UK Atomic Energy Authority

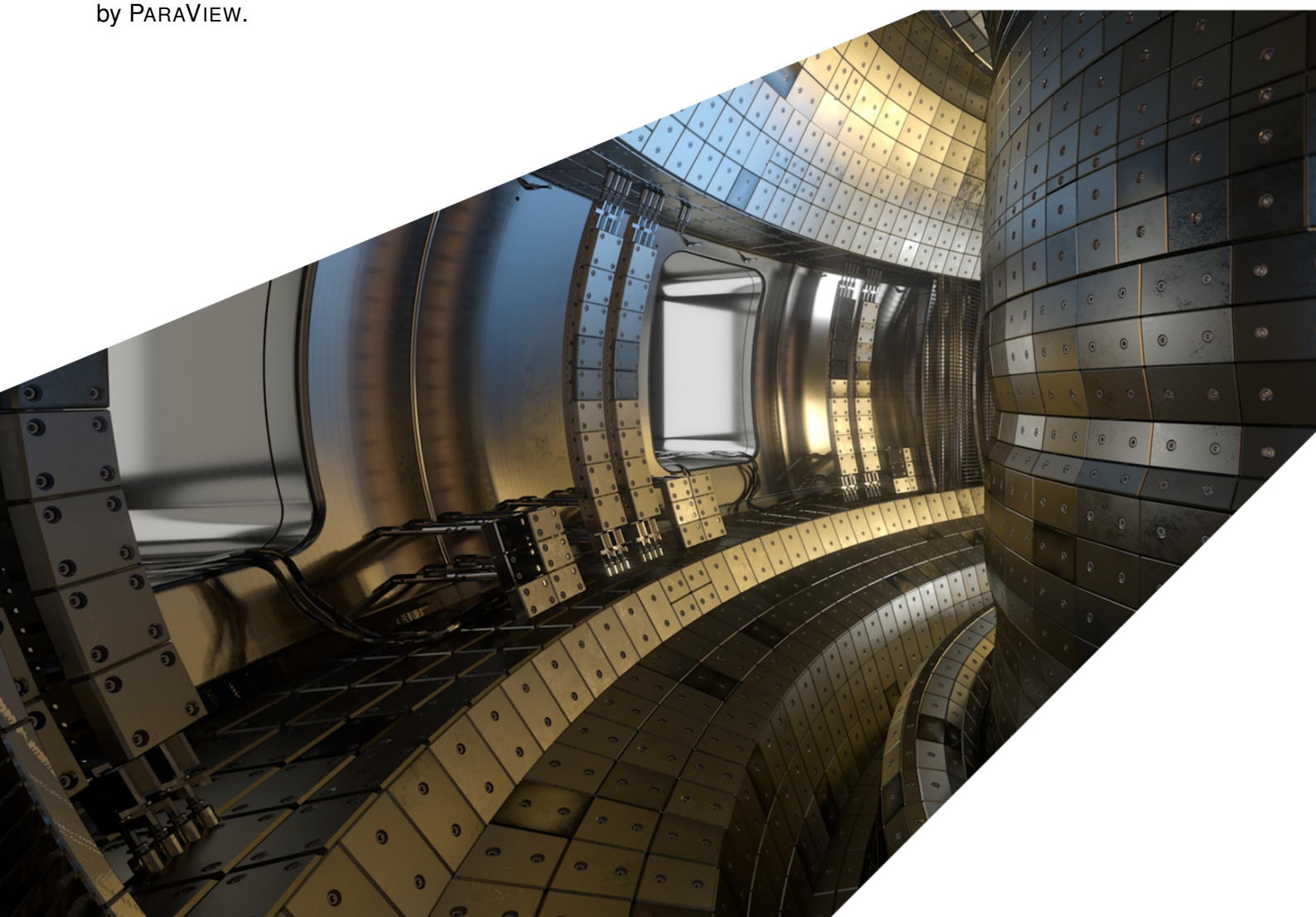
ExCALIBUR

Complementary actions. Code integration, acceptance and operation 4.

M6c.4 Version 1.00

Abstract

The report describes work for ExCALIBUR project NEPTUNE at Milestone M6c.4. It provides material concerning the implementation of plasma fluid solvers in NEKTAR++. Restricting attention to finite element problems *without* particles, there is a basic exploration of two-way code coupling, including time-independent and time-dependent problems. There are also heuristic discussions of the role of dimensionless parameters, particularly in relation to the numerical accuracy and stability of advection calculations. Thereafter are discussed numerical issues associated with solving highly-anisotropic diffusion problems, including a possible new option for reducing the difficulties caused by anisotropy. Appendices report progress (A) on the application of harmonic functions to 2-D and 4-D fluid problems, (B) a simple application of the Nvidia IndeX plugin for use with PARAVIEW and (C) a discovery of misleading 1-D plots of data in higher order representations by PARAVIEW.



UKAEA REFERENCE AND APPROVAL SHEET

	Client Reference:	
	UKAEA Reference:	CD/ExCALIBUR -FMS/0078
	Issue:	1.01
	Date:	29 September 2023
Project Name: ExCALIBUR Fusion Modelling System		

	Name and Department	Signature	Date
Prepared By:	Ed Threlfall Owen Parry BD	N/A N/A	29 September 2023 29 September 2023
Reviewed By:	Wayne Arter BD		29 September 2023

1 Introduction

The current phase of Project NEPTUNE involves the construction of proxyapps, implemented in the NEKTAR++ framework, capable of simulating plasma fluid equations. An outline of current code development work in this direction, which is being performed by UKAEA in collaboration with some of the grant holders, is presented in Sec.2. Following on from the realisation that NEKTAR++ does not support calculations where there is a finite volume of thermally conducting solid in addition to fluid, it has proved necessary to consider coupling of finite element code with finite element code, which is the topic of Section 3. (The coupling of finite element code with particle code will be discussed in the Deliverable 7c report due at the end of the FY.)

The computational results have thrown up a range of issues which have inspired corresponding work on code integration, acceptance and operation. Often there is difficulty is identifying the mechanism responsible for the computational problem, which might be as simple as a misunderstanding of the NEKTAR++ documentation, but could involve a deficiency in the numerical algorithm, a more fundamental problem due to the underlying nonlinear physics, or a combination of both the latter. Thus a number of lines of work have been pursued in parallel with the computations, then re-prioritised perhaps in light of new calculations indicating a different issue, so that parts of this present report inevitably describe work in progress.

Issues concerning stability of calculations, the need for implicit schemes and the accuracy of the results, are capable of being addressed by the dimensionless parameters introduced and described in Section 4. Section 5 contains a presentation of a simplified anisotropic diffusion problem and the numerical issues that arise in cases where the anisotropy is very large, including an idea for moderating these numerical difficulties that could be considered for implementation. Appendices include preliminary material on other lines of study which have been paused in the absence of any immediate application to Project NEPTUNE.

2 Plasma dynamics in NEKTAR++

This section contains a brief overview of plasma simulation codes implemented, and currently under implementation, in the NEKTAR++ framework. In the language of that framework the codes herein implemented are “novel NEKTAR++ solvers”.

2.1 2-D Proxyapps

The repository [1] contains a NEKTAR++ implementation, called *Nektar-Driftwave*, of the Hasegawa-Wakatani equations, which describe drift-wave turbulence in two dimensions under a number of simplifying assumptions. The outputs of this code are described briefly in the UKAEA report [2], in which context it should be noted that the ‘density’ in the model is not restricted to non-negative values because it actually describes a perturbation of a fixed background. Note that the small time-step means that the code is quite slow - c.32 core-hours to reproduce the example shown in Fig.1.

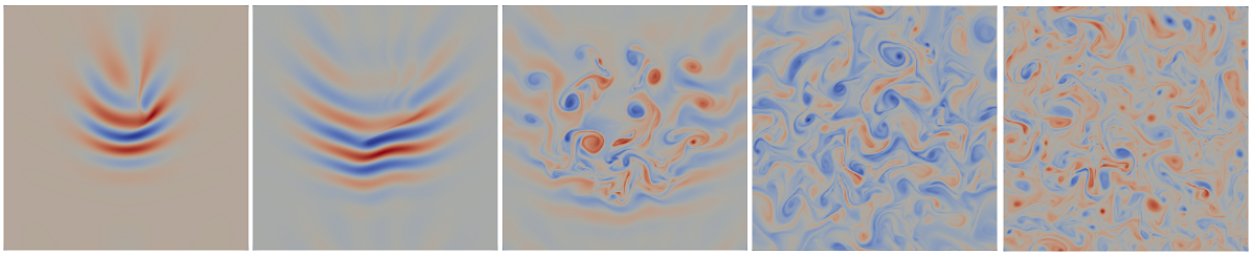


Figure 1: Plots showing the time-evolution in *Nektar-Driftwave* of the electron density, starting from an initial Gaussian-like density, vorticity, and potential profile.

The repository [3] contains NEKTAR++ implementations of the *Blob2D* and *Blob2D-Te-Ti* examples taken from the plasma simulation framework HERMES-3 [4], which is a currently active project led by Ben Dudson now of LLNL, differing from NEPTUNE in that it uses finite-difference methods as opposed to spectral/hp element. Both these implementations were written by the holder of NEPTUNE grant T/AW085/22 and were then further developed in collaboration with UKAEA NEPTUNE personnel. *Blob2d*, described in more detail in [5], is an isothermal model of an extended plasma filament propagating in two dimensions, with the forcing provided by the diamagnetic drift term. (The forcing arises as the centrifugal acceleration experienced by a region of increased density in a curved magnetic field). *Blob2D-Te-Ti* is an extension allowing different electron and ion temperatures.

There was an issue with the grantee's initial implementation of the above systems in that the Discontinuous Galerkin (DG) numerical flux term for the forcing term, eg. $\frac{\partial n}{\partial y}$, was absent, but the codes seemed to work acceptably well, giving plausible outputs. Plausibility is likely explained by the relatively small magnitude of the forcing term relative to pressure gradients and viscous effects, in any event this issue is currently being addressed in conjunction with the grantee. The issue does not affect the aforementioned *Nektar-Driftwave* as there the forcing term is a derivative of a continuous field. (To be more specific, the electrostatic potential Φ is a NEKTAR++ `ContField`, whereas the other fields are of type `DisContField` as they are used in a DG scheme).

2.2 3-D Proxyapps

A central goal of this phase of Project NEPTUNE is to implement a 3-D plasma fluid model capable of scaling ultimately to Exascale. Given the history of the project, it is natural to base this code on the NEKTAR++ framework. The code is necessary in order to meet upcoming UKAEA deliverable requirements beginning with M4c.3 *Integrated 3-D Particle and Continuum Model*, and is also to be supplied to grantees to focus eg. efforts on uncertainty quantification and reduced-order modelling during the remaining duration of the project.

Three possible models have been identified, with the idea that the simplest is a minimal system that nevertheless exhibits 3-D plasma turbulence and which still can be coupled to the UKAEA-developed particle capability, whereas the most complex represents the ability to address current problems of research interest. All three of these models exist in the same framework, using the same NEKTAR++ solver with different choices for `EquationSystem`, meaning there is a natural

upgrade path for any coupled code implemented using the simpler choices. All three examples use a uniform magnetic field aligned with the z-axis. Their initial implementations use purely explicit time-evolution schemes since in many tokamak applications the size of the timestep is likely to be determined by the parallel dynamics, whence implicit treatment of cross-field diffusion would not be required.

An important note is that one issue with NEKTAR++ affects all three examples: it is not yet entirely clear how to implement a purely transverse Laplacian operator $\nabla_\perp^2$, meaning basically the ability to invert $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ in a 3-D simulation, though the API for this appears to exist. The following applications currently use the full Laplacian inverse, awaiting full implementation of $\nabla_\perp^2$ into the library from the NEKTAR++ developers.

2.2.1 2-D Hasegawa-Wakatani equations implemented in 3-D

In this case the equations are those of the *Nektar-Driftwave* proxyapp referenced earlier [1]. The suppressed dimension is the one along the magnetic field direction. In this model the proxyapp website [1] gives values for all necessary values of scaling or physical parameters. There is no implicit parameter (see Section 4.1) because this model typically exhibits growth leading apparently to a saturated state regardless of initial perturbation. There is in *Nektar-Driftwave* a pre-existing example exhibiting saturated plasma turbulence.

In order to realise turbulence the accuracy of the simulation was found to need to be greater than a certain threshold, because the growth of turbulence is suppressed by numerical dissipation in low-accuracy simulations. This *Nektar-Driftwave* example has been run successfully in a 3-D configuration, see Fig.2. By starting with an axially-varying field and using a domain with a larger z-extent than this latter case, a more demanding test of 3-D numerical stability has also been produced, see Fig.3.

2.2.2 Fully 3-D Hasegawa-Wakatani equations

The 2-D Hasegawa-Wakatani system of the preceding sections is a reduction under the assumption of a single longitudinal Fourier mode, of a 3-D system that includes parallel dynamics. The 3-D equations are from B.Dudson (unpublished note, dated 13/5/2015)

$$\begin{aligned} \frac{\partial n}{\partial t} + [\Phi, n] + \frac{\omega_{ce}}{\nu_{ei}} \nabla \cdot \left(\mathbf{b} \left(\partial_\parallel \mathbf{\Phi} - \partial_\parallel \mathbf{n} \right) \right) &= 0 \\ \frac{\partial \omega}{\partial t} + [\Phi, \omega] + \frac{\omega_{ce}}{\nu_{ei}} \nabla \cdot \left(\mathbf{b} \left(\partial_\parallel \mathbf{\Phi} - \partial_\parallel \mathbf{n} \right) \right) &= 0 \\ Z \nabla_\perp^2 \Phi &= \omega. \end{aligned} \tag{1}$$

where in the above, the 'Poisson bracket' $[a, b] \equiv \frac{\partial a}{\partial x}\frac{\partial b}{\partial y} - \frac{\partial b}{\partial x}\frac{\partial a}{\partial y}$.

One appealing aspect of this system is that, in the above dimension-reduced form, one adjustable physical parameter, namely the ratio of the electron plasma oscillation frequency ω_{ce} to the electron-ion collision frequency ν_{ei}, stands out. However, although there is an implementation of this model in HERMES-3, at the time of writing it appears that this is simply a test that the code

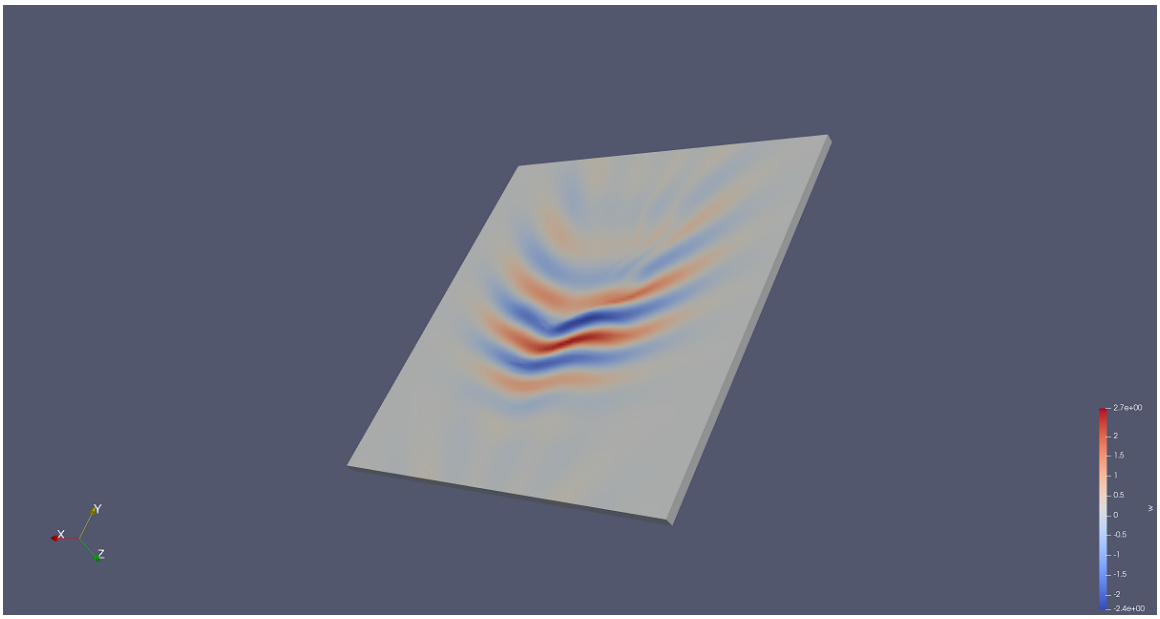


Figure 2: Plots showing the time-evolution of an initial Gaussian profile of electron density in a 3-D implementation of the 2-D Hasegawa-Wakatani equations. Periodic conditions are used in all dimensions and the initial data was symmetric in the axial direction. Note that this output corresponds to a time before the turbulence appears.

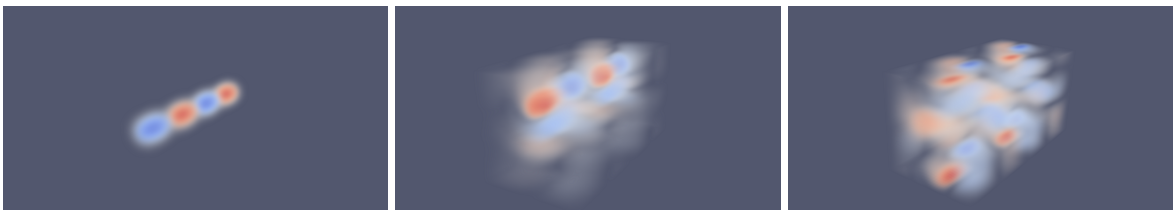


Figure 3: Plots showing the time-evolution of an initial Gaussian density profile modulated by a sinusoid in the axial direction, in a 3-D implementation of the 2-D Hasegawa-Wakatani equations. Periodic conditions are used in all dimensions. This output either precedes the appearance of turbulence or lacks the physical and / or numerical conditions needed to generate turbulence.

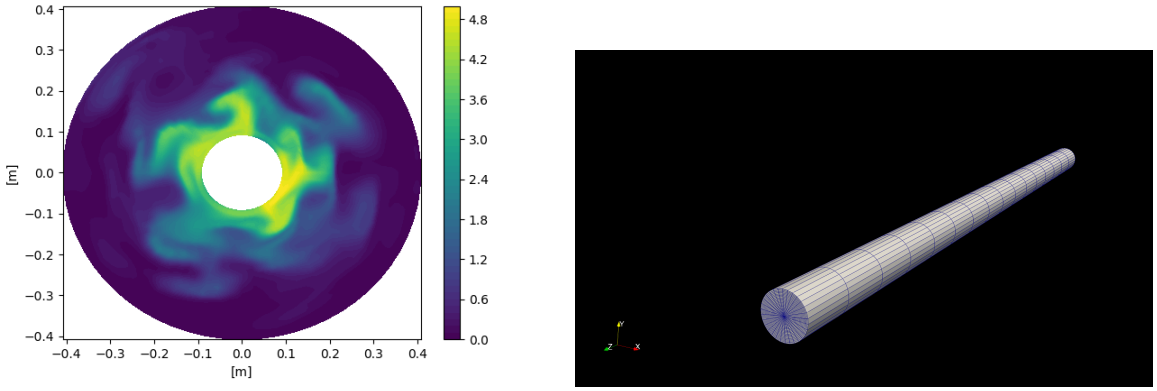


Figure 4: At left is output from the HERMES-3 LAPD example of the electron density at the end of the simulation. On the right is the computational mesh showing in particular the elongated aspect of the LAPD.

runs correctly on a GPU, hence its results are not certain to be physically relevant. The developers of HERMES-3 via the holders of Grant T/AW088/22, have advised that a value of approx. 2×10^5 for $\frac{\omega_{ce}}{\nu_{ei}}$ corresponds to physically-relevant values $T_e = 50\mathrm{eV}$ and $n_0 = 10^{19}\mathrm{m}^{-3}$ for the scrape-off layer of a tokamak such as MAST-U.

Work to implement this system in NEKTAR++ is ongoing in collaboration with the holders of Grant T/AW085/22.

2.2.3 Large Plasma Device (LAPD) simulation

The aim here is a NEKTAR++ implementation of the HERMES-3 model of the Large Plasma Device machine. The system of equations is a model for the physics of the Large Plasma Device [6] located at UCLA, which differs from a tokamak in that it has a long straight geometry. This model system was chosen as a relatively straightforward 3-D example of plasma turbulence and a problem of current research interest as agreed in discussion with representatives of UKAEA Tokamak Science. The HERMES-3 example executes in approximately eight core-hours; its output is shown in Fig.4, along with a computational mesh showing the basic geometry of the LAPD.

This two-fluid model has considerable additional complexity beyond the simpler Hasegawa-Wakatani treatments shown above. The equations, with subscript e used for electrons and i for ions which are assumed to be Deuterium, are for the evolution of respectively electron density, separate parallel species velocities, and a perpendicular plasma velocity in the streamfunction-vorticity for-

mulation

$$\begin{aligned}
\frac{\partial n_e}{\partial t} &= -\nabla\cdot\left(n_e\left(\mathbf{v}_{E\times B}+\mathbf{b}v_{\parallel}\right)\right);\\
m_e\frac{\partial n_e v_{\parallel e}}{\partial t} &= -\nabla\cdot\left(m_e n_e v_{\parallel e}\left(\mathbf{v}_{E\times B}+\mathbf{b}v_{\parallel}\right)\right)-\partial_{\parallel}p_e-en_eE_{\parallel}+m_e n_e\nu_{ei}\left(v_{\parallel i}-v_{\parallel e}\right);\\
m_i\frac{\partial n_i v_{\parallel i}}{\partial t} &= -\nabla\cdot\left(m_i n_i v_{\parallel i}\left(\mathbf{v}_{E\times B}+\mathbf{b}v_{\parallel}\right)\right)-\partial_{\parallel}p_i-en_iE_{\parallel}-m_i n_i\nu_{ei}\left(v_{\parallel i}-v_{\parallel e}\right);\\
\frac{\partial\omega}{\partial t} &= -\nabla\cdot\left(\omega\left(\mathbf{v}_{E\times B}+\mathbf{b}v_{\parallel}\right)\right)+\nabla\left(n_i v_{\parallel i}-n_e v_{\parallel e}\right);\\
\nabla\cdot\left(\frac{\overline{mn}}{B^2}\nabla_{\perp}\Phi\right) &= \omega.
\end{aligned} \tag{2}$$

where $\overline{mn}$ is plasma mass density, and the ion density and species pressures are

$$\begin{aligned}
n_i &= n_e;\\
p_e &= kn_eT_e;\\
p_i &= kn_iT_i;\\
\mathbf{v}_{E\times B} &= \frac{\mathbf{b}\times\nabla\Phi}{B}.
\end{aligned} \tag{3}$$

where T_e and T_i are user-specified, and the last equation shows how the electrostatic potential plays a role analogous to a streamfunction.

Fuller details appear in a new release of the Equations document [7]. An important feature of the boundaries is the sonic outflows of electrons and ions at the ends of the device, and it is worth noting the densities in the equations (2) are strictly non-negative. Work to implement this system in NEKTAR++ is ongoing in collaboration with the holders of Grants T/AW085/22 and T/AW088/22.

2.3 Demonstration of Kelvin-Helmholtz instability in NEKTAR++

The physics of 3-D plasma involves the generation of turbulence by several mechanisms, including the Kelvin-Helmholtz instability which means that a velocity field with sufficient velocity gradient transverse to the main flow experiences a growing instability. In order to study this, a simple flow field driven by a body force was implemented in the incompressible Navier-Stokes solver of NEKTAR++. The simple functional form of the body force produces flow fields reminiscent of the fluid convection studied in eg. [8] but lacking the complexities of the heat transfer physics such as diffusion of heat.

The strength of the forcing is scaled to a parameter akin to the Rayleigh Number Ra, so that the functional form is

$$F = \mathrm{Ra}\sin^9\left(2\pi\left(y-\frac{1}{2}\right)\right); \tag{4}$$

and the kinematic viscosity was set to a value of 0.1.

Three examples were obtained using different values of Ra, see Figs. 5, 6 and 7. It is seen that the resulting $\mathrm{Ra}=10$ case is a steady flow, $\mathrm{Ra}=10^2$ is an oscillatory flow and $\mathrm{Ra}=10^3$ is a

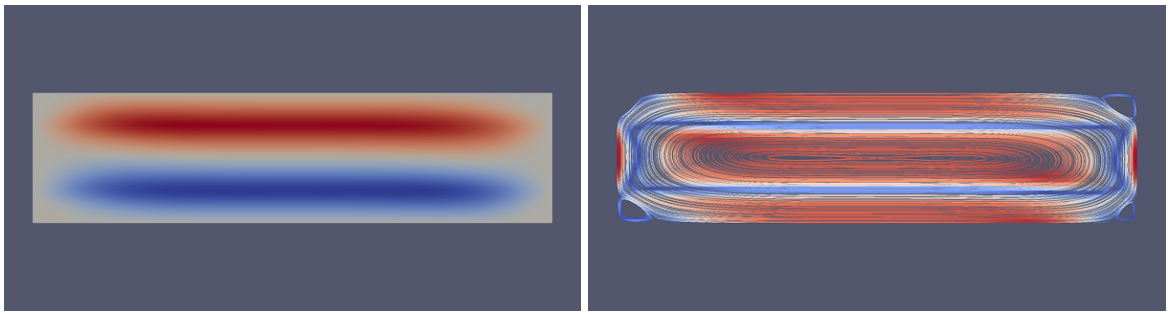


Figure 5: Horizontal velocity component (red is right-moving and blue is left-moving) (left), and streamlines coloured according to vorticity (right) for $Ra = 10^1$. The flow is steady and the vortex field is largely featureless. The scales are such that the largest values are approximately ± 1.4 for the velocity component and up to 7.4 for the vorticity in the units used by the code.

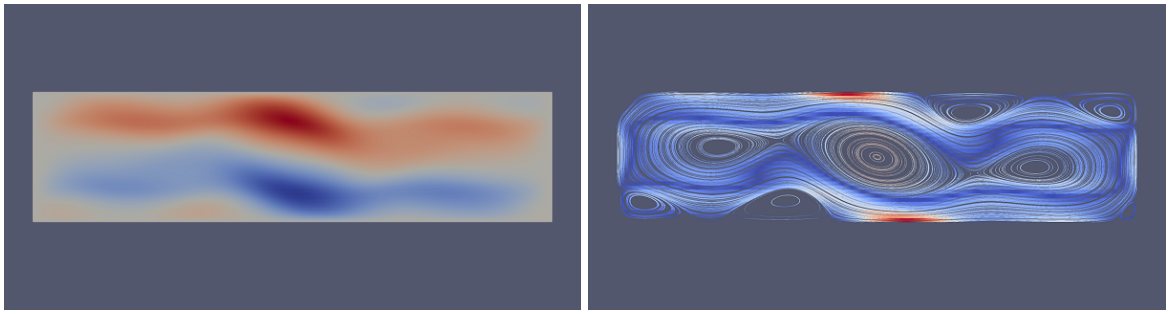


Figure 6: Horizontal velocity component (red is right-moving and blue is left-moving) (left) and streamlines coloured according to vorticity (right) for $Ra = 10^2$. The flow is oscillatory as there is not quite enough energy present for a stable three-vortex pattern to form. The scales are such that the largest values are approximately ± 14 for the velocity component and up to 140 for the vorticity in the units used by the code.

steady vortex flow. It is expected that larger Ra values would lead to turbulent flows, possibly via steady states with higher numbers of vortices.

3 Coupling Finite Element Codes

Thanks to the present limitations of NEKTAR++ it is necessary to treat problems with one or more regions of solid with different thermal conductivities using code coupling. This situation is mocked up by coupling two models with different thermal conductivities using FIREDRAKE [9]. This section contains some simple examples of coupling NEPTUNE -relevant codes, which could be used with coupling frameworks such as CWIPI and MUI. Note that in this section and the following one, diffusion and heat conduction are treated as being interchangeable.

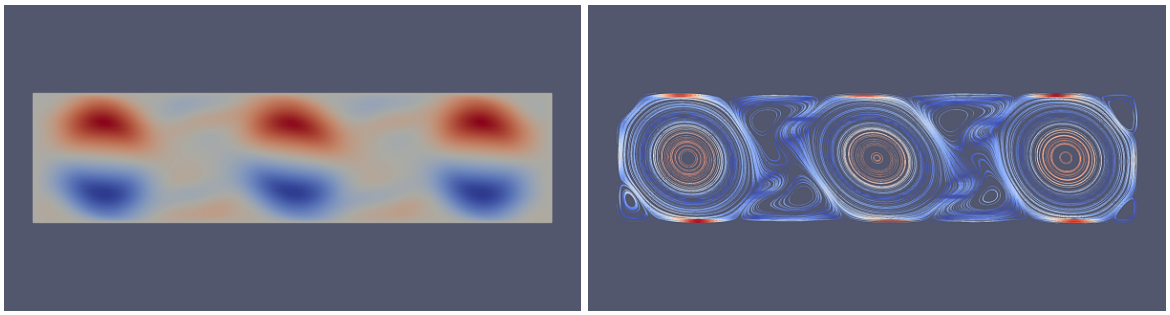


Figure 7: Horizontal velocity component (red is right-moving and blue is left-moving) (left) and streamlines coloured according to vorticity (right) for $Ra = 10^3$. The three vortex pattern is stable, though subject to minor wobbles. The scales are such that the largest values are approximately ± 66 for the velocity component and up to 750 for the vorticity in the units used by the code.

3.1 Time-independent problems

3.1.1 Prelude: one-dimensional diffusion or heat conduction

The model described here involves two regions of thermally conducting material, both unit square. Let

1. the temperature of the left-hand side of the composite be unity and that of the right side zero.
2. the top and bottom of both squares be insulated, and boundary conditions be chosen so that the heat flow is one-dimensional.
3. the parameters of the left-hand region be labelled with index 1 and those of the right-hand with index 2; thus the thermal conductivities are κ_1 for the leftmost region and κ_2 for the other.
4. the ratio of thermal conductivities be $\beta \equiv \frac{\kappa_1}{\kappa_2}$.

It is useful to know that for a one-dimensional conduction problem the mid-boundary temperature is given by $T_{mid} = \frac{\beta}{1+\beta}$ and also that β is the ratio of the temperature gradient in the second region to that in the first. These results are trivially obtained by matching the temperature and heat flux at the internal boundary.

The natural approach is to couple the two regions (left and right) at their common boundary, separately updating the boundary conditions in a series of iterations, leading to the following algorithm:

- Select initial guess T_0 for the internal boundary temperature.
- Solve the left-hand-side (LHS) conduction problem.
- Use the value of the left-hand-side temperature gradient to give a Neumann boundary condition for the right-hand-side (RHS) conduction problem, by matching heat flux.

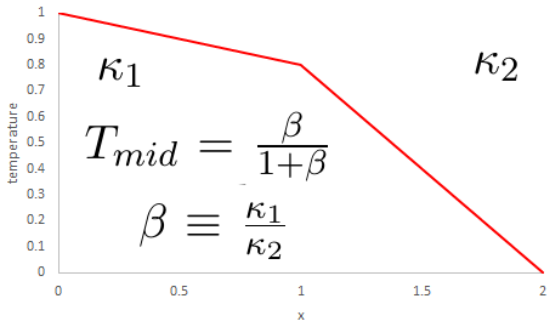


Figure 8: One-dimensional conduction.

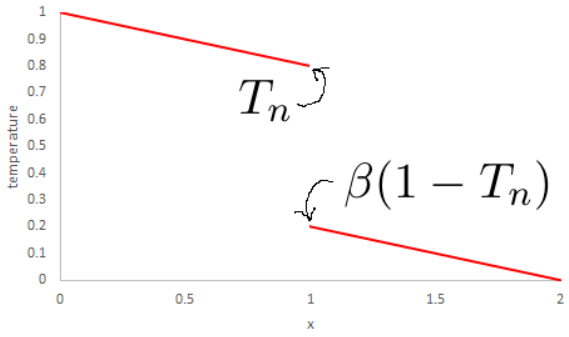


Figure 9: Sketch of coupling algorithm in 1-D.

- Solve the RHS conduction problem.
- Use a weighted average of the internal boundary temperatures of the LHS and RHS region to overwrite the initial guess for the internal Dirichlet boundary condition for the LHS region, and go to the second step.
- Iterate all but the first step until convergence.

The above algorithm successfully gives a converged answer for a range of values of β. The algorithm is sufficiently simple that its working can be reproduced analytically: let the midpoint temperature after the nth iteration be labelled by T_n, then there follows the recurrence relation (letting the LHS temperature be weighted by $\frac{w-1}{w}$ and the RHS one $\frac{1}{w}$ in the Dirichlet boundary condition)

$$T_{n+1} = \frac{1}{w}\left((w-1)T_n + \beta(1-T_n)\right)) \tag{5}$$

which has solution

$$T_n = \left(\frac{w-1-\beta}{w}\right)^n \left(T_0 - \frac{\beta}{1+\beta}\right) + \frac{\beta}{1+\beta}. \tag{6}$$

This has the obvious form of the desired answer $\frac{\beta}{1+\beta}$ plus an error term proportional to the distance between the initial guess T_0 and the desired answer. It is not difficult to see that if the recurrence relation converges then it converges as a nested product to

$$\frac{\beta}{1+\beta} = \frac{1}{w}\left(\beta + (w-1-\beta)\,\frac{1}{w}\left(\beta + (w-1-\beta)\,\frac{1}{w}\left(\beta + (w-1-\beta)\,\ldots\,\right)\right)\right). \tag{7}$$

If expanded this is seen to be equivalent to the Taylor series for $\frac{\beta}{1+\beta}$ around $\beta = w-1$ which is just

$$\frac{\beta}{1+\beta} = \frac{w-1}{w} + \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{w^{n+1}}(\beta - w + 1)^n, \tag{8}$$

that obviously has radius of convergence $r_c = w$ and for real β converges for $-1 < \beta < 2w-1$.

Another approach is to use the analytic solution in one dimension to choose the weighting coefficients. This leads to the conclusion that the LHS temperature should have weight $\frac{\beta}{1+\beta}$ and the RHS $\frac{1}{1+\beta}$ (obviously these sum to unity). In 1-D this value yields the exact solution after one iteration because

$$\frac{\beta}{1+\beta} \equiv \frac{\beta}{1+\beta}T_0 + \frac{1}{1+\beta}\,\beta(1-T_0). \tag{9}$$

(The Taylor series terminates after the constant term.) This choice of weighting will be referred to as the 'optimal' value.

For more complex cases, notably where one of the regions contains moving fluid, it is possible to draw on the theory of domain decomposition / Schwarz method, eg. [10].

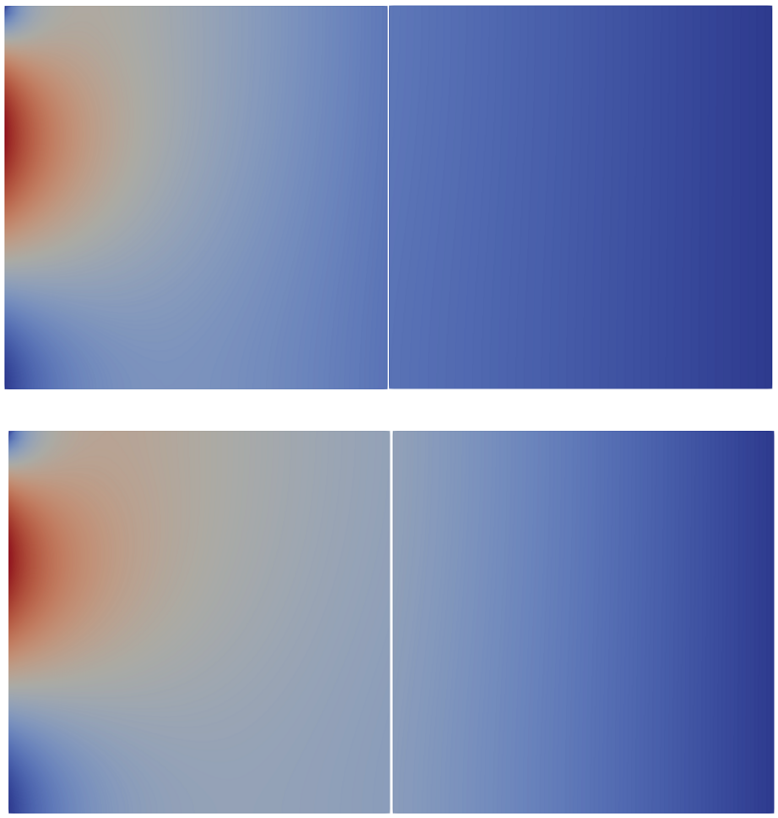


Figure 10: Solid conducting regions temperature profile for $\beta = 0.5$ (top) and $\beta = 2.0$ (bottom).

3.1.2 Two-dimensional diffusion, fluid heat transfer, and anisotropic diffusion

The method outlined above has been found to be effective for less-trivial two-dimensional problems, ie. a PDE rather than just an ODE. In this case, the 1-D boundary profile of the temperature and its x-derivative are used as boundary conditions. The problem is made two-dimensional by having a non-constant temperature profile on the left-hand boundary of the composite domain. The function $T = y^2(1-y)$ was used (chosen not to have any particular symmetry). The temperature and the heat flux output from the code match on either side of the internal boundary, see Fig. 10. Note that these results used $w = 2$ in the above formulae rather than the superior 'optimal' value. Further, since only the 'zero' transverse Fourier mode ever transmits non-zero heat flux, in important aspects higher-dimensional cases resemble the one-dimensional problem.

It can be shown that the same approach works when the first conductor is replaced with a laminar convecting fluid cell. The fluid problem is incompressible Navier-Stokes with Boussinesq-approximation buoyancy term and is parametrized by the usual Rayleigh (Ra) and Prandtl (Pr) numbers. The temperature at the internal boundary is allowed to vary, giving what is described by Nield [11] as the 'Rayleigh-Jeffreys problem'. Parameter β now represents the ratio of the thermal conductivity of the fluid to that of the solid.

An example output temperature profile is shown in Fig.11, which has $\beta = 0.5$, $\mathrm{Ra} = 10^5$, $\mathrm{Pr} = 1.0$. The temperature on the left-most boundary is set to be unity. Example streamlines in the fluid

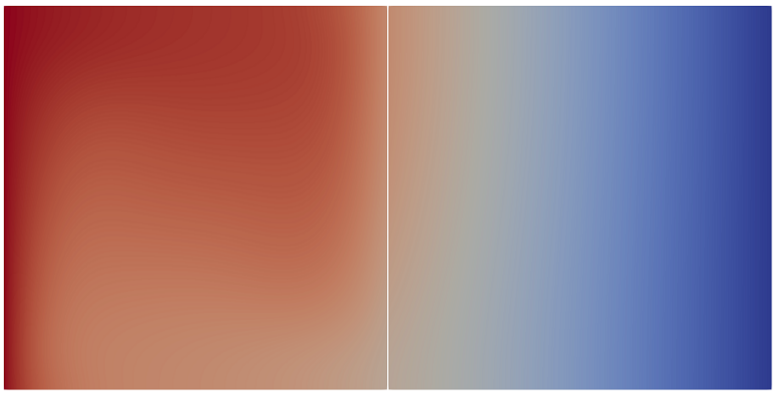


Figure 11: Fluid (left) and solid conducting regions temperature profile for $\mathrm{Ra} = 10^5$, $\mathrm{Pr} = 1.0$, $\beta = 0.5$.

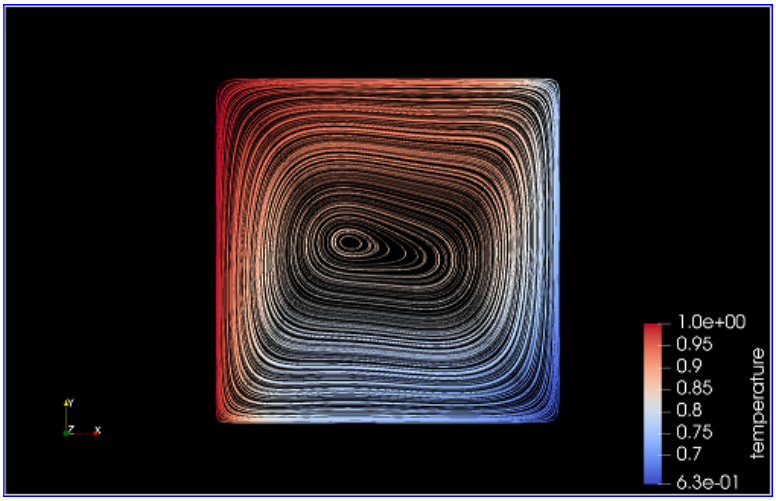


Figure 12: Fluid region flow streamlines coloured by temperature profile for $\mathrm{Ra} = 10^5$, $\mathrm{Pr} = 1.0$, $\beta = 0.75$.

region are shown in Fig.12 (for $\beta = 0.75$) where it can be seen that the coupled solution does not have the same symmetry as would be found in a typical slot problem with constant temperatures on the left and right uprights.

The method works also for problems with anisotropic diffusion ie. the diffusivity in either domain can be a symmetric tensor. The temperature and the normal component of the heat flux are made continuous across the boundary, and it is found that the method works correctly provided the expression for the flux component is correct - this is easily obtained using the usual rules for tensor transformation, giving the x-component of the LHS flux needed for the Neumann data as

$$\frac{\kappa_{\parallel}}{\kappa_2}\left(\cos^2\theta\frac{\partial T}{\partial x} + \sin\theta\cos\theta\frac{\partial T}{\partial y}\right) + \frac{\kappa_{\perp}}{\kappa_2}\left(\sin^2\theta\frac{\partial T}{\partial x} - \sin\theta\cos\theta\frac{\partial T}{\partial y}\right). \tag{10}$$

This has been tested for cases with $\kappa_{\parallel} = 10$ and $\kappa_{\perp} = 0$ and 1 and non-zero angles.

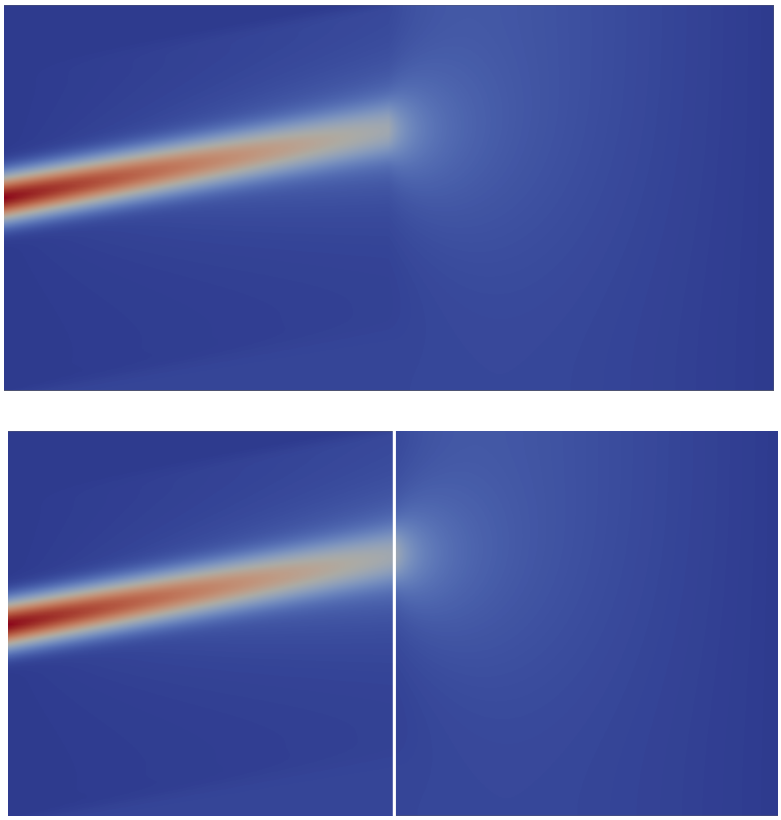


Figure 13: Anisotropic diffusion, tightly-coupled, $\kappa_\perp = 0$ in the left-hand-side region and $\beta = 10$. At top: all one FIREDRAKE solve, using a position-dependent diffusivity tensor. Below: the same problem done using two solves coupled as discussed in the main text, showing very good agreement which was further checked by inspection of 1-D sections.

The outputs from coupling of this type can be checked against a tight-coupled FIREDRAKE implementation (ie. the whole problem is one big domain with a diffusivity tensor that depends on position) - see Fig.13. Note that this two-way coupled implementation extends a simple one-way coupled model used earlier in the project and supplied to the holder of the grant T/AW085/22 concerning uncertainty quantification and reduced-order models.

3.2 Time-dependent problems

It is necessary to study time-dependent problems in order to treat cases with turbulence or with particle noise, both of which preclude a strictly time-independent solution.

One approach is to treat each timestep as an elliptic problem and use the 'optimal' method outlined above. This has been tested for a coupled time-dependent diffusion problem with scalar diffusivity $D = 1$ in both coupling regions and using a simple finite-difference code (a finite-element code could easily be substituted). The method to solve the elliptic problem is to call the time-step algorithm but not update the initial data (in the code this is achieved by copying back the original

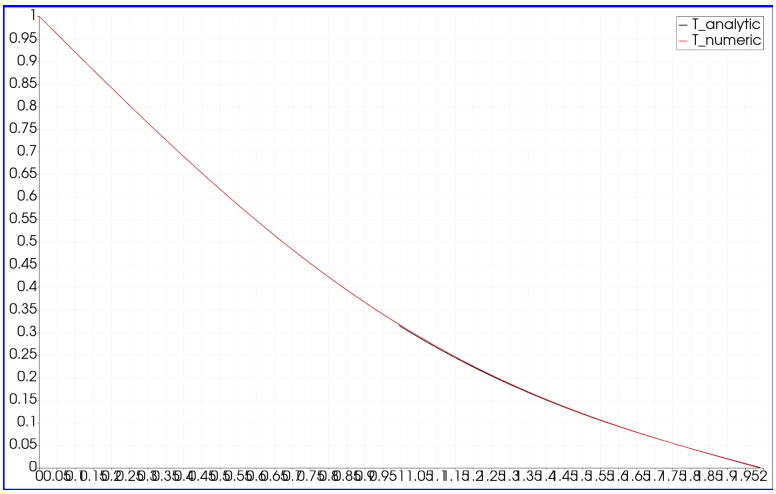


Figure 14: One frame at $t = 0.5$ from a time-dependent simulation of diffusion with coefficient $D = 1$, coupling in the middle of the domain. The initial data was $T = 0$ everywhere, with boundary Dirichlet values of 1 and 0. The horizontal axis is x and the vertical axis is temperature T.

data after the timestep).

For the backward Euler (implicit) scheme the elliptic problem is easily formulated from

$$\frac{T^{n+1} - T^n}{D\Delta t} = \nabla^2 \, T^{n+1} \tag{11}$$

where the solution T_n is assumed known at the start of the timestep, as

$$-\frac{d^2 T^{n+1}}{dx^2} + (D\Delta t)^{-1} \, T^{n+1} = (D\Delta t)^{-1} \, T^n. \tag{12}$$

The initial guess is the Dirichlet data at step n on the right-hand-side of the left-hand-side region. The Neumann boundary condition used to solve the RHS region comes from a simple finite difference at the RHS end of the LHS region. The Crank-Nicolson time-stepping method used in the LHS region needed to be downgraded to backward Euler since oscillations inherent to the former method resulted in non-convergence. These oscillations are a well-known feature of Crank-Nicolson which is absent in backward Euler, appearing when the initial data is $T = 0$ and the LHS boundary of the region is at $T = 1$.

The problem is to calculate diffusion on $[0, 2]$ with unit Dirichlet on the LHS and homogeneous Dirichlet boundary conditions on the RHS, for which the analytic solution is

$$T = 1 - \frac{x}{2} - \sum_{n=1}^{\infty} \frac{2}{\pi n} e^{-\frac{Dn^2\pi^2 t}{4}} \sin \frac{n\pi x}{2}; \tag{13}$$

Outputs from the coupled problem, see Fig.14 agree with the analytic solution.

4 Dimensionless Parameters and Their Use

This section develops ref [12, § 2.2], concerning discretisations of the time dependent advection-diffusion equation for a scalar field $f(\mathbf{x}, t)$ in a flow $\mathbf{u}(\mathbf{x}, t)$ with diffusion κ and source term $S(\mathbf{x}, t)$

$$\frac{\partial f}{\partial t} + \nabla \cdot (\mathbf{u} f) = \nabla \cdot (\kappa \nabla f) + S \tag{14}$$

Suppose that f only depends on a single spatial coordinate x, $\mathbf{u}$ has a single component u, and also that the diffusion $\kappa =$ const. Neglecting gradients of $\mathbf{u}$, the analytic model equation may be taken

$$\frac{\partial f}{\partial t} + u \frac{\partial f}{\partial x} = \kappa \frac{\partial^2 f}{\partial x^2} + S \tag{15}$$

Solution is taken to be needed in the finite domain $0 \leq x \leq L$ over a time interval $[0, T]$. Note that the assumptions made do not exclude the nonlinear case $f = u$, when Equation (15) is known as Burgers equation and κ becomes the viscous diffusion.

It will be important for subsequent work to distinguish different approaches to a dimensionless treatment of Equation (15) and thence Equation (14), see Section 4.1. The following Section 4.2 shows how discretisation of Equation (15) introduces new dimensionless quantities based on mesh spacing, and then Section 4.3 goes onto discuss the implications of these parameters for numerical stability and accuracy. Section 4.2 also includes treatment of the hyper-diffusion term which may be represented as an addition to the source in Equation (15) of form

$$S_4 = -\kappa_4 \frac{\partial^4 f}{\partial x^4} \tag{16}$$

where κ_4 is the hyper-diffusion, *aka* hyper-viscosity in the nonlinear case that $f = u$.

4.1 Dimensionless Parameters

Naturally because of the finite domain, distances are scaled by L, when by far the commonest approach, particularly when $S = 0$ is to make time t dimensionless in terms of the diffusion timescale L^2/κ. It will be recalled that this leads to a model in dimensionless variables of the form

$$\frac{\partial f}{\partial t} + Pe \cdot u \frac{\partial f}{\partial x} = \frac{\partial^2 f}{\partial x^2} \tag{17}$$

where the Peclet number

$$Pe = \frac{U_0 L}{\kappa} \tag{18}$$

with $U_0 = ||\mathbf{u}||$ as a representative absolute value of u, usually the maximum flow-speed. Making t dimensionless in terms of the turnover timescale L/U_0, similarly leads to the appearance of (reciprocal of) Pe in the coefficient of the diffusion term. The point is that in the linear problem, not only is κ fixed at least in order of magnitude, but also so is u/U_0 similarly of order unity throughout.

When u is varying appreciably, eg. in time as a result of instability in a coupled momentum equation (or say due to driving by S when $f = u$), it is often customary to omit the Peclet number, and work with a model in dimensionless variables of the form

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = \frac{\partial^2 f}{\partial x^2} \tag{19}$$

where the dimensionless u may vary over orders of magnitude. However, even when u is fixed, this formulation is workable. The fixed flow case using the model Equation (19) is then apparently parameter-free, although it should be evident that the scale of the initial values of u serves to set an implicit or 'hidden' Peclet number. (Similar remarks apply in respect of hyperdiffusion upon introducing $Pe_4 = ||\mathbf{u}||h^3/\kappa_4$.)

4.2 Discrete Dimensionless Parameters

Suppose h is the spatial mesh separation implicitly assumed to be uniform, ie. $h = L/N$, where $N + 1$ with $N \gg 1$ is the number of mesh-points in $0 \leq x \leq L$. Let Δt be the timestep, then the discrete time advance is of general form

$$\frac{\Delta_t f}{\Delta t} + \frac{u}{h}\Delta_x f = \kappa\frac{\Delta_x^2 f}{h^2} \tag{20}$$

where Δ_x and Δ_x^2 are discrete difference operators in the x-coordinate, eg. $\Delta_x f = f_{i+1} - f_i$, operating on the discrete values of f_i at the mesh-points i. (Δ_t is defined analogously.) If $\kappa = 0$ and there is only hyper-diffusion

$$\frac{\Delta_t f}{\Delta t} + \frac{u}{h}\Delta_x f = -\kappa_4\frac{\Delta_x^2 f}{h^4} \tag{21}$$

Evidently the discrete Equation (20) may be made dimensionless using the mesh-scale h, introducing the mesh Peclet number parameter

$$Pe_h = \frac{U_0 h}{\kappa} \tag{22}$$

There are questions concerning the role of Pe_h inherited from a good deal of controversy about the role of the mesh Reynolds number Re_h, viz. Pe_h for the nonlinear case. Nonetheless, at least heuristically it would seem that when $Pe_h = \mathcal{O}(1)$, there is an approximate balance between the *discrete* advective and diffusive terms. For smaller Pe_h the latter dominates, which is numerically significant in that the discrete Laplacian operator, being symmetric and definite, is much easier to treat numerically than the discrete advection operator. (Similar remarks apply to Equation (21) upon introducing

$$Pe_{4h} = \frac{U_0 h^3}{\kappa_4}) \tag{23}$$

Other important well-known dimensionless groups for Equations(20) and (21) are the Courant or CFL number and the diffusion limit parameters, respectively

$$c_h = \frac{U_0\Delta t}{h}, \quad d_h = \frac{\kappa\Delta t}{h^2}, \quad \text{and} \quad d_4 = \frac{\kappa_4\Delta t}{h^4} \tag{24}$$

When these are separately unity, the corresponding timestep has a simple physical interpretation as respectively the timescale for the turnover, diffusion and hyper-diffusion of one cell of the discretisation. Note also the identities

$$Pe_h = \frac{c_h}{d_h} \quad \text{and} \quad Pe_{4h} = \frac{c_h}{d_4} \tag{25}$$

4.3 Stability Parameters and Accuracy

If the temporal discretisation of Equation (20) is explicit, it is plausible (as is confirmed by rigorous analysis up to factors of order unity) that for stability the timestep must be less than or equal to the turnover and diffusion times of one cell, viz.

$$c_h \leq 1, \quad d_h \leq 1, \quad \text{and/or} \quad d_4 \leq 1 \tag{26}$$

Since maximal size of timestep is to be preferred for computational efficiency, then to within $\mathcal{O}(1)$ factors, $c_h = d_h = 1$ will be taken and so $Pe_h = 1$ at the stability boundary.

It is important to consider accuracy, again this is posed in heuristic terms by assuming that the smallest lengthscale ℓ important in the dynamics of f is capable of numerical estimation, or known say on the basis of analytic results. It follows that for minimally accurate results $h < \ell$. For advection-diffusion problems, whether or not they are linear, provided $S = 0$, then typically

$$\ell = LPe^{-\alpha}, \quad \text{or} \quad \ell_4 = LPe_4^{-\beta} \tag{27}$$

for positive $\alpha \leq 1$ and $\beta \leq 1$. Simple balancing of coefficients gives $\alpha = \beta = 1$, whereas smaller values follow in higher dimensions when the flow is incompressible (typically $\alpha = 1/2$) or from estimates of turbulent microscales in the nonlinear case, giving $\alpha = 3/4$ [13].

Suppose further that hyper-diffusion is added to an advection-diffusion problem for additional numerical smoothing, then it is desirable that the minimum scale h_4 attained by the latter satisfies $\ell_4 < \ell$, equivalently $Pe_4^{-\beta} < Pe^{-\alpha}$, which when $\alpha = \beta$ implies $\kappa_4 < \kappa L^2$, and hence $Pe_4 > Pe_h/N^2$.

Remembering that $L = Nh$ and $Pe = U_0 L/\kappa$, then $h < LPe^{-\alpha}$ gives

$$h < N^{\frac{1-\alpha}{\alpha}} \frac{\kappa}{U_0}, \quad \text{ie.} \quad Pe_h < N^{-1+\frac{1}{\alpha}}, \quad \text{implying} \quad c_h < d_h N^{-1+\frac{1}{\alpha}} \tag{28}$$

Similarly for hyper-diffusion

$$h^3 < N^{\frac{1-3\beta}{\beta}} \frac{\kappa_4}{U_0}, \quad \text{ie.} \quad Pe_{4h} < N^{-3+\frac{1}{\beta}}, \quad \text{implying} \quad c_h < d_4 N^{-3+\frac{1}{\beta}} \tag{29}$$

The results Equations(28) and (29) are key. From Equation (28) it follows that stability of the explicit scheme implies accurate advection-diffusion for all reasonable values of α. Moreover there is benefit from use of implicit schemes, even on uniform meshes if $\alpha < 1$, since then accuracy is possible even when $c_h \gg 1$.

Whether accuracy is achieved in a time dependent calculation, depends on the smallest important timescale τ. The work of Hunt et al. [13] implies $\tau = \ell^2/\kappa$, ie. then $\Delta t < \tau$ implies $d_h < 1$ since $h < \ell$ has been assumed, so that temporal accuracy may also be consistent with $c_h \gg 1$. (For hyperdiffusion, $d_4 < \kappa_4/(\kappa h^2)$, so if $\kappa_4 < \kappa L^2$, $d_4 < N^2$.) However if $\tau = \ell/U_0$, which could occur if there were say large initial transients, then for minimal accuracy in the time advance, $\Delta t < \tau$ implies $c_h < 1$.

5 Numerical issues with anisotropic diffusion problems

5.1 Anisotropic diffusion - simple FIREDRAKE treatment

5.1.1 Straight magnetic fieldlines case: null space problem

This section presents a case similar to that in the paper by Deluzet and Narski [14, § 3.2]. From their p.12, they have an anisotropy corresponding to $\epsilon = 10^{-15}$ below.

Consider the diffusion problem

$$\frac{\partial^2 u}{\partial x^2} + \epsilon \frac{\partial^2 u}{\partial y^2} = -\epsilon. \tag{30}$$

The corresponding magnetic field lines in this case are straight and horizontal, ie. the more-diffusive direction lies parallel to the x-axis. The domain is a unit square and the boundary conditions are homogeneous Dirichlet at $y = 0, 1$ and homogeneous Neumann at $x = 0, 1$, where 'homogeneous' implies zero values for function and normal gradient respectively. The solution is $u = \frac{1}{2}y(1 - y)$ and the problem is singular as $\epsilon \to 0$ because then any function of y that satisfies the Dirichlet conditions is a solution, ie. there is a null space in this limit.

This problem is implemented in FIREDRAKE. Using a unit square domain discretized into 20×20 first-order Classical Galerkin elements, the solver finds the correct solution until ϵ is approx. 10^{-14}, below that the quality of the solution degrades, see Fig.15. The nature of the failure is not that the solution takes excessive time to obtain but rather than the numerics appear to converge to the wrong answer. It is found that the problem is worse at higher-order, with order-2 elements only remaining good down until about $\epsilon = 10^{-13}$ and order-6 down to about $\epsilon = 10^{-11}$. The problem seems to level off for sufficiently high order, since order-22 gives correct results at $\epsilon = 10^{-10}$ but not at 10^{-11}. All these cases run in seconds on the 20×20 mesh.

The singular limit can be removed by replacing the homogeneous Neumann conditions with a Dirichlet condition corresponding to the analytic solution. In this case there is not a null space for $\epsilon \to 0$, and it is found that this case works fine even for $\epsilon = 10^{-20}$ (and indeed, only one of the constant-x boundaries needs to be Dirichlet).

Exploring the parameter space by taking $\epsilon = 0$ gave the result that the solver returned a everywhere-zero solution without throwing any errors. Again, the replacement of one or more of the Neumann boundary conditions with Dirichlet meant that the solver gave the correct solution.

5.1.2 Curved magnetic fieldlines case: locking

Now consider the case with non-straight magnetic field lines. The anisotropy field is given by calculating the flux as, with $k_\parallel = 1$ and $k_\perp = \epsilon$,

$$F = k_\parallel \hat{b}\, \hat{b} \cdot \nabla u + k_\perp \left(\nabla u - \hat{b}\, \hat{b} \cdot \nabla u \right) \tag{31}$$

using the unit vector

$$\hat{b} = \frac{1}{\sqrt{1 + \lambda^2 x^2 y^2 (1-x)^2 (1-y)^2}} \left(1, \lambda x y (1-x)(1-y)\right) \tag{32}$$

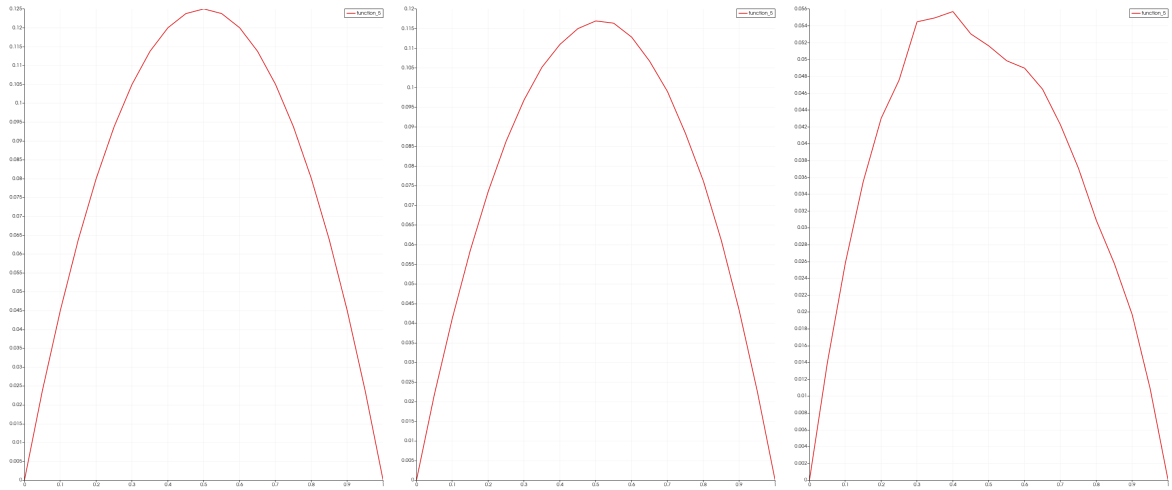


Figure 15: Anisotropic diffusion solution as a function of y (analytic solution is $u = \frac{1}{2}y(1-y)$), L-R is $\epsilon = 1$, 10^{-14}, 10^{-15}; straight magnetic fieldlines case.

to define the direction of the magnetic field, ie. the more-diffusive direction. The fieldlines for this configuration are exhibited in Fig.16.

With the above definitions, the diffusion equation is $\nabla \cdot F = -\epsilon$. Apart from the curved fieldlines, the equation is as it was in the preceding section, and is identical if the fieldline-bending parameter $\lambda = 0$. The interaction with the boundary is as before, ie. the field lines are parallel to the $y = 0, 1$ (Dirichlet) boundaries and normal to the $x = 0, 1$ (Neumann) ones.

Selecting $\lambda = 10$ and $\epsilon = 10^{-6}$ an accurate solution is obtained by running the solver with order-5 elements, see Fig.16, where the results presented compare well with order-6 elements.

The paper [14] indicates that there is an problem additional to the null-space issue, if the finite-element function spaces do not contain functions that are constant in the direction of the fieldlines in the anisotropic case, and also that this problem manifests itself for less extreme ϵ values than those needed to reveal the null-space problem described in the preceding subsection (which is, of course, still present for the curved magnetic fieldlines case). Deluzet and Narski refer to this as 'locking', which is apparently a known problem for low-order finite-element expansions.

The paper [14] indicates that there is an problem additional to the null-space issue, if the finite-element function spaces do not contain functions that are constant in the direction of the fieldlines in the anisotropic case, and also that this problem manifests itself for less extreme ϵ values than those needed to reveal the null-space problem described in the preceding subsection (which is, of course, still present for the curved magnetic fieldlines case). Deluzet and Narski refer to this as 'locking', which is apparently a known problem for low-order finite-element expansions. 'Locking' is is again a case of numerics converging to the wrong answer; as stated in [14], for low-order elements the solution amplitude is much smaller than it ought to be, see the right-hand plot in Fig.16. In this case, increasing the element order quickly fixes the problem, as expected from ref [15], and it does not currently seem necessary to examine how rapidly the situation is remedied by using meshes with a larger number of low-order elements.

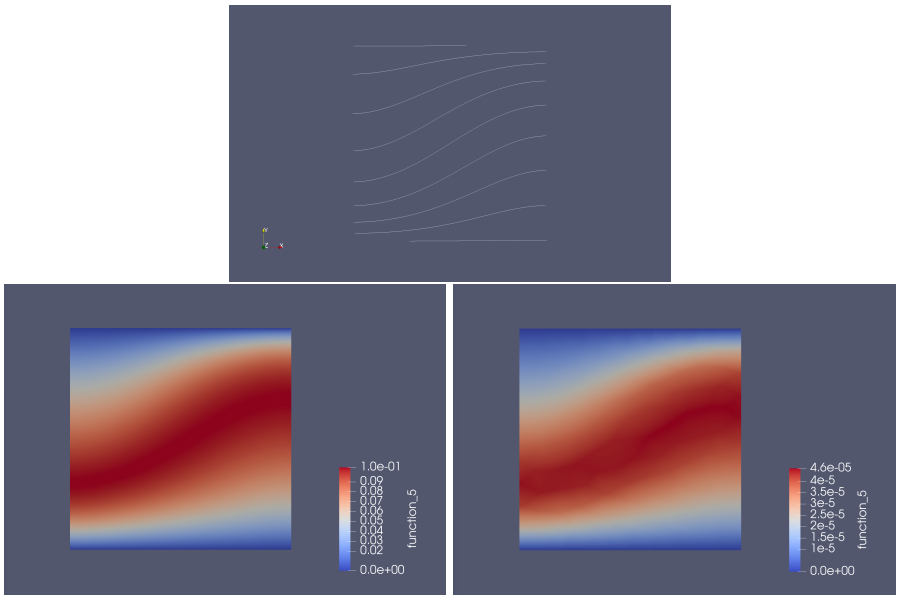


Figure 16: Anisotropic diffusion, $\epsilon = 10^{-6}$, in case with curved magnetic fieldlines as shown at top. Order-5 elements were used to produce the left-hand picture, order-1 to produce the right-hand contours, which although they have the correct general shape correspond to much too small an amplitude. This example used the same triangular mesh as in Fig.17, but the over-plotting of the mesh has been suppressed in order to bring out small differences between the solutions.

Note that changing the Neumann boundary condition for a Dirichlet condition ($\frac{1}{2}y(1-y)$ was used for this test), at both or even at one of the two boundaries somewhat unexpectedly makes the problem go away. This indicates that locking might be avoided by careful choice of boundary conditions.

5.2 A Dirac equation-like formulation of anisotropic diffusion

Consider once more the anisotropic diffusion problem.

$$\frac{\partial^2 u}{\partial x^2} + \epsilon \frac{\partial^2 u}{\partial y^2} = -\epsilon. \tag{33}$$

Since u is a scalar, this will be referred to as the 'scalar case' in spite of the fact that the diffusivity is described by an anisotropic tensor. The domain is $[-0.5, 0.5]^2$, a different unit square displaced relative to that used before. The magnetic field lines are straight, ie. the more-diffusive direction lies parallel to the x-axis. The boundary conditions are homogeneous Dirichlet at $y = -0.5, 0.5$ and homogeneous Neumann at $x = -0.5, 0.5$. The solution is $u = \frac{1}{8} - \frac{1}{2}y^2$ and the problem is singular as $\epsilon \to 0$ because then any function of y that satisfies the Dirichlet conditions is a solution, ie. there is a null space in this limit.

A fairly crude triangle mesh is used as shown in Fig.17, with second-order Classical Galerkin elements. The numerics are seen to give the correct answer until ϵ falls below a critical value in the range $[10^{-14}, 10^{-15}]$, with the answer clearly wrong for smaller ϵ.

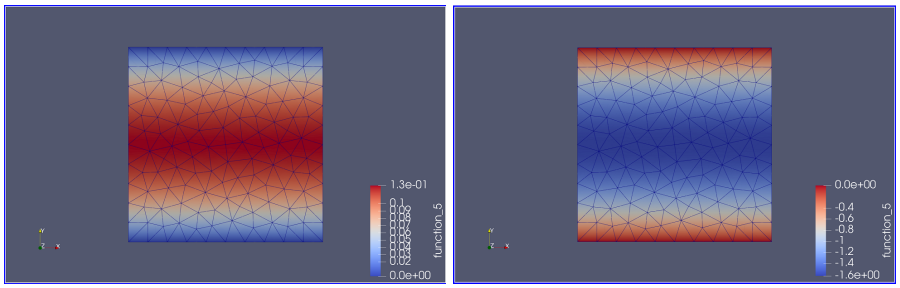


Figure 17: Outputs of script for $\epsilon = 10^{-14}$ (left, correct) and $\epsilon = 10^{-15}$ (right, incorrect). Note that smaller values of ϵ result in the solution becoming increasingly negative.

Now consider the same problem reformulated as a spin-half, Dirac-like equation [16, § 22], noting in particular the appearance of a square root:

$$\left(\Gamma^x \frac{\partial}{\partial x} + \sqrt{\epsilon}\,\Gamma^y \frac{\partial}{\partial y}\right)\psi = \psi_0. \tag{34}$$

With the choice

$$\Gamma^x = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{35}$$

and

$$\Gamma^y = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{36}$$

the two matrices obey the relation $\Gamma^a\Gamma^b + \Gamma^b\Gamma^a = 2g^{ab}I$, which is the Clifford algebra for 2-D rotations, ie. they anticommute when $g^{ab} = 0, \;\; a \neq b$ and separately square to unity.

The two-component spinors are

$$\psi = \begin{pmatrix} u \\ v \end{pmatrix} \tag{37}$$

and

$$\psi_0 = \begin{pmatrix} f \\ g \end{pmatrix}. \tag{38}$$

Spinor theory then implies $f = g = -\sqrt{\epsilon}\, y$, and the spin-half Dirac form follows as

$$\begin{pmatrix} \frac{\partial}{\partial x} & \sqrt{\epsilon}\,\frac{\partial}{\partial y} \\ \sqrt{\epsilon}\,\frac{\partial}{\partial y} & -\frac{\partial}{\partial x} \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = -\sqrt{\epsilon}\, y \begin{pmatrix} 1 \\ 1 \end{pmatrix}. \tag{39}$$

The advantages of the above formulation are that now the system is first-order and the ϵ in the original equation is replaced by $\sqrt{\epsilon}$ so the anisotropy is expected to be less of a problem.

The above system Equation (39) is implemented as a simple FIREDRAKE script. Promising outputs are obtained for the triangle mesh as used in the scalar case described above, see Fig.18. As the anisotropy is increased in the spin-half equation, it is found that for $\epsilon = 10^{-20}$ the numerical solution is still good, but that it is becoming unreliable at $\epsilon = 10^{-22}$; see Fig.19. If we take $\epsilon = 10^{-20}$ as the limit of effective operation of the solver, it is possible to claim a factor of greater than 10^5 improvement in the accessible range of ϵ over the scalar solver in this case.

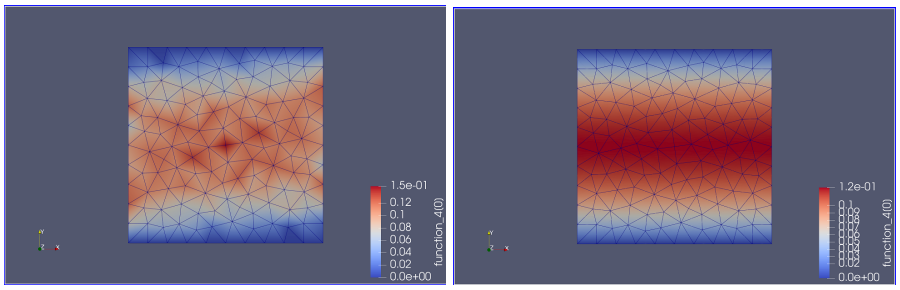


Figure 18: Outputs of the spin-half equation for a mesh of triangular first-order (left) and second-order (right) elements; $\epsilon = 1$.

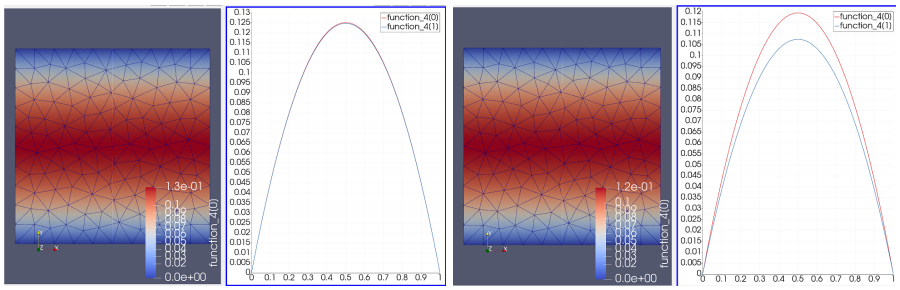


Figure 19: Outputs of the spin-half equation for a mesh of triangular second-order elements $\epsilon = 10^{-20}$ (left) and 10^{-22} (right). The latter is beginning to diverge from the correct solution. (Do not be misled by the labelling of the coordinate axis as $[0, 1]$ instead of $[-0.5, 0.5]$ caused by a PARAVIEW quirk.)

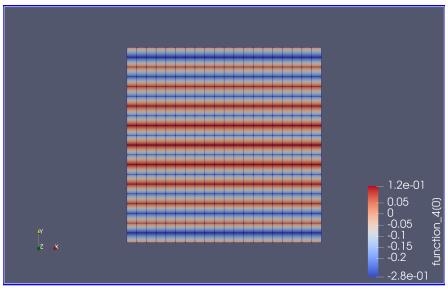


Figure 20: Output of the spin-half equation for a mesh of first-order elements, 20×20 uniform squares, $\epsilon = 1$, indicating a problem for the approach.

Unfortunately the Dirac form appears to lead to incorrect results if square Classical Galerkin elements are used, even for $\epsilon = 1$, see Fig.20. This failure is the most concerning of a number of issues concerning the spinor approach, which extend to unexpected sensitivity to the order of element used, and to the need for a formulation to treat curved magnetic fieldlines. More detailed investigations might yet lead to understanding and overcoming of these difficulties, but they could be paused, since for the project to proceed, it is only strictly necessary to estimate quantitatively the level of numerical diffusion. Should it become necessary to proceed with the approach, scripts and documentation for this work have been placed in the repository [17].

6 Summary

This report has presented work towards 3-D plasma turbulence simulations using NEKTAR++. Results of an on-going, incremental development strategy have been presented, showing progress towards the development of a solver capable of addressing problems of current research interest. Its outputs will be coupled to the UKAEA's particle modelling capability in order to include kinetic effects from, in particular, the neutral particle simulation capability, to be described in subsequent NEPTUNE reports.

The current report has also outlined work towards understanding the coupling between different finite element solvers and mitigating the numerical difficulties associated with demanding advection-diffusion problems and highly anisotropic diffusion problems. Lines of enquiry which were begun, but then subsequently paused in the light of ongoing discoveries have been recorded.

Acknowledgement

The support of the UK Meteorological Office and Strategic Priorities Fund is acknowledged. ET acknowledges assistance from the rest of the UKAEA NEPTUNE team, and also from the holders of Grants T/AW085/22 (particularly David Moxey and Chris Cantwell), T/AW087/22 (particularly Hussam al-Daas) and T/AW088 (particularly Mike Kryjak).

A Higher-dimensional harmonics

This work builds on the work on harmonic functions in the report [18], particularly § 2.4, and also, for an introduction to their relationship with fluid vortices, see Appendix B. The motivation for Section A.1 and Section A.2 is to gain greater analytic understanding of certain types of fluid flow that could produce additional test cases. To improve understanding, it is important to achieve informative visual representations, see Section A.3.

A.1 Two-dimensional harmonics and complex numbers

The understanding of functions of a complex variable is aided by effective means of visualizing them. This is complicated by the fact that a complex function $w(z) = u(x, y) + iv(x, y)$ is a two-dimensional function of a two-dimensional variable; separate contour or 3-D plots of u and v contain in principle all the information about the function but do not give a feel for how it all fits together. The utility of plots of the complex phase as a function of (x, y) is advocated in eg. [19].

Another appealing way of visualizing a complex function is to note that any function of the complex conjugate, $w(\bar{z})$ (*aka* anti-holomorphic function), is a 2-D harmonic in the Hodge sense in the (x, y)-plane, ie. it represents a divergence- and curl-free two dimensional fluid flow, referred to as the Pòlya vector field. This gives the simple picture that the magnitude of w is the local flow speed and its phase gives the local direction of the flow. Zeros in the complex plane correspond to stagnation points of the flow, and the relationship has many other important consequences, such as the picture of dipoles as a merger of two simple poles [18, App. B], and the fact that flows corresponding to the neighbourhood of an essential singularity exhibit all possible velocity values infinitely many times, by the Great Picard Theorem, see Henrici [20, Theorem 4.4g, Ex. 5].

An illustration of the Pòlya vector field is the simple vortex given by $w = \frac{i}{\bar{z}}$. Observe that the multiplication of this function by complex i switches between vortex solutions and draining bathtub solutions - the 90^o phase of i causes the local direction of the flow field to rotate through a right angle, hence the vortex flow and the draining bathtub flow are given by the one-forms:

$$\begin{aligned} u_0 &= \frac{-ydx + xdy}{x^2 + y^2} \\ u_1 &= \frac{-xdx - ydy}{x^2 + y^2}. \end{aligned} \tag{40}$$

Intermediate values of the phase give a draining or filling bathtub solution where the flow lines have a certain amount of 'twist' around the drain / source. Another example is the function:

$$w(\bar{z}) = \frac{\bar{z}}{1 + \bar{z}} \tag{41}$$

for which the corresponding flow field is shown in Fig.21 along with a plot showing the magnitude of the velocity. The flowfield for the Taylor series, truncated to 10 terms, and expanded about $\bar{z} = 0$ is shown also, along with the corresponding velocity magnitude. These plots show the expected agreement in the flowfields within the unit circle of convergence. The zeros of the Taylor polynomial can be seen to be clustering about the circle of convergence, in accordance with Jentzsch's theorem.

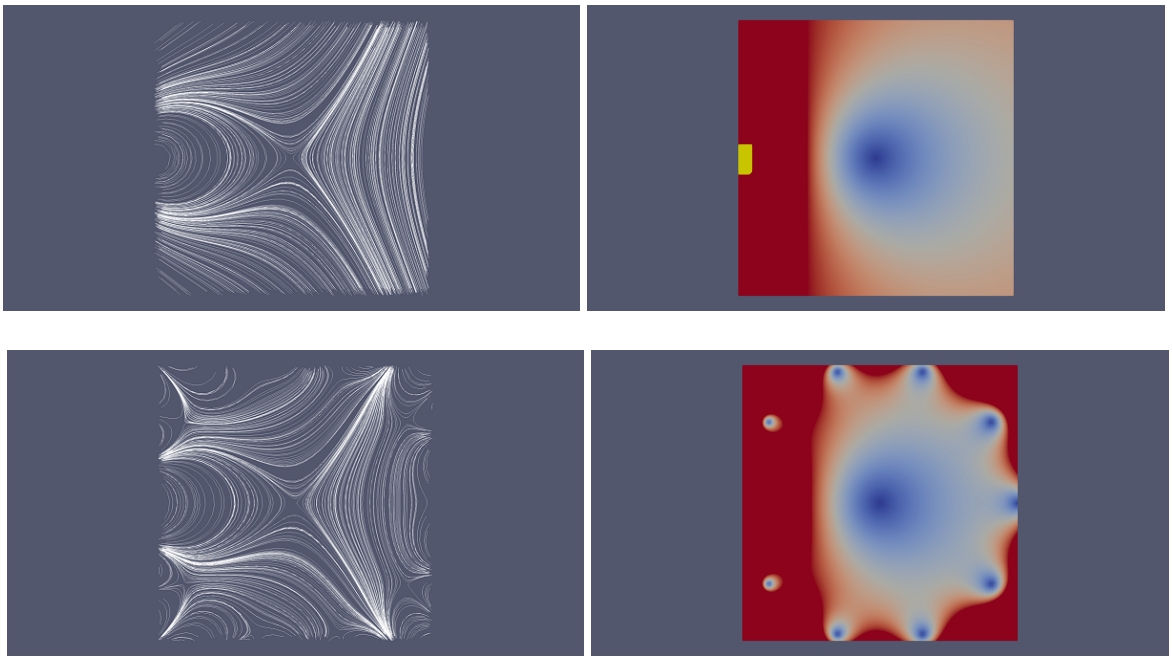


Figure 21: Streamlines at left and flow speed at right corresponding to at top, $w(\bar{z}) = \frac{\bar{z}}{1+\bar{z}}$ and at bottom, the tenth-order Taylor expansion of $w(\bar{z})$ about $\bar{z} = 0$. The flow speed scales are common. Note the yellow area in the top RHS plot is a plotting artifact due to the singularity.

Noting the identity

$$\nabla(\mathbf{u}^2) \equiv 2\left(\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{u}\times(\nabla\times\mathbf{u})\right) \tag{42}$$

and using the fact that the Pólya vector field is curl-free, there follows Bernoulli's principle:

$$\nabla\left(\frac{1}{2}\mathbf{u}^2 + p\right) = 0 \tag{43}$$

Applied to the simple vortex, for which

$$\mathbf{u} = \left(\frac{-y}{x^2+y^2}, \frac{x}{x^2+y^2}\right). \tag{44}$$

the pressure follows as

$$p = -\frac{1}{2}\frac{1}{x^2+y^2}. \tag{45}$$

which is lower near the centre of the vortex as expected.

Another interesting relation for time-stationary harmonic flows follows from their satisfaction in two dimensions of the inertial law

$$\mathbf{u}\cdot\nabla\mathbf{u} = -\nabla p. \tag{46}$$

Writing out in Cartesians $\mathbf{u} = (u, v)$, using

$$\begin{aligned} \frac{\partial u}{\partial x} &= -\frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y} &= \frac{\partial v}{\partial x} \end{aligned} \tag{47}$$

and taking the divergence, gives the eikonal equation

$$\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 = -\frac{1}{2}\nabla^2 p. \tag{48}$$

This relation implies a method of finding solutions to the eikonal equation in the case where the field is known to be harmonic.

When the pressure is specified the squared modulus of the holomorphic velocity field is known which means the vector field is defined up to a local rotation, ie. to within an arbitrary factor of complex phase $\exp^{i\varphi}$. However, the argument φ of the phase function must be purely real must also be holomorphic or the harmonic property is lost, implying that the rotation is global, since the only pure real holomorphic functions are constants. An example is the vortex / bathtub solution family introduced in ref [18, App. B].

It is worth noting here that as well as the anti-holomorphic functions, ie. functions of $\bar{z}$ considered above, holomorphic functions, ie. functions of z, also play a role in hydrodynamics. The simplest example is $w(z) = -iz$, which corresponds to the linear field in the terminology of Arter [21], viz. $\mathbf{u} = (y, -x)$ in the (x, y)-plane, which brings non-zero vorticity into play. Its streamlines correspond to the phase-space representation of an harmonic oscillator.

A.2 Four-dimensional harmonics and quaternions

The equations for harmonics are [18, § 2.4]

$$\begin{aligned} du &= 0, \\ \delta u &= 0. \end{aligned}$$

where the harmonic u is a one-form. In the two-dimensional Euclidean space previously considered, du represents the curl and δu the divergence of a 2-vector u, but henceforth the space is four-dimensional Euclidean space.

Evaluating du where $u = \phi\, dw + \psi \cdot dx$ gives the two-form

$$\begin{aligned} du = &- \left(\frac{\partial \phi}{\partial x} dw \wedge dx + \frac{\partial \phi}{\partial y} dw \wedge dy + \frac{\partial \phi}{\partial z} dw \wedge dz\right) \\ &+ \left(\frac{\partial \psi_2}{\partial x} - \frac{\partial \psi_1}{\partial y}\right) dx \wedge dy + \left(\frac{\partial \psi_3}{\partial y} - \frac{\partial \psi_2}{\partial z}\right) dy \wedge dz + \left(\frac{\partial \psi_1}{\partial z} - \frac{\partial \psi_3}{\partial x}\right) dz \wedge dx \\ &+ \frac{\partial \psi_1}{\partial w} dw \wedge dx + \frac{\partial \psi_2}{\partial w} dw \wedge dy + \frac{\partial \psi_3}{\partial w} dw \wedge dz. \end{aligned}$$

so that $du = 0$ is equivalent to

$$\nabla \phi = \frac{\partial \psi}{\partial w} + \nabla \times \psi. \tag{49}$$

Evaluating $\delta u \equiv *d*u$, taking

$$\begin{aligned} *dw &= -dx\wedge dy\wedge dz, \\ *dx &= dy\wedge dz\wedge dw, \\ *dy &= -dz\wedge dw\wedge dx, \\ *dz &= dw\wedge dx\wedge dy, \end{aligned}$$

gives the equation $\delta u = 0$ as

$$\left(-\frac{\partial\phi}{\partial w}-\frac{\partial\psi_1}{\partial x}-\frac{\partial\psi_2}{\partial y}-\frac{\partial\psi_3}{\partial z}\right) dw\wedge dx\wedge dy\wedge dz = 0. \tag{50}$$

which is the equation of a vanishing four-divergence

$$\frac{\partial\phi}{\partial w} = -\nabla\cdot\psi. \tag{51}$$

Now, it is known that a left-regular (right-regular in $+$ sign case) quaternion function satisfies (see eg. [22])

$$\frac{\partial\phi}{\partial w} = \nabla\cdot\psi, \tag{52}$$

$$\nabla\phi = -\frac{\partial\psi}{\partial w}\mp\nabla\times\psi. \tag{53}$$

Thus, the differential form equations Equations(49) and (52) have solutions corresponding to the conjugate $\phi - i\psi$ of a left-regular function of a quaternion variable $q \equiv w + ix + jy + kz$.

A simple quaternion function (left- and right-regular) is

$$r(q) = \frac{1}{q} \equiv \frac{w-(xi+yj+zk)}{(w^2+x^2+y^2+z^2)^2}. \tag{54}$$

which corresponds to the harmonic

$$u = \frac{wdw+xdx+ydy+zdz}{(w^2+x^2+y^2+z^2)^2}. \tag{55}$$

Post-multiplications by i, j, k respectively generate the harmonics (with the progenitor first):

$$\begin{aligned} u_0 &= \frac{wdw+xdx+ydy+zdz}{(w^2+x^2+y^2+z^2)^2}, \\ u_1 &= \frac{xdw-wdx+zdy-ydz}{(w^2+x^2+y^2+z^2)^2}, \\ u_2 &= \frac{ydw-zdx-wdy+xdz}{(w^2+x^2+y^2+z^2)^2}, \\ u_3 &= \frac{zdw+ydx-xdy-wdz}{(w^2+x^2+y^2+z^2)^2}. \end{aligned} \tag{56}$$

This is the quaternionic analogue of obtaining the draining bathtub by multiplying by i to get the vortex. There are thus three quaternion analogues of the vortex solution, separately corresponding to a rotation about one of the three axes in a Cartesian three-space. (Pre-multiplication does not generate harmonics.)

A.3 Visualization of 4-D harmonics

Visualization of quaternionic functions is obviously even more of a challenge than that of complex functions. A projection may be attempted from the four-dimensional vector field into a three-dimensional subspace. Here a stereographic projection, equivalent to a conformal representation from an embedded three-sphere onto the $w = 0$ subspace, is performed.

First is derived the coordinate transform from the 3-sphere $w^2 + x^2 + y^2 + z^2 = a^2$ to the $w = 0$ subspace: The mapped point is where the line connecting the 'North Pole' $(a, 0, 0, 0)$ and the point with coordinates (w, x, y, z) intersects the $w = 0$ subspace $(0, x', y', z')$. The formula for the new coordinates is derived from

$$(a, 0, 0, 0) + \lambda(w - a, x, y, z) = (0, x', y', z') \tag{57}$$

ie.

$$\begin{aligned} x' &= \frac{ax}{a - \sqrt{a^2 - x^2 - y^2 - z^2}}, \\ y' &= \frac{ay}{a - \sqrt{a^2 - x^2 - y^2 - z^2}}, \\ z' &= \frac{az}{a - \sqrt{a^2 - x^2 - y^2 - z^2}}. \end{aligned} \tag{58}$$

(Do not get confused and assume $w = 0$ - the correct version is $w' = 0$.) The covector field is harmonic 1 and its scaling factor may be neglected as this is constant on the 4-sphere, thus

$$u = (x, -w, z, -y). \tag{59}$$

The transformation law for this type of object is the covariant form

$$u_{a'} = \frac{\partial x^a}{\partial x^{a'}} u_a, \tag{60}$$

where the indices range over $1, 2, 3$ since the mapping is from the 3-sphere to the equatorial 3-space. The derivatives of the pre-transformation coordinates with respect to the post-transformation ones are required. They may be obtained more easily by squaring the coordinate relations and summing to form

$$r'^2 = \frac{a^2 r^2}{(a - \sqrt{a^2 - r^2})^2}. \tag{61}$$

where r and r' are defined by the expected Pythagorean formulae. Writing $\Delta \equiv a - \sqrt{a^2 - r^2}$ the algebra simplifies to give $\Delta = \frac{2a^3}{r'^2 + a^2}$ and the relations

$$\begin{aligned} w &= a\frac{r'^2 - a^2}{r'^2 + a^2}, \\ x &= \frac{2a^2 x'}{r'^2 + a^2}, \\ y &= \frac{2a^2 y'}{r'^2 + a^2}, \\ z &= \frac{2a^2 z'}{r'^2 + a^2}. \end{aligned} \tag{62}$$

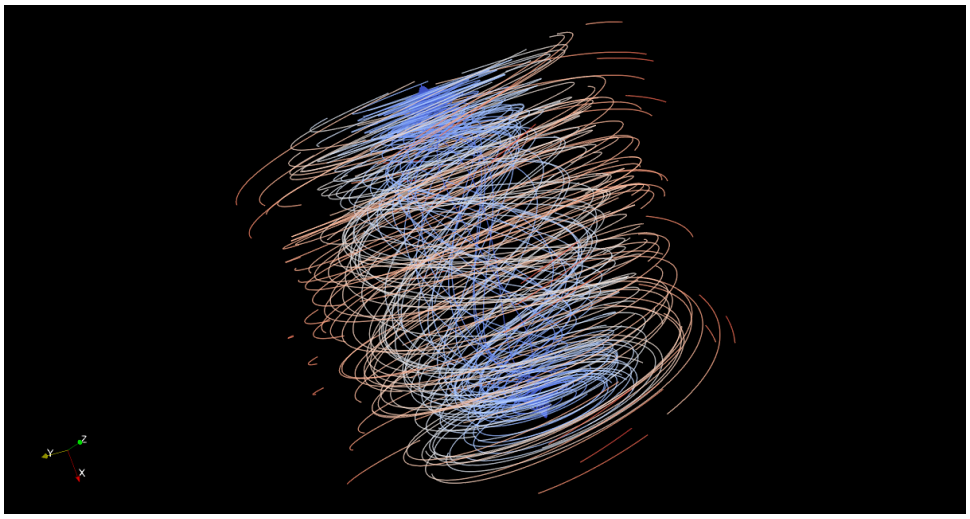


Figure 22: Streamlines for the 4-D field projected into 3-D as discussed in the text. The plot was performed on a cubical volume, hence streamlines intersecting the boundary of the cube appear to terminate.

whence it now straightforward to perform the coordinate transformations on the components of the 1-form.

The vector field in the (x', y', z') 3-subspace resulting from the above transformations can be visualized as a streamline plot using PARAVIEW. Thus for example, the harmonic u_1 in Eq. (56), transformed from (x, y, z) to (x', y', z') using the covariant transformation law, gives the following components:

$$\begin{aligned} u_{x'} &= (-x'^2 + y'^2 + z'^2 + a^2)(a^2 - x'^2 - y'^2 - z'^2) \\ u_{y'} &= 2x'y'(x'^2 + y'^2 + z'^2 - a^2) + 2az'(x'^2 + y'^2 + z'^2 + a^2)) \\ u_{z'} &= 2x'z'(x'^2 + y'^2 + z'^2 - a^2) - 2ay'(x'^2 + y'^2 + z'^2 + a^2)) \end{aligned}$$

Streamlines of the above vector field are plotted in Fig.22 on a cube of side $2a$.

B Advanced visualization: Nvidia IndeX PARAVIEW plugin

Rendering of volumetric fields in 3-D involves the processing of large amounts of data. Nvidia offers *Nvidia IndeX* for visualizing volumetric data, see [23], which includes an example showing the visualization of 150TB of data. This software is, at the time of writing, offered free of charge for a single workstation or for academic use. There is also a plugin for PARAVIEW (v5.11 and later).

A workflow for a simple demonstration of the plugin is:

- Open PARAVIEW.

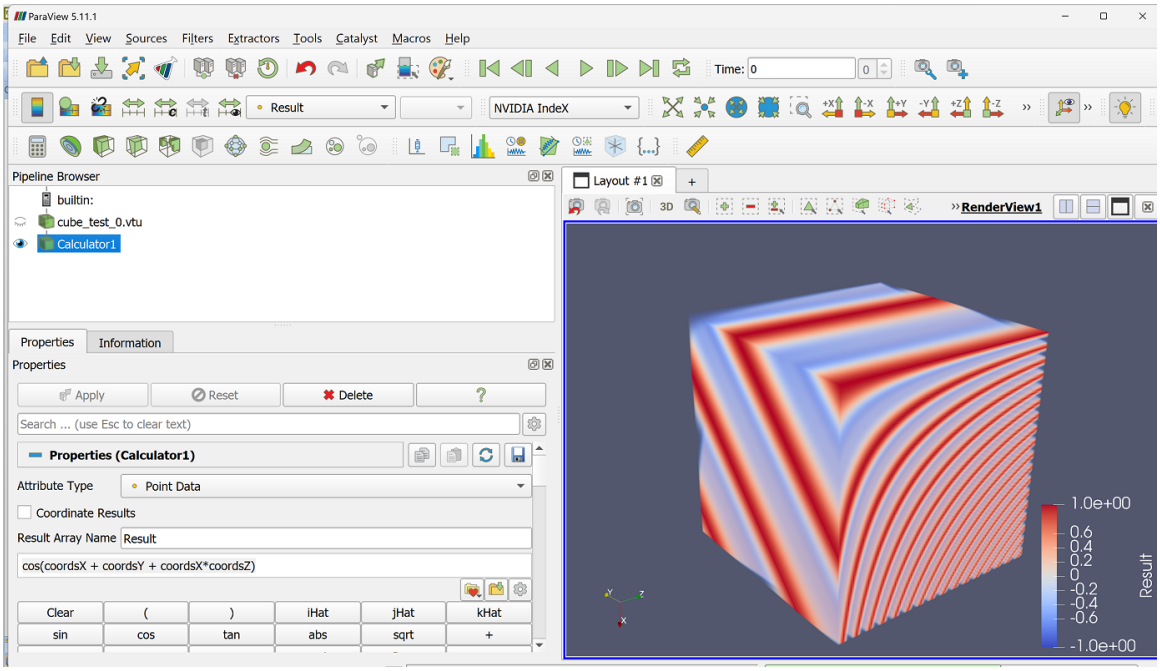


Figure 23: Output of the NVIDIA IndeX test outlined in the text.

- Go to tools/manage plugins; load pvNVIDIAIndeX.
- Open .vtu file. Click Apply.
- Add a calculator to input a test function in 3-D. One possible example is `cos(coordsX + coordsY + coordsX*coordsZ)`.
- Visualize as `Volume` to render volumetric data using the default renderer.
- Switch visualization to `NVIDIA IndeX` to render using *Nvidia IndeX*.

The performance using the plugin was found to be significantly faster than the default renderer in the case of the cube-shaped domain used for Fig.23 and using the plugin allowed the visualization to be rotated and zoomed in real time (the default renderer shows a defeatured visualization during rotate / zoom). The plugin also produced slightly higher-quality renderings than the default. It was, however, found that the plugin appeared slower than the default renderer in one case that involved an elongated domain.

C A plotting bug in ParaView

A simple Firedrake script (Fig.24) was used to integrate the trivial ODE $u'(x) = -\lambda u(x)$, $u(0) = 1$. It was found that the ParaView `Plot Over Line` filter worked as expected for first-order CG elements but produced strange artifacts when higher-order CG elements were used. The artifacts are not evident if the same data is output to a .csv file and plotted by other methods, see Fig.25.

```
from firedrake import *

mesh = IntervalMesh(20,0,1)

V = FunctionSpace(mesh, "CG", 3)  # why don't higher orders work seem to properly?
u = TrialFunction(V)
v = TestFunction(V)
lam = Constant(1.0)
zee = Constant(0.0)

a = (grad(u)[0]*v + lam*u*v)*dx
L = zee*v*dx

g = Function(V)

bc = DirichletBC(V, 1.0, 1)

solve(a==L, g, bcs=bc)

File("first_order_ode.pvd").write(g)
```

Figure 24: FIREDRAKE script for solving the ODE in the text.

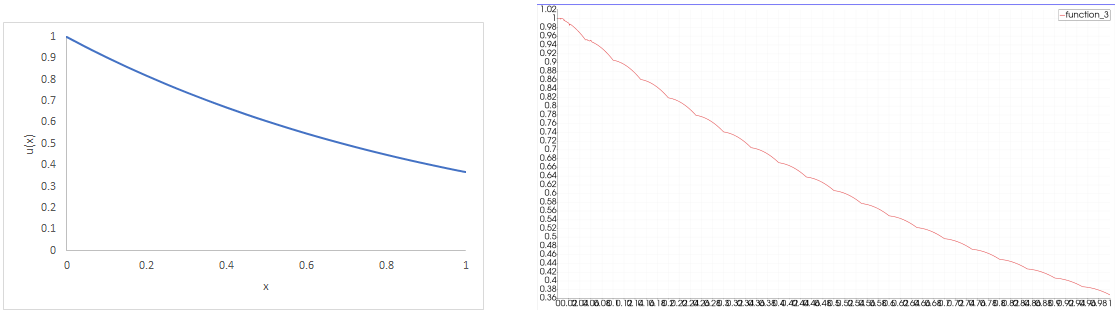


Figure 25: FIREDRAKE outputs for the numerical solution of the ODE in the text using second-order elements. The left-hand plot was generated by exporting the data to .csv and plotting in *Excel*; the right-hand plot used the `Plot Over Line` filter of PARAVIEW and is clearly incorrect.

The concerning aspect of the bug is that it causes the appearance of features that indicate a particular numerical method is not working correctly. In short, caution must be applied when using higher-order plots generated by `Plot Over Line` when working with one-dimensional data.

This bug was observed in version 5.10.1 of PARAVIEW on Windows. Since an earlier version (5.8.0) was seen to crash every time when attempting to run `Plot Over Line` for orders higher than one, there is the suggestion that the relevant part of the software is still actively being developed, so that reporting the problem has been deferred as unlikely to speed a fix.

References

[1] Nektar-Driftwave. https://github.com/ExCALIBUR-NEPTUNE/nektar-driftwave. Accessed: March 2023.

[2] Saunders W., E. Threlfall, and W. Arter. Finite Element Models: Performance. Technical Report CD/EXCALIBUR-FMS/0047, UKAEA Project Neptune, 2021.

[3] Nektar-Driftplane. https://github.com/ExCALIBUR-NEPTUNE/nektar-driftplane. Accessed: March 2023.

[4] Hermes-3 2D. https://hermes3.readthedocs.io, 2021. Accessed: June 2021.

[5] E. Threlfall, O. Parry, and W. Arter. Finite Element Models Complementary Actions: Code Integration, Integration and Operation 3. Technical Report CD/EXCALIBUR-FMS/0074, UKAEA Project Neptune, 2023.

[6] LArge Plasma Device (LAPD). https://plasma.physics.ucla.edu/large-plasma-device.html. Accessed: September 2023.

[7] W. Arter et al. Equations for EXCALIBUR/NEPTUNE Proxyapps. Technical Report CD/EXCALIBUR-FMS/0021-1.31-M1.2.1, UKAEA, 10 2023. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/ukaea_reports/CD-EXCALIBUR-FMS0021-1.30-M1.2.1.pdf.

[8] E. Threlfall, Powell S., and W. Arter. Complementary Actions: Uncertainty Quantification Code Integration, Acceptance and Operation 1. Technical Report CD/EXCALIBUR-FMS/0073, UKAEA Project Neptune, 2023.

[9] F. Rathgeber, D. A. Ham, L. Mitchell, M. Lange, F. Luporini, A.T.T. McRae, G.-T. Bercea, G.R. Markall, and P.H.J. Kelly. Firedrake: automating the finite element method by composing abstractions. *ACM Transactions on Mathematical Software (TOMS)*, 43(3):1–27, 2016.

[10] A. Toselli and O. Widlund. *Domain Decomposition Methods: Algorithms and Theory*. Springer, 2004.

[11] D.A. Nield. The Rayleigh-Jeffreys problem with boundary slab of finite conductivity. *Journal of Fluid Mechanics*, 32(2):393–398, 1968. https://doi.org/10.1017/S0022112068000790.

[12] W. Arter. Numerical simulation of magnetic fusion plasmas. *Reports on Progress in Physics*, 58:1–59, 1995. http://dx.doi.org/10.1088/0034-4885/58/1/001.

[13] J.C.R. Hunt, I. Eames, J. Westerweel, P.A. Davidson, S. Voropayev, J. Fernando, and M. Braza. Thin shear layers-the key to turbulence structure? *Journal of Hydro-environment Research*, 4(2):75–82, 2010.

[14] F. Deluzet and J. Narski. A two field iterated asymptotic-preserving method for highly anisotropic elliptic equations. *Multiscale Modeling & Simulation*, 17(1):434–459, 2019.

[15] E.T. Meier, V.S. Lukin, and U. Shumlak. Spectral element spatial discretization error in solving highly anisotropic heat conduction equation. *Computer Physics Communications*, 181(5):837–841, 2010. https://doi.org/10.1016/j.cpc.2009.12.018.

[16] M. Fecko. *Differential geometry and Lie groups for physicists*. Cambridge University Press, 2006.

[17] Spin-half repository. https://github.com/ethrelfall/spin-half/tree/main. Accessed: August 2023.

[18] E. Threlfall. Finite Element Models Complementary Actions 2. Technical Report CD/EXCALIBUR-FMS/0064-M6.2, UKAEA, 3 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/ukaea_reports/CD-EXCALIBUR-FMS0064-M6.2.pdf.

[19] L.N. Trefethen's review of Visual Complex Functions: an Introduction with Phase Portraits by Elias Wegert. https://people.maths.ox.ac.uk/trefethen/wegert_review_SIREVDec13.pdf. Accessed: August 2023.

[20] P. Henrici. *Applied and computational complex analysis. Volume 1*. John Wiley & Sons, New York, 1974.

[21] W. Arter. Beyond Linear Fields: the Lie-Taylor Expansion. *Proc. Roy. Soc. A*, 473:20160525, 2017. http://dx.doi.org/10.1098/rspa.2016.0525,http://arxiv.org/abs/1606.08763.

[22] C.A. Deavours. The Quaternion Calculus. *The American Mathematical Monthly Vol.80 No.9 (Nov. 1973) pp.995-1008*, 1973.

[23] Nvidia IndeX. https://developer.nvidia.com/nvidia-index. Accessed: September 2023.