphics Processing Units (GPUs). Depending upon the size of local memory stores on a given accelerator, there will be a point of diminishing returns as the number of moments is increased. The expense of the C-K procedure is known to grow exponentially for a generic PDE system because of the expansion of derivative product and chain rules [12]. Therefore, the ADER scheme is typically very expensive at high-order accuracy for non-linear problems.

### 2.3. Computing space–time derivatives with differential transforms

To reduce the expense of computing time derivatives in the C-K procedure for higher spatial dimensions and more complex flux and source functions, we use a technique based on transforming the differential equation into a recurrence relation for the coefficients of its power-series solution. This technique is well known to the creators of automatic (or algorithmic) differentiation software [13], and recent literature often refers to it as the Differential Transform Method (DTM) [14]. The differential transforms (DTs) exhibit polynomial complexity and make evaluating the C-K procedure concise and efficient. The DT of a function $f(x,t)$ is given as [15]

$$F(k,h) = \frac{1}{k!h!}\frac{\partial^{k+h} f(x,t)}{\partial x^k \partial t^h}\bigg|_{x=x^*,t=t^*},$$

and the inverse is a straightforward polynomial (Taylor series):

$$f(x,t) = \sum_{k=0}^{\infty}\sum_{h=0}^{\infty} F(k,h)(x-x^*)^k (t-t^*)^h.$$

These transforms have been applied to most common arithmetic operators and functions by various authors. The transforms we used in this study are in Table 1. A comprehensive list is available in [13].

Using the rules from Table 1, we transform the time–space PDEs into recurrence relations that relate a given spatiotemporal derivative to lower-order derivatives. In this manner, high-order space–time derivatives are obtained in polynomial complexity (a function of the number of spatial moments) from at full set of space-only derivatives.

We found that for Burger's equation, the DTM gave comparable runtime results as direct output from symbolic mathematical software. For the shallow-water (SW) model, however, the total output from symbolic mathematical software for the C-K procedure at eighth-order accuracy was thousands of lines long in Fortran which we consider unacceptable for sustainable software development practices. In addition, even modern compilers have trouble performing common subexpression elimination and other optimizations on such verbose expressions. In this case, the direct output was also over five times slower than using DTs. We believe the large expression size of direct output from symbolic mathematical software and high computational expense (especially when performed at every quadrature point in space) is a major reason that most ADER studies have not proceeded beyond fifth- or sixth-order accuracy for non-linear PDEs. Using DTs, one can proceed to any order of accuracy without much coding effort, and computational expense is decreased.

The 1-D SW system (with terrain) stated in terms of components of the state variable vector is:

$$\frac{\partial}{\partial t}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} u_2 \\ \frac{u_2^2}{u_1} + \frac{u_1^2}{2} \end{bmatrix} = \begin{bmatrix} 0 \\ -u_1 \frac{\partial \phi_B}{\partial x} \end{bmatrix}$$

where $u_1 = \phi$ is the geopotential thickness, $u_2 = \phi u$ is "momentum" (with $u$ being wind), and $\phi_B$ is geopotential of bottom orography. These terms are explained more in Section 4.1. Its DT is given by:

$$U_1(k,h+1) = -\frac{k+1}{h+1} U_2(k+1,h)$$

$$U_2(k,h+1) = -\frac{k+1}{h+1} G_1(k+1,h) - \frac{k+1}{h+1} G_2(k+1,h) - \frac{1}{h+1} G_3(k,h)$$

where the auxiliary functions are given by:

$$G_1(k,h) = \sum_{r=0}^{k}\sum_{s=0}^{h} G_{1,a}(r,s) G_{1,b}(k-r,h-s)$$

$$G_2(k,h) = \frac{1}{2}\sum_{r=0}^{k}\sum_{s=0}^{h} U_1(r,s) U_1(k-r,h-s)$$

**Table 1**
Differential transforms of the functions encountered in this study. $c$ is a constant in space and time. $U(0,0)$ is the function $u(x,t)$ itself, and the terms causing $F$ to depend on itself in the last transform should be omitted during summation.

| Rule | Original function | Differential transform |
|---|---|---|
| 1 | $f(x,t) = cu(x,t)$ | $F(k,h) = cU(k,h)$ |
| 2 | $f(x,t) = \frac{\partial^{i+j} u(x,t)}{\partial x^i \partial t^j}$ | $F(k,h) = \frac{(k+i)!}{k!}\frac{(h+j)!}{h!} U(k+i,h+j)$ |
| 3 | $f(x,t) = u(x,t)v(x,t)$ | $F(k,h) = \sum_{r=0}^{k}\sum_{s=0}^{h} U(r,s)V(k-r,h-s)$ |
| 4 | $f(x,t) = \frac{1}{u(x,t)}$ | $F(k,h) = -\frac{1}{U(0,0)}\sum_{r=0}^{k}\sum_{s=0}^{h} U(r,s)F(k-r,h-s)$ |

$$G_3(k,h) = \sum_{r=0}^{k} U_1(r,h) G_{3,a}(k-r)$$

$$G_{1,a}(k,h) = \sum_{r=0}^{k}\sum_{s=0}^{h} U_2(r,s) U_2(k-r,h-s)$$

$$G_{1,b}(k,h) = -\frac{1}{U_1(0,0)}\sum_{r=0}^{k}\sum_{s=0}^{h} U_1(r,s) G_{1,b}(k-r,h-s)$$

$$G_{3,a}(k) = (k+1)\Phi_B(k+1)$$

Note that this gives a method of obtaining the time derivatives by stepping forward in $h$ starting from $h = 0$, the spatial derivatives which are either interpolated or explicitly evolved or a combination (e.g. Hermite methods). For efficiency, $G_{1,a}$, $G_{1,b}$, and $G_{3,a}$ must be cached at each value of $k$ and $h$ so as not to be recomputed. The computational complexity grows with $p^4$ if $p$ is the order of accuracy of the scheme. For the second equation, the terms involving $G_1$ and $G_2$ give the DT for the flux, and the term involving $G_3$ gives the DT for the source term. DTs of the terrain geopotential, $\phi_B$, are precomputed because they are constant in time. The DT for the flux term in Burger's equation is given by $G_2$. The DT of uniform linear advection follows directly from Rule 1 of Table 1. Implementation details can be cumbersome in performing these summations. For this reason, we include our Fortran 90 implementation for the SW case in Fig. 1.

### 2.4. State-expansion ADER scheme

How one utilizes the time derivatives from the C-K procedure varies. Here, we describe the state-expansion ADER approach for computing fluxes in 1-D. First, a reconstruction is sampled to obtain values and derivatives of the state vector at left- and right-hand limits of each interface valid at the beginning of the time step:

$$\mathbf{U}^{(k),\pm}_{i-\frac{1}{2}}(x_{i-\frac{1}{2}}, t_n), \quad k = 0, 1, \ldots, m-1$$

where $\mathbf{U}^{(k)} = \partial^k \mathbf{U}/\partial x^k$ and $m$ is the spatial order of accuracy of the scheme. From these, a non-linear Riemann solver is typically used to compute the interface state of $\mathbf{U}$ : $\mathbf{U}^{(0)}_{i-\frac{1}{2}} = f_{Riemann}\left(\mathbf{U}^{(0),-}_{i-\frac{1}{2}}, \mathbf{U}^{(0),+}_{i-\frac{1}{2}}\right)$. For the Godunov state of derivatives of $\mathbf{U}$, the system is typically linearized to compute: $\mathbf{U}^{(k)}_{i-\frac{1}{2}} = \hat{f}_{Linear}\left(\mathbf{U}^{(k),-}_{i-\frac{1}{2}}, \mathbf{U}^{(k),+}_{i-\frac{1}{2}}\right)$ for $k > 0$. Finally, the C-K procedure is used to compute time derivatives of $\mathbf{U}$ from the spatial derivatives via the PDE definition. This gives a temporal Taylor polynomial $\widetilde{\mathbf{U}}_{i-\frac{1}{2}}(t)$ at each interface, which is in turn used to compute the integrand of (3):

$$\mathbf{F}(x_{i-\frac{1}{2}}, \tau) = \mathbf{F}(\widetilde{\mathbf{U}}_{i-\frac{1}{2}}(\tau)).$$

For source term quadrature points within the body of a cell, the Godunov state does not need to be computed. Also, the temporal Taylor series, $\widetilde{\mathbf{U}}(x^*, t)$ (where $x^*$ is an arbitrary spatial quadrature sampling point), is formed by a local C-K procedure and used to compute source terms in time.

The ADER scheme becomes expensive for higher-order temporal accuracy because the cost of each C-K procedure grows exponentially with the order of accuracy for a general non-linear PDE system. Also, in the traditional ADER scheme, the C-K procedure is performed at each spatial quadrature point. Given a problem in $d$ spatial dimensions, the number of quadrature points to compute fluxes and source terms grows as $n^{d-1}$ and $n^d$, respectively.

It is the compounding effect of performing the C-K procedure at all quadrature points that we are removing in the present modification to reduce the cost of the ADER scheme. Instead of computing time derivatives at each quadrature point, we compute them only at the cell center and form a space–time Taylor polynomial over the cell and time step domain. This polynomial is then sampled at each space–time quadrature point. The reduction in overall cost requires that sampling a Taylor polynomial be cheaper than performing the C-K procedure. This would likely be only true for non-linear systems because for linear systems, the C-K procedure is largely pre-computed and is much less expensive. The savings associated with the single space–time Taylor expansion will be smaller in 1-D with simple problems, but in 2-D with more complex non-linear fluxes and sources, the savings are expected to be significantly greater.

### 2.5. How ADER-Taylor differs from traditional ADER

We first introduce the ADER-Taylor scheme by describing how it differs from the traditional state-expansion ADER scheme. We propose two modifications. First, the C-K procedure is only computed once at the cell center, $x_i$, to obtain time and mixed space–time derivatives, which are then used to construct a space–time Taylor polynomial, $\mathbf{U}_{Tay}(x,t)$. The space–time Taylor polynomial is then sampled at quadrature points. Because of determining $\mathbf{U}_{Tay}(x,t)$ *before* computing Godunov states at the interface, the interface fluxes must be computed in a different manner than with traditional ADER methods.

```
subroutine cauchy_kowalewski( nmom , a , ders , i )
  integer          , intent(in ) :: nmom       !# moments
  double precision , intent(in ) :: a(0:nmom-1,2) !fluid state
                                    !Dims = ( moment , variable )
  double precision , intent(out) :: ders(0:nmom-1,0:nmom-1,2) !space-time derivs
                                    !Dims = ( sp order , tm order , variable )
  integer          , intent(in ) :: i          !Cell index
  double precision :: g1a(0:nmom-1,0:nmom-1) !Cache DT values
  double precision :: g1b(0:nmom-1,0:nmom-1) !Cache DT values
  double precision :: g3a(0:nmom-1)          !Cache DT values
  double precision :: g1, g2, g3             !Final auxiliary DTs
  double precision :: terder(0:nmom)         !Terrain DTs
  double precision :: tot_g1a, tot_g1b       !For summations
  integer :: k,h,s,r                         !iterators
  g1b = 0. !g1b will reference itself at uninitialized values.
           !Must initialize to 0 to remove those terms from summation.
  terder = terder_prec(0:nmom,i) !Load pre-computed terrain derivs
  !diff_trans_x0 computes space-only DTs from evolved moments valid
  !at cell center. Store into ders at h=0 to initialize the C-K procedure
  ders(0:nmom-1,0,1) = diff_trans_x0( a(:,1) )
  ders(0:nmom-1,0,2) = diff_trans_x0( a(:,2) )
  !Pre-compute DTs of terrain. terder and factorial are pre-computed globals
  do k = 0 , nmom-2
    g3a(k) = (k+1)*terder(k+1) / factorial(k+1)
  enddo
  g1b(0,0) = 1./ders(0,0,1) !Initialize 1st value of g1b before summation
  do h = 0 , nmom-2  !We're storing to h+1, so only want to go up to nmom-1
    !Compute g1a and g1b for all needed k values for time level h
    do k = 0 , nmom-h-1
      tot_g1a = 0.
      tot_g1b = 0.
      do s = 0 , h
        do r = 0 , k
          tot_g1a = tot_g1a + ders(r,s,2)*ders(k-r,h-s,2)
          tot_g1b = tot_g1b + ders(r,s,1)*g1b (k-r,h-s  )
        enddo
      enddo
      g1a(k,h) =  tot_g1a
      if ( .not. (h==0 .and. k==0) ) g1b(k,h) = -tot_g1b / ders(0,0,1)
    enddo
    !Now compute g1, g2, and g3
    do k = 0 , nmom-h-2
      g1 = 0.
      g2 = 0.
      do s = 0 , h
        do r = 0 , k+1
          g1 = g1 + g1a (r,s  )*g1b (k+1-r,h-s  )
          g2 = g2 + ders(r,s,1)*ders(k+1-r,h-s,1)
        enddo
      enddo
      g3 = 0.
      do r = 0 , k
        g3 = g3 + ders(r,h,1)*g3a(k-r)
      enddo
      g2 = g2*0.5
      ders(k,h+1,1) = -ders(k+1,h,2) * (k+1) / (h+1)
      ders(k,h+1,2) = -g1 * (k+1) / (h+1) - g2 * (k+1) / (h+1) - g3 / (h+1)
    enddo
  enddo
end subroutine cauchy_kowalewski
```

**Fig. 1.** Fortran 90 subroutine computing arbitrarily high-order space–time derivatives of state variables for the 1-D SW system using DTs.

Therefore, the second modification is the handling of fluxes. Any flux function can be used for this purpose, but the flux function must be performed at each quadrature point in time. In fact, this also decreases the number of Riemann solves which must be performed because for Gauss–Legendre quadrature, the number of quadrature points is lower than the number of moments. The Local Lax Friedrichs (LLF) flux function would likely be a good choice in multiple dimensions in which

the number of quadrature points grows dramatically with scheme order. In 1-D, this would limit the CFL number to 1/2 instead of unity, but in 2-D or greater, the ADER scheme is limited to the linear multi-dimensional CFL limit of $1/d$ where $d$ is the number of dimensions. Therefore, the LLF flux would not limit the time step in multiple dimensions. Because this study is in 1-D, and we wish to show that the MM-ADER scheme is stable to a CFL of unity, we use the linear Godunov (therefore, upwind) state to keep the schemes stable at a CFL number of unity. The linear Godunov state is determined by locally freezing the flux Jacobian for diagonalization using the fluid state interface average.

### 2.6. Description of the ADER-Taylor scheme

We start with the fully-discrete FV discretization (1). To update the cell-averaged state variable vector $\overline{\mathbf{U}}_i$, we need the fluxes and source term (3) and (4). As mentioned earlier, we use Gauss–Legendre quadrature for the purposes of integration in both space and time. Terms (3) and (4) now take the form

$$\widehat{\mathbf{F}}_{i\pm1/2} = \sum_{l=1}^{n_G} \omega_l' \widetilde{\mathbf{F}}(\mathbf{U}_L(x_{i\pm\frac{1}{2}}, t_n + \kappa_l' \Delta t), \mathbf{U}_R(x_{i\pm\frac{1}{2}}, t_n + \kappa_l' \Delta t)) \tag{5}$$

$$\widehat{\overline{\mathbf{S}}}_i = \sum_{l=1}^{n_G} \sum_{k=1}^{n_G} \omega_k \omega_l' \mathbf{S}(\mathbf{U}(x_{i-\frac{1}{2}} + \kappa_k \Delta x, t_n + \kappa_l' \Delta t), x_{i-\frac{1}{2}} + \kappa_k \Delta x, t_n + \kappa_l' \Delta t) \tag{6}$$

where $n_G$ is the number of quadrature points, $\kappa$ and $\kappa'$ are integration points scaled to the domain $[0,1]$ for space and time (respectively), $\omega$ and $\omega'$ are convex integration weights for space and time (respectively), $\mathbf{U}_L$ and $\mathbf{U}_R$ denote limits approaching an interface from the negative and positive direction (respectively), and $\widetilde{\mathbf{F}}$ is a flux function for the flux terms (linear Godunov state in our case). If the reconstruction is $m$th-order accurate in time and space, only $\lceil m/2 \rceil$ Gauss–Legendre quadrature points need to be used. This is in contrast to a Discontinuous Galerkin scheme which typically requires as many quadrature points as there are moments (or polynomial basis functions) for the body integral.

We performed a von Neumann stability analysis on this formulation using the linear advection equation $u_t + u_x = 0$ with Vandermonde polynomials up to 25th-order accuracy, and the scheme is consistently stable up to a CFL number of unity. Because the scheme is stable with non-limited polynomial interpolants, it will also be stable with the WENO interpolants.

#### 2.6.1. Time–space Taylor expansion of $\mathbf{U}$

From here, the definition of $\mathbf{U}(x,t)$ within the cell will close the scheme. We assume a reconstruction of arbitrary order and definition, subject to the constraint that it render well-bounded values for $\mathbf{U}$ and up to $n$th-order derivatives of $\mathbf{U}$ at the cell center. For accuracy purposes, we also constrain that the $k$th-order derivative when sampled at the cell center be accurate to a truncation of $\Delta x^{n+1-k}$. Then, the C-K procedure provides up to $n$th-order time and mixed space–time derivatives. We now have the following information valid at the cell center:

$$\left\{ \left. \frac{\partial^{k+l}\mathbf{U}}{\partial x^k \partial t^l} \right|_{x=x_i, t=t_n} \right\} : \quad 0 \leqslant k; \;\; 0 \leqslant l; \;\; k+l \leqslant n$$

Using this information, we construct a space–time Taylor polynomial that is $(n+1)$th-order accurate in space and time. The Taylor polynomial is sampled at integration points to provide the information necessary to compute (5) and (6). Sampling the space–time Taylor series actually becomes quite expensive because of the quadratic growth of space–time terms in 1-D. Further increasing the expense in the multi-moment case is that we have to sample all derivatives from the space–time Taylor series. The Taylor polynomials are sampled on a rectangular space–time grid to compute fluxes and source terms at quadrature points. Because of this, we can reduce the polynomial sampling expense by splitting the Taylor evaluations into spatial and temporal computations:

$$\mathbf{U}(x^*, t^*) = \sum_k \sum_l \frac{1}{k!l!} \frac{\partial^{k+l}\mathbf{U}}{\partial x^k \partial t^l} (x^* - x_i)^k (t^* - t_n)^l = \sum_l \left( \sum_k a_{k,l} (x^* - x_i)^k \right)$$

Looping in this manner, the coefficients $a_{k,l}$ can be computed at the beginning of each temporal quadrature point and remain constant until the next temporal quadrature point. Furthermore, since the expression in the innermost summation is in the form of a differential transform, we can use values directly obtained from the DT procedure for obtaining space–time derivatives. Finally, to efficiently compute all spatial derivatives at a given point from the coefficients $a_{k,l}$, we use a modification of [16] given as Fortran 90 code in Fig. 2. The switch (`multFac`) to keep factorials out of the derivative expressions (essentially keeping them in the form of DTs) is useful when these samples feed directly into a routine computing flux derivatives via DTs.

#### 2.6.2. Computational aspects of ADER-Taylor

We would like to review the competitive computational aspects of this scheme. The scheme is fully-discrete, meaning there are no stages involving inter-node parallel communication within a time step, which keeps communication overhead

```
function derivs( n , x0 , a , multFac ) result(r)
  implicit none
  integer, intent(in) :: n         !Order of polynomial
  real   , intent(in) :: a(0:n)    !Coefficients (a(0) is constant)
  real   , intent(in) :: x0        !Sampling point
  logical, intent(in) :: multFac   !if .true., return derivatives
                                   !if .false., return differential transform
  real                :: r(0:n)
  integer :: i, j
  r(0:n) = a(n)
  do j = 1 , n
    r(0) = r(0) * x0 + a(n-j)
    do i = 1 , n-j
      r(i) = r(i) * x0 + r(i-1)
    enddo
  enddo
  if (multFac) r(2:n) = r(2:n) * fac(2:n)
end function derivs
```

**Fig. 2.** Fortran 90 code for function derivatives to compute all derivatives at point $x_0$ of a polynomial defined by: $p(x) = \sum_{i=0}^{n} a_i x^i$. The array $fac(i)$ is the factorial of the integer $i$. The switch *multFac* allows the user to keep factorials out of the result so that the derivatives are differential transforms.

low. Second, the ADER approach fully couples multiple spatial dimensions (for the future multi-dimensional extension) and source terms to reduce the parallel communication overhead even further compared to operator splitting methods for these components. Third, the scheme runs stably up to a CFL number of unity regardless of the order of accuracy in space or time or how many moments are evolved per cell. This is in contrast to Galerkin methods whose MSTS reduces superlinearly as more moments are simulated per cell. Finally, the modification to reduce the number of C-K computations via a Taylor series also gives a boost in efficiency by reducing the cost of the scheme.

For multi-moment FV ADER methods, the MSTS remains constant during $p$-refinement and decreases linearly with $h$-refinement. Therefore, it is better regarding the time step to perform $p$-refinement for the MM-ADER approach whereas it is worse for explicit Galerkin methods.

## 3. Multi-moment ADER-Taylor methods

The advantage of using multiple moments in a FV framework is that the MSTS does not depend on the number of moments or the spatial order of accuracy, giving significantly larger time steps than explicit Galerkin methods for non-linear systems of conservation laws. For the multi-moment ADER scheme, we need a framework for evolving higher-order moments (derivatives of $\mathbf{U}$). For this, we choose to use the same FV approach used for $\mathbf{U}$ itself.

We spatially differentiate the PDE to give:

$$\frac{\partial}{\partial t}\left(\frac{\partial^m \mathbf{U}}{\partial x^m}\right) + \frac{\partial}{\partial x}\left(\frac{\partial^m \mathbf{F}}{\partial x^m}\right) = \frac{\partial^m \mathbf{S}}{\partial x^m}$$

Integrating over a space–time domain, applying the divergence theorem, and computing integrals via quadrature give the same FV framework used for $\mathbf{U}$ itself. Now, we have new flux and source terms:

$$\frac{\partial^m \mathbf{F}}{\partial x^m} \equiv \mathbf{F}^{(m)}(\{\mathbf{U}^{(k)}\}_{k=0\ldots m})$$

$$\frac{\partial^m \mathbf{S}}{\partial x^m} \equiv \mathbf{S}^{(m)}(\{\mathbf{U}^{(k)}\}_{k=0\ldots m}, x, t)$$

to provide a FV framework for derivatives of $\mathbf{U}$ of the form:

$$\overline{\mathbf{U}}^{(m)}_{i,n+1} = \overline{\mathbf{U}}^{(m)}_{i,n} - \Delta t\left(\frac{\widehat{\mathbf{F}}^{(m)}_{i+\frac{1}{2}} - \widehat{\mathbf{F}}^{(m)}_{i-\frac{1}{2}}}{\Delta x} - \widehat{\mathbf{S}}^{(m)}_{i}\right)$$

For the new flux terms (spatial derivatives of the original flux terms), we use differential transforms based on the known derivatives of $\mathbf{U}$.

From here, derivatives of $\mathbf{U}_{Tay}(x,t)$ feed into the higher-moment flux and source terms to complete the scheme. To show the range of options within the multi-moment ADER-Taylor framework, we implement three classes of methods:

1. Fifth-order accurate single-moment ADER-Taylor method using the WENO reconstruction of [17] (abbreviated as "WENO")

2. Fifth-order accurate two-moment ADER-Taylor method using a Hermite WENO reconstruction (modified from [4]) (abbreviated as "HWENO")
3. Entirely local multi-moment ADER-Taylor methods which explicitly evolve all moments (abbreviated as "MM-Loc")

The MM-Loc scheme is implemented at up to 16th-order accuracy in space and time for all experiments in order to determine the nature of error reduction with increasing moments. We do not necessarily believe that a 16th-order accurate method will be the most useful but want to show the properties of these methods over a large range of $p$-refinement.

The MM-Loc approach is similar in nature to a modal expansion such as the modal Discontinuous Galerkin method. Because of this, we believe that relatively minor modifications to the numerical machinery developed to limit modes in a DG scheme (e.g. [18,6,5,19]) should also sufficiently limit the MM-Loc scheme. Moreover, we also see no reason why the hybrid RKDG+HWENO philosophy should not also work here as well.

### 3.1. Multi-moment ADER-Taylor Hermite WENO method (HWENO)

For this method, we evolve the cell mean value $\overline{\mathbf{U}}_i$ and cell mean derivative $\overline{\mathbf{U}}_i^{(1)}$ and use them to reconstruct a fifth-order accurate limited polynomial using a 3-cell stencil (nearest neighbor spatial dependence). WENO accuracy ranges from low- to high-order depending upon the smoothness of the flow.

In general, two types of WENO procedures appear in literature. Type 1 (e.g., [20]) is for situations where the WENO interpolant is sampled at points known beforehand. These methods pre-compute optimal weights such that the weighted combination of the low-ordered interpolants renders the higher-order interpolant value at one specific point. In the time stepping the higher-order interpolant is never actually formed because the optimal weighting of lower-order values renders the higher-order value at a point. These weights are a function of the grid geometry and point location only (not of the data itself), and they are altered from optimal values based on the smoothness of the flow such that the least oscillatory stencils are weighted the most.

Type 2 (e.g., [17]) is for situations in which either the interpolation points are not known beforehand or a continuous limited function is needed perhaps for analytical integration over an interval or, in our case, to have access to all spatial derivatives to high-order accuracy. Type 2 WENO methods explicitly form the optimal polynomial as well as the lower-ordered polynomials during the time stepping. A set of weights are chosen by the user. Then what we term a "bridge" polynomial is formed such that the weighted combination of the lower-ordered polynomials with the bridge polynomial renders the optimal polynomial. The weights chosen by the user (an added parameter range to control smoothness) are then altered based on flow smoothness to limit the interpolant. Type 2 WENO methods render interpolants that are continuous and limited across the intersection of all candidate polynomial stencils.

In the ADER-Taylor approach, in order to have an $m$th-order accurate method in space and time, we need up to $(m-1)$th-order spatial derivatives valid at the cell center. Using the Type 1 WENO method, there is no interpolant formed of high enough order to obtain these derivatives because the optimal polynomial is never formed. Thus, it is simplest to use the Type 2 method wherein the optimal polynomial is formed, and all needed derivatives are immediately available. This is why we chose to use the WENO procedure of [17] for our WENO methods. We did formulate a method based on Type 1 WENO interpolation, sampling at multiple points within a cell, and using those points to constrain a high-order limited polynomial, and it was successful. However, we found using a Type 2 method to be far more flexible in controlling smoothness versus accuracy, and thus, we obtained greater accuracy. Also, the Type 2 approach was computationally cheaper because we did not have to sample at five points and reconstruct a polynomial from them.

For the HWENO procedure, we do not use the method of [21] because though it successfully implements a non-oscillatory HWENO method, the reconstruction can reduce to first-order accuracy at discontinuities in some cases. As mentioned in that study, this is not a problem for shocks which are reinforced. However, other discontinuities, which are not necessarily reinforced, can become excessively diffused.

Therefore, we alter the method of [21] by ridding the center cell's first derivative from all reconstructions. This reduces the optimal and sub-optimal stencils by one order of accuracy, meaning this interpolation will range from third- to fifth-order accuracy. The four polynomials used in this HWENO procedure obey the following constraints:

$$\frac{1}{\Delta x}\int_{x_k-\frac{\Delta x}{2}}^{x_k+\frac{\Delta x}{2}} P_{Opt,i}(x)dx = \overline{\mathbf{U}}_k: \quad k = i-1, i, i+1$$

$$\frac{1}{\Delta x}\int_{x_k-\frac{\Delta x}{2}}^{x_k+\frac{\Delta x}{2}} \frac{\partial}{\partial x}P_{Opt,i}(x)dx = \overline{\mathbf{U}}_k^{(1)}: \quad k = i-1, i+1$$

$$\frac{1}{\Delta x}\int_{x_k-\frac{\Delta x}{2}}^{x_k+\frac{\Delta x}{2}} P_{L,i}(x)dx = \overline{\mathbf{U}}_k: \quad k = i-1, i$$

$$\frac{1}{\Delta x}\int_{x_{i-1}-\frac{\Delta x}{2}}^{x_{i-1}+\frac{\Delta x}{2}}\frac{\partial}{\partial x}P_{L,i}(x)dx=\overline{\mathbf{U}}_{i-1}^{(1)}$$

$$\frac{1}{\Delta x}\int_{x_k-\frac{\Delta x}{2}}^{x_k+\frac{\Delta x}{2}}P_{C,i}(x)dx=\overline{\mathbf{U}}_k:\quad k=i-1,i,i+1$$

$$\frac{1}{\Delta x}\int_{x_k-\frac{\Delta x}{2}}^{x_k+\frac{\Delta x}{2}}P_{R,i}(x)dx=\overline{\mathbf{U}}_k:\quad k=i,i+1$$

$$\frac{1}{\Delta x}\int_{x_{i+1}-\frac{\Delta x}{2}}^{x_{i+1}+\frac{\Delta x}{2}}\frac{\partial}{\partial x}P_{R,i}(x)dx=\overline{\mathbf{U}}_{i+1}^{(1)}$$

The "bridge" polynomial is formed by $P_{Brg,i}(x)=(P_{Opt,i}(x)-\sum_k\omega_k P_{k,i}(x))/\omega_{Brg}$ where $k\in\{L,C,R\}$. We choose the weights such that the bridge polynomial carries 16 times the weight of the lower-ordered stencils: $\{\omega_{L,i},\omega_{C,i},\omega_{R,i},\omega_{Brg,i}\}=\{1,1,1,16\}/19$. Our process for determining these weights was to increase the weight of the bridge polynomial by powers of 2 until accuracy no longer increased significantly. Then numerous simulations with the shock experiments were performed over long periods of time and a variety of CFL values and refinements to ensure the scheme was reliably limiting all oscillations. Users can tune these parameters to the needs of their applications. For instance, if strong shocks in the vicinity of near-zero air densities were being simulated, one would use a lower optimal weight for the bridge polynomial and possibly a larger exponent to the Total Variation (TV) values described in the next paragraph.

We compute the TV across all derivatives of each candidate stencil over the center cell:

$$TV_{k,i}=\frac{1}{\Delta x}\int_{x_i-\frac{\Delta x}{2}}^{x_i+\frac{\Delta x}{2}}\sum_j\left(\frac{\partial^j}{\partial x^j}P_{k,i}(x)\right)^2\Delta x^{2j}dx:\quad k\in\{L,C,R,Brg\};\ j\in\{1,2,3,4\}$$

The optimal weights chosen by the user are then altered by an inverse power of the TV values for each stencil:

$$\omega'_{k,i}=\frac{\omega_{k,i}}{TV_{k,i}^p+\epsilon}:\quad k\in\{L,C,R,Brg\}$$

where we use $p=2$ (yet another parameter that can be tuned for smoothness) and $\epsilon$ is a small number to avoid dividing by zero. To limit polynomials, one must constrain $p>0$, and $p=2$ is the most common choice in literature. Finally, the weights are mapped to the domain $[0,1]$:

$$\gamma_{k,i}=\frac{\omega_{k,i}}{\sum_l\omega_{l,i}+\epsilon}:\quad k,l\in\{L,C,R,Brg\}$$

The final interpolant is given by a combination of candidate polynomials:

$$P_{WENO,i}(x)=\sum_k\gamma_{k,i}P_{k,i}(x):\quad k\in\{L,C,R,Brg\}$$

An important part of this process is that the cell-averaged derivative, $\overline{\mathbf{U}}_i^{(1)}$, must be replaced with the cell-averaged derivative of the resulting HWENO interpolant:

$$\overline{\mathbf{U}}_i^{(1)}\leftarrow\frac{1}{\Delta x}\int_{x_i-\frac{\Delta x}{2}}^{x_i+\frac{\Delta x}{2}}\frac{\partial}{\partial x}P_{HWENO,i}(x)dx$$

There are three advantages computationally for WENO and HWENO methods as compared to MM-Loc. First, WENO is a more adaptive method of controlling oscillations than a popular alternative: hyperdiffusion. It is robust enough to control shocks and retains optimal accuracy in smooth portions of the flow while hyperdiffusion indiscriminately damps small-scale modes throughout the domain. Second, the expense of the WENO procedure is offset some by the reduction in operation count for sampling and flux/source term computations. For WENO, only the first moment of the state is sampled and only the first moment of the flux and source functions is computed. Similarly for HWENO, only two moments are sampled and computed. Third, because there are fewer moments within cells to exchange in parallel, the volume of inter-node communication will also be reduced. Finally, we will show that HWENO accuracy is equal or better than fifth-order MM-Loc accuracy for smooth flows, a surprising but not unique finding [6].

For systems of equations, it is generally best to use the same WENO weights for all dynamical variables. For the SW system of equations, we reused the weights computed for $\phi$ to limit $\phi u$. In fact, for any system of equations with a similar Riemann structure to the Euler equations of gas dynamics, reusing the WENO weights of the density-type variable for all other dynamical variables is a stable choice. The reason is that shock, rarefaction, and contact discontinuities all show up in the density field Riemann solution, while only a subset show up in the other dynamical fields. Therefore, the density WENO weights are a conservative choice for limiting the entire system with the same WENO weights. There are other methods

of homogenizing the WENO weights for a system of equations such as computing a norm over all variables (see [22,23]), but using the density-type variable's weights for all other components is a smoother choice. Also, reusing density weights is the least computationally expensive.

Many WENO studies involving hyperbolic systems transform the state variables into Riemann invariants, apply the WENO limiting there, and then transform back into limited state variables. This is known to be a smoother and more robust method of applying WENO limiting to hyperbolic systems, but it is an expensive procedure. We found no need to do this for the experiments in this study, but nothing precludes this treatment in the multi-moment FV ADER framework for more challenging flows.

We also save some computational effort by reusing weights. However, when most studies mention the expense of the WENO procedure being in the computation of weights, they are referring to Type 1 methods for which the polynomial coefficients are only computed to compute WENO weights. Computing the actual polynomial coefficients is the majority of the expense of computing limited weights, and this is the reason Type 1 WENO studies consider it expensive to compute WENO weights. For Type 2 methods, the polynomial coefficients are already computed, and therefore computing WENO weights is a relatively small overhead. So the computational savings associated with reusing weights for the WENO and HWENO methods used herein are smaller than they would be for a Type 1 WENO method.

### 3.2. Local multi-moment ADER-Taylor method (MM-Loc)

In this method, we evolve all moments explicitly using derivatives of the flux and source terms (computed with differential transforms), giving the spatial locality of Galerkin schemes. We implement this method at up to 16th-order accuracy in space and time for all test cases.

#### 3.2.1. Linear analysis

We performed a von Neumann stability analysis on MM-Loc using the 1-D linear advection equation, $u_t + u_x = 0$, to verify that an $n$-moment method is always stable up to a CFL value of unity. Beyond a CFL value of unity, these methods are unstable except for the spatially uniform case for which the stability is neutral. The linear analysis was performed up to a 25th-order accurate method, and direct numerical experimentation confirms stability for up to 16th-order accurate methods when the data is smooth.

### 3.3. Review of multi-moment ADER-Taylor algorithm

To review the process of multi-moment ADER integration, we give the following algorithm for a single time step.

1. (Comm) Exchange any halo regions required for reconstruction (not necessary for MM-Loc)
2. Reconstruct a spatial polynomial, $P_i(x)$, over the cell from available moments
3. Sample reconstruction values and derivatives at the cell center $\partial_x^{(m)}\mathbf{U}(x_i)$
4. Perform C-K procedure to obtain time and mixed space–time derivatives valid at the cell center and beginning of the time step: $\partial_x^{(m)}\partial_t^{(n)}\mathbf{U}(x_i, t_n)$
5. Use space–time derivatives at the cell center and beginning of the time step to construct a time–space Taylor polynomial: $P_{Tay,i}(x,t)$
6. Sample $P_{Tay,i}(x,t)$ at cell boundary quadrature points in time (later used for flux computations)
7. (Comm) Exchange boundary values and derivatives of $\mathbf{U}$ with neighboring cells
8. Compute source term, $\mathbf{S}$, by sampling $P_{Tay,i}(x,t)$ at source term quadrature points in space and time
9. Compute interface fluxes using positive and negative limits of $\mathbf{U}$ and its derivatives at the interface
10. Perform the cell update using the source term and fluxes

The exchange of boundary values of $\mathbf{U}$ and its derivatives in step 7 can be overlapped with the source term computation in step 8 to hide some of the communication time. Steps involving inter-process communication are prefixed with "(Comm)".

## 4. Numerical experiments

We test these schemes using a variety of 1-D problems including linear advection, Burger's equation, and a 1-D SW model with terrain. At the moment, MM-Loc is not limited spatially. The development of the HWENO scheme demonstrates that limiting the modes in MM-Loc will also limit the time derivatives satisfactorily to produce a non-oscillatory simulation in the presence of shocks. Therefore, we feel that the development of a hybrid HWENO mode-limited scheme can be left to future study without loss of any novel exploration of the MM-Loc scheme's intrinsic properties. Simple modifications of [24] or [5] would likely give rise to such a scheme. Therefore, for MM-Loc, we only simulate smooth flows to determine the error convergence with increasing moments and decreasing grid spacing. Since all schemes herein are stable up to a CFL value of unity, we run all test cases with a maximum CFL value of 0.9 or greater.

There are five experiments in all. First, linear advection, $u_t + u_x = 0$, experiments will be performed by initializing (via 16-point Gauss–Legendre (GL) quadrature) with a one-period sine wave, $(\sin(2\pi x) + 1)/2$, on a domain of 1 m and revolving it cyclically around the domain 10 times with an advecting speed of 1 m s$^{-1}$. This test case is used to compute error norms for comparison. In order to do this, we stop the model at the ceiling of 10 s divided by the time step. The "analytic" cell means are computed with 16-point GL quadrature applied to a translation of the initial conditions to ensure compatibility with the final state and position. Error norms are computed by the cell means as follows:

$$L_1 = \frac{\sum_i \left|\bar{u}_{i,numer} - \bar{u}_{i,anal}\right|}{\sum_i \left|\bar{u}_{i,anal}\right|}$$

$$L_2 = \left( \frac{\sum_i \left(\bar{u}_{i,numer} - \bar{u}_{i,anal}\right)^2}{\sum_i \left(\bar{u}_{i,anal}\right)^2} \right)^{\frac{1}{2}}$$

$$L_\infty = \frac{\max_i \left(\bar{u}_{i,numer} - \bar{u}_{i,anal}\right)}{\max_i \left(\bar{u}_{i,anal}\right) - \min_i \left(\bar{u}_{i,anal}\right)}$$

where $\bar{u}_{i,numer}$ and $\bar{u}_{i,anal}$ are the numerical and analytical cell means, respectively.

Second, Burger's equation, $u_t + \left(u^2/2\right)_x = 0$, will be simulated with the WENO and HWENO methods on a domain of 1 m, initialized (again via 16-point GL quadrature) with the same one-period sine wave as the linear advection experiment. It will be simulated to a time of $t = 0.5$ s at which point a shock will have formed to test the oscillatory properties of the WENO and HWENO schemes. The rest of experiments are SW experiments described in further detail in Section 4.1.

When we perform experiments for $h$- and $p$-refinement, we always use the MSTS. It is true that if we are interested purely in the spatial aspects, that we should reduce the time step to very small CFL values in order to guarantee that only spatial convergence is manifesting. We believe the steady state SW simulations should suffice for this purpose. However, we are not most interested in purely spatial or theoretical viewpoints in this paper, but rather in real applications. At least in the climate field, one would not run an explicit integration method at a time step many times smaller than the MSTS. Therefore, as we refine the grid or the moments, we always use a time step close to the MSTS to ascertain convergence properties in a realistic context.

### 4.1. 1-D SW model test case

The conservation form of the SW equations in 1-D are as follows:

$$\frac{\partial}{\partial t}\begin{bmatrix} \phi \\ \phi u \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} \phi u \\ \phi u^2 + \frac{\phi^2}{2} \end{bmatrix} = \begin{bmatrix} 0 \\ -\phi \frac{\partial \phi_B}{\partial x} \end{bmatrix} \tag{7}$$

where $\phi(x,t) = gh(x,t)$ is the geopotential thickness of the fluid, $h(x,t)$ is the fluid thickness in meters, $g = 9.80616$ m s$^{-2}$ is acceleration due to gravity, $u(x,t)$ is the wind velocity, and $\phi_B(x)$ is the geopotential height of bottom orography. The free surface fluid geopotential height would be given by $\phi_{fs}(x,t) = \phi(x,t) + \phi_B(x)$. The simulation domain is similar to the radius of the Earth, $\Omega \in [0, L]$ with $L = 4 \times 10^7$ m.

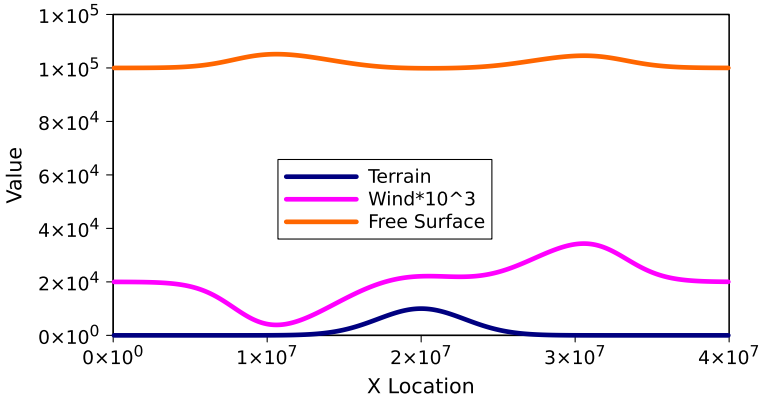
For the sake of converging to a steady state and ensuring that no shocks develop, this equation set is subjected to Dirichlet boundary conditions: $\phi(0,t) = \phi(L,t) = \phi_0 = 10^5$ m$^2$ s$^{-2}$ and $u(0,t) = u(L,t) = u_0 = 20$ m s$^{-1}$. This value of the geopotential is specified to simulate motions of the external atmospheric vertical mode, and it leads to a gravity wave speed on the order of 300 m s$^{-1}$. These test cases are unique compared to linear advection and Burger's equation because they introduce both a system and a source term. We will always use the following Gaussian terrain:

$$\phi_B(x) = \phi_{B,0} e^{-\left(\frac{x - x_C}{w}\right)^2}$$

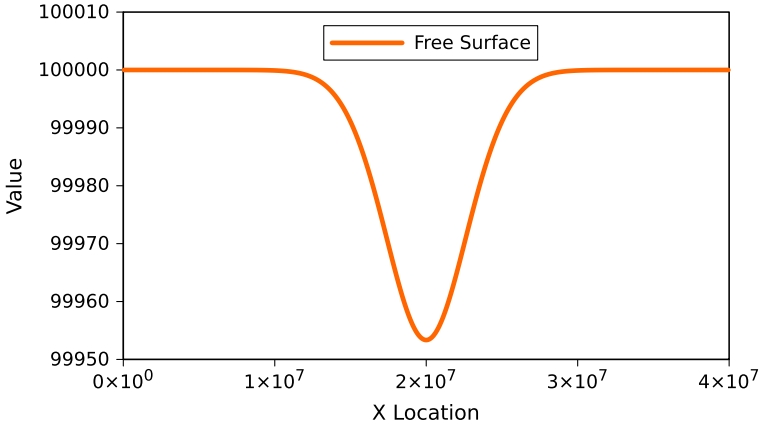
where $x_C = L/2$, $\phi_{B,0} = 10^4$ m$^2$ s$^{-2}$, and $w = L/10$. We initialize the model with $\phi(x,0) = \phi_0$ and $u(x,0) = u_0$, and the Dirichlet boundary conditions cause the transient motions to converge to a steady state before any shocks are allowed for form. The third test case we use is a transient SW solution taken at time $t = 3 \times 10^4$ s. Plots of the solution at $t = 3 \times 10^4$ s are given in Fig. 3(a).

As a reference for error norms, we compute the solution with a 16-moment MM-Loc scheme with 4000 points. In order to ensure that each model run, regardless of CFL number or time step, ends at exactly $t = 3 \times 10^4$ s, the last time step is reduced to bring the total model time to exactly this value. This means the last CFL number may be small for the last step. This would act to the advantage of a Runge–Kutta (RK) scheme, which will be discussed later, and to the disadvantage of the ADER schemes because the RK solution, dominated by temporal error, becomes more accurate with smaller time step while the ADER schemes become more diffusive with smaller time steps.

The fifth test case is to change the boundary conditions to cyclic and allow shocks to develop. For this test case, we initialize exactly like the transient SW test case above, meaning there is a Gaussian ridge in the free surface geopotential height at the domain center (due to terrain) that will propagate outward. These traveling waves will steepen and eventually form




(a) Plots of the free surface geopotential, terrain geopotential, and wind (scaled by $10^3$ for clarity) for the transient SW solution at $t = 3 \times 10^4$ seconds.




(b) Plots of the free surface geopotential steady-state SW solution.

**Fig. 3.** Plots of SW solutions for transient and steady state cases.

shocks which can further test the WENO and HWENO schemes within the context of a system with a source term. We stop the shock simulations at a time of $2.55 \times 10^5$ s for plotting, though they have been tested for much longer periods of time to ensure robustness.

*4.1.1. Steady-state solution*

Additionally, we specify a steady-state case by setting the time derivatives to zero, giving:

$$\frac{\partial}{\partial x}\begin{bmatrix} \phi u \\ \phi u^2 + \frac{\phi^2}{2} \end{bmatrix} = \begin{bmatrix} 0 \\ -\phi \frac{\partial \phi_B}{\partial x} \end{bmatrix}.$$

The first constraint implies that the "momentum", $\phi(x)u(x) = M_0 \equiv \phi_0 u_0$, is spatially uniform. Substituting this into the second constraint, we now have one boundary value ODE of one unknown with two boundary conditions:

$$\frac{d}{dx}\left(\frac{M_0}{\phi} + \frac{\phi^2}{2}\right) + \phi \frac{d\phi_B}{dx} = 0 \tag{8}$$

$$\phi(0) = \phi(L) = \phi_0$$

To solve this, we integrate the constraint over computational cells to place it into FV form and discretize spatially with piecewise linear functions. We then use a tridiagonal linear solver wrapped inside a Newton iteration with a convergence criterion of $10^{-12}$ relative tolerance. In practice, the spatially uniform momentum, $M_0$, serves as an easier measure for accuracy when computing error norms since the constant value is accurate to machine precision, and we find similar results with each approach.

There will be numerical imbalance between the body source term and the interface fluxes. Comparing against the accurate initial data, we can obtain robust comparative error measures in order to compare each method's handling of the system at steady state. Also, this gives a good estimate of convergence with grid and moment refinement. The solution to (8) qualitatively looks like $\phi(x,0) = \phi_0 - \phi_B(x)$, that is a uniform free surface geopotential, when viewed at scale, but this is not the actual solution due to the $M_0/\phi$ term arising from the fact that this is not an infinite domain experiment. See Fig. 3(b) for a zoomed view of the free-surface geopotential in steady state. It must be solved non-linearly to get full accuracy at steady state on the finite Dirichlet-bounded domain.

### 4.2. Runge–Kutta Discontinuous Galerkin method for comparison

To provide a reference for accuracy from a comparable type of scheme that is common in literature, we include a three-stage Strong Stability Preserving (SSP) Runge–Kutta (RK) modal Discontinuous Galerkin (DG) method with a spatial order of accuracy varying from 1 to 16. For a detailed description of the scheme in 1-D, please see [25]. For discretization of the source term for the 1-D SW system, we perform a straightforward extension of the scheme by multiplying the test function to the source term and integrating, thereby using a mapping of the source term to the Legendre basis functions for time evolution.

We found that this RKDG scheme is dominated by temporal error when the MSTS is used due to the RK time integrator. Results from this method and that of Section 4.3 demonstrate that the RK integrator itself has the property of being more accurate at smaller time steps. This is in contrast to the ADER implementations which exhibit the property of being more accurate at time steps closer to unity. As the RK time step is reduced to sufficiently low values, the error converges super-exponentially with $p$-refinement for DG, and the polynomial convergence with $h$-refinement is larger than the order of the polynomials denoting, again, super-convergence. For realistic simulations, however, the MSTS will be used, meaning temporal error will dominate after roughly third-order accuracy. Linear analysis and numerical experimentation show that the MSCFL value for this RKDG scales close to

$$CFL_{max} = \min(1, 1.318p^{-1.672}) \tag{9}$$

where $p$ is the number of moments. We use $\Delta t = 0.8\Delta x(CFL_{max}/c_{max})$ in all RKDG experiments herein, and we provide the maximum initial wave speed, used as $c_{max}$, so that the user may compute the RKDG time step used in each simulation. We do not use an adaptive time step. Rather, we use a constant time step such that the maximum CFL number during the simulation is close to $CFL_{max}$. Time steps will not play a significant role for the steady-state SW test case, and the RKDG scheme is expected to converge significantly faster because of the DG spatial accuracy.

This RKDG scheme is not fully representative of other potential RKDG (or SE) schemes because the RKDG MSCFL value is dependent on the quadrature of the DG scheme and the type of RK scheme used [26]. Generally, for a fixed RK integration method, the larger the time step used, the lower the accuracy. Therefore a comprehensive comparison of overall effectiveness incorporating accuracy, MSCFL value, computational effort per time step, local memory requirements, and number of stages is no simple task even within RKDG methods themselves, much less between RKDG methods and the methods in this study. Still, we wish to provide the reader with some reference point for accuracy and runtime comparison.

### 4.3. Runge–Kutta multi-moment FV method for comparison

In Section 2.2, it was mentioned that locally storing the $n(n+1)/2$ space–time derivatives (where $n$ is the number of moments) could cause some difficulties at very high $n$ with the limited local memory stores on accelerator devices, such as Graphics Processing Units (GPUs), being included on modern computing platforms. In light of this, it is natural to consider whether a similar method with lower local memory requirements might be competitive. Therefore, we implemented a SSP RK analog of the MM-Loc scheme wherein we still evolve all moments via derivatives of the flux and source vectors but use a RK integrator instead of ADER. We label this scheme RK-MM-Loc, and its time stepping only requires storing the $n$ derivatives that are explicitly evolved, and the MSTS still remains constant during $p$-refinement.

We will show that RK-MM-Loc fails in terms of accuracy, mainly due to temporal errors. As the scheme undergoes $p$-refinement, the spatial error decreases rapidly as it does also in the DG and MM-Loc schemes. However, unlike the RKDG scheme, the MSTS remains at unity because it is a multi-moment FV scheme. For the SSP RK3 time integrator we use (same as RKDG), temporal error dominates quickly as $p$-refinement increases because of the size of the time step. While we report a similar phenomenon for the RKDG scheme, the RKDG MSTS decreases as the number of moments increases. Therefore the error is significantly lower for RKDG, and $p$-refinement still adds value to the simulation in terms of error. When $p$-refining the RK-MM-Loc scheme, the error does not decrease significantly past sixth-order accuracy.

To obtain a benefit from $p$-refinement (or $h$-refinement at high-order spatial accuracy) at very high orders of accuracy for the RK-MM-Loc scheme, one must either decrease the time step with $p$-refinement or improve the accuracy of the RK scheme, which means more stages. In a distributed-memory setting, either option results in increased inter-node communication in parallel, the very attribute we wish to minimize. Though RK-MM-Loc is not a viable scheme for higher-orders of accuracy without a much more accurate RK integrator, comparing MM-Loc against RK-MM-Loc serves to highlight the excellent accuracy that the ADER time integration achieves in general for such a large (near unity) fully-discrete time step. This is because the time integration remains at the same order of accuracy as the spatial as all components of the PDE are evolved

together to high order over the length of a time step. We believe this makes the MM-Loc scheme unique in that it is highly accurate at quite large time steps compared to Galerkin schemes and achieves this accuracy in a fully-discrete manner.

### 4.4. Discussion of accuracy

$L_1$ error norms for linear advection, transient SW, and steady-state SW test cases as well as a solution plot of the SW and Burger's equation shocks for the WENO and HWENO schemes are plotted in Figs. 4–12.

#### 4.4.1. Linear advection

For MM-Loc, WENO, and HWENO, we use the relation $CFL = \Delta t/\Delta x = 0.95$ to compute the time step in each linear advection simulation. For linear advection, WENO, HWENO, and MM-Loc achieved $h$-refinement convergences of 5.00, 5.00, and 4.96, respectively. Surprisingly, MM-Loc and WENO errors were very similar; and at the higher refinements, HWENO error was five times less than MM-Loc and WENO in all error norms. The RKDG method shows exactly third-order $h$-convergence even though five moments are used. This is due to the third-order accurate temporal discretization and the fact that the MSTS is used as the grid is refined. This convergence shows that this RKDG error is dominated by temporal truncation. To achieve full fifth-order accurate convergence using the MSTS, one would need to use a fifth-order accurate RK integrator or decrease the time step more rapidly with spatial refinement.

For the $p$-refinement tests, we show refinement for two grid sizes: 25 and 100 cells. Again, because the MSTS is used, the RKDG scheme does not see full exponential $p$-refinement convergence even though it is super-convergent spatially. In fact, for the first few moments in $p$-refinement, the RKDG scheme undergoes rapid convergence, and past three moments the convergence relaxes from exponential to a polynomial convergence. Curiously, the polynomial convergence follows a $p^5$ power law past three moments for both 25 and 100 cells. The MM-Loc scheme exhibits exactly exponential convergence until machine precision effects come into play, though this convergence is slower than that initially shown by RKDG. Because of this, there is a range of $p$-refinement for which RKDG is more accurate. However, after break-even points of roughly $p = 7$ and $p = 5$ moments when using 25 and 100 cells (respectively), MM-Loc is more accurate. As more cells are used, this break-even point occurs at a lower number of moments. Exponential convergence for MM-Loc shows that the ADER time discretization keeps temporal error from dominating overall truncation, allowing expected convergence with spatial refinement even with a MSTS that remains constant with $p$-refinement in a fully-discrete treatment.

#### 4.4.2. Burger's equation shock

For WENO and HWENO, we use the relation $CFL = \Delta t/\Delta x = 0.95$ to compute the time step in each Burger's equation simulation. Fig. 6 shows solution plots of Burger's equation after a shock has developed. This is to demonstrate that the WENO and HWENO limiting is working appropriately. We have run Burger's equation over a variety of less-than-unity CFL numbers for up to 5 s with no signs of developing oscillations. We clip the $y$-axis for clarity, leaving off the minima in first derivative which are limited to $-27.3$ in the numerical solutions. Note that neither the value nor derivative are exhibiting oscillations. Plotting points plotted along the shock shows that the HWENO solution is less diffused than WENO.

#### 4.4.3. Transient SW

For WENO, HWENO, and MM-Loc, we use the relation $CFL = c_{max}\Delta t/\Delta x = 0.93$ to compute the time step in each simulation, and $c_{max}$ is considered to be the initial maximum wave speed: $c_{max} = u_0 + \sqrt{\phi_0} = 20 + \sqrt{10^5} \approx 336.23$. In Figs. 7 and 8, we give $h$- and $p$-convergence plots for the transient SW test case as simulated by the WENO, HWENO, MM-Loc, RKDG, and RK-MM-Loc methods. HWENO $h$-refinement convergence ranged form an initial 6.07 to a final 4.55 in our experiments.

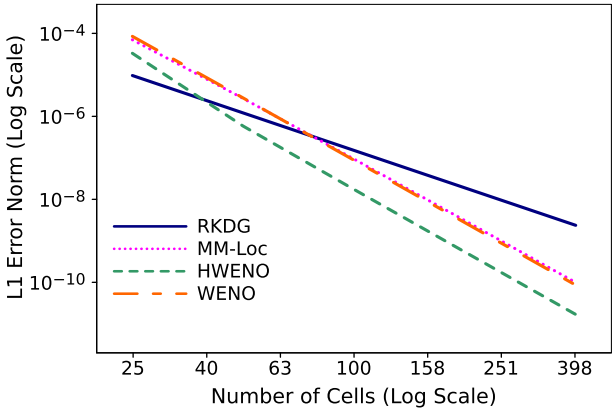



**Fig. 4.** $h$-Refinement $L_1$ error norm plots for the linear advection test case.

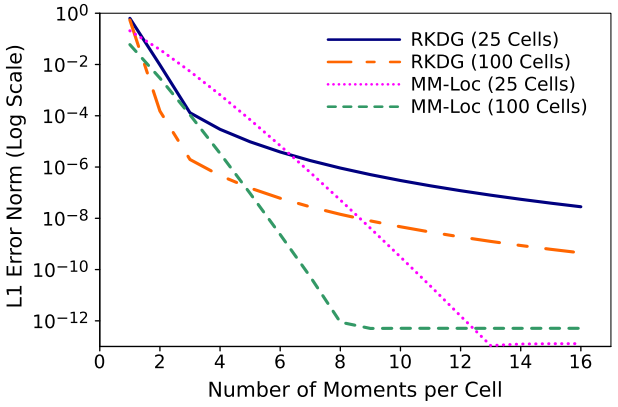



**Fig. 5.** $p$-Refinement $L_1$ error norm plots for the linear advection test case.

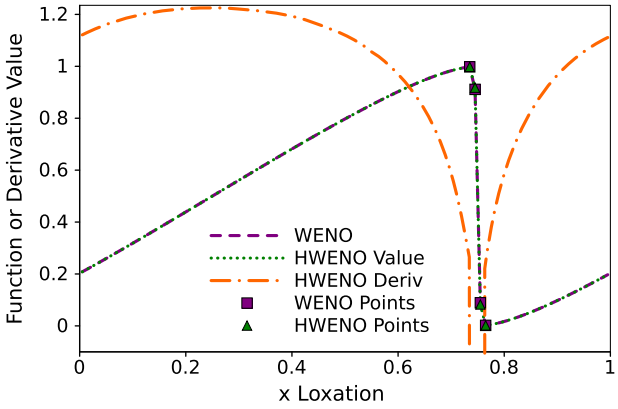



**Fig. 6.** Plot of the WENO and HWENO values and HWENO derivative after simulating Burger's equation initialized with a one-period sine wave up to $t = 0.5$. Blue boxes and magenta triangles show points along the shock for the WENO and HWENO values, respectively. The only visually discernible differences are along the shock. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

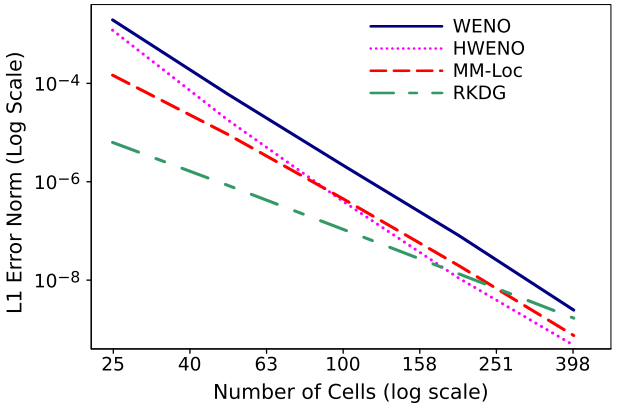



**Fig. 7.** $h$-Refinement $L_1$ error norm plots for the transient SW test case.

HWENO errors at the lowest refinement were an order of magnitude higher than MM-Loc, but they quickly become smaller than MM-Loc due to relatively higher convergence rates. At 100 and 200 cells, all HWENO error norms were roughly two times lower than MM-Loc, but at 400 cells, though HWENO $L_1$ error was lower, the $L_\infty$ error was five times higher. This $L_\infty$ behavior is not uncommon for limited functions as it generally denotes poorer handling of extrema in smoother cases such as this SW test case.

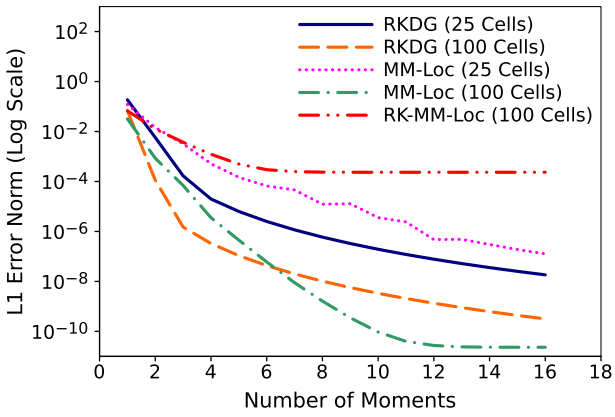



**Fig. 8.** $p$-Refinement $L_1$ error norm plots for the transient SW test case.

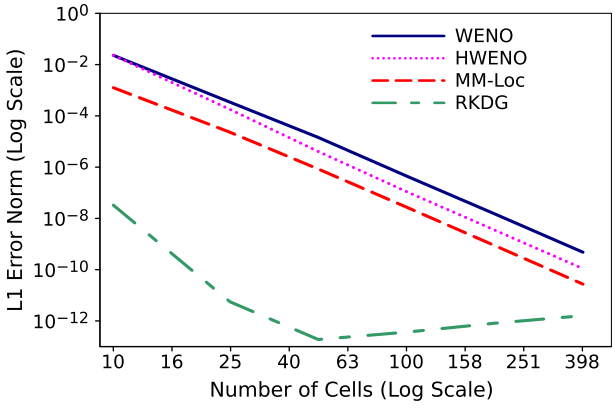



**Fig. 9.** $h$-Refinement $L_1$ error norm plots for the steady SW test case.

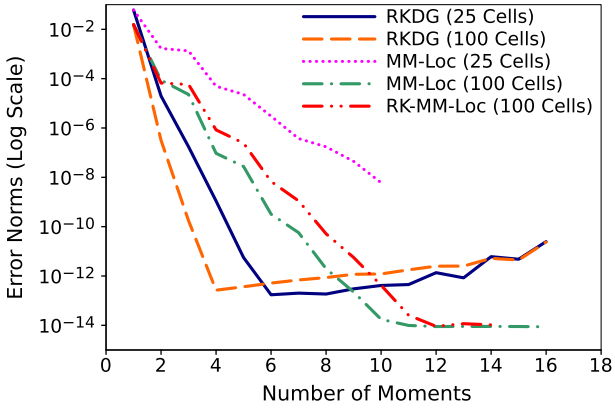



**Fig. 10.** $p$-Refinement $L_1$ error norm plots for the steady SW test case.

MM-Loc showed an initial convergence of 4.00 and increased up to 4.73 at our highest refinement values. WENO consistently gave a convergence of 5.02 to 5.05 throughout. RKDG again showed exactly third-order convergence throughout the experiments, again due to the time discretization and dominating temporal error. After 100 cells, HWENO $L_1$, $L_2$, and $L_\infty$ errors averaged 6.0, 6.7, and 11.3 times lower than WENO error norms, showing that using more local moments helps improve the largest errors the most in a WENO context.

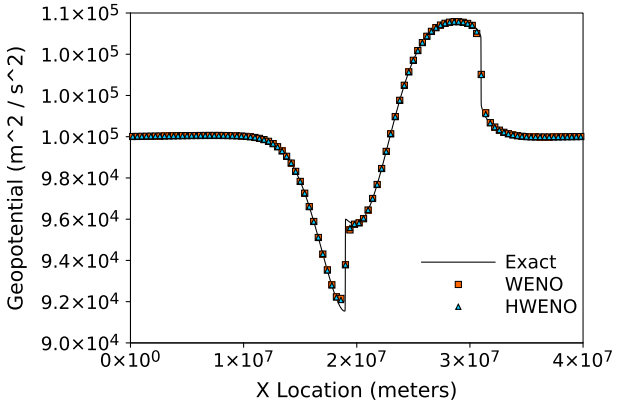



**Fig. 11.** Plot of $\phi$ after $2.55 \times 10^5$ s of simulation of the SW equation set with cyclic boundaries.

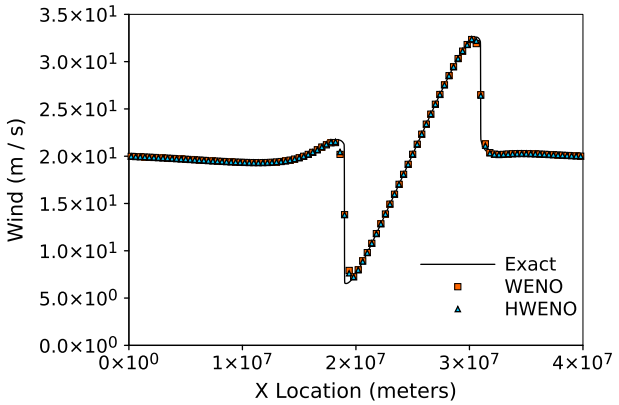



**Fig. 12.** Plot of $u$ after $2.55 \times 10^5$ s of simulation of the SW equation set with cyclic boundaries.

For $p$-refinement, we find that convergence rates are altogether lower than those found in the linear advection test case, which is not surprising. Non-linear dynamics along with more sharply varying solution features and a source term give a greater challenge for numerical methods. RKDG shows polynomial $p$-refinement convergence past three moments, again following a $p^5$ power law. Though MM-Loc convergence is clearly sub-exponential, it is also super-polynomial when viewed in log–log space. This means that with increasing moments, the exponential coefficient decreases, but the polynomial coefficient increases. Again, there is greater convergence with MM-Loc than RKDG, and therefore using 100 cells, MM-Loc is more accurate after $p \approx 6$ moments.

The RK-MM-Loc scheme is also included in this test case to demonstrate the point that $p$-refinement does not add value to the solution past the point where temporal error dominates. The RKDG does become more accurate with $p$-refinement past three moments because the MSTS decreases by a power law based on the number of moments. Thus, with a lower time step, the error continues to decrease. The RK-MM-Loc scheme does not share this property, but the MSTS remains at unity no matter how many moments are used. When undergoing $p$-refinement with a RK time integrator, one must either decrease the time step or increase the order/accuracy of the RK integration. With the ADER time discretization, temporal error is decreased without added stages or decreased time step.

*4.4.4. Steady-state SW*

These simulations use the same time step as in Section 4.4.3. We use the steady-state error norms for pure spatial refinement considerations. Figs. 9 and 10 show error plots for $h$- and $p$-convergence of WENO, HWENO, MM-Loc, RKDG, and RK-MM-Loc schemes for the steady-state SW test case. At the higher $h$-refinement levels, WENO error was consistently four times lower than HWENO error, and MM-Loc error was in turn four to five times lower than HWENO error. RKDG error was clearly in a different regime altogether, showing how good the spatial accuracy of a Galerkin method is and further demonstrating how temporal error is dominating the overall RKDG numerical solution. HWENO $h$-convergence begins at order 5.6 but asymptotes to fifth-order accuracy along with the rest of the methods except RKDG which exhibits superconvergence.

For $p$-refinement, values past 10 moments for MM-Loc using 25 cells are omitted because the simulation became unstable in the absence of limiting. Also, RK-MM-Loc became unstable with 15 and 16 moments using 100 cells and those values are omitted from the plots as well. RKDG shows exponential convergence down to the level of machine precision effects, and

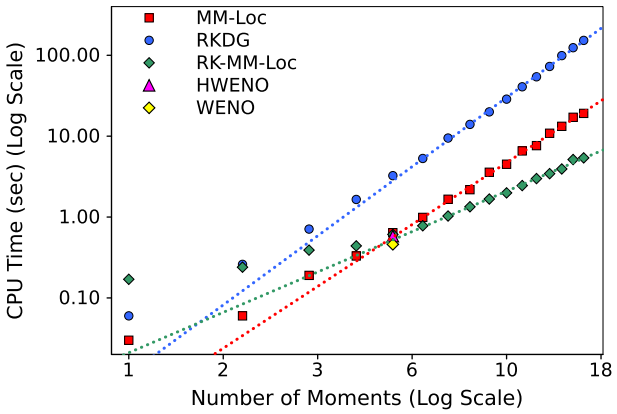



**Fig. 13.** Plot of run times for RKDG and MM-Loc for the transient SW test case run with 1000 cells over a range of moments. Also plotted are HWENO and WENO run times at the 5-moment location along the $x$-axis. All simulation times were performed on one core of a 2.7 GHz Intel processor with the GNU Fortran compiler (gfortran) using the following optimization flags: "-finline-limit=2000 -O3 -msse4.2 -mfpmath=sse -fdefault-real-8 -funroll-loops". Power regressions fitting the asymptotic tendencies of the run times are plotted.

MM-Loc also shows exponential convergence as well but at a lower rate. We include the RK-MM-Loc scheme to show that temporal error was indeed dominating the solution, since it exhibits exponential spatial convergence in the steady state at a rate close to MM-Loc using the same number of cells.

#### 4.4.5. SW shock formation

These simulations use the same time step as in Section 4.4.3. Figs. 11 and 12 give solution plots of the geopotential and wind at $2.55 \times 10^5$ s to demonstrate the limiting of WENO and HWENO within the context of a system of equations with a source term. The line showing the exact solutions was a refined solution run with the HWENO method using 4000 cells to show the structure between shocks more clearly. It is clearer in this case than in the Burger's equation shock that HWENO is diffusing the shocks less than WENO. Though we ran the simulations plotted at a maximum CFL value near 0.99, we have tested for longer periods of time and other CFL values to ensure the oscillations are controlled and the method is robust.

### 4.5. Discussion of run times

Run times (CPU time in seconds) for all methods implemented in this study using the transient SW test case and 1000 cells are plotted in Fig. 13. We also plot a power law matching asymptotic runtime complexity with $p$-refinement. For all methods, expected optimizations are made. Loops respect the column-major array addressing of Fortran, redundant computation is generally avoided, and all values that remain constant are pre-computed. This includes Legendre polynomial and derivative evaluations at spatial quadrature points for RKDG and terrain sampling values and derivatives for all methods. While run times may vary based on processor or compiler, we wish to give the reader some sense of how both accuracy and run time of the WENO, HWENO, and MM-Loc schemes compare to a more familiar type of scheme.

RKDG, MM-Loc, and RK-MM-Loc run times grow with $p^{3.42}$, $p^{3.06}$, and $p^{2.00}$, respectively. Though RK-MM-Loc shows only quadratic complexity with increasing moments, accuracy does not improve without decreasing the time step or increasing the number of stages which itself increases the overall complexity and increases parallel communication costs. At $p = 4$, 8, and 16, MM-Loc in serial runs 5.0×, 6.4×, and 8.0× faster than RKDG. WENO and HWENO run 28% and 9% faster (respectively) than MM-Loc. Though the WENO and HWENO reconstruction procedures are computationally expensive, the sampling and flux/source term computation takes less work because only the value and/or derivative must be evolved.

Though MM-Loc is significantly more costly than RKDG per time step, many fewer time steps are taken because the CFL remains at unity during $p$-refinement. Using the RKDG MSTS scaling relation from Section 4.2 and that there are three stages per time step; at $p = 4$, 8, and 16, MM-Loc requires 23×, 74×, and 235× fewer stages of parallel communication than RKDG. We believe that the increase in computation-to-communication ratio will lead to greater parallel efficiency, especially in extreme strong scaling cases up to order $10^4$ compute nodes. Given that these methods run faster in serial and that much better parallel efficiency can be achieved, these multi-moment ADER methods may allow for further refinement for the same overall throughput. Thus, they are competitive options for further investigation in multiple dimensions and with more realistic dynamics and grid topologies.

## 5. Conclusions and future work

We have introduced a new integration method for systems of conservation laws in one dimension with source terms by combining the high-order-accurate, fully-discrete time integration of ADER methods, the spatial locality afforded by evolving multiple moments per cell, and the constant CFL limit with $p$-refinement that the FV framework allows. Also, we modified

the traditional ADER approach to reduce cost by performing the expensive C-K procedure only once at the cell center. Using the Differential Transform Method for high-order space–time derivatives further reduces the cost of the C-K procedure, especially for complex flux and source functions. Though these multi-moment FV ADER methods are applied to hyperbolic conservation laws here, this is not a restriction of the scheme. It is applicable to the more general class of time-dependent PDE systems with low enough stiffness to afford a time-explicit discretization.

These space–time derivatives then form a space–time Taylor polynomial over the cell which is sampled when needed. Therefore, we call these methods ADER-Taylor methods. Sampling a Taylor series is cheaper than performing the C-K procedure, leading to a reduction in run time. Because multi-stage time integration is no longer needed and because of the larger time step that does not reduce with added moments, the proposed methods require minimal communication for time-explicit integration. This also means computation is better clustered between fewer stages of communication which leads to enhanced parallel efficiency on distributed memory architectures.

We implemented three methods to explore the multi-moment ADER-Taylor framework: (1) a single-moment WENO ADER-Taylor method (abbreviated as "WENO"), (2) a two-moment HWENO ADER-Taylor method (abbreviated as "HWENO"), and (3) an entirely local multi-moment ADER-Taylor method (abbreviated as "MM-Loc"). MM-Loc requires no communication during the reconstruction phase, similar to Galerkin methods. For all methods, the time step remains bounded only by the linear CFL condition which is unity in 1-D. This was verified both by linear analysis and experimentation. These methods were also tested with five experiments: (1) linear advection of a sine wave, (2) a shock with Burger's equation, (3) a transient SW system, (4) a steady-state SW system, and (5) a SW shock formation.

The WENO and HWENO methods successfully limited oscillations for the shocks of Burger's and SW equations, and this supports the proposal that a hybrid scheme with the MM-Loc method in smooth regions and the HWENO method in so-called problem cells could be an effective strategy for robust simulation of shocks and smooth regions. All schemes achieved their desired order of accuracy during $h$-refinement. Also, during $p$-refinement for linear advection, the MM-Loc method exhibited the expected exponential convergence. For the non-linear transient SW system, the MM-Loc method showed super-polynomial yet sub-exponential convergence, and exponential convergence was shown for the steady-state SW system. Particularly, this demonstrates that the source term and fluxes were coupled to the order of accuracy of the scheme successfully by including the source term in the C-K procedure.

In comparison to a similar type of method in literature, the Runge–Kutta Discontinuous Galerkin method, the MM-Loc method was often less accurate at low refinement levels. However, because of better $h$- and $p$-convergence rates for time-dependent test cases when using the MSTS, MM-Loc accuracy usually surpassed RKDG after some refinement level. MM-Loc was also significantly cheaper in serial and requires many fewer stages of parallel inter-node communications, leading us to believe that there is room to refine the MM-Loc method and still achieve the same overall runtime. For these reasons, we believe the multi-moment ADER-Taylor-based methods may be competitive for time-explicit simulations on very large distributed-memory computers.

In going forward with these methods, the most immediate considerations are limiting oscillations and multi-dimensionality in space. Because the scheme relies on a full tensor of multi-dimensional space–time derivatives, it should be applicable to arbitrary unstructured meshes. A hybrid MM-Loc – HWENO scheme using troubled-cell indicators is likely a quick follow-onto what is presented here. This may be challenging in a distributed parallel environment, however, because some processing elements will have more troubled cells than others. This would likely lead to computational load imbalances. Also, though literature exists for two-dimensional HWENO methods, it has room to grow, particularly in using derivatives of order greater than one. Hyperdiffusion is another option, but it indiscriminately damps the entire grid. Also, it needs to be established whether it should be performed in modal space or by transferring between modal and nodal expressions with standard finite-difference hyperdiffusion.

Work is currently being performed that removes the use of quadrature from the multi-moment, ADER-Taylor algorithm. We are investigating the reuse of intermediate steps in the C-K procedure such that flux and source terms are directly expanded using DTs, making them space-time polynomial expansions that may be sampled and integrated directly in space and time with a simple vector dot product. Fluxes, in this manner, may be computed only once per time step using time-averaged states to either diagonalize the PDEs for a linear Godunov state or compute a central flux such as local Lax-Friedrichs. The future removal of quadrature is expected to provide much speed-up of the multi-moment, ADER-Taylor algorithm and will be published in a future work.

Extending MM-Loc to very high-order accuracy in multiple spatial dimensions, though a straightforward task in theory and implementation, will change the computational landscape, meaning different efficiency attributes compared to existing methods in multiple spatial dimensions. The greatly reduced communication frequency will remain, but the serial runtime compared to Galerkin schemes may change. The main difficulty in extending to multiple spatial dimensions, however, will be the robust limiting of oscillations. We believe this task should be dealt with carefully, perhaps improving upon exiting multi-dimensional HWENO methods and/or developing new ways to realize hyperdiffusion within the discretization.

## Acknowledgements

The corresponding author is thankful to internal reviewers at Oak Ridge National Laboratory for feedback that greatly improved the manuscript clarity. This research used resources of the National Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-

AC05-00OR22725. The submitted manuscript contains contributions by UChicago Argonne, LLC, Operator of Argonne National Laboratory (“Argonne”). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357.

## References

[1] V.A. Titarev, E.F. Toro, ADER: arbitrary high order Godunov approach, Journal of Scientific Computing 17 (1) (2002) 609–618.
[2] E. Toro, V. Titarev, Derivative Riemann solvers for systems of conservation laws and ADER methods, Journal of Computational Physics 212 (1) (2006) 150–165.
[3] E. Toro, V. Titarev, ADER schemes for scalar non-linear hyperbolic conservation laws with source terms in three-space dimensions, Journal of Computational Physics 202 (1) (2005) 196–215.
[4] G. Capdeville, A Hermite upwind WENO scheme for solving hyperbolic conservation laws, Journal of Computational Physics 227 (2008) 2430–2454.
[5] J. Qiu, Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method: one-dimensional case, Journal of Computational Physics 193 (1) (2004) 115–135.
[6] D. Balsara, C. Altmann, C. Munz, M. Dumbser, A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG+HWENO schemes, Journal of Computational Physics 226 (1) (2007) 586–620.
[7] M. Dumbser, C. Munz, ADER discontinuous Galerkin schemes for aeroacoustics, Comptes Rendus Mécanique 333 (9) (2005) 683–687.
[8] R.B. Neale, C.-C. Chen, A. Gettelman, P.H. Lauritzen, S. Park, D.L. Williamson, A.J. Conley, R. Garcia, D. Kinnison, J.-F. Lamarque, D. Marsh, M. Mills, A.K. Smith, S. Tilmes, F. Vitt, H. Morrison, P. Cameron-Smith, W.D. Collins, M.J. Iacono, R.C. Easter, S.J. Ghan, X. Liu, P.J. Rasch, M.A. Taylor, Description of the NCAR Community Atmosphere Model (CAM 5.0), Technical Report, 2010.
[9] M. Lukáčová-Medvid'ová, J. Saibertová, G. Warnecke, Finite volume evolution Galerkin methods for nonlinear hyperbolic systems, Journal of Computational Physics 183 (2) (2002) 533–562.
[10] M.R. Norman, R.D. Nair, F.H.M. Semazzi, A low communication and large time step explicit finite-volume solver for non-hydrostatic atmospheric dynamics, Journal of Computational Physics 230 (4) (2011) 1567–1584.
[11] S. Ii, F. Xiao, CIP/multi-moment finite volume method for Euler equations: a semi-Lagrangian characteristic formulation, Journal of Computational Physics 222 (2) (2007) 849–871.
[12] A.M. Gofen, The ordinary differential equations and automatic differentiation unified, Complex Variable and Elliptic Equations 54 (9) (2009) 825–854.
[13] I. Tsukanov, M. Hill, Fast Forward Automatic Differentiation Library (FADLIB), User Manual, 2000.
[14] C.-K. Chen, S.-H. Ho, Application of differential transformation to eigenvalue problems, Applied Mathematics and Computation 79 (2–3) (1996) 173–188.
[15] A. Kurnaza, G. Oturanç, M.E. Kiris, n-Dimensional differential transformation method for solving PDEs, International Journal of Computer Mathematics 82 (3) (2005) 369–380.
[16] W. Pankiewicz, Calculation of a polynomial and its derivative values by Horner scheme, Communications of the ACM 11 (9) (1968) 633.
[17] G. Capdeville, A central WENO scheme for solving hyperbolic conservation laws on non-uniform meshes, Journal of Computational Physics 227 (2008) 2977–3014.
[18] L. Zhang, W. Liu, L. He, X. Deng, H. Zhang, A class of hybrid DG/FV methods for conservation laws I: basic formulation and one-dimensional systems, Journal of Computational Physics 231 (4) (2012) 1081–1103.
[19] J. Qiu, C. Shu, Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method II: two dimensional case, Computers & Fluids 34 (6) (2005) 642–663.
[20] C.W. Shu, High order ENO and WENO schemes for computational fluid dynamics, in: T.J. Barth, H. Deconinck (Eds.), High-Order Methods for Computational Physics, Lecture Notes in Computational Science and Engineering, vol. 9, Springer, 1999, pp. 439–582.
[21] G. Capdeville, A Hermite upwind WENO scheme for solving hyperbolic conservation laws, Journal of Computational Physics 227 (4) (2008) 2430–2454.
[22] D. Levy, G. Puppo, G. Russo, A fourth-order central WENO scheme for multidimensional hyperbolic systems of conservation laws, SIAM Journal on Scientific Computing 24 (2002) 480–506.
[23] D. Levy, G. Puppo, G. Russo, Central WENO schemes for hyperbolic systems of conservation laws, Mathematical Modelling and Numerical Analysis 33 (1999) 547–571.
[24] D.S. Balsara, C. Altmann, C.D. Munz, M. Dumbser, A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG+HWENO schemes, Journal of Computational Physics 226 (2007) 586–620.
[25] R.D. Nair, S.J. Thomas, R.D. Loft, A discontinuous Galerkin transport scheme on the cubed sphere, Monthly Weather Review 133 (4) (2005) 814–828.
[26] E. Kubatko, J. Westerink, C. Dawson, Semi discrete discontinuous Galerkin methods and stage-exceeding-order, strong-stability-preserving Runge-Kutta time discretizations, Journal of Computational Physics 222 (2) (2007) 832–848.