PAPER • **OPEN ACCESS**

Validation of the static forward Grad–Shafranov equilibrium solvers in FreeGSNKE and Fiesta using EFIT++ reconstructions from MAST-U

To cite this article: K Pentland *et al* 2025 *Phys. Scr.* **100** 025608

Physica Scripta

PAPER

Validation of the static forward Grad–Shafranov equilibrium solvers in FreeGSNKE and Fiesta using EFIT++ reconstructions from MAST-U

K Pentland[1], N C Amorisco[1], O El-Zobaidi[1], S Etches[1], A Agnello[2], G K Holt[2], A Ross[2], C Vincent[1], J Buchanan[1], S J P Pamela[1], G McArdle[1], L Kogan[1] and G Cunningham[1]

[1] United Kingdom Atomic Energy Authority, Culham Campus, Abingdon, Oxfordshire, OX14 3DB, United Kingdom
[2] STFC Hartree Centre, Sci-Tech Daresbury, Keckwick Lane, Daresbury, Warrington, WA4 4AD, United Kingdom

E-mail: kamran.pentland@ukaea.uk

Keywords: MHD equilibria, Grad-Shafranov, FreeGSNKE, Fiesta, EFIT++, MAST-U

OPEN ACCESS

RECEIVED
20 November 2024

REVISED
10 December 2024

ACCEPTED FOR PUBLICATION
19 December 2024

PUBLISHED
17 January 2025

Abstract

A key aspect in the modelling of magnetohydrodynamic (MHD) equilibria in tokamak devices is having access to fast, accurate, and stable numerical simulation methods. There is an increasing demand for reliable methods that can be used to develop traditional or machine learning-based shape control feedback systems, optimise scenario designs, and integrate with other plasma edge or transport modelling codes. To handle such applications, these codes need to be flexible and, more importantly, they need to have been validated against both analytically known and real-world tokamak equilibria to ensure they are consistent and credible. In this paper, we are interested in solving the static forward Grad–Shafranov (GS) problem for free–boundary MHD equilibria. Our focus is on the validation of the static forward solver in the Python-based equilibrium code *FreeGSNKE* by solving equilibria from magnetics-only *EFIT*++ reconstructions of MAST-U shots. In addition, we also validate FreeGSNKE against equilibria simulated using the well-established MATLAB-based equilibrium code *Fiesta*. To do this, we develop a computational pipeline that allows one to load the same (a)symmetric MAST-U machine description into each solver, specify the required inputs (active/passive conductor currents, plasma profiles and coefficients, etc.) from EFIT++, and solve the GS equation for all available time slices across a shot. For a number of different MAST-U shots, we demonstrate that both FreeGSNKE and Fiesta can successfully reproduce various poloidal flux quantities and shape targets (e.g. midplane radii, magnetic axes, separatrices, X-points, and strikepoints) in agreement with EFIT++ calculations to a very high degree of accuracy. We also provide public access to the code/data required to load the MAST-U machine description in FreeGSNKE/Fiesta and reproduce the equilibria in the shots shown.

1. Introduction

1.1. Motivation and aims

Developing fast and accurate numerical methods for simulating the ideal magnetohydrodynamic (MHD) equilibrium of a magnetically-confined plasma is a crucial element in the design and operation of existing and future tokamak devices. These solvers are used extensively to analyse different plasma scenarios, shapes, and stability, in addition to playing a critical role in the operation and optimisation of control and real-time feedback systems.

Many different equilibrium solvers have evolved over time and vary widely in terms of their design, purpose, ease-of-use, computational speed, and availability. For instance, they have been implemented in different programming languages and have harnessed various spatial discretisation schemes, techniques for solving nonlinear systems, and approaches for tackling optimisation problems. They can, however, be broadly classified into two primary categories: *static* and *dynamic* (sometimes called *evolutive*) solvers.

Our focus is on static solvers, which are time-independent and are designed to solve one of the following:

- '**Forward problem**': Solve for the plasma equilibrium using user-defined poloidal field coil currents, passive structure currents, and plasma current density profiles.
- '**Inverse problem**': Estimate poloidal field coil currents using user-defined constraints (e.g. isoflux and X-point locations) and plasma current density profiles for a desired plasma equilibrium shape.
- '**Reconstruction problem**': Estimate (or fit) poloidal field coil currents and plasma current density profiles that yield a plasma equilibrium that 'best' matches (noisy) measurement data from diagnostics around a tokamak (e.g. plasma/coil currents, magnetic readings, and density profiles).

Dynamic 'forward' and 'inverse' solvers which tackle time-dependent equilibrium problems are also available. These solvers couple the static plasma equilibrium with time-evolving external conductor currents and voltages, however, they will not feature here.

In this paper, we focus on the *free–boundary static forward MHD equilibrium problem*, which involves solving the Grad–Shafranov (GS) equation for a toroidally symmetric, plasma equilibrium. As mentioned above, this requires user-defined poloidal field coil currents, passive structures currents (if available), and plasma current density profiles—see section 3 for further details. A vast array of numerical codes exist for solving this problem, however, our focus will be on two in particular: *FreeGSNKE* and *Fiesta* (more details to follow in section 2). The aim of this work is to:

(i) validate that the static forward solver in FreeGSNKE can reproduce the equilibria obtained by a magnetics-only *EFIT*++ reconstruction on the MAST-U tokamak (in addition to the equilibria produced by Fiesta).

(ii) compare poloidal flux quantities, shape control measures (e.g. midplane radii, magnetic axes, separatrix positions, X-points, and strikepoints), and magnetics measurements from the solvers for a number of physically different MAST-U shots, using EFIT++ as the reference solution.

To enable an accurate and valid comparison of the results, we need to ensure that the static forward solvers in both FreeGSNKE and Fiesta are set up using the same set of *input* quantities as *output* by EFIT++ (which itself solves the reconstruction mentioned before). Firstly, this will require a consistent description of the MAST-U machine that includes the active coils, passive structures, wall/limiter, and magnetic diagnostics. We require details of their positions, orientations, windings, and polarity. Secondly, for each equilibrium, we need to appropriately assign values for both the active/passive conductor currents and plasma profiles parameters as calculated by EFIT++. In section 4, we provide more details on these input quantities and highlight differences between each code implementation.

We should stress that while the reference EFIT++ equilibria are obtained as equilibrium reconstructions, and are therefore the result of a fitting procedure from experimental measurements on the MAST-U tokamak, here we do not perform the same fitting procedure in FreeGSNKE or Fiesta. We instead use the coil currents and plasma profiles parameters output by EFIT++ as inputs to the static forward GS problems in FreeGSNKE and Fiesta. Given a consistent set of inputs across all codes, we will demonstrate that all three return quantitatively equivalent equilibria.

Carrying out robust validation of static GS solvers[3], against both analytic solutions and real-world tokamak plasmas, is critical for users that require consistent, and more importantly, trustworthy equilibrium calculations. We carry out a rigorous comparison of the poloidal flux quantities, shape control targets, and magnetics measurements between each code, validating FreeGSNKE against both a forward (Fiesta) and reconstruction (EFIT++) equilibrium code. Such in-depth and meticulous validation studies are rarely carried out for new or existing equilibrium solvers and as such we hope to make more consistent and quantitative validation possible by making the scripts required to do so publicly available.

1.2. Related validation studies

In this brief section, we will discuss a few validation efforts concerning static forward GS solvers, omitting inverse and reconstruction solvers here. While a comprehensive review of validation studies for all solver types, both static and dynamic (forward and inverse), would be worthwhile, it is far beyond the scope of the present work and warrants a dedicated future effort.

We begin with Hansen *et al* (2023), who demonstrate that the finite element-based static equilibrium solver within *TokaMaker*, provides accurate solutions to the analytic 'Solov'ev' and 'Spheromak' fixed-boundary equilibrium problems. In addition, they simulate a SPARC equilibrium, comparing the last closed flux surface obtained to one from the inverse solver within *FreeGS* (see section 2 for more information). Beyond this qualitative comparison, however, there is no in-depth quantitative validation of the accuracy of this flux surface

[3] The validation of FreeGSNKE's *dynamic* (evolutive) solver will not feature here and will be addressed in future work.

or any other poloidal flux quantities associated with the SPARC equilibrium. The all-purpose code *NICE* is presented by Faugeras (2020) with a wide demonstration of its reconstruction and dynamic solver capabilities on WEST, TCV, and JET-like equilibria, however, the static solver appears to be untested (directly at least). While Jeon (2015) demonstrate that the static forward solver within *TES* can simulate equilibria for the KSTAR tokamak, their is no explicit investigation into the accuracy of the solutions obtained.

The lack of rigorous testing of static forward solvers highlights an urgent need for much more consistent benchmarking and validation for all different types of equilibrium codes that currently exist (or are yet to be developed). The detailed cross-validation provided in this paper is rare due to the complexity of obtaining, setting up, and running different equilibrium codes with varying levels of documentation and access. We hope that the detailed description of our validation process and the availability of the corresponding data will enable easier validation of codes in the future (Pentland *et al* 2025).

1.3. Paper structure

The rest of this paper will be structured as follows. In section 2, we provide an introduction to the three solvers FreeGSNKE, Fiesta, and EFIT++, briefly discussing their capabilities and prior usage in different areas of tokamak equilibrium modelling. In section 3, we outline the free–boundary static forward GS problem and note differences between the FreeGSNKE and Fiesta solution methods. Following this, we supply a more detailed description of the MAST-U machine and other more specific inputs required by each solver in section 4.

In section 5, we present our numerical experiments, focusing on two different MAST-U shots, one featuring a conventional divertor configuration and the other a Super-X (Morris *et al* 2014, 2028). We begin by comparing FreeGSNKE and Fiesta, ensuring that we understand any key differences between the codes and how these differences may filter through when comparing with EFIT++. After this, we begin to assess the differences between equilibria (and other shape targets) from the solvers and those reconstructed from the diagnostics via EFIT++. We find excellent agreement between all quantities assessed and highlight the accuracy of both FreeGSNKE and Fiesta. Finally in section 6, we discuss the implication of these results and close with a few suggestions for avenues of future work.

2. The solvers

In this section, we give a short introduction to the codes described in this paper and briefly discuss their capabilities.

2.1. FreeGSNKE

FreeGSNKE is a Python-based, finite difference, dynamic free–boundary toroidal plasma equilibrium solver developed by Amorisco *et al* (2024) and built as an extension of the publicly available *FreeGS* code (Dudson 2024). FreeGS features a Picard iteration-based static inverse solver for identifying the coil currents required to maintain different types of plasma configuration prior to experimentation—a brief introduction to constrained analysis can be found in Jeon (2015)[Sec. II.7.]. In conjunction with other equilibrium codes, FreeGS has been used extensively in recent years for the design of various tokamaks. To our knowledge, FreeGS has aided the design of SPARC (Creely *et al* 2020), KSTAR (Lee *et al* 2021), WEST (Maquet *et al* 2023), Thailand Tokamak-1 (S. Sangaroon *et al* 2023), and MANTA (MANTA Team 2023). It was also used in the design of ARC (de Boucaud *et al* 2022), with further work on DIII-D EFIT reconstructed equilibria, and to design COMPASS-U (Kripner *et al* 2018), alongside Freebie (Artaud and Kim 2012) and Fiesta, and to help develop the BLUEPRINT framework (Coleman and McIntosh 2020).

FreeGSNKE inherits the FreeGS inverse solver and introduces a static forward solver that uses a Newton–Krylov method (see e.g. Knoll and Keyes (2004); Carpanese (2021)) to overcome the well-known numerical instability affecting Picard iterations. Also introduced is a solver for the evolutive (dynamic) equilibrium problem, also based on the Newton–Krylov method. In the dynamic problem, Poynting's theorem is enforced on the plasma, coupling the circuit equations (that govern currents in the active coils/passive structures) and the GS equation itself (Amorisco *et al* 2024).

These features, as well as the widespread validation and use of the underlying FreeGS code, make FreeGSNKE a particularly versatile tool for studying the shape and control of plasma equilibria. Its compatibility with other Python libraries, especially those with machine learning capabilities, facilitate its future development and integration with other plasma modelling codes. For example, FreeGSNKE has been used to emulate scenario and control design in a MAST-U-like tokamak by Agnello *et al* (2024), where their objective was to emulate flux quantities and shape targets (some of which we calculate here) based on a training library of input plasma profile parameters and active conductor currents.

The static forward solver in FreeGSNKE has been validated against analytic solutions of the GS equation (Amorisco *et al* 2024) and so now we go a step further by implementing the full MAST-U machine description and validate against EFIT++ reconstructions.

2.2. Fiesta

Fiesta is a free–boundary static equilibrium solver written in MATLAB and developed by Cunningham (2013). In addition to being able to carry out forward and inverse equilibrium calculations, it is also capable of linearised dynamic modelling using the RZIp rigid plasma framework (Coutlis *et al* 1999). It has been used to inform design choices and carry out equilibrium analyses on a number of tokamaks including JET, DIII-D, NSTX, TCV, MAST(-U) (Windridge *et al* 2011, Cunningham 2013), MEDUSA-CR (Araya-Solano *et al* 2021), COMPASS-U (Vondracek *et al* 2021), SMART (Doyle *et al* 2021, Mancini *et al* 2023), STEP (Hudoba *et al* 2023), and EU-DEMO (Morris *et al* 2021).

Having already been used to simulate MAST(-U) and other tokamak equilibria, we run Fiesta alongside FreeGSNKE to demonstrate they both return quantitatively equivalent results given the same set of input data from EFIT++. This cross-validation process should also help identify and explain any differences between the two different implementations.

2.3. EFIT++

EFIT, first proposed by Lao *et al* (1985), is a computational method for solving the reconstruction problem and is widely used as a first port of call for 'fitting' plasma equilbria to measurement data from diagnostics within real-world tokamaks. These measurements come from diagnostics such as poloidal flux loops, pickup coils, Rogowski coils, motional Stark effect (MSE), and Thomson scattering systems, which are strategically located at key locations around the tokamak. Written in Fortran, EFIT is used primarily for post-shot equilibrium reconstruction and has been implemented on a number of different tokamak devices (see below). Our focus is on EFIT++, a substantial re-write in which the original EFIT code has been wrapped in a C++ driver to handle data flow, which in turn is wrapped in a highly configurable Python layer for input and output checking. It is currently in use on the MAST-U tokamak (Appel *et al* 2006) and was previously deployed on JET (Appel and Lupelli 2018). We note that EFIT++ is run routinely for all MAST-U plasma shots using magnetic diagnostic data only and, if available, MSE data to improve the accuracy of core profiles. In addition to this, EFIT++ is also set up to use Thomson scattering data if required (Berkery *et al* 2021, MASTUpgrade Team *et al* 2022).

To solve the inverse problem, EFIT++ requires descriptions of the plasma pressure and toroidal current profiles which are typically expressed using basis functions (whose coefficients are to be adjusted during the fitting process). Next, the linearised GS equation is solved using an initial guess for the poloidal flux. The feasibility of the calculated flux with respect to the diagnostic measurement data is then measured by solving a linearised least-squares minimisation problem. During this process, the variable parameters such as the conductor currents and profile coefficients are adjusted to improve the fit. This iterative process repeats until the conductor currents, profile coefficients, and poloidal flux, together, return a valid solution to the GS equation at the required tolerance. For more technical details, refer to Lao *et al* (2005), Appel and Lupelli (2018), and Bao *et al* (2023).

Different versions of EFIT, each with their own configurations and modifications, have been used for equilibrium reconstruction on a vast array of tokamak devices. Without providing an exhaustive list, it has been deployed on JET (Appel and Lupelli, 2018), MAST(-U) (MAST Upgrade Team *et al* 2022), EAST (Bao *et al* 2023), DIII-D (Lao *et al* 2005), START (Appel *et al* 2001), KSTAR (Lee *et al* 1999), NSTX (Sabbagh *et al* 2001), and ITER (Lao *et al* 2022). Given its history of widespread use on many different tokamak devices, we use EFIT++ as a source of trusted reference equilibria, against which to compare those produced by FreeGSNKE and Fiesta.

3. The static forward Grad–Shafranov problem

In this paper, we are interested in solving the GS equation

$$\Delta^*\psi = -\mu_0 R \underbrace{(J_p + J_c)}_{=J_\phi}, \quad (R, Z) \in \Omega, \tag{3.1}$$

in the cylindrical coordinate system (R, ϕ, Z) for the poloidal flux $\psi(R, Z)$[4] (Grad and Rubin 1958, Shafranov 1958). This equation describes the equilibrium of a magnetically confined plasma in which the plasma pressure and magnetic forces acting upon it are in balance. It can be derived by exploiting toroidal symmetry in the ideal MHD equations (see Jardin (2010)[Chp. 4]).

[4] Note that some numerical solvers (e.g. Fiesta) define ψ using the Weber whereas some (e.g. FreeGSNKE and EFIT++) define it using the Weber$/2\pi$.

In (3.1), μ_0 represents magnetic permeability in a vacuum and $\Delta^* \coloneqq R\partial_R R^{-1}\partial_R + \partial_{ZZ}$ is a linear elliptic operator. The toroidal current density $J_\phi(\psi, R, Z) \coloneqq J_p(\psi, R, Z) + J_c(R, Z)$ contains a contribution from both the plasma J_p and any toroidally symmetric conducting metal structures external to the plasma J_c (e.g. active poloidal field coils and passive structures around the tokamak). The total poloidal flux $\psi \coloneqq \psi_p + \psi_c$ is also made up of a plasma ψ_p and external conductor ψ_c contribution. We wish to solve (3.1) over a two-dimensional computational domain $\Omega \coloneqq \Omega_p \cup \Omega'$ where Ω_p represents the plasma region[5] and Ω' is its complement.

The plasma current density, non-zero only within the plasma region Ω_p, takes the form

$$J_p(\psi, R, Z) = R\frac{\mathrm{d}p}{\mathrm{d}\psi} + \frac{1}{\mu_0 R}F\frac{\mathrm{d}F}{\mathrm{d}\psi}, \quad (R, Z) \in \Omega_p, \tag{3.2}$$

where $p \coloneqq p(\psi)$ is the isotropic plasma pressure profile and $F \coloneqq F(\psi) = RB_\phi$ is the toroidal magnetic field profile (B_ϕ is the toroidal component of the magnetic field). The particular choice of profile functions used in J_p will be discussed in section 4. The current density generated by N_c external conductors is given by

$$\begin{aligned} J_c(R, Z) &= \sum_{j=1}^{N_c} \frac{I_j^c(R, Z)}{A_j^c}, \quad (R, Z) \in \Omega, \\ I_j^c(R, Z) &= \begin{cases} I_j^c & \text{if} \quad (R, Z) \in \Omega_j^c, \\ 0 & \text{elsewhere}, \end{cases} \end{aligned} \tag{3.3}$$

where Ω_j^c, I_j^c, and A_j^c are the domain region, current, and cross-sectional area of the j^{th} conductor, respectively. Note that external conductors can lie inside Ω as well as outside of it.

To complete the free–boundary problem, an appropriate Dirichlet boundary condition must also be specified on the domain boundary $\partial\Omega$—which we discuss in the next section. The dependence of J_p on ψ makes (3.1) a nonlinear elliptic partial differential equation.

3.1. Solving the problem

Here, we briefly outline the steps typically carried out when numerically[6] solving the static free–boundary (forward) GS problem (Jardin 2010, Jeon 2015). For more specific details on how each of the solvers do this in practice, we refer the reader to the respective code documentation.

Before solving, we assume that a number of input parameters have already been provided by the user including: a machine (tokamak) description, conducting structure (active coil and passive structure) currents, and plasma profile functions (and parameters). More details on the specific inputs required for generating free–boundary equilibria on MAST-U with each of the codes will be described in section 4.

Step one

Denote the total flux by $\psi^{(n)}(R, Z)$, where $n = 0, 1, \ldots$ is the iteration number, and generate an appropriate guess $\psi^{(0)}$ to initialise the solver[7].

Step two

Calculate the values of the flux on the computational boundary $\partial\Omega$ (i.e. the Dirichlet boundary condition) using

$$\begin{aligned} \psi^{(n)}|_{\partial\Omega} = &\int_{\Omega_p} G(R, Z; R', Z')J_p(\psi^{(n)}, R', Z') \quad \mathrm{d}R'\mathrm{d}Z' \\ &+ \sum_{j=1}^{N_c} \frac{1}{A_j^c}\int_{\Omega_j^c} G(R, Z; R', Z')I_j^c(R', Z') \quad \mathrm{d}R'\mathrm{d}Z', \end{aligned} \tag{3.4}$$

where the first and second terms are the contributions from $\psi_p^{(n)}$ and $\psi_c^{(n)}$ on the boundary, respectively. G is a Green's function for the operator Δ^* containing elliptic integrals of the first and second kind—it can be calculated by solving (3.1) with ψ_c alone (see Jardin (2010)[Chp. 4.6.3]). To calculate (3.4), the plasma domain Ω_p (i.e. the area contained within the last closed flux surface) needs to be identified—see Jeon (2015)[Sec. 5] for how to do this. Once found, the integral itself can be calculated a number of different ways, for example, using von Hagenow's method (Jardin 2010)[Chp. 4.6.4].

Step three

To solve the nonlinear problem, both EFIT++ and Fiesta use Picard iterations (Kelley 1995), where the n-th iteration consists of calculating the total flux $\psi^{(n+1)}$ according to

[5] The boundary of Ω_p is defined as the closed (R, Z) contour in Ω that passes through the X-point closest to the magnetic axis (see closed red contour in figure 1).

[6] Analytic solutions to the GS problem do exist in limited cases—see Cerfon and Freidberg (2010) for some examples.

[7] Note that $\psi_c^{(n)}$ is known exactly (it is given by the second term in (3.4)) and so we only require an initial guess for $\psi_p^{(0)}$.

$$\Delta^*\psi^{(n+1)} = -\mu_0 R J_\phi(\psi^{(n)}, R, Z), \quad (R, Z) \in \Omega, \tag{3.5}$$

together with boundary condition (3.4). In a finite difference implementation this requires spatially discretising the elliptic operator Δ^*. For example, FreeGSNKE uses fourth-order accurate finite differences while Fiesta uses a second-order accurate (fast) discrete sine transform.

Step four

Check whether or not the solution meets a pre-specified tolerance, e.g. a relative difference such as

$$\frac{\max|\psi^{(n+1)} - \psi^{(n)}|}{\max(\psi^{(n)}) - \min(\psi^{(n)})} < \varepsilon. \tag{3.6}$$

If so, we stop the iterations, otherwise we continue.

Both FreeGSNKE and Fiesta are set to use the same relative tolerance $\varepsilon = 1\text{e-}6$ and while FreeGSNKE uses the criterion in (3.6), we should note that Fiesta uses a slightly different relative criterion based on values of J_p at successive iterations instead of ψ. This should make little difference to the comparison.

Comments

Picard iterations are very effective at tackling inverse GS problems, which is the primary use case for EFIT++ and Fiesta. However, it is well-known that these iterations are unstable when applied to forward GS problems. This manifests itself in the form of vertically unstable equilbria that artificially move between successive Picard iterations (Carpanese 2021). This arises as a result of a combination of known physical instabilities in highly-elongated plasmas and mathematical features of the Picard method itself (which stem from a combination of steep gradients in the nonlinear function and a poor initial guess to the solution). Newton-based methods can overcome this instability (see e.g. Carpanese (2021)). FreeGSNKE implements a Jacobian-free Newton–Krylov method (see Amorisco *et al*. (2024)[App. 1] for further details). This is used to solve directly for the roots, ψ, of

$$\Delta^*\psi + \mu_0 R J_\phi(\psi, R, Z) = 0, \quad (R, Z) \in \Omega. \tag{3.7}$$

As with the Picard iteration, solving this problem still requires an appropriate initial guess and the calculation of the (nonlinear) boundary condition (3.4).

4. Input parameters (for MAST-U)

To solve the forward problem, we need to ensure that the inputs to both FreeGSNKE and Fiesta are consistent with those used by EFIT++ on MAST-U. We require:

1. an accurate and representative MAST-U machine description containing the:
   a) active poloidal field coils.
   b) passive structures.
   c) limiter/wall structure.
   d) poloidal fluxloops and magnetic pickup coils.
2. the fitted values of the coil currents of both active coil and passive structures.
3. the functional form chosen for the plasma profile functions and the corresponding fitted parameter values.
4. any additional parameters specific to either FreeGSNKE or Fiesta.

In the following sections, we outline how these inputs are configured for each of the codes. We note that the fluxloops and magnetic pickup coils are *not required* to solve the forward problem, but rather are used during the post-simulation analysis in section 5.

4.1. Machine description

The following machine description had already been implemented in both EFIT++ and Fiesta and has now been set up in FreeGSNKE. We note here that numerical experiments (in section 5) across all codes are simulated on a 65×65 computational grid on $\Omega = [0.06, 2.0] \times [-2.2, 2.2]$, as this is the resolution EFIT++ is run at during MAST-U reconstructions.

4.1.1. Active coils

MAST-U contains 12 active poloidal field coils whose voltages can be varied for shaping and controlling the plasma (Ryan *et al* 2023). In figure 1, we display a poloidal cross-section of the machine (as is implemented in all three codes) with an example equilibrium from a MAST-U shot (all simulated and plotted using FreeGSNKE). The solenoid, named P1 on MAST-U, generates plasma current and a poloidal magnetic field while P4/P5/PC (the latter of which is not currently connected to the machine) are used for core radial position and shape control. P6 is used for core vertical control, D1/D2/D3/PX for X-point positioning and divertor leg control, and DP for further X-point positioning and flux expansion. Coils D5 and D6/D7 are used for Super-X leg radius and expansion control, respectively.

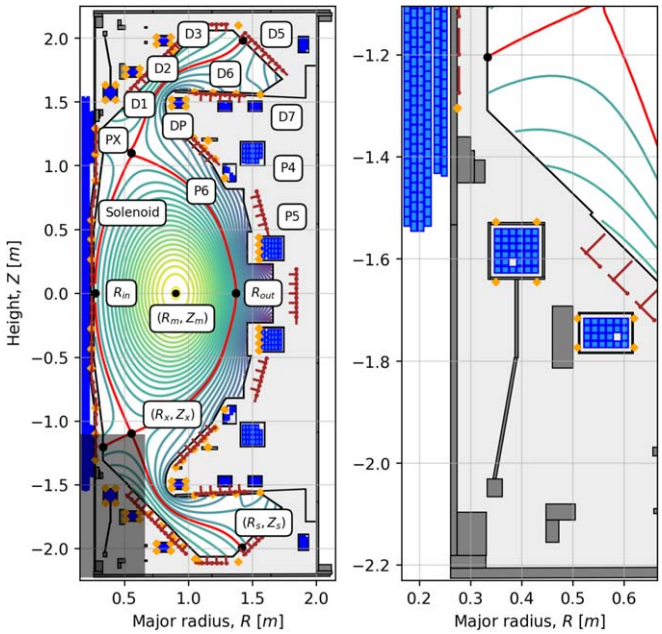

Figure 1. Left: poloidal cross-section of the MAST-U machine (as used across all three codes) with FreeGSNKE-simulated ψ contours of shot 45292 ($t = 0.55$s). Shown are the 12 active coils (blue), the passive structures (dark grey), the limiter/wall (black line, white interior), and the locations of shape targets to be tracked in experiments later on. Also shown are locations/orientations of magnetic diagnostics, including the flux loops (orange diamonds) and the pickup coils (brown dots/lines). Right: magnified view of the lower left corner of the machine (dark shaded box on left figure), showing individual filaments/windings inside some of the active coils and passive structures.

All active coils (except the solenoid) have an upper (labelled in figure 1) and lower component (not labelled) that are wired together in the same circuit. All upper and lower coils are wired in series, except for the P6 coil, whose upper and lower components are connected in anti-series so that it can be used for vertical plasma control. Each coil consists of a number of filaments/windings (plotted as small blue rectangles on the right hand side of figure 1) each with their own central position (R, Z), width and height (dR, dZ), polarity ($+1$ in series, -1 in anti-series), and current multiplier factor (used for the solenoid only). For the scope of the poloidal field, individual windings are modelled as infinitesimally thin toroidal filaments in both FreeGSNKE and Fiesta. Each filament also features its own resistivity value, however, this is not used here where we only deal with static equilibria.

We should note that when EFIT++ fits the active coil currents to diagnostic data, it *does not* treat the upper and lower windings of the same active coil as being linked in series. Instead, current values in the upper and lower windings are measured using independent Rogowski coils[8] and are therefore fit to slightly different values. The relative difference between the upper/lower coil current values is very small and so using this configuration makes little difference to equilibria generated by EFIT++. We refer to this configuration as having *asymmetric* (or *up-down independent*) coil currents. For both FreeGSNKE and Fiesta, we have the option to model the pairs of active coils as either symmetric (connected in series/anti-series as they are in the real MAST-U machine) or asymmetric (as in EFIT++). All of the experiments presented in section 5 are carried out using the asymmetric coil setup so that we can recreate the EFIT++ configuration as closely as possible.

4.1.2. Passive structures

Both the active coils and the plasma itself induce significant eddy currents in the toroidally continuous conducting structures within MAST-U (McArdle & Taylor 2008, Berkery *et al* 2021). This is especially the case in spherical tokamak devices, due to the close proximity of passive structures to the plasma core and active coils. These currents significantly impact the plasma shape and position, making their inclusion in the modelling process essential to obtaining accurate equilibrium simulations.

The complete MAST-U machine description includes a total of 150 passive structures, making up the vessel, centre column, support structures, gas baffles, coil cases, etc. This number excludes a few structures that are not included in the EFIT++ model. These include the graphite tiles (which do not carry much current) and the

[8] The active coil currents are approximated using the difference between measured internal (coil only) and external (coil plus coil case) Rogowski coil currents (Ryan *et al* 2023) and are then fit (alongside all other quantities of interest). The same process is used to fit coil case currents—see section 4.2.

cryopump (which contains a toroidal break to prevent large toroidal currents flowing around the machine). Each passive structure is represented by a parallelogram in the poloidal plane, defined by its central position (R, Z), width and height (dR, dZ) and two angles, (θ_1, θ_2)[9]. Such parallelograms can be seen on the right hand side of figure 1. Given that all three codes require these parallelogram structures to simulate equilibria in the (R, Z) plane, we should note that this passive structure model is a reduced axisymmetric representation of the true three-dimensional MAST-U vessel which contains toroidal breaks for vessel ports among other things—see Berkery *et al* (2021)[figure 12] for a depiction of the full 3D model.

Both FreeGSNKE and Fiesta model the poloidal field associated with each passive structure by uniformly distributing its current density over the poloidal cross-section. This can be done by 'refining' (i.e. subdividing) each passive structure into individual filaments. We revisit this in section 4.2 when we discuss how to assign currents to the passive structures.

4.1.3. Limiter/wall structure

The purpose of the limiter/wall in FreeGSNKE and Fiesta is to confine the boundary of the plasma. In all three codes it is described by 98 pairs of (R, Z) coordinates that form the closed polygonal shape seen in figure 1 (enclosing the flux contours). The plasma core is forced to reside within the limiter region, with the last closed flux surface being either fully contained in this region or tangent to its polygonal edge. The limiter and the wall are taken to be the same.

4.1.4. Fluxloops and magnetic pickup coils

Magnetic diagnostics are crucial for stable plasma control and equilibrium reconstruction, among other tasks. Although MAST-U is equipped with several different types of magnetic diagnostics such as Rogowski, saddle, and Mirnov coils, here we focus on the setup of the flux loops and magnetic pickup coils within FreeGSNKE and EFIT++. For a comprehensive overview of MAST-U's magnetic diagnostics, refer to Ryan *et al* (2023).

A fluxloop in MAST-U is a copper cable wound in a single toroidal loop, located at a fixed poloidal location (R, Z). There are currently 102 fluxloops installed at various poloidal locations on MAST-U where they are used to directly measure the $\psi(R, Z)$ at their specific locations (represented by the orange diamonds in figure 1).

The magnetic pickup coils are multi-turn copper coils that measure the strength of the magnetic field (in Tesla) orthogonal to its orientation. MAST-U contains 354 pickup coils, each positioned at a fixed location[10](R, ϕ, Z) with (normalised) orientation $\hat{\boldsymbol{n}} = (\hat{R}, \hat{\phi}, \hat{Z})$. The magnetic field strength can be determined by calculating $\boldsymbol{B} \cdot \hat{\boldsymbol{n}}$, where

$$\boldsymbol{B}(R, \phi, Z) = (B_R, B_\phi, B_Z) = \left(-\frac{1}{R}\frac{\mathrm{d}\psi}{\mathrm{d}Z}, \frac{F(\psi)}{R}, \frac{1}{R}\frac{\mathrm{d}\psi}{\mathrm{d}R}\right).$$

The pickup coils are represented by the brown dots and lines (indicating the locations and orientations of the coils, respectively) in figure 1.

The simulated readings from these two different sets of diagnostics can be straightforwardly calculated (*after* an equilibrium has been simulated) in FreeGSNKE. We stress that experimental diagnostic measurements from MAST-U are not required to solve the static forward GS problem. Later in section 5, we will compare the simulated readings from FreeGSNKE with the measurement data from MAST-U (i.e. data used to carry out the EFIT++ reconstruction) over the course of a shot to further validate the accuracy of the FreeGSNKE simulations.

4.2. Assigning currents

4.2.1. Active coils

As mentioned before, both FreeGSNKE and Fiesta have the option to use up-down symmetric or asymmetric (independent) active coil current assignments. To set the coil currents in the asymmetric setting, we assign the individually calculated upper/lower coil currents from EFIT++ directly to the corresponding coils in FreeGSNKE and Fiesta without modification. If we were to use the symmetric coil setting, however, each of the 12 active coils in FreeGSNKE and Fiesta require a single current value. To set each one, we could, for example, take the average of the corresponding upper and lower coil currents from EFIT++, making sure that the correct polarity of each current is also assigned.

4.2.2. Passive structures

Due to the different ways they are modelled in EFIT++, care needs to be taken when assigning the fitted passive structure currents to the 150 structures defined in Fiesta and FreeGSNKE. For example, current values for each

[9] θ_1 is the angle between the horizontal and the base edge of the parallelogram while θ_2 is the angle between the horizontal and right hand edge (i.e. $\theta_1 = \theta_2 = 0$ defines a rectangle).

[10] Due to assumed toroidal symmetry, we effectively ignore ϕ here.

coil case are fit by EFIT++ explicitly (using the Rogowski coil measurements mentioned before), which makes it easy to assign them directly in both FreeGSNKE and Fiesta. Other passive structure currents in the vessel, centre column, gas baffles, and support structures, are not, however, measured (and therefore fit) directly. To reduce the degrees of freedom in EFIT++, these passive structures are modelled in groups, each referring to a single current value[11], thereby reducing the computational runtime and avoiding some issues created by having too much freedom in the distribution of current around the machine (Berkery *et al* 2021)[Sec. 3]. In total, in MAST-U there are 20 groups: 14 for the vacuum vessel, 2 for the gas baffles, 2 for passive stabilisation plates, and 2 for divertor coil supports (Kogan *et al* 2021). To set the correct current for each structure in a group, we follow Berkery *et al* (2021)[Sec. 3] and distribute the group current proportionally to each structure based on its fraction of the total cross-sectional area within the group.

As mentioned before, both FreeGSNKE and Fiesta have the option to 'refine' the 150 passive structures into sets of filaments for improved electromagnetic modelling. This involves taking each parallelogram structure, dividing its area (or length) into filaments of approximately the same size, and then evenly distributing the structure current amongst them uniformly. The density of such filaments over the poloidal section of each structure can be adjusted as desired in FreeGSNKE and Fiesta[12].

4.3. Profile functions

To complete the set of input parameters for the static forward GS problem we need consistent plasma current density profile functions across the codes. In a magnetics-only EFIT++ reconstruction on MAST-U (Appel and Lupelli 2018; MAST Upgrade Team *et al* 2022), the pressure and toroidal current profiles in (3.2) are defined using the following polynomials, sometimes referred to as the 'Lao profiles' as first introduced by Lao *et al* (1985) in the original EFIT code:

$$\begin{aligned}\frac{\mathrm{d}p}{\mathrm{d}\tilde{\psi}} &= \sum_{i=0}^{n_p} \alpha_i \tilde{\psi}^i - \bar{\alpha}\tilde{\psi}^{n_p+1}\sum_{i=0}^{n_p} \alpha_i,\\ F\frac{\mathrm{d}F}{\mathrm{d}\tilde{\psi}} &= \sum_{i=0}^{n_F} \beta_i \tilde{\psi}^i - \bar{\beta}\tilde{\psi}^{n_F+1}\sum_{i=0}^{n_F} \beta_i,\end{aligned} \tag{4.1}$$

with coefficients $\alpha_i,\ \beta_i \in \mathbb{R}$. Note here that

$$\tilde{\psi} = \frac{\psi - \psi_a}{\psi_b - \psi_a} \in [0,\ 1], \tag{4.2}$$

is the normalised poloidal flux where $\psi_a = \psi(R_m, Z_m)$ and $\psi_b = \psi(R_X, Z_X)$ are the values of the flux on the magnetic axis and plasma boundary[13], respectively.

To avoid over-fitting and solution degeneracy problems, EFIT++ uses lower-order polynomials ($n_p = n_F = 1$) for the magnetics-only reconstructions we validate against in section 5. The logical parameters $\bar{\alpha} = \bar{\beta} = 1$ are set to enforce homogeneous Dirichlet boundary conditions on the plasma boundary (i.e. $p'(\tilde{\psi} = 1) = FF'(\tilde{\psi} = 1) = 0$). Neumman boundary conditions (on the profile derivatives) can also be enforced if required—see Berkery *et al* (2021)[Sec. 2].

We therefore assign values of the coefficients α_i and β_i as determined by EFIT++ and proceed to normalise the profile functions using the value of the total plasma current I_p fitted by EFIT++. This step is nominally redundant but represents an additional check that ensures the profile functions set in FreeGSNKE and Fiesta are exactly the same as in EFIT++.

For EFIT++ reconstructions that use data from both the magnetics and the MSE diagnostics, *tension spline* representations of the profiles are used—see appendix Appendix. Although FreeGSNKE can indeed simulate equilibria using spline-based profiles (and a number of other commonly used profiles), we do not present those results here as they are of a comparable accuracy to the magnetics-only results presented in section 5. Additionally, the spline profiles were unavailable in Fiesta for comparison.

4.4. Other parameters and code specifics

Here, we detail a few other parameters that need to be set in order to run the forward solvers in both Fiesta and FreeGSNKE.

Firstly, in Fiesta, we need to specify a `feedback` object that mitigates the vertical instability that manifests itself via the Picard solver. One option (via the `feedback2` object) is to monitor the vertical plasma position

[11] An electromagnetic induction model is used to calculate current values to adopt as priors in the fit—refer to McArdle & Taylor (2008) and Berkery *et al.* (2021) for more details.

[12] In our numerical experiments, FreeGSNKE uses 7, 297 refined filaments with cross-sectional areas ranging from 0.03cm^2 to 2.4cm^2 (median area is 0.13cm^2). In Fiesta, the refinement is carried out slightly differently and uses 7, 030 refined filaments, with areas ranging from 0.16cm^2 to 0.62cm^2 (median 0.24cm^2).

[13] In non-diverted (limited) plasmas, the boundary flux value is instead defined where the plasma contacts the limiter.

during the solve and modify the P6 coil current(s) (i.e. the radial field it produces) to correct the position error. Modifying the P6 coil current would, however, defy the purpose of the comparison with the EFIT++ equilibria. To keep the P6 coil current(s) (and all other inputs currents fixed), we therefore opt to use the `feedback3` object instead—this uses a variation of the method presented by Yoshida *et al* (1986). This object introduces a second (outer) nonlinear solver loop. The inner loop solves for the equilibrium (via the Picard iterations) with respect to a specified magnetic axis location by adding 'synthetic' radial and vertical magnetic fields. The outer loop then minimises these synthetic fields using a gradient search method, returning an optimal solution for the magnetic axis position, and therefore the equilibrium. This additional outer loop drastically slows Fiesta compared to the `feedback2` setting, however, we believe that it is necessary for a direct comparison with FreeGSNKE and EFIT++.

Secondly, both Fiesta and FreeGSNKE require a prescription for the toroidal field. This is provided through a parameter called `irod` in Fiesta, specifying the total current in the central toroidal field conductor bundle, and through the value of $f_{\text{vac}} := RB_{tor}$ in FreeGSNKE. We note, however, that the toroidal field does not affect the equilibrium calculations themselves. Calculation, for example, of safety factors and beta values would be affected, but we do not consider them here. For completeness, in FreeGSNKE we set f_{vac} using the EFIT++ sourced value, and in Fiesta we use $\texttt{irod} = 5e6 f_{\text{vac}}$.

Finally, for the simulations in section 5, Fiesta occasionally struggles to converge for a number of time slices in the two shots shown. This could be due to a combination of the nonlinearity of the GS equation, a poor initial guess for ψ_p (or J_p), and perhaps the instability of the Picard solver. To remedy this, the results we present here are obtained by providing Fiesta with the J_p field calculated by EFIT++, as an initial guess in the Fiesta forward solve—this rectified the non-convergence issues in almost all cases.

5. Numerical experiments: FreeGSNKE vs. Fiesta vs. EFIT++

In this section, we compare the equilbria and related shape targets simulated by all three codes across two different MAST-U shots: one with a conventional divertor configuration and one with a Super-X configuration. We should reiterate that, although we consider EFIT++ to be our reference solver, we have no actual *ground truth* equilibria. Therefore, in our comparisons, we measure *differences* between the equilibria produced by the various codes rather than *errors*.

We start by briefly outlining the steps taken to obtain these results. First, we select the MAST-U shot that we wish to simulate in FreeGSNKE/Fiesta and store[14] the corresponding EFIT++ data that we require for each time slice. This includes the inputs described in section 4 and the poloidal flux/shape target output data that we wish to compare to after we have run FreeGSNKE and Fiesta. After building/loading the machine description in both FreeGSNKE and Fiesta, we then solve the forward GS problem at each time slice sequentially, starting at the first time step for which EFIT++ produces a valid GS equilibrium[15]. For FreeGSNKE, we initialise each simulation with a default initial guess for the plasma flux ψ_p—this is obtained automatically by requiring the presence of an O-point in the total flux within the limiter region. As mentioned before, to ensure convergence, Fiesta is initialised using the J_p field calculated by EFIT++. While this initialisation is already very close to the desired reference GS solution produced by EFIT++, a comparison with the equilibrium on which Fiesta eventually converges is still informative of the code's performance.

5.1. MAST-U shot 45425: Conventional divertor

We first simulate MAST-U shot 45425, which has a flat-top plasma current of approximately 750kA, a double-null shape, and a conventional divertor configuration. The plasma is heated using two neutral beam injection systems delivering a total power of approximately 2.5MW and remains in H-mode confinement for the majority of the shot.

5.1.1. Single time slice

Before analysing the entire shot, we wish to briefly discuss and compare a few minor differences between the flux quantities produced by each code for a single time slice of the shot ($t = 0.7$s). We begin by comparing FreeGSNKE and Fiesta without EFIT++ in figure 2. The left panel displays the ψ contours from FreeGSNKE and an almost perfect overlap of the separatrices from both codes.

[14] We should state here that we do not (re-)run EFIT++, we simply extract existing data generated by a post-shot reconstruction stored in the MAST-U database. Data accessed 14/06/24.

[15] During some of the ramp-up and ramp-down of the plasma, EFIT++ may struggle to converge to a valid GS equilibrium or produce spurious fits (e.g. on the plasma profile coefficients or passive structure currents). In these cases, we exclude these time slices from the comparison.

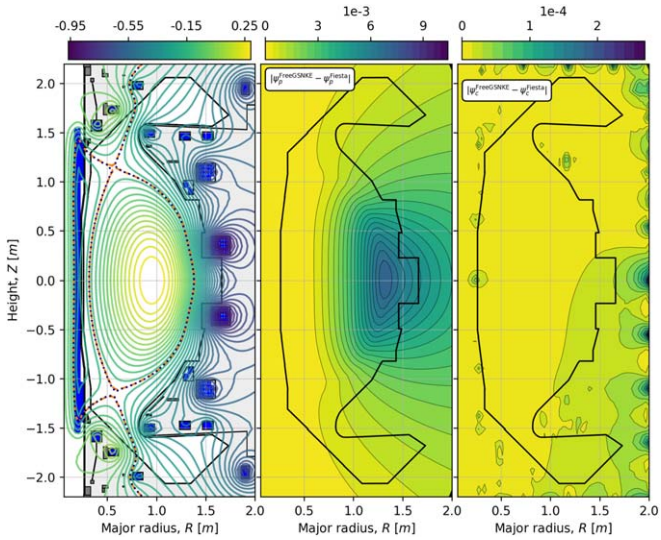

Figure 2. Simulated MAST-U shot 45425 ($t = 0.7$s) equilibrium. Left: separatrices from both Fiesta (orange) and FreeGSNKE (dotted blue) equilibria alongside the FreeGSNKE ψ contours. Centre: absolute difference in ψ_p between the two codes. Right: same as centre but for ψ_c.

We break this down in the centre and right panels. The right panel shows the magnitude of differences in ψ_c, the plasma flux generated by active coils and passive structures. It can be seen that differences are generally small, at a level of $\lesssim$ 2e-4, compared to a total flux that spans a range $|\max(\psi) - \min(\psi)| \sim 1$. These differences appear co-localised with the vessel's passive structures, suggesting they are driven by implementation details in the methods used by either code to distribute the passive structure currents over their poloidal sections[16]. The central panel shows differences in the plasma flux ψ_p. The largest differences are localised in the top-right and bottom-right edge pixels of the discretised domain. We attribute this to implementation details in the way Fiesta imposes the boundary conditions (see also the right panel of figure 3, where the same discrepancy is visible again). The remaining differences are at a level of $\lesssim$ 5e-3. We believe these are largely due to implementation differences in the routines that identify the last closed flux surface of the plasma, rather than being due to the nonlinear solvers themselves. In fact, we find that a comparison between the plasma current density distributions J_p calculated by FreeGSNKE and Fiesta for the same total flux ψ results in differences of the same order of magnitude.

In figure 3, we compare differences in the total flux ψ between all three codes[17]. The left panel is similar to the one seen in figure 2 (centre), with differences between FreeGSNKE and Fiesta dominated by differences in ψ_p as just discussed. Similarly to the differences between FreeGSNKE and Fiesta, the differences between FreeGSNKE/Fiesta and EFIT++ (shown in the centre/right panels, respectively) are largest close to the plasma outer edge, and qualitatively analogous (if not slightly different in magnitude).

It is worth highlighting explicitly that the mismatches shown in figure 2 and figure 3 are, nominally, beyond the relative tolerance used in both FreeGSNKE and Fiesta—recall this was $\varepsilon =$ 1e-6. However, as already mentioned: i) differences in the implementation of the passive structures and ii) differences in routines that build the plasma core mask between the three codes at hand, are responsible for introducing mismatches with similar orders of magnitude to those we are seeing. Besides, as hinted by the left panel in figure 2 and as we show in the following results, this level of difference has a negligible impact on the shape control targets (and therefore for most practical modelling purposes).

5.1.2. Entire shot

Over the course of the entire shot, the differences in ψ (for both codes) remain at the levels seen in figure 3. For FreeGSNKE, the median differences in ψ_a and ψ_b over the shot are 1e-3 and 6e-4, respectively. For Fiesta, the corresponding differences are on average 3e-3 and 1e-3, respectively.

In the top panel of figure 4, we plot the evolution of the separatrices produced by all three codes over the lifetime of the shot, observing an excellent match in the majority of shot times. We plot two metrics in the lower panels of figure 4 to compare the core plasma boundary (divertor legs not included). For both metrics, we find

[16] We also explicitly checked any differences in the flux contribution from the active coils alone and found them to be $\mathcal{O}(10^{-15})$.

[17] We should note that EFIT++ did not produce a breakdown of ψ into ψ_p and ψ_c, making the comparison of plasma and conductor flux contributions more difficult.

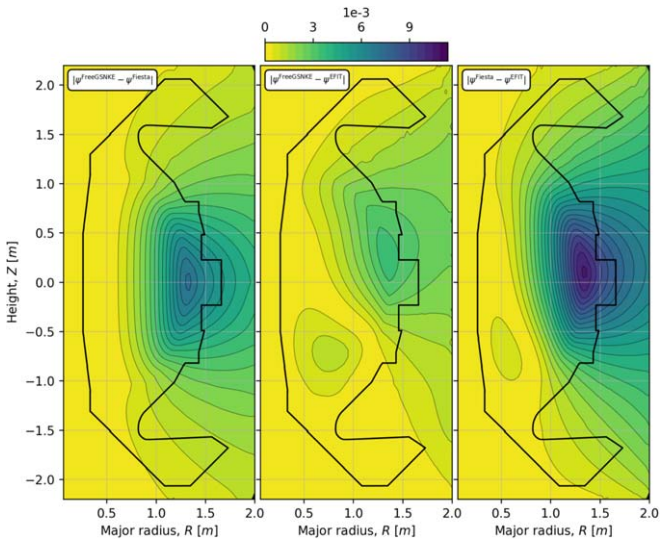

Figure 3. Differences in the total flux ψ between all three codes for MAST-U shot 45425 ($t = 0.7$s). Left: absolute difference in ψ between FreeGSNKE and Fiesta. Centre: difference between FreeGSNKE and EFIT++. Right: difference between Fiesta and EFIT++.

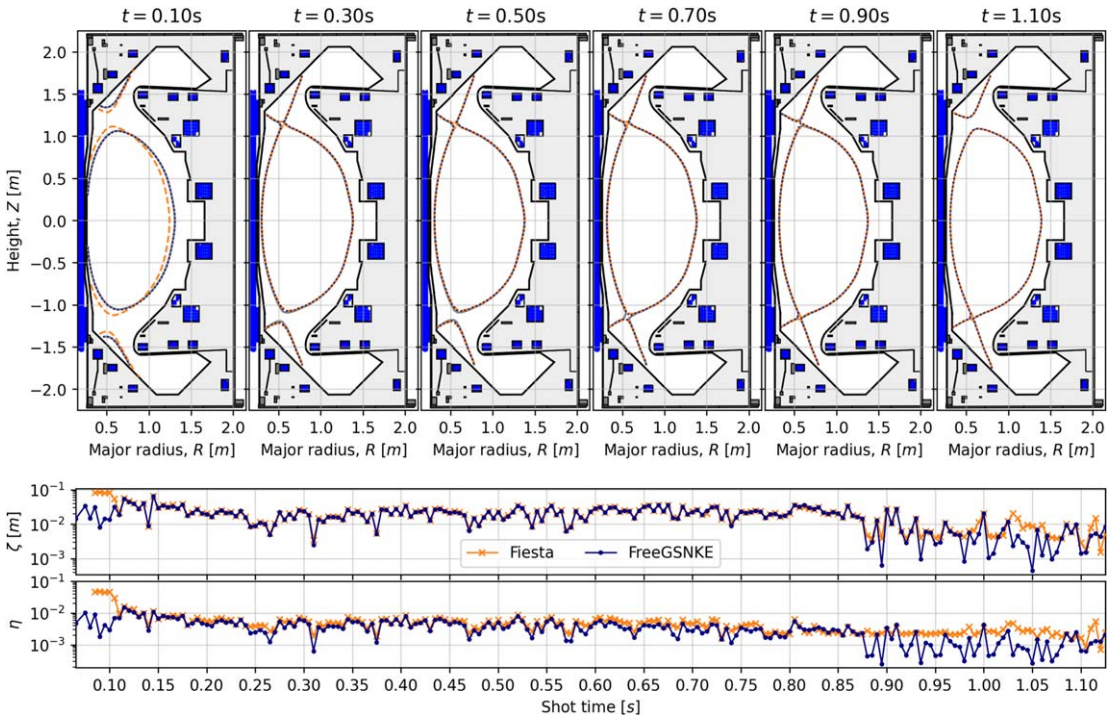

Figure 4. Top: evolution of EFIT++ (solid grey), Fiesta (dashed orange), and FreeGSNKE (dotted blue) separatrices at different shot times. Middle: evolution of the ζ metric from figure 2 over time for Fiesta (orange) and FreeGSNKE (blue) compared to EFIT++ (divertor legs not included). Bottom: similarly, the evolution of the η metric from figure 3 over time.

360 (R, Z) points that lie on the boundary $\partial\Omega_p$ of each code, where each point is equally spaced in the geometric poloidal angle, centred on the magnetic axis (of the EFIT++ core).

The first metric, ζ, quantifies the largest Euclidean distance between corresponding (R, Z) points on the FreeGSNKE/Fiesta boundary and the EFIT++ boundary. This is defined as

$$\zeta(\partial\Omega_p^1, \partial\Omega_p^2) \coloneqq \max_{j=1,\ldots,360} \| \partial\Omega_{p,j}^1 - \partial\Omega_{p,j}^2 \|_2, \tag{5.1}$$

where $\partial\Omega_{p,j}^1$ denotes the jth (R, Z) point on the boundary from the first code (same for the second code). An illustration of how this metric is similarly calculated can be found in Stewart *et al* (2023)[figure 2]. We find

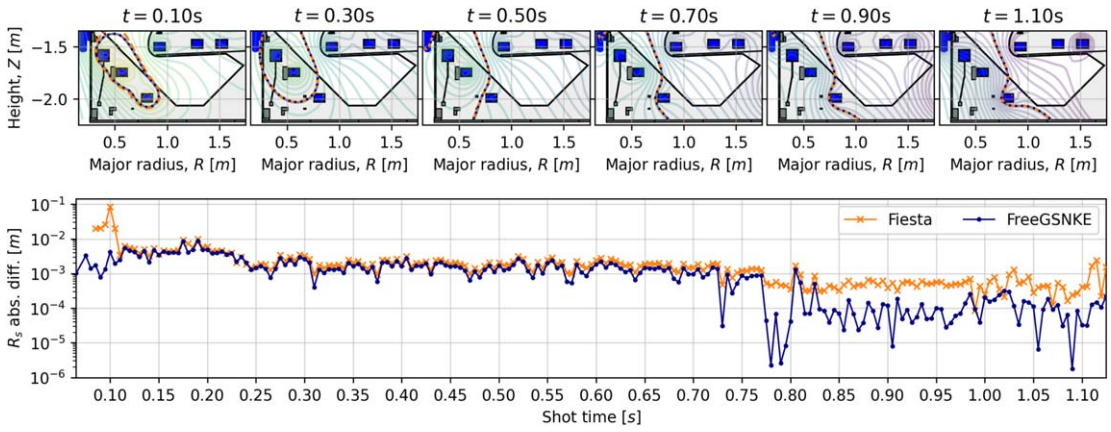

Figure 5. Top: evolution of EFIT++ (solid grey), Fiesta (dashed orange), and FreeGSNKE (dotted blue) lower divertor strikepoints at different shot times. Bottom: absolute difference between between EFIT++ and Fiesta/FreeGSNKE for R_s (errors are at the same level for Z_s).

ζ to be below 3.5cm in 95% of cases for FreeGSNKE equilibria (in 96% of cases the median distance is below 1cm). For the case of Fiesta, ζ is below 4.2cm for 95% of the time slices (and the median is below 1.13cm for 96% of cases).

The second metric, η, first presented by Bardsley *et al* (2024), is defined as

$$\eta(\Omega_p^1, \Omega_p^2) := \frac{|\Omega_p^1 \cup \Omega_p^2| - |\Omega_p^1 \cap \Omega_p^2|}{|\Omega_p^1| + |\Omega_p^2|} \in [0, 1], \tag{5.2}$$

where $|\cdot|$ denotes the cross-sectional area of a domain in the poloidal plane. This dimensionless parameter quantifies the ratio of the total non-overlapping area of the two plasma domains to the sum of their areas. Values closer to zero indicate a good match between the plasma domains and we can see from the results that both codes yield $\eta \leqslant 0.01$ for the entire shot (ignoring the early Fiesta time slices).

Any gaps in the time series (and in later plots) are where Fiesta failed to converge to an equilibrium given the tolerance ε—these time slices have been excluded when calculating the quantiles mentioned above. We would note that during these times slices (where the equilibrium is in a limiter-type configuration), Fiesta can converge when using the `feedback2` object mentioned in section 4.

In figure 5 we monitor the evolution of a strikepoint along the lower divertor tiles, again, observing an excellent match from both codes. The difference in (R_s, Z_s) compared to EFIT++ is less than a centimetre for most of the shot—very early times being the exception in the case of Fiesta. Note that while the difference in Z_s is not explicitly shown, it is of the same order as that of R_s.

In figure 6, we plot the differences in the magnetic axis (R_m, Z_m), the midplane inner radius R_{in}, and the midplane outer radius R_{out} (recall figure 1). Typical differences in the inner/outer radii are of the order of millimetres for FreeGSNKE, with only marginally higher values for Fiesta. With respect to the magnetic axis, differences in both codes vs. EFIT++ track one another to sub-centimetre precision as well. There are, however, some isolated times during the initial phase of the ramp up where the differences between Fiesta and EFIT++ are significantly higher (see similar differences in later plots) than in later time slices. While FreeGSNKE has found (precisely) the EFIT++ GS equilibria during these early slices, we suspect that Fiesta may have simply found another set of (equally valid) GS equilibria. The physical difference in these (limited, not yet diverted) equilibria can be seen more clearly in figure 4 at $t = 0.10s$. While we do not have a conclusive reason for the presence of multiple GS equilibria here (given the same input parameters), identifying under what conditions these equilibria co-exist may be worth further investigation (Ham and Farrell 2024).

In figure 7, we plot the difference in the lower core chamber X-point over the shot. As we did for the separatrix calculations, we identify all X-points for each code's equilibria using FreeGSNKE's built-in functionality (calculated by finding saddle points of ψ). This was because Fiesta and EFIT++ would inconsistently return only a single X-point, sometimes in the lower core chamber, sometime the upper. There appears to be a small vertical bias towards higher X-points by a few millimetres in both Fiesta and FreeGSNKE—also visible in some of the upper panels of figure 4. Despite this slight bias, both codes are accurate with respect to EFIT++ to within half a centimetre for 98% and 97% of the shot, respectively (100% are within 1cm).

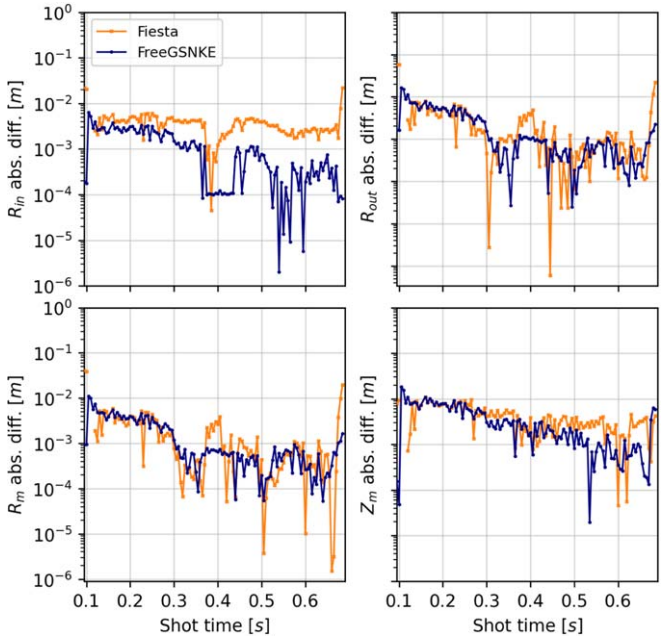

Figure 6. Absolute differences between EFIT++ and Fiesta (orange crosses)/FreeGSNKE (blue dots) shape targets. Top: absolute difference in midplane inner R_{in} and outer R_{out} radii. Bottom: absolute difference in magnetic axis components R_m and Z_m.

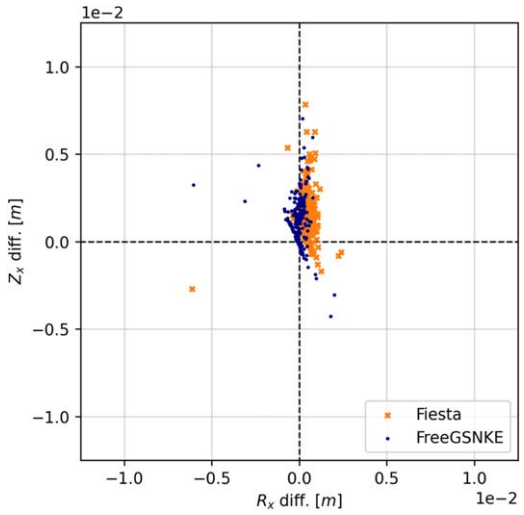

Figure 7. Difference between EFIT++ and Fiesta (orange crosses)/FreeGSNKE (blue dots) lower core chamber X-points (over the entire shot).

The runtimes, both per time slice and cumulatively across the shot, are displayed in figure 8. Median runtimes are 6.7 s per forward solve for Fiesta, 0.07 s for FreeGSNKE. Fiesta is significantly hindered by the use of the `feedback3` object which demands a second nonlinear solver loop to stabilise the Picard iterations[18]. FreeGSNKE makes use of the faster and more stable convergence of the Newton–Krylov method, as well as using `numba` just-in-time compilation for some core routines. As mentioned before, all equilibria were simulated sequentially with both codes. However, given the independence of each time slice, nothing prevents an embarrassingly parallel implementation—though this was not an objective in this paper.

Finally in figure 9, we compare readings from the fluxloops and pickup coils in FreeGSNKE with those used by EFIT++ to reconstruct the equilibria. Note that the measurements are only compared on a fraction of the total number of fluxloops and pickup coils in MAST-U because EFIT++ automatically excludes diagnostics which have failed or whose readings exceed a certain calibration threshold (MAST Upgrade Team *et al* 2022,

[18] However, if one is willing to allow the P6 coil current(s) be modified, Fiesta can run much faster using the `feedback2` object instead—the median runtime was around 0.22 s per forward solve in this case.

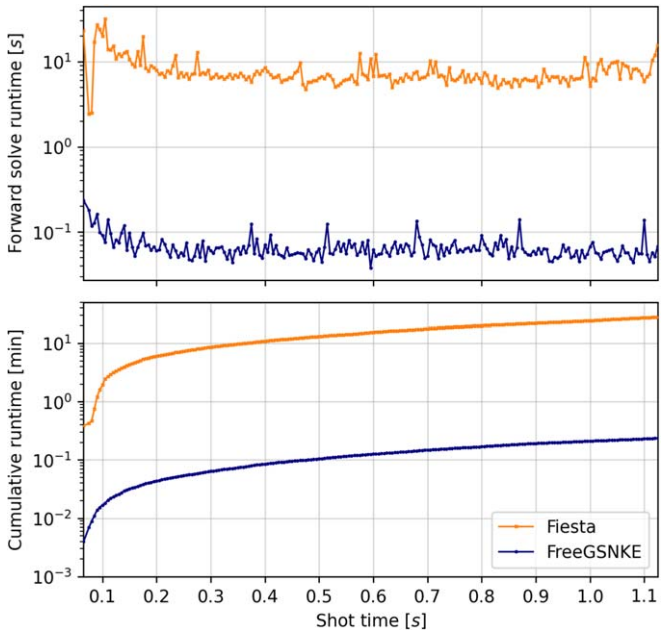

Figure 8. Fiesta (orange) and FreeGSNKE (blue) forward solve runtimes over the shot (in seconds). Top: runtime per forward solve. Bottom: cumulative runtime of forward solves (in total, Fiesta takes 27 min 48 s and FreeGSNKE takes 16 s.

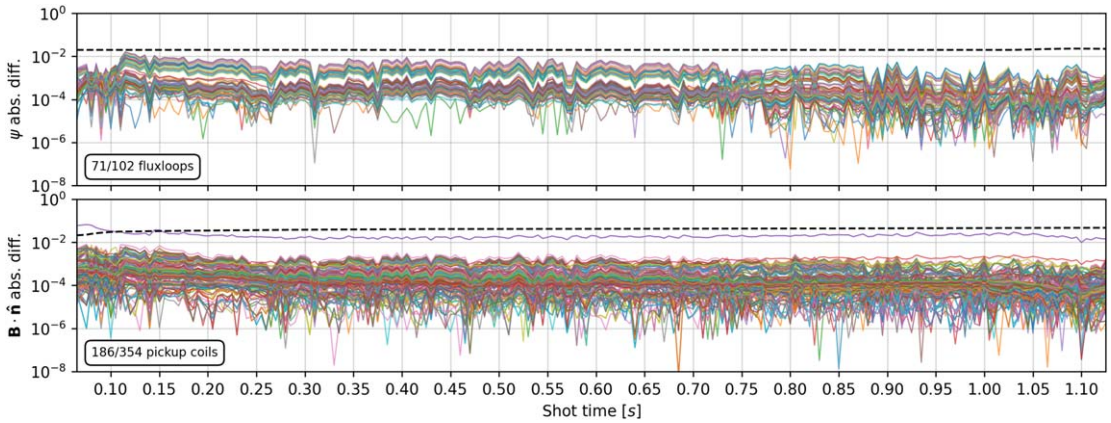

Figure 9. Absolute differences between FreeGSNKE and EFIT++ magnetics diagnostics. Top: absolute difference in poloidal flux ψ, measured at 71 out of a total 102 fluxloops. Bottom: absolute difference in magnetic field strength $\boldsymbol{B} \cdot \hat{\boldsymbol{n}}$, measured at 186 out of a total 354 pickup coils. Each multicoloured line represents an individual diagnostic while the dashed black line represents the (maximum) standard deviation associated with any diagnostic in EFIT++ at a given time slice (i.e. the acceptable level of difference we should find between FreeGSNKE and EFIT++).

Ryan *et al* 2023). As expected, we can see that the absolute difference between readings from both sets of diagnostics is very small and below the (maximum) level of standard deviation assigned to each diagnostic during the EFIT++ reconstruction (i.e. below the dashed black line). The one exception to this was one of the pickup coils (located between the upper P6 and DP coils) that can be seen exceeding this threshold during the ramp-up and remaining consistently higher than other pickups throughout the shot. The measurements from this pickup coil could have been affected by calibration issues or perhaps maybe evaded the EFIT++ faulty probe detection.

5.2. MAST-U shot 45292: Super-X divertor

Here, we focus on MAST-U shot 45292, which has a flat-top plasma current of approximately 750kA, a double-null shape, and a Super-X divertor configuration. The plasma is Ohmically heated, i.e. there is no neutral beam heating, and remains in the L-mode confinement regime throughout the shot.

As we did for the previous shot, we plot the evolution of the separatrices from all three codes over time in figure 10, again discerning a very good agreement between all time slices (including qualitatively on the divertor

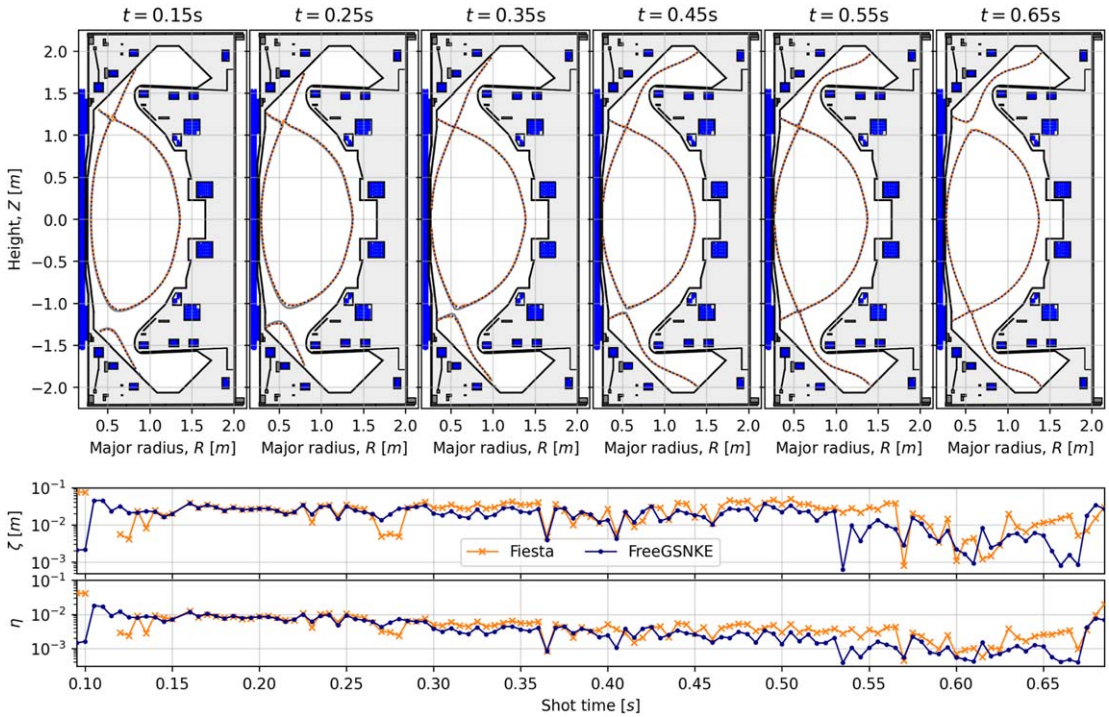

Figure 10. Top: evolution of EFIT++ (solid grey), Fiesta (dashed orange), and FreeGSNKE (dotted blue) separatrices at different shot times. Middle: evolution of the ζ metric from figure 2 over time for Fiesta (orange) and FreeGSNKE (blue) compared to EFIT++ (divertor legs not included). Bottom: similarly, the evolution of the η metric from figure 3 over time.

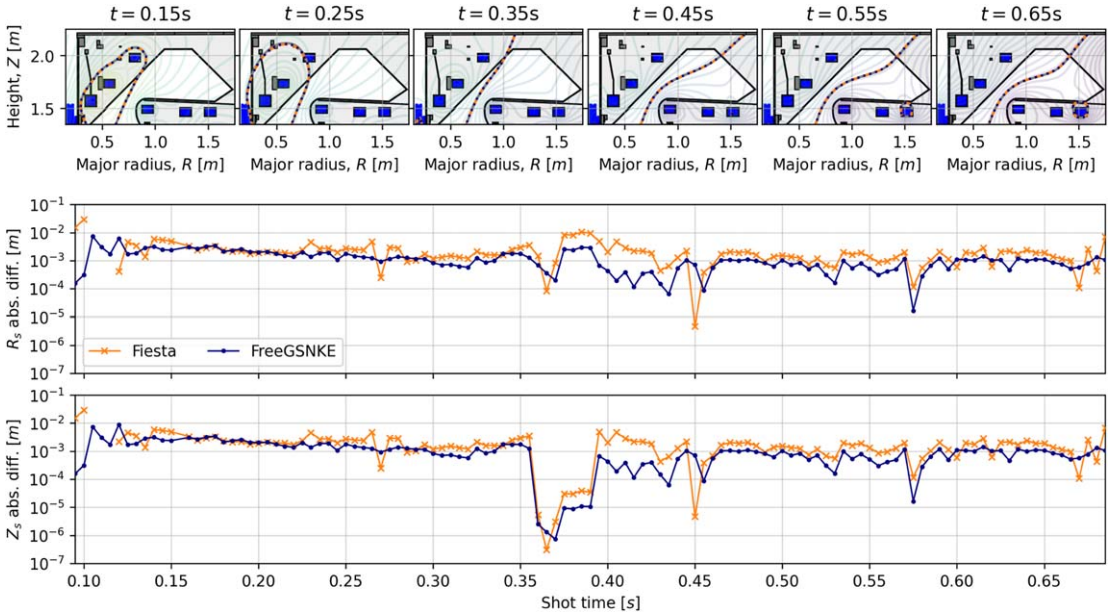

Figure 11. Top: evolution of EFIT++ (solid grey), Fiesta (dashed orange), and FreeGSNKE (dotted blue) lower divertor strikepoints at different shot times. Centre: absolute difference between between EFIT++ and Fiesta/FreeGSNKE for R_s. Bottom: same as centre but for Z_s.

legs). Again, ζ reveals centimetre level differences in the core boundaries (in the worst cases) and η returns values approximately less than 0.01. Given the Super-X configuration, the upper divertor strikepoint now evolves across the tiles. We can see, in figure 11, good agreement with differences in R_s and Z_s remaining at similar levels (though marginally different).

Differences in poloidal fluxes remain at similar levels as seen in the conventional divertor shot while shape targets again match to sub-centimetre precision—see figure 12. Upper core chamber X-points from FreeGSNKE

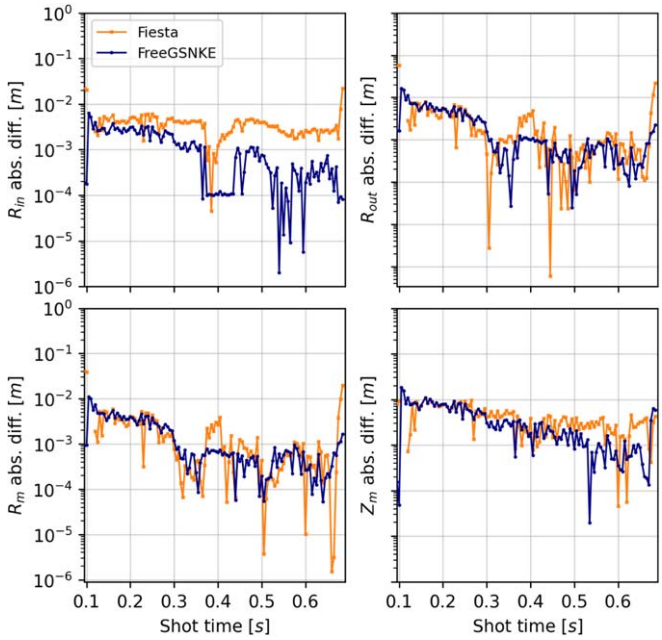

Figure 12. Absolute differences between EFIT++ and Fiesta (orange crosses)/FreeGSNKE (blue dots) shape targets. Top: absolute difference in midplane inner R_{in} and outer R_{out} radii. Bottom: absolute difference in magnetic axis components R_m and Z_m.

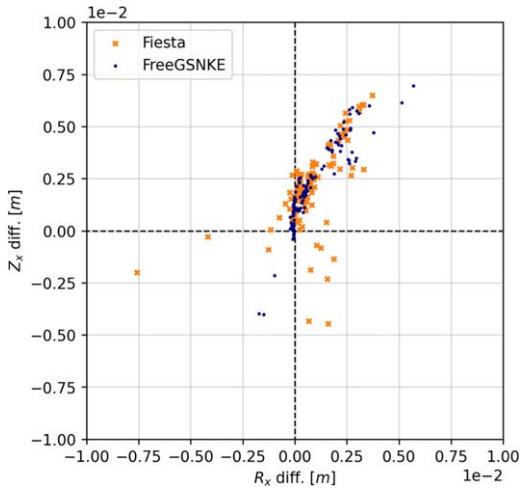

Figure 13. Absolute differences between EFIT++ and Fiesta (orange crosses)/FreeGSNKE (blue dots) lower core chamber X-points (over entire shot).

and Fiesta are again accurate to within half a centimetre for 92% and 87% of time slices shown (100% within 1cm), respectively (see figure 13). Runtimes for both simulations were almost identical to those seen in the previous experiment and are therefore not shown again. To see the additional results not shown here for the Super-X shot, refer to the code repository.

6. Conclusions

In this paper, we have demonstrated that the static forward GS solvers (see section 3) in FreeGSNKE and Fiesta can accurately reproduce equilibria generated by magnetics-only EFIT++ reconstructions on MAST-U.

To achieve this, we began (section 4) by outlining which features of the MAST-U machine would be included in the forward solver machine descriptions, using those that most closely matched the one by EFIT++. We highlighted the capability of both codes being able to model the active poloidal field coils as either up/down symmetric or asymmetric, noting that EFIT++ uses the asymmetric setting. In addition, both codes have the

option to refine passive structures into smaller filaments in order to distribute the induced current in them across their surface areas for better electromagnetic modelling. Following this, we set the conductor currents and prescribed appropriate plasma current density profiles—in this case the polynomial-based "Lao" profiles whose coefficients are determined by EFIT++. In addition to some other code-specific parameters, we then used this computational pipeline to begin simulating the MAST-U equilibria.

The poloidal flux quantities and shape targets generated by the FreeGSNKE and Fiesta simulations in section 5 show excellent agreement with the corresponding quantities from EFIT++ for both MAST-U shots tested. More specifically, separatrices from both codes match those of EFIT++, with the largest distances between the core boundaries found to be on the order of centimetres. Strikepoints, X-points, magnetic axes, and inner/outer midplane radii differences between the codes were simulated to sub-centimetre precision.

The static GS solver in FreeGSNKE has now been validated against both analytic equilibria (see Amorisco *et al* (2024)) and experimental reconstructions from MAST-U (this paper). It has also been shown to produce numerically equivalent equilbria to the Fiesta code, which itself has been validated on experimentally reconstructed equilibria from a number of different tokamak devices (refer back to section 1). Given its user-friendly Python interface and superb computational speed/accuracy, we hope that this work will enable the more widespread adoption of FreeGSNKE for machine learning-based plasma control (e.g. building libraries of plasma equilbria) and for power plant design optimisation studies (e.g. identifying optimal poloidal field coil or magnetic diagnostic locations). We also stress that FreeGSNKE itself is not designed (and therefore not intended) for use in real-time plasma control. Furthermore, we hope that the code and datasets made available with this paper will be of use in validation studies for other equilibrium modelling codes.

Some avenues for future work include validating the dynamic forward GS solver in FreeGSNKE using real-world plasma reconstructions, incorporating more complex/unconstrained plasma profile functions, and making use of data assimilation techniques to carry out probabilistic (uncertainty-aware) equilibrium reconstruction.

Acknowledgments

The authors would like to thank Stephen Dixon and Oliver Bardsley (UKAEA) for some very helpful discussions around the MAST-U data handling and for help in quantifying differences in the plasma boundaries. We would also like to thank Ben Dudson (LLNL) for pointing us in the direction of a number of very useful FreeGS references. This work was part-funded by the FARSCAPE project, a collaboration between UKAEA and the UKRI-STFC Hartree Centre, and by the EPSRC Energy Programme (grant number EP/W006839/1). To obtain further information, please contact PublicationsManager@ukaea.uk.

Data availability statement

The data that support the findings of this study can be found at: http://doi.org/10.14468/26m5-ey02.

Declarations

The authors have no conflicts of interest to declare.

Appendix. Alternate profile functions: tension spline

The pressure and toroidal current profiles for an EFIT++ reconstruction that uses both magnetics and MSE measurements are defined by

$$\frac{\mathrm{d}p}{\mathrm{d}\tilde{\psi}} = \sum_{i=0}^{n_p-1} f_i(\tilde{\psi}) \quad \text{and} \quad F\frac{\mathrm{d}F}{\mathrm{d}\tilde{\psi}} = \sum_{i=0}^{n_F-1} f_i(\tilde{\psi})$$

where $f_i(\tilde{\psi})$ are the *tension spline* (basis) functions (Costantini *et al* 1999, Boulila *et al* 2021). Note that we have slightly abused notation as the functions f_i are not the same for each profile function, they each depend on different parameters (see below). They are defined as

$$f_i(\hat{\psi}) = \begin{cases} g_i^{(1)}(\hat{\psi}) + g_i^{(2)}(\hat{\psi}) + g_i^{(3)}(\hat{\psi}) & \hat{\psi} \in [\hat{\psi}_i, \hat{\psi}_{i+1}), \\ 0 & \text{otherwise}, \end{cases}$$

where

$$g_i^{(1)}(\hat{\psi}) = y_i \frac{\hat{\psi}_{i+1} - \hat{\psi}}{\hat{\psi}_{i+1} - \hat{\psi}_i} + y_{i+1} \frac{\hat{\psi} - \hat{\psi}_i}{\hat{\psi}_{i+1} - \hat{\psi}_i},$$

$$g_i^{(2)}(\hat{\psi}) = \frac{z_i}{\sigma^2} \left[\frac{\sinh(\sigma(\hat{\psi}_{i+1} - \hat{\psi}))}{\sinh(\sigma(\hat{\psi}_{i+1} - \hat{\psi}_i))} - \frac{\hat{\psi}_{i+1} - \hat{\psi}}{\hat{\psi}_{i+1} - \hat{\psi}_i} \right],$$

$$g_i^{(3)}(\hat{\psi}) = \frac{z_{i+1}}{\sigma^2} \left[\frac{\sinh(\sigma(\hat{\psi} - \hat{\psi}_i))}{\sinh(\sigma(\hat{\psi}_{i+1} - \hat{\psi}_i))} - \frac{\hat{\psi} - \hat{\psi}_i}{\hat{\psi}_{i+1} - \hat{\psi}_i} \right].$$

Note that the final function f_{n_p-1} is defined over the entire interval $[\hat{\psi}_{n_p-1}, \hat{\psi}_{n_p}]$ (similarly for f_{n_F-1}).

As mentioned, the parameters in each f_i are different for both profile functions (hence the abuse of notation) and are defined as follows:

- $\hat{\psi}_i$ are the knot points (i.e. locations in interval [0, 1]).
- $y_i = f(\hat{\psi}_i)$ are the values of the profile function at the knot points.
- $z_i = f''(\hat{\psi}_i)$ are the values of the second derivative of the profile function at the knot points.
- $\sigma > 0$ is the tension parameter (sending $\sigma \to \infty$ will results in a piecewise linear spline, whilst smaller σ will results in a smoother spline).

Both the knot points and the tension parameter are provided as inputs to EFIT++ so that it can calculate (fit) the profile function values and second derivatives at the knot points. These parameters are all that are required to reconstruct p' and FF'. The tension spline enables one to specify more complex profile function shapes and, as with the polynomial profiles, different boundary (and sometimes internal) conditions can be enforced by providing EFIT++ with the relevant constraints on the parameter fits (i.e. constraints on specific y_i and z_i).

ORCID iDs

K Pentland https://orcid.org/0000-0001-9530-1890
N C Amorisco https://orcid.org/0000-0002-3205-5335
A Agnello https://orcid.org/0000-0001-9775-0331
G K Holt https://orcid.org/0000-0001-6814-9117
A Ross https://orcid.org/0000-0002-0787-6756
C Vincent https://orcid.org/0000-0002-7227-409X
J Buchanan https://orcid.org/0009-0009-0743-7655
S J P Pamela https://orcid.org/0000-0001-8854-1749
G McArdle https://orcid.org/0000-0003-0888-0105

References

Agnello A, Amorisco N C, Keats A, Holt G K, Buchanan J, Pamela S, Vincent C and McArdle G 2024 Emulation techniques for scenario and classical control design of tokamak plasmas *Phys. Plasmas* **31** 043901

Amorisco N C, Agnello A, Holt G, Mars M, Buchanan J and Pamela S 2024 FreeGSNKE: A python-based dynamic free-boundary toroidal plasma equilibrium solver *Phys. Plasmas* **31** 042517

Appel L C and Lupelli I 2018 Equilibrium reconstruction in an iron core tokamak using a deterministic magnetisation model *Comput. Phys. Commun.* **223** 1–17

Appel L C, Bevir M K and Walsh M J 2001 Equilibrium reconstruction in the START tokamak *Nucl. Fusion* **41** 169

Appel L C, Huysmans G T A, Lao L L, McCarthy P J, Muir D G, Solano E R, Storrs J, Taylor D and Zwingmann W 2006 A unified approach to equilibrium reconstruction *Proc. 33rd EPS Conf.*

Araya-Solano L, Vargas V I, Solano-Piedra R, Ramìrez A A, Hernández-Cisneros M, Coto-Vlchez F, Rojas-Quesada M A, Pérez-Hidalgo J E and Vlchez-Coto F 2021 Implementation of the spherical tokamak MEDUSA-CR *In Proceedings of the 8th International Conference* P7–22

Artaud J F and Kim S H 2012 A new free-boundary equilibrium evolution code, FREEBIE *In 39th EPS Conference & 16th Int. Congress on Plasma Physics* 1178–81

Bao N, Yan X, Wei S and Wang Z 2023 Py-EFIT: a new python package for plasma equilibrium reconstruction on EAST tokamak *Comput. Phys. Commun.* **282** 108549

Bardsley O P, Baker J L and Vincent C 2024 Decoupled magnetic control of spherical tokamak divertors via vacuum harmonic constraints *Plasma Phys. Control. Fusion* **66** 055006

Berkery J W, Sabbagh S A, Kogan L, Ryan D, Bialek J M, Jiang Y, Battaglia D J, Gibson S and Ham C 2021 Kinetic equilibrium reconstructions of plasmas in the MAST database and preparation for reconstruction of the first plasmas in MAST upgrade *Plasma Phys. Control. Fusion* **63** 055014

Boulila M, Hossain M, Mai C and Wright D 2021 *Tension spline* https://catxmai.github.io/pdfs/Math212_ProjectReport.pdf

Carpanese F 2021 Development of free-boundary equilibrium and transport solvers for simulation and real-time interpretation of tokamak experiments *PhD Thesis* EPFL https://infoscience.epfl.ch/handle/20.500.14299/175817

Cerfon A J and Freidberg J P 2010 One size fits all analytic solutions to the GradShafranov equation *Phys. Plasmas* **17** 032502
Coleman M and McIntosh S 2020 The design and optimisation of tokamak poloidal field systems in the BLUEPRINT framework *Fusion Eng. Des.* **154** 111544
Costantini P, Kvasov B I and Manni C 1999 On discrete hyperbolic tension splines *Adv. Comput. Math.* **11** 331–54
Coutlis A, Bandyopadhyay I, Lister J B, Vyas P, Albanese R, Limebeer D J N, Villone F and Wainwright J P 1999 Measurement of the open loop plasma equilibrium response in TCV *Nucl. Fusion* **39**
Creely A J *et al* 2020 Overview of the SPARC tokamak *J. Plasma Phys.* **86** 865860502
Cunningham G 2013 High performance plasma vertical position control system for upgraded MAST *Fusion Eng. Des.* **88** 3238–47
de Boucaud N, Golfinopoulos T, Marinoni A, Golfinopoulos T and Marinoni A 2022 On the synergy between easier plasma operation and affordable coil-set requirements enabled by Negative Triangularity in the prospective ARC fusion reactor arXiv:2212.08218
Doyle S J *et al* 2021 Magnetic equilibrium design for the SMART tokamak *Fusion Eng. Des.* **171** 112706
Dudson B 2024 FreeGS: Free-boundary Grad-Shafranov solver https://github.com/freegs-plasma/freegs
Faugeras B 2020 An overview of the numerical methods for tokamak plasma equilibrium computation implemented in the NICE code *Fusion Eng. Des.* **160** 112020
Grad H and Rubin H 1958 Hydromagnetic equilibria and force-free fields *In Proceedings of the Second United Nations International Conference on the Peaceful Uses of Atomic Energy: Theoretical and experimental aspects of controlled Nucl. Fusion* **volume 31** 400 United Nations
Ham C J and Farrell P E 2024 On multiple solutions of the gradshafranov equation *Nucl. Fusion* **64** 034001
Hansen C *et al* 2023 TokaMaker: An open-source time-dependent Grad-Shafranov tool for the design and modeling of axisymmetric fusion devices arXiv:2311.07719
Hudoba A, Cunningham G and Bakes S 2023 Magnetic equilibrium optimisation and divertor integration in spherical tokamak reactors *Fusion Eng. Des.* **191** 113704
Jardin S 2010 *Computational Methods in Plasma Physics* 1st edn (CRC Press) p 372
Jeon Y M 2015 Development of a free-boundary tokamak equilibrium solver for advanced study of tokamak equilibria *J. Korean Phys. Soc.* **67** 843–53
Kelley C T 1995 Iterative Methods for Linear and Nonlinear Equations *SIAM*
Knoll D A and Keyes D E 2004 Jacobian-free Newton-Krylov methods: a survey of approaches and applications *J. Comput. Phys.* **193** 357–97
Kogan L, Ryan D, Berkery J, Sabbagh S, Blalek J M, Appel L, Scannell R, Farley T, Thornton A and Cunningham G 2021 *EFIT++ reconstructions for MAST-U. MUDDMC*
Kripner L *et al* 2018 Equilibrium design for the COMPASS-U tokamak *In WDS'18 Proceedings of Contributed Papers Physics* pp 99–104
Lao L L *et al* 2022 Application of machine learning and artificial intelligence to extend EFIT equilibrium reconstruction *Plasma Phys. Control. Fusion* **64** 074001
Lao L L, John H S, Stambaugh R D, Kellman A G and Pfeiffer W 1985 Reconstruction of current profile parameters and plasma shapes in tokamaks *Nucl. Fusion* **25** 1611
Lao L L, John H E S, Peng Q, Ferron J R, Strait E J, Taylor T S, Meyer W H, Zhang C and You K I 2005 MHD equilibrium reconstruction in the DIII-D tokamak *Fusion Sci. Technol.* **48** 968–77
Lee C Y *et al* 2021 Development of integrated suite of codes and its validation on KSTAR *Nucl. Fusion* **61** 096020
Lee B, Pomphrey N and Lao L L 1999 Physics design of poloidal field, toroidal field, and external magnetic diagnostics in KSTAR *Fusion Technol.* **36** 278–88
Mancini A *et al* 2023 Predictive simulations for plasma scenarios in the SMART tokamak *Fusion Eng. Des.* **192** 113833
MANTA Team 2023 Modular adjustable negative-triangularity ARC (MANTA): A fusion pilot plant *USBPO Webinar*
Maquet V, Ragona R, van Eester D, Hillairet J and Durodie F 2023 ANSYS HFSS as a new numerical tool to study wave propagation inside anisotropic magnetized plasmas in the ion cylotron range of frequencies 309.14015.
MAST Upgrade Team *et al* 2022 First MAST-U equilibrium reconstructions using the EFIT++ code *48th EPS Conference on Plasma Physics (Maastricht, Netherlands, 27 June - 1 July 2022)* https://epsplasma2022.eu/
McArdle G J and Taylor D 2008 Adaptation of the MAST passive current simulation model for real-time plasma control *Fusion Eng. Des.* **83** 188–92
Morris W *et al* 2014 MAST accomplishments and upgrade for fusion next-steps *IEEE Trans. Plasma Sci.* **42** 402–14
Morris J, Muldrew S, Kovari M, Kahn S and Pearce A 2021 Preparing the systems code process for EU-DEMO conceptual design *28th IAEA Fusion Energy Conference, Nice, France (Virtual)* 10–5
Morris W, Harrison J R, Kirk A, Lipschultz B, Militello F, Moulton D and Walkden N R 2018 MAST upgrade divertor facility: a test bed for novel divertor solutions *IEEE Trans. Plasma Sci.* **46** 1217–26
Pentland K *et al* 2024 Validation data from equilibrium solvers in FreeGSNKE, Fiesta, and EFIT++ on MAST-U *UKAEA Open Data Register*
Ryan D A, Martin R, Appel L, Ayed N B, Kogan L and Kirk A 2023 and MAST Upgrade Team. Initial progress of the magnetic diagnostics of the MAST-U tokamak *Rev. Sci. Instrum.* **94** 073501
Sabbagh S A 2001 Equilibrium properties of spherical torus plasmas in NSTX *Nucl. Fusion* **41** 1601
Sangaroon S *et al* 2023 Feasibility study of neutral beam injection in Thailand Tokamak-1 *Fusion Eng. Des.* **188** 113419
Shafranov V D 1958 On magnetohydrodynamical equilibrium configurations *Soviet Journal of Experimental and Theoretical Physics* **6** 545
Stewart I G, Granetz R S, Myers C E, Paz-Soldan C, Sweeney R, Hansen C J, Garnier D T, Battaglia D J, Creely A J and Reinke M L 2023 Optimization of the equilibrium magnetic sensor set for the SPARC tokamak *Nucl. Fusion* **63** 126014
Vondracek P *et al* 2021 Preliminary design of the COMPASS upgrade tokamak *Fusion Eng. Des.* **169** 112490
Windridge M J, Cunningham G, Hender T C, Khayrutdinov R and Lukash V 2011 Non-linear instability at large vertical displacements in the MAST tokamak *Plasma Phys. Control. Fusion* **53** 035018
Yoshida H, Ninomiya H, Azumi M and Seki S 1986 Numerical method for tokamak equilibrium with outside limiter *J. Comput. Phys.* **63** 477–85