Reconstruction of plasma equilibrium and separatrix using convolutional physics-informed neural operator

Matteo Bonotto[a,b], Domenico Abate[a], Leonardo Pigatto[a]

[a] *Consorzio RFX, CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA, Corso Stati Uniti 4, Padova, 35127, State Two, Italy*
[b] *INFN-LNL, Viale dell'Università 2, Legnaro, 35020, State Two, Italy*

Abstract

In this paper, we present PlaNet (*PLA*sma equilibrium reconstruction *NET*work), a Deep-Learning-based tool for performing plasma equilibrium and separatrix reconstruction considering both magnetic and non-magnetic measurements. PlaNet consists of three Neural Networks (NNs) running at three stages: (i) *Equilibrium Net*, the physics-informed neural operator accounting for the accurate reconstruction of the poloidal flux map; (ii) *XPlim Net*, which classifies if the equilibrium is diverted or limiter and computes the poloidal flux at the separatrix; and (iii) *Separatrix Net*, which gives the (r, z) coordinates of the separatrix starting from the outputs of the previous stages. The plasma equilibrium solver at the basis of this tool exploits a convolutional physics-informed neural operator, which solves the plasma equilibrium problem satisfying the Grad-Shafranov Equation by including a physics-informed term in the loss function. To constrain the equilibrium and achieve a better reconstruction, especially in the core region of the plasma, PlaNet uses also non-magnetic measures, such as the pressure profile or the poloidal current profile. PlaNet has been trained and validated on a dataset of 81986 synthetic equilibria of an ITER-like device, showing to be an effective tool for fast and accurate equilibrium ad separatrix reconstruction.

Keywords: Plasma Equilibrium Reconstruction, Grad-Shafranov, Artificial Intelligence, Physics-Informed Machine Learning, Physics-Informed Neural Operator, Convolutional Neural Networks

1. Introduction

Real-time reconstruction of plasma equilibrium from diagnostics is of paramount importance for reaching the expected performance during machine operations and for ensuring the required safety of fusion devices during these operations. Plasma equilibrium reconstruction requires solving an inverse problem based on the Grad-Shafranov Equation (GSE) [1, 2], a nonlinear Partial Differential Equation (PDE) describing the equilibrium problem of a magnetized plasma. This task requires the iterative procedure of adjusting the equilibrium over the iterations to match the experimental measures.

Due to its computational complexity, plasma equilibrium reconstruction is a difficult task to perform in real-time. Many equilibrium reconstruction codes have been developed in the last two decades [3, 4, 5, 6, 7], all of which are iterative codes that require many iterations to converge to a good equilibrium. Some of these tools have been made real-time compliant by limiting the number of iterations performed (usually, a maximum of 1 to 4 iterations are performed [5, 6, 7]), with the inevitable drawback of reducing the reconstruction accuracy. Limiting the number of iterations is mandatory even if the codes are optimized to run on modern Graphics Processing Units (1 or 2 iterations in [5]).

In addition to this, most of the codes used nowadays for equilibrium reconstruction use only magnetic measures. It is well known that magnetic diagnostics are not enough to reconstruct the magnetic equilibrium properly, especially in the core region of the plasma [3, 8]. To overcome this limitation, *kinetic reconstruction* codes have been developed [9, 10, 11, 12], but, as already said earlier, the use of these tools in a real-time environment is not possible, because of their computational complexity.

In this context, Neural Networks (NNs) can be very appealing, having been widely used for solving nonlinear PDEs in real-time. The idea of using a Multi-Layer Perceptron (MLP), a NN composed of fully connected layers, as a *universal approximator* for solving a Partial Differential Equation dates back to the 90s' [13]. In the same period, this approach has been also used for solving the Grad-Shafranov Equation (GSE) [14]. However, in recent years, important advancements have been made in the use of NNs for solving linear and nonlinear PDEs [15, 16, 17], introducing also the nomenclature *Physics-Informed* Neural Network (PINN).

However, a PINN is trained for a specific set of boundary and initial conditions, as well as for a specific source term, and required to be re-trained if any of the previous terms changes, making them not particularly effective for real-world operations [18, 19]. To overcome this issue, in 2019, Lu *et. al.* introduced the concept of *neural operator* (DeepONet [20, 21]), a NN architecture able to learn the mapping between two functional spaces (the so-called *manifold learning*). Neural operators can learn the PDE operators for arbitrary source functions and boundary conditions, opening the possibility of using them as surrogate models of complex systems taking into account the governing PDEs.

Concerning the reconstruction of plasma equilibrium from magnetic measures via NNs, the first example dates back to 1994, coming from the seminal work of Coccorese *et al.*. However, in recent years, a huge effort has been spent on this topic [22, 23, 24], especially thanks to the advent of modern deep learning frameworks [25, 26, 27, 28]. However, among the aforementioned works, only S. Joung *et al.* [22] developed an NN which can be considered similar to a physics-informed neural operator[1], while all the others do not impose any physics-based

[1]A proper neural operator requires two input stages, the *branch network* and the *trunk network*, being multiplied together before the output stage. In the architecture used in [22], there is only a single input stage.

constraint on the equilibrium solver [23, 24]. Moreover, all the aforementioned works are based on MLPs. The use of Convolutional Neural Networks (CNN) to solve PDEs, which has been introduced more recently [29, 30], has a clear advantage regarding the computational point of view, especially with the advent of modern Graphics Processing Units (GPUs), and even the more recent Tensor Processing Units (TPUs). On top of that, the common procedure followed by the aforementioned works is to take into account only magnetic measures, which can lead to an inaccurate reconstruction of the equilibrium.

The benefit of using a physics-informed neural operator for solving the inverse plasma equilibrium problem is two-fold. First of all, these NNs are trained on many reference equilibria computed $offline$ (i.e., not during machine operations, and thus without time constraints proper of real-time environments) using physics-based codes, therefore the accuracy can be improved by refining the computational domain without affecting the execution time of the NN in real-time. Secondly, the use of recent GPUs and TPUs with tensor cores and automatic mixed-precision training can speed up the computations by orders of magnitude if compared to conventional CPUs, opening the possibility of computing equilibrium reconstructions as accurately as offline codes, but in less than 0.1ms. This speed-up cannot be acieved *in toto* by conventional physics-based equilibrium reconstruction codes, as they are *iterative* codes, and only some steps can be fully parallelized (e.g., matrix-matrix products, some for loops, etc.), but their structure is inherently sequential, being each iteration dependent on the solution of the previous one. This is the main bottleneck of physics-based solvers, which cannot be solved simply by using faster hardware. NNs, on the other hand, do not involve iterative steps to compute the plasma equilibrium and are optimized to benefit as much as possible from GPUs/TPUs architectures.

In this paper, we present PlaNet (*PLA*sma equilibrium reconstruction *NET*work), a deep-learning-based tool for plasma equilibrium and separatrix reconstruction considering both magnetic and non-magnetic measurements. PlaNet consists of three NNs running at three stages: (i) *Equilibrium Net*, the physics-informed neural operator accounting for the accurate reconstruction of the poloidal flux map; (ii) *XPlim Net*, which classifies if the equilibrium is diverted or limiter and computes the poloidal flux at the separatrix; and (iii) *Separatrix Net*, which gives the (r, z) coordinates of the separatrix. The plasma equilibrium solver at the basis of this tool exploits a convolutional physics-informed neural operator, which solves the plasma equilibrium problem satisfying the GSE by including a physics-informed term in the loss function. To constrain the equilibrium and achieve a better reconstruction, especially in the core region of the plasma, PlaNet uses also non-magnetic measures, such as the pressure profile or the poloidal current profile, being both obtainable in real-time [10, 31].

The paper is organized as follows. Section 2 describes the plasma equilibrium problem in axial symmetry, in sec. 3 the relations needed to compute the Grad-Shafranov operator via convolution are described, sec. 4 gives a general formulation of the problem of reconstructing plasma equilibrium from magnetic and non-magnetic measures, sec. 5 introduces the concept of the neural operator, sec. 6 describes PlaNet architecture, sec. 7 presents the validation of the model,

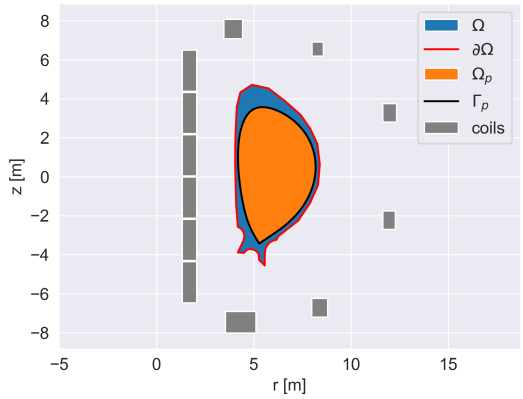


Figure 1: Domains used in the definition of plasma equilibrium problem. A cylindrical coordinate system (r, ϕ, z) is considered. Ω is the vacuum region inside the first wall and allowed to the plasma, $\partial\Omega$ is the computational boundary, Ω_p is the plasma cross-section and Γ_p is the plasma separatrix. An ITER-like device is considered.

and, finally, in sec. 8 conclusions and further developments are reported.

2. Mathematical formulation of plasma equilibrium problem

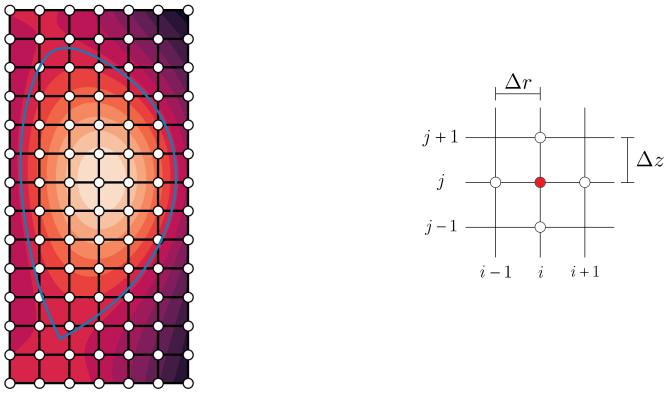
In a cylindrical coordinate system (r, ϕ, z), let us consider the domain Ω, which is the vacuum region inside the first wall and allowed to the plasma (blue + orange in fig. 1, where an ITER-like device is shown). The computational boundary of Ω is labeled as $\partial\Omega$ (red in fig. 1). Inside Ω, the region which is occupied by the plasma, i.e. the plasma cross-section, is labeled Ω_p (orange in fig. 1). In addition to this, we define also the plasma separatrix Γ_p, which is the *plasma-vacuum interface* (black in fig. 1).

The Free-Boundary Plasma Equilibrium Problem (FB-PEP) in axisymmetric geometry is described by the Grad-Shafranov Equation (GSE), together with suitable boundary conditions (BCs):

$$-\nabla \cdot \left(\frac{1}{r} \nabla \psi(\vec{\boldsymbol{r}}) \right) = \begin{cases} \mu_0 j_\phi(\vec{\boldsymbol{r}}, \psi) & \text{if } \boldsymbol{r} \in \Omega_p \\ 0 & \text{elsewhere} \end{cases} \tag{1}$$

$$\psi(\vec{\boldsymbol{r}})\big|_{\partial\Omega} = \hat{\psi}_p(\vec{\boldsymbol{r}}, \psi) + \hat{\psi}_{ext}(\vec{\boldsymbol{r}}) \tag{2}$$

where μ_0 is the magnetic permeability of vacuum, and $\psi(\vec{\boldsymbol{r}})$ is the poloidal magnetic flux *per radian* [32] (i.e. providing the poloidal magnetic field as $\frac{\nabla\psi}{r} \times \vec{\boldsymbol{e}}_\phi$, where $\vec{\boldsymbol{e}}_\phi$ unit vector in ϕ direction), and $\hat{\psi}_p(\vec{\boldsymbol{r}}, \psi)$ and $\hat{\psi}_{ext}(\vec{\boldsymbol{r}})$ are the boundary conditions due to the plasma and to the external environment (i.e., active colitis in the magnetostatic regime, gray in fig. 1), respectively.



(a) (b)

Figure 2: Fig. 2a exemplification of a structured rectangular grid covering the computational domain Ω; fig. 2b stencil used for the FD scheme.

The current density in the Right Hand Side (RHS) of eq. (1) is described as:

$$j_\phi(\vec{\boldsymbol{r}}, \psi) = r\frac{dp(\bar{\psi})}{d\bar{\psi}} + \frac{f(\bar{\psi})}{\mu_0 r}\frac{df(\bar{\psi})}{d\bar{\psi}} \tag{3}$$

that is, the current density is described in terms of the profiles of the pressure $p(\bar{\psi})$ and the poloidal current function $f(\bar{\psi})$ [32], being $\bar{\psi} = (\psi - \psi_a)/(\psi_b - \psi_a)$ the normalized poloidal flux, where ψ_a and ψ_b are respectively the value of the flux at the magnetic axis and the plasma separatrix Γ_p.

The Left-Hand Side (LFS) of eq. (1) and the RHS of eq. (3) give the Grad-Shafranov Equation:

$$-\nabla \cdot \left(\frac{1}{r}\nabla\psi(\vec{\boldsymbol{r}})\right) = r\frac{dp(\bar{\psi})}{d\bar{\psi}} + \frac{f(\bar{\psi})}{\mu_0 r}\frac{df(\bar{\psi})}{d\bar{\psi}} \tag{4}$$

which is the equilibrium equation, in ideal magnetohydrodynamics (MHD), of a two-dimensional plasma.

3. Computing Grad-Shafranov operator via convolution

Among the several numerical techniques which can be used to solve equation GSE numerically [33, 34, 35, 36, 37], Finite Difference (FD) scheme is certainly one of the most extensively used in literature [38, 39, 6, 36, 40]. To solve GSE via FD, at first we define a structured rectangular grid covering the computational domain Ω, as sketched in fig. 2a. The GSE (4) is written as:

$$\frac{\partial^2\psi(\vec{\boldsymbol{r}})}{\partial r^2} + \frac{\partial^2\psi(\vec{\boldsymbol{r}})}{\partial z^2} - \frac{1}{r}\frac{\partial\psi(\vec{\boldsymbol{r}})}{\partial r} = -\mu_0 r j_\phi(\vec{\boldsymbol{r}}, \psi) \tag{5}$$

where the Left-Hand Side (LHS) is commonly named as the *Grad-Shafranov Operator* (GSO) $\Delta^*\psi$. With respect to the notation shown in fig. 2b, the partial derivatives in the point $\vec{\boldsymbol{r}}_{ij}$ are written as:

$$\left(\frac{\partial^2\psi}{\partial r^2}\right)\bigg|_{i,j} \approx \frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{\Delta r^2} \tag{6}$$

$$\left(\frac{\partial^2\psi}{\partial z^2}\right)\bigg|_{i,j} \approx \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{\Delta z^2} \tag{7}$$

$$\left(\frac{\partial\psi}{\partial r}\right)\bigg|_{i,j} \approx \frac{\psi_{i+1,j} - \psi_{i-1,j}}{2\Delta r} \tag{8}$$

Here, the centered finite differences have been used, which have quadratic convergence with the grid size. By following these relations, the RHS of eq. (5) can be written as:

$$\Delta^*\psi_i = \frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{\Delta r^2} + \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{\Delta z^2} - \frac{\psi_{i+1,j} - \psi_{i-1,j}}{2r_i\Delta r} \tag{9}$$

One of the most interesting features of FD is that the discrete counterpart of the GS operator can be performed by applying two convolutions to the function $\psi(r, z)$ with suitable 3×3 filters. The only requirement is the computational domain Ω being discretized using a regular structured grid, which is automatically satisfied when the standard isotropic FD approach is used. It follows, after some algebraic manipulations of eq. (9), that these filters are:

$$\mathcal{D}_{\nabla^2} = \begin{bmatrix} 0 & \frac{\Delta r^2}{\alpha} & 0 \\ \frac{\Delta z^2}{\alpha} & 1 & \frac{\Delta z^2}{\alpha} \\ 0 & \frac{\Delta r^2}{\alpha} & 0 \end{bmatrix} \qquad \mathcal{D}_{\partial_r} = \frac{\Delta r^2\Delta z^2}{2\alpha\Delta r} \begin{bmatrix} 0 & 0 & 0 \\ +1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} \tag{10}$$

where $\alpha = -2(\Delta r^2 + \Delta z^2)$. Equation (9) can be therefore written as:

$$\mathcal{D}_{\nabla^2}[\psi] - \frac{1}{r}\mathcal{D}_{\partial_r}[\psi] = \beta\Delta^*\psi \tag{11}$$

where $\beta = \frac{\Delta r^2\Delta z^2}{\alpha}$, and $\mathcal{D}_{\#}[\psi]$ is the convolution operation with the filter $\mathcal{D}_{\#}$ applied to ψ. Thus, the GS operator $\Delta^*\psi_i$ can be computed as:

$$\Delta^*\hat{\psi} = \frac{1}{\beta}\left(\mathcal{D}_{\nabla^2}[\psi^p] - \frac{1}{r}\mathcal{D}_{\partial_r}[\psi^p]\right) \tag{12}$$

This fact is of particular interest because convolution is the core operation of CNNs, and eq. 12 can be implemented straightforwardly in this context.

4. Reconstruction of plasma equilibrium from measurements

From a mathematical point of view, the problem of plasma equilibrium from measurements from measures is formalized by constraining (1) - (2) to satisfy

some *external measurements* $\boldsymbol{m}$, which can be either magnetic (e,g, pickup-coils, flux loops, saddle coils) or non-magnetic (e.g., Thomson scattering, x-ray imaging crystal spectrometer (XCS), etc.).

The constraint on the magnetic measures can be written by imposing that the current density $j_\phi(\vec{\boldsymbol{r}}, \psi)$ satisfies the following relation on each magnetic measure:

$$m(\vec{\boldsymbol{r}}', \psi) = \int_{\Gamma_m} \left(\int_{\Omega_p} \boldsymbol{G}(\vec{\boldsymbol{r}}, \vec{\boldsymbol{r}}') j_\phi(\vec{\boldsymbol{r}}, \psi) d\Omega \right) \cdot \vec{\boldsymbol{n}} d\Gamma_m \tag{13}$$

where $\vec{\boldsymbol{r}}'$ is the geometrical position of the sensor, $\boldsymbol{G}(\vec{\boldsymbol{r}}, \vec{\boldsymbol{r}}')$ the Biot-Savart kernel linking the plasma current to the magnetic sensor (usually called also *Green* matrix in its discrete version), and the integral $\int_{\Gamma_m}$ is a surface integral evaluated on the surface linked to the magnetic probe, with $\vec{\boldsymbol{n}}$ its normal unit vector.

Concerning the non-magnetic measures, we consider two different cases, both feasible for real-time implementations, that are (i) constraining the pressure profile p and (ii) constraining the toroidal current density profile j_ϕ. A constraint on the pressure profile can be obtained by imposing that [10]:

$$p = n_e T_e + (n_i + n_z) T_i + P_f \tag{14}$$

where T_e and n_e profiles are from Thomson scattering measurements [41], T_i profiles are from the x-ray imaging crystal spectrometer (XCS) [42], n_i and n_z are obtained from the quasi-neutrality condition and the Z_{eff} measurement, and P_f is the pressure contribution given by the fast particles, which can be obtained via fast ion D_α imaging (FIDA) [43].

On the other hand, the current density j_ϕ can be constrained via the flux diffusion equation [31]:

$$\sigma_{||} \frac{\partial \psi}{\partial t} = \frac{R^0 J^2}{\mu_0 \rho} \frac{\partial}{\partial \rho} \left(\frac{G_2}{J} \frac{\partial \psi}{\partial \rho} \right) - \frac{V'}{2\pi\rho} (J_{BS} + J_{CD}) \tag{15}$$

where:

$$J = \frac{f}{R_0 B_0} \tag{16}$$

$$V' = \frac{\partial V}{\partial \rho} \tag{17}$$

$$G_2 = \frac{1}{4\pi^2} V' \langle (\nabla \rho)^2 / r^2 \rangle \tag{18}$$

ρ is an arbitrary flux surface label, $\sigma_{||}$ is the parallel conductivity, J_{BS} and J_{CD} are the current densities due to bootstrap and current drive, R_0 is the tokamak major radius, B_0 is the toroidal equilibrium field, and μ_0 is the magnetic permeability of vacuum.

In general, all these constraints can be summarized in a system of nonlinear equations for each of the diagnostic m_i, $i = 1, ..., N_m + N_n$ (here N_m, N_n are the number of magnetic and non-magnetic diagnostics, respectively):

$$\boldsymbol{\Lambda}(\boldsymbol{m}, j_\phi, p, \psi) = 0 \tag{19}$$

depending on which profile is constrained, (19) will depend either on j_ϕ or p. The plasma equilibrium reconstruction problem can be formulated as:

$$-\nabla \cdot \left(\frac{1}{r}\nabla\psi(\vec{\boldsymbol{r}})\right) = \begin{cases} \mu_0 j_\phi(\vec{\boldsymbol{r}}, \psi) & \text{if } \boldsymbol{r} \in \Omega_p \\ 0 & \text{elsewhere} \end{cases} \tag{20}$$

$$\psi(\vec{\boldsymbol{r}})\big|_{\partial\Omega} = \hat{\psi}_p(\vec{\boldsymbol{r}}, \psi) + \hat{\psi}_{ext}(\vec{\boldsymbol{r}}) \tag{21}$$

$$\boldsymbol{\Lambda}(\boldsymbol{m}, j_\phi, p, \psi) = 0 \tag{22}$$

The problem (20) - (22) is a nonlinear problem as well, and has to be solved iteratively. From an initial guess $\boldsymbol{\psi_0}$ chosen as a starting equilibrium, the solution is then iterated using an iterative scheme, such as Picard or Newton-Raphson, until convergence is reached (i.e., the solution $\boldsymbol{\psi}$ does not change over iterations).

5. Neural Operators

The concept of Neural Operator was introduced by Lu et. al. in 2019 with DeepONet [20, 21], with the idea of learning the mapping between two functional spaces (the so-called *manifold learning*). A schematic view of this architecture and its inputs and outputs is shown in figure 3.

Neural Operators are based on the *Universal Approximation Theorem for Operator* [44], stating that any nonlinear continuous operator $G : \mathbb{R}^m \mapsto \mathbb{R}^n$ mapping a function $u(\boldsymbol{x})$, known at the coordinates $\boldsymbol{x} \in \mathbb{R}^m$, into $\boldsymbol{y} \in \mathbb{R}^n$ (called *query points*, usually referred to the coordinates in the physical space where we want to know the function), can be approximated via the following relation:

$$G(u)(y) \approx \sum_{k=1}^{K} b_k(u(\boldsymbol{x}))t_k(\boldsymbol{y}) \tag{23}$$

where $b(u(\boldsymbol{x}))$ and $t(u(\boldsymbol{x}))$ are the so-called *branch network* and the *trunk network*, respectively (see fig. 3), and K the number of units of such networks.

This means that we can build up a map between the points $\boldsymbol{x}$ where the input function is known, and the desired points where we need the output $\boldsymbol{y}$. By decoupling the inputs $\boldsymbol{x}$ and $\boldsymbol{y}$, we can train the network for different values of query points $\boldsymbol{y}$, allowing the network to lean the mapping *for arbitrary* values of $\boldsymbol{y}$.

Neural Operators can be made *physics-informed* (PINO: Physics-Informed Neural Operator [45]) as common PINNs by adding a physics-based term to the loss function. This way, by choosing a suitable set of source terms and/or BCs for training, PINOs can be used as physics-aware surrogate models to reconstruct scalar and vector fields from a given set of measures, yet satisfying the underlying PDEs, both for linear and nonlinear problems.

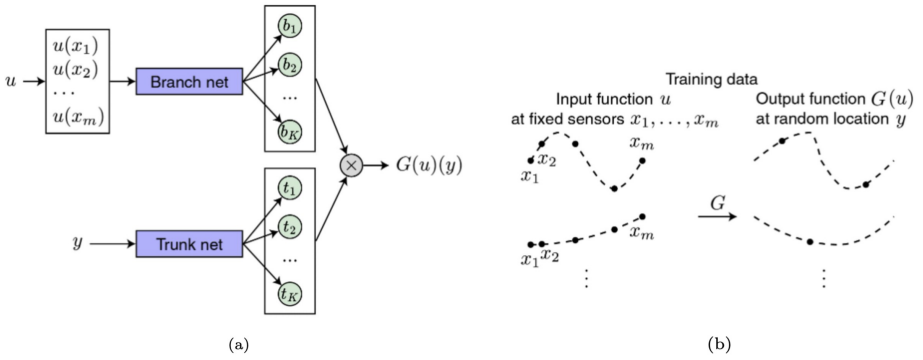


Figure 3: DeeoONet architecture (a) and training data (b).

6. PlaNet: reconstruction of plasma equilibrium and separatrix via a convolutional physics-informed neural operator

6.1. Network architecture

Plasma equilibrium reconstruction is a computationally heavy task. Depending on the numerical scheme that is used, it can require, to reach convergence, from $5 \div 10$ iterations for Newton-like-based codes, and up to ≈ 50 iterations for Picard-based ones. Thus, during real-time operations, these codes cannot run until convergence at each time instant, and a limited amount of iterations are allowed (usually, from 1 to 4 [5, 6, 7]), even if the codes are optimized to run on modern GPUs. In this context, physics-informed neural operators are very appealing, requiring limited computational time to reconstruct an equilibrium while keeping satisfied the physical constraints (i.e., the GSE in this context).

PlaNet (*PLA*sma equilibrium reconstruction *NET*work) has been developed to address this task. It is composed of three NNs running at three stages (see fig. 4):

1. *Equilibrium Net*, the convolutional physics-informed neural operator which computes the poloidal flux $\psi(r, z)$ in the computational domain Ω starting from the coils current $\boldsymbol{i}_c$ and measurements $\boldsymbol{m} = [\boldsymbol{m}_m, \boldsymbol{m}_n]$, being $\boldsymbol{m}_m$ and $\boldsymbol{m}_n$ the magnetic and non-magnetic measures, respectively. Further details will be provided in par. 6.2.
2. *XPlim Net* solves the combined task of classifying the equilibrium type (i.e., diverted or limiter equilibrium) together with recovering the flux ψ_b at the plasma separatrix;
3. *Separatrix Net* reconstructs the coordinate of the separatrix starting from the $\psi(r, z)$ map given by the *Euqilibrium Net* and from the outputs of XPlim Net;

These three networks operate in three distinct steps, and each one runs on top of the outputs of the previous one.

6.2. Euqilibrium Net *: reconstruction of poloidal flux using convolutional physics-informed neural operator*

Equilibrium Net reconstructs the poloidal flux map ψ starting from the currents in the active coils, the magnetic measures and the non-magnetic measures. The input $\boldsymbol{u}$ and the output $\boldsymbol{y}$ of the network thus are:

$$\boldsymbol{u} = [\boldsymbol{i}_c, \boldsymbol{m}_m, \boldsymbol{m}_n, \boldsymbol{r}_q, \boldsymbol{z}_q] \tag{24}$$

$$\boldsymbol{y} = \boldsymbol{\psi}^p(\boldsymbol{r}_q, \boldsymbol{z}_q) \tag{25}$$

where the superscript p stands for *predicted*, $\boldsymbol{i}_c$ are the currents in the active coils, $\boldsymbol{m}_m$ are the magnetic measures, and $\boldsymbol{m}_n$ the non-magnetic measures, $\boldsymbol{r}_q$, $\boldsymbol{z}_q$ are the *query points* forming the structured grid where the output $\boldsymbol{\psi}^p(\boldsymbol{r}_q, \boldsymbol{z}_q)$ is computed. The input $\boldsymbol{u}_{branch} = [\boldsymbol{i}_c, \boldsymbol{m}_m, \boldsymbol{m}_n]$ has a dimension $N_b \times (N_c + N_m + N_n)$, where N_b is the batch size, the input $\boldsymbol{u}_{trunk} = [\boldsymbol{r}_q, \boldsymbol{z}_q]$ has a dimension $N_b \times 2 \times N_r \times N_z$, and the output $\boldsymbol{y}$ has dimension $N_b \times N_r \times N_z$. In addition to $\boldsymbol{y}$, we need to provide also $\Delta^*\boldsymbol{\psi}$ on $\boldsymbol{r}_q$, $\boldsymbol{z}_q$ to train the NN to satisfy the GS operator, and so the training dataset is $\mathbb{D}_t = \{\boldsymbol{u}, \boldsymbol{\psi}, \Delta^*\boldsymbol{\psi}\}$.

The architecture for the physics-informed neural operator used in this application is shown in fig. 5, and it has:

- a *trunk network* for encoding (i.e., obtaining a lower-dimensional representation of the input) the geometrical information of the query points. We used two separate convolution-based channels for this task, to treat the r and z coordinates independently. The encodings are concatenated together and further compressed using fully connected layers.
- a *branch network*, used to encode information on the input functions (i.e., coils' currents, magnetic measures and j_ϕ or p profiles), and based on fully connected layers.
- an upsampling + convolution-based decoder to process the information from the branch and trunk networks and to reconstruct the poloidal flux map on the query points $(\boldsymbol{r}_q, \boldsymbol{z}_q)$.

To enhance the capabilities of the network we used a Swish activation function for the convolutional layers and the fully-connected layers, defined as:

$$f = x\sigma(\beta x) = \frac{xe^{\beta x}}{1 + e^{\beta x}} \tag{26}$$

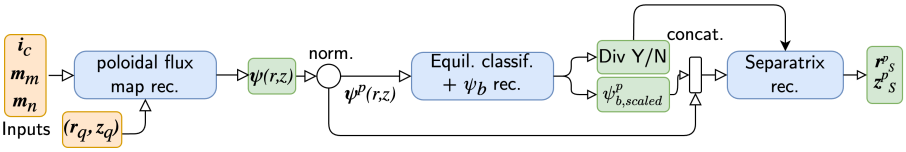


Figure 4: PlaNet architecture.

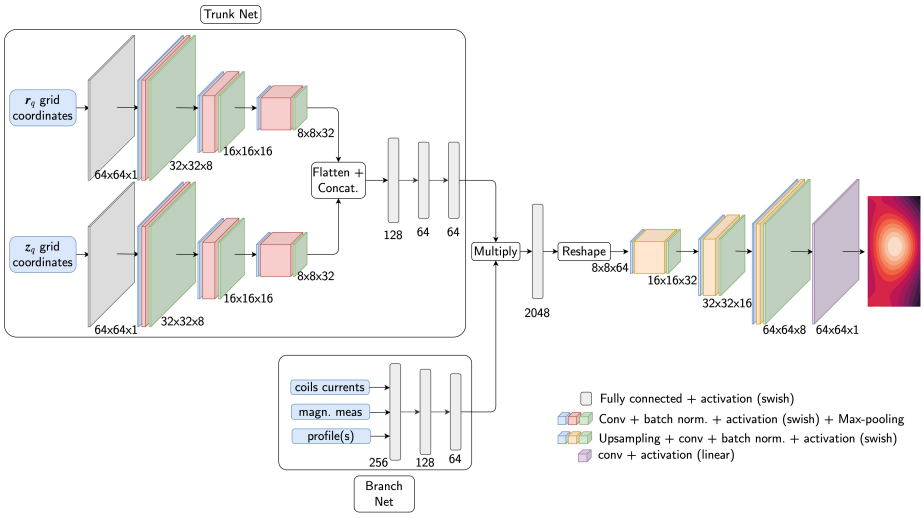


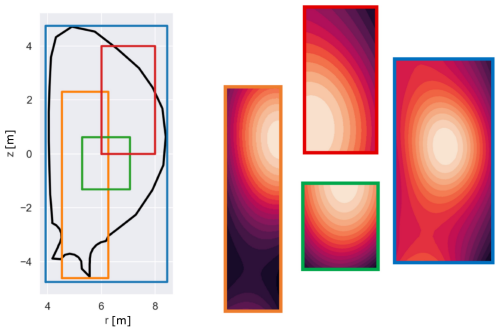
Figure 5: Architecture of the *Euqilibrium Net* .

where $\sigma(.)$ is the *sigmoid* activation function, and β is a learnable parameter.

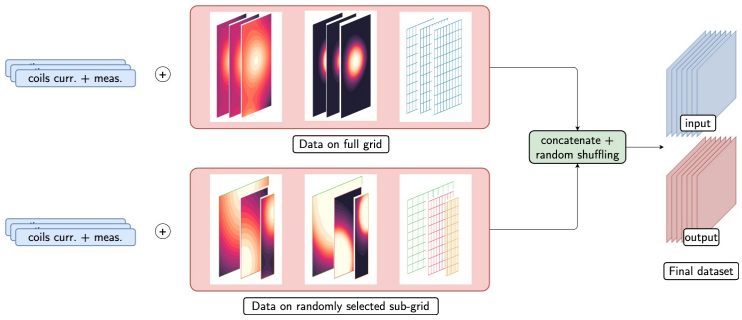
As pointed out in sec. 5, a neural operator uses the *trunk network* to encode information about the locations where the output function $\psi(r, z)$ has to be evaluated, and the *branch network* to encode information related to the inputs proper of each equilibrium (the active coils' currents and the measures). The advantage of using a neural operator is that, if a sufficiently large dataset of different query points (r_q, z_r) and flux maps $\psi(r_q, z_q)$ is provided, the network can learn the value of ψ at *arbitary* location. This means that, in principle, the reconstruction of ψ is not limited to a fixed grid covering the vacuum chamber, but arbitrary grids can be used, as far as such grids match the $N_r \times N_z$ dimension. This can be seen in fig. 6a, where the outline of both the initial grid (blue) and other randomly defined smaller grids (orange, green, blue) are shown, together with $\boldsymbol{\psi}^p(\boldsymbol{r}_q, \boldsymbol{z}_q)$ on these grids. In this case, the inputs $[\boldsymbol{i}_c, \boldsymbol{m}_m, \boldsymbol{m}_n]$ are the same, only $[\boldsymbol{r}_q, \boldsymbol{z}_q]$ are varied. This is particularly appealing because the *same* network can be used for reconstructing ψ on the entire plasma domain, but also on a smaller region, for example, around the X-point or the magnetic axis. This allows to arbitrarily "zoom-in" without the need to retrain the network on smaller grids.

To achieve this capability, we need to train the network both on equilibria defined on the whole grid, as well as on randomly generated smaller grids. The training dataset $\mathbb{D}_t$ is assembled as shown in fig. 6b, and described as follows. We consider two different datasets,

- $\mathbb{D}'_t$ containing $\boldsymbol{\psi}$ equilibria on the entire domain (*Data on full grid* in fig. 6b). The outline of the grid covering the entire computational domain is shown in blue in fig. 6a, and the related equilibria are also shown in the same figure with a blue outline.



(a) Outline of both the initial grid (blue) covering the entire computational domain and other randomly defined smaller grids (orange, green, blue), together with $\boldsymbol{\psi}^p(\boldsymbol{r}_q, \boldsymbol{z}_q)$) on these grids.



(b) Procedure for building the training dataset $\mathbb{D}_t$. Inputs are shown in blue, and outputs in red.

Figure 6

- $\mathbb{D}_t''$ containing portions of equilibria on randomly selected sub-regions of the original grid (*Data on randomly selected sub-grids* in fig. 6b). For this purpose, a large number of rectangular grids smaller than the original one are randomly generated, as can be seen in fig. 6a where the outlines of some grids are shown in orange, green and red. Both $\boldsymbol{\psi}$ and $\Delta^*\boldsymbol{\psi}$ have to be computed on the sub-grids from the original, high-resolution data (see orange, green and red outlined flux maps in fig. 6a)

The datasets $\mathbb{D}_t'$ and $\mathbb{D}_t''$ are then concatenated together and shuffled, providing the input and output datasets.

The considered equilibria refer to an ITER-like device, whose geometry can be seen in fig. 1. The active currents are $N_c = 14$, while $N_m = 187$ as we considered the actual system of pick-up coils of the ITER device [46], and $N_n = 100$, as each profile is defined on a set of 100 points along the normalized coordinate ρ. The target variable is reconstructed on a $N_r \times N_z = 64 \times 64$ grid (called *sensors* in [20]). The input vectors are standardized to have zero mean and unit variance.

This step is necessary since we are combining input values whose magnitude differs by several orders of magnitude.

Before going on in the description, it is worth mentioning that there are two possible ways to reconstruct a quantity on a structure starting from a vector of scalar quantities acting as "measurements". It can be done using either *2D upsampling + 2D convolution*, as done in this work, or by using the so-called *transposed convolution* (also known as *deconvolution*), as done for example in [47]. However, the transposed convolution is well known to suffer from the so-called *check-box artifacts* [48], which can strongly affect the output and reduce the quality of the reconstruction, giving a noisy solution. For this reason, after investigating both approaches, for our network, we choose the former solution (i.e., 2D upsampling + 2D convolution).

The loss function $\mathcal{L}(\boldsymbol{\psi}, \boldsymbol{\psi}^p)$ between the true solution $\boldsymbol{\psi}$ and the predicted one $\boldsymbol{\psi}^p$ is the sum of two terms:

$$\mathcal{L}(\boldsymbol{\psi}, \boldsymbol{\psi}^p) = \mathcal{L}_\psi(\boldsymbol{\psi}, \boldsymbol{\psi}^p) + w\mathcal{L}_{\Delta^*}(\boldsymbol{\psi}, \boldsymbol{\psi}^p) \tag{27}$$

where $\mathcal{L}_\psi$ and $\mathcal{L}_{\Delta^*}$ evaluate the quality of the predicted poloidal flux and of the GS operator computed on such prediction, respectively, and w is a weighting factor to ensure that both the loss terms have roughly the same order of magnitude. The loss terms are defined as:

$$\mathcal{L}_\psi(\boldsymbol{\psi}, \boldsymbol{\psi}^p) = \frac{1}{M}\sum_{i=1}^{M}\left(\psi_i - \psi_i^p\right)^2 \tag{28}$$

$$\mathcal{L}_{\Delta^*}(\boldsymbol{\psi}, \boldsymbol{\psi}^p) = \frac{1}{M}\sum_{i=1}^{M}\left(\Delta^*\psi_i - \Delta^*\psi_i^p\right)^2 \tag{29}$$

where M is the total number of examples in the database. The GS operator $\Delta^*\boldsymbol{\psi}^p$ can be computed by using eq. (12) on the predictions $\boldsymbol{\psi}_i^p$.

The additional constraint (29), added to the standard loss function (28), has two advantages. At first, as already said, $\boldsymbol{\psi}^p$ has a physical meaning. In addition, the term (29) is a regularization term [49], similar to L_2 regularization. This helps the network to achieve better generalization (i.e., avoid overfitting), even for smaller training datasets.

6.3. XPlim Net: equilibrium classification and boundary flux reconstruction

As already mentioned, *XPlim Net* has to carry out, knowing $\boldsymbol{\psi}^p$, the following tasks:

1. classification of the equilibrium type (limiter or diverted);
2. reconstruction of the poloidal flux (per radian) at the separatrix ψ_b^p.

This additional information is necessary to perform the separatrix reconstruction, and the reasons will be explained in the next paragraph, where the *Separatrix Net* is described.

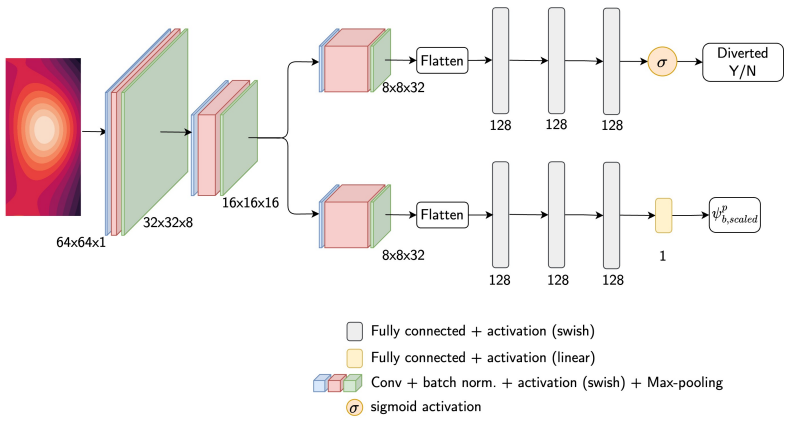


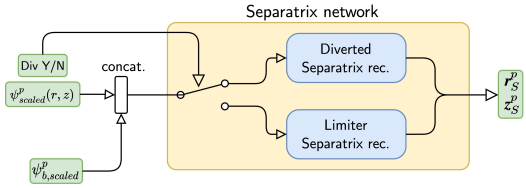
Figure 7: Architecture of the XPlim Net.

The architecture of *XPlim Net* is shown in fig. 7. The input of this network is the poloidal flux map $\boldsymbol{\psi}^p_{scaled}$, which is scaled in the range [0,1]. We can see from fig. fig. 7 that after some shared layers (2 blocks of 2D upsampling + 2D convolution + batch normalization), the network bifurcates in two branches for classification and regression. These branches share the same architecture (i.e., a 2D upsampling + 2D convolution + batch normalization block, followed by three fully connected layers), the only difference between them is the activation function of the last, 1 unit layers, which is a sigmoid activation for the classifier and a linear function for the regressor.

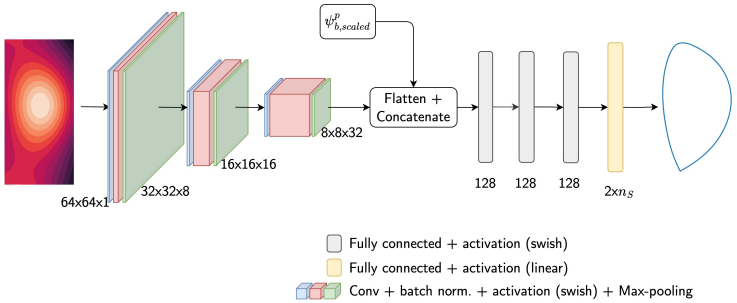
6.4. Separatrix Net: reconstruction of plasma separatrix

Separatrix Net is composed of two separate sub-networks, (fig. 8a, *Diverted Separatrix rec.* and *Limiter separatrix rec.*), both with the same architecture, but the first one used if the equilibrium is diverted and trained only on diverted separatrices, while the second one follows the same concept but for the limiter case. The equilibrium classification step, which has been performed by the XPlim Net, allows one to choose which of the two sub-networks to use (the switch in fig. 8a). Each one fulfills the task of predicting the $(\boldsymbol{r}^p_S, \boldsymbol{z}^p_S)$ coordinates of the Separatrix curve (Γ_p in fig. 1).

The need to use NNs for computing the separatrix from the flux map and the flux at the separatrix rather than using a conventional contouring algorithm (e.g., *marching squares* [50]) is because, for this task, such contouring algorithms can be used straightforwardly only in very few cases. For example, if we consider the separatrix in the diverted case (i.e., a contour line with an intersection at the X point position), depending on their implementation, contouring algorithms can return either a single curve intersecting itself, or they can divide the curve into multiple pieces, requiring further post-processing to obtain the desired single-pieced curves. This is the reason why usually ad-hoc algorithms are developed for separatrix identification [51, 52, 53, 54], which, in general, can



(a) Overall architecture of the *Separatrix Net*, showing its internal structure with the two sub-networks.



(b) Architecture of both the diverted and limiter separatrix reconstruction networks.

Figure 8

be computationally expensive and often require a careful implementation to be real-time applicable. NNs can be a good solution in this context, being (i) computationally efficient, (ii) easy to set up for a specific device as they can be trained on a database of already available separatrix curves of a specific tokamak, and (iii) possible to integrate them into the control loop.

The use of two different NNs for diverted and limiter equilibria (i.e., *Diverted Separatrix rec.* and *Limiter separatrix rec.*) is done because it can be shown that two networks, trained respectively on only limiter or diverted equilibria, can give a much more accurate result than a single network trained on the entire dataset. This approach of splitting the problem allows the model to learn more easily two specific tasks rather than a single, more general one. This explains the need to perform equilibrium classification as described in par. 6.3.

The inputs of these networks are (i) the scaled poloidal flux map $\boldsymbol{\psi}^p_{scaled}$, and the scaled flux at the separatrix $\psi^p_{b,scaled}$. The output is the separatrix $(\boldsymbol{r}^p_S, \boldsymbol{z}^p_S)$, shifted by $(-R_0, -Z_0)$, where (R_0, Z_0) is the machine center (i.e., $(6.2, 0)$m for the ITER-like device considered here). This is done to have a similar range of values for both r and z.

The need to provide also the flux at the separatrix as an input to the Sepatatrix network follows directly from the definition of separatrix for the limiter and diverted cases [32]. The separatrix for a diverted equilibrium has a clear definition, being the last closed flux surface intersecting itself at the

X-point. On the contrary, for a limiter equilibrium, the separatrix is defined using the last close flux surface that touches the first wall. It follows that, in this case, the same poloidal flux map can have a different separatrix if different first walls are considered. This does not happen for the diverted configuration, where the separatrix can be defined only by the X-point position. For this reason, adding as an input the flux at the separatrix can help for more efficient training for the limiter case.

7. Validation

The ITER-like device shown in fig. 1 has been used for validation. Starting from a given set of j_ϕ or p profiles, a database of 81986 synthetic equilibria has been generated by using the codes IET[55] and FRIDA[33]. By doing this, the equilibria are consistent with the set of profiles, providing a controlled procedure that allows for a quantification of the impact of non-magnetic measures on reconstruction accuracy.

The dataset $\mathbb{D}$ consists of all the 81986 equilibria, where 46735 are limiter and 35251 are diverted equilibria. A wide variety of cases has been considered, as can be seen in figure 9. The train/validation split ratio is 80%/20%, resulting in 65588 for the training dataset $\mathbb{D}_t$ and 16398 for the validation dataset $\mathbb{D}_v$.

For PlaNet implementation, we used TensorFlow-Keras [25, 56].

7.1. Equilibrium reconstruction

PlaNet *Equil Net* has been trained using *Adam* optimization algorithm [57] and batch size $N_b = 1024$, for three different sets of inputs:

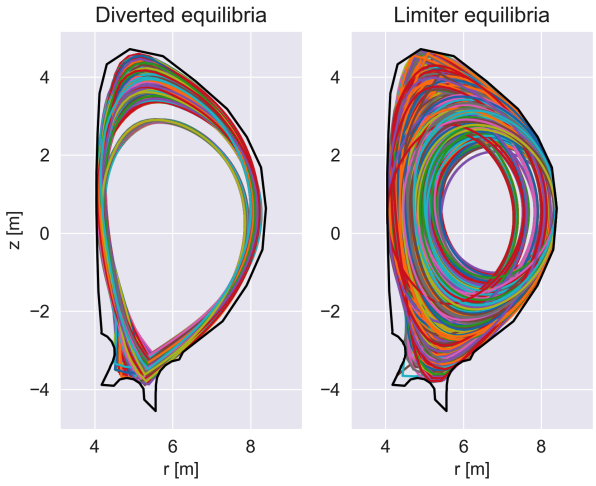


Figure 9: Overview of the diverted (left) and limiter (right) equilibria of the dataset.

1. active coil current, magnetic measures (*mag only* case);
2. active coil current, magnetic measures, j_ϕ profile (*mag + j* case);
3. active coil current, magnetic measures, p profile (*mag + p* case).

The first difference between these cases is clear when looking at the training history (i.e., the loss function against the training epochs), which is shown in fig. 10 (blue = *mag + only*, orange = *mag + j*, green = *mag + p*). We see that providing information internal to the plasma, in addition to the magnetic measures and the coils' current, allows for faster and more effective training. This is expected, being well known that information given by magnetic measures is limited to the external region of the plasma [8, 10]. However, this is even more clear by considering the point of view of the NN: let's consider two equilibria with the same set of magnetic measures, but different profiles. For the *mag only* case, the two inputs are equal, but the two outputs are different, causing the network to be misled and preventing it from learning because there is no clue to figure out why two equilibria are different starting from the same inputs. Conversely, including the profile j_ϕ pr p will lead to two different inputs, ensuring a proper distinction between the two cases on the input side, and allowing the NN to learn effectively.

The results achieved in the three cases are summarized in tables 1 and 2 in terms of R^2 and *Mean Squared Error* (MSE)[2], for $\boldsymbol{\psi}^p(r,z)$ and $\Delta^*\boldsymbol{\psi}^p(r,z)$, respectively. For $\boldsymbol{\psi}^p(r,z)$, the reconstruction achieves almost the same accuracy concerning R^2, while the MSE is smaller if the profiles are considered. On the other hand, $\Delta^*\boldsymbol{\psi}^p(r,z)$ is reconstructed with much greater accuracy if j_ϕ or p are taken into account, by a factor of 10 and 5 respectively w.r.t. the *mag only* case. It is important to notice also that the model achieves the same metrics both on the validation and the test set, meaning that it is not overfitting, i.e., it is able not only to learn but also to generalize. This is one of the inherent advantages of using a physics-informed loss function.

The fact that the improved accuracy achieved in the *mag + j* and *mag + p* cases is generally not evident if the poloidal flux map $\boldsymbol{\psi}^p(r,z)$ is considered is highlighted also in figs. from 11a to 11d and from 12a to 12d, where a comparison between the reference $\boldsymbol{\psi}(r,z)$ and the reconstructed $\boldsymbol{\psi}^p(r,z)$ in the *mag only* and *mag + p* cases is shown for two equilibria[3]. To be specific, this comparison is performed both via a contour plot of $\boldsymbol{\psi}(r,z)$ and $\boldsymbol{\psi}^p(r,z)$ (black dashed = reference, color = PlaNet), and by plotting a heat map of the percentage error. It can be seen from the heat maps that, for both cases, the percentage error is $\approx$1%, showing that both *mag only* and *mag + p* can accurately predict the poloidal flux.

Conversely, the difference between cases *mag + j*, *mag + p* and *mag only* is much more evident as the GS operator is considered. Cases *mag + j* and *mag + p* achieve better accuracy, especially in the core region of the plasma, as can

[2]See App. Appendix A for further details on these metrics.

[3]We report only the results obtained for the *mag + p* case, being the least accurate among *mag + j* and *mag + p* as shown in tab. 2.

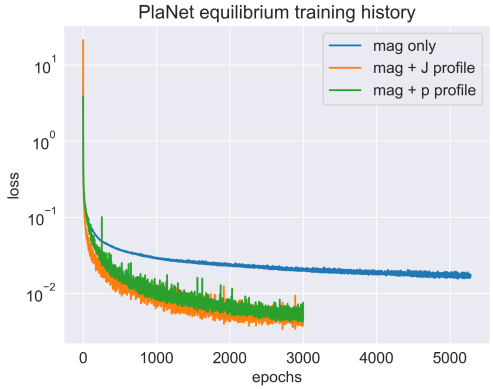


Figure 10: Training history (loss versus epochs) of *Euqilibrium Net* .

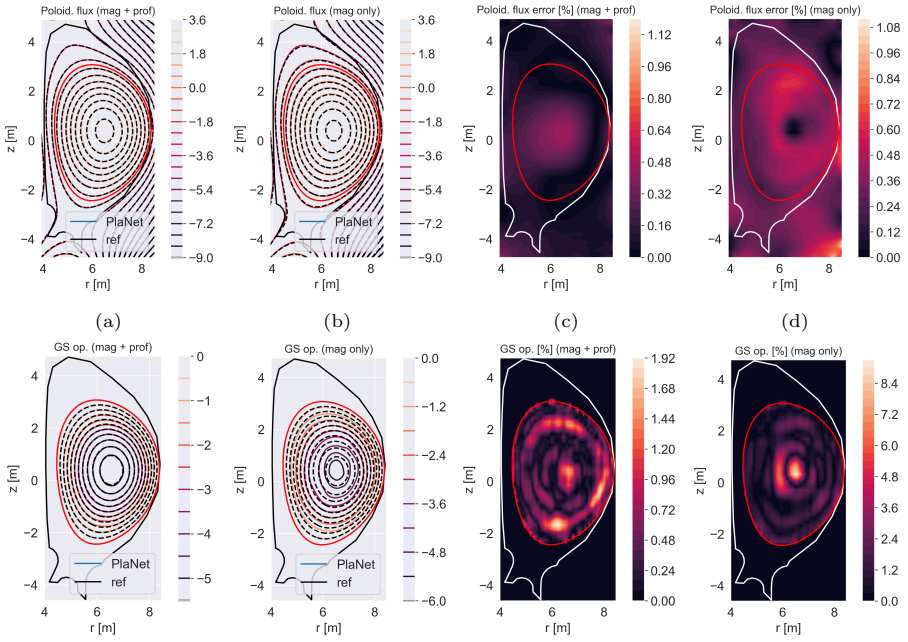


Figure 11: Comparison of *Euqilibrium Net* results with reference cases for an example taken from the validation dataset. Top = ψ^p comparison, bottom = $\Delta^*\psi^p$. From left to right: contour plots = comparison of contour levels for mag + p and *mag + only* cases; heat map = percentage error for *mag + p* and *mag + only* cases.

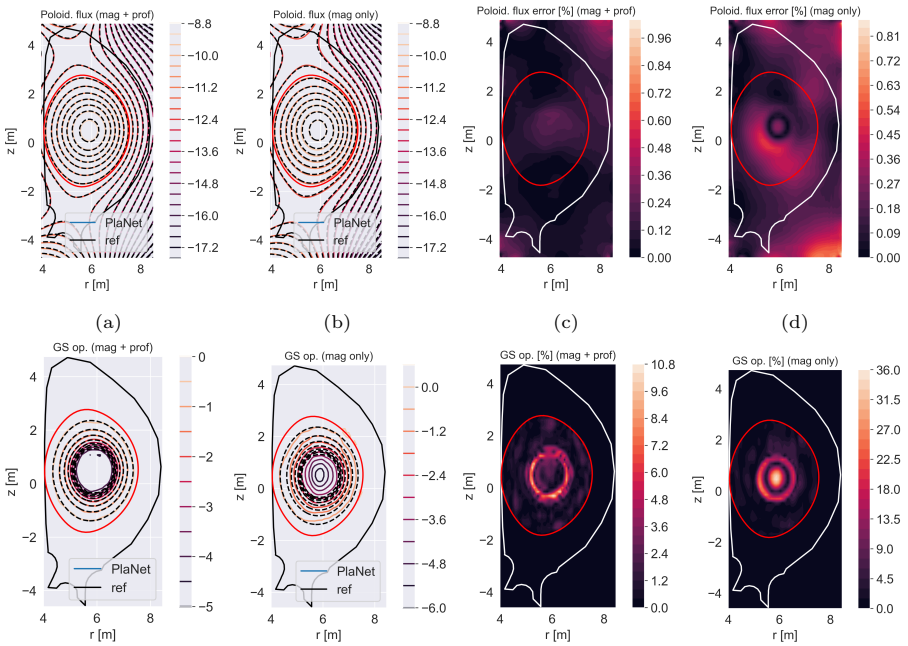


Figure 12: Comparison of *Euqilibrium Net* results with reference cases for an example taken from the validation dataset. Top = $\boldsymbol{\psi}^p$ comparison, bottom = $\Delta^*\boldsymbol{\psi}^p$. From left to right: contour plots = comparison of contour levels for *mag + p* and *mag + only* cases; heat map = percentage error for *mag + p* and *mag + only* cases.

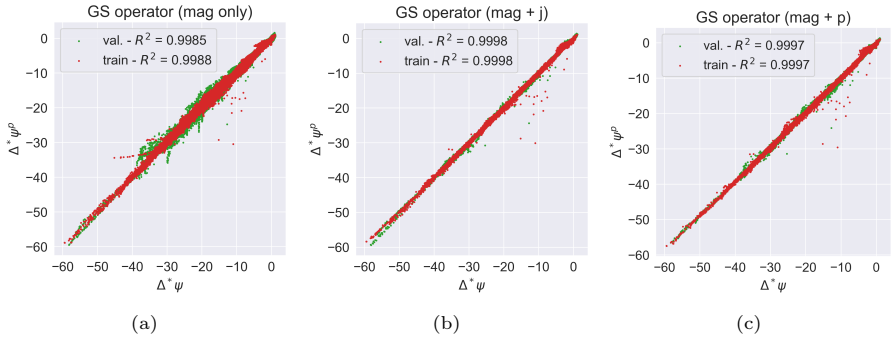


Figure 13: Performance comparison of *Euqilibrium Net* in computing the GS operator via eq. (12) depending on which input is considered. Here $\Delta^*\psi$ is the reference GS operator, while $\Delta^*\psi^p$ is the predicted one.

$\boldsymbol{\psi}^p(r,z)$				
	MSE_{train}	MSE_{val}	R^2_{train}	R^2_{val}
mag only	0.00152	0.00154	0.99997	0.99997
mag + j_ϕ prof.	0.00083	0.00083	0.99998	0.99998
mag + p prof.	0.00082	0.00083	0.99998	0.99998

Table 1: Scores of equilibrium network on $\boldsymbol{\psi}^p(r,z)$ predictions.

$\Delta^*\boldsymbol{\psi}^p(r,z)$				
	MSE_{train}	MSE_{val}	R^2_{train}	R^2_{val}
mag only	0.01958	0.02402	0.99878	0.99849
mag + j_ϕ prof.	0.00322	0.00312	0.99980	0.99980
mag + p prof.	0.00547	0.00547	0.99966	0.99966

Table 2: Scores of equilibrium network on $\Delta^*\boldsymbol{\psi}^p(r,z)$ predictions.

be seen in figs. 11e to 11h and 12e to 12h. The contour plots for the *magnetic only* case show a clear mismatch between the $\Delta^*\boldsymbol{\psi}^p(r,z)$ and $\Delta^*\boldsymbol{\psi}(r,z)$ iso-flux lines, especially in the core region. This is confirmed by the heat map of the percentage error.

Additionally, the improvement on $\Delta^*\boldsymbol{\psi}^p(r,z)$ can be seen also in fig. 13a, 13b and 13c, where the results for training validation datasets are separately reported, together with the coefficient of determination R^2.

As pointed out earlier, by using a neural operator we can specify the query points where $\boldsymbol{\psi}^p$ has to be predicted. This can be seen in fig. 14, where the input $\boldsymbol{u}$ has the same values of $[\boldsymbol{i}_c, \boldsymbol{m}_m, \boldsymbol{m}_n]$, while $[\boldsymbol{r}_q, \boldsymbol{z}_q]$ are different because we are considering two different grids (blue and green outline in fig. 14). This means that by properly selecting the query points $[\boldsymbol{r}_q, \boldsymbol{z}_q]$ we can predict $\boldsymbol{\psi}^p$ in any grid which is located inside the original one (blue in fig. 6a). We can see a comparison of $\boldsymbol{\psi}$ and $\boldsymbol{\psi}^p$ showing a good agreement in both cases.

7.2. Equilibrium classification and boundary flux reconstruction

The *XPlim Net* is trained with a batch size $N_b = 256$ and using the *Adam* optimization algorithm. The classification performance is summarized in tab. 15a in terms of accuracy, F_1 score, and the Receiver Operating Characteristic - Area Under Curve (ROC AUC). Concerning regression, *XPlim Net* scored $R^2_{train} = 0.9935$ and $R^2_{val} = 0.9937$. Additionally, the performance is also shown in fig. 15b. We can see that the results for the training and validation sets are almost identical, showing that the network can generalize well and avoid overfitting.

7.3. Separatrix reconstruction

The two *Separatrix Nets* are trained with a batch size $N_b = 256$ and using the *Adam* optimization algorithm as well. The training history for the two sub-networks can be seen in fig. 16, where we can notice how the training is

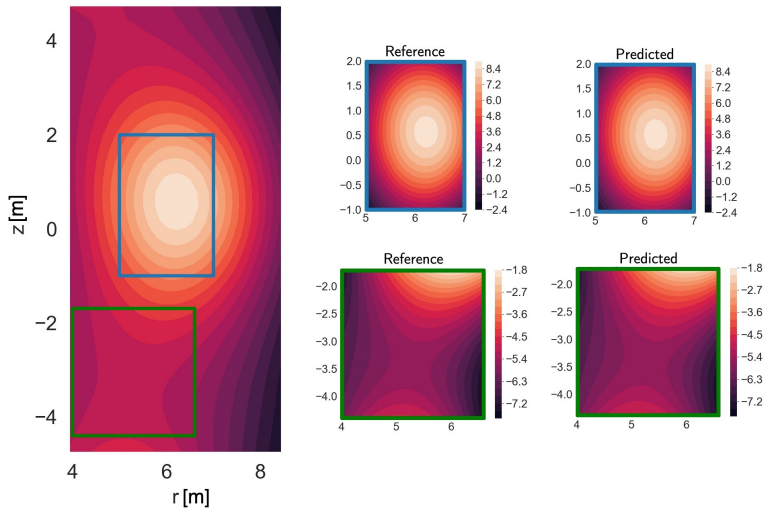
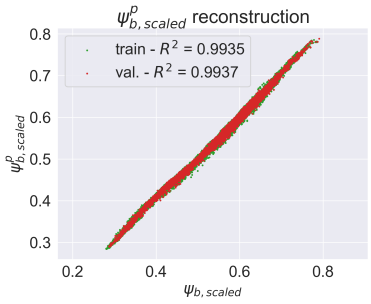


Figure 14: Comparison of $\boldsymbol{\psi}$ and $\boldsymbol{\psi}^p$ for two different sets of query points (outline is highlighted in blue and green on the left).

	Training	Validation
Accuracy	0.9896	0.9899
F_1 score	0.9880	0.9884
ROC AUC	0.9997	0.9997

(a)



(b)

Figure 15: Fig. 15b: performance of reconstruction of $\psi^p_{b,scaled}$, the scaled poloidal flux at the separatrix; tab. 15a: performance of equilibrium classification task.

	MSE_{train}	MSE_{val}	R^2_{train}	R^2_{val}
Diverted eq.	9.3×10^{-5}	9.3×10^{-5}	0.99997	0.99997
Limiter eq.	3.9×10^{-3}	7.2×10^{-3}	0.99845	0.99714

Table 3: Scores of *Separatrix Net* on $(\boldsymbol{r}^p_S, \boldsymbol{z}^p_S)$ reconstruction.

much more effective for the diverted case. This behavior is partly explainable by considering two facts:

- as we can see in fig. 9, diverted equilibria are more similar in terms of shape, while there is more variability for the limiter case. This fact can have a huge implication in affecting this behavior.
- For diverted equilibria, the separatrix is defined by considering only the value of the flux at the X-point, therefore not depending on the first wall position. Conversely, for limiter equilibria, different positions of the first wall give different separatrix curves. For this reason, the NN has to learn also this relation to properly map the inputs into the separatrix coordinates. Even if the flux at the separatrix $\psi^p_{b,scaled}$ is provided, this may not be enough to improve the training.

The first point however is much more likely to be the main reason to explain this behaviour, and further investigation on this would involve a training dataset with much more limiter equilibria, as well as including more detailed information on the first wall geometry. However, since this is outside the scope of this paper, this activity will be tackled in further work.

Table 3 summarizes the performance of the two sub-networks. We can see that the results for the limiter case are less accurate, as we expected. However, for both cases, the MSE is bonded below 10mm, which is consistent with the accuracy obtained by model-based separatrix identification techniques specifically developed for this purpose (e.g., ±10mm in [53]). Figs. 17a to 17d show, for some diverted and limiter equilibria, a comparison between the reference separatrix (black-solid) and the reconstructed ones (cyan-dashed).

8. Conclusion and further development

In this paper, we present PlaNet (*PLA*sma equilibrium reconstruction *NET*work), a deep-learning-based tool for performing plasma equilibrium and separatrix reconstruction considering both magnetic and non-magnetic measurements. PlaNet

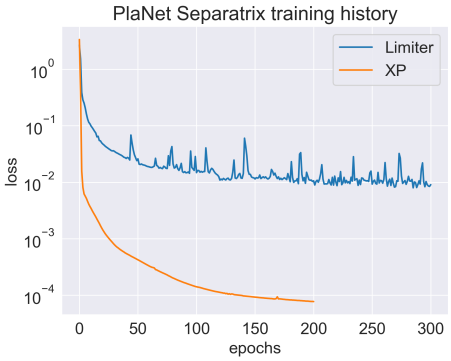
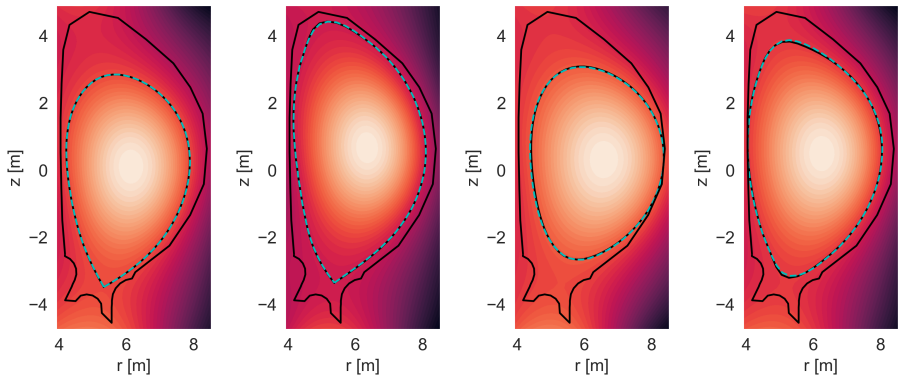


Figure 16: Training history for Diverted Separatrix rec. and Limiter separatrix rec. networks



(a) (b) (c) (d)

Figure 17: Examples of separatrix reconstruction: reference (black) and predicted (cyan dashed).

consists of three NNs running at three different stages: (i) *Equilibrium Net*, the physics-informed neural operator accounting for the accurate reconstruction of the poloidal flux map; (ii) *XPlim Net*, which classifies if the equilibrium is diverted or limiter and computes the poloidal flux at the separatrix; and (iii) *Separatrix Net*, which gives the (r, z) coordinates of the separatrix. PlaNet can use magnetic measures only, as common equilibrium reconstruction tools, but also use non-magnetic measures such as j_ϕ or p profiles.

The plasma equilibrium solver behind this tool exploits a convolutional physics-informed neural operator, which solves the plasma equilibrium problem satisfying the GSE, meaning that the poloidal flux map $\boldsymbol{\psi}^p$ is differentiable, and we can compute the GS operator, and then the source term, directly differentiating $\boldsymbol{\psi}^p$. Concerning $\boldsymbol{\psi}^p$ reconstruction, using non-magnetic measures or not does not affect the accuracy very much w.r.t. the case of using only magnetic measures. Conversely, for $\Delta^*\boldsymbol{\psi}^p$ there is a clear improvement (i.e., 5 to 10 times less Mean Squared Error) if non-magnetic measures are considered, especially in the core region of the plasma.

We showed also that *Euqilibrium Net* can predict $\boldsymbol{\psi}^p$ at an *arbitrary* location, and not only on a specified grid defined *a priori*, ensuring accurate results for localized reconstructions.

Separatrix reconstruction is performed with two different NNs with the same architecture, but trained specifically to work with diverted or limiter equilibria. *Separatrix Net* achieves an error that is bound below $\pm$0.1mm for diverted and $\pm$10mm for limiter, values that are consistent with error bounds achieved by other model-based separatrix reconstruction tools (e.g., $\pm$10mm in [53]).

These results, together with the reduced computational time required by PlaNet to perform the computations, show that PlaNet is an effective tool, suitable to be used to help plan experimental operations [58, 59, 33, 60], or used as a real-time plasma state estimation [61, 62, 63] integrated within a Plasma

Control System (PCS). Further development would indeed go in this direction, improving PlaNet reliability and robustness.

Appendix A. Validation metrics used in this work

- **Mean Squared Error** (MSE): it measures the average of the squares of the errors

$$MSE = \frac{1}{M}\sum_{i=1}^{M}\left(y_i - y_i^p\right)^2 \tag{A.1}$$

  where y_i is the ith item of the database, y_i^p is the corresponding prediction, and M is the total number of examples in the considered database.

- **coefficient of determination:** denoted as R^2, it is the proportion of the variation in the dependent variable(s) that is predictable from the independent variable(s). $R^2 = 1$ means that the model can reproduce the reference results exactly. It is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{M}(y_i - y_i^p)^2}{\sum_{i=1}^{M}\left(y_i + \frac{1}{M}\sum_{j=1}^{M} y_j\right)^2} \tag{A.2}$$

- **Classification accuracy:** In a classification problem, the classification accuracy is the ratio of the number of correct predictions over the total number of input examples.

- F_1 **score:**

$$F_1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{A.3}$$

  where $precision$ is the number of true positive results divided by the number of all positive results, and the $recall$ is the number of true positive results divided by the number of all samples that should have been identified as positive.

- **Receiver Operating Characteristic (ROC):** Receiver Operating Characteristic (ROC) curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. Area Under Curve (ROC AUC) gives a quantitative metric of this performance.

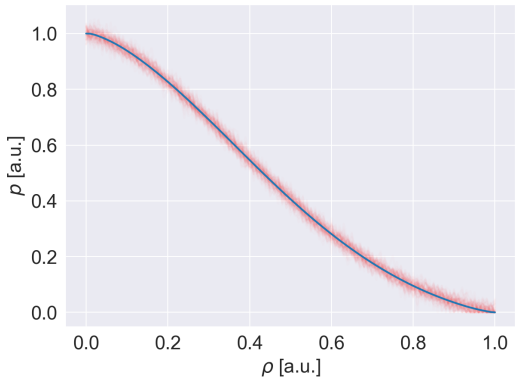


Figure B.18: Input pressure profiles p (normalized) used in the sensitivity analysis. Here ρ is the same coordinate defined in sec. 4

Appendix B. Sensitivity of eqilibirum reconstruction to noise on p, j_ϕ profiles

The real-time reconstruction of p and j_ϕ profiles relies on measures coming from several diagnostics [10, 31], typically affected by different levels of noise. In this section, we briefly summarize a sensitivity analysis carried out to assess the robustness of PlaNet *Equilibrium Net* to noise that may affect the profiles. The sensitivity analysis has been carried out separately for the two cases of reconstruction starting from p and j_ϕ.

For each profile, a family of profiles has been obtained by superimposing a Gaussian noise with a standard deviation of 5% with respect to the original one. Figure B.18 shows an example of these families for a single pressure profile p. The reconstructed equilibria have been compared, in terms of the R^2 parameter, to those reconstructed without noise and presented in par. 7.1. For both the cases of noisy p or j_ϕ, the relative error $\epsilon = (R^2 - R^2_{noise})/R^2$ is bound below $< 0.1\%$.

A reason for this low sensitivity to the noise can be found in the intrinsic regularizing properties typical of PINNs and physics-informed neural operators. By including a physical term in the loss function, the NN can achieve better generalization capabilities and is less prone to overfitting.

Acknowledgments

The authors thank Dr. Cihan Akcay and Dr. Paolo Zanca for the interesting and useful discussions on this topic.

References

[1] H. Grad, H. Rubin, Hydromagnetic equilibria and force-free fields, Journal of Nuclear Energy (1954) 7 (3-4) (1958) 284–285.

[2] V. Shafranov, Plasma equilibrium in a magnetic field, Reviews of plasma physics 2 (1966) 103.

[3] L. Lao, H. S. John, R. Stambaugh, A. Kellman, W. Pfeiffer, Reconstruction of current profile parameters and plasma shapes in tokamaks, Nuclear Fusion 25 (11) (1985) 1611. doi:10.1088/0029-5515/25/11/007.

[4] J. Blum, Numerical simulation and optimal control in plasma physics (1989).

[5] X. Yue, B. Xiao, Z. Luo, Y. Guo, Fast equilibrium reconstruction for tokamak discharge control based on gpu, Plasma Physics and Controlled Fusion 55 (8) (2013) 085016.

[6] J.-M. Moret, B. Duval, H. Le, S. Coda, F. Felici, H. Reimerdes, Tokamak equilibrium reconstruction code liuqe and its real time implementation, Fusion Engineering and Design 91 (2015) 1–15. doi:https://doi.org/10.1016/j.fusengdes.2014.09.019.

[7] M. Rampp, R. Preuss, R. Fischer, A. U. Team, Gpec: A real-time–capable tokamak equilibrium code, Fusion Science and Technology 70 (1) (2016) 1–13.

[8] P. Mc Carthy, Analytical solutions to the grad–shafranov equation for tokamak equilibrium with dissimilar source functions, Physics of Plasmas 6 (9) (1999) 3554–3560.

[9] F. Alladio, P. Micozzi, Experimental plasma equilibrium reconstruction from kinetic and magnetic measurements in the ftu tokamak, Nuclear Fusion 35 (3) (1995) 305. doi:10.1088/0029-5515/35/3/I05.

[10] G. Q. Li, Q. L. Ren, J. P. Qian, L. L. Lao, S. Y. Ding, Y. J. Chen, Z. X. Liu, B. Lu, Q. Zang, Kinetic equilibrium reconstruction on east tokamak, Plasma Physics and Controlled Fusion 55 (12) (2013) 125008. doi:10.1088/0741-3335/55/12/125008.

[11] R. Fischer, A. Bock, M. Dunne, J. Fuchs, L. Giannone, K. Lackner, P. McCarthy, E. Poli, R. Preuss, M. Rampp, et al., Coupling of the flux diffusion equation with the equilibrium reconstruction at asdex upgrade, Fusion Science and Technology 69 (2) (2016) 526–536.

[12] Z. Xing, D. Eldon, A. Nelson, M. Roelofs, W. Eggert, O. Izacard, A. Glasser, N. Logan, O. Meneghini, S. Smith, R. Nazikian, E. Kolemen, Cake: Consistent automatic kinetic equilibrium reconstruction, Fusion Engineering and Design 163 (2021) 112163. doi:https://doi.org/10.1016/j.fusengdes.2020.112163.

[13] M. W. M. G. Dissanayake, N. Phan-Thien, Neural-network-based approximations for solving partial differential equations, Communications in Numerical Methods in Engineering 10 (3) (1994) 195–201. doi:https://doi.org/10.1002/cnm.1640100303.

[14] B. P. van Milligen, V. Tribaldos, J. Jiménez, Neural network differential equation and plasma equilibrium solver, Physical review letters 75 (20) (1995) 3594.

[15] M. Raissi, P. Perdikaris, G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics 378 (2019) 686–707.

[16] E. Haghighat, M. Raissi, A. Moure, H. Gomez, R. Juanes, A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics, Computer Methods in Applied Mechanics and Engineering 379 (2021) 113741.

[17] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, L. Yang, Physics-informed machine learning, Nature Reviews Physics 3 (6) (2021) 422–440.

[18] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, B. Liu, K. Azizzadenesheli, A. Anandkumar, Physics-informed neural operator for learning partial differential equations, arXiv preprint arXiv:2111.03794 (2021).

[19] S. Goswami, A. Bora, Y. Yu, G. E. Karniadakis, Physics-informed neural operators, arXiv preprint arXiv:2207.05748 (2022).

[20] L. Lu, P. Jin, G. E. Karniadakis, Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators, arXiv preprint arXiv:1910.03193 (2019).

[21] L. Lu, P. Jin, G. Pang, Z. Zhang, G. E. Karniadakis, Learning nonlinear operators via deeponet based on the universal approximation theorem of operators, Nature machine intelligence 3 (3) (2021) 218–229.

[22] S. Joung, J. Kim, S. Kwak, J. Bak, S. Lee, H. Han, H. Kim, G. Lee, D. Kwon, Y.-C. Ghim, Deep neural network grad–shafranov solver constrained with measured magnetic signals, Nuclear Fusion 60 (1) (2019) 016034. doi:10.1088/1741-4326/ab555f.

[23] J. Wai, M. Boyer, E. Kolemen, Neural net modeling of equilibria in nstx-u, arXiv preprint arXiv:2202.13915 (2022).

[24] L. L. Lao, S. Kruger, C. Akcay, P. Balaprakash, T. Bechtel, E. Howell, J. Koo, J. Leddy, M. Leinhauser, Y. Liu, et al., Application of machine learning and artificial intelligence to extend efit equilibrium reconstruction, Plasma Physics and Controlled Fusion (2022).

[25] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, software available from tensorflow.org (2015).

[26] Theano Development Team, Theano: A Python framework for fast computation of mathematical expressions, arXiv e-prints abs/1605.02688 (May 2016).

[27] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, Q. Zhang, JAX: composable transformations of Python+NumPy programs (2018).

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[29] Q. Wang, M. Ihme, Y.-F. Chen, J. Anderson, A tensorflow simulation framework for scientific computing of fluid flows on tensor processing units, Computer Physics Communications 274 (2022) 108292. doi:https://doi.org/10.1016/j.cpc.2022.108292.

[30] X. Zhao, Z. Gong, Y. Zhang, W. Yao, X. Chen, Physics-informed convolutional neural networks for temperature field prediction of heat source layout without labeled data, Engineering Applications of Artificial Intelligence 117 (2023) 105516. doi:https://doi.org/10.1016/j.engappai.2022.105516.

[31] F. Felici, O. Sauter, S. Coda, B. Duval, T. Goodman, J.-M. Moret, J. Paley, the TCV Team, Real-time physics-model-based simulation of the current density profile in tokamak plasmas, Nuclear Fusion 51 (8) (2011) 083052. doi:10.1088/0029-5515/51/8/083052.

[32] J. P. Freidberg, ideal MHD, Cambridge University Press, 2014.

[33] M. Bonotto, D. Abate, P. Bettini, F. Villone, A coupled fem-bem approach for the solution of the free-boundary axi-symmetric plasma equilibrium problem, Commun. Comput. Phys. 31 (1) (2022) 27–59.

[34] J. Lee, A. Cerfon, Ecom: a fast and accurate solver for toroidal axisymmetric mhd equilibria, Computer Physics Communications 190 (2015) 72–88.

[35] L. LoDestro, L. Pearlstein, On the grad–shafranov equation as an eigenvalue problem, with implications for q solvers, Physics of plasmas 1 (1) (1994) 90–95.

[36] Y. M. Jeon, Development of a free-boundary tokamak equilibrium solver for advanced study of tokamak equilibria, Journal of the Korean Physical Society 67 (5) (2015) 843–853.

[37] L. Lao, S. Hirshman, R. Wieland, Variational moment solutions to the grad–shafranov equation, The Physics of Fluids 24 (8) (1981) 1431–1440.

[38] F. Helton, T. Wang, Mhd equilibrium in non-circular tokamaks with field-shaping coil systems, Nuclear Fusion 18 (11) (1978) 1523.

[39] K. Kim, G. Jeun, Analysis of equilibrium in a tokamak by the finite-difference method, Research Reports. Hanyang Research Institute of Industrial Sciences 16 (1983).

[40] J. López, E. Orozco, V. Dougar-Zhabon, Fixed boundary grad-shafranov solver using finite difference method in nonhomogeneous meshgrid, in: Journal of Physics: Conference Series, Vol. 1159, IOP Publishing, 2019, p. 012017.

[41] Q. Zang, J. Zhao, L. Yang, Q. Hu, X. Xi, X. Dai, J. Yang, X. Han, M. Li, C. L. Hsieh, Upgraded multipulse laser and multipoint Thomson scattering diagnostics on EAST, Review of Scientific Instruments 82 (6) (2011) 063502. doi:10.1063/1.3599039.

[42] Y. Shi, F. Wang, B. Wan, M. Bitter, S. Lee, J. Bak, K. Hill, J. Fu, Y. Li, W. Zhang, A. Ti, B. Ling, Imaging x-ray crystal spectrometer on east, Plasma Physics and Controlled Fusion 52 (8) (2010) 085014. doi:10.1088/0741-3335/52/8/085014.

[43] M. A. V. Zeeland, W. W. Heidbrink, J. H. Yu, Fast ion d_α imaging in the diii-d tokamak 51 (5) (2009) 055001. doi:10.1088/0741-3335/51/5/055001.

[44] T. Chen, H. Chen, Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems, IEEE transactions on neural networks 6 (4) (1995) 911–917.

[45] S. Goswami, A. Bora, Y. Yu, G. E. Karniadakis, Physics-informed deep neural operator networks, in: Machine Learning in Modeling and Simulation: Methods and Applications, Springer, 2023, pp. 219–254.

[46] G. Vayakis, S. Arshad, D. Delhom, A. Encheva, T. Giacomin, L. Jones, K. Patel, M. Pérez-Lasala, M. Portales, D. Prieto, et al., Development of the iter magnetic diagnostic set and specification, Review of Scientific Instruments 83 (10) (2012) 10D712.

[47] D. R. Ferreira, P. J. Carvalho, H. Fernandes, J. Contributors, Full-pulse tomographic reconstruction with deep neural networks, Fusion Science and Technology 74 (1-2) (2018) 47–56. doi:10.1080/15361055.2017.1390386.

[48] A. Odena, V. Dumoulin, C. Olah, Deconvolution and checkerboard artifacts, Distill (2016). doi:10.23915/distill.00003.

[49] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations, arXiv preprint arXiv:1711.10561 (2017).

[50] C. Maple, Geometric design and space planning using the marching squares and marching cube algorithms, in: 2003 international conference on geometric modeling and graphics, 2003. Proceedings, IEEE, 2003, pp. 90–95.

[51] S. Tsaun, H. Jhang, Real-time plasma boundary reconstruction in the kstar tokamak using finite element method, Fusion Engineering and Design 82 (2) (2007) 163–174.

[52] A. Beghi, A. Cenedese, Advances in real-time plasma boundary reconstruction: From gaps to snakes, IEEE Control Systems Magazine 25 (5) (2005) 44–64.

[53] A. Cenedese, P. Bettini, M. Bonotto, Model-based approach for magnetic reconstruction in axisymmetric nuclear fusion machines, IEEE Transactions on Plasma Science 46 (3) (2018) 636–644.

[54] M. Bonotto, P. Bettini, Ares: a fast and accurate tool for the identification of plasma stationary points and separatrix, Nuclear Fusion 62 (7) (2022) 076035.

[55] D. Abate, P. Bettini, An inverse equilibrium tool to define axisymmetric plasma equilibria, Plasma Physics and Controlled Fusion 61 (10) (2019) 105016. doi:10.1088/1361-6587/ab3f09.

[56] F. Chollet, et al., Keras (2015).
URL `https://keras.io`

[57] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

[58] H. Heumann, J. Blum, C. Boulbe, B. Faugeras, G. Selig, J.-M. Ané, S. Brémond, V. Grandgirard, P. Hertout, E. Nardon, Quasi-static free-boundary equilibrium of toroidal plasma with cedres++: Computational methods and applications, Journal of Plasma Physics 81 (3) (2015) 905810301.

[59] R. Albanese, R. Ambrosino, M. Mattei, Create-nl+: A robust control-oriented free boundary dynamic plasma equilibrium solver, Fusion Engineering and Design 96 (2015) 664–667.

[60] M. Bonotto, D. Abate, P. Bettini, A. Iaiunese, N. Isernia, F. Villone, Efficient numerical solution of coupled axisymmetric plasma equilibrium and eddy current problems, IEEE Access 11 (2023) 27489–27505. doi:10.1109/ACCESS.2023.3253380.

[61] D. Piglowski, J. Ferron, P. Gohil, R. Johnson, B. Penaflor, Enhancements in the second generation diii-d digital plasma control system, Fusion engineering and design 82 (5-14) (2007) 1058–1063.

[62] Y. Huang, B. Xiao, Z. Luo, Q. Yuan, X. Pei, X. Yue, Implementation of gpu parallel equilibrium reconstruction for plasma control in east, Fusion Engineering and Design 112 (2016) 1019–1024.

[63] Y. Huang, B. Xiao, Z. Luo, Q. Yuan, Improvement of gpu parallel real-time equilibrium reconstruction for plasma control, Fusion Engineering and Design 128 (2018) 82–85.