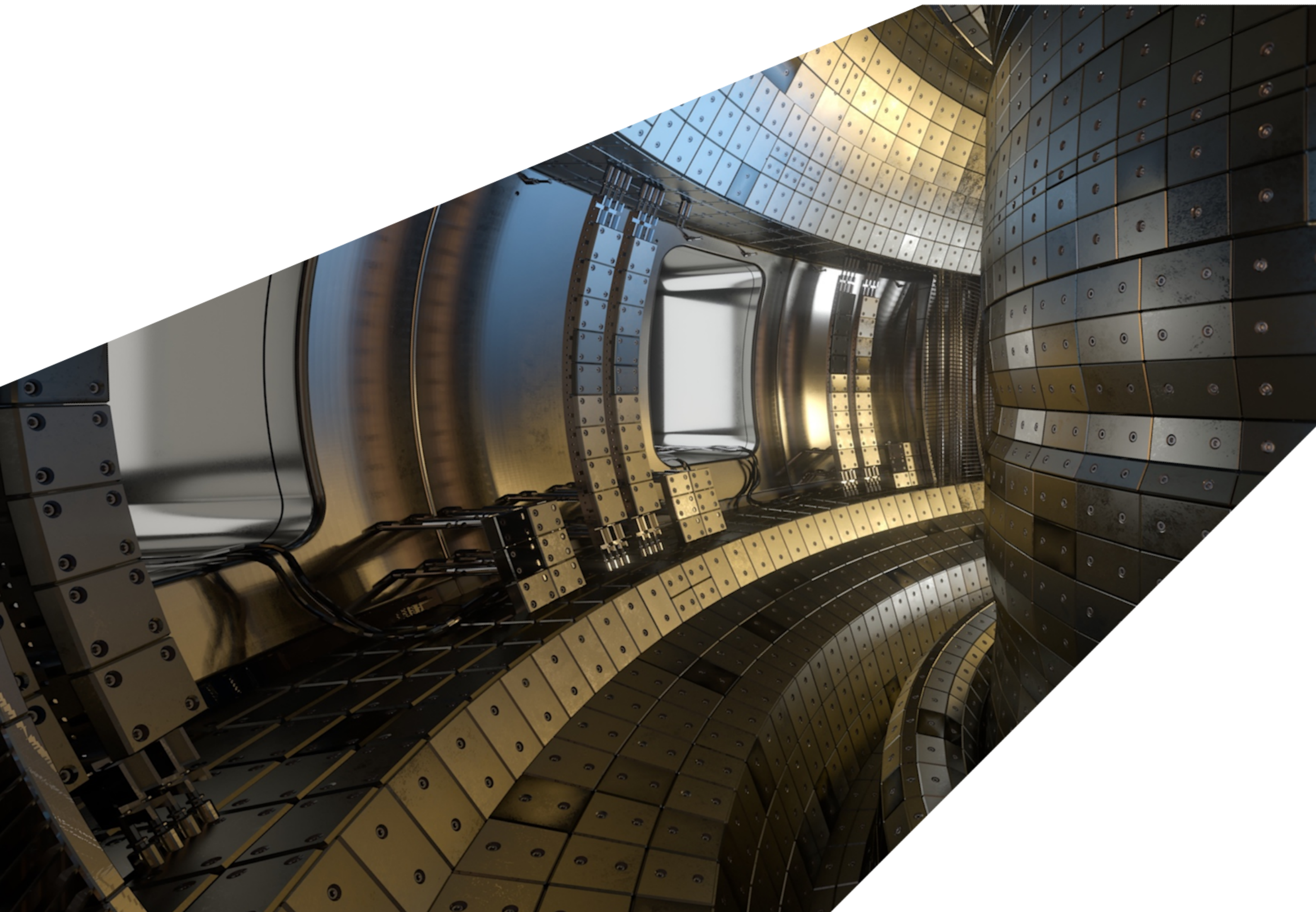
UK Atomic Energy Authority

ExCALIBUR

Complementary actions. Code-coupling, physics databases and benchmarking techniques 2.

M7c.2 Version 1.00

Abstract
The report describes work for ExCALIBUR project NEPTUNE at Milestone M7c.2. Complementary actions are described which assist code-coupling and the use of benchmarking techniques and physics databases, generally providing support to more than one Y4/Y5 deliverable. Code-coupling activities have extended to the provision of information needed to ensure successful collaboration between different sites, such as introductory technical material, consistent use of physical units, text processing tools designed to enforce uniformity of document production and improvements to the developers' website. Work by the grantees at York, on the development of benchmark problems, DSLs and multi-species physics, and at STFC on relevant preconditioning and timestepping numerical techniques, is summarised. Internal work on atomic physics databases is annexed as a separate report.



UKAEA REFERENCE AND APPROVAL SHEET

	Client Reference:	
	UKAEA Reference:	CD/EXCALIBUR-FMS/0075
	Issue:	1.00
	Date:	23 March 2023
Project Name: ExCALIBUR Fusion Modelling System		

	Name and Department	Signature	Date
Prepared By:	Wayne Arter Matthew Barton Owen Parry BD	N/A N/A N/A	23 March 2023 23 March 2023 23 March 2023
Reviewed By:	Wayne Arter Project Technical Lead		23 March 2023

1 Introduction

This work for Deliverable 7c is part of a programme of work items designed to provide an uninterrupted programme of coordination work providing support to more than one Y4/Y5 deliverable. The procurement of software support to ensure smooth continuation of external work was described in the preceding M7c.1 Report [1]. Work on the production of classes and interfaces to atomic physics data for collisions and radiation formed a separate Deliverable 8c, the note on which appears as an annex to the present report.

Work conducted internally by UKAEA is described in Section 2. Aside from routine maintenance tasks such as posting and indexing reports received from grantees after acceptance, this is efficiently described under four subsection headings, namely (1) educational material in Section 2.1, (2) technical material in Section 2.2, (3) text processing tools in Section 2.3, and (4) improvements to the developers' website Section 2.4. Work described in 21 reports accepted from the external grantees during the year Y4, is summarised in Section 3.

2 Task Work by UKAEA

2.1 Educational Material

Introductory material has been drafted to demonstrate important features of timestepping and preconditioning algorithms relevant to Project NEPTUNE. In due course it is intended that this “SIAM Review” style material will be made public along with reports already available internally to UKAEA (such as the guide to finite elements aimed at people who have formerly only used finite difference schemes).

The timestepping material shows how different adaptive strategies perform on a model problem of simple harmonic motion, for third and fourth order accurate updating. The effects of adaptation using absolute and relative error are compared. It is pointed out that the cancellations from step-to-step occurring in quadrature-like schemes when fixed timestep is used, may make these schemes unexpectedly competitive.

The preconditioning material examines the structure of the inverses of matrices arising in 1-D models of diffusion, advection and advection-diffusion. Advection is here supposed treated both by centred and upwind schemes, especially the Patankar scheme used by state-of-the-art finite difference codes for the tokamak edge. Since the aim of preconditioning matrices is approximately to apply the matrix inverse, the properties of the inverse matrix are plausibly very important for the behaviour of the preconditioned iterations. It emerges that for matrices representing advection, there is strong dependence of the properties of the inverse on stagnation point location. There may also be a large difference in condition number between very similar formulations, remarkably indeed whether the discretisation uses an odd or an even number of grid points. The structure of inverse matrices for diffusion is much less variable, highlighting the much greater challenge presented by advection-dominated compared to diffusion problems.

2.2 Technical Material

2.2.1 Physical Units and Internal Scalings

As far as is compatible with numerical stability and accuracy, NEPTUNE will employ SI units, with the main exception that temperature may be specified in electron-Volts eV. Algorithmically the exception is easily handled by always writing kT for temperature T where the dimensional quantity k ensures that the product kT has units of energy. (So $k = |e|$ if T is given in eV or Boltzmann’s constant if T is measured in Kelvin.) A further practical exemption arises from the need to interface with CAD systems, which typically work in millimetres mm, where the recommendation is to scale geometry to metres as soon as is practicable.

NEPTUNE will *not* accept models expressed in dimensionless units because of the potential for confusion when a range of different physical time and length-scales are involved. It is of course, entirely reasonable for theorists to pick scalings appropriate to their particular problem, and indeed to incorporate purely numerical factors in the scalings to simplify the coefficients of the equations, and hence their analytic work. Why this causes problems for NEPTUNE can be understood by considering the situation when a user or developer wants to introduce an additional physical effect not considered by the original theorist, for then the newcomer needs to know precisely what was done before, and then has to scale terms representing the new effect before their introduction into the software. It is worthwhile emphasising the word ‘precisely’ - the inclusion of purely numerical factors in the scaling may well introduce ambiguity. Moreover, there are only a small number of fundamental dimensional quantities, namely time, length, charge and mass, so a limited number of ways in which terms can be non-dimensionalised unambiguously. (Temperature in Kelvin and angular quantities are dimensionless fundamental quantities.)

An example of the problem of ambiguity in the scaling, occurs when augmenting Magnetohydrodynamics (MHD) with electrostatic phenomena. In MHD, scaling magnetic field by a reference value B_0 causes no difficulty, but as soon as electric charge is introduced and scaled, typically by the charge on the electron $|e|$, since in SI units magnetic induction has unit 1 T= $kgs^{-1}C^{-1}$, there may arise the ambiguity as to whether B_0 or say $m_u/(|e|t_s)$ is appropriate for non-dimensionalising a new magnetic field term.

Internally, however, NEPTUNE needs to scale fields so that their values are expressible efficiently in floating point, meaning that large exponents of either sign are to be avoided. The suggested limit is to exponents smaller than 38 in absolute value, a criterion set by the standard single precision representation of floating-point numbers. This is expected to be best achieved by non-dimensionalising, but ‘under the hood’, ie. with minimal user interaction. There will be allowed one permissible exception to the rule that no purely numerical factors should appear in scalings, namely to introduce a factor N_{ref} to reduce number densities to more reasonable magnitudes, and a factor w_{ref} to allow for the representation of many plasma particles by a single computational super-particle, in effect to allow for the smallness of an individual particle’s charge and mass. With that, the key scalings are

1. time in units of t_s,
2. length in units of L_s,
3. mass in units of m_u, the atomic mass unit
4. electric charge in units of $|e| = |q_e|$, the absolute value of the charge on the electron
5. number densities such as n_e made dimensionless with respect to an additional factor of N_{ref} as well as L_s^{-D} where D is the spatial dimensionality of the problem
6. superparticles carry weight w_{ref}, eg. a charge $w_{ref}q_e$ is carried by each ‘super-’electron

The NEPTUNE website has a section on “Physical properties of the edge plasma” which gives likely values for t_s and L_s. Equivalently L_s and the velocity scale U_s may be specified instead, thus depending on the relevant physics $U_s = 10^3 - 10^7\,ms^{-1}$, $L_s = 10^{-5} - 10\,$m. These two scalings could be set by the user (together with N_{ref} and w_{ref}), *once* for a given simulation. In keeping with the application to the Exascale, the pure number $N_{ref} = 10^{18}$, whilst taking $w_{ref} = 10^{10}$ seems most convenient. In practice, the ratio of absolute electron charge to atomic mass unit ($\simeq$ proton mass) appears frequently enough to warrant its own symbol $K_M = \frac{|e|}{m_u} \approx 10^8$, eg. the electric field unit becomes $\frac{U_s^2}{LsK_M}$ and the magnetic field unit becomes $\frac{U_s}{L_sK_M}$.

The next subsections Section 2.2.2, Section 2.2.3 and Section 2.2.4 illustrate the application of the proposed approach to the treatment of the Braginskii transport coefficients and to charged particle motion in a magnetic field.

2.2.2 Introduction to Scaling

The Branginskii coefficients might be used in a multispecies context over a range of different timescales. Ultimately finite element codes, indeed most numerical schemes, depend on solution of a linear algebra problem

$$A\mathbf{x} = \mathbf{b} \tag{1}$$

where $\mathbf{x}$ are field values to be determined, and A and $\mathbf{b}$ follow from the model. Take the simplest case of a 2×2 model

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \quad \mathbf{b} = (b_1 \;\; b_2)^T, \quad \mathbf{x} = (x_1 \;\; x_2)^T \tag{2}$$

where the demands of multispecies and multiscale may lead to widely different values of all coefficients and unknowns. To make these values comparable, which is desirable from the numerical standpoint, exploit the fact that it is possible to multiply each each equation separately without changing the value of the solution $\mathbf{x}$. Further, provided the coefficients a_{ij} are appropriately adjusted, each component x_j of $\mathbf{x}$ may be scaled separately. Thus

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \tag{3}$$

and

$$\begin{pmatrix} s_1 a_{11} t_1 & s_1 a_{12} t_2 \\ s_2 a_{21} t_1 & s_2 a_{22} t_2 \end{pmatrix} \begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{pmatrix} = \begin{pmatrix} \tilde{b}_1 \\ \tilde{b}_2 \end{pmatrix} \tag{4}$$

where

$$\tilde{\mathbf{x}} = (x_1/t_1 \;\; x_2/t_2)^T, \quad \tilde{\mathbf{b}} = (s_1 b_1 \;\; s_2 b_2)^T \tag{5}$$

give the same $\mathbf{x}$, although likely by solving equations with very different sized coefficients for very different sized unknowns $\tilde{\mathbf{x}}$. This approach is written more elegantly in matrix notation as

$$SA\boldsymbol{T}^{-1}\boldsymbol{T}\tilde{\mathbf{x}} = S\tilde{\mathbf{b}} \tag{6}$$

where S and $\boldsymbol{T}$ are diagonal matrices

$$S = \begin{pmatrix} s_1 & 0 \\ 0 & s_2 \end{pmatrix}, \quad \boldsymbol{T} = \begin{pmatrix} 1/t_1 & 0 \\ 0 & 1/t_2 \end{pmatrix} \tag{7}$$

(so that $\boldsymbol{T}$ may be viewed as an elementary preconditioner).

Scaling each row of the matrix and each unknown becomes very expensive for a nonlinear and/or timestepping problem where A and $\mathbf{b}$ are large and continually changing. Thus it is helpful on this account only to scale by equation subsystem and by field, where by subsystem is meant eg. momentum balance for species X, and field is meant say number density of the coupled species Y. Hence it is convenient to regard x_1 as corresponding to an entire vector of discrete values of the number density of species X and x_2 similarly to represent the density of species Y, so that the coefficients a_{ij} for each i and each j separately represent a possibly very large matrix. Such a grouping is convenient for making the equations dimensionless, in that s_1 may compensate for the dimensions of the equation

$$a_{11}x_1 + a_{12}x_2 = b_1 \tag{8}$$

and s_2 for the dimensions of

$$a_{21}x_1 + a_{22}x_2 = b_2 \tag{9}$$

whereas t_j may serve make dimensionless x_j for each $j = 1,\ 2$. There should be a reduced, hopefully avoidable, need to rescale equations as fields evolve, and properly dimensioned values should be easily restored as needed.

2.2.3 Fluid Models

In the above context, there is no need, having evaluated the Braginskii coefficients to convert them to non-dimensional form. Selected terms from the energy evolution equation demonstrate how the above scalings are to implemented in practice. Most likely, an equation for energy evolution would be considered instead of one divided by m_i, and for demonstration purposes, the factor of B can be omitted from System 2-3 [2]. The following terms in the energy evolution equation are representative

$$\frac{\partial}{\partial t}\left(\frac{3}{2}(NkT_d)\right) + \ldots = N\frac{\partial}{\partial s_{\|}}\kappa_{e\|}\frac{\partial kT_e}{\partial s_{\|}} + \ldots \tag{10}$$

The overall scaling of this equation to render its terms dimensionless is evidently $\frac{t_s}{kT_s} \cdot \frac{L_s^3}{N_{ref}}$ and the unknowns to be solved for, T_d and N, have respectively dimensions/scalings T_s and $\frac{L_s^3}{N_{ref}}$. The time dependent term in Equation (10) involves both unknowns, hence needs re-expressing as

$$\frac{\partial}{\partial t}\left(\frac{3}{2}(NkT_d)\right) = \frac{3}{2}Nk\frac{\partial T_d}{\partial t} + \frac{3}{2}kT_d\frac{\partial N}{\partial t} \tag{11}$$

so that $\frac{3}{2}Nk$ and $\frac{3}{2}kT_d$ are equivalents of coefficients a_T and a_N of matrix A. The partial derivative $\partial/\partial t$ leads to discrete operators $\tilde{\Delta}_t/\Delta t$ where $\tilde{\Delta}_t$ is a (dimensionless) difference operator so the scaling factorisations have

$$\frac{\frac{3}{2}Nk}{\Delta t}\tilde{\Delta}_t(T) \to \frac{L_s^3 t_s}{N_{ref}kT_s} \cdot \frac{\frac{3}{2}Nk}{\Delta t} \cdot T_s\tilde{\Delta}_t\left(\frac{T}{T_s}\right) \tag{12}$$

and

$$\frac{\frac{3}{2}kT_d}{\Delta t}\tilde{\Delta}_t(N) \to \frac{L_s^3 t_s}{N_{ref}kT_s} \cdot \frac{\frac{3}{2}kT_d}{\Delta t} \cdot \frac{N_{ref}}{L_s^3}\tilde{\Delta}_t(n) \tag{13}$$

where $\tilde{\Delta}_t$ operates on unknowns of dimensionless temperature T/T_s and density $n = L_s^3 N/N_{ref}$ respectively. The partial derivative $\partial^2/\partial s_{\|}^2$ leads to a discrete operator $\tilde{\Delta_s^2}/(\Delta s_{\|})^2$ where $\tilde{\Delta}_s$ is a (dimensionless) difference operator so the scaling factorisation for a_κ is

$$\frac{L_s^3 t_s}{N_{ref}kT_s} \cdot \frac{Nk\kappa_{e\|}}{(\Delta s_{\|})^2} \cdot T_s\tilde{\Delta_s^2}\left(\frac{T}{T_s}\right) \tag{14}$$

where $\tilde{\Delta_s^2}$ operates on the unknown of dimensionless temperature T/T_s.

The algorithm above producing the scaled matrix coefficients involves multiplying 3 factors together. If T is measured in eV, then $k = |e|$. Suppose $L_s = 10\,\mathrm{m}$ and $t_s = L_s/10^5\, ms^{-1} = 10^{-4}\, s$, $N = 10^{16}\, m^{-3}$, $T = 10\, eV$, and approximating $|e| \to 10^{-19}$, $3/2 \to 1$, taking $\Delta t = 0.1 t_s$ and $\Delta s_{\|} = 0.1 L_s$, then numerically

$$\frac{L_s^3 t_s}{N_{ref} k T_s} \cdot \frac{Nk}{\Delta t} \cdot T_s \to 10^{3-4-18+19-2} \cdot 10^{-19+16+5} \cdot 10^2 \tag{15}$$

$$\frac{L_s^3 t_s}{N_{ref} k T_s} \cdot \frac{kT_d}{\Delta t} \cdot \frac{N_{ref}}{L_s^3} \to 10^{3-4-18+19-2} \cdot 10^{-19+1+5} \cdot 10^{18-3} \tag{16}$$

$$\frac{L_s^3 t_s}{N_{ref} k T_s} \cdot \frac{Nk\kappa_{e\|}}{(\Delta s_{\|})^2} \cdot T_s \to 10^{3-4-18+19-2} \cdot 10^{16-19} \kappa_{e\|} \cdot 10^2 \tag{17}$$

so that each factor is not unmanageably different from unity when using single precision arithmetic. (The largest numerical value is 10^{15} if $\kappa_{e\|}$ is estimated as $10^9\, m^2 s^{-1}$.)

Ultimately, the scaled values of the coefficients a_T, a_N and a_κ are easily shown to be equal to respectively

$$\tilde{a}_T = \frac{t_s n}{\Delta t}, \ \ \tilde{a}_N = \frac{t_s}{\Delta t}\frac{T_d}{T_s}, \ \ \tilde{a}_\kappa = \frac{t_s \kappa_{e\|}}{(\Delta s_{\|})^2} n \tag{18}$$

which are all evidently of order unity.

If it is necessary to iterate to a solution for T_e or T_i, then the coefficient should be evaluated for $T = T_s$. For example $\kappa_{e\|}$, which varies proportional to $T^{5/2}$, should be evaluated as

$$\kappa_{e\|}(T) = \kappa_{e\|}(T_s) \left(\frac{T}{T_s}\right)^{\frac{5}{2}} \tag{19}$$

and then any further work, for example linearisation, may proceed using the dimensionless group T/T_s.

2.2.4 Scalings for Particle Models

The above considerations also apply in the case of particle evolution. Given the choice of units, if the equations of motion are posed in terms of superparticles of weight w_p,

$$w_p m_\alpha \frac{d\mathbf{v}}{dt} = w_p q_\alpha (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \tag{20}$$

It is convenient to introduce a reference particle weight w_{ref}, cf. N_{ref}, to make scalings more manageable numerically. The overall equation scaling to render the terms dimensionless is then evidently $\frac{t_s}{w_{ref} m_u} \cdot \frac{t_s}{L_s} = \frac{t_s^2}{w_{ref} L_s m_u}$ and the unknown to be solved for, namely $\mathbf{v}$, has dimensions $\frac{L_s}{t_s}$. Consider the acceleration in Equation (20), which leads to discrete coefficient terms of the form $\frac{m_\alpha}{\Delta t}$, then the scaling factorisation is

$$\frac{w_p m_\alpha}{\Delta t}\mathbf{v} \to \frac{t_s^2}{L_s w_{ref} m_u} \cdot \frac{w_p m_\alpha}{\Delta t} \cdot \frac{L_s}{t_s} \tag{21}$$

to be applied to dimensionless velocity

$$\mathbf{v} \to \frac{t_s}{L_s}\mathbf{v} \tag{22}$$

As explained above, the numerical values of the 3 factors should be estimated. For definiteness, suppose $|\mathbf{v}| = 10^5\, ms^{-1}$, $B = 1$ T, then the gyro-radius is approx. $10^{-3}\, m$, so take $L_s = 10^{-3}$ m as a typical lengthscale, then $t_s = 10^{-3-5} = 10^{-8}$ s, and a timestep size Δt of order 10^{-9} is indicated. Approximate the amu$\to$ 10^{-27} kg, and if $A = Z = 1$ and suppose $w_p = w_{ref} = 10^{10}$, the resulting factors follow as

$$\frac{t_s^2}{L_s w_{ref} m_u} \cdot \frac{w_p m_\alpha}{\Delta t} \cdot \frac{L_s}{t_s} \to 10^{-16+3-10+27} \cdot 10^{10-27+9} \cdot 10^{-3+8} \tag{23}$$

so that each factor is not unmanageably different from unity when using single precision arithmetic. (The smallest numerical value is 10^{-8}.)

The scaled coefficient is easily shown to simplify as

$$\tilde{a} = \frac{t_s^2}{w_{ref} m_u L_s} \cdot \frac{w_p m_\alpha}{\Delta t} \cdot \frac{L_s}{t_s} = \frac{t_s A}{\Delta t} \tag{24}$$

if $w_{ref} = w_p$, which is clearly of order unity. Evidently $\tilde{a}$ has a numerical value of order the number of timesteps chosen to discretise a gyro-period, typically a number of order unity. The magnetic field term may similarly be treated

$$w_p |q_\alpha \mathbf{v} \times \mathbf{B}| \to \frac{t_s^2}{L_s w_{ref} m_u} \cdot w_p |q_\alpha| B \cdot \frac{L_s}{t_s} \tag{25}$$

whence

$$\frac{t_s^2}{L_s w_{ref} m_u} \cdot w_p |q_\alpha| B \cdot \frac{L_s}{t_s} \to 10^{-16+3-10+27} \cdot 10^{10-19} \cdot 10^5 \tag{26}$$

ie. the smallest numerical factor is 10^{-9}, and

$$\tilde{a} = t_s Z K_M |B| \approx 10^{-8+8} = 10^0 = 1 \tag{27}$$

2.3 Text processing tools

The tools are only available on the private CCFE gitlab site. Their wider publication awaits user testing and feedback. For UKAEA staff with access to CCFE gitlab, the tools are to be found either in directory `tex` or in its subdirectory `importools`. They are largely based on Linux `sed` or `vim` editors. The more mature tools appear as bash shell scripts and are as follows:

1. In directory tex
   - skeltorp1.bash produces a report directory skeleton from arguments <milestone> <FMS report number> <title string>
   - pub.bash is transmitted by skeltorp1.bash into subdirectories as pub$N$.bash script that produces final versions of pb$N$ as a PDF file with name appropriate for web posting.

2. In subdirectory tex/importools

   - fromed.bash converts text file produced by Windows editor to Linux-friendly ASCII, enforcing many of the NEPTUNE text conventions
   - getsymb.bash looks for definitions of mathematics variables and produces file `allp.var.srt` for user inspection and intervantion
   - md2tex.bash converts Markdown produced by PANDOC to LaTeX
   - sear.bash finds all the files in the list in file `tex.lis` which contain a specified string
   - toctortf.bash produces WYSIWYG version of a report's table of contents from LaTeX .toc file output, useful for .rtf conversion

Subdirectory tex/importools also contains .ed files containing `vim` edit commands, many of which are invoked by the bash scripts. Two other of these files help produce .bib files. If XXXbib.inp contains the essential bibliographic information, then

```
cp XXXbib.inp XXX.bib
vim XXX.bib -S YYYbib.ed
```

will over-write XXX.bib with Bibtex-readable references. The string "YYY" may be either

- `reportplus` when XXX.bib will contain bibliographic entries corresponding to a grantee's report (see file examplebib.inp for an example of the required input format)
- `ukaeareport` when XXX.bib will contain bibliographic entries corresponding to a UKAEA report (see file ukaeareportbib.inp for an example of the required input format)

Other .ed files are briefly described in the README, beware that the input is invariably overwritten so should be copied before editing.

2.4 Improvements to the Developers' Website

Consistent with the conclusions of the grantees' report [3] about the inadequacy of ReadThe-Docs, work has continued to produce a website capable of aiding and encouraging development of NEPTUNE software by a community spread across UKAEA and UKRI. During the year, the developers' website has been subject to a re-examination as to how best the pages might be produced from LaTeX source, resulting in the replacement of LATEX2HTML by LWARP, since the latter package appears to be maintained far more actively, supports cascading style sheets (css) so websites look more attractive, enables download of LaTeX source of mathematics via MathJax, and offers an option to produce a single PDF document of the entire website. The webpages have now been made publicly available on GitHub [4], and can be viewed on mobile phones and tablets as well as larger screens.

Webpage source material has been separated from the private CCFE gitlab repository and has, like the webpages, been posted on GitHub, specifically as one of the repos forming the ExCALIBUR-NEPTUNE organisation [5]. A feature has been added whereby a named git branch can be used to produce a correspondingly named set of developers' webpages, see Section 2.4.1. Bibliographic .bib files, and files describing mathematical notation (symbols) and listing acronyms have been frequently updated throughout the year, and need to be kept synchronised as explained in Section 2.4.2. Many minor improvements have been made to the site, notably the mathematical notation used in the developers' webpages has been changed to conform with the list of symbols.

2.4.1 Website branches

The instructions to produce a separate set of developers' webpages corresponding to a particular branch of repo [5] are

1. Create a branch which is copy of `main`.
2. Commit changes to that branch.
3. This branch of the developer website should become available online within about 5 minutes assuming a successful build process by the Github actions workflow 'Deploy static content to Pages' , by accessing the repo branch described as Github-Pages.
4. After testing, the new branch should be deployed by merging the branch with the main branch

2.4.2 Synchronising key webpages

Key pages (as .tex files) common to the CCFE gitlab ExCALIBUR repo may be checked for synchrony using the script checksync.bash (currently only available on branch `wa` of the developers' website). Successful working of the script relies on the simultaneous presence in the user's disc-space of the CCFE gitlab repo as directory `excalibur-wa` and the webpages' source repo as directory `devweb`. The script pre-processes the gitlab files to conform to the format required for web-posting then performs a simple comparison. It is up to the user to ensure that any differences are reconciled, eg. by use of Linux `meld`.

3 Task Work by Grantees

Each section summarises reports received from a grantee, one report per subsection, with a link in the title to the location of the accepted report. Accepted reports are held on the Documents repository of the ExCALIBUR-NEPTUNE organisation on GitHub, to which belong other repos containing software developed under project NEPTUNE. Note that some of these repos may be access-controlled, please email `neptune@ukaea.uk` if difficulties are encountered in seeing the material.

3.1 Reports received under Grant T/NA083/20, PO 2047356

The York grantee produced the following reports in association with KCL

- 2047356-TN-06-1 - Task 1.4 Preconditioning 2D elliptic solvers [6]
- 2047356-TN-07-1 - Task 2.2 BOUT++ 1D fluid solver with realistic boundary conditions : implementation [7]
- 2047356-TN-08-1 - Task 2.4 Non-intrusive UQ with BOUT++ 1D fluid solver [8]
- 2047356-TN-09-1 - Task 4.1 Test cases for 1D multispecies plasma model [9]
- 2047356-TN-10-1 - Task 0.2 Development of a performance testing and aggregation tool for NEPTUNE proxy-apps [10]
- 2047356-TN-11-1 - Task 3 Considerations for 1D1V models Report covering tasks 3.1, 3.2 and 3.3 [11]
- 2047356-TN-12-2 - (D1.3): Elliptic solvers within Nektar++ [12]
- 2047356-TN-13 - Spatially 2-D plasma model incorporating velocity space effeects [13]
- 2047356-TN-14-1 - DSL and code design pattern for NEPTUNE [14]
- 2047356-TN-15-2 - Task 2.5 Performance considerations for non-intrusive uncertainty quantification and the influence of the boundary [15]
- 2047356-TN-16-1 - Task 0.3 Community and environment [3]
- 2047356-TN-17 - Progress on implementation of system 2-6 equations [16]

3.1.1 Report 2047356-TN-06 [6]

This report simply indicates that the work was done in conjunction with STFC and appears in STFC's reports refs [17, 18, 19, 20].

3.1.2 Report 2047356-TN-07 [7]

This report summarises the implementation of system 2-3 from the Equations document [2] using the 1-D model SD1D including fluid neutrals and realistic boundary conditions. The model and approach is described in the SD1D manual which forms an appendix, providing a reference implementation of a 1-D fluid solver with realistic boundary conditions for use in later work, see ref [8]. The report also discusses test cases selected in earlier work, for which carefully commented input files are included with the source code in the SD1D repository [21]. The timing results presented demonstrate a factor 10 or more reduction in time to solution when using preconditioning techniques.

3.1.3 Report 2047356-TN-08 [8]

The focus of UQ in NEPTUNE has shifted in favour of *non*-intrusive uncertainty quantification (UQ) since the first date of writing the equations document [22], with which shift this report duly conforms. It summarises experience gained in the application of non-intrusive UQ to a sample test case from the SD1D website [21] described in ref [23]. It showcases the EasyVVUQ package to facilitate the construction of UQ workflows. The work provides a testing ground for the practical application of UQ to more expensive realistic plasma simulations. Useful, clear example scripts are included, as are links to relevant repositories detailing the work of the BOUT++ team at the recent VECMA hackathons [24].

This report opens with an executive summary stating the scope: non-intrusive UQ using EasyVVUQ for a one-dimensional SD1D plasma simulation with sheath boundary conditions. An introductory section contains brief motivation for sensitivity analysis and UQ, then introduces the EasyVVUQ toolkit and the major object types (Encoder, Decoder, Campaign, Sampler). Section 3 details the specific use case of SD1D within the EasyVVUQ framework. Section 4 presents details of the SD1D test problems and how the EasyVVUQ framework was applied, with simulations done on a single node of Archer2. Useful cautionary notes are included eg. inefficient resource usage with `local_execute` if some parts of sampled parameter space result in outlying long-duration simulations. Example Python scripts for specifying a UQ / sensitivity analysis campaign using polynomial chaos (PCE) on a simple neutral-free problem are exhibited, which should be very useful for pedagogical purposes (as should be the scripts in Appendix A for the work outlined in Section 5). Section 5 presents UQ for a more realistic plasma model problem including neutrals. Subsection 5.1 includes discussion of a PCE-sampler approach and Subsection 5.2 a stochastic collocation approach which has the advantage of being able to sample adaptively based on which input has the most influence on the uncertainty, and also the ability to terminate the campaign based on convergence of Sobol indices (it is mentioned that this approach may not always be appropriate), though at the cost of not being easily parallelizable. Section 6 concludes the report, noting that extensions to the examples provided are necessary in order to manage campaigns involving more demanding HPC simulations but that suitable machinery is becoming increasingly available among the outputs of the VECMA project, notably EasyVVUQ. Overall, this report provides an instructive comparison of some of the merits and shortcomings of the PCE and stochastic collocation approaches to UQ embodied in EasyVVUQ.

3.1.4 Report 2047356-TN-09 [9]

This very brief report is primarily a link to places where work performed on Hermes-3 may be found, namely its manual (readthedocs) pages [25], and the Hermes-3 [26] and SD1D [21] repos on GitHub. The work has extended the 1-D single plasma model to treat (1) separate ion and electron temperatures, (2) neutral gas models including different excited states of Hydrogen, and (3) impurity species, including both low atomic number, for which each charge state forms a separate species, and high atomic number where different states are bundled together to be represented by a single fluid.

3.1.5 Report 2047356-TN-10 [10]

This report outlines the requirements for, and subsequent design of, a software framework for the automated testing of performance including performance regression analysis, of software under git management, such as the proxyapps developed as part of the ExCALIBUR NEPTUNE project. The requirements are (1) to allow easy tracking of performance improvements during the development of the proxyapps, (2) to identify if and when any code changes negatively affect the performance of such applications across a range of hardware and compiler combinations. The design for the framework, called 'RoundTable' is based on GitHub runners and uses TOML. RoundTable will allow acceptance tests and performance benchmarks to be run automatically using GitHub Actions on self-hosted runners at a range of local and HPC facilities such as Bede or Archer2. The results from these runs will then be collated and displayed in a web dashboard along with results from other applications, sites and code versions.

3.1.6 Report 2047356-TN-11 [11]

This report documents preliminary work to develop a $1d1v$ model, ie. a 1-D plasma model with 1-D velocity space effects, to the specifications of System 2-4 in ref [2]. The $1d1v$ target model consists of the Vlasov-Ampere system in a doubly periodic domain, although the use of wall boundary conditions is important for the edge, periodicity is important for testing. It is viewed as a stepping-stone exercise from which valuable lessons should be learnt, in the move towards more realistic, production codes.

A sequence of nine test cases is defined in Section 3, beginning with elementary tests of advection and advancing to sound wave models for coupled species in a self-consistent electric field. Simple Python implementations demonstrate key features of these test problems. Section 4 compares different representations of phase-space (ie. here the velocity coordinate) using finite difference (FD), finite volume (FV) and finite element (FE) methods, supporting the choice of the spectral/hp element for project NEPTUNE. Section 4.3 provides advice on the use of Python to prototype software, and Section 4.4 briefly discusses schemes used in widely available fusion software ('community codes') such as GS2 and STELLA, mentioning the latter's use of a non-interpolating semi-Lagrangian scheme. Section 5 discusses time integration schemes, indicating how the wide range of schemes implemented in the PETSc package might be made available to NEPTUNE, as well as the other popular explicit SSP-RK schemes and implicit stiff CVODE code from the SUNDIALS package. Adaptive and/or local timestepping is briefly discussed, as is the need for preconditioning matrices arising in implicit time advance schemes.

3.1.7 Report 2047356-TN-12 [12]

This report focuses on the performance of the elliptic solver within NEKTAR++ and its scaling in the limit of large processor counts. Building on the report [27], which introduces a number of sample test cases for the elliptic solver, the work tests both the correctness and the functionality of the elliptic solver. This solver has been used within the anisotropic diffusion solver developed under project NEPTUNE, proxyapp located in the NEPTUNE GitHub repository. The report discusses the formulation of an elliptic solver within a high-order finite element setting, the turning of this formulation into performant software, and the preconditioning strategies that may be employed to solve the resulting systems.

The report opens with an executive summary indicating its scope : the performance and scaling of the NEKTAR++ elliptic solver, with reference to instances of elliptic solves in existing proxyapps (viz. NEKTAR-DRIFTWAVE and NEKTAR-DIFFUSION) and the main NEKTAR++ code eg. the incompressible Navier-Stokes solver. Section 2 indicates that the report is limited to simple slab-type geometries as treated in the existing NEKTAR++-based proxyapps and related CFD problems. Insights into specific issues with elliptic spectral/hp are indicated, eg. matrix-free operations for discretized operators and the lack of effectiveness of multigrid preconditioners at higher-order. Section 3 presents the formulation of the matrix equations in the continuous Galerkin approach. Section 4 presents specific numerical approaches for iterative solvers, where the main work comprises repeated computations of the action of the matrix on the solution vector. Specifically, methods for addressing the $\mathcal{O}(p)^3$ scaling of the local element matrix size (in three dimensions) are presented, eg. static condensation, which decouples element boundary and interior degrees of freedom and is appropriate for two of the major basis sets in NEKTAR++, viz. the modified basis of Karniadakis and Sherwin, and the Lagrange interpolant basis. Subsection 4.4 covers the matrix-free operator implementation likely to be critical for Exascale exploitation, where the key idea is to remove element-specific information from matrices apart from geometric scale factors, and such that the action of the resulting matrices is computed by local matrix-free operations. The final Subsection 4.5 gives a brief overview of the global linear system solvers available in NEKTAR++. Section 5 concerns preconditioners, with a brief overview of the basic types, plus a longer description of the "Low energy" preconditioner of Sherwin and Casarin. Section 6 includes a brief performance study of the `IterativeStaticCond` solver for a massively-parallel problem where it is shown that (1) the implementation of the Laplace solve with a Jacobi preconditioner scales fairly well up to $131\,000$ cores on the Argonne Mira supercomputer (results published previously [28]), and (2) a more realistic incompressible Navier-Stokes problem involving an elliptic solve for pressure, with a low-energy block preconditioner, scales strongly to $50\,000$ cores on ARCHER2. Lastly, Section 7 outlines ongoing development work on the geometrically informed algebraic multigrid approach (GIAMG) which should be a very effective preconditioning technique as it has phases wherein order p and mesh-spacing h are separately varied.

The software developed is to be found as follows:

- The solution of electrostatic potential for the Hasegawa-Wakatani proxyapp: `https://github.`

com/ExCALIBUR-NEPTUNE/nektar-driftwave

- The solution of the Laplacian term as part of the anisotropic heat proxyapp: https://github.com/ExCALIBUR-NEPTUNE/nektar-diffusion
- The main NEKTAR++ code base: https://gitlab.nektar.info/nektar/nektar

3.1.8 Report 2047356-TN-13 [29]

This report briefly outlines potential routes for implementation of the system 2-6 equations in [2] within the NEKTAR++ framework, focussing on the steps required to implement such a system and highlighting where improvements should be made within NEKTAR++ to enable future implementation. It is suggested that testing should make use of the Method of Manufactured Solutions (MMS), and the value is noted of the capability of NEKTAR++ to combine classical and discontinuous Galerkin schemes in one solver.

3.1.9 Report 2047356-TN-14 [30]

This report gives a brief overview of the Domain Specific Languages (DSLs) provided by BOUT++ [31] and SD1D [21], which are both at a high-level level intended for user specification of a physical model including parameters. The report treats the point that a NEPTUNE DSL will have to be suitable for describing multiple species and their interactions in a manner that is suitable for a large number of such species and interactions. Specifically this report covers:

1. Approaches to code design in BOUT++ that allow for single-species plasma simulations to be extended to multiple species.
2. How these multi-species models might best be expressed in a high-level DSL.
3. An overview of the DSLs that exist already in the BOUT++ and NEKTAR++ frameworks and how a common DSL could be developed from the Unified Form Language (UFL) to drive both BOUT++ and NEKTAR++ solvers.

3.1.10 Report 2047356-TN-15 [15]

This report considers performance considerations relevant to the use of non-intrusive uncertainty quantification (UQ) through tools such as the EASYVVUQ package. It is demonstrated that for variations in the boundary conditions applied to the physical model, depending on their precise nature, convergence of the measures of sensitivity (first order Sobol indices) needs between 2nd and 5th order modes to be used in the Polynomial Chaos Expansion (PCE).

Section 3 contains the discussion of non-intrusive UQ, notably describing the benefits of non-intrusive UQ, when it comes to minimising the number of simulations and making effective use of workflow management tools, both of which are often important routes to improving the time to solution and freeing up researcher time. The disadvantage of adaptive (non-intrusive) UQ of making parallelisation harder has to be traded against the likelihood that the required number of simulations will be smaller. Section 4 describes the model system, which one of the most complicated described in [8], although omitting treatment of electric field, which anyway cannot apparently be treated self-consistently using SD1D [21]. The uncertainties allowed in the boundary conditions treated are variation of: (1) upstream sources, (2) neutral recycling fraction for three distinct boundary conditions on the plasma, namely (A) free at constant gradient, (B) zero gradient and (C) constant flux. The last case (C) gives rise to the apparent need for high order PCE for accurate Sobol index calculation. Although all three cases suffer from the problem of unphysical, negative values of density, there is the suggestion that this could be confounding the results in particular in case (C) and thus that the logarithm of density should be analysed in future UQ work.

3.1.11 Report 2047356-TN-16 [3]

This report discusses ExCALIBUR-NEPTUNE community activities in the October 2021 to October 2022 period and the state of the community environment at the date of this report. Section 2 describes the numerous workshops and events hosted by Neptune project partners during this period, and their value in linking members from the different groups. Section 3 describes how the digital tools set up at the start of the project to support community interactions are still in use, and suggests how to modify the digital environment both so as to increase engagement and to assist with bringing new members up to speed. It is pointed out that this is likely to become increasingly important as the project moves forwards and further stakeholders emerge. Most of the suggestions require dedicated software admin support. The limitations of ReadTheDocs to relatively small projects reported, helps makes the case for the production in fact taking place, of a developers' website that can support the considerable complexity project NEPTUNE.

3.1.12 Report 2047356-TN-17 [16]

This brief report outlines the initial efforts on implementing the equation set System 2-6 [2] within the NEKTAR++ framework. The set has been changed from the model adopted by the European Boundary Code project, equations for which were described in Release 1.00 of the Equations document [22], to the similar equations capable of solution by the Hermes-3 software to be found in Release 1.26. The report describes implementation and solution of a simplified set of equations possessing properties shared by classical computational fluid dynamical models. It indicates a future development pathway for system 2-6, employing the sequence of models of increasing complexity described on the Hermes-3 website [25].

3.2 Reports received under Grant T/AW086/21, PO 2060049

The STFC grantee produced the following reports

- 2060049-TN-02-5 - Techniques and software relevant to the coupling of continuum (fluid) and particle models of plasma for NEPTUNE [32]

- 2060049-TN-03 - Time Stepping Techniques and Preconditioning for Hyperbolic and Anisotropic Elliptic Problems: Numerical Results [33]

3.2.1 Report 2060049-TN-02 [32]

Report 2060049–TN–02 reviews a number of possible techniques and frameworks for coupling fluid and particle codes in project NEPTUNE. It goes on to present results from a proxyapp which demonstrates the use of the CWIPI coupling library.

In Section 2, three forms of coupling are discussed for fluid-particle systems: Spatial Hybridization (SpH), the Micro-Macro Hybrid (MMH) approach and the Kinetic Diffusion Monte Carlo Method (KDMCM). SpH and MMH are both identified as reasonable strategies to use in NEPTUNE, but KDMCM is disfavoured on the grounds that it has only been demonstrated on a few idealised physics problems and is sufficiently novel in implementation that its use would require significant changes to the way NEPTUNE code is being developed.

Section 3 describes six different open source coupling libaries: MUI, OPENPALM, PRECICE, PLE, CWIPI and MUSCLE3. In each case, a high-level description of the implementation is provided and a number of factors are discussed that could affect the framework's suitability for use in NEPTUNE. These include the form of data which may be exchanged, the types of API that are provided, example use cases in which the coupling has already been applied and the quality of the documentation.

Of the different frameworks, MUI and CWIPI are deemed most suitable, in that they are minimally invasive, provide good quality APIs and could be used to couple fluid and particle codes with no obvious negative impact on scalability. CWIPI is slightly to be preferred as it has already been integrated into NEKTAR++ to some degree. The appendix is used to present early results from the coupling of a "particle-style code" to a NEKTAR++ fluid solver using the CWIPI library.

The former is a simple executable that can send and receive data across the coupling interface, serving as a placeholder for an actual particle solver (eg. NESO-PARTICLES). Two alternative NEKTAR++ executables are discussed, one configured to solve an anisotropic diffusion problem, the other a convection problem.

In the first two test cases, different diffusion coefficients, first spatially uniform, then spatially varying, are sent to the NEKTAR++ solver and the resulting velocity fields are plotted in order to check the results.

The third test case demonstrates both time-variable communication and two-way coupling. The Navier-Stokes solver sends the temperature field (at each timestep) to the particle code, which sends back a corresponding source term that is added to the fluid equations. The error associated with translating between particle and continuum data can be examined by comparing this to a non-coupled version of the problem, wherein the fluid solver computes the source term directly. A number of different mesh resolutions were used for the particle code, between 1.25 and 32 times coarser than that used by the fluid solver. The authors find that the maximum error in the temperature field does not start to increase significantly until the coarseness ratio exceeds a factor of approx. 6, suggesting that some level of mismatch between the fluid and particle resolutions may be tolerated in exchange for reduced runtimes.

The authors conclude that the SpH coupling approach would be the most natural fit for the current NEPTUNEproject, which is reasonable from the stand-point of the software developer, although they fail to consider the efficiency of the resulting code. The noise associated with particles has to be tolerated when it is too expensive to model 6-D with a continuum, or the physical processes involved do not have a simple collective effect, but the first justification almost never applies and the second rarely applies, in 3-D. Thus NEPTUNE will continue to use high order elements to calculate 3-D fluid effects, and indeed software developed by the fusion community increasingly sees use of continuum models for $2d3v$ and $3d3v$ work.

3.2.2 Report 2060049-TN-03 [33]

Report 2060049-TN-03 seeks answers to open questions regarding numerical solvers applied to the dynamics of magnetised plasmas, specifically by solving advection-diffusion equations with a combination of anisotropic grids with highly anisotropic flows and diffusion tensors. The topics of preconditioners and time integrators are discussed in this context, with comments on error, scalability and performance.

The first section discusses the machine on which the tests are run, followed by a second section that describes how anisotropy may be incorporated through the equations themselves or via Kershaw grids, which are topologically equivalent to regular (ie. finite-difference type) meshes, but made up of greatly distorted quadrilaterals.

Section 3 concerns preconditioners and begins with an overview of algebraic multigrid (AMG), p-multigrid, hp-multigrid, and preconditioning of matrix-free linear operators. p-multigrid, the authors explain, involves solving a system with reduced polynomial order via, eg. AMG and then propagating that result as a starting point for the 'full-p' problem. One method that the authors focus on particularly is Low-Order Refined (LOR) preconditioning. The $p = 1$ version of a matrix-free operator is assembled, refined spatially in a manner that is sensitive to the high-p origins of the problem, solved via traditional low-order AMG methods, and finally projected back up to higher-p. The Overlapping Domain Decomposition Method (ODDM) is dicussed next, with reference to its useful convergence properties, but the authors note that the leading library for ODDM has not yet reached maturity. Sparse approximate inverse preconditioning (SPAI) is also considered. The section concludes with a comparison of the performance of the different preconditioners for various combinations of grid and physics anisotropy. LOR is found to perform best, with no obvious weaknesses.

Section 4 introduces a time-stationary advection-diffusion problem with tunable Peclet number. AMG-LOR gives the best results in terms of both error achieved and runtime.

The 5^{th} section compares backwards Euler, Crank-Nicolson and a variety of singly-diagonal implicit Runge-Kutta (SDIRK) schemes by applying them to an advection equation. For scales of interest, SDIRK33 is found to give the best performance. Next, a variety of SDIRK, IRK-Gauss and IRK-Lobatto methods are compared for low, medium and high-order spatial discretisations, with the number of DOFs per stage held fixed. It is found that the best results are obtained using high-order methods both temporally and spatially. While the number of iterations appears to be robust, the overall scaling is found to be suboptimal. The authors suggest that scaling could be improved by combining the fully implicit RK methods with LOR and AMG.

3.3 Reports received under Grant T/AW087/21, PO 2057699

The York grantee produced the following reports in association with Warwick

1. 2047358-TN-01-2 - Approaches to Performance Portable Applications for Fusion [34]
2. 2047358-TN-02-2 - Identification of Testbed Platforms and Applications [35]
3. 2047358-TN-03-2 - Evaluation of Approaches to Performance Portability [36]
4. 2057699-TN-01-5 - Hardware at the Exascale Revision 5.0 [37]
5. 2057699-TN-02-5 - Developing an Exascale-Ready Fusion Simulation Revision 5.0 [38]
6. 2057699-TN-03-3 - Progress on Development of an FEM-PIC Miniapp [39]
7. 2057699-TN-04-2 - Proposal for a Particle DSL [40]

The first three reports (produced under Grant T/NA086/20, PO 2047358) were superseded by material produced under the new grant and are not described below.

3.3.1 Report 2057699-TN-01-5 [37]

There have been many changes to the hardware makeup of supercomputers as they have reached ever higher levels of computing performance. Be it the move towards to first mainstream multicore CPUs in 2004, to the move from CPU only compute nodes to the majority of machines now having both a CPU and GPU. The UKRI intends to build the UK's first exascale supercomputer by 2025. This report details the hardware that Neptune is likely to encounter both pre and post exascale. This report is split into three parts. The first part outlines the hardware currently deployed or scheduled to be delivered in next five years according to each vendor's respective roadmap. This material includes details such as the core count range for a particular CPU range, supported memory technology used, and peak double precision performance. The second part outlines what machines are currently being built or proposed within Europe and the USA, and elsewhere. The final part details both homogeneous and heterogenous machines currently available with which the performance portability of codes developed as part of NEPTUNE could be evaluated.

3.3.2 Report 2057699-TN-02-5 [38]

As the Exascale era is entered, all machines, with the noticeable exception of Fukgaku, will be made of heterogeneous nodes containing both a multi-core CPU and a GPU. NEPTUNE software will be required to be performance portable, targeting these multiple all these CPU and GPU architectures, while maintaining high levels of performance across them. Several programming models exist to parallelise code, but as of yet there is no definitive frontrunner which shows itself as the best candidate for use at the Exascale. Therefore, the objective of this report is to investigate the effectiveness of different programming models used for parallelisation to achieve performance portable code. This was achieved by evaluating several fluid and particle proxyapps using a variety of performance benchmarks, across different hardware architectures and compilers.

The key findings of this investigation were that:

- The majority of the NEPTUNE codebase should be written in modern C++
- While vendor-specific parallelisation techniques can achieve higher performance than open standards, they suffer from the disadvantage of producing unportable code. The disadvantage of open standards is that they general take longer to target newer hardware and hardware features.
- NEPTUNE will likely require both low level and high level DSLs
- Data storage should be abstracted away from algorithms as much as possible
- Code such be modularised as much as possible to allow key kernels which are performance critical to be re-evaluated in isolation as new technologies emerge
- Where possible code should be reused including external libraries

3.3.3 Report 2057699-TN-03-3 [39]

A finite element particle in cell (FEM-PIC) miniapp which uses an unstructured grid was developed for evaluation when applied to problems the Neptune project is likely to encounter. The particle pusher kernel, the kernel for depositing charge and the kernel for calculating the fields were identified in a previous report as key components needed for performance evaluation. The performance was evaluated by the total time for which the functions relating to each particular kernel took to run. A test case looking at Deuterium ion flow for a pipe was chosen for this study. Ten runs were performed ranging from around $10-130$ particles/element, using both the Intel and GNU compiler on the Viking cluster. For both compilers was found that for anything above approximately 20 ions per element that the majority of the time was taken up by the particle pusher routines, which grows approximately linearly with the number of particles/element. The time take for injecting ions onto the grid and calculating ϕ from Poisson's equation was found to be approximately independent of the number of particles/element. The current status of this work can be found at .

3.3.4 Report 2057699-TN-04-2 [40]

Domain specific languages (DSLs) allows one to provide a bridge between domain scientists and application developers, allowing them to write high level abstractions of the problem they wish to solve, which can then be turned into low level parallelised code. Previous DSLs for use within plasma codes often are dedicated to describing only the fluid nature of the plasma. The development of a DSL that can also be used for particle methods was therefore a necessity moving forward with the NEPTUNE project. The development of this DSL is outlined in this report.

The new DSL was done by modifying the OP2 DSL to accommodate the particle in cell (PIC) method. Since no complete unstructured mesh 3-D electromagnetic FEM PIC code is available, three PIC codes, namely SIMPIC, CABANAPIC and FEMPIC were selected for conversion to make use of this DSL, to demonstrate its functionality. The rewritten SIMPIC and CABANAPIC codes were written in C++ for a serial implementation, and the calculated particle and grid point data were verified to be unchanged. FEMPIC was converted to work both sequential and with OpenMP. The current status of this work can be found at .

Acknowledgement

The support of the UK Meteorological Office and Strategic Priorities Fund is acknowledged.

References

[1] J. Parker and W. Arter. Management of external research. Numerical Analysis Procurement. Technical Report CD/EXCALIBUR-FMS/0068-M7c.1, UKAEA, 6 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/ukaea_reports/CD-EXCALIBUR-FMS0068-M7c.1.pdf.

[2] W. Arter et al. Equations for EXCALIBUR/NEPTUNE Proxyapps. Technical Report CD/EXCALIBUR-FMS/0021-1.26-M1.2.1, UKAEA, 1 2023. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/ukaea_reports/CD-EXCALIBUR-FMS0021-1.26-M1.2.1.pdf.

[3] B. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 0.3 Community and environment. Technical Report 2047356-TN-16-1, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-16-1.pdf.

[4] ExCALIBUR Project NEPTUNE website. https://excalibur-neptune.github.io/Developers-Website/main/main.html, 2023. Accessed: January 2023.

[5] Source material for ExCALIBUR Project NEPTUNE website. https://github.com/ExCALIBUR-NEPTUNE/Developers-Website, 2023. Accessed: January 2023.

[6] B. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 1.4 Preconditioning 2D elliptic solvers. Technical Report 2047356-TN-06-1, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-06-1.pdf.

[7] B. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 2.2 BOUT++ 1D fluid solver with realistic boundary conditions : implementation. Technical Report 2047356-TN-07-1, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-07-1.pdf.

[8] B. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 2.4 Non-intrusive UQ with BOUT++ 1D fluid solver. Technical Report 2047356-TN-08-1, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-08-1.pdf.

[9] B. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 4.1 Test cases for 1D multispecies plasma model. Technical Report 2047356-TN-09-1, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-09-1.pdf.

[10] E. Higgins and D. Dickinson. Task 0.2 Development of a performance testing and aggregation tool for NEPTUNE proxy-apps. Technical Report 2047356-TN-10-1, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-10-1.pdf.

[11] B. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 3 Considerations for 1D1V models Report covering tasks 3.1, 3.2 and 3.3. Technical Report 2047356-TN-11-1, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-11-1.pdf.

[12] D. Moxey, B. Dudson, P.A. Hill, E. Higgins, D. Dickinson, and S. Wright. (D1.3): Elliptic solvers within Nektar++. Technical Report 2047356-TN-12-2, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-12-2.pdf.

[13] D. Moxey, B.D. Dudson, P.A. Hill, E. Higgins, D. Dickinson, and S. Wright. Spatially 2-D plasma model incorporating velocity space effeects. Technical Report 2047356-TN-13, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-13-2.pdf.

[14] B.D. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. DSL and code design pattern for NEPTUNE. Technical Report 2047356-TN-14-1, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-14-1.pdf.

[15] B. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 2.5 Performance considerations for non-intrusive uncertainty quantification and the influence of the boundary. Technical Report 2047356-TN-15-2, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-15-2.pdf.

[16] D. Moxey, B.D. Dudson, P.A. Hill, E. Higgins, D. Dickinson, and S. Wright. Progress on implementation of system 2-6 equations. Technical Report 2047356-TN-17, UKAEA Project Neptune, 2023. to be posted at https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-17.pdf.

[17] S. Thorne. Priority Equations and Test Cases. Technical Report 2047353-TN-01, UKAEA Project Neptune, 2021. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047353/TN-01.pdf.

[18] V. Alexandrov, A. Lebedev, E. Sahin, and S. Thorne. Linear systems of equations and preconditioners relating to the NEPTUNE Programme: A brief overview. Technical Report 2047353-TN-02, UKAEA Project Neptune, 2021. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047353/TN-02.pdf.

[19] M. Abalenkovs, V. Alexandrov, A. Lebedev, E. Sahin, and S. Thorne. Implicit-factorization preconditioners for NEPTUNE Programme. Technical Report 2047353-TN-03, UKAEA Project Neptune, 2021. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047353/TN-03.pdf.

[20] M. Abalenkovs, V. Alexandrov, A. Lebedev, E. Sahin, and S. Thorne. Implicit-factorization preconditioners for non-symmetric problems. Technical Report 2047353-TN-04, UKAEA Project Neptune, 2021. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047353/TN-04.pdf.

[21] One-dimensional plasma-neutral simulation for SOL and divertor studies. https://github.com/boutproject/SD1D, 2021. Accessed: August 2021.

[22] W. Arter. Equations for EXCALIBUR/NEPTUNE Proxyapps. Technical Report CD/EXCALIBUR-FMS/0021-1.00-M1.2.1, UKAEA, 03 2020. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/ukaea_reports/CD-EXCALIBUR-FMS0021-1.00-M1.2.1.pdf.

[23] B.D. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 2.1 1D fluid model tests. Technical Report 2047356-TN-04-2, UKAEA Project Neptune, 2021. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-04.pdf.

[24] P.A. Hill, J. T. Parker, D. Dickinson, and B.D. Dudson. BOUT++ VECMA hackathon repository. https://github.com/boutproject/VECMA-hackathon, 2021. Accessed: June 2021.

[25] Hermes-3 2D axisymmetric tokamak. https://hermes3.readthedocs.io/en/latest/examples.html#d-axisymmetric-tokamak, 2021. Accessed: June 2021.

[26] Hermes plasma edge simulation model: Hermes-3, a hot ion multifluid drift-reduced model. https://github.com/bendudson/hermes-3, 2021. Accessed: June 2021.

[27] B.D. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Task 1.1 Elliptic solver tests. Technical Report 2047356-TN-02-2, UKAEA Project Neptune, 2021. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-02.pdf.

[28] D. Moxey. Nektar++ strong scaling performance on Mira. https://www.nektar.info/nektar-strong-scaling-performance-on-mira/, 2017.

[29] D. Moxey, B.D. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. Spatially 2-D plasma model incorporating velocity space effects. Technical Report 2047356-TN-13, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-13.pdf.

[30] B.D. Dudson, P.A. Hill, E. Higgins, D. Dickinson, S. Wright, and D. Moxey. DSL and code design pattern for NEPTUNE. Technical Report 2047356-TN-14, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047356/TN-14.pdf.

[31] B.D. Dudson. BOUT++ website. https://boutproject.github.io/, 2020. Accessed: June 2020.

[32] H. Al Daas, J. Williams, N. Bootland, T. Rees, P. Rubin, S. Thorne, and A. Sunderland. Techniques and software relevant to the coupling of continuum (fluid) and particle models of plasma for NEPTUNE. Technical Report 2060049-TN-02-5, UKAEA Project Neptune, 2023. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2060049/TN-02-5.pdf.

[33] H. Al Daas, N. Bootland, T. Rees, and S. Thorne. Time Stepping Techniques and Preconditioning for Hyperbolic and Anisotropic Elliptic Problems: Numerical Results. Technical Report 2060049-TN-03, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2060049/TN-03.pdf.

[34] S. Wright, B. Dudson, P.A. Hill, D. Dickinson, and G. Mudalige. Approaches to Performance Portable Applications for Fusion. Technical Report 2047358-TN-01-2, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047358/TN-01-2.pdf.

[35] S. Wright, B. Dudson, P.A. Hill, D. Dickinson, and G. Mudalige. Identification of Testbed Platforms and Applications. Technical Report 2047358-TN-02-2, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047358/TN-02-2.pdf.

[36] S. Wright, B. Dudson, P.A. Hill, D. Dickinson, and G. Mudalige. Evaluation of Approaches to Performance Portability. Technical Report 2047358-TN-03-2, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2047358/TN-03-2.pdf.

[37] S. Wright, E. Higgins, B.D. Dudson, P.A. Hill, D. Dickinson, G. Mudalige, B.F. McMillan, and T. Goffrey. Hardware at the Exascale Revision 5.0. Technical Report 2057699-TN-01-5, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2057699/TN-01-5.pdf.

[38] S. Wright, E. Higgins, B.D. Dudson, P.A. Hill, D. Dickinson, G. Mudalige, B.F. McMillan, and T. Goffrey. Developing an Exascale-Ready Fusion Simulation Revision 5.0. Technical Report 2057699-TN-02-5, UKAEA Project Neptune, 2022. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2057699/TN-02-5.pdf.

[39] S. Wright, E. Higgins, G. Mudalige, B.F. McMillan, and T. Goffrey. Progress on Development of an FEM-PIC Miniapp. Technical Report 2057699-TN-03-3, UKAEA Project Neptune, 2023. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2057699/TN-03-3.pdf.

[40] S. Wright, E. Higgins, G. Mudalige, Z. Lantra, B.F. McMillan, and T. Goffrey. Proposal for a Particle DSL. Technical Report 2057699-TN-04-2, UKAEA Project Neptune, 2023. https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/2057699/TN-04-2.pdf.

A Annex: D8c Report

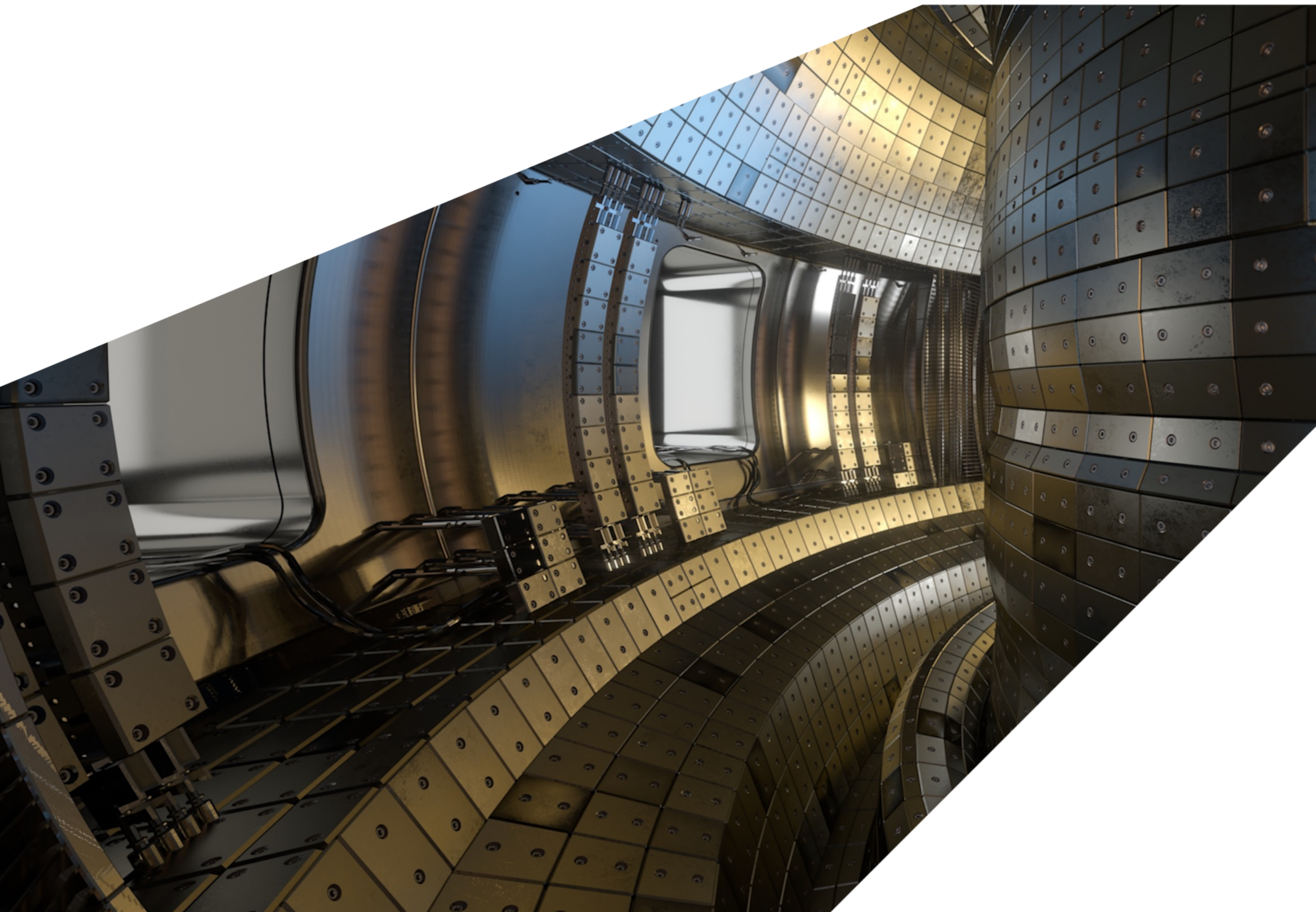
UK Atomic Energy Authority

ExCALIBUR

Complementary actions. Atomic data usage 1.

M8c.1 Version 1.00

Abstract
The report describes work for ExCALIBUR project NEPTUNE at Milestone M8c.1. This report describes NEPTUNE use of analytic formulae and atomic physics databases for describing the rate of atomic collision processes. Scripts have been developed that make use of the Atomic Data and Analysis Structure (ADAS) database to predict the electron impact ionisation rate, and the charge exchange cross section. The analytic formula being used to predict rate of ionisation from electron impacts on hydrogen has found good agreement with ADAS.



UKAEA REFERENCE AND APPROVAL SHEET

	Client Reference:	
	UKAEA Reference:	CD/EXCALIBUR-FMS/0076
	Issue:	1.00
	Date:	March 23, 2023
Project Name: ExCALIBUR Fusion Modelling System		

	Name and Department	Signature	Date
Prepared By:	Matthew Barton BD	N/A	March 23, 2023
Reviewed By:	Wayne Arter Project Technical Lead		March 23, 2023

1 Introduction

Collisions in plasmas between electrons, atoms and molecules can happen in a variety of ways. Such processes include but are not limited to electron impact ionisation, recombination, electron impact excitation, charge exchange, etc. Accurately knowing the effect these collisions are having on the abundance of various ions and electron populations in the system is necessary to make predictions about the radiative properties of atoms and molecules. The rate at which these different processes occur can be determined in a variety of ways, such as atomic databases, analytic formula or collisional radiative models. The former two ways have been investigated for this Milestone M8c.1, and then compared against each other.

With regard to the use of atomic databases used for this Milestone, it was decided that the Atomic Data and Analysis Structure (ADAS) database would be used. ADAS is made up of two components, atomic data, which can be searched online using OPEN-ADAS (see Ref. [1]), and a Fortran codebase to process the aforesaid data to extract the atomic processes rates of interest. Rather than using the Fortran code directly, it was decided that use would be made of the Python interface which is available. Scripts were made which could utilise ADAS to output the electron impact ionisation rate, and charge exchange cross section for reactions for which the data is available. These scripts can be found within the Neso-Reactions GitHub repo.

For the initial introduction of the effect of electron collisions in a plasma, a formula was found which can predict the ionisation rate for electron impact ionisations. The accuracy of this formula was tested by comparing the result obtained by the ADAS database for the following reaction

$$H + e \Rightarrow H^{+} + e + e \tag{1}$$

to that of the formula. Once the accuracy of the formula had been tested it was integrated into the SOL2D-neutral proxyapp. The results of this integration can be found in the report on Milestone 4c.2 [2].

2 Task Work

The purpose of this section is to give a more detailed breakdown of the work undertaken for Milestone M8c.1.

2.1 ADAS

One way to predict atomic transitions is to make use of atomic databases . There are numerous atomic databases currently employed with plasma codes, such as ALADDIN [3], AMJUEL [4] and ADAS [5]. For initial investigations ADAS was chosen. The ADAS database considered consists of two parts. The first part is the atomic data which has been collected from numerous experiments, and the second part is a Fortran codebase to process said atomic data. The Python interface to the underlying Fortran code was utilised for investigating its use by Neso. Scripts were created which could allow the calculation of charge exchange cross sections and electron impact ionisation rates. Charge exchange is the following process

$$A_p^{(Z_i+1)+} + H \Rightarrow A_p^{Z_i+} + H^+ \tag{2}$$

and electron impact ionisation is

$$A_p^{Z_i+} + e \Rightarrow A^{Z_i+} + e + e. \tag{3}$$

In the energy range of concern for Neso these are the 2 most prominent reactions which can occur in a plasma for atomic Hydrogen (See Fig 37 of [6]). The Python scripts can be found in Neso-Reactions Github repo.

2.2 Analytical Formula

The starting point to defining a ionisation rate is the cross section. A cross section can be thought of as an approximate area around a target that an electron has to be in for an interaction to take place. The following formula taken from Ref. [7] is used for the cross section (originally from Ref. [8])

$$\begin{aligned} \sigma &= \sum_{i=1}^{N} a_i q_i \frac{ln\left(\frac{E}{E_i}\right)}{EE_i}[1 - b_i e^{-c_i\left(\frac{E}{E_i}-1\right)}] \text{ for} \quad & E \geq E_i \\ &= 0 & E < E_i \end{aligned} \tag{4}$$

where E is the energy of the impacting electron, E_i is the binding energy of electrons in the i-th subshell. q_i is the number of electrons in the i-th subshell. a_i,b_i and c_i are constants to be determined from theory or experiment. N=1 for hydrogen and helium like ions. The reference values taken from Ref. [7] for hydrogen are as follows (i can only take the value 1)

Parameter	Value
a_1	$4\text{x}10^{-14} cm^2 (eV)^2$
b_1	0.6
c_1	0.56
E_1	13.6 eV
q_1	1

This cross section can be folded in with a Maxwellian electron distribution at temperature T to give the following rate (see Ref. [7])

$$S = -6.7\text{x}10^7 \sum_{i=1}^{N} \frac{a_i q_i}{T^{\frac{3}{2}}} \left\{ \frac{T}{E_i} Ei\Big(-\frac{E_i}{T} \Big) - \frac{b_i e^{c_i}}{\frac{E_i}{T} + c_i} Ei\Big(-\frac{E_i}{T} - c_i \Big) \right\} \quad (5)$$

where T is in eV and S is in units of cm^{3}s^{-1}. In the above formulas multiple ionization, lowering of ionization potential, or collision limit are not taken into account. Ei(x) is the exponential integral defined by

$$-\int_{-x}^{\infty} \frac{e^{-t}}{t} dt. \quad (6)$$

Fig. 2 shows a comparison between the rate obtained from this formula against the rate obtained by ADAS for

$$H + e \Rightarrow H^+ + e + e. \quad (7)$$

As can be seen from Fig. 2 the rates are in approximately agreement over the energy range chosen. When integrating this rate into Neso the total rate is required. This is found using the following

$$\frac{dN_p}{dt} = SN_pN_e \quad (8)$$

where N_p is the population of state p, and N_e is the electron density [9, 10]. This rate was integrated into the SOL2D-neutral proxyapp, the results of which can be found in the report documenting Milestone 4c.2.

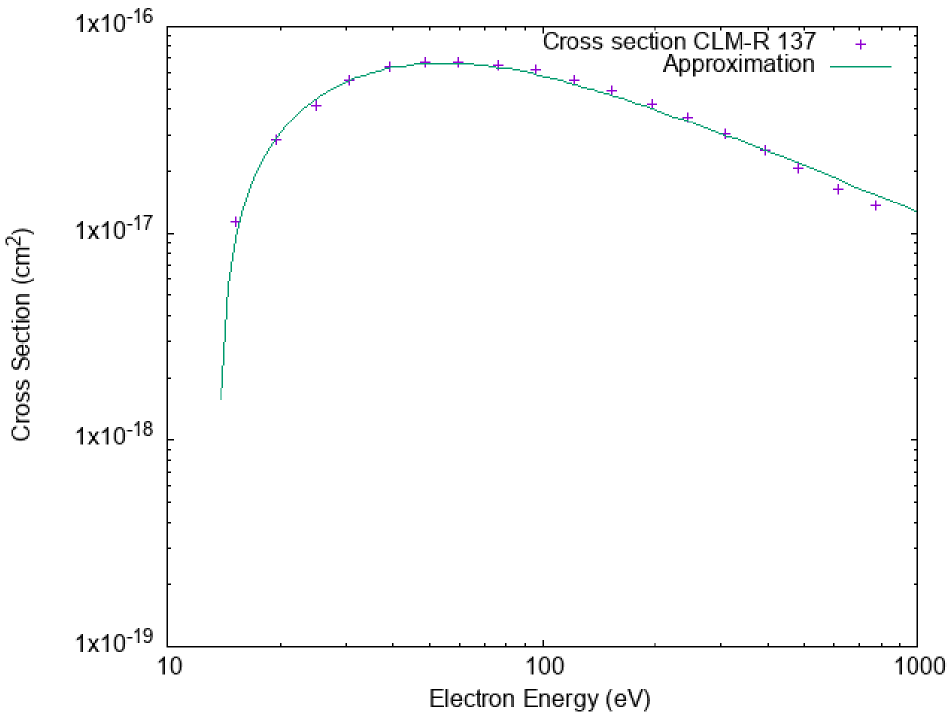


Figure 1: The above plot shows the cross section obtained from the analytic formula (solid line) 4 against those obtained Ref. [6] (crosses).

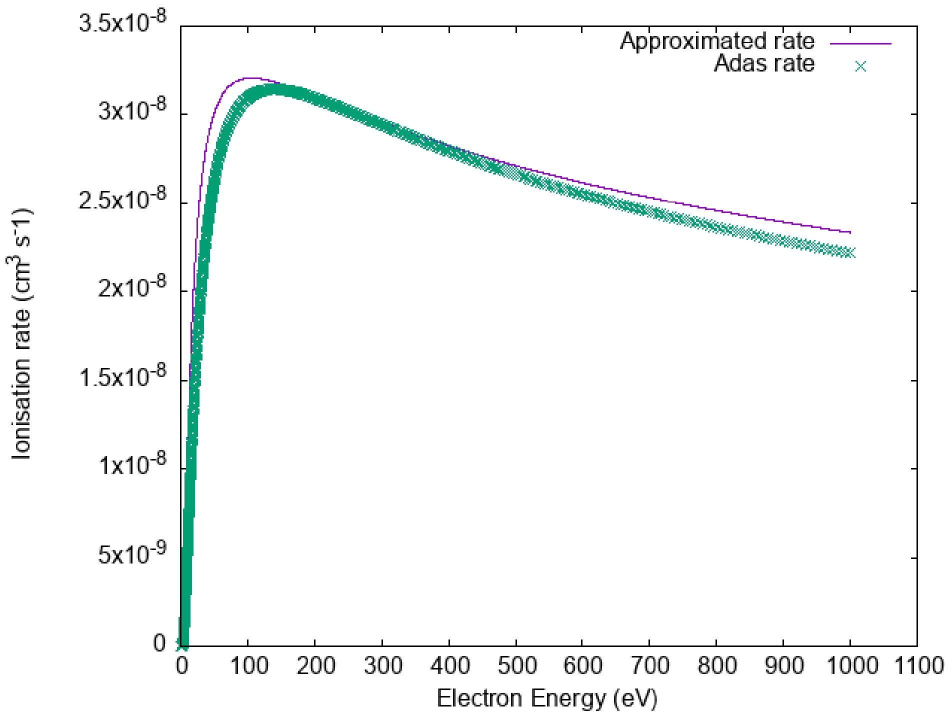


Figure 2: The above plot shows the ionisation rate obtained from Eq. 5 (solid line) against those obtained from ADAS (crosses).

3 Summary

Investigations have been conducted to compare the prediction of an analytical formula for the rate of electron impact ionisation, against atomic databases for use within NESO. Moderate agreement was found between the two methods in terms of the predicted rate for electron impact ionisation. Consequently the analytical formula for the rate has been included within the SOL2D-neutral prox-yapp [2].

Acknowledgement

The support of the UK Meteorological Office and Strategic Priorities Fund is acknowledged.

References

[1] OPEN-ADAS Atomic Data and Analysis Structure. `https://open.adas.ac.uk`.

[2] J. Cook, E. Threlfall, , O. Parry, W. Saunders, and W. Arter. Three-Dimensional integrated particle and continuum model. Technical Report CD/EXCALIBUR-FMS/0072-M4c.2, UKAEA, 3 2023. `https://github.com/ExCALIBUR-NEPTUNE/Documents/blob/main/reports/CD-EXCALIBUR-FMS-0072-M4c.2.pdf`.

[3] ALADDIN database. `https://www-amdis.iaea.org/ALADDIN/`.

[4] AMJUEL database. `https://www.eirene.de/old_eirene/html/amjuel.html`.

[5] Atomic Data and Analysis Structure (ADAS). `https://www.adas.ac.uk/about.php`.

[6] Atomic Collision Processes in Plasma Physics Experiments CLM-R137. `https://scientific-publications.ukaea.uk/wp-content/uploads/CLM-R137.pdf`.

[7] Wolfgang Lotz. Electron-impact ionization cross sections and ionization rate coefficients for atoms and ions from hydrogen to calcium. *Z. Phys., 216: 241-7(1968).*, 1 1968.

[8] Wolfgang Lotz. An empirical formula for the electron-impact ionization cross-section. *Zeitschrift für Physik*, 206(2):205–211, 1967.

[9] G. J. Tallents. *An Introduction to Special Relativity for Radiation and Plasma Physics*. Cambridge University Press, 2022.

[10] G. J. Tallents. *An Introduction to the Atomic and Radiation Physics of Plasmas*. Cambridge University Press, 2018.