

An Introduction to Spectral Element Method

Kenn K. Q. Zhang, Ph.D.

Limiton Software, Inc.

Shaoxing, Zhejiang 312000, China

1 Weighted Residual

Before divulging the power of spectral element method and the reasons behind, let us take a brief tour in an exemplary application field of numerical simulation, Computational Fluid Dynamics (CFD). Fluid mechanics permeates everywhere, in aeronautics, in ship building and offshore structures, in nuclear reactor cooling, in mechanical lubrication, in combustion chamber, in polymer forming, in solidification and material science in general, in aerosol and electro-hydrodynamics, in plasmas physics, in human tissues and blood vessels, and in fluidic micro-channels, just to name a few. Besides the descriptive approach, the means to study fluid mechanics are theoretical, experimental, and computational. The multi-variable, multi-dimension, complex geometry, viscosity, and nonlinearity associated with fluid equations preclude theoretical approach to solve any problem but under simplest idealized circumstances. The theory is useful primarily during the equation system formation, which both the experimental and computational means depend on, and useful in some approximate analytical methods [17], which occasionally can manage to capture some main features of the flows with omission of details. To solve practical problems, one has to count on experimental and computational means. While experimental approach prevails in fields such as biology and chemistry, the situation in fluid mechanics is somewhat different, in that fluid motion can be reliably quantified by a set of equations. computational fluid dynamics edges out experiments in several respects.

First of all, CFD predicts flows, and this may be extremely hard for experiments. For instance, weather is forecasted based on software calculations which use measured data as inputs. Experiments would be virtually always behind schedule if being used in this business. Experiments could predict, but are well versed only for something repeatable and do very poorly for real-time situations. Unfortunately for many flows, including that in weather forecasting, the repeatability is virtually always not the case but the simultaneity is virtually always expected from the predicting method. Secondly, CFD has a much closer connection to theory. CFD shares to great extent with theoretical

approach during equation system formation, and it separates from theory during system solving stage. So it is sometimes regarded as an arm stretch of theories. In contrast, experiments rather proceed in their own ways, with less dependency on theories. Hence, CFD can better reveal the underlying mechanism. Thirdly, experiments cannot be applied to certain circumstances. For instance, the magnetohydrodynamics around the sun can be simulated but can not be experimented (it should be stressed that observation is the common means in astrophysics but observation only postdicts), and the flow-disturbing support for the aircraft prototype in wind tunnel has be present in experiments but can be removed in CFD simulations. Finally, CFD is much less expensive. The development of a CFD software is expensive and time consuming, but once developed it can be re-used in multiple circumstances and on multiple computers. Also, when solving a new problem an experimental setup typically takes much longer time than CFD. Therefore, in recent years the demand for CFD has been displaying a monotonic growth and CFD has been recognized as an essential tool in many industrial sectors.

Symbolically, a differential equation system can be written as

$$L(u) = 0 \tag{1}$$

where L represents any operator and u represents the dependent variable in exact sense. If the exact u (defined at all points in the domain) is replaced by an approximate u_i (defined at selected points), equation (1) is no longer satisfied. Hence, the original system is generalized to the integral form of weighted residual (weak form)

$$\int_{\Omega} wL(u_i)d\Omega = 0 \tag{2}$$

where w represents a weight function, and $L(u_i)$ is the residual. Then the processing of finding the solution is transformed into the process of minimizing the residual. Dependent on how the weight function is selected and how the exact u is replaced by u_i , various numerical methods can be generated.

In Eq. (2), if the exact u is replaced by u_i at a set of finite number of points in the space (this is actually equivalent to expressing the exact solution in some specific type of polynomial), and the weight function is selected as the Dirac delta shift function δ_{ij} , then the integral Eq. (2) reduces to

$$L(u_j) = 0$$

This is the Finite Difference Method (FDM). Implicit-in-space FDM is called compact finite difference method [30, Zhang *et al.* 2006]. Straightforward in regular geometry, FDM also features easy implementation for high-order schemes and narrow bandwidth of matrices. FDM can be implemented on structured grid for problems with modest complex geometry. This is accomplished through a global mapping which transforms an irregular geometry into a regular one, and through boundary fitted

coordinates [26, 27]. As a tradeoff, a simple equation in complicated geometry is transformed into a complicated equation in simple geometry. FDM is employed in a popular software in electromagnetics, FDTD (Finite Difference Time Domain).

In Eq. (2), the exact u can be replaced by $u_i\psi_i$, where ψ_i is the Lagrangian interpolation function and u_i is the coefficient. Further, if the weight function is also the Lagrangian polynomial, then it leads to the widely known Finite Element Method (FEM). The local geometric mapping, which employs Lagrangian polynomials, was co-born with FEM. The whole computational domain is structurelessly discretized into a large quantities of little irregular elements, called unstructured mesh. A local geometric mapping transforms these irregular elements into regular ones, on which Gauss Legendre quadrature (numerical integration) is carried out. FEM is adopted in software such as ANSYS CFX, Comsol, Adina, and ANSYS HFSS (High Frequency Structure Simulator, the most popular software in computational electromagnetics). The versatility of unstructured mesh and local geometric mapping makes FEM the dominant force in the field.

In Eq. (2), if the exact u is replaced by u_i as in FDM, but the weight function is selected as unity, then it leads to Finite Volume Method (FVM). FVM is a cell-based (locally) conservative method, in contrast to point-based FDM which may not be conservative. After evaluation of cell surface integrals, FVM turns out to be very close to FDM in the sense of the discrete systems. The most popular FVM grid on regular geometry is clearly the MAC staggered grid [11, 21], which is robust and conservative. Lattice grid was put forth [29, Zhang *et al.* 2006] as another conservative grid. For complex geometry, a variety range of grids/meshes are used [13, 19, for example], however, the MAC staggered grid is hardly seen. The complex geometry may be handled as in Immersed Boundary Method on regular grid [5, 23], handled via global mapping on structured grid [1, for example], handled locally on unstructured grid as in [13, 19], or handled as in the local geometric mapping approach of FEM. Discontinuities can be more easily manipulated in FVM, due to local conservation and inter-element flux control. FVM prevails in simulation software for convection dominant hyperbolic systems, with ANSYS Fluent and OpenFoam as towering examples.

In Eq. (2), if the interpolation function is spectral polynomials (such as Fourier, Legendre, or Chebyshev) and if Eq. (2) is imposed on the overall domain, then it leads to (single-domain) Spectral Methods [3, 4]. The order of polynomials in single domain spectral method is $\mathcal{O}(100)$. Spectral methods feature high accuracy, owing to the completeness and orthogonality properties of spectral functions. Single-domain Fourier spectral is currently still the most widely used method in turbulence simulation. However, (single-domain) spectral methods suffer from severe inconvenience in handling complex geometry, though it is partially doable with the global mapping as in FDM. Spectral methods also suffer from the excessive high order of polynomials, which deteriorates the nature of discrete

systems. One consequence is, due to a large range of eigenvalues the system needs more iterations to converge. The more severe consequence is, for general problems not applicable with Fourier spectral the time step size must be tiny, especially in diffusion dominant problems. Single-domain Fourier spectral method has been applied to American meteorological software for global climate forecast.

In Eq. (2), if the interpolation function is spectral polynomial and if Eq. (2) is imposed on each element, then it leads to continuous Spectral Element Method (SEM) [9, 15, 8]. Equipped with the same local geometric mapping, SEM overwhelmingly outperforms single-domain spectral method in complex geometry handling. The order of polynomials in SEM is $\mathcal{O}(10)$, so that the discrete system is bettered natured and the restriction on time step sizes is greatly relaxed. A continuous Spectral Element Method with inter-element flux control is named Discontinuous Spectral Element Method (DSEM) [16, 24, 14, 7, 10], which excels in local conservation, discontinuity, and parallelization handling. At time being, DSEM is the distant front runner of all numerical methods for continuum.

In Eq. (2), if the Lagrangian polynomial is retained as the interpolation function, but the fundamental solution to the operator L is selected as the weight function, then the weak form originally imposed on the interior of the domain is converted into integrals on the overall boundary. This leads to Boundary Element Method (BEM) [2, 25]. Different from all previous methods, the primary working place of BEM is the boundary of the domain, not the domain itself. Thus, BEM can be regarded as an indirect domain solver, while above mentioned methods belong to the category of direct solvers. Owing to the reduction of dimensionality, BEM gains some efficiency in some situations. However, BEM formulations typically give rise to fully populated matrices. This means that the computational time will tend to grow according to the square of the problem size, a severe disadvantage compared with the linear growth in direct domain solvers. The merit of dimensionality reduction is countervailed by the square growth in three-dimensional problems. The second severe disadvantage of BEM is its limitation to linear operators, whose Green's could be found. To handle nonlinearity, BEM has to be combined with a direct domain solver [28, for example], consequently forfeiting its main point. Fortunately, physics is rich in linear operators and sometimes two-dimensional models are sufficient to reveal the underpinning principles, so that BEM can find decent applications in this business. In addition, BEM achieves high accuracy for problems where unknowns on boundaries are what exactly to seek, and it enjoys simplicity in moving boundary problems. BEM-based electromagnetics simulation software has been developed.

In Eq. (2), if spectral polynomial is used as the interpolation function, and the fundamental solution to the operator L is selected as weight function, further, if the fundamental solution (such as $\frac{1}{r}$ for Laplace operator) is expanded in spectral polynomials, then it leads to Spectral Boundary Element Method (SBEM) [22, 20, 12, 18]. SBEM is the least investigated virginland. Up to year 2011,

only single-digit number of journal publications on this topic can be retrieved. We have learned that one of two major issues with BEM is dense matrix. Owing to orthogonality of spectral polynomial, there exists a possibility to make SBEM matrix sparse. Then, for large scale computation the reduction of dimensionality might reduce the size of problem by 1000 times, an astonishing speedup. In future, in case Discontinuous Spectral Element Method got a rival, it must be Spectral Boundary Element Method.

2 Chebyshev Collocation Spectral Method

Now let us have some flavor of spectral-based methods. The popular interpolation functions for spectral methods are Fourier, Legendre, and Chebyshev. The weighted residual integral can be discretized in an exact fitting manner or in best fitting. The weighted residual integral can evolve in the physical space or in transform space. Then, we may possibly have 12 combinations. Fourier has two major advantages. The quadrature points for fourier are evenly spaced, hence the time step restriction is the same as traditional non-spectral methods such as FDM. Derivatives of any order of fourier function stay in the same direction, hence orthogonality is in the simplest form. Unfortunately, fourier demands periodicity of the solution field. Therefore, fourier is ruled out for general-purpose computation. Both Legendre and Chebyshev are applicable to any type of problems, however, as a relative to fourier function, Chebyshev approach can utilize Fast Fourier Transform (FFT). Even in SEM where polynomial order is $\mathcal{O}(10)$, FFT runs faster than the matrix product necessary in Legendre approach. Therefore, Chebyshev is selected as the interpolation polynomial. The canonical spectral method should process primary computation in transform space, and it is named ‘‘Galerkin’’ in this article. Unfortunately, the nonlinearity renders a convolution in physical space necessary. Hence, it is pointless to take ‘‘Galerkin’’, and we instead shall simply do the majority work in the physical space, as in FEM, FDM, FVM, and BEM. Finally, exact fitting or best fitting? An analysis is omitted in this introductory article. ‘‘Chebyshev Collocation Spectral’’ (CCS) implies the exact fitting approach, and CCS is adopted here for its simplicity and maturity.

Consider a heat equation on $(-1, 1)$

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \quad (3)$$

with boundary conditions and initial data

$$u(-1, t) = u(1, t) = 0, \quad u(x, 0) = \sin(\pi x) \quad (4)$$

We consider the weighted integral equation

$$\int_{\Omega} \left(\frac{\partial u^N}{\partial t} - \frac{\partial^2 u^N}{\partial x^2} \right) \delta(x - x_j) dx = 0, \quad j = 1, 2, \dots, N - 1$$

where $u^N(x, t)$ is an approximation to $u(x, t)$ and x_j are a set of *collocation points* on $[-1, 1]$. The weighted integral equation is equivalent to

$$\left(\frac{\partial u^N}{\partial t} - \frac{\partial^2 u^N}{\partial x^2} \right) \Big|_{x_j} = 0, \quad j = 1, 2, \dots, N - 1 \quad (5)$$

that is, the approximate differential equation is (strongly) imposed at a set of collocation points in the physical space. For convenience and accuracy Gauss quadrature points are selected as collocation points and we seek the evolution of the approximate solution $u(x_j, t)$, where the superscript N has been dropped. Let the approximate solution expanded with Chebyshev polynomials $T_k(x)$

$$u(x, t) = \sum_{k=0}^N \tilde{u}_k(t) T_k(x)$$

where $\tilde{u}_k(t)$ is mapped from $u(x_j, t)$ in physical space. Then, in the merit of Chebyshev polynomials and Gauss quadrature, it can be shown

$$\frac{\partial^2 u}{\partial x^2} \Big|_{x_j} = \sum_{l=0}^N D_{jl}^2 u(x_l, t)$$

where D_{jl}^2 is a matrix associated with geometric quantities only. As a consequence, equation (5) reduces to an ordinary differential equation

$$\frac{du(x_j, t)}{dt} = \sum_{l=0}^N D_{jl}^2 u(x_l, t)$$

which can be marched in time with initial data. Boundary conditions are imposed during every time step on $u(x_0, t)$ and $u(x_N, t)$. Some good references on spectral methods are [4] and [6].

2.1 Chebyshev Polynomials $T_k(x)$

Chebyshev polynomials (of the first kind) $T_k(x)$ are nothing but cosine functions after change of independent variable

$$\begin{aligned} T_k(x) &= \cos k\theta, & k &= 0, 1, 2, \dots \\ \cos \theta &= x \end{aligned}$$

where $\theta \in [0, \pi]$ and $x \in [-1, 1]$. The first several $T_k(x)$ are

$$\begin{aligned} T_0(x) &= 1 & T_1(x) &= x \\ T_2(x) &= 2x^2 - 1 & T_3(x) &= 4x^3 - 3x \\ T_4(x) &= 8x^4 - 8x^2 + 1 & T_5(x) &= 16x^5 - 20x^3 + 5x \end{aligned}$$

Some properties of $T_k(x)$ are

$$T_k(1) = 1 \quad |T_k(x)| \leq 1 \quad |T'_k(1)| = k^2 \quad |T'_k(x)| \leq k^2$$

All $T_k(x)$ can be worked out by recursive relation

$$T_{k+1} = 2xT_k - T_{k-1}$$

together with T_0 and T_1 . Another type of recursive relation is

$$(1 - x^2)T'_k = -kxT_k + kT_{k-1}$$

which is valid for $k \geq 1$. A more important variant of the right above relation is

$$2T_k = \frac{1}{k+1}T'_{k+1} - \frac{1}{k-1}T'_{k-1} \quad (6)$$

valid for $k > 1$.

The general eigenvalue problem for second-order linear ordinary differential equations can be presented in the self-adjoint form

$$\frac{d}{dx} \left(p(x) \frac{du(x)}{dx} \right) + q(x)u(x) + \lambda w(x)u(x) = 0$$

where λ is the eigenvalue and $w(x)$ is the weight function. A special case is the simple harmonic oscillator

$$T''_k(\theta) + k^2 T_k(\theta) = 0$$

which, after a change of variable $x = \cos \theta$, can be converted into another eigenvalue problem on $(-1, 1)$

$$\left(\sqrt{1-x^2} T'_k \right)' + \frac{k^2}{\sqrt{1-x^2}} T_k = 0$$

Chebyshev polynomials $T_k(x)$ are exactly the solutions to the right above equation, where the weight function $w(x) = (1-x^2)^{-1/2}$. Chebyshev polynomials are complete orthogonal polynomials on $(-1, 1)$ satisfying orthogonality relation

$$\langle T_k | T_l \rangle_w = \int_{-1}^1 T_k(x) T_l(x) w(x) dx = \frac{\pi}{2} c_k \delta_{kl} \quad (7)$$

where

$$c_k = \begin{cases} 2, & k = 0 \\ 1, & k \geq 1 \end{cases}$$

As a reduced case,

$$\|T_k\|_w^2 = \int_{-1}^1 T_k^2(x) w(x) dx = \frac{\pi}{2} c_k$$

Derivatives of Chebyshev polynomials have a minor orthogonal relation

$$\int_{-1}^1 T'_k(x) T'_l(x) (1-x^2)^{1/2} dx = 0, \quad k \neq l$$

Using recursive relation Eq. (6) and orthogonality property Eq. (7) it can be shown

$$\langle T'_k | T_l \rangle_w = \begin{cases} k\pi & k > l, \quad k+l \text{ odd} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

2.2 Chebyshev Interpolation

A function $u(x)$ in physical space can be expanded in *Chebyshev series*

$$u(x) = \sum_{k=0}^{\infty} \hat{u}_k T_k(x) \quad (9)$$

where *Chebyshev coefficients* in transform space

$$\hat{u}_k = \frac{2}{\pi c_k} \int_{-1}^1 u(x) T_k(x) w(x) dx \quad (10)$$

can be obtained through orthogonality relation Eq. (7). The conversion, equation (10), from function $u(x)$ in physical space into coefficients \hat{u}_k in transform space is named *Chebyshev transform*; the conversion, equation (9), from \hat{u}_k in transform space back to $u(x)$ in physical space is arguably named *inverse Chebyshev transform*. The *truncated Chebyshev series* of $u(x)$ refers to

$$P_N u = \sum_{k=0}^N \hat{u}_k T_k(x) \quad (11)$$

To evaluate integrals numerically a set of *quadrature points* x_j in physical space as well as corresponding weights W_j are selected so that

$$\int_{-1}^1 p(x) w(x) dx = \sum_{j=0}^N p(x_j) W_j \quad \forall p(x) \in \mathcal{P}_{2N-1}$$

The quadrature points and weights for *Chebyshev Gauss-Lobatto integration* are

$$x_j = \cos \frac{\pi j}{N}, \quad W_j = \frac{\pi}{N \tilde{c}_j} \quad (12)$$

where

$$\tilde{c}_j = \begin{cases} 2, & j = 0, N \\ 1, & j = 1, 2, \dots, N-1 \end{cases}$$

At these quadrature points $T_k(x_j) = \cos \frac{\pi j k}{N}$.

Chebyshev transform Eq. (10) can be approximated by *discrete Chebyshev transform*

$$\tilde{u}_k = \sum_{j=0}^N \frac{2}{N \tilde{c}_j \tilde{c}_k} u(x_j) \cos \frac{\pi j k}{N} \quad (13)$$

where \tilde{u}_k is named *discrete Chebyshev coefficients*. Equation (13) can be handled with *Fast Fourier Transform* at cost of $\mathcal{O}(N \log_2 N)$ operations. *Discrete inverse Chebyshev transform*

$$u(x_j) = \sum_{k=0}^N \tilde{u}_k \cos \frac{\pi j k}{N} \quad (14)$$

can be verified through substitution of \tilde{u}_k by Eq. (13) and orthogonality properties of cosine/exponential functions.

Equipped with discrete Chebyshev coefficients, equation (11) can be further approximated by *Chebyshev interpolant* in transform space

$$I_N u = \sum_{k=0}^N \tilde{u}_k T_k(x) \quad (15)$$

which is also named *discrete Chebyshev series*. By substituting discrete Chebyshev coefficients Eq. (13) into Eq. (15) we obtain Chebyshev interpolant in physical space

$$I_N u = \sum_{j=0}^N u(x_j) \psi_j(x) \quad (16)$$

where

$$\psi_j(x) = \frac{(-1)^{j+1}(1-x^2)T'_N(x)}{\tilde{c}_j N^2(x-x_j)} \quad (17)$$

is named *Chebyshev Lagrangian polynomials*. Obviously,

$$I_N u(x_j) = u(x_j)$$

in the merit of selected quadrature.

2.3 Chebyshev Interpolation Derivatives

Since differentiation and summation are commutable under most circumstances, equation (9) can be used to carry out the derivative

$$u'(x) = \sum_{k=0}^{\infty} \hat{u}_k T'_k(x)$$

Unfortunately, the primary orthogonal property Eq. (7) is in terms of Chebyshev polynomials. Each derivative of Chebyshev polynomial can be expanded in Chebyshev polynomials, and this is equivalent to expand the derivative of the function directly in Chebyshev series

$$u'(x) = \sum_{k=0}^{\infty} \hat{u}'_k T_k(x) \quad (18)$$

The goal here is to express \hat{u}' in \hat{u} implicitly (i.e., to find a couple relation) or explicitly. Projecting the above two forms of $u'(x)$ onto Chebyshev “coordinates” we obtain

$$\left\langle \sum_{l=0}^{\infty} \hat{u}_l T'_l(x) \mid T_k(x) \right\rangle_w = \left\langle \sum_{l=0}^{\infty} \hat{u}'_l T_l(x) \mid T_k(x) \right\rangle_w$$

which leads to explicit expression of \hat{u}' in terms of \hat{u} and, more importantly, an recursive relation

$$c_k \hat{u}'_k = \hat{u}'_{k+2} + 2(k+1) \hat{u}_{k+1} \quad (19)$$

The explicit expression costs $\mathcal{O}(N^2)$ operations while the recursive relation Eq. (19) takes $\mathcal{O}(N)$ operations. Equation (19) can be extended to

$$c_k \hat{u}''_k = \hat{u}''_{k+2} + 2(k+1) \hat{u}'_{k+1}$$

Also, the explicit expressions of \hat{u}'' in terms of \hat{u}' or \hat{u} can be worked out. The complete version of recursive relation related to Chebyshev Gauss-Lobatto quadrature points is

$$\begin{cases} \tilde{u}'_{N+1} = 0 \\ \tilde{u}'_N = 0 \\ \tilde{c}_k \tilde{u}'_k = \tilde{u}'_{k+2} + 2(k+1)\tilde{u}'_{k+1} & k = N-1, N-2, \dots, 0 \end{cases} \quad (20)$$

Similarly,

$$\begin{cases} \tilde{u}''_{N+1} = 0 \\ \tilde{u}''_N = 0 \\ \tilde{c}_k \tilde{u}''_k = \tilde{u}''_{k+2} + 2(k+1)\tilde{u}''_{k+1} & k = N-1, N-2, \dots, 0 \end{cases} \quad (21)$$

The derivative of the function in physical space, u' , is approximated by *Chebyshev interpolation derivative* $(I_N u)'$ at best. Derivatives of the function in physical space at quadrature points

$$D_N u(x_j) \equiv (I_N u)'|_{x_j} = \sum_{k=0}^N \tilde{u}'_k T_k(x_j) \quad (22)$$

where \tilde{u}'_k are derivatives in transform space. Calculation of $D_N u(x_j)$ is as follows: $u(x_j)$ at quadrature points are converted into \tilde{u}_k with discrete Chebyshev transform Eq. (13), then \tilde{u}'_k are calculated with recursive relation Eq. (20), finally discrete inverse Chebyshev transform Eq. (22) is used. This procedure is named vector approach at cost of $\mathcal{O}(N \log_2 N)$ operations. With recursive relation Eq. (21) utilized, $D_N^2 u$ can be worked out similarly. The vector approach is efficient, however, in the differential equation the term involving spatial differentiation need to be treated time-explicitly.

Alternatively,

$$D_N u(x_j) = \sum_{l=0}^N u(x_l) \psi'_l(x_j) = \sum_{l=0}^N D_{jl} u(x_l) \quad (23)$$

where Chebyshev derivative matrix at quadrature points

$$D_{jl} = \begin{cases} \frac{\tilde{c}_j}{\tilde{c}_l} \frac{(-1)^{j+l}}{x_j - x_l}, & l \neq j, \\ -\frac{x_l}{2(1-x_l^2)}, & 1 \leq l = j \leq N-1 \\ \frac{2N^2+1}{6}, & l = j = 0 \\ -\frac{2N^2+1}{6}, & l = j = N \end{cases} \quad (24)$$

This alternative matrix approach costs $\mathcal{O}(N^2)$ operations, however, in the differential equation the term involving spatial differentiation can be treated time-implicitly. In spectral element methods where N is typically $\mathcal{O}(10)$, the matrix approach performs with similar efficiency as the vector approach. The explicit structure of D_{jl}^2 is also available, or simply $D_{jl}^2 = D_{jm} D_{ml}$.

Note that $(I_N u)' \neq P_N u'$ and interpolation does not commute with differentiation, and particularly for Chebyshev polynomials the truncation does not commute with differentiation. $(P_N u)'$

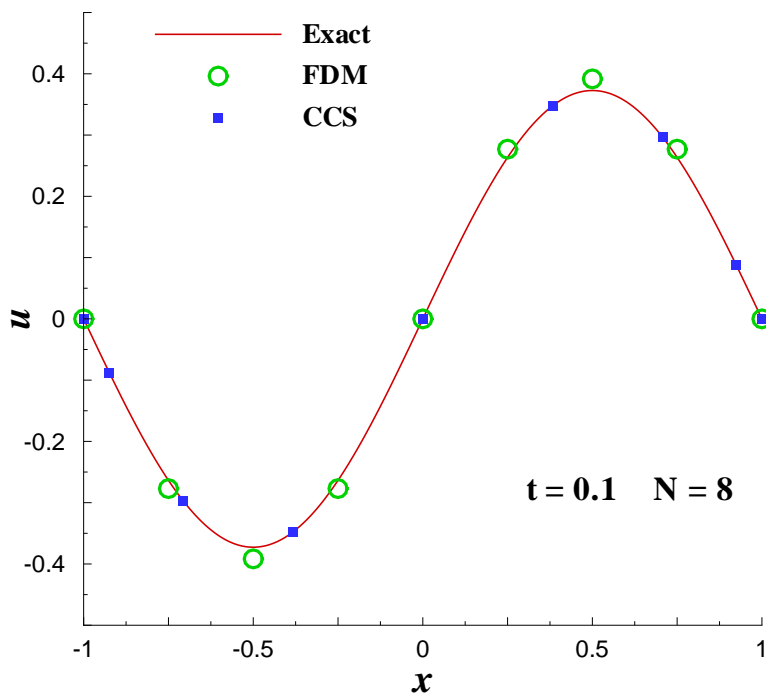


Figure 1: Accuracy comparison between Chebyshev Collocation Spectral (CCS) method and Finite Difference Method (FDM).

or $(I_N u)'$ are asymptotically worse approximation of u' than $P_{N-1} u'$ and $I_{N-1} u'$, respectively, for functions with finite regularity.

2.4 Performance Test

Now let us combine Chebyshev collocation spectral method (CCS) with 4th-order Runge-Kutta time scheme to solve the diffusion problem Eqs. (3) and (4), whose exact solution is $e^{-\pi^2 t} \sin(\pi x)$.

N	CCS (max error)	CCS (root mean square)	FDM (max error)	FDM (root mean square)
4	0.0569	0.0360	0.0766	0.0485
8	4.79×10^{-5}	2.66×10^{-5}	0.0190	0.0127
16	1.42×10^{-12}	7.41×10^{-13}	4.73×10^{-3}	3.25×10^{-3}
32	4.56×10^{-17}	1.30×10^{-14}	1.18×10^{-3}	8.23×10^{-4}

Table 1: Accuracy and convergence rate comparison between CCS and FDM, both against the exact solution.

Within each time marching step

$$\begin{aligned}
\mathbf{K}_1 &= \Delta t \mathbf{D}^2 \mathbf{U}^n \\
\mathbf{K}_2 &= \Delta t \mathbf{D}^2 \left(\mathbf{U}^n + \frac{1}{2} \mathbf{K}_1 \right) \\
\mathbf{K}_3 &= \Delta t \mathbf{D}^2 \left(\mathbf{U}^n + \frac{1}{2} \mathbf{K}_2 \right) \\
\mathbf{K}_4 &= \Delta t \mathbf{D}^2 \left(\mathbf{U}^n + \mathbf{K}_3 \right) \\
\mathbf{U}^{n+1} &= \mathbf{U}^n + \frac{1}{6} (\mathbf{K}_1 + 2\mathbf{K}_2 + 2\mathbf{K}_3 + \mathbf{K}_4)
\end{aligned}$$

where \mathbf{U} is the solution vector and $\mathbf{D}^2 = \mathbf{D}\mathbf{D}$ (\mathbf{D} is the matrix in Eq. (24)). Initial data \mathbf{U}^0 is given at all collocation points and at the end of each time step Dirichlet boundary conditions are applied to the first and last members of \mathbf{U}^{n+1} . The performance of the spectral method is compared with standard Finite Difference Method (FDM), based on the same number of grid points and the same time scheme. The accuracy of CCS over FDM is demonstrated in Fig. 1, where the grid-point numerical values with CCS stay on the exact curve accurately. Table 1 shows that the errors in FDM drop approximately at a rate of 4 (which tells the 2^{nd} -order accuracy); in contrast, the errors in CCS drop at a spectral rate.

The key of spectral method is Gauss quadrature. A warn should be issued immediately that we shall not jump on conclusion that SEM is an alias to FEM since both employ Gauss quadrature. The quadrature in FEM is merely used for numerical integration in isolated locus; in contrast, Gauss quadrature penetrates into the entire structure of spectral methods. At the end, the quadrature in CCS produces a set of points in physical space. So we may look at CCS in this way: it is a finite difference method with judicious distribution of points and weights. But this judiciousness is not trivial: it attributes the spectral convergence rate.

3 Spectral Element Method

It was reported that a numerical simulation of seconds of a nuclear explosion as well as the aftermath took some hyper fast computer system in California the first half of year 2006. This gives a basic idea of the magnitude and the time cost of scientific computing. Hence, numerical researchers and practitioners have an insatiable quest for speed. As for a general-purpose software, we ought to seek a method able to handle a large range of problems, capture finest details, and offer greatest speed. Although we shall admit that all aspects and directions of research and development in scientific computing have their own rights and they interact with each other, there is a priority. Numerical method is the foundation of simulation software. While there are dozens of equation systems and hundreds of variants and thousands of numerical problems, the number of general families of numerical methods is very limited. This tells that one numerical method has multiple interactions with structures above, which involve extensive programming. While most difficult part of scientific computing is not

the computer programming, the most time consuming part is. So, if the foundation of a fully loaded simulation software turns out inferior to some outside method, it runs into big trouble.

The past five decades witnessed the great success of finite element methods and finite volume methods in engineering applications. FEM performs well in relatively smooth elliptic and parabolic types of partial differential equations. Unfortunately, a large sets of problems involve sharp gradients which are increasingly drawing more attention. To meet the resolution requirements in regions of sharp gradients, one may use more lower-order elements, or use high order lagrangian polynomials. A growing number of evidence shows that the first approach either amounts computational cost too high, or fails to address the issue for problems where high-order quantities are exactly what to seek, such as in aeroacoustics and turbulence. The second approach is even more hopeless. First, high order lagrangian polynomials have a tendency to cluster together or look similar to each other; in other words, they are almost linearly dependent; that is, the resulting discrete system is virtually ill posed so that no solution could be found. Second, the bandwidth of the matrix is proportional to N^3 for 3-D case, where N for the order of lagrangian polynomial. This precludes serious applications of high-order lagrangian finite element methods. The qualified polynomials should dissimilate to each other so that every mode in the solution can be efficiently represented, and should result in a matrix with narrow bandwidth or even diagonalizable.

Complete (which means, every reasonable function can be expressed by these polynomials) orthogonal (which means, a weighted integral of two different functions vanishes) polynomials such like Chebyshev possesses these two vital properties. There are only two general categories of methods in solving partial differential equations analytically, the major category of series expansions and the minor category of integral transforms. Polynomials employed in series expansion are mostly the complete orthogonal polynomials, such like Fouries, Bessel, Legendre, Hermite, Laguerre, Gamma, etc. But no mathematical physicists use lagrangian polynomials to solve PDEs. Therefore, it is natural to extend these complete orthogonal polynomials to computation, by truncating higher order terms in series expansion. This kind of methods are called (single-domain) spectral methods. Concurrently spectral methods generally fall into two categories, Galerkin spectral method, where the primary unknowns are solved in the transform space (with possible evaluation of some quantities in the physical space in case of nonlinear problems, and this is called pseudo-spectral), and the collocation spectral method, where the primary unknowns are solved in the physical space and the transform space is used merely for evaluating derivatives.

Unfortunately, single-domain spectral methods have several serious shortcomings. Besides the inability in complex geometry handling, the excessively high-order polynomials brings in aliasing errors. De-aliasing techniques can be used to remove aliasing errors but it intertwines with convolution

for nonlinearity and implicit time schemes. Excessive high-order polynomials also incurs deterioration of system conditioning number and constrains time step size. Spectral element methods do away with all these issues. If in spectral element method the communications between two elements are realized only via the inter element boundary and domain points in one element are prohibited to directly interfere with points in another element, it leads to Discontinuous Spectral Element Method. Some benefits of discontinuous SEM (as versus continuous SEM) are as follows:

- Local conservation can be easily imposed via inter element flux. A locally conservative scheme is more robust and converges faster.
- Shock capturing schemes can be much more easily implemented, so that discontinuities or high gradients can be much better handled.
- Different polynomials can be assigned on different elements to ensure optimal effect.
- Mesh rezone can be fulfilled more easily, with less interference with neighborhood.
- Adaptive mesh can be more easily implemented. Adaptive mesh enhances efficiency and is particularly useful in simulation of problems with moving boundaries.
- Multigrid technique can be easily implemented. Multigrid technique can typically speed up a solver by 3 times.
- Parallelization can be more easily implemented, and performs much more effectively.

4 Why Spectral Element Method Much Faster?

Finally, we may want to ask an epicentral question: why is Spectral Element Method much faster than Finite Elements?

- Construction of discrete system accounts for 5% or less of computational time, and iteratively solving the discrete system accounts for 95% time. The process of iteration is the process of diagonalizing the discrete system. Thus, a numerical method producing a matrix with broad bandwidth is slow, and a method producing narrow bandwidth is fast. The bandwidth of Finite Element Method for three-dimensional problem is $\mathcal{O}(100)$, correspondingly it is $\mathcal{O}(10)$ or even $\mathcal{O}(1)$ for Spectral Element Method. This makes SEM one or two orders of magnitudes faster. This is due to the merit of orthogonality of spectral polynomials.

- SEM produces a discrete system with better conditioning number (the ratio of largest eigenvalue over the smallest one), thus it needs less iterations to converge. This may make SEM half order of magnitudes faster. This is due to merit of completeness of spectral polynomials.
- Suppose we want to solve a three-dimensional problem on parallel computers. In each processor handling the subdomain, let N be the number of discrete unknowns in one direction, so that the cost of local computation is $\mathcal{O}(k_1 N^3)$, where k_1 is number of iteration. A processor only needs to communicate with processors dealing with the neighborhood subdomains, so that the cost of communication is $\mathcal{O}(k_2 N^2)$, where k_2 is communication overhead coefficient. Typically, the order of discrete unknown N is $\mathcal{O}(100)$, k_1 is $\mathcal{O}(100)$ for time dependent problem, and k_2 is $\mathcal{O}(100)$. Therefore, bottleneck of parallel computing is not the communication issue most researchers take for granted. The bottleneck is: the good-natured iterative process in serial computation is ruined by the inter-processor iteration in parallel computation, which is similar to source-term iteration. Numerical experiments show that all FEM-based and FVM-based software do very poorly in parallelization. In contrast, Discontinuous SEM demonstrates impressive parallel performance scale up. The “discontinuous” in SEM is the reason behind. This merit alone could render SEM outperform FEM by one or two orders of magnitudes.

Owing to above remarkable properties, Spectral Element Method may proceed 100 times faster than Finite Element Method for moderate to large scale computation.

References

- [1] P. Ananthakrishnan. Radiation hydrodynamics of floating vertical cylinder in viscous fluid. *Journal of Engineering Mechanics*, 125:836–847, 1999.
- [2] P. K. Banerjee. *The Boundary Element Methods in Engineering*. McGraw-Hill College, 1994.
- [3] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods in Fluid Dynamics*. Springer-Verlag, 1987.
- [4] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods, Fundamentals in Single Domains*. Springer-Verlag, 2006.
- [5] R. K. C. Chan and R. L. Street. A computer study of finite-amplitude water waves. *Journal of Computational Physics*, 6:68–94, 1970.

- [6] D. L. Chopp. *Spectral Methods for Partial Differential Equations*. Web Lecture Notes, 2008. http://people.esam.northwestern.edu/chopp/course_notes/446-2.pdf.
- [7] B. Cockburn, G. E. M. Karniadakis, and C. W. Shu, editors. *Discontinuous Galerkin Methods*. Springer, 1999.
- [8] M. O. Deville, P. F. Fischer, and E. H. Mund. *High-Order Methods for Incompressible Fluid Flow*. Cambridge University Press, 2002.
- [9] P. F. Fischer. *Spectral Element Solution of Navier-Stokes Equations on High Performance Distributed-Memory Parallel Processors*. PhD thesis, Massachusetts Institute of Technology, 1989.
- [10] J. E. Flaherty, L. Krivodonova, J. F. Remacle, and M. S. Shephard. Aspects of Discontinuous Galerkin methods for hyperbolic conservation laws. *Finite Elements in Analysis and Design*, 38:889–908, 2002.
- [11] F. H. Harlow and J. F. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids*, 8:2182–2189, 1965.
- [12] L. R. Hill and T. N. Farris. Fast fourier transform of spectral boundary elements for transient heat conduction. *Int. J. Num. Meth. Heat Fluid Flow*, 5:813–827, 1995.
- [13] C. W. Hirt, A. A. Amsden, and J. L. Cook. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. *Journal of Computational Physics*, 14:227–253, 1974.
- [14] G. B. Jacobs. *Numerical simulation of two-phase turbulent compressible flows with a multidomain spectral method*. PhD thesis, University of Illinois at Chicago, 2003.
- [15] G. E. M. Karniadakis and S. J. Sherwin. *Spectral/hp Element Methods for CFD*. Oxford, 1999.
- [16] D. A. Kopriva and J. H. Kolas. A conservative staggered-grid Chebyshev multidomain method for compressible flows. *Journal of Computational Physics*, 125:244–261, 1996.
- [17] E. Lauga. *Slip, Swim, Mix, Pack: Fluid Mechanics at the Micron Scale*. PhD thesis, Harvard University, 2005.
- [18] G. P. Muldowney and J. J. L. Higdon. A spectral boundary element approach to three-dimensional Stokes flow. *Journal of Fluid Mechanics*, 298:167–192, 1995.
- [19] B. Niceno and E. Nobile. Numerical analysis of fluid flow and heat transfer in periodic wavy channels. *International Journal of Heat and Fluid Flow*, 22:156–167, 2001.

- [20] J. M. Occhialini, G. P. Muldowney, and J. J. L. Higdon. Boundary integral/spectral element approaches to the Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 15:1361–1381, 1992.
- [21] S. V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Hemisphere, Washington, DC, 1980.
- [22] A. P. Peirce, S. Spottiswoode, and J. A. L. Napier. The spectral boundary element method: a new window on boundary elements in rock mechanics. *Int. J. Rock Mech. Min. Sci. & Geomech. Abstr.*, 29(4):379–400, 1992.
- [23] C. S. Peskin. Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25:220, 1977.
- [24] D. Stanescu. *A multidomain spectral method for computational aeroacoustics*. PhD thesis, Concordia University, 1999.
- [25] C. Wang and B. C. Khoo. An indirect boundary element method for three-dimensional explosion bubbles. *Journal of Computational Physics*, 194:451–480, 2004.
- [26] R. W. Yeung and P. Ananthkrishnan. Oscillation of a floating body in a viscous fluid. *Journal of Engineering Mathematics*, 26:211–230, 1992.
- [27] R. W. Yeung and P. Ananthkrishnan. Viscosity and surface-tension effects on wave generation by a translating body. *Journal of Engineering Mathematics*, 32:257–280, 1997.
- [28] D. L. Young, J. T. Chang, and T. I. Eldho. Solution of three-dimensional unsteady external flow using a coupled arbitrary lagrangian fem-bem model. *Engineering Analysis with Boundary Elements*, 28:711–723, 2004.
- [29] K. K. Q. Zhang, B. Rovagnati, Z. Gao, W. J. Minkowycz, and F. Mashayek. An introduction to lattice grid. *Numerical Heat Transfer Part B-Fundamentals*, 51:415–431, 2007.
- [30] K. K. Q. Zhang, B. Shotorban, W. J. Minkowycz, and F. Mashayek. A compact finite difference method on staggered grid for Navier-Stokes flows. *International Journal for Numerical Methods in Fluids*, 52:867–881, 2006.