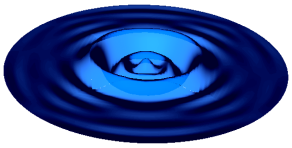
Spectral Element Libraries in Fortran

(SELF)

Shallow Water Equations



Reference Manual

Joseph Schoonover

Contents

1 Equations and Discretization

The inviscid shallow water equations are a system of coupled partial differential equations that is hyperbolic. The Nodal Discontinuous Galerkin Spectral Element Method (NDGSEM) is well suited for hyperbolic systems. It is highly accurate through its use of arbitrarily high order piecewise polynomial representation and exhibits minimal numerical dispersion and dissipation errors. It additionally allows for a flexible treatment of the geometry through the use of a structured or unstructured arrangement of *spectral elements* that can represent boundaries through high order polynomial interpolants. The discretization yields dense and local matrix-vector multiplies for computing derivatives and only requires nearest neighbor communication to calculate exchanges between elements. This last property makes the NDGSEM trivial to parallelize. For these reasons, the NDGSEM is applied to the shallow water equation systems (1.1), (1.2), and (1.3) to demonstrate its accuracy and scalability for a system of equations that hold many of the important dynamics in geophysics.

1.1 Flavors of the Shallow Water Equations

The shallow water equations can be written in either "conservative" (Eqs. 1.1) or "skew-symmetric" (Eqs. 1.2) form :

$$U_t + \nabla \cdot \left(\frac{\vec{U}U}{H} + \frac{gH^2}{2}\hat{x} \right) - fV = gHh_x + q_U \tag{1.1a}$$

$$V_t + \nabla \cdot \left(\frac{\vec{U}V}{H} + \frac{gH^2}{2}\hat{y} \right) + fU = gHh_y + q_V \tag{1.1b}$$

$$H_t + \nabla \cdot \vec{U} = q_H \tag{1.1c}$$

$$u_t + \nabla \cdot ((e+p)\hat{x}) - (f+\zeta)v = q_u \tag{1.2a}$$

$$v_t + \nabla \cdot ((e+p)\hat{y}) + (f+\zeta)u = q_v \tag{1.2b}$$

$$p_t + \nabla \cdot ((gh+p)\vec{u}) = q_p \tag{1.2c}$$

For either system, and throughout the rest of this documentation the symbols have the following meaning :

x, y : Cartesian coordinate positions ($Length$)

t : Time

U, V : The x, y components of the the fluid *transport* ($\frac{Length^2}{Time}$)

H : The total fluid thickness, bathymetric plus free surface ($Length$)

h : Resting fluid depth, the fluid bathymetry ($Length$)

g : Local acceleration of gravity ($\frac{Length}{Time^2}$)

f : Coriolis parameter ($Time^{-1}$)

u, v : The x, y components of the *depth average* fluid velocity ($\frac{Length}{Time}$)

e : The kinetic energy of the depth averaged velocity field ($\frac{Length^2}{Time^2}$)

p : Barotropic pressure, the product of gravitational acceleration and free surface anomaly ($\frac{Length^2}{Time^2}$)

ζ : The relative vorticity of the depth averaged velocity field ($Time^{-1}$)

q_ϕ : Any additional non-conservative source for the variable ϕ ($\frac{\phi}{Time}$)

The linear shallow water equations (1.3) are given as

$$u_t + \nabla \cdot (p\hat{x}) - fv = q_u \tag{1.3a}$$

$$v_t + \nabla \cdot (p\hat{y}) + fu = q_v \tag{1.3b}$$

$$p_t + \nabla \cdot (g\vec{u}) = q_p \tag{1.3c}$$

Any flavor of the shallow water equations arise from a conservation law, and as such, (1.1), (1.2), or (1.3) can be written in the compact form

$$\vec{s} + \nabla \cdot \vec{f} = \vec{q}. \tag{1.4}$$

The solution vector is represented by $\vec{s}$, $\vec{f}$ is a conservative flux tensor, and $\vec{q}$ is a vector of non-conservative source terms. The derivation of the discrete equations is presented through a discretization of (1.4).

1.2 Nodal Discontinuous Galerkin Discretization

The NDGSEM discretizes (1.4) in its weak form. To obtain the weak form, (1.4) is weighted with a test function and integrated over the physical domain, call it Ω. To approximate the integrals in the weak form, the domain is first divided into non-overlapping elements (Ω^κ) . Integration is performed over each element and the solution and test function are permitted to be piecewise discontinuous across elements. These assumptions lead to the statement

$$\int_{\Omega^\kappa} (\vec{s}_t + \vec{q})\phi \, d\Omega^\kappa - \int_{\Omega^\kappa} \vec{f} \cdot \nabla\phi \, d\Omega^\kappa + \oint_{\partial\Omega^\kappa} \phi \vec{f} \cdot \hat{n} \, dA^\kappa = 0, \quad \forall \phi \in \mathbb{C}_0(\Omega_\kappa), \quad \kappa = 1, 2, ..., K \tag{1.5}$$

where $\mathbb{C}_0(\Omega^\kappa)$ is the space of functions that are continuous over Ω^κ.

Equation (1.5) assumes that the integration over each element is independent; this *compactness* results from allowing piecewise discontinuous solutions. The third term in (1.5) is the integral of the conservative flux over the element boundary. This is the only term that involves communication with other elements. For hyperbolic problems, this communication is only between *neighboring elements* which share a common node (1-D), edge (2-D), or face (3-D).

The formulation presented in (1.5) only requires that we know the geometry of each element and the connectivity of a collection of elements. This allows for the use of either structured or unstructured mesh frameworks. Additionally, the elements which comprise the mesh can have curvilinear geometry. Define the mapping from physical space $\vec{x}$ to computational space $\vec{\xi}$ using

$$\vec{x} = \vec{x}(\vec{\xi}). \tag{1.6}$$

Section **??** provides the details on the metric terms that are introduced along with the form of the divergence, gradient, and curl under such a mapping. For simplicity, the computational domain is formed from tensor products of intervals $[-1,1]$ in each coordinate direction. This criteria restricts the elements to be logically segments (1-D), quadrilaterals (2-D), or hexahedrons (3-D). Under the mapping (1.6), the weak form (1.5) becomes

$$\int_{\Omega^\xi} (\tilde{s}_t^\kappa + \tilde{q}^\kappa)\phi \, d\Omega^\xi - \int_{\Omega^\xi} \tilde{f}^\kappa \cdot \nabla_\xi \phi \, d\Omega^\xi + \oint_{\partial\Omega^\xi} \phi \tilde{f}^{\kappa,*} \cdot \hat{n}^\xi \, dA^\xi = 0, \;\; \forall \phi \in \mathbb{C}^0(\Omega^\xi) \tag{1.7}$$

where $\tilde{s} = J^\kappa \vec{s}$, $\tilde{q} = J^\kappa \vec{q}$, $\tilde{f} = (J^\kappa \vec{a}^{\kappa,i} \cdot \vec{f})\hat{a}^{\kappa,i}$, J^κ is the Jacobian of the transformation over the κ^{th} element, and $\vec{a}^{\kappa,i}$ are the contravariant basis vectors associated with the transformation of element κ (the $\hat{a}$ denotes a unit vector).

Given a conservative flux, a non-conservative source, internal element metrics, and global element connectivity, the discrete algorithm solves (1.7) by approximating the integrands with interpolants and the integrals by discrete quadratures. First, the prognostic solution, source term, conservative flux, and mapping are approximated by Lagrange interpolants of degree N.

$$\vec{s} \approx I^N(\vec{s}) = \sum_{i,j,k=0}^{N} \vec{S}_{i,j,k} l_i(\xi^1) l_j(\xi^2) l_k(\xi^3) \tag{1.8a}$$

$$\vec{q} \approx I^N(\vec{q}) = \sum_{i,j,k=0}^{N} \vec{Q}_{i,j,k} l_i(\xi^1) l_j(\xi^2) l_k(\xi^3) \tag{1.8b}$$

$$\vec{f} \approx I^N(\vec{f}) = \sum_{i,j,k=0}^{N} \vec{F}_{i,j,k} l_i(\xi^1) l_j(\xi^2) l_k(\xi^3) \tag{1.8c}$$

$$\vec{x} \approx I^N(\vec{x}) = \sum_{i,j,k=0}^{N} \vec{X}_{i,j,k} l_i(\xi^1) l_j(\xi^2) l_k(\xi^3) \tag{1.8d}$$

An additional simplification is to approximate the product of interpolants by an interpolant

of degree N which incurs an additional aliasing error.

$$\tilde{s} \approx I^N(I^N(J)I^N(\vec{s})) = \tilde{S} = \sum_{i,j,k=0}^{N} (J\vec{S}_{i,j,k}) l_i(\xi^1) l_j(\xi^2) l_k(\xi^3) \tag{1.9a}$$

$$\tilde{q} \approx I^N(I^N(J)I^N(\vec{q})) = \tilde{Q} = \sum_{i,j,k=0}^{N} (J\vec{Q})_{i,j,k} l_i(\xi^1) l_j(\xi^2) l_k(\xi^3) \tag{1.9b}$$

$$\tilde{f}_{(i,n)} \approx I^N(I^N(Ja^i_{(n)})I^N(\vec{f})) = \tilde{F}_{(i,n)} = \sum_{i,j,k=0}^{N} (Ja^i_{(n)}\vec{F}_{(n)})_{i,j,k} l_i(\xi^1) l_j(\xi^2) l_k(\xi^3) \tag{1.9c}$$

Last, the space of test functions ($\mathbb{C}_0$) is approximated by the $\mathbb{P}^N$, the space of polynomials of degree N. Thus, the test function ϕ is replaced by each of the Lagrange interpolating polynomials,

$$\phi_{m,n,p} = l_m(\xi^1) l_n(\xi^2) l_p(\xi^3) \tag{1.10}$$

With this, equation (1.7) becomes

$$\int_{\Omega^\xi} (\tilde{S}^\kappa_t + \tilde{Q}^\kappa)\phi_{m,n,p}\, d\Omega^\xi - \int_{\Omega^\xi} \tilde{F}^\kappa \cdot \nabla_\xi \phi_{m,n,p}\, d\Omega^\xi + \oint_{\partial\Omega^\xi} \phi_{m,n,p} \tilde{F}^{\kappa,*} \cdot \hat{n}^\xi\, dA^\xi = 0, \quad m,n,p = 0,1,...,N \tag{1.11}$$

The final step is to replace the integrals in (1.11) with discrete quadrature. For this we use the Legendre-Gauss quadrature, which yields exact integration for each term in (1.11). Additionally, the interpolation nodes are specified as the Legendre-Gauss nodes, which simplifies the integration. The algorithm is now presented in one, two, and three dimesnions.

Algorithm in One-Dimensions

In one-dimension, the global physical domain is partitioned into a set of line segments. The first term in (1.11) becomes

$$\int_{\Omega^\xi} \tilde{S}^\kappa_t \phi_m\, d\Omega^\xi = \sum_{\alpha=0}^{N} \left[\sum_{i=0}^{N} \left((\tilde{S}^\kappa_i)_t l_i(\xi_\alpha) \right) l_m(\xi_\alpha) \omega_\alpha \right] \tag{1.12}$$

where the ξ_α are the Legendre-Gauss nodes and the ω_m are the Legendre-Gauss weights. Since the interpolation nodes are equivalent to the quadrature nodes, (1.12) is greatly simplified by the use of the Kronecker-delta property of Lagrange interpolating polynomials,

$$l_i(\xi_m) = \delta_{i,m}. \tag{1.13}$$

This yields (1.12) as

$$\int_{\Omega^\xi} \tilde{S}_t^\kappa \phi_\alpha \, d\Omega^\xi = (\tilde{S}_\alpha^\kappa)_t \omega_\alpha. \tag{1.14}$$

Similarly,

$$\int_{\Omega^\xi} \tilde{Q}^\kappa \phi_\alpha \, d\Omega^\xi = \tilde{Q}_\alpha^\kappa \omega_\alpha. \tag{1.15}$$

The third term in (1.11) becomes

$$\int_{\Omega^\xi} \tilde{F}^\kappa \cdot \nabla_{\vec{\xi}} \phi_m \, d\Omega^\xi = \sum_{i=0}^{N} \tilde{F}_i^\kappa l'_m(\xi_i) \omega_i. \tag{1.16}$$

Lastly, the boundary flux becomes

$$\oint_{\partial\Omega^\xi} \phi_\alpha \tilde{F}^{\kappa,*} \cdot \hat{n}^\xi \, dA^\xi = \tilde{F}^{\kappa,*}(1) l_\alpha(1) - \tilde{F}^{\kappa,*}(-1) l_\alpha(-1). \tag{1.17}$$

Thus, the spatially discretized system can be written

$$\left(J_m \vec{S}_m\right)_t = -\left[\sum_{i=0}^{N} \hat{D}_{m,i} \tilde{F}_i + \left(\frac{l_m(1)}{w_m} \tilde{F}^*(1) - \frac{l_m(-1)}{w_m} \tilde{F}^*(-1)\right)\right] + J_m \vec{Q}_m; \quad m = 0, 1, ..., N, \tag{1.18}$$

where

$$\hat{D}_{m,i} = -\frac{l'_m(\xi_i) w_i}{w_m^{\xi^1}} \tag{1.19}$$

is the DG-Derivative matrix.

Up to this point, the treatment of the boundary flux has been relatively vague. In practice, the boundary flux is calculated using an approximate Riemman solver. This uses the solution state from the neighboring elements to approximate the flux across the shared boundary. The specification of the Riemman solver depends on the PDE system. Because of this, a discussion of the boundary flux is delayed until specific PDE systems are discussed.

Algorithm in Two-Dimensions

$$\begin{aligned}\left(J_{m,n} \vec{S}_{m,n}\right)_t = &-\left[\sum_{i=0}^{N} \hat{D}_{m,i}^{(\xi^1)} \tilde{F}_{i,n}^{(\xi^1)} + \left(\frac{l_m(1)}{w_m^{(\xi^1)}} \tilde{F}^*(1, \xi_n^2) - \frac{l_m(-1)}{w_m^{(\xi^1)}} \tilde{F}^*(-1, \xi_n^2)\right) \cdot \hat{\xi^1}\right] \\ &-\left[\sum_{j=0}^{N} \hat{D}_{n,j}^{(\xi^2)} \tilde{F}_{m,j}^{(\xi^2)} + \left(\frac{l_n(1)}{w_n^{(\xi^2)}} \tilde{F}^*(\xi_m^1, 1) - \frac{l_n(-1)}{w_n^{(\xi^2)}} \tilde{F}^*(\xi_m^1, -1)\right) \cdot \hat{\xi^2}\right] \\ &+ J_{m,n} \vec{Q}_{m,n}; \quad m, n = 0, 1, ..., N\end{aligned} \tag{1.20}$$

Computing the divergence of the conservative flux in this framework can be viewed as calculating a sequence of derivatives in each computational direction. Two steps are required to compute the derivative in each direction. The first is an internal matrix-vector multiply, and the second is computing the weighted Riemann fluxes at the element boundaries. The latter is the only step which requires element-to-element communication.

The discretization presented here distinguishes itself from previous shallow water models presented in the literature. The models of Iskandarani et al. (1995) (and Ma1993) make use of the Gauss-Lobatto quadrature points, which were shown in Kopriva (2006) to not satisfy the discrete metric identities and lead to the introduction of spurious modes associated with a curvilinear mesh. Here, the more accurate Gauss quadrature is used. The increased accuracy and satisfication of the discrete metric identities increases the algorithm cost; since the Gauss quadrature points do not include the element boundaries, an additional step (with $\mathcal{O}(N^2)$ operations per element) is needed to interpolate the solution to the element boundaries before computing Riemann fluxes. An adpative filtering procedure, similar to **?** is used for maintaining stability when marginally resolved modes violate energy conservation. In this way, the filter lowers the accuracy of the method only in locations where the solution is marginally resolved; the gain in performing the filtering is that numerical stability is maintained. This filtering procedure is different than Taylor et al. (1997) where the filter is applied everywhere at each time step. In the conservative form of the shallow water equations, care was taken in the computation of the bathymetry gradient to guarantee that a motionless fluid remains motionless to machine precision.

Bibliography

M. Iskandarani, D.B. Haidvogel, and J.P. Boyd. A staggered spectral element model with application to the oceanic shallow water equations. *International Journal For Numerical Methods in Fluids*, 20:393–414, 1995.

D.A. Kopriva. Metric identities and the discontinuous galerkin spectral element method on curvilinear meshes. *Journal of Scientific Computing*, 26-3:301–327, 2006.

M. Taylor, J. Tribbia, and M. Iskandarani. The spectral element method for the shallow water equations on the sphere. 130:92–108, 1997.