A time-spectral method with a novel subdomain scheme for initial-value problems

Kristoffer Lindvall, Jan Scheffel

Department of Fusion Plasma Physics, School of Electrical Engineering
KTH Royal Institute of Technology, Stockholm, Sweden

Abstract

We analyze a new subdomain scheme for a time-spectral method CBC-GWRM that applies to initial-value problems. Whilst spectral methods are commonplace for spatially dependent systems, finite difference schemes are typically applied for the temporal domain. The GWRM is a fully spectral method in that it spectrally decomposes all specified domains, including the temporal domain, with multivariate Chebyshev polynomials. The CBC-GWRM is benchmarked against two finite difference methods. For the linearised Burger equation the CBC-GWRM is $\sim 30\%$ faster than the semi implicit Crank-Nicolson method for a maximum error $\sim 10^{-5}$. For the forced waved equation the CBC-GWRM managed to average over the small time-scale in the entire temporal domain, whilst the explicit Lax-Wendroff method transitions from fast scale to the slow. The CBC-GWRM has also been applied to the linearised ideal MHD equations for a screw pinch equilibrium. The growth rate of the most unstable mode was computed to an accuracy $< 1\%$.

Keywords: Time-spectral, spectral, weighted residual methods, Chebyshev

1. Introduction

In the field of fusion plasma physics there is a sub-discipline devoted to the analysis of linearised magnetohydrodynamic equilibria and the evolution of unstable modes [2, 5, 7, 12, 17]. The analysis consists of linearising the governing magnetohydrodynamic equations for a fusion plasma equilibrium. The equilibrium is then perturbed and the different mode growth rates are calculated. These initial-value partial differential equations are routinely solved

numerically. This gives valuable insight into the dynamics of fusion plasmas and their evident limitations set by turbulence. However, the simulations are very time-consuming and computationally inefficient.

The original time-spectral method [13] is a fully spectral method that spectrally decomposes all domains including spatial, temporal, and parameter domains. The spatial domains are overlapped for the purpose of having a two-point contact. This subdomain method ensures spectral accuracy in the entire domain. Unfortunately, the resulting global matrix may become too large when dealing with physical relevant systems [3, 4, 8].

There are two popular methods for dealing with this issue. One such method is to use parallel algorithms that solve for large sparse matrices [10, 16], and another is to create independent spatial domains that can be solved locally with boundary-solvers that connect the subdomains [4, 6, 8, 11, 15].

This brings us to the current spatial subdomain scheme termed the Common-Boundary-Condition (CBC) Method. This method solves the "private" (independent subdomains) locally and separately, but the "boundary" equations that connect the subdomains are solved globally. This is strikingly similar to finite element/spectral methods where continuous subdomains are used. However the advantages of the CBC method over global spectral methods are that the amount of global equations have been decreased and that all of the computation is done in spectral space.

The paper is organized as follows. The first section briefly summarizes the generalized weighted residual method, which is the foundation of the time-spectral method. Section two lays out the new spatial subdomain scheme that has been implemented, which is then followed by benchmarks in section three. The benchmarks include the linearised 1D Burger equations, forced wave equation, and the linearised ideal MHD equations. Section four contains a discussion, including some observations, possible objections, and resolutions. Then the paper closes with a conclusion in section six.

2. Method

The time-spectral method is built upon the generalized weighted residual method GWRM. This is the basis of all schemes henceforth employed. A brief review of the GWRM is presented here. For a full and detailed presentation the reader is directed to the publication [13].

2.1. Weighted residual formulation

We start by assuming an arbitrary partial differential equation,

$$\frac{\partial \mathbf{u}}{\partial t} = D\mathbf{u} + f. \tag{1}$$

where D is an arbitrary linear or non-linear operator and $f(t, \mathbf{x}; \mathbf{p})$ is a force term. The solution $u(t, x; p)$ is approximated with a multivariate Chebyshev expansion series in time, space, and parameter space.

$$u(t, x; p) \approx \mathrm{U}(\tau, \xi; P) = \sum_{k=0}^{K}{}' \sum_{l=0}^{L}{}' \sum_{m=0}^{M}{}' a_{klm} T_k(\tau) T_l(\xi) T_m(P) \tag{2}$$

Since Chebyshev polynomials are orthogonal in the range $[-1, 1]$, a change of variables is performed; $\sigma = (z - A_z)/B_z$, $A_z = (z_1 - z_0)/2$, and $B_z = (z_1 + z_0)/2$. Here σ signifies the transformed variable and z is the physical variable. The weighted residual formulation is obtained by substituting (2)-(1), multiplying the residual with weight functions, and integrating over the entire domain. The result being

$$\int_{p_0}^{p_1} \int_{x_0}^{x_1} \int_{t_0}^{t_1} R T_q(\tau) T_r(\xi) T_s(P) \mathrm{w}_t \mathrm{w}_x \mathrm{w}_p dt dx dp = 0. \tag{3}$$

The residual has the form

$$R = u(t, x; p) - [u(t_0, x; p) + \int_{t_0}^{t_1} (Du + f) dt'], \tag{4}$$

and the weight function,

$$\mathrm{w}_z = (1 - \sigma^2)^{-1/2}. \tag{5}$$

The final form of the algebraic equations takes the form,

$$a_{qrs} = 2\delta_{q0} b_{rs} + A_{qrs} + F_{qrs} \tag{6}$$

This is a simple and general formula that contains the initial conditions b_{rs}, the linear/non-linear operator term A_{qrs}, and the force term F_{qrs}. Equation (6) can then be solved with a semi-implicit root solver (SIR) [14]. However, if the PDE was solved in the entire computational domain then the CPU time scales as $((K+1)(L+1)(M+1))^3$ and the memory as $((K+1)(L+1)(M+1))^2$. This is of course unacceptable.

2.2. Subdomain scheme

A remedy for high CPU times and memory requirements is introducing subdomains. The entire computational domain $D = \{(t, x) : [0, T], [0, L]\}$ is discretized into s overlapping spatial elements, see Figure 1. For example $s = 3$ would give a discretized domain $\Omega = \{\Omega_1 = [0, x_1], \Omega_2 = [x_1 - \varepsilon, x_2 + \varepsilon], \Omega_3 = [x_2 - \varepsilon, L]\}$, where ε is a small overlapping distance. This procedure allows us to use fewer Chebyshev modes in our ansatz for each subdomain. Also, by overlapping the subdomains we increase convergence rates because point-wise and gradient continuity is ensured.

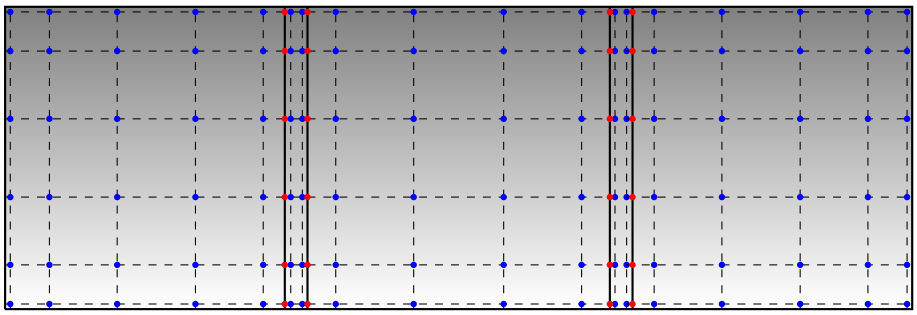


Figure 1: GWRM subdomains with overlapping region $dx{\sim}10^{-4} - 10^{-6}$ (*red dots*) and Chebyshev roots $x_k = \cos((2k - 1)\pi/2K), \quad k = 1 \ldots K$ (*blue dots*).

The system of equations we wish to solve include N_s spatial subdomains, N_e number of physical equations, K temporal modes, and L spatial modes. This results in $N = N_s N_e (K+1)(L+1)$ equations to solve for the coefficients in the ansatz 2. Can the global amount of equations be reduced in some way?

The idea now is to realize that only $\bar{x}^s_{BC} = \bar{\Phi}^s_{BC}$ equations need to be solved globally, where $\bar{x}^s_{BC}$ is the common boundary condition variables. The reason for this is that the equations in each individual subdomain $\bar{x}^s_P = \bar{\Phi}^s_P$ are only dependent on $\bar{x}^s_{BC}$. A consequence is that the private equations can be solved locally in each iteration level. Furthermore, we have the functional dependence $\bar{x}^s_{BC} = \bar{\Phi}^s_{BC}(s - 1, s, s + 1)$ which can be seen graphically in Figure 2.

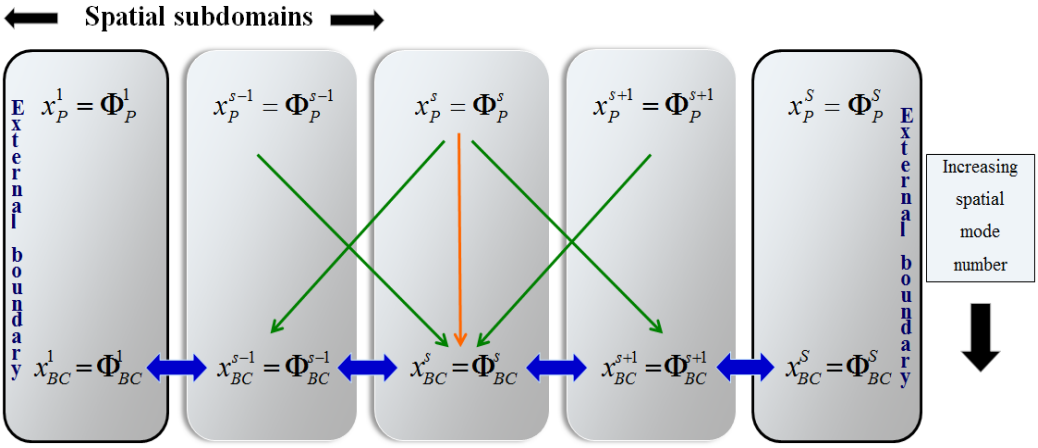


Figure 2: CBC subdomain schematic with algebraic equations. The private variables are explicitly dependent on the boundary variables in the given subdomain and implicitly on those of the neighbouring subdomains.

The reduced set of global equations are then solved iteratively by SIR [14]. SIR solves the algebraic set of equations by iterating the following equation,

$$\bar{x}_{BC}^{i+1} = [\bar{\bar{I}} + (\bar{\bar{R}}^i - \bar{\bar{I}})(\bar{\bar{J}}_{BC}^i)^{-1}](\bar{x}_{BC}^i - \bar{\varphi}_{BC}^i) + \bar{\varphi}_{BC}^i \equiv \bar{\Phi}_{BC}^i(\bar{x}_{BC}^i). \tag{7}$$

Thus the Jacobian $\bar{J}_{BC}^i$ needs to be computed,

$$J_{BCpq} = \frac{\partial(\bar{x}_{BC} - \bar{\varphi}_{BC})_p}{\partial \bar{x}_{BCq}} = \delta_{pq} - T_{pq}, \tag{8}$$

where δ_{pq} is the Dirac delta function and T_{pq} includes the explicit and implicit derivatives;

$$T_{pq} = \frac{\partial \varphi_{BCp}}{\partial x_{BCq}} + \sum_{\nu=s-1}^{\nu=s+1} \sum_{i=1}^{N_e N_p} \frac{\partial \varphi_{BCp}}{\partial x_{Pi}^{\nu}} \frac{\partial x_{Pi}^s}{\partial x_{BCq}^s}. \tag{9}$$

The indices p and q refer to common BC variables and the index i refers to the private variables. The first term is the explicit derivative. The second term shows that the common BC variables are indirectly dependent on the private variables in the neighbouring subdomains, hence the implicit derivatives. The index ν is introduced in the sum to neglect subdomains that are not directly influencing the current common BC variables.

A work flow of the Common Boundary Condition subdomain scheme can be seen in the box below.

CBC - subdomain scheme

Comment: Define the bounded computational domain $D = \{(t, x) : [t_0, t_1], [x_0, x_1]\}$, then discretize the spatial domain into overlapping subdomains N_s. The partial differential equations numbering N_e with initial conditions are then spectrally decomposed

Precompute: Total set of equations $x_i = \varphi_i(\bar{x})$, $i = 1..N$, where $N = N_e N_s (K + 1)(L + 1)$ is the total number of equations to solve. Create set of private equations $\bar{x}_p^s = \Phi_p^s(\bar{x}_p^s, \bar{x}_{BC}^s)$ and common BC equations $\bar{x}_{BC}^s = \Phi_{BC}^s(\bar{x}_{BC}^s, \bar{x}_{BC}^s)$ by extracting the appropriate indexes from the global set of equations.

Step 1: Initial common boundary condition values $\bar{X}_{BC}^s$ are substituted into $\bar{x}_p^s = \Phi_p^s(\bar{x}_p^s, \bar{X}_{BC}^s)$, so that only private variables are unknown. Call on SIR to solve private equations.

Step 2: Compute all common BC $\bar{\varphi}_{BC}$ expressions by solving global system of equations consisting of the private values from (1).

Step 3: Repeat step 1) with the common BC values from step 2.

Final: Update solution vector.

The explicit and implicit derivatives in Eq. 9 are straightforward to obtain analytically. However, the $\partial x_P^s / \partial x_{BC}^s$ coefficients in the sum require some attention. The first step is to create a vector of all private equations $f_i = z_i - \Phi_i(z)$, $i = 1 \ldots N_e N_p$, where z and $\Phi(z)$ only contain the private variables and equations from x and $\Phi(x)$. An implicit dependence of the common BC variables on the private variables is then established on the z variables, and then the partial derivatives are saved in matrix $F_{ij} = \partial f_i / \partial z_j$, $i = 1 \ldots N_e N_p$ and $j = 1 \ldots N_e N_{BC}$. This matrix then contains expressions that describe how the private variables depend on the common BC variables.

The implicit derivatives in F_{ij} are evaluated with current private values form current scheme iteration and the $b_j = [\partial x_P^s / \partial x_{BC}^s]_j$ variables are obtained by solving the linear algebraic system of equations $F_{ij} b_j = 0$.

3. Benchmark

The benchmark tests chosen are the linearised Burger equation, the Wave equation, and the linearised ideal MHD equations. Non-linear problems will be solved in a forthcoming paper. All CBC-GWRM simulations are carried out on one global temporal interval, i.e. there are no time steps/intervals.

3.1. Linearised 1D Burger equation

The first case deals with accuracy. The linearised 1D Burger equation is stated as follows,

$$\frac{\partial u}{\partial t} + b\frac{\partial u}{\partial x} = \kappa \frac{\partial^2 u}{\partial x^2}. \tag{10}$$

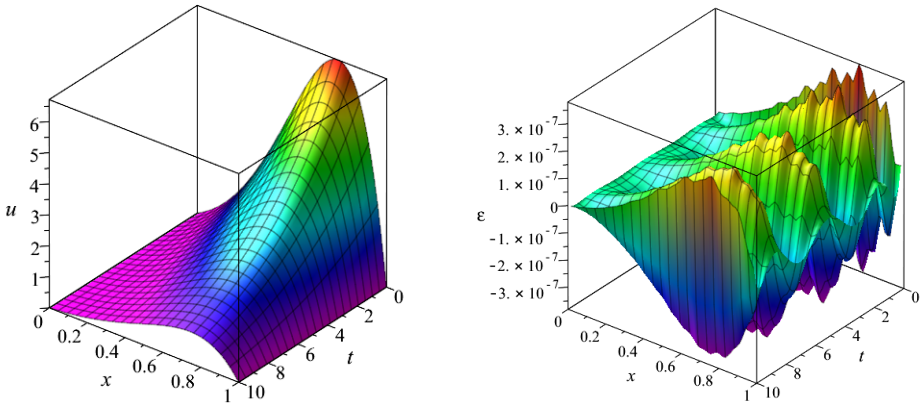
where κ is the viscosity and b is a constant. The initial condition and boundary conditions are,

$$u(0, x) = \sin(\pi x) e^{bx/2\kappa}, \tag{11}$$

$$u(t, 0) = u(t, 1) = 0. \tag{12}$$

This gives an exact solution that can be used for accurate benchmarks (Figures 3a,3b),

$$u(t, x) = \sin(\pi x) e^{bx/2\kappa - (b^2/4\kappa + \kappa\pi^2)t} \tag{13}$$



(a) Solution (b) Error

Figure 3: CBC-GWRM solution and error with $\kappa = 0.01$ and $b = 0.06$; $m = 7$, $n = 6$ and $N_s = 7$.

The CBC-GWRM has been compared with two finite-difference methods concerning CPU time vs. accuracy (see Figure 4), and memory consumption. The first is the explicit Lax-Wendroff method and the second is the implicit Crank-Nicolson method. For comparable accuracies of $\varepsilon \sim 10^{-3}$ the CBC-GWRM got a CPU runtime of 0.250 s and memory of 22.2 MB. This was achieved with the parameters $N_s = 2$, $m = 4$, and $n = 5$. The Lax-Wendroff method with $M = 340$ time steps and $N = 45$ spatial steps achieved a runtime of 0.940 s and memory 54.2 MB. The Crank-Nicolson method achieved 0.090 s and 20.2 MB with parameters $M = 60$ and $N = 80$. Thus we see that for the small accuracies the C-N method is approximately two times faster than CBC-GWRM and 10 time faster than L-W with comparable memory consumption to CBC-GWRM.

However, when the accuracy is increased to $\varepsilon \sim 10^{-5}$, the CBC-GWRM with $N_s = 3$, $m = 6$, and $n = 6$ achieves a CPU time of 0.733 s and memory of 38.2 MB. The C-N method achieved a CPU time of 1.06 s and memory of 43.1 MB, whilst the L-W method was unable to reach a comparable accuracy. The result being that CBC-GWRM is faster and more memory efficient than both FD-methods for higher accuracies. With the same parameters and accuracy $\varepsilon \sim 10^{-5}$ the GWRM without the CBC-scheme achieves a CPU time $t = 0.266$ s and memory 23.2 MB. See Discussion for more details concerning the CPU time of the GWRM with and without CBC scheme.

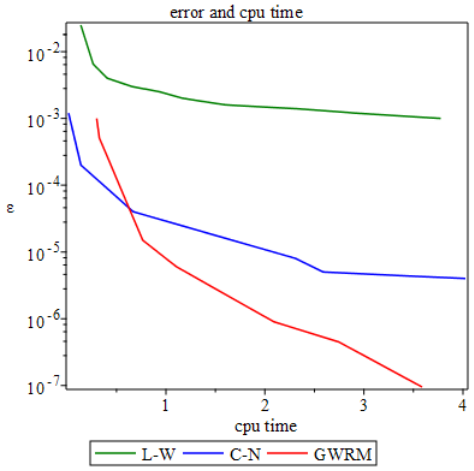


Figure 4: Error plot vs CPU time [s]

The three methods have been analysed with optimal parameters so as to receive the best accuracy for a given computational time. The data points can be seen in Figure 4, which shows how all three methods scale in this regard. A simple curve-fit of the data in Excel shows that the L-W method scales $\varepsilon_{LW} \sim t_{CPU}^{-0.6}$, and C-N method scales slightly better at $\varepsilon_{LW} \sim t_{CPU}^{-0.9}$. The spectral convergence properties of the CBC-GWRM allows it to scale much better than the FD methods, leading with a scaling $\varepsilon_{GWRM} \sim t_{CPU}^{-3.47}$. It can also be calculated that the CPU time scales linearly $t_{CPU} \sim N_s^1$ with regards to the number of subdomains used, see Figure 11, in Discussion. This feature of the CBC-GWRM becomes highly advantageous when considering physical systems that require high local spatial accuracy.

3.2. Wave Equation

The forced wave equation employed is a second order hyperbolic equation that features two time scales. The wave equation is used to test efficiency, because in some cases, accuracy at small scales may be ignored in favour of large scale dynamics. The wave equation has the form,

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} + f(t, x). \tag{14}$$

which can be posed as two partial differential equations:

$$\frac{\partial v}{\partial t} = c^2 \frac{\partial^2 u}{\partial x^2} + f(t,x),$$
$$\frac{\partial u}{\partial t} = v.$$

and the initial and boundary conditions are,

$$u(t,0) = u(t,1) = 0,$$
$$v(t,0) = v(t,1) = 0,$$
$$u(0,x) = sin(\pi x),$$
$$v(0,x) = \alpha A \sin(\beta x).$$

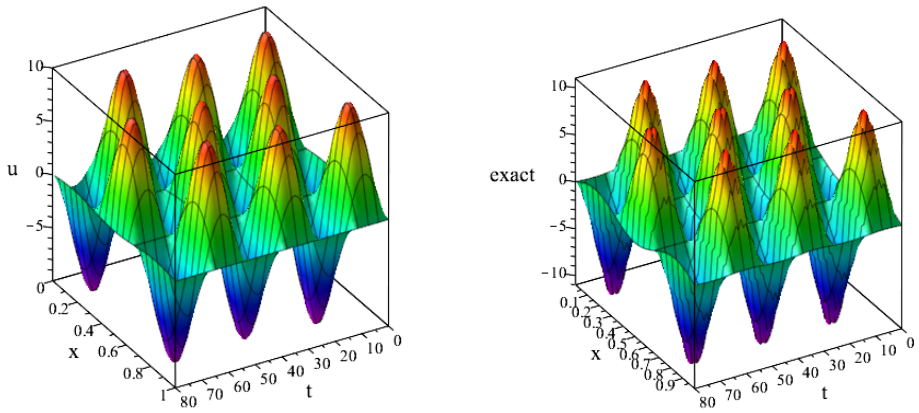
Here A, n, α, β and c are free parameters. The forcing equation chosen is,

$$f(t,x) = A(c^2\beta^2 - \alpha^2)\sin(\alpha t)\sin(\beta x).$$

This gives an exact solution,

$$u(t,x) = \cos(nc\pi t)\sin(n\pi x) + A\sin(\alpha t)\sin(\beta x). \tag{15}$$

For the test shown here the free parameters are set to $A = 10$, $n = 3$, $T = 80$, $\alpha = 4\pi/T$, $\beta = 3\pi$ and $c = 1$. The CBC-GWRM solution and the exact solution can be seen in Figure 5a,5b. For the time interval $t \in [0, 80]$ the CBC-GWRM requires a high number of temporal Chebyshev modes $m \geq 12$ to achieve a desirable average of the fast time scale dynamics. With the parameters $N_s = 4$, $m = 14$, and $n = 5$ the computation of the wave equation took 13.7 s to complete with a memory of 283 MB. The solution at $x = 0.2$ can be seen in Figure 6.



(a) CBC-GWRM (b) Exact

Figure 5: CBC-GWRM and exact solution of the forced wave equations in the range $t = [0, 80]$. CBC-GWRM parameters are $N_s = 5$, $m = 12$, and $n = 5$.

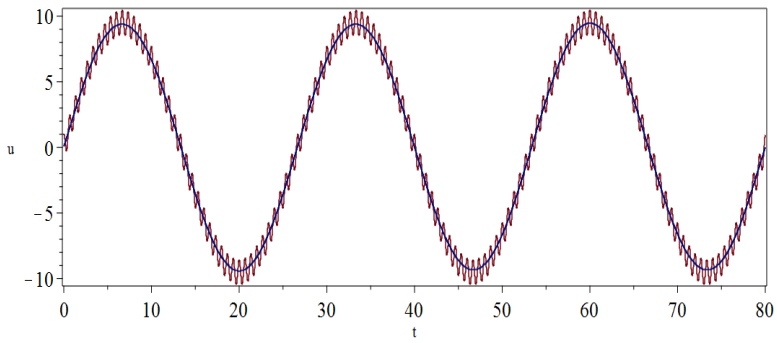


Figure 6: Top-Left: CBC-GWRM solution in the range $t = [0, 80]$ with $N_s = 4$, $m = 14$, and $n = 5$. Temporal plot at $x = 0.2$ showing how the CBC-GWRM method resolves the slower time scale.

The L-W method reaches a comparable average with parameters $M = 3200$ and $N = 40$ in 4.51 s. It can be seen in Figure 7 initially follows the fast time scale until the solution advances further away from the wall, in which case it transitions to the slow time scale.

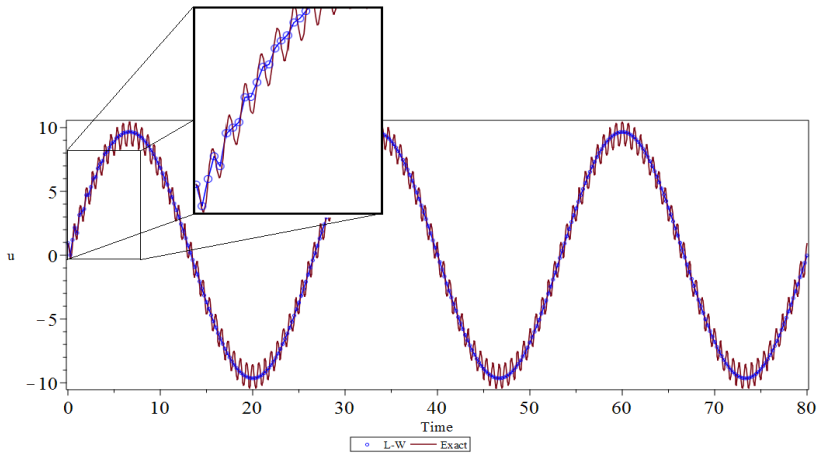


Figure 7: Lax-Wendroff temporal plot at $x = 0.2$ with $t = [0, 80]$ compared to the exact solution. $M = 3200$ and $N = 40$

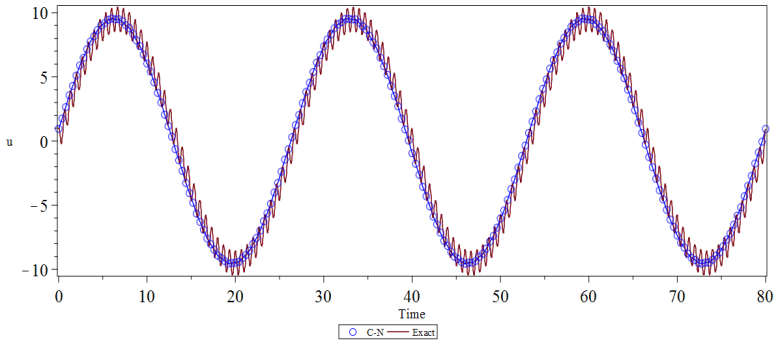


Figure 8: Crank-Nicolson temporal plot at $x = 0.2$ with $t = [0, 80]$ compared to the exact solution. $M = 200$ and $N = 50$

In Figure 8 the C-N solution is presented with the exact solution. The parameters used for the C-N solution is $M = 200$ and $N = 50$, which gives a run time of 4.37 s. The C-N method fails to average correctly as it is slightly out of phase. With implicit finite difference schemes the problem of stability becomes a weak criterion, compared to the strong stability criterion for explicit methods. However, since the forced wave equation is a hyperbolic equation, the C-N method needs to resolve the phase of the wave in order to be accurate. Furthermore, the phase can only be resolved by following the

strong stability criterion, in which case implicit method shows no favourable qualities over explicit schemes.

The CBC-GWRM is approximately $2-3$ times slower for the forced wave equation test than its finite difference counterparts. The main reasons being that the implicit derivatives in Eq. 9 are solved for analytically, and the amount of $\partial x^s_{Pi}/\partial x^s_{BCq}$ coefficients increase substantially with the number of equations and temporal modes. However, the attractive properties of stability and efficiency are both present in the spectral scheme. Therefore, it still stands that the CBC-GWRM will be competitive at higher accuracies. With the same parameters the GWRM without the CBC-scheme achieved a CPU time $t = 4.51\ s$ and memory 30.1 MB (see Discussion).

3.3. Ideal MHD

The ideal mhd model is a set of coupled partial differential equations that describe the dynamics of a perfectly conductive fluid. The ideal mhd equations provide the simplest description of plasma dynamics. The equations are stated as

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{v}) = 0, \tag{16}$$

$$\rho\Big(\frac{\partial}{\partial t} + \vec{v} \cdot \nabla\Big)\vec{v} + \nabla p - \vec{J} \times \vec{B} = 0, \tag{17}$$

$$\frac{\partial p}{\partial t} + \nabla \cdot (p\vec{v}) + (\Gamma - 1)p\nabla \cdot \vec{v} = 0, \tag{18}$$

$$\frac{\partial \vec{B}}{\partial t} + \nabla \times \vec{E} = 0, \tag{19}$$

$$\nabla \times \vec{B} - \mu_0 \vec{J} = 0, \tag{20}$$

$$\vec{E} + \vec{v} \times \vec{B} = 0. \tag{21}$$

First we have the fluid equations; the continuity Eq. 16 that describes the conservation of mass density ρ; the equation of motion (17) which solves for the fluid velocity $\vec{v}$; and the energy equation (18) describes the evolution of the pressure profile p;

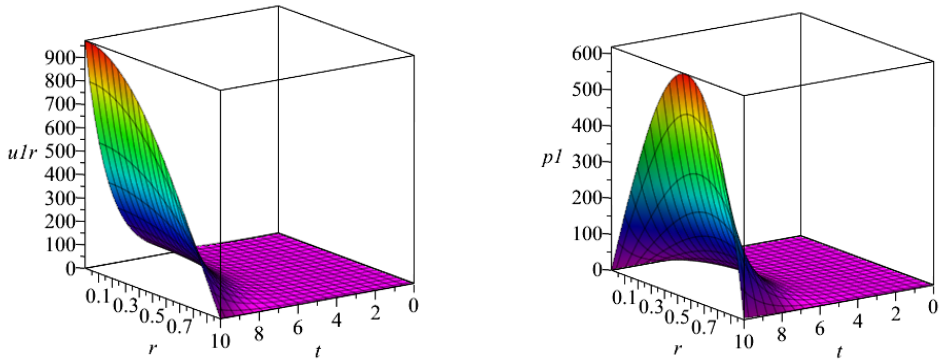
$$\Big(\frac{\partial}{\partial t} + \vec{v} \cdot \nabla\Big)\Big(\frac{p}{\rho^\Gamma}\Big) = 0, \tag{22}$$

where $\Gamma = 5/3$ is the ratio of specific heats. The second set involves the electromagnetic equations that describe the evolution of the magnetic field

$\vec{B}$, electric field $\vec{E}$, and the current density $\vec{J}$; Faraday's law (19), Ampere's law (20), where μ_0 is the permeability in vacuum, and Ohm's law (21).

The goal is to solve for ideal MHD instabilities that occur in a magnetically confined cylindrical plasma with coordinates (r, θ, z). The CBC-GWRM is applied to the linearised ideal mhd equations about an equilibrium, which consequently consists of 14 (7 real and 7 imaginary) coupled partial differential equations. A perturbation $\propto \exp[i(m\theta + kz)]$ is then introduced, where m and k are the azimuthal and transverse mode numbers, respectively.

The inner boundary conditions need to ne handled with some care to avoid singularities at $r = 0$ [9]. The outer boundary $r = 1$ condition is that of a perfectly conducting wall, hence the radial variable components are set to zero at the wall.



(a) Perturbed radial velocity

(b) Perturbed radial pressure

Figure 9: Perturbed screw-pinch, CBC-GWRM $m = 7$, $n = 5$ and $N_s = 3$.

The equilibrium chosen is a simple screw-pinch equilibrium,

$$B_r = 0,\ B_t = r,\ B_z = 0.05,$$
$$p_0 = 1 - r^2,\ J_t = 0,\ J_z = 2.$$

The equilibrium was then perturbed with the modes $m = 1$ and $k = 5$. The perturbed radial velocity and pressure is shown in Figures 9a,9b. Initially the perturbed variables are dominated by a host of different waves, which is why the simulation has to advance far enough $t \in [0, 10]$ for the dominating unstable mode to become distinguishable.

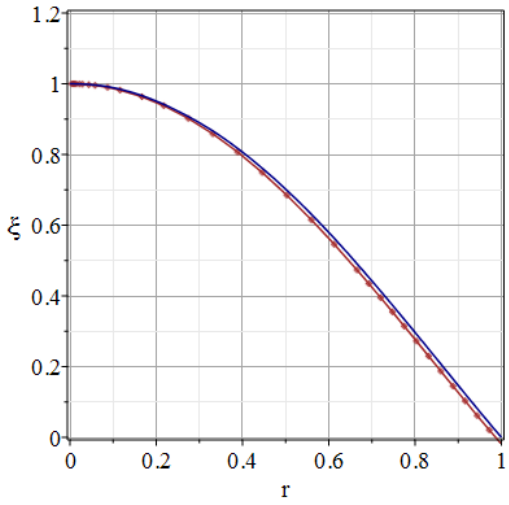


Figure 10: Eigenfunction $\xi(r)$ of perturbation $m = 1$, $k = 5$. CBC-GWRM (*line*) $\omega_{CBC} = 0.841$ $m = 7$, $n = 5$ and $N_s = 3$; Bateman $\omega_B = 0.835$ (*point-line*)

The growth rate obtained from the CBC-GWRM is compared to a shooting code developed in [2]. The CBC-GWRM calculated a growth rate $\omega_{CBC} = 0.841$ whilst the shooting code obtained a growth rate $\omega_B = 0.835$. This is in an excepted range of less than a 1% error. For higher accuracy it is advised to use multiple time intervals with fewer temporal modes. Also, this linearised MHD test shows the advantage of using a Chebyshev spectral ansatz because of its ability to average over small scale dynamics. The initial waves that propagate are of no interest, so they are average out, which leaves the dominating unstable mode.

4. Discussion

The CBC-GWRM has several favourable properties; 1) Spectral convergence and accuracy in both time and space 2) Chebyshev polynomial properties such as the mini-max property and the polynomials are purely real 3) Sparse matrices 4) Reduced global matrix where only the internal and external boundary conditions are solved 5) Subdomains are parallelizable. The first three points are common properties of spectral methods, while the work here is focused on points 4-5. The number of operations reduces by $\sim (2/L)^3$ and the memory by $\sim (2/L)^2$ when only the internal and external boundary conditions are solved globally. This is because the two highest spatial

Chebyshev modes are allocated for the two boundary conditions.

Another venue of potential gains in speed for the CBC-GWRM lies in the fact that the $\partial x_P/\partial x_{\mathrm{BC}}$ variables can be calculated numerically. This can be done with either forward difference or center difference. The number of private equations for one physical equation in each subdomain of, for example the Burger equation is $(L-1)(K+1)$. The private set of equations need to be solved twice for each common boundary condition and temporal mode. The total number of operations is then $4N_s(K+1)(L-1)^3(K+1)^3$. The standard GWRM scheme with sparse band matrix algorithms scales approximately as $N_s^{1.5}$. Thus the ratio of the total number of operations is,

$$\frac{4N_s(L-1)^3(K+1)^4}{N_s^{1.5}(L+1)^3(K+1)^3} = \frac{4(K+1)}{N_s^{0.5}}\left(\frac{L-1}{L+1}\right)^3. \tag{23}$$

From Eq. 23 it can thus be argued that the CBC-GWRM could be more efficient than the standard method given that N_s is large. For example; $L = 5$ and $K = 5$ gives the criterion that $N_s > 50$ for the CBC-GWRM to be more efficient. This is perfectly reasonable since many fluid dynamics applications suggest subdomains $N_s \gg 100$.

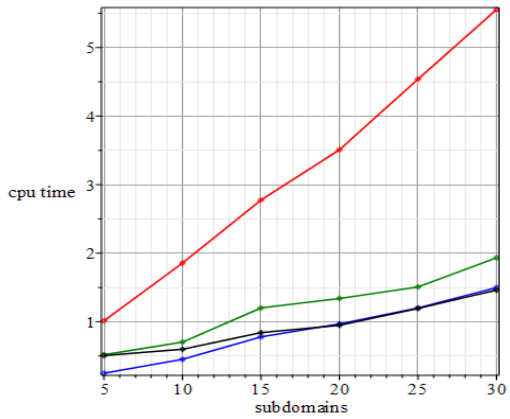


Figure 11: CPU time [s] with increasing subdomains for 1D linearised Burger computation; CBC-GWRM (*red*), standard GWRM (*blue*), crude approximation of theoretical parallelized CBC-GWRM $t_p{\sim}0.7(t_{CPU}/N_s)$ (*green*), and an approximated parallelized time that follows Amdahl's law for CBC-GWRM (*black*)

Also, it is not surprising that the unoptimized CBC-GWRM, i.e. no parallel implementation, is slower than its counterparts for bigger problems

since the extra steps involved in calculating the Jacobian of boundary conditions is solved in consecutive order. The work done in each subdomain can be allocated to multiple processors, which would decrease the computation time. Figure 11 shows how the CPU time scales with the number of subdomains $N_s \in [5, 30]$ for the CBC-GWRM and the standard GWRM (all equations solved globally). The standard GWRM outperforms the CBC-GWRM quite substantially. However, when approximations of the parallel speed-up gains are made we see that CBC-GWRM could potentially outperform the standard GWRM, whilst using less memory consumption. A crude approximation can be made by positing that $\sim$70% of the code is parallelizable and dividing by the number of subdomains. This assumes that one processor can be allocated for each subdomain. Another, more realistic picture, follows the speed-up gains ascertained from Amdahl's Law [1]. Amdahl's law is formulated as $S = 1/[(1-p) + p/s]$, where S is the speed-up gain, s is the proportion of the code that can be computed in parallel, and p is the unoptimized computation time of p. If parallelism is accounted for the discrepancy in CPU time can be resolved (see Figure 11).

5. Conclusion

The CBC-GWRM solves a set of ordinary or partial differential equations in the temporal domain and the spatial domain with a weighted residual method. This allows the method to obtain a high accuracy and to efficiently average over small scale dynamics, which can be seen in the linearised Burger test and forced wave equation test. From the outset of this work the time-spectral method solved all subdomains simultaneously from the global set of algebraic equations. The goal then was to break down the problem into smaller pieces. CBC-method provides a solution to that problem.

For the 1D linearised Burger equation the CBC-GWRM was compared to finite difference methods. For a lower accuracy the implicit scheme outperformed both the explicit and the spectral method. The maximum error of the CBC-GWRM solution scales $\varepsilon_{GWRM} \sim t_{CPU}^{-3.47}$, compared to $\varepsilon_{LW} \sim t_{CPU}^{-0.6}$ and $\varepsilon_{LW} \sim t_{CPU}^{-0.9}$. This allows CBC-GWRM to be 30% times faster than Crank-Nicolson for accuracies $\varepsilon \sim 10^{-5}$. The CBC-GWRM CPU time also scales linearly with increasing subdomains N_s^1 which is and improvement from the standard GWRM $N_s^{1.5}$.

A case has been made that the CBC-GWRM is capable of being more efficient than the global GWRM if the $\partial x_p / \partial dx_{BC}$ variables are computed

numerically. This allows the CBC-GWRM to be more competitive when many subdomains are used. More importantly, the CBC subdomain scheme can be parallelized so that the full potential of multiple CPUs and GPU acceleration can be harnessed.

The ideal MHD equations were solved within 1% accuracy for the instability growth in a screw-pinch, which agrees well with previous simulations [12]. This shows that the CBC-GWRM is capable of solving complex physical systems that are relevant for numerical modelling of fusion plasma physics.

References

[1] Amdahl, G. M., 1967. Validity of the single processor approach to achieving large scale computing capabilities. In: AFIPS '67 (Spring) Proceedings of the April 18-20, 1967, spring joint computer conference. ACM, pp. 483–485.

[2] Bateman, G., Schneider, W., Grossmann, W., 1974. Mhd Instabilities as an Initial Boundary-Value Problem. nuclear Fusion 14.

[3] Boyd, J. P., 2000. Chebyshev and Fourier Spectral Methods. Dover, New York.

[4] Canuto, C., Hussaini, M. Y., Quarteroni, A., Tang, T. A., 2006. Spectral Methods, Fundamentals in Single Domains. Springer.

[5] Furth, H. P., Rosenbluth, J. K. N., 1963. Finite-Resistivity Instabilities of a Sheet Pinch. Phys. Fluids 6.

[6] Giraldo, F. X., Restelli, M., 2007. A study of spectral element and discontinuous galerkin methods for the navier-stokes equations in nonhydrostatic mesoscale atmosphere modelling: Equation sets and test cases. Journal of Computational Physics 227, 3849–3877.

[7] Glasser, A. H., Greene, J. M., Johnson, J. L., 1976. Resistive instabilities in a tokamak. Phys. Fluids 19.

[8] Kopriva, D. A., 2009. Implementing Spectral Methods for partial Differential Equations. Springer.

[9] Lundin, D., 2006. Semi-analytical solution of initial-value problems. KTH Royal Insitute of Technology, Report XR-EE-ALF-2006:001.

[10] Mittal, R. C., Al-Kurdi, A., 2002. Lu-decomposition and numerical structure for solving large sparse nonsymmetric linear systems. Computers and Mathematics with Applications 43, 131–155.

[11] Peraire, J., Persson, P., 2010. High-Order Disctontinuous Galerkin Methods for CFD. Computational Fluid Dynamics 2, 119–152.

[12] Scheffel, J., 2011. Time-spectral solution of initial-value problems. In: Partial Differential Equations: Theory, Analysis and Applications. Nova Science Publishers, Inc., pp. 1–49.

[13] Scheffel, J., 2012. A Spectral Method in Time for Initial-Value Problems. American Journal of Computational Mathematics AJCM.

[14] Scheffel, J., Håkansson, C., 2009. Solution of systems of nonlinear equations, a semi-implicit approach. Applied Numerical Mathematics.

[15] Scheffel, J., Mirzal, A. A., 2012. Time-Spectral Solution of Initial-Value Problems - Subdomain Approach. American Journal of Computational Mathematics AJCM, 72–81.

[16] Spagnoli, K. E., Humphrey, J. R., Price, D. K., Kelmelis, E. J., 2011. Accelerating sparse linear algebra using graphics processing units. SPIE Defense and Security Symposium 8060.

[17] Strauss, H. R., 1976. Nonlinear, three-dimensional magnetohydrodynamics of noncircular tokamaks. Phys. Fluids 19.