Tokamak equilibrium reconstruction code LIUQE and its real time implementation

J.-M. Moret[1], B.P. Duval, H.B. Le, S. Coda, F. Felici[2], H. Reimerdes

CRPP - EPFL
Centre de Recherches en Physique des Plasmas
École Polytechnique Fédérale de Lausanne
CH-1015, Lausanne, Switzerland

Highlights.

- Algorithm vertical stabilisation using a linear parametrisation of the current density
- Experimentally derived model of the vacuum vessel to account for vessel currents
- Real-time contouring algorithm for flux surface averaged 1.5 D transport equations
- Full real time implementation coded in SIMULINK runs in less than 200 µs
- Applications: shape control, safety factor profile control, coupling with RAPTOR

Abstract. Equilibrium reconstruction consists in identifying, from experimental measurements, a distribution of the plasma current density that satisfies the pressure balance constraint. The LIUQE code adopts a computationally efficient method to solve this problem, based on an iterative solution of the Poisson equation coupled with a linear parametrisation of the plasma current density. This algorithm is unstable against vertical gross motion of the plasma column for elongated shapes and its application to highly shaped plasmas on TCV requires a particular treatment of this instability. TCV's continuous vacuum vessel has a low resistance designed to enhance passive stabilisation of the vertical position. The eddy currents in the vacuum vessel have a sizeable influence on the equilibrium reconstruction and must be taken into account. A real time version of LIUQE has been implemented on TCV's distributed digital control system with a cycle time shorter than 200 µs for a full spatial grid of 28 by 65, using all 133 experimental measurements and including the flux surface average of quantities necessary for the real time solution of 1.5 D transport equations. This performance was achieved through a thoughtful choice of numerical methods and code optimisation techniques at every step of the algorithm, and was coded in MATLAB and SIMULINK for the off-line and real time version respectively.

Keywords. Tokamak, MHD, equilibrium, reconstruction, real-time.

Published in Fusion Engineering and Design 91 (2015) 1.

[1] Corresponding author at: Centre de Recherches en Physique des Plasmas, École Polytechnique Fédérale de Lausanne, Station 13, CH-1015, Lausanne, Switzerland, Tel.: +41 21 693 34 14, e-mail address: jean-marc.moret@epfl.ch.

[2] Present address: Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands.

1 Introduction

The need to operate Tokamak plasmas in scenarios optimised for fusion performance by all available means calls for simultaneous real time control of many parameters. This covers monitoring of the plasma cross-section shape in view of its role in plasma stability and transport processes, novel divertor configurations to mitigate the erosion of plasma facing components, together with temperature, pressure and current profile tailoring, e.g. to maximise the bootstrap current or sustain internal transport barriers. A crucial component in this control task is the MHD equilibrium that defines the geometry of the confining magnetic field and of the divertor. Knowledge of the immediate geometry may be directly used to feedback control the plasma shape and divertor parameters, and guide the choice of location of heating and current drive. The situation in TCV is especially challenging. TCV often accommodates a large variety of plasma shapes, together with various open divertor configurations including single and double null divertor. This diversity has recently been further extended to high order null points with more than two divertor legs such as in the so called "snow flake" divertor [1]. In TCV, the vacuum vessel, and thus the magnetic measurements, may be located far from the plasma boundary and from the X points, thus complicating shape and divertor control based on flux extrapolation.

Quantities derived from the geometry alone can also serve in other physics based analysis or prediction real time codes, such as RAPTOR [2], which solves flux surface averaged 1.5D equations for particle and heat transport and for magnetic field resistive diffusion. This code then estimates the spatial distribution of relevant plasma parameters such as the pressure and current profiles. A control law based on this parameter estimate may then determine actuator settings or the future trajectory towards a favoured operational point.

The inverse equilibrium problem consists in identifying an equilibrium satisfying the MHD pressure balance condition and best matching the available measurements. There are several proposed approaches to this problem that have been continuously developed during the last decades; the closest implementation to the LIUQE code [3] used on TCV is EFIT [4]. It is in essentially based on a parametrisation of the plasma current density by a linear combination of selected base functions. However TCV poses specific requirements to be fulfilled by LIUQE. Firstly the highly elongated configurations operated on TCV [5] make the inverse equilibrium algorithm unstable; so a proper, and computationally efficient, stabilisation method must be engineered. To increase the passive stabilisation of the vertical position of an elongated configuration, the vacuum vessel of TCV was designed with low electrical resistivity; the resulting large image currents in the vacuum vessel have a sizeable influence on the equilibrium and must be accounted for in the inverse equilibrium reconstruction. LIUQE also has provision, not reported in the present article, for treating equilibria with more than one magnetic axis, such as the doublet configuration already created on TCV [6]. Despite these specificities, LIUQE can be readily implemented on any air cored Tokamak with axisymmetric passive conducting structures. To that purpose, machine specific code was grouped in dedicated initialisation blocks, and its interface with the *European International Tokamak Modelling Task Force* data structure [7] is under construction.

A real time implementation of the inverse equilibrium reconstruction for TCV imposes stringent constraints on the computation cycle time. Ignoring, for the present, disruptive and vertical displacement events, the required cycle time is imposed by the characteristic time constant of the vacuum vessel image currents and the corresponding response time of the poloidal field coil power supplies, typically 1ms. Despite a continuous increase in the computation speed of processors, only a few of the available real time equilibrium reconstruction codes can approach this figure, by distributing the computation load across several processors requiring advanced programming techniques [8] or dedicated hardware [9]. For the real time implementation of LIUQE on TCV, careful attention to the efficiency of competing numerical techniques in all steps of the algorithm together with an efficient, yet user friendly, code generation using the SIMULINK programming environment yielded a cycle time of less that 200 μs on a single INTEL processor core. This was achieved using the full spatial grid and the entire available measurement set, and including flux contour identification and calculation of flux averaged quantities involved in the 1.5 D transport equations.

The path to this achievement is the subject of this article. Section 2 introduces the background for the inversion equilibrium reconstruction, with a particular emphasis on the stability of the algorithm with respect to vertical position and the consequences on the parametrisation of the plasma current density. Section 3 lists the used measurements, cautiously treating their coupling to the vertical position and the vacuum vessel image currents. Section 4 presents the details of the numerical methods chosen for the steps of the algorithm with, in particular, a fast fitting method for the plasma current source terms. Finally, the real time implementation is presented in Section 5, including details of the real time algorithm scheme, accompanied by a real time dedicated contouring algorithm for mapping the flux surfaces, permitting the subsequent evaluation of flux surface averaged geometrical quantities, the hardware, and the applied code optimisation techniques.

2 Inverse equilibrium reconstruction

2.1 Background

In axisymmetric geometry, assuming isotropic pressure and no net fluid velocity, the equations describing ideal magnetohydrodynamic equilibrium

$$\begin{aligned} \boldsymbol{j} \times \boldsymbol{B} &= \nabla p \\ \nabla \times \boldsymbol{B} &= \mu_0 \boldsymbol{j} \\ \nabla \cdot \boldsymbol{B} &= 0 \end{aligned} \tag{1}$$

can be combined, by writing $\boldsymbol{B}$ in cylindrical coordinates r, z, φ as

$$\boldsymbol{B} = -\frac{1}{2\pi r}\frac{\partial\psi}{\partial z}\nabla r + \frac{1}{2\pi r}\frac{\partial\psi}{\partial r}\nabla z + T\nabla\varphi\,, \tag{2}$$

into a second order differential Poisson like equation:

$$\Delta^*\psi = -2\pi\mu_0 r j_\varphi \tag{3}$$

with

$$j_\varphi = 2\pi\left(r\frac{dp}{d\psi} + \frac{T}{\mu_0 r}\frac{dT}{d\psi}\right) \tag{4}$$

and the definition of the elliptical operator $\Delta^* = r\frac{\partial}{\partial r}\frac{\partial}{r\partial r}+\frac{\partial^2}{\partial z^2}$. Here p and T are arbitrary functions of the poloidal magnetic flux ψ only. Combining these two equations gives the Grad-Shafranov equation:

$$\Delta^*\psi = -4\pi^2\mu_0 r\left(rp' + \frac{TT'}{\mu_0 r}\right). \tag{5}$$

The goal of the inverse equilibrium problem is to identify the two functions $p'(\psi)$ and $TT'(\psi)$, together with a flux function $\psi(r, z)$, that satisfy equation (5) and that best reproduce the available experimental measurements of parameters related to the physical quantities behind these functions.

In the presence of an external - other than the plasma current - toroidal current density, j_e, equation (3) must be completed as follows:

$$\Delta^*\psi = -2\pi\mu_0 r(j_\varphi + j_e). \tag{6}$$

Dirichlet boundary condition must be specified at the boundary of the finite integration domain Ω. For clarity, the contributions from the plasma current ψ_p and the external currents ψ_e are separated: $\psi(r, z) = \psi_p(r, z) + \psi_e(r, z)$ on $\partial\Omega$ where

$$\psi_p(r, z) = \int M(r, z, r', z')j_\varphi(r', z')dr'dz' \tag{7}$$

and

$$\psi_e(r, z) = \int M(r, z, r', z')j_e(r', z')dr'dz'. \tag{8}$$

Here $M(r, z, r', z')$ is the mutual inductance between coaxial circles located at (r, z) and (r', z'); its calculation is detailed in the Appendix A. The singularity in M when (r, z) and (r', z') coincide is avoided as

$$j_\varphi = j_e = 0 \text{ on } \partial\Omega. \tag{9}$$

Except for trivial functions p' and TT', that are inadequate for realistic equilibrium, and especially because the plasma is surrounded by a vacuum region whose boundary depends on the solution itself, equation (5) is intrinsically non linear and calls for dedicated solving methods. The simplest algorithm, the so called Picard iterations, consists in solving iteratively equation (6):

$$\Delta^*\psi^{(t+1)} = -2\pi\mu_0 r(j_\varphi^{(t)}(r, \psi^{(t)}) + j_e) \tag{10}$$

where t labels the iteration number. In the framework of equilibrium reconstruction, the boundary between the plasma volume and the vacuum is updated at each iteration based on $\psi^{(t)}$. The functions $p'^{(t)}(\psi^{(t)})$ and $TT'^{(t)}(\psi^{(t)})$ forming the plasma current density are also adjusted at each iteration to best reproduce available experimental measurements, thus justifying the t index in $j_\varphi^{(t)}$ in equation (10). The iterative procedure is seeded with an initial guess for the current density $j_\varphi^{(0)}(r, z)$ obtained from the method presented in Section Fig. 4. Thus the first step consists in solving equation (10) for $\psi^{(1)}$ with a known $j_\varphi^{(0)}$.

2.2 Vertical position stabilisation

It was soon recognised that the Picard iteration scheme is unstable against bulk displacement of the plasma column, especially in the vertical direction [10]. This can be demonstrated as follows. Assume that one exact solution for equation (6) is known, $\psi_0(r, z)$ for a given source function $j_{\varphi 0}(r, \psi_0)$. The corresponding boundary condition can be written $\psi_0 = \psi_{p0} + \psi_e$ on $\partial\Omega$ where ψ_{p0} is obtained by substituting $j_{\varphi 0}$ for j_φ in equation (7). Now also assume that at iteration t, for some reason related for example to numerical round-off error, to he approximation of the differential operator or to the evaluation of the boundary condition, the flux distribution is vertically shifted by $\delta z^{(t)}$; this modifies the plasma current density that can be approximated by its first order expansion:

$$j_\varphi^{(t)} = j_{\varphi 0}(r, \psi_0(r, z + \delta z^{(t)})) \cong j_{\varphi 0} + \delta z^{(t)} \partial_z j_{\varphi 0}. \tag{11}$$

In the iteration process, this is used to solve the Poisson equation and update the flux; to wit

$$\Delta^* \psi^{(t+1)} = -2\pi\mu_0 r(j_{\varphi 0} + \delta z^{(t)} \partial_z j_{\varphi 0} + j_e) \tag{12}$$

with the boundary conditions obtained with the shifted plasma current distribution and the contribution from the external current kept constant; this can also be approximated by its first order expansion:

$$\psi^{(t+1)} \cong \psi_{p0} + \delta z^{(t)} \partial_z \psi_{p0} + \psi_e \text{ on } \partial\Omega. \tag{13}$$

The reader can verify that the solution of (12) and (13) is $\psi^{(t+1)} = \psi_0 + \delta z^{(t)} \partial_z(\psi_0 - \psi_e)$.

The position of the magnetic axis of the exact solution (r_A, z_A) can be defined as the position in the plasma where ψ_0 presents an extremum, i.e. $\nabla\psi_A = 0$ and $det\ H(\psi_A) > 0$, where H is the Hessian matrix. The subscript A indicates that the quantity must be evaluated on (r_A, z_A). Near this point, a second order Taylor expansion in $r - r_A$ and $z - z_A$ for ψ_0 and ψ_e are used to search the magnetic axis vertical position of the updated flux distribution as the solution of $\nabla\psi^{(t+1)}(r_A^{(t+1)}, z_A^{(t+1)}) = 0$:

$$\delta z^{(t+1)} = z_A - z_A^{(t+1)} = \left(1 - \frac{\partial_z^2 \psi_{eA}}{\partial_z^2 \psi_{0A}}\right) \delta z^{(t)}. \tag{14}$$

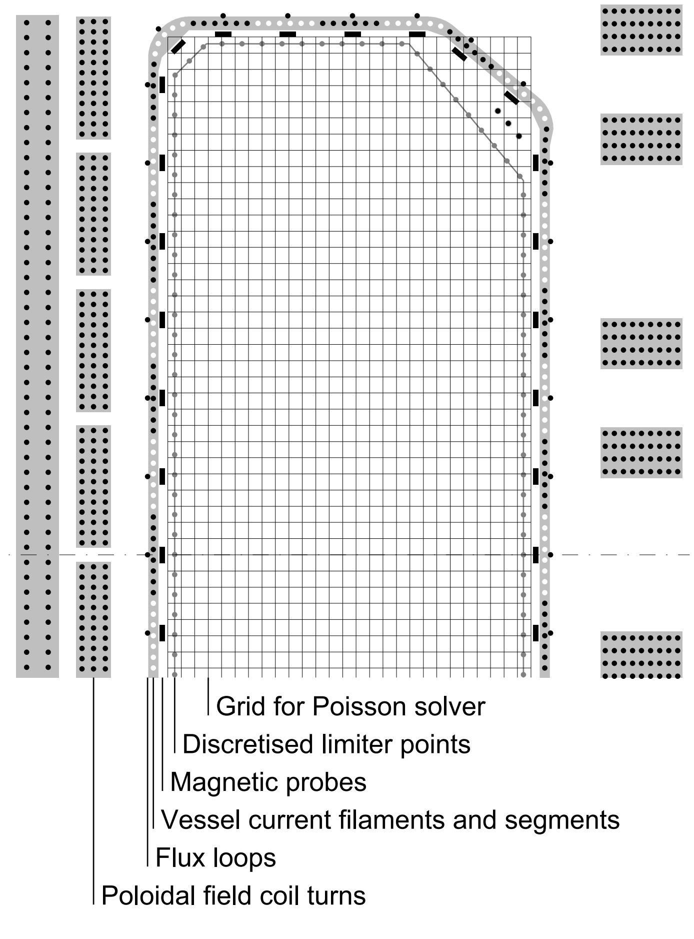
The quantity $\partial_z \psi_{eA}$ is the horizontal magnetic field, applied with external coils, to elongate the plasma shape whose sign changes when crossing the equatorial plane. The quantities $\partial_z^2 \psi_{eA}$ and $\partial_z^2 \psi_{0A}$ are of opposite sign, so the growth factor in equation (14) exceeds one. For a high elongation equilibrium in TCV, the value is close to 2 and illustrates why the Picard iteration scheme is vertically unstable.

To counteract this instability, it is argued that the set of measurements used to constrain the solutions for p' and TT' contain enough information to determine the equilibrium vertical position. A vertical shift in the flux distribution introduced by the algorithm can be legitimately cancelled by adding a free parameter δz determined from these measurements and by then seeking for solutions of the form $p'(\psi(r, z + \delta z))$ and $TT'(\psi(r, z + \delta z))$. This approach was also adopted by both EFIT [11] and the fast version of CLISTE [12].

2.3 Plasma domain

The functions p' and TT' that form the plasma current j_φ must restrict that current to the confinement region where the magnetic field lines are closed. This is a region of nested flux surfaces given by contours of the flux distribution $\psi(r, z)$ with a flux value decreasing (increasing)[1] from the magnetic axis to the plasma edge. This region is, itself, physically restricted to the toroidally projected, and therefore axisymmetric, aperture of the device given by the contour $\partial\Omega_l \subset \Omega$ (see Fig. 1). The most extended closed flux surface thus has a flux value ψ_L equal to the maximum (minimum) flux value along the aperture contour, attained at point (r_L, z_L).

Fig. 1 Geometry for the implementation of LIUQE on TCV (only the top half is shown, the dash-dotted line indicates the symmetry plane): black dots in the poloidal field coils (not all displayed) show the coil discretisation in individual turns (see Section 3.1); black dots just outside the vacuum vessel show the ideal position of the flux loops (see Section 3.2); black rectangles just inside the vacuum vessel show the field probe volume (see Section 3.3); dots in the vacuum vessel width show the vessel current filament positions, white and black colours depict the vessel segments (see Section 3.4); dots on the tile contour show the position of the discretised limiter points (see Sections 3.2 and 4.3); the grid is that used for the Poisson solver (see Section 4.1); the three circular cross section conductors in the outer vessel corner are part of the in vessel coil (see Section 4.1).


Grid for Poisson solver
Discretised limiter points
Magnetic probes
Vessel current filaments and segments
Flux loops
Poloidal field coil turns

The situation is somewhat more complicated in the presence of one or more so called X points, defined as positions (r_X, z_X) where ψ presents a saddle point, i.e. $\nabla\psi = 0$ and $det\ H(\psi) < 0$. Each of these points locally delimits two domains where the flux is larger (smaller) than $\psi(r_X, z_X)$. One of these contains the magnetic axis and therefore the plasma. The

[1] In this section, statements are expressed for a positive total plasma current I_p; wording for a negative plasma current are given in parenthesis.

other may contain a subset of the aperture contour so that the point (r_L, z_L) may not be on the plasma boundary. To handle this situation, for each X point, two simpler domains are defined, separated by a line perpendicular to $\overline{(r_A, z_A)(r_X, z_X)}$ and including the X point (see Fig. 2). One of these domains, $\omega_X(r_X, z_X)$, contains the magnetic axis and the plasma. One practical implementation may be written as

$$\omega_X(r_X, z_X) = \{(r, z) \in \Omega | (r_A - r_X)(r - r_X) + (z_A - z_X)(z - z_X) \geq 0\}. \tag{15}$$

For more than one simultaneous X point, the plasma is restricted to lie in the, possibly open, polygon defined by the intersection of all these domains (see Fig. 2):

$$\Omega_X = \bigcap_X \omega_X(r_X, z_X). \tag{16}$$

Although this may sometimes be over restrictive, it is suitable for convex plasma shapes. The search for (r_L, z_L) is consequently restricted to $\partial\Omega_l \cap \Omega_X$.

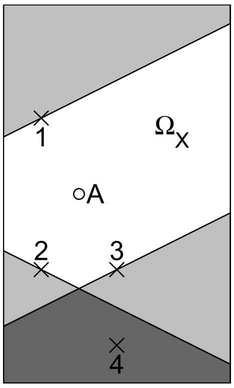


Fig. 2 Sketch of the determination of the domain Ω_X (see Section 2.3); A labels the magnetic axis; crosses show the X points; X point number 4 can be discarded from the potential active X points since it does not lie on the domain boundary.

The case of a diverted plasma, where the last closed flux surface contains an X point, must also be considered. This is characterised by a flux at the X point being larger (smaller) than ψ_L. For both limited and diverted cases, the flux value on the last closed flux surface ψ_0 is the maximum (minimum) flux value in the set $(\partial\Omega_l \cup \{r_X, z_X\}) \cap \Omega_X$, explicitly excluding X points outside Ω_X (see Fig. 3). Finally, as stated above, in the confinement region the flux is larger (smaller) than ψ_0; this defines the plasma domain Ω_p as the subset of Ω_X where the flux $\psi(r, z)$ is larger (smaller) than ψ_0.

2.4 Source function parametrisation

The MHD equilibrium and the derived Grad-Shafranov equation do not specify the arbitrary functions p' and TT'. The second step of the iterative algorithm for solving the inverse equilibrium problem consists in identifying such functions that best reproduce the available measurements. This is performed by first parametrising p' and TT':

$$\begin{aligned} p' &= g_p(\psi; a_p) \\ TT' &= g_T(\psi; a_T) \end{aligned} \tag{17}$$

with parameters a_p and a_T that are adjusted to match the measurements according to a chosen fitting criterion. The choice for the parametrised functions g_p and g_T is guided by the need to cover the range of possible realistic equilibria whilst retaining the capability, from the available measurement set, to unambiguously determine these parameters. This general form for g_p and g_T yields a non linear minimisation problem, with the usual drawbacks. This is the approach adopted for the slow version of CLISTE [12].

It would then be favourable to restrict the choice of the parametrised functions to linear combinations of functions $g_g(\psi)$ that depend on ψ only, the coefficients of the linear combination a_g becoming the free parameters:

$$j_\varphi = 2\pi\left(rp' + \frac{TT}{\mu_0 r}\right) = \sum_g a_g r^{\nu_g} g_g(\psi) . \tag{18}$$

Here $\nu_g = 1$ for the terms contributing to p' and $\nu_g = -1$ for those contributing to TT, and g_g is set to 0 outside Ω_p. This parametrisation, together with a quadratic cost function and measurements which have a strict - or can be approximated with a - linear relationship with p' and TT, leads to a linear regression problem, i.e. safe and fast to solve.

Finally, a free vertical shift in the flux distribution to stabilise the algorithm is introduced (see Section 2.2): $j_\varphi = \sum_g a_g r^{\nu_g} g_g(\psi(r, z + \delta z))$. This violates the Grad-Shafranov equation and can be retained only if the algorithm converges towards a negligible value of δz. In practise, it is observed to stay in the 0.1mm range, approaching 1mm only for highly elongated equilibria. This also destroys the linearity in the free parameters, now the set $\{a_g, \delta z\}$, calling for a linearisation to preserve computation efficiency. As a first step, a first order expansion in δz is taken:

$$j_\varphi \cong \sum_g a_g r^{\nu_g} g_g(\psi + \partial_z \psi \delta z) \cong \sum_g a_g r^{\nu_g} (g_g(\psi) + \delta z g'_g(\psi) \partial_z \psi) . \tag{19}$$

This, however, is still non linear in the free parameters. A work around consists of replacing, in the δz term, the coefficients of the base functions a_g by their approximate value obtained at the previous Picard iteration:

$$j_\varphi^{(t)} = \sum_g a_g^{(t)} r^{\nu_g} g_g(\psi^{(t)}) + \delta z^{(t)} \sum_g a_g^{(t-1)} r^{\nu_g} g'_g(\psi^{(t)}) \partial_z \psi^{(t)} . \tag{20}$$

If, with the same degree of approximation, the flux distribution from the previous iteration is also used, one obtains a simpler expression:

$$j_\varphi^{(t)} = \sum_g a_g^{(t)} r^{\nu_g} g_g(\psi^{(t)}) + \delta z^{(t)} \sum_g a_g^{(t-1)} r^{\nu_g} g'_g(\psi^{(t-1)}) \partial_z \psi^{(t-1)} = \quad . \tag{21}$$
$$\sum_g a_g^{(t)} r^{\nu_g} g_g(\psi^{(t)}) + \delta z^{(t)} \partial_z j_\varphi^{(t-1)}$$

It is shown in Section 3 that, with this expression, the contribution of a δz term to the measurements is directly derivable. Once the free parameters $\{a_g, \delta z\}$ have been identified, the corresponding current distribution is evaluated. It requires a calculation of $\partial_z j_\varphi$ that is somewhat cumbersome as j_φ is truncated at the plasma boundary. More simply, one can calculate the unshifted current distribution

$$j_{\varphi+}^{(t)}(r, z_+) = \sum_g a_g^{(t)} r^{\nu_g} g_g(\psi^{(t)}(r, z_+)) \tag{22}$$

in the new vertical coordinate $z_+ = z + \delta z$ and then solve the Poisson equation for the next iteration in the same domain:

$$\Delta^* \psi_+^{(t+1)}(r, z_+) = -2\pi\mu_0 r(j_{\varphi+}^{(t)}(r, z_+) + j_{e+}(r, z_+)), (r, z_+) \in \Omega \tag{23}$$

where $j_{e+}(r, z_+) = j_e(r, z_+ - \delta z)$ is the external current density shifted in the new vertical coordinate.

The contribution from the plasma current to the boundary condition is

$$\psi_{p+}(r, z_+) = \int M(r, z_+, r', z_+')j_{\varphi+}(r', z_+')dr'dz_+' . \tag{24}$$

The contribution from the external current requires some care; it may be written as

$$\begin{aligned} &\psi_{e+}(r, z_+) = \\ &\int M(r, z_+, r', z_+')j_{e+}(r', z_+')dr'dz_+' = \\ &\int M(r, z_+, r', z_+')j_e(r', z_+' - \delta z)dr'dz_+' = \\ &\int M(r, z_+, r', z_+' + \delta z)j_e(r', z_+')dr'dz_+' \end{aligned} \tag{25}$$

which can be approximated by

$$\psi_{e+}(r, z_+) = \int M(r, z_+, r', z_+')j_e(r', z_+')dr'dz_+' - \delta z \int \partial_{z_+} M(r, z_+, r', z_+')j_e(r', z_+')dr'dz_+' . \tag{26}$$

In summary, the same resolution procedure can be applied using the unshifted plasma current density, except for a shift in the external current density in the source term and a correction in the contribution from the external currents to the boundary condition. The obtained solution $\psi_+(r, z_+)$ is finally used to update the flux function:

$$\psi^{(t+1)}(r, z) = \psi_+(r, z + \delta z) \cong \psi_+(r, z) + \delta z \partial_z \psi_+(r, z) . \tag{27}$$

The advantage lies in the fact that the flux function, unlike the plasma current density, is differentiable everywhere.

3 Measurements

This section lists the set of available measurements on TCV and gives their response, or their linearised approximation, to the base functions of the source terms, to the vertical shift of the flux distribution, as well as their response to external currents, when relevant. A detailed description of all the magnetic measurements can be found in [13]. Although specific to TCV, this section illustrates how the code may readily be adapted to another Tokamak with a different measurement set and layout.

3.1 Active coil currents

The current distribution in the active coils is modelled by dividing each coil in its individual turns with a filament current of amplitude I_a flowing at its center (see Fig. 1). The spacing between turns is 1 to 3 cm, providing sufficient precision for the equilibrium reconstruction.

3.2 Flux loops

A set of 38 flux loops, mounted on the vacuum vessel, measure the poloidal flux (see Fig. 1). Some of the loops do not follow a toroidally axisymmetric path as they must circumnavigate the port holes. This is accounted for by using linear combinations of the raw flux measurements, as explained in [13]. To remove a large net flux from the ohmic transformer, differences, with respect to a selected reference loop, are preferred. The response to the source terms, to the active coil currents, M_{fa}, and that to the vessel currents, introduced in Section 3.4, M_{fv}, are calculated using the mutual inductance formula given in Appendix A for ideal loops at positions (r_f, z_f). The contribution from the vertical shift δz may be written as

$$\psi_f = \delta z \int M(r_f, z_f, r, z) \partial_z j_\varphi(r, z) dr dz \tag{28}$$

which can be exactly replaced by

$$\psi_f = \delta z \int \partial_{z_f} M(r_f, z_f, r, z) j_\varphi(r, z) dr dz . \tag{29}$$

This expression is advantageous as it replaces an undesirable calculation of $\partial_z j_\varphi$ by an analytical differentiation of M, also explicitly reported in Appendix A.

3.3 Magnetic probes

Two poloidal arrays of 38 magnetic probes each are used for the equilibrium reconstruction (see Fig. 1). They are located 180° apart toroidally and summed to cancel any contribution from MHD modes with a toroidal number of 1. The response to the source terms, to the active coil currents, B_{ma} and that to the vessel currents, B_{mv} are calculated using the magnetic field formulae given in Appendix A with an average over the probe volume, which improves the accuracy of the response for current sources located near the probe by about 1%. The contribution from the vertical shift δz follows an argument similar to that for the flux loops.

3.4 Vacuum vessel currents

The TCV vacuum vessel has continuous weld resulting in a low toroidal resistance (45 μΩ) designed for passive stabilisation of vertical motion of the plasma column. A toroidal vessel current of 20 kA per Volt of loop voltage is induced; this is a sizable fraction of the plasma current (25 kA to 1 MA) and its influence on the equilibrium must be accounted for.

The vacuum vessel is made of stainless steel of 15 mm thickness in the cylindrical parts and 20 mm at the top and bottom. Port holes in the vessel make its electromagnetic modelling difficult. Nevertheless, instead of a complicated 3-D representation, a toroidally symmetric model with an ad hoc spatial distribution of the electrical resistivity to model the port holes was chosen. Since the vessel thickness is much smaller than the machine dimensions, the current density is modelled with a set of $n_v = 256$ current filaments evenly spaced along the poloidal perimeter of the vacuum vessel at locations (r_v, z_v) (see Fig. 1).

This geometrical description of the vessel currents is completed with a voltage equation for the current filaments I_v, that, without plasma, may be written

$$0 = R_v I_v + M_{vv'} \cdot \frac{dI_{v'}}{dt} + M_{va} \cdot \frac{dI_a}{dt}. \tag{30}$$

Here R_v is the filament resistance and M_{va} and $M_{vv'}$ the mutual inductance matrices calculated with the formulas from Appendix A. The singularity for the case with $v = v'$ is handled using $M_{vv} = \mu_0 r_v\left(ln\frac{8r_v}{w_v} - \frac{1}{2}\right)$ where w_v is the local vessel thickness plus the distance between two adjacent filaments. This approximation is sufficient because the contribution from the diagonal elements in M_{vv} is inversely proportional to the number of filaments.

The basic idea in setting up an observer for the current distribution in the vacuum vessel is to use the n_f flux loops distributed on the vacuum vessel to provide a measurement of the spatial distribution of the inductive voltage applied to the vacuum vessel. The vacuum vessel current filaments are first grouped in $n_s = n_f$ segments: the current filament v belongs to the segment s if the corresponding flux loop $f = s$ is the closest flux loop to this filament (see Fig. 1); with the use of Iverson brackets, this can be cast as elements of a rectangular matrix with $T_{sv} = [v \in s]$. The proximity of the flux loop f to the current filaments that belong to the segment $s = f$ means that the voltage on that flux loop

$$U_{f=s} = M_{f=s,v} \cdot \frac{dI_v}{dt} + M_{f=s,a} \cdot \frac{dI_a}{dt} \tag{31}$$

is close to the inductive voltage applied to those vessel current filaments, so that their voltage equation (30) may be written as

$$R_{v \in s} I_v = -M_{v \in s, v'} \cdot \frac{dI_{v'}}{dt} - M_{v \in s, a} \cdot \frac{dI_a}{dt} \cong -M_{f=s,v} \cdot \frac{dI_v}{dt} - M_{f=s,a} \cdot \frac{dI_a}{dt} = -U_{f=s}. \tag{32}$$

An estimator is finally obtained for the vessel filament currents:

$$I_{v \in s} = -R^{-1}_{v \in s} U_{f=s}. \tag{33}$$

The result holds if the plasma current is taken into account in equations (31) and (32).

The only independent estimators are the total currents in the segments that can be derived from U_f as

$$I_s = \sum_{v \in s} I_v = -\sum_{v \in s} R_v^{-1} U_{f=s} = -R_s^{-1} U_{f=s} \tag{34}$$

with $R_s^{-1} = \sum_{v \in s} R_v^{-1}$. The inverse relation is derived by combining equations (33) and (34):

$$I_{v \in s} = R_{v \in s}^{-1} R_s I_s \tag{35}$$

which can be cast in the elements of a rectangular matrix with $T_{vs} = R_v^{-1} R_s [v \in s]$ and used to express the sensitivity of the flux loops to the segment currents, i.e. $M_{fs} = M_{fv} \cdot T_{vs}$ and that of the magnetic probes, $B_{ms} = B_{mv} \cdot T_{vs}$.

Note that the time evolution for this estimator of the vessel currents does not necessarily follow the electromagnetic model of the vacuum vessel contained in equation (30). To improve this situation an estimator based on a Kálmán filter, as proposed in [14], or an approach including such a model, similar to that reported in [15], are envisaged.

The effective electrical resistance of each vessel current filament required to estimate vessel segment currents using equation (33) is experimentally determined as follows [16]. Trapezoidal current waveforms with ramps and flat-top considerably longer than the vessel characteristic time constant (13 ms) are separately applied to each active coil. During the ramps, I_v and dI_a/dt are constant and from equation (30) the vessel currents are derived:

$$I_v = -R_v^{-1} M_{va} \cdot \frac{dI_a}{dt}. \tag{36}$$

The flux and magnetic field measured by the magnetic sensors are

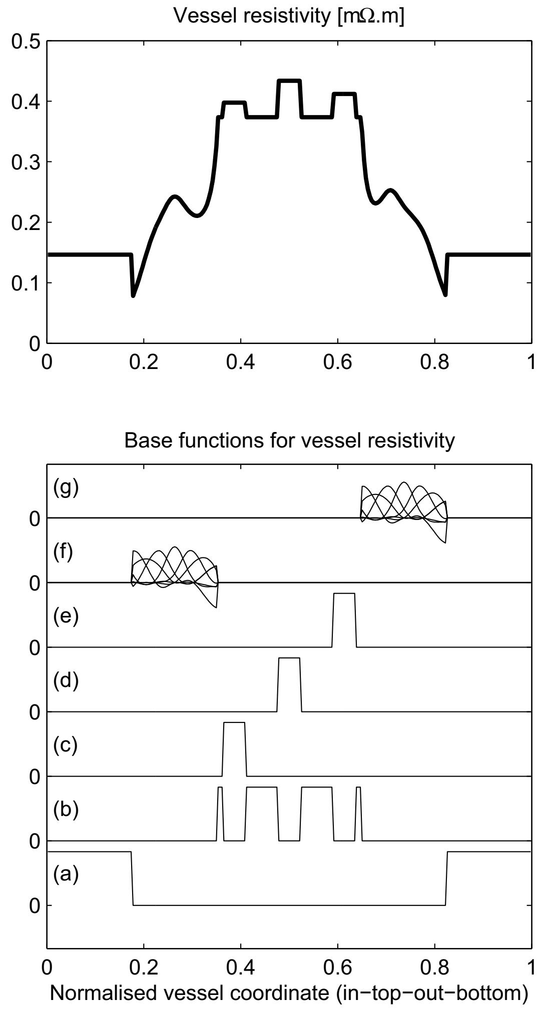
$$\begin{aligned} \psi_f &= M_{fa} \cdot I_a + M_{fv} \cdot I_v \\ B_m &= B_{ma} \cdot I_a + B_{mv} \cdot I_v \end{aligned}. \tag{37}$$

The mutual inductance matrices depend only on geometry and are assumed exactly known. The vessel filament resistance can be deduced from these measurements by solving for R_v^{-1}, in a least square sense; the equation set is

$$\begin{cases} \psi_f - M_{fa} \cdot I_a = M_{fv} \cdot I_v = -M_{fv} \cdot R_v^{-1} M_{va} \cdot \dfrac{dI_a}{dt} \\ B_m - B_{ma} \cdot I_a = B_{mv} \cdot I_v = -B_{mv} \cdot R_v^{-1} M_{va} \cdot \dfrac{dI_a}{dt} \end{cases}. \tag{38}$$

Each individual equation is multiplied by a weight related to the inverse of the measurement precision. The differences in the left hand side are readily calculated using the flux and field measured during the current flat-top, where $I_v = 0$, $\psi_f = M_{fa} \cdot I_a$ and $B_m = B_{ma} \cdot I_a$. The available data allows for the determination of only a number of parameters smaller than n_v. Thus the resistivity distribution along the vessel perimeter is obtained from a linear combination of $n_w \le n_f + n_m < n_v$ base functions that, effectively, describe the port hole geometry. The elements of the rectangular matrix T_{vw} represent the amplitude of the base function w at filament location v; they are shown in Fig. 3. In the equation set (38), R_v^{-1} is first replaced by $T_{vw} \cdot R_w^{-1}$, then the equation set solved for R_w^{-1}, and finally the filament resistance calculated using $R_v^{-1} = T_{vw} \cdot R_w^{-1}$, also shown in Fig. 3.

Fig. 3 Experimentally derived linear vessel resistivity (bold line) and the base functions used to derive it (see Section 3.4): base function(s) for (a) the inner cylinder, (b) the part of the outer cylinder without ports, (c) the top row of outer ports, (d) the equatorial row of outer ports, (e) the bottom row of outer ports, (f) the top part (7 functions), (g) bottom part (7 functions). The geometry for the parts (a) to (e) is simple and constant base functions are chosen; for parts (f) and (g), the geometry of the ports is complex and 7 base functions are used to allow for the resistivity to vary along the vessel coordinate. The x axis runs clockwise from the equatorial inner point.



3.5 Plasma current

There is no continuous Rogowski coil installed around the plasma column, so the total plasma current is estimated with a discrete integral based on the n_m magnetic probe measurements [13]. Using a trapeze approximation and a polygonal integration contour defined by the probe locations (r_m, z_m), this estimator may be written in matricial form $I_p = I_{pm} \cdot B_m$; the formula for the coefficients I_{pm} is given in Appendix B. This estimator is sensitive to I_a with a factor $I_{pa} = I_{pm} \cdot B_{ma}$ and to I_s with a factor $I_{ps} = I_{pm} \cdot B_{ms}$. So the final plasma current estimator is corrected for these sensitivities: $I_p = I_{pm} \cdot B_m - I_{pa} \cdot I_a - I_{ps} \cdot I_s$. Although this estimator is simply a combination of used measurements, it is retained as a separate measurement and given an appropriate weight. Its relation to the plasma current density is trivial:

$$I_p = \int j_\varphi(r, z) drdz = \sum_g a_g \int r^{\nu_g} g_g(\psi(r, z)) drdz. \qquad (39)$$

3.6 Diamagnetic flux loop

The diamagnetic flux loop measures the contribution of the plasma to the toroidal flux integrated over its poloidal cross-section, Φ_t. A detailed description of the measurement was given in [17]. This flux can be expressed, using the parametrisation of the magnetic field given by equation (2) and the value of the vacuum toroidal magnetic field imposed by the axisymmetry, as

$$\Phi_t = \int \frac{T - r_0 B_{\varphi 0}}{r} drdz \qquad (40)$$

where $B_{\varphi 0}$ is the vacuum toroidal field at radius r_0. Since the source function in the Grad-Shafranov equation is proportional to TT', the relationship between Φ_t and the source terms is not linear. Writing $T^2 = r_0^2 B_{\varphi 0}^2 + F^2$ where $F^2/2$ is a primitive of TT', with an integration constant selected to satisfy $F = 0$ outside the plasma, and in the small diamagnetism approximation where $F \ll r_0 |B_{\varphi 0}|$, one can write

$$T = \sqrt{r_0^2 B_{\varphi 0}^2 + F^2} \cong r_0 B_{\varphi 0} + \frac{F^2}{2 r_0 B_{\varphi 0}}. \tag{41}$$

This approximation introduced in equation (40) and the source function parametrisation (18) yield a linear response of the diamagnetic flux to the source terms:

$$\Phi_t = T_{tg} \cdot a_g = \sum_g a_g \left(\frac{\mu_0}{2\pi r_0 B_{\varphi 0}} [\nu_g = -1] \int \frac{G_g(\psi(r, z))}{r} dr dz \right) \tag{42}$$

where $G_g(\psi)$ is a primitive of $g_g(\psi)$ with $G_g = 0$ outside the plasma. The diamagnetic measurement, because it is sensitive only to the TT' function, is effective for disentangling the known degeneracy between the rp' and TT'/r terms at large aspect ratio.

3.7 Optional measurements

Local measurements of the plasma temperature and density can provide total pressure at given points $p(r_k, z_k)$. From equation (18) the response of this measurements to the source terms is $\frac{1}{2\pi}[\nu_g = 1] G_g(\psi(r_k, z_k))$. Since they are sensitive only to the p' terms, these measurements are effective, as with the diamagnetic flux, in separating rp' and TT'/r terms, and in constraining the p' profile.

It may be desirable, in some circumstances, to specify the value of the safety factor on the magnetic axis, q_A. It is shown in Section 4.7 that this is related to the equilibrium and the central plasma current density:

$$q_A^{-1} = \frac{\sqrt{det\ H(\psi_A)}}{|tr\ H(\psi_A)|} \frac{\mu_0 r_A^2 j_{\varphi A}}{T_A}. \tag{43}$$

Intrinsically this is non linear in the source function T. However it can be made linear in $j_{\varphi A}$ by fixing T_A to its value obtained at the previous Picard iteration.

Information on the flux surface shape is sometimes available from spatially resolved measurement of the plasma emissivity, typically in the soft X-ray or visible wave length ranges [18]. If this information is given as a set of points (r_o, z_o) lying on the same flux surface or on a specified flux surface ψ_o, the flux at these points can be evaluated using the mutual inductance formulae in Appendix A. Then, either the flux difference between each point and another reference point can be set to 0, or the absolute flux set to ψ_o.

As reported in [3], LIUQE also has provision to incorporate measurements from Faraday rotation when they become available.

4 Algorithm details

4.1 Grid and Poisson solver

The almost rectangular geometry of the TCV vacuum vessel makes the choice of a rectangular integration domain Ω straightforward: to satisfy the conditions (9), the boundary is placed between the vacuum vessel and the tile aperture (see Fig. 1). Although a rectangular grid perfectly fits the TCV vacuum vessel, it must be emphasised that most of the time the plasma occupies only a fraction of the grid. To use LIUQE for a D-shaped or circular vacuum vessel, the rectangular grid must be chosen such that it contains the limiter contour (see Section 2.3); a similar filling ratio would thus be obtained.

This domain is meshed with n_{rx} and n_{zx} points in the r direction and z direction respectively, with an even spacing Δr and Δz; usually for TCV $n_{rx} = 28$ and $n_{zx} = 65$. Two widely used subdomains are defined as that of the boundary points (r_b, z_b) and the inner grid (r_y, z_y) formed by the points not on the domain boundary. External currents falling in Ω must be added to the right hand side of the Poisson equation (6). For TCV these are the vessel filaments in the corners of the vacuum vessel and the 6 turn in-vessel coils (see Fig. 1). These provide a relatively small contribution to the poloidal flux and can be safely assigned to their closest mesh points not on the domain boundary. Using the same argument, the vertical shift, which must be introduced in the same external currents before solving equation (23), is neglected, as it is much smaller than Δz.

The Poisson equation (6) is written on the computational grid in finite difference form:

$$\psi_{i,j+1} + \psi_{i,j-1} + a_i\psi_{i+1,j} + b_i\psi_{i-1,j} - c_i\psi_{ij} = -d_{ij} \qquad (44)$$

where i and j label the radial and vertical position on the computational grid and where the coefficients a_i, b_i and c_i are selected between various differentiation methods by their achieved numerical accuracy [19]:

$$\begin{aligned} a_i &= \left(\frac{\Delta z}{\Delta r}\right)^2 \frac{r_i}{r_i + \Delta r/2} \\ b_i &= \left(\frac{\Delta z}{\Delta r}\right)^2 \frac{r_i}{r_i - \Delta r/2} \\ c_i &= 2 + a_i + b_i \\ d_{ij} &= 2\pi\mu_0 \Delta z^2 r_i (j_\varphi(r_i, z_j) + j_e(r_i, z_j)) \end{aligned} \qquad . \qquad (45)$$

Equation (44) is solved by a direct algorithm [19], using cyclic reduction stabilised with the Buneman method [20] in the vertical direction, coupled with a tridiagonal matrix inversion in the radial direction. This method is as fast as the double cyclic reduction algorithm and does not restrict the number of grid cells in the radial direction to powers of two.

4.2 Boundary condition

The discretisation on the computational grid of the boundary condition for the Poisson equation as expressed by equations (7) and (8) takes the form:

$$\psi_b = M_{by} \cdot I_y + M_{ba} \cdot I_a + M_{bs} \cdot I_s . \qquad (46)$$

The calculation of the contribution from the plasma current $M_{by} \cdot I_y$, with $I_y = j_\varphi(r_y, z_y)\Delta r \Delta z$ consists of a matrix vector multiplication with size 182 by 1638 for TCV. It is computationally demanding but can be replaced by a method reported in [10]. A surface current j^* flowing on the domain boundary $\partial\Omega$ is superimposed onto the plasma current density and is chosen to cancel the flux at the boundary:

$$\psi^*(r, z) = \int_\Omega M(r, z, r', z') j_\varphi(r', z') dr' dz' + \int_{\partial\Omega} M(r, z, r'(s'), z'(s')) j^*(r'(s'), z'(s')) ds' = 0 \text{ on } \partial\Omega . \qquad (47)$$

where s is a length coordinate along the contour $\partial\Omega$. The corresponding Poisson equation becomes

$$\Delta^*\psi^* = -2\pi\mu_0 r j_\varphi \text{ in } \Omega - \partial\Omega , \qquad (48)$$

$$\nabla_n \psi^* = -2\pi\mu_0 r j^* \text{ on } \partial\Omega . \qquad (49)$$

The second equation is obtained by Lebesgue integration through $\partial\Omega$, with ∇_n the gradient in the inward direction normal to $\partial\Omega$. The value for ψ^* is obtained by solving $\Delta^*\psi^* = -2\pi\mu_0 j_\varphi$ with the boundary condition $\psi^* = 0$. Combining equations (47) and (49) yields the contribution of the plasma current density to the boundary condition:

$$\psi_p(r, z) = \int_\Omega M(r, z, r', z') j_\varphi(r', z') dr' dz' = -\oint_{\partial\Omega} M(r, z, r', z') \frac{\nabla_n \psi^*}{2\pi\mu_0 r'} ds' \text{ on } \partial\Omega . \qquad (50)$$

The double surface integral has been advantageously replaced by a contour integral, at the cost of the resolution of a Poisson equation.

In discretising this contour integral, the logarithmic divergence of $M(r, z, r', z')$, when (r, z) and (r', z') coincide, must be treated with care; the particularly accurate Weinstein’s formula [21] is used:

$$M(r, z, r, z) = \mu_0 r \left(\left(1 + 2\left(\frac{\Delta z}{8r}\right)^2 + \frac{2}{3}\left(\frac{\Delta r}{8r}\right)^2 \right) ln \frac{8r}{\Delta r + \Delta z} - \frac{1}{2} + \frac{1}{2}\left(\frac{\Delta z}{8r}\right)^2 \right) \qquad (51)$$

where Δr or Δz is set to zero at the vertical or horizontal boundaries respectively. The normal gradient is computed using a second order backward or forward finite difference, keeping in mind that $\psi^*(r_b, z_b) = 0$:

$$\nabla_n \psi^*(r_b, z_b) \Delta s = \frac{\Delta z}{\Delta r} \left(2\psi^*(r_b + \Delta r, z_b) - \frac{1}{2}\psi^*(r_b + 2\Delta r, z_b) \right) \qquad (52)$$

for the left boundary side, and similarly for the other sides. It is ill defined at the boundary corners where it is set to zero.

4.3 Identification of the plasma domain

The identification of the plasma domain, as formally introduced in Section 2.3, first requires the magnetic axis where the flux distribution has an extremum. The discrete flux function is thus interpolated near each inner grid point with a 6 point interpolation:

$$\psi(r_y + dr, z_y + dz) = \psi(r_y, z_y) + a\frac{dr}{\Delta r} + b\frac{dz}{\Delta z} + \frac{c}{2}\left(\frac{dr}{\Delta r}\right)^2 + \frac{d}{2}\left(\frac{dz}{\Delta z}\right)^2 + e\frac{dr}{\Delta r}\frac{dz}{\Delta z} \qquad (53)$$

with

$$\begin{aligned} a &= \frac{1}{2}(\psi(r_y + dr, z_y) - \psi(r_y - dr, z_y)) \\ b &= \frac{1}{2}(\psi(r_y, z_y + dz) - \psi(r_y, z_y - dz)) \\ c &= -2\psi(r_y, z_y) + \psi(r_y + dr, z_y) + \psi(r_y - dr, z_y) \\ d &= -2\psi(r_y, z_y) + \psi(r_y, z_y + dz) + \psi(r_y, z_y - dz) \\ e &= \psi(r_y, z_y) - \psi(r_y + dr, z_y) + \psi(r_y + dr, z_y + dz) - \psi(r_y, z_y + dz) \end{aligned} \quad . \qquad (54)$$

Then the points where $\nabla\psi = 0$ are sought that satisfy

$$\begin{cases} \dfrac{dr}{\Delta r} = \dfrac{eb - ad}{cd - e^2} \\ \dfrac{dz}{\Delta z} = \dfrac{ea - bc}{cd - e^2} \end{cases} \qquad (55)$$

with $\frac{|dr|}{\Delta r} < 1$ and $\frac{|dz|}{\Delta z} < 1$ to retain only the points in the grid cells where interpolation is valid. This interpolation also provides the Hessian matrix used in many of the specific calculations in the inverse equilibrium code:

$$H(\psi) = \begin{pmatrix} \dfrac{c}{\Delta r^2} & \dfrac{e}{\Delta r \Delta z} \\ \dfrac{e}{\Delta r \Delta z} & \dfrac{d}{\Delta z^2} \end{pmatrix} . \qquad (56)$$

The magnetic axis is the point where ψ has an extremum, i.e. $\nabla\psi = 0$ and $det\ H > 0$. Similarly the X points are those with $\nabla\psi = 0$ and $det\ H < 0$. Since the interpolation on a grid cell $[r_x, r_x + \Delta r] \times [z_x, z_x + \Delta z]$ is not unique and depends on up to four possible central points, either (r_x, z_x), $(r_x + \Delta r, z_x)$, $(r_x + \Delta r, z_x + \Delta z)$ or $(r_x, z_x + \Delta z)$, this procedure may identify several extrema or saddle points in a single cell. This procedure was chosen because the entire mesh must, in any case, be scanned to search for potential X points; keeping the same procedure for both magnetic axis and X points thus becomes computationally efficient. The first encountered extremum is kept as the magnetic axis. For the non unique X points, those outside Ω_X are automatically discarded by definition (16) for Ω_X.

The identification of the plasma domain also requires the value of the flux at the aperture $\partial\Omega_l$. This is represented by discrete points (r_l, z_l) and the discretised flux is interpolated at those points using a four point interpolation (see Fig. 1).

4.4 Source term fitting

Identifying the free parameters $\{a_g, \delta z\}$ in the parametrisation of the plasma current distribution (21) is the second step in the iterative inverse equilibrium problem algorithm. The most frequent choice for the source term base functions are the three polynomials

$$\nu_1 = 1, g_1 = (\psi - \psi_0)[(r, z) \in \Omega_p] \text{ for } p' \text{ and} \tag{57}$$

$$\left.\begin{array}{l} \nu_2 = -1, g_2 = (\psi - \psi_0)[(r, z) \in \Omega_{mpp}] \\ \nu_3 = -1, g_3 = (\psi - \psi_0)(\psi - \psi_A)[(r, z) \in \Omega_p] \end{array}\right. \text{ for } TT' . \tag{58}$$

This base function set is restricted to only g_2 when the elongation, estimated as the aspect ratio of the bounding box of Ω_p, is below a threshold close to 1 and the separation of the rp' from TT'/r terms becomes increasingly difficult. Other alternatives were occasionally tried, but the question of the choice of the base functions and their number in relation to the set of available measurements, and their error and correlation, is not addressed here. The value of these functions on the inner computational grid is stored in the elements of a rectangular matrix $T_{yg} = r_y^{\nu_g} g_g(\psi(r_y, z_y))\Delta r\Delta z$ so that the expected measurements, using their response describes in Section 3, may be written:

$$\begin{aligned} \psi_f &= M_{fa} \cdot I_a + M_{fs} \cdot I_s + (M_{fy} \cdot T_{yg}) \cdot a_g + (\partial_{z_f} M_{fy} \cdot I_y)\delta z \\ B_m &= B_{ma} \cdot I_a + B_{ms} \cdot I_s + (B_{my} \cdot T_{yg}) \cdot a_g + (\partial_{z_m} B_{my} \cdot I_y)\delta z \\ I_p &= T_{pg} \cdot a_g \\ \Phi_t &= T_{tg} \cdot a_g \end{aligned} \tag{59}$$

where $I_y = j_\varphi^{(t-1)}(r_y, z_y)\Delta r\Delta z$ and $T_{pg} = \sum_y T_{yg}$. The measurements of the active coil currents as well as the observer for the vessel currents contain uncertainties and are included in the fitting procedure as free parameters J_a and J_s respectively. This leads to a system of linear equations:

$$\begin{bmatrix} \Psi_f \\ B_m \\ I_a \\ I_s \\ I_p \\ \Phi_t \end{bmatrix} = \begin{bmatrix} M_{fa} & M_{fs} & M_{fy} \cdot T_{yg} & \partial_{z_f} M_{fy} \cdot I_y \\ B_{ma} & B_{ms} & B_{my} \cdot T_{yg} & \partial_{z_m} B_{my} \cdot I_y \\ 1_a & 0 & 0 & 0 \\ 0 & 1_s & 0 & 0 \\ 0 & 0 & T_{pg} & 0 \\ 0 & 0 & T_{tg} & 0 \end{bmatrix} \cdot \begin{bmatrix} J_a \\ J_s \\ a_g \\ \delta z \end{bmatrix} \tag{60}$$

where 1_α is the identity matrix of size n_α. This system is solved in a least square sense, each equation being given a weight inversely proportional to the associated measurement error. On TCV, the typical uncertainties are

$$\begin{aligned} w_f^{-1} &= 0.75 mWb \\ w_m^{-1} &= 1 mT \\ w_a^{-1} &= 20A \\ w_s^{-1} &= 20A \\ w_p^{-1} &= 2.4kA \\ w_t^{-1} &= 0.01 mWb \end{aligned} \quad . \tag{61}$$

The direct resolution of equation (60), for example by QR factorisation of the implied matrix, requires considerable computation; for TCV the matrix size is 133 by 60. The conjugate gradient method was tried but proved to be unstable. The resolution method can take advantage of the block structure of this matrix; measurements sensitive to both the source terms and the external currents are grouped in Y_r, those only to the source terms in Y_i and those only to the external currents in Y_e; the free parameters are also grouped in external currents J_e and source term parameters a_j. These can be written explicitly:

$$\begin{gathered} Y_r = \begin{bmatrix} w_f \Psi_f \\ w_m B_m \end{bmatrix}, Y_i = \begin{bmatrix} w_p I_p \\ w_t \Phi_t \end{bmatrix}, Y_e = \begin{bmatrix} w_a I_a \\ w_s I_s \end{bmatrix} \\ J_e = \begin{bmatrix} J_a \\ J_s \end{bmatrix}, a_j = \begin{bmatrix} a_g \\ \delta z \end{bmatrix} \end{gathered} \tag{62}$$

A weighted version of equation (60) can then be formed:

$$\begin{bmatrix} Y_r \\ Y_e \\ Y_i \end{bmatrix} = \begin{bmatrix} W_{re} & W_{rj} \\ W_{ee} & 0 \\ 0 & W_{ij} \end{bmatrix} \cdot \begin{bmatrix} J_e \\ a_j \end{bmatrix} \tag{63}$$

with

$$\begin{aligned}
W_{re} &= \begin{bmatrix} w_f M_{fa} & w_f M_{fs} \\ w_m B_{ma} & w_m B_{ms} \end{bmatrix} \\
W_{rj} &= \begin{bmatrix} w_f M_{fy} \cdot T_{yg} & w_f \partial_{z_f} M_{fy} \cdot I_y \\ w_m B_{my} \cdot T_{yg} & w_m \partial_{z_m} B_{my} \cdot I_y \end{bmatrix} \\
W_{ij} &= \begin{bmatrix} w_p T_{pg} & 0 \\ w_t T_{tg} & 0 \end{bmatrix} \\
W_{ee} &= \begin{bmatrix} w_a & 0 \\ 0 & w_s \end{bmatrix}
\end{aligned} . \qquad (64)$$

The equation system (63) can be separated in two subsystems that can be solved iteratively first for $a_j^{(k)}$ and then for $J_e^{(k)}$, seeding the iteration process with $J_e^{(0)} = \begin{bmatrix} I_a & I_s \end{bmatrix}$:

$$\begin{cases} \begin{bmatrix} Y_r - W_{re} \cdot J_e^{(k-1)} \\ Y_i \end{bmatrix} = \begin{bmatrix} W_{rj} \\ W_{ij} \end{bmatrix} \cdot a_j^{(k)} \\ \begin{bmatrix} Y_r - W_{rj} \cdot a_j^{(k)} \\ Y_e \end{bmatrix} = \begin{bmatrix} W_{re} \\ W_{ee} \end{bmatrix} \cdot J_e^{(k)} \end{cases} . \qquad (65)$$

Typically 12 iterations are sufficient to reach a precision close to one less significant digit. The explicit solutions for these two equations are

$$\begin{cases} a_j^{(k)} = (A_{jj}^{-1} \cdot (W_{rj}^t \cdot Y_r + W_{ij}^t \cdot Y_i)) - (A_{jj}^{-1} \cdot W_{rj}^t \cdot [W_{re}]) \cdot J_e^{(k-1)} \\ J_e^{(k)} = ([A_{ee}^{-1} \cdot W_{re}^t] \cdot Y_r + [A_{ee}^{-1} \cdot W_{ee}^t] \cdot Y_e) - ([A_{ee}^{-1} \cdot W_{re}^t] \cdot W_{rj}) \cdot a_j^{(k)} \end{cases} \qquad (66)$$

with $A_{jj} = W_{rj}^t \cdot W_{rj} + W_{ij}^t \cdot W_{ij}$ and $A_{ee} = W_{re}^t \cdot W_{re} + W_{ee}^t \cdot W_{ee}$. Expressions (66) were optimised for computational load: quantities that do not depend on the equilibrium are enclosed in square brackets, those to be calculated at each Picard iteration in parenthesis and those changing at each iteration k marked with a corresponding superscript. The inverse of the 4 by 4 symmetrical positive definite matrix A_{jj} is computed with an unrolled LDL decomposition that does not require, unlike the Cholesky decomposition, the slow square root function, followed by an unrolled forward and back substitution. This method also correctly handles the case where one or more base functions is not included and the corresponding elements in A_{jj}^{-1} are set to zero.

The computational load in the source term fitting step of the inverse equilibrium problem is strongly weighted by the calculation of the W_{rj} matrix in equation (64), because the base functions of the source term, and therefore T_{yg} and I_y, change at every Picard iteration. This calculation is bypassed in the fast version of CLISTE [12] by independently, and in parallel, solving the Poisson equations for each base functions to obtain the corresponding flux distributions; the response of most of the measurements to the base functions is then approximated by a simple spatial interpolation and differentiation on these flux distributions. In the case of TCV, and with the achieved computation time in the involved operations (see Section 5.3), this approach would not yield any advantage and the exact approach is preferred.

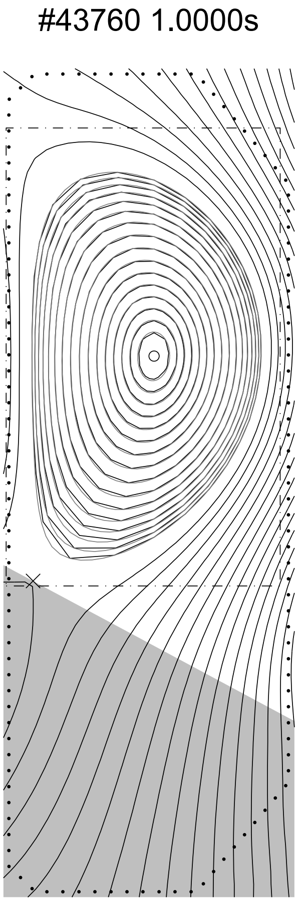


Fig. 4 LIUQE equilibrium reconstruction: black dots show the discretised limiter points; the circle shows the position of the magnetic axis; the cross shows the position of the active X point; the closed flux surfaces in black are obtained with the real time contouring algorithm (see Section 4.7); for comparison the flux surfaces in grey are obtained with the Ψ-toolbox; the open flux surfaces are drawn using a four point interpolation; the grey region shows the domain outside Ω_X (see Section 2.3); the dash-dotted rectangle is that used for the initial guess for the plasma current density (see Section Fig. 4).

4.5 Initial guess for plasma current density

An initial guess for the plasma current density $j_\varphi^{(0)}$ is needed to seed the Picard iterations. This is obtained from the measurements by decomposing the plasma current density into typically $m_h = 2$ radially by $n_h = 3$ to 4 vertically bilinear finite elements distributed on a rectangle whose boundaries are set on the basis of the pre programmed equilibria defined during shot preparation (see Fig. 4). Storing the value, explicitly given in Appendix C, of the finite elements on the inner grid points in the matrix T_{yh} with the index h spanning the $m_h \times n_h$ finite elements enables the current density to be written as a linear combination with coefficients a_h:

$$j_\varphi^{(0)}(r_y, z_y)\Delta r\Delta z = I_y = T_{yh} \cdot a_h . \qquad (67)$$

For such a combination, the expected measurements yield a set of equations similar to (60):

$$\begin{bmatrix} \Psi_f \\ B_m \\ I_p \\ I_a \\ I_s \end{bmatrix} = \begin{bmatrix} M_{fa} & M_{fs} & M_{fy} \cdot T_{yh} \\ B_{ma} & B_{ms} & B_{my} \cdot T_{yh} \\ 0 & 0 & \sum_y T_{yh} \\ 1_a & 0 & 0 \\ 0 & 1_s & 0 \end{bmatrix} \cdot \begin{bmatrix} J_a \\ J_s \\ a_h \end{bmatrix} \qquad (68)$$

to be similarly solved for a_h. This operation can be cast as a matrix multiplication of the measurement vector. The matrix, depending only on the geometry, may thus be precalculated.

The contribution of this plasma current density to the boundary condition is directly calculated as

$$\psi_b = M_{by} \cdot I_y = (M_{by} \cdot T_{yh}) \cdot a_h \tag{69}$$

where $M_{by} \cdot T_{yh}$ is also precalculated.

4.6 Convergence

Iteration convergence is practically reached when the update in the poloidal flux on the inner grid points normalised by $|\psi_A - \psi_0|$ is below 5×10^{-5}. This usually requires less than 10 iterations.

4.7 Post processing

Post processing are those calculations performed once the Picard iteration process has converged to an equilibrium given by $\psi(r, z)$ and the associated source functions $p'(\psi)$ and $TT'(\psi)$. The most computationally demanding post processing involves determining flux contours and subsequent integrals along those contours. In particular, flux surface averaged transport equations involve quantities in the form $Q_{\mu\nu} = (2\pi)^{-\mu}\oint|\nabla\psi|^{\mu} r^{\nu} dl_{\theta}$, and explicitly require $Q_{-1,0}$, $Q_{-1,1}$,$Q_{-1,-1}$,$Q_{1,-1}$ and $Q_{1,1}$ [2], or the ratios $Q_{-1,0}/Q_{-1,1}$, $Q_{-1,-1}/Q_{-1,1}$, $Q_{1,-1}/Q_{-1,1}$, $Q_{1,1}/Q_{-1,1}$ and $1/Q_{-1,1}$. One particular application is the calculation of the safety factor profile, using [22]

$$q = \oint \frac{T}{r|\nabla\psi|} dl_{\theta} = \frac{TQ_{-1,-1}}{2\pi}. \tag{70}$$

In the off-line version of LIUQE, these quantities are obtained using the Ψ-toolbox [23]. It is based on a bi-cubic spline interpolation of $\psi(r_x, z_x)$ and invokes the Halley method to iteratively solve

$$\psi(r_A - r^* cos\theta, z_A + r^* sin\theta) = \psi_q \tag{71}$$

for r^*, the distance between the magnetic axis and the contour point at poloidal angle θ on the flux contour level ψ_q. The definition of a normalised flux coordinate ρ and relations from [23]

$$\rho^2 = \frac{\psi_A - \psi}{\psi_A - \psi_0} \tag{72}$$

$$\nabla\psi = -2(\psi_A - \psi_0)\rho\nabla\rho \tag{73}$$

$$|\nabla\rho| = \sqrt{1 + \left(\frac{\partial_{\theta} r^*}{r^*}\right)^2} (\partial_{\rho} r^*)^{-1} \tag{74}$$

$$dl_{\theta} = r^* \sqrt{1 + \left(\frac{\partial_{\theta} r^*}{r^*}\right)^2} d\theta \tag{75}$$

are used to express $Q_{\mu\nu}$ as integrals over θ of functions of r^* :

$$Q_{-1,\nu} = -\frac{\pi}{2(\psi_A - \psi_0)} \oint_0^{2\pi} r^\nu \rho^{-1} \partial_\rho r^{*2} d\theta \tag{76}$$

$$Q_{1,\nu} = -\frac{2(\psi_A - \psi_0)}{\pi} \oint_0^{2\pi} r^\nu \left(r^{*2} + \frac{r^{*-2} (\partial_\theta r^{*2})^2}{4} \right) \rho (\partial_\rho r^{*2})^{-1} d\theta . \tag{77}$$

Once the flux contours are known on the computational grid (ρ_q, θ_q) in the form $(r_q(\psi_q, \theta_q), z_q(\psi_q, \theta_q))$, the calculation of integrals (76) and (77) can proceed, setting $r^{*2} = (r_q - r_A)^2 + (z_q - z_A)^2$ and $r^\nu = r_q^\nu$. Note that the expressions for these integrals use only r^{*2} to circumvent the slow square root calculation. The partial derivative with respect to θ and the integration over θ are based on a periodic cubic spline interpolation. The partial derivative with respect to ρ is based on a cubic spline interpolation, zero and symmetric at $\rho = 0$. Note that ρ is kept in the integral to be combined as a single matrix multiplication with the derivation operation ∂_ρ. For this operation to be sufficiently accurate with only a limited number of grid points, care must be taken in selecting the point spacing. This is particularly true near the X points where $\nabla\psi$ is small and equidistant ρ_q values result in a large physical distance between flux contours, and a poor approximation of the ∂_ρ operator. The following function describes the variation of the normalised flux along a line from the magnetic axis to an X point, where x is the normalised distance along this line:

$$\rho^2 = \begin{cases} 2x^2, x \in [0, 1/2] \\ 1 - 2(1-x)^2, x \in [1/2, 1] \end{cases} . \tag{78}$$

Thus substituting

$$x_q = \frac{\rho_1}{\sqrt{2}} + \frac{q-1}{n_q - 1} \frac{\sqrt{\rho_{n_q}^2 + 1} - \rho_1}{\sqrt{2}}, q = 1 \ldots n_q \tag{79}$$

in this function yields a ρ_q grid with almost equidistant flux contours near the X points (see Fig. 4). To avoid the undefined quantity $|\nabla\psi|^{-1}$ at the X point in $Q_{-1,\nu}$, a typical choice for the outmost contour is $\rho_{n_q} = \sqrt{0.95}$. The value for ρ_1 is selected so that the corresponding contour spans more than one grid cell, otherwise the interpolation scheme used in the real time contouring algorithm fails. For smaller value of ρ, and for the magnetic axis where the integrand of $Q_{-1,\nu}$ is undefined, another approach is adopted. In the vicinity of the magnetic axis, the flux is approximated using a second order Taylor expansion

$$\psi \cong \psi_A + \frac{1}{2} \begin{pmatrix} r - r_A & z - z_A \end{pmatrix} \cdot H(\psi_A) \cdot \begin{pmatrix} r - r_A \\ z - z_A \end{pmatrix} \tag{80}$$

which defines elliptical flux contours. The Hessian matrix is diagonalised,

$$H(\psi_A) = \begin{pmatrix} \cos\theta_0 & \sin\theta_0 \\ -\sin\theta_0 & \cos\theta_0 \end{pmatrix} \cdot \begin{pmatrix} h_1 & 0 \\ 0 & h_2 \end{pmatrix} \cdot \begin{pmatrix} \cos\theta_0 & -\sin\theta_0 \\ \sin\theta_0 & \cos\theta_0 \end{pmatrix}, \tag{81}$$

to define a new coordinate system:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\theta_0 & -\sin\theta_0 \\ \sin\theta_0 & \cos\theta_0 \end{pmatrix} \cdot \begin{pmatrix} r - r_A \\ z - z_A \end{pmatrix} = r^* \begin{pmatrix} \cos(\theta + \theta_0) \\ \sin(\theta + \theta_0) \end{pmatrix} \tag{82}$$

in which the flux function Taylor approximation (80) now becomes

$$\psi - \psi_A \cong \frac{1}{2}(h_1 x^2 + h_2 y^2) = \frac{r^{*2}}{2}(h_1 \cos^2(\theta + \theta_0) + h_2 \sin^2(\theta + \theta_0)). \tag{83}$$

With this approximation, the definition of ρ and the second order Taylor expansion of r^{-1} in r^*/r_A one obtains

$$Q_{-1,\nu} = \frac{4\pi^2 r_A^\nu \operatorname{sgn} h_1}{\sqrt{h_1 h_2}}\left(1 - \frac{[\nu = -1](\psi_A - \psi_0)\rho^2}{r_A^2}\frac{h_1 \sin^2\theta_0 + h_2 \cos^2\theta_0}{h_1 h_2}\right), \tag{84}$$

valid for $\nu \in \{-1, 0, 1\}$. For almost diagonal $H(\psi_A)$, i.e. a small tilt of the elliptical flux surfaces, this can be approximated by

$$Q_{-1,\nu} = -\frac{4\pi^2 r_A^\nu \operatorname{sgn}(\psi_A - \psi_0)}{\sqrt{\partial_r^2\psi_A \partial_z^2\psi_A - (\partial_r\partial_z\psi_A)^2}}\left(1 - \frac{[\nu = -1](\psi_A - \psi_0)\rho^2}{\partial_r^2\psi_A r_A^2}\left(1 + \frac{(\partial_r\partial_z\psi_A)^2}{\partial_r^2\psi_A \partial_z^2\psi_A}\right)\right). \tag{85}$$

Since the Hessian matrix on the magnetic axis is one of the outputs in the localisation of the magnetic axis position, this formulation provides a low cost way to estimate the value of $Q_{-1,\nu}$ in the region of the magnetic axis. In particular the central safety factor may be written as

$$q_A = \frac{2\pi \operatorname{sgn}(\psi_0 - \psi_A)}{\sqrt{\det H(\psi_A)}}\frac{T_A}{r_A} \tag{86}$$

which, combined with the Poisson equation evaluated on the magnetic axis $\operatorname{tr} H(\psi_A) = -2\pi\mu_0 r_A j_{\varphi A}$, yields

$$q_A = \frac{|\operatorname{tr} H(\psi_A)|}{\sqrt{\det H(\psi_A)}}\frac{T_A}{\mu_0 r_A^2 j_{\varphi A}}. \tag{87}$$

The second kind of flux integrals $Q_{1,\nu}$ using the same approximate near the magnetic axis, varies as $Q_{1,\nu} = -2(\psi_A - \psi_0)r_A^\nu\rho^2$.

5 Real time implementation

5.1 Algorithm

For the real time version of the equilibrium reconstruction LIUQE, a sufficiently short sampling time is required so that the change in the equilibrium between two successive samples is small. When realised, the Picard iteration scheme for one sample may be seeded with the plasma current distribution from the previous sample instead of the approximate initial distribution from finite element fitting (see Section Fig. 4). The number of iterations then required to reach a satisfactory convergence of the inverse equilibrium problem is considerably reduced; push to the extreme, only one iteration is performed and a new measurement set is employed by the source term fitting at each cycle. This is the approach adopted for the real time EFIT [11].

For such a scheme to be applicable, the computation time for one Picard iteration must be reduced until compatible with the evolution time of the equilibrium, typically 1 ms for TCV. This scheme, and the omission of the now unnecessary convergence test, are the only differences between the real time and the off-line equilibrium reconstruction code LIUQE. A thorough study of the difference in the results and of the applicability of this approach will be the subject of a future work.

The post processing step, on which most of the interesting control applications occur and depend, is also executed at each sample. This step begins with the identification of the flux contours. The needed computation to perform this identification by solving equation (71) with a bicubic spline interpolation is too heavy for a real time implementation, that calls for a simpler interpolation scheme. A real time contouring algorithm was developed that uses a 3 point interpolation around a grid point (r_{qx}, z_{qx}) close to the contour point; the contour point (r_q, z_q) is then the intersection of the radius through the magnetic axis at angle θ

$$-(r_q - r_A) sin\theta = (z_q - z_A) cos\theta \qquad (88)$$

and the contour at level ψ_q

$$\begin{aligned} &\psi(r_{qx}, z_{qx}) + \\ &\frac{r_q - r_{qx}}{\Delta r}(\psi(r_{qx} + \Delta r, z_{qx}) - \psi(r_{qx}, z_{qx})) + \\ &\frac{z_q - z_{qx}}{\Delta z}(\psi(r_{qx}, z_{qx} + \Delta z) - \psi(r_{qx}, z_{qx})) = \psi_q \end{aligned} \quad . \qquad (89)$$

The cell for the interpolation is taken as the one containing the contour point at the previous Picard iteration, or at the previous sample for the real time scheme. This choice is corrected, in anticipation, for a displacement of the magnetic axis, so that the selected cell contains the point

$$(r_q^{(t-1)} + r_A^{(t)} - r_A^{(t-1)}, z_q^{(t-1)} + z_A^{(t)} - z_A^{(t-1)}) . \qquad (90)$$

The initial guess $(r_q^{(0)}, z_q^{(0)})$ for all values of ψ_q is taken as a circle centered on the magnetic axis with a diameter of 0.8 times the smallest horizontal or vertical width of Ω_p. Additionally, the interpolation cell is forced to stay in its θ quadrant and is not allowed to move by more than one grid cell per iteration. The difference between the contours obtained so and with the bicubic spline interpolation is at most 5% of the cell size (see Fig. 4).

In practise, the real time iteration scheme is seeded with an initial guess of the plasma current density, as described in Sections Fig. 4, and flux surface contours, as described in this section. It is started when the plasma current exceeds a given threshold, set to 25kA for TCV. Sample after sample, the validity of the equilibrium is tested by many criteria, including the existence of a magnetic axis and the positivity of the profile of $p(\psi)$ and $T^2(\psi)$. If these fail, the iteration is restarted with the seeding procedure as explained above.

It is also possible to run off line the real time algorithm with the time reversed, since in the absence of a dynamic electromagnetic model of the vessel, nothing depends on the orientation of the time: in this case the equilibrium reconstruction for sample t is seeded with the current density of the sample $t + 1$. If this procedure is started at a time where the plasma current is well established, with a sufficient number of iterations at each sample to reach a converged equilibrium, it allows to extend the equilibrium reconstruction to the early phase of the plasma current ramp up, with plasma current as low as 10kA on TCV.

5.2 Real time hardware and software

The primary hardware host for the real time version of LIUQE is the distributed digital control system of TCV [24]. It consists of a network of customised Linux PC nodes; the real time operating mode is obtained by suspending CPU interrupts and memory paging. All nodes are connected via a reflective memory network, providing a reliable and deterministic method for sharing 128Mbytes of data between all nodes without CPU resources including interrupts. One node is dedicated to the real time LIUQE; it houses no analogue input or output hardware and takes its data, namely the measurement samples, and sends its results to the control nodes via the reflective memory network. It is presently based on a single core of an Intel i7 processor, overclocked, only for the duration of the shot, at 5GHz.

Algorithms running in real time on the TCV digital control system are programmed in SIMULINK, from which C code is automatically generated and compiled for the target node [25]. This approach offers many advantages. The abstraction from the specific target hardware makes the algorithm code portable, so that it can be developed and tested with simulated or previously acquired data, and debugged, profiled and optimised with powerful tools on any computer running MATLAB and SIMULINK. The code can also readily be shared with other fusion facilities or imported from collaborating laboratories with only minor adaptations. With the aim of using that programming language, the FORTRAN version of LIUQE was first fully rewritten to only a few hundred lines of MATLAB, permitting a straightforward conversion to SIMULINK.

5.3 Code optimisation

After several development iterations, the execution time of one cycle was optimised below the required 1ms. Each cycle comprises one Picard iteration and the necessary post processing as presented in Section 4.7 and 5.1, that is the determination of the flux contours and the evaluation of the contour integrals $Q_{-1,0}$, $Q_{-1,1}$, $Q_{-1,-1}$, $Q_{1,-1}$ and $Q_{1,1}$, the q profile and some other short calculations. With the present hardware and software, a cycle time less than 200μs was achieved with the full computational grid in (r, z), namely 28 by 65, fitting the source term with 4 parameters using all the 133 scalar measurements available in real time, and with a post processing grid in (ρ, θ) of size 17 by 32. The most time consuming steps are the calculation of the response matrix W_{rj} in equation (64) and the Poisson solver. The former consists of two inner products with dimension 75×1638 times 1638×3 and 75×1638 times 1638, and takes about $1/6$ of the cycle which is very close to the CPU's theoretical computational capability. The Poisson solver takes also roughly $1/6$ of the cycle, but is used twice, once to update the flux, equation (10), and once in the computation of the boundary condition, equation (48).

The programming options necessary to reach this performance level are presented below.

- Parameter values on the computational grid (r, z) are stored with z along the columns where the index varies most rapidly. Because the grid is elongated and has more points in the z axis, the Poisson solver involves more arithmetic operations in the z direction. This storage ensures more compact or contiguous indexing. This also facilitates differentiation along z that appears in many of the steps of the reconstruction procedure.

- Some of the steps require a complex or conditional array indexing that is not easily translated either in MATLAB or in SIMULINK without using *for* loops. They pervade the Poisson solver, the localisation of the magnetic axis and the X points, flagging points inside or outside the domain Ω_X, finding the bounding box of Ω_p and the efficient evaluation of the source term base functions for a given Ψ and Ω_p. For efficiency, they are coded as C functions and installed as SIMULINK *S blocks* built with its *Legacy Code Tool* so that the C code generation issues efficient calls to these functions.

- All multiplications involving vector or matrix operands are executed by the *Intel Math Kernel Library* (MKL). Again they are incorporated in SIMULINK *S blocks* built with the *Legacy Code Tool*, although it may become possible to automate the process by properly configuring the SIMULINK *Embedded Coder* with the *Code Replacement Library*.

- Taking advantage of the parallel computation capability of multi-core processors is also an option to decrease cycle time. The cyclic reduction scheme in the Poisson solver can be potentially optimised this way; however the amount of computation in each loop decreases with the number of possible parallel loops, making the achievable gain marginal. The parallel version of MKL using *Intel Open Multi Processing* (IOMP) was also compared with its sequential version, but here again resulting in only a small gain. This indicates that the overhead accompanying the additional thread management is not compensated by parallelisation for the data size involved. Moreover, IOMP turns out to be incompatible with interrupt suspension used in TCV's real time system. The *pthread* pragma is also out of scope, since instantiation of a thread takes longer than the achieved sequential cycle time. All code parallelisation has thus been rejected for the time being, in contrast with other real time equilibrium reconstruction implementations [8][9].

- Code vectorisation using the specific instruction set of the target processor and automatically generated with the corresponding compiler option (*-ax*) brought an appreciable and effort free acceleration of the code. Moreover this vectorisation is eased if all the variables, either integer or floating point, have the same binary length. This vectorisation is also facilitated by SIMULINK which generates code with explicit *for* loop limits.

- All calculations are performed in single precision, 32 bit floating point, as none of the algorithm steps are unduly dependent on the rounding error. This generates a reduction of 20% in the cycle time. This is attributed to an improved vectorisation by intensively using *Single Instruction Multiple Data* registers, and to the smaller data size, which should result in less frequent cache faults. The overall required memory for variables and parameters is about 8 Mbytes for the grid sizes given before, that is comparable to the L3 cache of the processor. Furthermore, limiting the processor to a single CPU core avoided L3 cache sharing issues.

- As mentioned before, use of mathematical functions like square root are limited to when strictly necessary. Where possible, multiplication by a precalculated inverse is preferred to division. Use of infinity and NaNs, defined by the IEEE floating point format, as a result of undefined mathematical operations, or as a way to mark for example grid values outside Ω_p, is proscribed since it triggers prohibitively slow FPU exceptions.
- The C code generated by the SIMULINK Embedded Coder was also examined for efficiency. *Selector blocks*, which, depending on a given condition, output one of their inputs, or *Conditionally executed blocks*, which update their outputs only if a given condition is satisfied, produce proper conditionally executed blocks of C code. The *Signal storage reuse* option is also set, that allows for memory buffers allocated to store block input and output signals to be reused later in the computation flow, thus reducing the required memory for variables.

6 Conclusions

The equilibrium reconstruction code LIUQE was presented in this article. The adopted algorithm is based on Picard iterations to solve the Poisson equation coupled with a linear parametrisation of the plasma current distribution adjusted to reproduce the available measurements at each iteration. In conclusion, some of the innovations brought in LIUQE and in its real time implementation are recalled.

It was demonstrated in this article that the Picard iteration algorithm is unstable against gross motion of the plasma column for elongated plasma shapes but can be stabilised by allowing for a vertical shift of the flux distribution in the measurement fitting step. Using algebraic manipulations, this small vertical shift can be introduced in the Poisson equation resolution without introducing difficulties in interpolating the truncated plasma current density. This can then be treated as an additional coefficient in the linear parametrisation of the plasma current density involving only the calculation of the vertical gradient of the smooth poloidal flux distribution and the analytically precalculated vertical gradient of the Green's functions. This stabilisation method is thus numerically sound and its real time implementation proved straightforward.

Due to the low electrical resistivity of the TCV vacuum vessel, designed to passively increase the stabilisation the plasma vertical position, large vessel currents are generated which have a sizeable influence on, and must be accounted for, the reconstructed equilibrium. An experimentally derived electromagnetic model of the vacuum was developed. It provides an estimation of the vessel current distribution that is included in the measurement fitting step.

Post processing of a reconstructed equilibrium mainly consists in determining the geometry of the flux contours and performing integrals along these contours, including the calculation of the safety factor profile. An adequate formulation for the calculation of the integrals, together with an appropriate treatment of their, a priori undefined, values near the magnetic axis and on the last closed flux surface, and a well selected unevenly spaced ρ grid were key elements for a successful coupling of the equilibrium reconstruction with the resolution of the 1.5D flux averaged transport equations already implemented in TCV's real time control system. A dedicated contouring algorithm was developed and incorporated in the real time equilibrium reconstruction to update the flux contour position and the related integrals at each time sample.

Adjusting the coefficients in the plasma current density parametrisation consists of a linear regression problem with about 133 measurements and 60 free parameters. Instead of a direct resolution of this problem, which would be too slow for the real time implementation of LIUQE, a fast iterative method is used that takes advantage of the block structure of the involved matrix when the measurements are grouped according to their sensitivity to plasma current and external currents.

The real time version of LIUQE is coded in SIMULINK, the block programming environment already adopted to generate the real time code for the whole TCV distributed digital control system. A cycle time below 200 µs was reached on a single core of a processor overclocked at 5 GHz, with the full 28 by 65 spatial grid, all the 133 available measurements and including post processing on a grid of 17 radial by 32 angular points. This was achieved with carefully selected numerical techniques, in particular the Poisson solver combines cyclic reduction and tridiagonal matrix inversion, coded in C for efficiency. Linear algebra is executed by the *Intel Math Kernel Library* that was found to be well tuned for the target processor. Parallelisation was tested but brought only marginal gains because of the relatively small data size involved that was negated by the increased overheads. Code vectorisation, automatically generated by the compiler to take benefit of the particular instruction set of the processor, yielded a substantial, and free, improvement.

Real time LIUQE is currently used as a basis for the development of shape control based on flux feedback on boundary points, to obtain the safety factor profile that guides real time stabilisation of MHD or NTM modes with ECH (Electron cyclotron heating), and to feed RAPTOR with the geometrical quantities involved in 1.5D transport equations to control current and pressure profiles with ECCD (Electron cyclotron current drive) and ECH.

Appendix A Green's function calculation

The mutual inductance between two circles located at (r, z) and (r', z') is evaluated using the analytical formula involving complete elliptic integrals of first and second kind, K and E respectively:

$$M(r, z, r', z') = \mu_0 u\left(\left(1 - \frac{k^2}{2}\right)K(k^2) - E(k^2)\right) \tag{91}$$

with

$$\begin{aligned} h &= z - z' \\ u^2 &= (r + r')^2 + h^2 \\ k^2 &= \frac{4rr'}{u^2} \end{aligned} . \tag{92}$$

The magnetic field per unit current at (r, z) is evaluated using the analytical formulae:

$$\begin{aligned} B_r(r, z, r', z') &= -\frac{\partial_z M(r, z, r', z')}{2\pi r} = \frac{\mu_0}{2\pi}\frac{h}{ru}\left(\frac{v^2}{d^2}E(k^2) - K(k^2)\right) \\ B_z(r, z, r', z') &= \frac{\partial_r M(r, z, r', z')}{2\pi r} = \frac{\mu_0}{2\pi}\frac{1}{u}\left(\frac{w^2}{d^2}E(k^2) + K(k^2)\right) \end{aligned} \tag{93}$$

with

$$\begin{aligned} d^2 &= (r' - r)^2 + h^2 \\ v^2 &= r'^2 + r^2 + h^2 \\ w^2 &= r'^2 - r^2 - h^2 \end{aligned} \cdot \tag{94}$$

The field in the poloidal direction θ is $B_r cos\theta + B_z sin\theta$. The following derivatives are also used:

$$\begin{aligned} \partial_z M &= -2\pi r B_r \\ \partial_z B_r &= \frac{\mu_0}{2\pi}\frac{1}{rd^2u^3}\left(\left(v^2u^2 - h^2\left(d^2 + \frac{k^2u^4}{d^2}\right)\right)E(k^2) + (h^2v^2 - u^2d^2)K(k^2)\right) \\ \partial_z B_z &= \frac{\mu_0}{2\pi}\frac{h}{d^2u^3}\left(-\left(3u^2 + 4\frac{v^2w^2}{d^2}\right)E(k^2) + w^2K(k^2)\right) \end{aligned} . \tag{95}$$

For discrete points, the following notation is used for any of the above functions G: $G_{\alpha\beta} = G(r_\alpha, z_\alpha, r_\beta, z_\beta)$; for a measurement α averaged over volume Ω_α, such as the magnetic probes,

$$G_{\alpha\beta} = \frac{\int\limits_{\Omega_\alpha} G(r_\alpha, z_\alpha, r_\beta, z_\beta)dr_\alpha dz_\alpha}{\int\limits_{\Omega_\alpha} dr_\alpha dz_\alpha}; \tag{96}$$

for a multiturn winding connected in series β such as the actively driven coils: $G_{\alpha\beta} = \sum\limits_\beta G(r_\alpha, z_\alpha, r_\beta, z_\beta)$.

Appendix B Plasma current estimator

Applying the integral form of Ampère's law on a polygonal integration contour defined by the probe locations (r_m, z_m), and using a trapeze approximation of the integration allows for an estimator of the plasma current to be written as

$$I_p = \frac{1}{\mu_0}\sum_{m=1}^{n_m}\frac{B_m + B_{(m \bmod n_m)+1}}{2}d_m \tag{97}$$

with $d_m = \sqrt{(r_{(m \bmod n_m)+1} - r_m)^2 + (z_{(m \bmod n_m)+1} - z_m)^2}$, or put in matrix form $I_p = I_{pm} \cdot B_m$ with

$$I_{pm} = \frac{d_m + d_{(m-2 \bmod n_m)+1}}{2\mu_0}. \tag{98}$$

Appendix C Bilinear finite elements

The $m \times n$ bilinear finite elements used to obtain the initial guess for the plasma current density (see Section Fig. 4) are defined in a rectangle $[a, b] \times [c, d]$, and their value is given by

$$h_{ij}(r, z) = \left(1 - \frac{|r - r_i|}{\Delta r}\right)\left(1 - \frac{|z - z_j|}{\Delta z}\right)[|r - r_i| \le \Delta r][|z - z_j| \le \Delta z], i = 1\ldots m, j = 1\ldots n \tag{99}$$

where $\Delta r = \frac{b-a}{m+1}$, $\Delta z = \frac{d-c}{n+1}$, $r_i = a + i\Delta r$, $z_j = c + j\Delta z$ and the square brackets are Iverson brackets whose value is 1 if the enclosed condition is satisfied, 0 otherwise.

Acknowledgments. This work was supported in part by the Swiss National Science Foundation. We owe special thanks to Dr Ferdinand Hofmann who elaborated over many decades the fundamental concepts in LIUQE and its original FORTRAN version.

References

[1] F. Piras, S. Coda, I. Furno, J.-M. Moret, R. A. Pitts, O. Sauter, B. Tal, G. Turri, A. Bencze1, B. P. Duval, F. Felici, A. Pochelon and C. Zucca, Snowflake divertor plasmas on TCV, Phys. Control. Fusion 51 (2009) 055009

[2] F. Felici, O. Sauter, S. Coda, B.P. Duval, T.P. Goodman, J.-M. Moret, J.I. Paley and the TCV Team, Real-time physics-model-based simulation of the current density profile in tokamak plasmas, Nucl. Fusion 51 (2011) 083052

[3] F. Hofmann, G. Tonetti, Tokamak equilibrium reconstruction using Faraday rotation measurements, Nucl. Fusion 28 (1998) 1871

[4] L.L. Lao, H. St. John, R.D. Stambaugh, A.G. Kellman and W. Pfeiffer, Reconstruction of current profile parameters and plasma shapes in tokamaks, Nucl. Fusion 25 (1985) 1611

[5] F. Hofmann, S. Coda, P. Lavanchy, X. Llobet, Ph. Marmillod, Y. Martin, A. Martynov, J. Mlynar, J.-M. Moret, A. Pochelon and O. Sauter, Extension of the TCV operating space towards higher elongation and higher normalized current, Nucl. Fusion 42 (2002) 743

[6] F. Hofmann, M. J. Dutch, J. B. Lister, Y. Martin and J.-M. Moret, On the possibility of creating doublet-shaped plasmas in TCV, 23rd EPS Conference on Controlled Fusion and Plasma Physics, Kiev, vol. 1, p. 127, 1996

[7] F. Imbeaux et al., A generic data structure for integrated modelling of Tokamak physics and subsystems, Comput. Phys. Comm. 181 (2010) 987

[8] M. Rampp, R. Preuss, R. Fischer, K. Hallatschek and L. Giannone, A parallel Grad-Shafranov solver for real-time control of Tokamak plasmas, Fusion Science and Technology 62 (2012) 409

[9] X. N. Yue, B. J. Xiao, Z. P. Luo and Y. Guo, Fast equilibrium reconstruction for tokamak discharge control based on GPU, Plasma Phys. Control. Fusion 55 (2013) 085016

[10] K. Lackner, Computation of ideal MHD equilibria, Computer Physics Communications 12 (1976) 33

[11] J.R. Ferron, M.L. Walker, L.L. Lao, H.E. St. John, D.A. Humphreys and J.A. Leuer, Real time equilibrium reconstruction for tokamak discharge control, Nucl. Fusion 38 (1998) 1055

[12] P. J. Mc Carthy, P. Martin, W. Schneider, The CLISTE interpretative equilibrium code, IPP report 5/85 (1999)

[13] J.-M. Moret, F. Buhlmann, D. Fasel, F. Hofmann, G. Tonetti, Magnetic measurements on the TCV Tokamak, Review of Scientific Instruments 69 (1998) 2333

[14] Y. Ou, M.L. Walker, E. Schuster and J.R. Ferron, Equilibrium reconstruction improvement via Kalman-filter-based vessel current estimation at DIII-D, Fusion Engineering and Design 82 (2007) 1144

[15] P. Bettini, M. Cavinato and F. Trevisan, Dynamic identification of plasma magnetic contour in fusion machines, Nucl. Fusion 45 (2005) 1

[16] J.-M. Moret, F. Hofmann, B.P. Duval and J.B. Lister, Breakdown in a continuous low resistivity vessel in TCV, 23rd EPS Conference on Controlled Fusion and Plasma Physics, Kiev, 1996

[17] J.-M. Moret, F. Buhlmann and G. Tonetti, Fast single loop diamagnetic measurements on the TCV tokamak, Review of Scientific Instruments 74 (2003) 4634

[18] G. Hommen, M. de Baar, P. Nuij, G. McArdle, R. Akers and M. Steinbuch, Optical boundary reconstruction of tokamak plasmas for feedback control of plasma position and shape, Rev. Sci. Instrum. 81 (2010) 113504

[19] F. Hofmann, FBT - a free-boundary Tokamak equilibrium code for highly elongated and shaped plasmas, Computer Physics Communication 48 (1988) 207

[20] Walter Gander and Gene H. Golub, Cyclic reduction, history and applications, Proceedings of the Workshop on Scientific Computing, 1997, Hong Kong, Springer Verlag, New York, 1997

[21] E.B. Rosa, F.W. Grover, Formulas and tables for the calculation of mutual and self-inductance [Revised], Scientific Papers of the Bureau of Standards 169 (1916).

[22] O. Sauter and S. Yu. Medvedev, Tokamak coordinate conventions: COCOS, Computer Physics Communications 184 (2013) 293

[23] J.-M. Moret, A software package to manipulate space dependencies and geometry in magnetic confinement fusion, Review of Scientific Instruments 76 (2005) 073507

[24] H.B. Le, F. Felici, J.I. Paley, B.P. Duval, J.-M. Moret, S. Coda, O. Sauter, D. Fasel, P. Marmillod and the TCV team, Distributed digital real-time control system for TCV tokamak, Fusion Engineering and Design 89 (2014) 155

[25] F. Felici, H.B. Le, J.I. Paley, B.P. Duval, S. Coda, J.-M. Moret, A. Bortolon, L. Federspiel, T.P. Goodman, G. Hommen, A. Karpushov, F. Piras, A. Pitzschke, J. Romero, G. Sevillano, O. Sauter, W. Vijvers and the TCV Team, Development of real-time plasma analysis and control algorithms for the TCV tokamak using Simulink, Fusion Engineering and Design 89 (2014) 165