JET–R(99)08

L C Ingesson

The Mathematics of some Tomography Algorithms used at JET

JET–R(99)08

The Mathematics of some Tomography Algorithms used at JET

L C Ingesson.

JET Joint Undertaking, Abingdon, Oxfordshire, OX14 3EA, UK.

March 2000

CONTENTS

1 Introduction 1

2 Mathematical details of the constrained optimization method 1

2.1 Matrix equation for constrained optimization 2

2.2 Irregular grid for variable grid size 4

2.3 Basis functions 6

2.4 Line integrals through basis functions 8

2.5 Derivative matrices 12

2.6 Isotropic and anisotropic unsmoothness 17

2.7 Analytical solution of constrained-optimization problem 22

2.8 Sparse matrix storage 26

3 Some mathematical details of other methods 27

3.1 Projection space coordinates 27

3.2 Cormack method in elliptical coordinates 30

3.3 Interpolative generalized natural basis function in object space 31

3.4 Details of filtered backprojection 33

ABSTRACT

Mathematical details are given of various tomographic reconstruction algorithms that are in use at JET. These algorithms include constrained optimization (CO) with local basis functions, the Cormack method, methods with natural basis functions and the iterative projection-space reconstruction method. Topics discussed include: derivation of the matrix equation for constrained optimization, variable grid size, basis functions, line integrals, derivative matrices, smoothness matrices, analytical expression of the CO solution, sparse matrix storage, projection-space coordinates, the Cormack method in elliptical coordinates, interpolative generalized natural basis functions and some details of the implementation of the filtered backprojection method.

1. INTRODUCTION

The algorithms for tomography presently used at JET for emission tomography have been described in several publications [1–5]. Much of the mathematical details of these algorithms were left out of those articles as their derivations are quite straightforward. The purpose of this report is to show in detail the derivations of the mathematical expressions used in the algorithms and to discuss other details of the implementation, in order that the algorithms and programs are fully documented. These mathematical expressions may also prove useful to readers who are implementing similar tomography algorithms.

The main tomography method discussed is the constrained optimization (CO) method with local basis functions [12]. Other methods known from the literature have been implemented at JET, such as the Cormack method [6,7], a truncated singular value decomposition method that uses natural basis functions [3], and the iterative projection-space reconstruction (IPR) method [5]. The IPR method uses the well-known filtered backprojection (FBP) method, also called convolution backprojection [8]. Other methods have been developed, such as a constrained-optimization method with natural basis functions [4]. All derivations of mathematical techniques that exceed a certain complexity and are not readily available from the literature have been included in this report. Although some context is given of the mathematical expressions, the present report is not self-contained: for the full context reference to the literature will be necessary. At JET much effort has been put into correcting for the beam widths of the imaging systems. Details of the mathematics of those techniques have been published elsewhere [9–11] and are not repeated here.

2. MATHEMATICAL DETAILS OF THE CONSTRAINED OPTIMIZATION METHOD

Constrained optimization methods are based on a standard way of solving ill-posed problems. Constrained optimization is particularly applicable to tomography when the coverage by lines of sight is coarse and irregular, as is the case in most applications of tomography in fusion research. The present implementation is based on the method implemented by Fuchs [12], but with several extensions: variable grid size, non-negativity constraints, taking into account a neutral-particle contribution [2] and taking into account beam widths [1,9–11]. The two first extensions will be discussed, in Secs. 2.2 and 2.7, respectively, as well as some features of which details have not been published or are not easy to find in the literature: the background of constrained optimization problems (Sec. 2.1), sparse-matrix storage (Sec. 2.8), derivative matrices (Sec. 2.5) used in the smoothness matrices, anisotropic diffusion on flux surfaces (Sec. 2.6), and line integrals (for poloidal and toroidal lines of sight) (Sec. 2.4) with bi-linear interpolative basis functions (Sec. 2.3).

2.1 Matrix equation for constrained optimization

In the constrained optimization method one tries to optimize an object function that describes some desired or expected property of the emission profile $g(x,y)$ (in emission tomography), which is constrained by the measurements. The emission profile is discretized onto a set of basis functions, which will be described in Secs. 2.2 and 2.3. If the object function $O(\boldsymbol{g})$, which is described in Sec. 2.6, is quadratic and based on smoothness, one can write

$$O(\boldsymbol{g}) = \langle \boldsymbol{g} | \boldsymbol{\Omega} | \boldsymbol{g} \rangle, \tag{1}$$

where the Dirac notation is used for the inner product and $\boldsymbol{\Omega}$ is a so-called smoothness matrix. With the discrepancy principle [13,14], one chooses the constraint to be

$$C(\boldsymbol{g}) = \|\boldsymbol{f} - \boldsymbol{K}\boldsymbol{g}\|^2 - \|\varepsilon\|^2, \tag{2}$$

where $\boldsymbol{f}$ are the measurements, $\boldsymbol{K}$ is a matrix that describes the geometric properties of the measurement process, and ε is the estimated noise. The object function can be optimized with respect to the constraint:

$$\min\{O(\boldsymbol{g}) \mid C(\boldsymbol{g}) \le 0\}. \tag{3}$$

In plain language this optimization methods finds the smoothest function for which the misfit is equal to the noise (i.e. the discrepancy principle). Alternatively it can be seen as selecting from all possible solutions that satisfy $C(\boldsymbol{g}) \le 0$ the solution that is smoothest.

The constrained problem of Eq. (3) can be solved by means of the Lagrange multiplier method. That is, the following system of equations is solved:

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{g}} O(\boldsymbol{g}) + \lambda \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{g}} C(\boldsymbol{g}) &= 0, \\ C(\boldsymbol{g}) &= 0, \end{aligned} \tag{4}$$

where λ is the Lagrange multiplier, which is required to be positive. In Cartesian coordinates for a vector $\boldsymbol{g}$: $\mathrm{d}/\mathrm{d}\boldsymbol{g} = \sum_i (\partial / \partial g_i) \boldsymbol{e}_i$, where $\boldsymbol{e}_i$ is the unit vector. It can be shown by writing out the inner products and matrix multiplications of $O(\boldsymbol{g})$ and $C(\boldsymbol{g})$ that

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{g}} \langle \boldsymbol{g} | \boldsymbol{A} | \boldsymbol{g} \rangle = \boldsymbol{A}\boldsymbol{g} + \boldsymbol{A}^{\mathrm{T}} \boldsymbol{g} \tag{5}$$

and

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{g}}\langle \boldsymbol{f}|\boldsymbol{A}|\boldsymbol{g}\rangle = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{f}. \tag{6}$$

Therefore, from Eq. (4) it follows that

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{g}}\left(\lambda\langle \boldsymbol{f}-\boldsymbol{Kg}|\boldsymbol{f}-\boldsymbol{Kg}\rangle+\langle \boldsymbol{g}|\boldsymbol{\Omega}|\boldsymbol{g}\rangle\right)=\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{g}}\left(\lambda\langle \boldsymbol{f}|\boldsymbol{f}\rangle+\lambda\langle \boldsymbol{Kg}|\boldsymbol{Kg}\rangle-2\lambda\langle \boldsymbol{f}|\boldsymbol{Kg}\rangle+\langle \boldsymbol{g}|\boldsymbol{\Omega}|\boldsymbol{g}\rangle\right)=\\&\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{g}}\left(\lambda\langle \boldsymbol{f}|\boldsymbol{f}\rangle+\lambda\langle \boldsymbol{g}|\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K}|\boldsymbol{g}\rangle-2\lambda\langle \boldsymbol{f}|\boldsymbol{K}|\boldsymbol{g}\rangle+\langle \boldsymbol{g}|\boldsymbol{\Omega}|\boldsymbol{g}\rangle\right)=2\lambda\boldsymbol{K}^{\mathrm{T}}\boldsymbol{Kg}+2\boldsymbol{\Omega}\mathrm{g}-2\lambda\boldsymbol{K}^{\mathrm{T}}\boldsymbol{f},\end{aligned} \tag{7}$$

where $(\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K})^{\mathrm{T}}=\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K}$ and $\boldsymbol{\Omega}^{\mathrm{T}}=\boldsymbol{\Omega}$ have been used[1]. Thus, the first line of Eq. (4) leads to the matrix equation

$$\left(\lambda\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K}+\boldsymbol{\Omega}\right)\boldsymbol{g}=\lambda\boldsymbol{K}^{\mathrm{T}}\boldsymbol{f}, \tag{8}$$

and the constraint on the second line of Eq. (4) gives the condition to determine λ. If the measurements are weighed by the covariance matrix $\boldsymbol{W}$, the constraint is

$$C(\boldsymbol{g})=\langle \boldsymbol{f}-\boldsymbol{Kg}|\boldsymbol{W}|\boldsymbol{f}-\boldsymbol{Kg}\rangle-\langle \boldsymbol{\Omega}|\boldsymbol{W}|\boldsymbol{\Omega}\rangle=\langle \boldsymbol{f}-\boldsymbol{Kg}|\boldsymbol{W}|\boldsymbol{f}-\boldsymbol{Kg}\rangle-M, \tag{9}$$

where M is the number of measurements, Eq. (8) becomes

$$\left(\lambda\boldsymbol{K}^{\mathrm{T}}\boldsymbol{WK}+\boldsymbol{\Omega}\right)\boldsymbol{g}=\lambda\boldsymbol{K}^{\mathrm{T}}\boldsymbol{Wf} \tag{10}$$

The reason for the equality in the constraint in the second line of Eq. (4) is as follows. Figure 1 shows $\langle \boldsymbol{f}-\boldsymbol{Kg}|\boldsymbol{W}|\boldsymbol{f}-\boldsymbol{Kg}\rangle$ and $O(\boldsymbol{g})$ as a function of $1/\lambda$. It is clear that to minimize $O(\boldsymbol{g})$ the maximum value of $\langle \boldsymbol{f}-\boldsymbol{Kg}|\boldsymbol{W}|\boldsymbol{f}-\boldsymbol{Kg}\rangle\leq M$ should be taken, i.e. $C(\boldsymbol{g})=0$. Note that both $O(\boldsymbol{g}(\lambda))$ and $C(\boldsymbol{g}(\lambda))$ are monotonic functions of λ, which can be proven for $C(\boldsymbol{g}(\lambda))$ for $\lambda>0$ [14].

The quadratic object function of Eq. (1) is only one of the possibilities. Sometimes non-linear object functions are chosen, such as in the maximum entropy method. In that case matrix equations such as Eqs. (8) and (10) do not exist and much more cumbersome numerical methods have to be used. Equations (8) and (10) also correspond to the Phillips-Tikhonov method to solve the problem. It is clear that if λ is chosen large, the measurements will dominate the solution of Eqs. (8) and (10), whereas if λ is chosen small, the object function will dominate. Instead of the discrepancy principle Eq. (2), it is common to choose other ways to determine the optimum regularization parameter λ. A large amount of literature exists on this topic, see for instance Ref. 13.

1 See Sec. 2.6 for the definition of Ω from which this property follows.

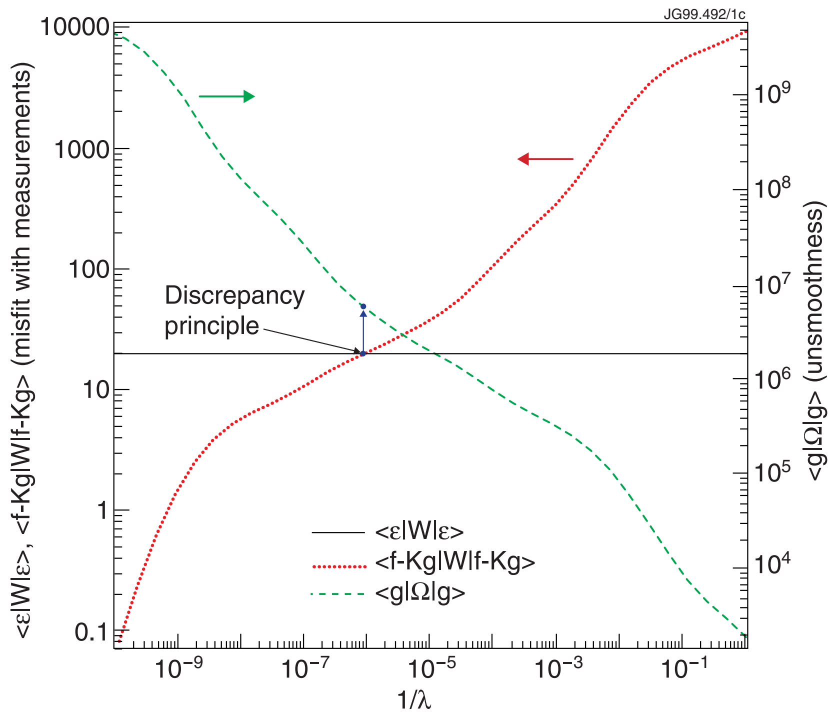

Fig.1: Typical example of the competing functions in constrained optimization, the misfit $\langle \boldsymbol{f} - \boldsymbol{Kg} | \boldsymbol{W} | \boldsymbol{f} - \boldsymbol{Kg} \rangle$ *(dotted line) and the unsmoothness* $\langle \boldsymbol{g} | \boldsymbol{\Omega} | \boldsymbol{g} \rangle$ *(dashed line), as a function of the inverse of the Lagrange multiplier 1/λ (the regularization parameter). The solid circles show the optimized Lagrange multiplier according to the discrepancy principle, i.e. where the misfit equals the estimated errors* $\langle \varepsilon | \boldsymbol{W} | \varepsilon \rangle = M$ *(solid line).*

The optimum λ for Eqs. (8) and (10) in combination with the constraint can be found iteratively because, as said, $O(\boldsymbol{g}(\lambda))$ and $C(\boldsymbol{g}(\lambda))$ are monotonic functions of λ. This is often done by solving Eqs. (8) or (10) iteratively for different λ. The solution can also be expressed analytically, which may be much faster, and can also include a non-negativity constraint [14,15], see Sec. 2.7.

2.2 Irregular grid for variable grid size

The total emission profile in JET shows little structure in the bulk plasma and much structure in the divertor plasma. The lines of sight of the bolometer systems therefore have large separations in the bulk plasma and small ones in the divertor. If a tomography method is well regularized, the grid size should not matter if the grid size (the distance between grid points) is chosen small enough (typically equal or smaller to the average separation between lines of sight in a region). Many grid points, however, are very expensive computationally and consume a lot of memory. For bolometer tomography at JET it makes much sense to choose a large grid size in the bulk plasma and a small grid size in the divertor. A rectangular reconstruction region with a constant grid size is by far the simplest to implement. A Cartesian grid with variable grid size is complicated, but straightforward. Non-Cartesian grids, for example in flux coordinates, are further complications with certain benefits if the emissivity is well represented in flux coordinates, such as soft x-ray radiation, but also have complications, for example when the line integrals of full geometric properties of the imaging system should be taken into account. Furthermore, there is a danger to impose a grid onto a solution when the grid may not be appropriate, for instance when the reconstructed flux is not accurate; a Cartesian grid is insensitive to such problems as it is more flexible (the fact that one expects the emission to be approximately constant on flux surfaces can, for instance, be built into the object function).

A irregular grid scheme for variable grid size was implemented. The requirements were that the basis functions used can be defined on the grid, that the local emissivity can be represented by these basis functions, and that derivative matrices can be calculated. Furthermore, the edge of the reconstruction region can be irregular, so that no grid points are wasted on the uninteresting region outside the irregularly shaped vacuum vessel. It was determined that for a given problem the grid would be built and remain fixed, i.e. the grid cannot be refined during tomographic reconstructions.

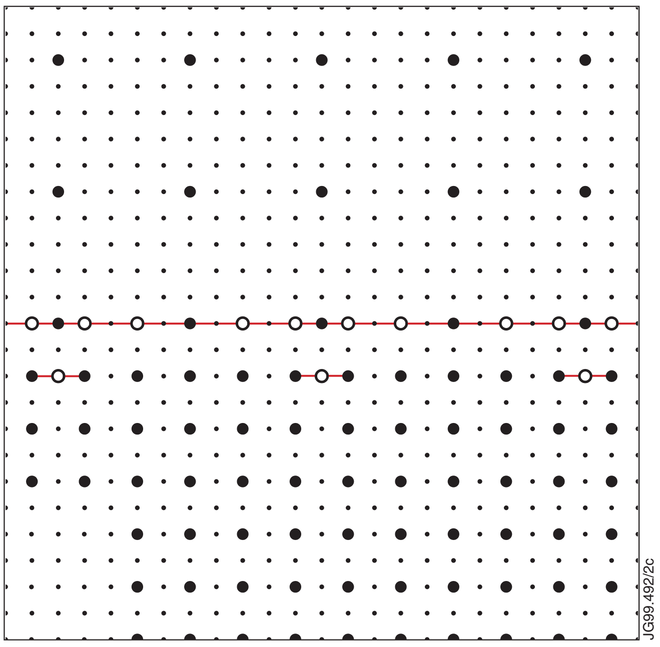

Fig.2: Typical example of a small part of a grid with variable grid size (used for the simulations presented in Ref. 21). The dots indicate the regular original grid, the solid circles the actual grid points and the open circles virtual grid points. Note how on the boundary of grid sizes the virtual grid points ensure that actual grid points always have four neighbours on the grid lines. The lines around virtual grid points indicate which actual points are their neighbours: the value held by the virtual point is the linear interpolation between the two actual neighbours. Note that several virtual points can have the same actual neighbours. Outside the reconstruction region no actual points are selected, as in the lower left corner of the figure.

The bi-interpolative basis functions described in Sec. 2.3 and the requirement of derivative matrices to be continuous over the boundary between grid sizes were the main factors that determined the implementation. The grid points are a subset of a fine rectangular regular Cartesian grid. There are non-used, actual and virtual grid points. Actual grid points are the only points to hold independent values. The virtual points are only used to ease the calculations on the edges between regions with various grid sizes. The virtual points hold a virtual value that is the linear interpolation between its two actual neighbours. An example of a grid is given in Fig.2. Each actual grid point is surrounded by four pixels, each of which has four corners which are either actual or virtual grid points. The way the grid sizes can be chosen in different regions is very flexible, although there is a strict requirement of the placement of virtual grid points in appropriate places, and is certainly not limited to two different sizes in one grid. To speed up the calculations, the two neighbours of each virtual point, and the four neighbours (actual or virtual) of each actual point are stored so that no searches are needed while building basis functions and derivative matrices. The main difficulties in generating grids are to ensure that: virtual grid points are located where required, there are no conflicts between different grid sizes and virtual points on the grid boundary, and that the correct neighbours are assigned to each point.

The chosen type of grid is very suitable for the pyramid basis functions described in Sec. 2.3, but it is less suitable for square constant pixels[2]. Higher order basis functions can in principle be defined on the grid, although in that case searches of the neighbours of neighbours have to be made and limitations may have to be imposed on the grid[3].

2.3 Basis functions

In series-expansion methods for emission tomography, the emissivity $g(x,y)$ is expanded into basis functions $b_j(x,y)$ by

$$g(x,y) \approx \sum_j g_j b_j(x,y). \tag{11}$$

See Refs. 1, 4 and 16 for more detailed descriptions (including that the basis functions do not necessarily need to be orthogonal) and discussions on possible basis functions (such as global and natural basis functions[4]).

A common discretization of $\boldsymbol{g}$ is into pixels of a grid (see for example Ref. 17). In this case the basis functions $b_j(x,y)$ are

$$b_j(x,y) = \begin{cases} 1 & \text{if } (x,y) \text{ inside the } j\text{th pixel,} \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

This is an example of *local* basis functions. In the present implementation local basis functions are used that describe bilinear interpolation between the grid points, which are pyramid shaped

2 In the current implementation each actual grid point holds values. For square constant pixels the pixel holds the values and a way has to be chosen to represent those values unambiguously (in, for instance, the lower left corner point of each pixel; but what happens if that is a virtual point?).

3 As Sec. 2.3 shows, the basis functions on the boundary between different grid sizes are fairly complicated and may cover many pixels depending on the virtual grid points. It should be possible to extend the definition for higher order basis functions, but much care has to be taken to do this properly. It may turn out to be impossible or very difficult to do this for arbitrary grids.

4 Basis functions can be divided into local, global and natural types [4,16]. Local basis functions are non-zero in only a small part of the reconstruction region in object space, but their Radon transforms are non-zero over a large part of projection space. Global basis functions and their Radon transforms are non-zero over most part of the reconstruction region in object space and projection space. Natural basis function are non-zero in a small part of projection space, but their backprojections are non-zero over a large part of object space.

with rounded corners between grid points (see Fig.3). On a regular grid these basis functions can be expressed as

$$b_j(x,y) = t_x(x,x_j)\, t_y(y,y_j), \tag{13}$$

where the triangle functions t_x and t_y are given by

$$t_z(z,z_c) = \begin{cases} 0 & \text{if } z < z_c - \Delta z \\ \dfrac{z - z_c + \Delta z}{\Delta z} & \text{if } z_c - \Delta z \le z < z_c \\ \dfrac{z_c + \Delta z - z}{\Delta z} & \text{if } z_c \le z < z_c + \Delta z \\ 0 & \text{if } z \ge z_c + \Delta z \end{cases} = \begin{cases} 0 & \text{if } |z - z_c| \ge \Delta z \\ 1 - \dfrac{|z - z_c|}{\Delta z} & \text{if } |z - z_c| < \Delta z \end{cases} \tag{14}$$

where z is x or y and Δz indicates the distance between grid points. These basis functions ensure that the resulting $g(x,y)$ in Eq. (11) is continuous. In the case of constant pixel basis functions, g_j corresponds to the average emissivity in the pixel. In the case of the bilinear interpolation, g_j corresponds to the actual value in the grid point and thus $g_j = g(x_j, y_j)$, (x_j, y_j) being the coordinates of grid point j.

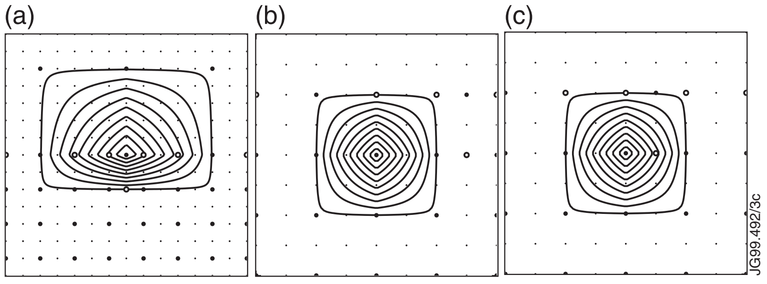

Fig.3: Three examples of pyramid basis functions on the boundary between different grid sizes. The symbols for grid points follow the convention of Fig. 2. The contour plots have contours at 0.01, 0.1, 0.2 ... 1.0. Note how the pixels that are covered by the basis function are determined by the virtual corner points of which the central actual point, to which the basis function corresponds, is a neighbour.

The geometric matrix $\boldsymbol{K}$ of Sec. 2.4 is given by

$$K_{ij} = \int\!\!\int K_i(x,y)\, b_j(x,y)\, \mathrm{d}x\, \mathrm{d}y, \tag{15}$$

for these basis functions, where $K_i(x,y)$ is the geometric function [1,9,11].

On the grid with variable grid size great care has to be taken to define the basis functions in a meaningful way. The basis function corresponding to actual grid point j will have non-zero values in all pixels that surround the point, and in pixels for which the point is a neighbour of a virtual corner point. This means that more than four pixels can be involved (see Fig. 3). The reason for this is that when the bi-linear interpolative values are assigned to the corners of a pixel, the values of virtual corner points are distributed over their two neighbouring actual points.

To integrate over the grid, for instance to determine the total radiated power from the reconstructed emission profile, it is necessary to consider all pixels, not the actual grid points.

All corner points, actual and virtual, have a value (for virtual points this is the linearly interpolated values between its two actual neighbours). Iterating through all pixels, a factor of 1/4 of the value of each of its corner points is added to the sum, weighted by the size of the pixel.

2.4 Line integrals through basis functions

The line integral through a basis functions is given by the Radon transform of the basis function. For square constant pixels the line integral is simply the length of the viewing chords through the pixel. The length l of the chord $p + x\sin\xi - y\cos\xi = 0$, where p is the distance to the lower-left corner of the pixel with size $\Delta x \times \Delta y$ and ξ the angle with the x axis and $0 \le \xi < \arctan(\Delta y / \Delta x)^5$, is:

$$l = \begin{cases} \dfrac{p}{\sin\xi\cos\xi}, & (\Delta y\sin\xi < \Delta x\cos\xi \text{ and } 0 < p < \Delta y\sin\xi) \text{ or} \\ & (\Delta y\sin\xi > \Delta x\cos\xi \text{ and } 0 < p < \Delta x\cos\xi), \\ \dfrac{\Delta y}{\cos\xi}, & \Delta y\sin\xi \le p \le \Delta x\cos\xi, \\ \dfrac{\Delta y}{\sin\xi}, & \Delta x\cos\xi \le p \le \Delta y\sin\xi, \\ \dfrac{\Delta y\sin\xi + \Delta x\cos\xi - p}{\sin\xi\cos\xi}, & \Delta x\cos\xi < p < \Delta y\sin\xi + \Delta x\cos\xi \text{ or} \\ & \Delta y\sin\xi < p < \Delta y\sin\xi + \Delta x\cos\xi, \\ 0 & p \ge \Delta y\sin\xi + \Delta x\cos\xi. \end{cases} \tag{16}$$

For ξ outside this range, Eq. (16) can be used after symmetry considerations. See Ref. [18] for the derivation (for square pixels).

For other basis functions the derivation is more complicated. In the case of the pyramid basis functions the line integral can still be expressed analytically both for poloidal and toroidal lines of sight. The expressions below can be used by iterating through the lines of sight, solving the intersections of the line with all the grid lines[6]. The expressions below give the contribution to the four corner points of each pixel through which the line of sight passes.

2.4.1 Lines of sight in the poloidal cross-section in case of pyramid basis functions

The line of sight in poloidal coordinates is given by

$$\begin{bmatrix} R \\ Z \end{bmatrix} = \begin{bmatrix} R_\mathrm{s} \\ Z_\mathrm{s} \end{bmatrix} + \lambda \begin{bmatrix} R_\mathrm{d} \\ Z_\mathrm{d} \end{bmatrix}, \tag{17}$$

where $(R_\mathrm{s}, Z_\mathrm{s})$ is the starting point of the line (on the reconstruction boundary) and $(R_\mathrm{d}, Z_\mathrm{d})$ the direction vector, which is assumed to be normalized.

5 See Sec. 3.1 for a description of projection-space coordinates p and ξ.

6 This is a certain method. A method in which the line is traced through the various neighbouring pixels can fail when numerical errors lead to an incorrect pixel.

The intersections of the line with all grid lines $R = R_j$ and $Z = Z_j$ are stored in the array λ_n, which are easily found from

$$R_j = R_\mathrm{s} + \lambda R_\mathrm{d} \text{ and } Z_j = Z_\mathrm{s} + \lambda Z_\mathrm{d}. \tag{18}$$

Each pair λ_n, λ_{n+1} give the boundaries of the pixels through which the line passes.

The line integral of the basis function in each pixel can be evaluated analytically in terms of the grid-point values. If g_j, g_{j+1}, g_{j+J} and g_{j+J+1} are the values contained in the grid points at the lower left, upper left, lower right and upper right corners of the pixel respectively, the bi-linearly interpolated value in any internal point (*x*,*y*) is given by (see the definition of the bi-linearly interpolative basis function Eqs. (13) and (14) in Sec. 2.3):

$$\begin{aligned} g(R,Z) = {} & \left(1 - \frac{R}{\Delta R} + \frac{R_j}{\Delta R}\right)\left(1 - \frac{Z}{\Delta Z} + \frac{Z_j}{\Delta Z}\right) g_j + \\ & \left(1 - \frac{R}{\Delta R} + \frac{R_j}{\Delta R}\right)\left(\frac{Z}{\Delta Z} - \frac{Z_j}{\Delta Z}\right) g_{j+1} + \\ & \left(\frac{R}{\Delta R} - \frac{R_j}{\Delta R}\right)\left(\frac{Z}{\Delta Z} - \frac{Z_j}{\Delta Z}\right) g_{j+J+1} + \\ & \left(\frac{R}{\Delta R} - \frac{R_j}{\Delta R}\right)\left(1 - \frac{Z}{\Delta Z} + \frac{Z_j}{\Delta Z}\right) g_{j+J}. \end{aligned} \tag{19}$$

This equation can be factored as follows:

$$g(R,Z) = A + BR + CZ + DRZ, \tag{20}$$

where

$$A = \frac{a_j g_j + a_{j+1} g_{j+1} + a_{j+J+1} g_{j+J+1} + a_{j+J} g_{j+J}}{\Delta R \Delta Z} \tag{21}$$

and similarly for *B*, *C* and *D*. The components are given by

$$\begin{aligned} & a_j = (R_j + \Delta R)(Z_j + \Delta Z), a_{j+1} = (R_j + \Delta R)(-Z_j + \Delta Z), \\ & a_{j+J+1} = (-R_j + \Delta R)(-Z_j + \Delta Z), a_{j+J} = (-R_j + \Delta R)(Z_j + \Delta Z), \end{aligned} \tag{22a}$$

$$b_j = -(Z_j + \Delta Z), b_{j+1} = -(-Z_j + \Delta Z), b_{j+J+1} = (-Z_j + \Delta Z), b_{j+J} = (Z_j + \Delta Z), \tag{22b}$$

$$c_j = -(R_j + \Delta R), c_{j+1} = (R_j + \Delta R), c_{j+J+1} = (-R_j + \Delta R), c_{j+J} = -(-R_j + \Delta R), \tag{22c}$$

$$d_j = 1, d_{j+1} = -1, d_{j+J+1} = 1, d_{j+J} = -1. \tag{22d}$$

It is clear that substituting the equation for the line (17) in $g(R(\lambda), Z(\lambda))$ will give terms in 1, λ and λ^2. Carrying out the integral for the interval λ_n to λ_{n+1} (which corresponds to the pixel *j*) gives the contributions

$$I_1 = \int_{\lambda_n}^{\lambda_{n+1}} \mathrm{d}\xi = \lambda_{n+1} - \lambda_n, \tag{23a}$$

$$I_2 = \int_{\lambda_n}^{\lambda_{n+1}} \lambda \,\mathrm{d}\lambda = \tfrac{1}{2}(\lambda_{n+1}^2 - \lambda_n^2), \tag{23b}$$

$$I_3 = \int_{\lambda_n}^{\lambda_{n+1}} \lambda^2 \,\mathrm{d}\lambda = \tfrac{1}{3}(\lambda_{n+1}^3 - \lambda_n^3). \tag{23c}$$

Thus,

$$\int_{\lambda_n}^{\lambda_{n+1}} g(R(\lambda), Z(\lambda))\,\mathrm{d}\lambda = AI_0 + BI_R + CI_Z + DI_{RZ}, \tag{24}$$

where

$$I_0 = I_1, \tag{25a}$$

$$I_R = R_\mathrm{s} I_1 + R_\mathrm{d} I_2, \tag{25b}$$

$$I_Z = Z_\mathrm{s} I_1 + Z_\mathrm{d} I_2, \tag{25c}$$

$$I_{RZ} = R_\mathrm{s} Z_\mathrm{s} I_1 + (R_\mathrm{s} Z_\mathrm{d} + R_\mathrm{d} Z_\mathrm{s}) I_2 + R_\mathrm{d} Z_\mathrm{d} I_3. \tag{25d}$$

Factoring out the g_js from Eq. (21), one finds that the integral contributes the following value corner point *j*

$$a_j I_0 + b_j I_R + c_j I_Z + d_j I_{RZ}, \tag{26}$$

and similarly for the other grid points. The sum over the contributions to the four corner points of the pixel must be equal to $\lambda_{n+1} - \lambda_n$.

2.4.2 Lines of sight in the toroidal direction in case of pyramid basis functions

A straight-line view in the toroidal direction is given by

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_\mathrm{s} \\ y_\mathrm{s} \\ z_\mathrm{s} \end{bmatrix} + \lambda \begin{bmatrix} x_\mathrm{d} \\ y_\mathrm{d} \\ z_\mathrm{d} \end{bmatrix}, \tag{27}$$

where $(x_\mathrm{s}, y_\mathrm{s}, z_\mathrm{s})$ is the starting point, and the direction vector $(x_\mathrm{d}, y_\mathrm{d}, z_\mathrm{d})$ again is assumed to be normalized. This straight line can be mapped to toroidal coordinates (R,Z,ϕ):

$$\begin{aligned} x &= R\cos\phi, & R &= \sqrt{x^2 + z^2}, \\ y &= Z, \quad \text{or} & Z &= y, \\ z &= -R\sin\phi, & \phi &= \arctan(-z/x). \end{aligned} \tag{28}$$

If one can assume the emissivity to be toroidally symmetrical, i.e. independent of ϕ, the toroidal line integral through the basis functions can be evaluated; the emissivity in a pixel is given by

Eq. (19). The starting point $(x_\mathrm{s}, y_\mathrm{s}, z_\mathrm{s})$ is assumed to be on the reconstruction boundary in *R* and *Z* coordinates.

The intersections with the grid lines $Z = Z_j$ are trivial:

$$Z_j = Z_\mathrm{s} + \lambda Z_\mathrm{d}. \tag{29}$$

The intersections with the grid lines $R = R_j$ are given by

$$(x_\mathrm{s} + \lambda x_\mathrm{d})^2 + (z_\mathrm{s} + \lambda z_\mathrm{d})^2 = R_j^2. \tag{30}$$

This simple quadratic equation can have zero, one or two solutions:

$$a = x_\mathrm{s}^2 + z_\mathrm{s}^2 - R_j^2,\, b = 2(x_\mathrm{s}x_\mathrm{d} + z_\mathrm{s}z_\mathrm{d}),\, c = x_\mathrm{d}^2 + z_\mathrm{d}^2,\, \Delta = 4ac - b^2, \tag{31}$$

$$\text{if } \Delta \le 0 : \lambda = \frac{-b \pm \sqrt{-\Delta}}{2c}. \tag{32}$$

Note that all negative solutions and solutions larger than the λ of the end point on the reconstruction boundary should be discarded (the latter solution would correspond to a bent line being able to see through the solid wall at the edge of the reconstruction region).

Given these found λ_n, the integral can be evaluated as in Sec. 2.4.1, with only I_R and I_{RZ} different. To express those integrals one requires the definitions (31), with $R_j = 0$, and

$$q_0(\lambda) = \int \frac{1}{\sqrt{a + b\lambda + c\lambda^2}}\mathrm{d}\lambda = \begin{cases} -\frac{1}{\sqrt{-c}}\arcsin\left(\frac{2c\lambda + b}{\sqrt{-\Delta}}\right) & c < 0 \text{ and } \Delta < 0 \\ \frac{1}{\sqrt{c}}\ln\left(2\sqrt{c(a + b\lambda + c\lambda^2)} + 2c\lambda + b\right) & c > 0 \\ \text{undefined} & c = 0 \text{ or } (c < 0 \text{ and } \Delta > 0) \end{cases} \tag{33}$$

$$q_1(\lambda) = \int \sqrt{a + b\lambda + c\lambda^2}\,\mathrm{d}\lambda = \frac{(2c\lambda + b)\sqrt{a + b\lambda + c\lambda^2}}{4c} + \frac{\Delta}{8c}q_0(\lambda), \tag{34}$$

$$q_2(\lambda) = \int \lambda\sqrt{a + b\lambda + c\lambda^2}\,\mathrm{d}\lambda = \frac{\sqrt{(a + b\lambda + c\lambda^2)^3}}{3c} - \frac{(2c\lambda + b)b\sqrt{a + b\lambda + c\lambda^2}}{8c^2} - \frac{b\Delta}{16c^2}q_0(\lambda), \tag{35}$$

which expressions are Eqs. (2.261), (2.262.1) and (2.262.2) of Ref. 19.[7] With Eq. (30), $R(\lambda) = \sqrt{(x_\mathrm{s} + \lambda x_\mathrm{d})^2 + (z_\mathrm{s} + \lambda z_\mathrm{d})^2}$,[8] one obtains

7 The case $c = 0$ corresponds to a vertical poloidal line, and should be treated as such with the expressions of Sec. 2.4.1.

8 Only positive *R* values are relevant.

$$I_R = \int_{\lambda_n}^{\lambda_{n+1}} R(\lambda)\,\mathrm{d}\lambda = q_1(\lambda_{n+1}) - q_1(\lambda_n), \tag{36a}$$

$$I_{RZ} = \int_{\lambda_n}^{\lambda_{n+1}} R(\lambda)Z(\lambda)\,\mathrm{d}\lambda = Z_\mathrm{s} I_R + Z_\mathrm{d}\big(q_2(\lambda_{n+1}) - q_2(\lambda_n)\big). \tag{36b}$$

2.4.3 Implementation with variable grid size

The formulae derived in Secs. 2.4.1 and 2.4.2 are valid for constant grid size. With some careful considerations they can also be used with the pyramid basis functions for variable grid size (Sec. 2.3). One has to consider all pixels with actual and virtual corners. The λ_n are solved for all intersections with the original grid, and only the solutions that are on the edges of pixels are retained. Then, as above, one steps through the pixels that are intersected (i.e. one steps through pairs of λ_n, λ_{n+1}) and calculates the contributions to the corner points of the pixel, making sure to use the Δx and Δy of the pixel in question. If a corner point is actual, the value is assigned (added) to that point, if it is virtual the value is spread over the two actual neighbours of the actual point[9]. This process has the effect of integrating correctly over basis functions such as the ones displayed in Fig.3.

2.5 Derivative matrices

2.5.1 Derivative matrices for a grid with constant grid size

The smoothness object function for constrained optimization can conveniently be expressed in terms of derivative matrices, see Sec. 2.6. Derivative matrices can be derived in the following way, see also Ref. 20. Here symmetric definitions are chosen for the first derivatives and the second derivative to x and y. Other definitions are of course possible [20], but are probably not very important. The main obstacle in defining derivative matrices for two-dimensional grids is the numbering of the grid points. In first instance, two indices x and y are used, where x runs from left to right and y from top to bottom (so, the signs for y will be reversed with respect to the formulae for x), start at 1 and end at X and Y, respectively. Δx and Δy are the distances between the grid points.

The first derivatives to x for central and boundary points are given by:[10]

$$\left(\frac{\partial g}{\partial x}\right)_{y,x} = \frac{\dfrac{g_{y,x} - g_{y,x-1}}{\Delta x} + \dfrac{g_{y,x+1} - g_{y,x}}{\Delta x}}{2} = \frac{g_{y,x+1} - g_{y,x-1}}{2\Delta x}, \tag{37a}$$

$$\left(\frac{\partial g}{\partial x}\right)_{y,1} = \frac{g_{y,2} - g_{y,1}}{\Delta x}, \tag{37b}$$

9 For example, if the virtual point is 2/3 between its two neighbours, 2/3 of the value will be allocated to the closest neighbour and 1/3 to the other neighbour.

10 Equation (37a) defines the elements $D_{[y,x][y,x+1]} = 1/(2\Delta x)$ and $D_{[y,x][y,x-1]} = -1/(2\Delta x)$, et cetera.

$$\left(\frac{\partial g}{\partial x}\right)_{y,X} = \frac{g_{y,X} - g_{y,X-1}}{\Delta x}; \qquad (37c)$$

the first derivative to *y*:

$$\left(\frac{\partial g}{\partial y}\right)_{y,x} = \frac{g_{y-1,x} - g_{y+1,x}}{2\Delta y}, \qquad (38a)$$

$$\left(\frac{\partial g}{\partial y}\right)_{1,x} = \frac{g_{1,x} - g_{2,x}}{\Delta y}, \qquad (38b)$$

$$\left(\frac{\partial g}{\partial y}\right)_{Y,x} = \frac{g_{Y-1,x} - g_{Y,x}}{\Delta y}; \qquad (38c)$$

the second derivative to *x*:

$$\left(\frac{\partial^2 g}{\partial x^2}\right)_{y,x} = \left[\frac{\partial}{\partial x}\left(\frac{\partial g}{\partial x}\right)\right]_{y,x} = \frac{\dfrac{g_{y,x+1} - g_{y,x}}{\Delta x} - \dfrac{g_{y,x} - g_{y,x-1}}{\Delta x}}{\Delta x} = \frac{g_{y,x+1} + g_{y,x-1} - 2g_{y,x}}{(\Delta x)^2}, \qquad (39a)$$

$$\left(\frac{\partial^2 g}{\partial x^2}\right)_{y,1} = \frac{g_{y,3} + g_{y,1} - 2g_{y,2}}{2(\Delta x)^2} \text{ or } \frac{g_{y,2} - g_{y,1}}{(\Delta x)^2}, \qquad (39b)$$

$$\left(\frac{\partial^2 g}{\partial x^2}\right)_{y,X} = \frac{g_{y,X} + g_{y,X-2} - 2g_{y,X-1}}{2(\Delta x)^2} \text{ or } \frac{g_{y,X} - g_{y,X-1}}{(\Delta x)^2}, \qquad (39c)$$

the second derivative to *y*:

$$\left(\frac{\partial^2 g}{\partial y^2}\right)_{y,x} = \frac{g_{y-1,x} + g_{y+1,x} - 2g_{y,x}}{(\Delta y)^2}, \qquad (40a)$$

$$\left(\frac{\partial^2 g}{\partial y^2}\right)_{1,x} = \frac{g_{1,x} + g_{3,x} - 2g_{2,x}}{2(\Delta y)^2} \text{ or } \frac{g_{1,x} - g_{2,x}}{(\Delta y)^2}, \qquad (40b)$$

$$\left(\frac{\partial^2 g}{\partial y^2}\right)_{Y,x} = \frac{g_{Y-2,x} + g_{Y,x} - 2g_{Y-1,x}}{2(\Delta y)^2} \text{ or } \frac{g_{Y-1,x} - g_{Y,x}}{(\Delta y)^2}; \qquad (40c)$$

and the mixed second derivative:

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{y,x} = \left(\frac{\partial}{\partial x}\left(\frac{\partial g}{\partial y}\right)\right)_{y,x} = \frac{\left(\dfrac{\partial g}{\partial y}\right)_{y,x+1} - \left(\dfrac{\partial g}{\partial y}\right)_{y,x-1}}{2\Delta x} = \qquad (41a)$$

$$\frac{g_{y-1,x+1} + g_{y+1,x-1} - g_{y-1,x-1} - g_{y+1,x+1}}{4\Delta x \Delta y},$$

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{1,x} = \frac{g_{1,x+1} + g_{2,x-1} - g_{1,x-1} - g_{2,x+1}}{2\Delta x \Delta y}, \quad [x = 2 \ldots (X-1)] \tag{41b}$$

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{Y,x} = \frac{g_{Y-1,x+1} + g_{Y,x-1} - g_{Y-1,x-1} - g_{Y,x+1}}{2\Delta x \Delta y}, \quad [x = 2 \ldots (X-1)] \tag{41c}$$

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{y,1} = \frac{g_{y-1,2} + g_{y+1,1} - g_{y-1,1} - g_{y+1,2}}{2\Delta x \Delta y}, \quad [y = 2 \ldots (Y-1)] \tag{41d}$$

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{y,X} = \frac{g_{y-1,X} + g_{y+1,X-1} - g_{y-1,X-1} - g_{y+1,X}}{2\Delta x \Delta y}, \quad [y = 2 \ldots (Y-1)] \tag{41e}$$

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{1,1} = \frac{g_{1,2} + g_{2,1} - g_{1,1} - g_{2,2}}{\Delta x \Delta y}, \tag{41f}$$

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{1,X} = \frac{g_{1,X} + g_{2,X-1} - g_{1,X-1} - g_{2,X}}{\Delta x \Delta y}, \tag{41g}$$

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{Y,1} = \frac{g_{Y-1,2} + g_{Y,1} - g_{Y-1,1} - g_{Y,2}}{\Delta x \Delta y}, \tag{41h}$$

and

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{Y,X} = \frac{g_{Y-1,X} + g_{Y,X-1} - g_{Y,X} - g_{Y-1,X-1}}{\Delta x \Delta y}. \tag{41i}$$

To obtain a two-dimensional matrix $\boldsymbol{D}^{\partial^b/\partial a^b}$, where $\boldsymbol{D}^{\partial^b/\partial a^b}\boldsymbol{g} = (\partial^b \boldsymbol{g}/\partial a^b)$, a is x or y and b indicates the order of the derivative, these indices (x,y) have to be translated into a one-dimensional numbering that corresponds to the numbering of the vector $\boldsymbol{g}$. From the expressions above, the nonzero elements $D_{ij}^{\partial^b/\partial a^b}$ are easily found by $i = (y-1)X + x$ with the indices (x,y) of $\boldsymbol{D}^{\partial^b/\partial a^b}$, and j similarly from the subscripts of $\boldsymbol{g}$ on the right-hand-side. For example, the derivative matrix $\boldsymbol{D}^{\partial/\partial x}$ can be built from components

$$\left[\boldsymbol{D}^{\partial/\partial x}\right]_y = \frac{1}{2\Delta x}\begin{bmatrix} -2 & 2 & 0 & \cdots & & & & & \\ -1 & 0 & 1 & 0 & \cdots & & & & \\ 0 & -1 & 0 & 1 & 0 & \cdots & & & \\ & & & & & & & & \\ 0 & \cdots & 0 & -1 & 0 & 1 & 0 & \cdots & 0 \\ & & & & & & & & \\ & & & \cdots & 0 & -1 & 0 & 1 & 0 \\ & & & & \cdots & 0 & -1 & 0 & 1 \\ & & & & & \cdots & 0 & -2 & 2 \end{bmatrix}, \tag{42}$$

to give with Y of these matrices

$$\boldsymbol{D}^{\partial/\partial x} = \begin{bmatrix} \left[\boldsymbol{D}^{\partial/\partial x}\right]_1 & \mathbf{0} & \mathrm{L} & \mathbf{0} \\ \mathbf{0} & \left[\boldsymbol{D}^{\partial/\partial x}\right]_2 & \mathbf{0} & \mathrm{M} \\ \mathrm{M} & & \mathrm{O} & \mathbf{0} \\ \mathbf{0} & \mathrm{L} & \mathbf{0} & \left[\boldsymbol{D}^{\partial/\partial x}\right]_Y \end{bmatrix}. \tag{43}$$

Note that the elements on the rows add up to zero. Because of the order of the one-dimensional numbering of the grid points, the expressions for the y-derivative matrices is more scrambled and cannot be described with simple submatrices.

Similarly, for the matrix of the second derivative to x, with the first alternative expression in Eqs. (39b–c), one finds:

$$\left[\boldsymbol{D}^{\partial^2/\partial x^2}\right]_y = \frac{1}{2(\Delta x)^2} \begin{bmatrix} 1 & -2 & 1 & 0 & \mathrm{L} & & & & \\ 2 & -4 & 2 & 0 & \mathrm{L} & & & & \\ 0 & 2 & -4 & 2 & 0 & \mathrm{L} & & & \\ & & & & & & & & \\ 0 & \mathrm{L} & 0 & 2 & -4 & 2 & 0 & \mathrm{L} & 0 \\ & & & & & & & & \\ & & & \mathrm{L} & 0 & 2 & -4 & 2 & 0 \\ & & & & \mathrm{L} & 0 & 2 & -4 & 2 \\ & & & & \mathrm{L} & 0 & 1 & -2 & 1 \end{bmatrix}, \tag{44}$$

and with the second alternative expression in Eqs. (39b–c)

$$\left[\boldsymbol{D}^{\partial^2/\partial x^2}\right]_y = \frac{1}{(\Delta x)^2} \begin{bmatrix} -1 & 1 & 0 & \mathrm{L} & & & & & \\ 1 & -2 & 1 & 0 & \mathrm{L} & & & & \\ 0 & 1 & -2 & 1 & 0 & \mathrm{L} & & & \\ & & & & & & & & \\ 0 & \mathrm{L} & 0 & 1 & -2 & 1 & 0 & \mathrm{L} & 0 \\ & & & & & & & & \\ & & & \mathrm{L} & 0 & 1 & -2 & 1 & 0 \\ & & & & \mathrm{L} & 0 & 1 & -2 & 1 \\ & & & & & \mathrm{L} & 0 & -1 & 1 \end{bmatrix}. \tag{45}$$

2.5.2 Implementation with variable grid size

When implementing the derivative matrices on a grid with variable grid size one has to take care that the implementation properly describes the discrete derivative. For the $\partial/\partial x$, $\partial/\partial y$, $\partial^2/\partial x^2$ and $\partial^2/\partial y^2$ this is relatively straightforward. One has to consider the two neighbouring actual or virtual grid points of the actual point (*x*,*y*) under consideration, which can have varying distances to the actual point (*x*,*y*). Thus, $\partial/\partial x$ can be implemented with the first term of Eq. (37a) with two different Δx, $\partial^2/\partial x^2$ with three different Δx in the first term in Eq. (39a), and likewise for $\partial/\partial y$ and $\partial^2/\partial y^2$. The values found are allocated to the neighbouring points directly if they are actual. If a neighbouring point is virtual, the a fraction of the value is allocated to the two neighbouring points of the virtual point[11].

The derivative $\partial^2/\partial x\partial y$ is more complicated as the grid sizes may vary in all four directions, which cannot be properly taken into account in Eq. (41a). This is clear if Eq. (41a) is written out with grid sizes:

$$\left(\frac{\partial^2 g}{\partial x \partial y}\right)_{y,x} = \frac{\dfrac{\left(\frac{\partial g}{\partial y}\right)_{y,x} - \left(\frac{\partial g}{\partial y}\right)_{y,x-1}}{x_{y,x}-x_{y,x-1}} + \dfrac{\left(\frac{\partial g}{\partial y}\right)_{y,x+1} - \left(\frac{\partial g}{\partial y}\right)_{y,x}}{x_{y,x+1}-x_{y,x}}}{2} =$$

$$\frac{\dfrac{g_{y-1,x}-g_{y,x}}{2(y_{y-1,x}-y_{y,x})} + \dfrac{g_{y,x}-g_{y+1,x}}{2(y_{y,x}-y_{y+1,x})} - \dfrac{g_{y-1,x-1}-g_{y,x-1}}{2(y_{y-1,x-1}-y_{y,x-1})} - \dfrac{g_{y,x-1}-g_{y+1,x-1}}{2(y_{y,x-1}-y_{y+1,x-1})}}{2(x_{y,x}-x_{y,x-1})} +$$

$$\frac{\dfrac{g_{y-1,x+1}-g_{y,x+1}}{2(y_{y-1,x+1}-y_{y,x+1})} + \dfrac{g_{y,x+1}-g_{y+1,x+1}}{2(y_{y,x+1}-y_{y+1,x+1})} - \dfrac{g_{y-1,x}-g_{y,x}}{2(y_{y-1,x}-y_{y,x})} - \dfrac{g_{y,x}-g_{y+1,x}}{2(y_{y,x}-y_{y+1,x})}}{2(x_{y,x+1}-x_{y,x})}.$$

(46)

For this expression to be valid, it must for example be the case that $x_{y,x}-x_{y,x-1}$ in the first term is the same as $x_{y+1,x}-x_{y+1,x-1}$ and $x_{y-1,x}-x_{y-1,x-1}$, which is not necessarily the case. Furthermore, the result should be independent of the order of derivatives, i.e. $\partial/\partial x(\partial/\partial y) = \partial/\partial y(\partial/\partial x)$. For these reasons, a different technique than for the other derivatives had to be developed to calculate the mixed second derivative on a grid with variable grid size. Instead of the actual and virtual grid points, the *original* grid points are used for Eq. (46), which ensures that correct values for Δx and Δy are used. Note that the surrounding original grid points will always lie in the four pixels surrounding the point, as the original grid determines the smallest

11 See footnote 9 on page 12.

possible dimension. Equation (46) gives the factors that should be assigned to the eight points surrounding the point in question. These values are then distributed over the four corner points of the four pixels in which they lie according to bi-linear interpolation (i.e. bi-linear interpolation of the corner points should give back the values[12]); if one of the corner points is virtual, the value is spread over its two neighbours. If one considers non-edge points, most terms in Eq. (46) will cancel each other: only the four diagonal points given by Eq. (41a) remain. For edge points some terms of Eq. (46) are indeterminate and have to be left out, as on the edge there can be one, two or three pixels surrounding the point. The values are assigned to these pixels in the same way as for four pixels, but from Eq. (46) a weight of 1, $\frac{1}{2}$ or $\frac{1}{3}$ follows instead of $\frac{1}{4}$. Furthermore, in addition to diagonal points, values are assigned to points on pixel edges as these do not cancel out in Eq. (46) in that case.

2.6 Isotropic and anisotropic unsmoothness

The object function in constrained optimization should be chosen appropriately for the expected types of solutions $g(x,y)$. In the literature, a requirement of smoothness is common, which involves second derivatives. Depending on the coordinate system, the functional describing unsmoothness can be chosen to be isotropic or anisotropic, and can also include other derivatives. A general expression in Cartesian coordinates is given in Sec. 2.6.1. Fuchs [12] introduced an objective function that describes anisotropic unsmoothness in flux coordinates, given by a diffusive type of functional. The mathematical expressions needed to implement that functional in Cartesian coordinates are given in Sec. 2.6.2.

2.6.1 Ordinary unsmoothness

The smoothness operator should quantify the unsmoothness of the function $\boldsymbol{g}$, and can be expressed by a scalar product

$$\begin{aligned}\langle \boldsymbol{g}|\boldsymbol{\Omega}|\boldsymbol{g}\rangle = \iint [c_0 g^2(x,y) + c_x\left(\frac{\partial g}{\partial x}\right)^2 + c_y\left(\frac{\partial g}{\partial y}\right)^2 + \\ c_{xx}\left(\frac{\partial^2 g}{\partial x^2}\right)^2 + 2c_{xy}\left(\frac{\partial^2 g}{\partial x \partial y}\right)^2 + c_{yy}\left(\frac{\partial^2 g}{\partial y^2}\right)^2]\, \mathrm{d}x\, \mathrm{d}y.\end{aligned} \tag{47}$$

The parameters c can be chosen to suit the particular problem, can be chosen differently for different directions, i.e. leading to anisotropic smoothness, and they can be chosen to be functions of x and y. The c_0 term forces the solution to be as small as possible in all points where c_0 is not zero, the first derivative terms force the solution to be as flat as possible, whereas the second

12 Some examples. If the point is in the middle of the pixel, each corner is assigned 1/4 of the value. If the point is $\Delta x/4$ in x and $2\Delta y/3$ in y, the fraction of the values assigned to the corners are (1/4)×(2/3), (3/4)×(2/3), (1/4)×(1/3) and (3/4)×(1/3).

derivative terms force it to be smooth. Often only the second derivative terms are used because one is searching for the smoothest function. The Cartesian coordinate system used in Eq. (47) only serves as an example: of course any coordinate system can be used in the definition of $\boldsymbol{\Omega}$ by a scalar product, as long as it is a valid definition and a valid scalar product. The coordinate system can be chosen to suit the particular problem. The derivatives can be implemented numerically in terms of the derivative matrices given in Sec. 2.4.1, so that the matrix $\boldsymbol{\Omega}$ of Eq. (47) is given by

$$\begin{aligned}\boldsymbol{\Omega} = \Big(& c_0 \boldsymbol{I} + c_x \boldsymbol{D}^{\partial/\partial x \mathrm{T}} \boldsymbol{D}^{\partial/\partial x} + c_y \boldsymbol{D}^{\partial/\partial y \mathrm{T}} \boldsymbol{D}^{\partial/\partial y} + c_{xx} \boldsymbol{D}^{\partial^2/\partial x^2 \mathrm{T}} \boldsymbol{D}^{\partial^2/\partial x^2} + \\ & 2c_{xy} \boldsymbol{D}^{\partial^2/\partial x \partial y \mathrm{T}} \boldsymbol{D}^{\partial^2/\partial x \partial y} + c_{yy} \boldsymbol{D}^{\partial^2/\partial y^2 \mathrm{T}} \boldsymbol{D}^{\partial^2/\partial y^2} \Big) \Delta x \Delta y,\end{aligned} \tag{48}$$

where $\boldsymbol{I}$ is the identity matrix. If the coefficients c depend on x and y, the c of Eq. (48) must be replaced by diagonal matrices that have the values of c in the grid points on the diagonal. To give an impression of the structure of the smoothness matrix, here the first and second derivative parts to x are given explicitly. With the expressions like Eq. (43) one obtains

$$\left[\boldsymbol{\Omega}^{\partial/\partial x}\right]_y = c_x \frac{\Delta y}{4\Delta x} \begin{bmatrix} 5 & -4 & -1 & 0 & \cdots & & & & & & \\ -4 & 5 & 0 & -1 & 0 & \cdots & & & & & \\ -1 & 0 & 2 & 0 & -1 & 0 & \cdots & & & & \\ & & & & & & & & & & \\ 0 & \cdots & 0 & -1 & 0 & 2 & 0 & -1 & 0 & \cdots & 0 \\ & & & & & & & & & & \\ & & & & \cdots & 0 & -1 & 0 & 2 & 0 & -1 \\ & & & & & \cdots & 0 & -1 & 0 & 5 & -4 \\ & & & & & & \cdots & 0 & -1 & -4 & 5 \end{bmatrix}. \tag{49}$$

Note that the matrix is symmetrical; this is the case because it is the product of the transpose of a matrix with a matrix itself. The complete matrix $\boldsymbol{\Omega}^{\partial/\partial x}$ consists of Y of these submatrices on the diagonal:

$$\boldsymbol{\Omega}^{\partial/\partial x} = \begin{bmatrix} \left[\boldsymbol{\Omega}^{\partial/\partial x}\right]_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \left[\boldsymbol{\Omega}^{\partial/\partial x}\right]_2 & \mathbf{0} & \vdots \\ \vdots & & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \left[\boldsymbol{\Omega}^{\partial/\partial x}\right]_Y \end{bmatrix}. \tag{50}$$

For the second derivative matrix one finds with Eq. (45) the submatrix

$$\left[\boldsymbol{\Omega}^{\partial^2/\partial x^2}\right]_y = c_{xx}\frac{\Delta y}{(\Delta x)^3}\begin{bmatrix} 2 & -3 & 1 & 0 & \mathrm{L} & & & & & & \\ -3 & 6 & -4 & 1 & 0 & \mathrm{L} & & & & & \\ 1 & -4 & 6 & -4 & 1 & 0 & \mathrm{L} & & & & \\ & & & & & & & & & & \\ 0 & \mathrm{L} & 0 & 1 & -4 & 6 & -4 & 1 & 0 & \mathrm{L} & 0 \\ & & & & & & & & & & \\ & & & & \mathrm{L} & 0 & 1 & -4 & 6 & -4 & 1 \\ & & & & & \mathrm{L} & 0 & 1 & -4 & 6 & -3 \\ & & & & & & \mathrm{L} & 0 & 1 & -3 & 2 \end{bmatrix}, \tag{51}$$

If, instead, one uses Eq. (44) the upper four rows of the matrix are

$$\begin{bmatrix} \frac{5}{4} & -\frac{5}{2} & \frac{5}{4} & 0 & \mathrm{L} & & & \\ -\frac{5}{2} & 6 & -\frac{9}{2} & 1 & 0 & \mathrm{L} & & \\ \frac{5}{4} & -\frac{9}{2} & \frac{25}{4} & -4 & 1 & 0 & \mathrm{L} & \\ 0 & 1 & -4 & 6 & -4 & 1 & 0 & \mathrm{L} \\ & & & & & & & \end{bmatrix}. \tag{52}$$

2.6.2 Anisotropic unsmoothness in flux coordinates

The diffusive anisotropic unsmoothness in flux coordinates can be defined similarly to Eq. (47) [12]:

$$\langle \boldsymbol{g}|\boldsymbol{\Omega}|\boldsymbol{g}\rangle = \iint \left\{\nabla \bullet \left[\boldsymbol{D} \bullet \nabla g(x,y)\right]\right\}^2 \mathrm{d}x\,\mathrm{d}y. \tag{53}$$

This means that instead of the smoothest solution, the solution that would give the least diffusion is obtained; in both cases the solution is chosen from the set of solutions that fit the data. Note that this diffusion is diffusion of the emissivity in the mathematical sense, and is only remotely related to physical diffusion in the plasma. Here $\boldsymbol{D}$ is an anisotropic diffusion tensor, which can be chosen to vary as a function of position. Defining the diffusion by the diffusion coefficients $D_\perp$ and $D_{//}$, perpendicular and parallel to the flux surfaces, Eq. (53) can be written as

$$\iint \left[\boldsymbol{\nabla} \bullet \left(\boldsymbol{n} D_\perp \boldsymbol{n} \bullet \boldsymbol{\nabla} g + \boldsymbol{t} D_{//} \boldsymbol{t} \bullet \boldsymbol{\nabla} g\right)\right]^2 \mathrm{d}x\,\mathrm{d}y, \tag{54}$$

where $\boldsymbol{n}$ and $\boldsymbol{t}$ are the unit vectors normal and tangential to the flux surfaces in the poloidal cross-section. Choosing Cartesian coordinates, Eq. (54) can be written very similarly to Eq. (47):

$$\iint\left[c_x\left(\frac{\partial g}{\partial x}\right)+c_y\left(\frac{\partial g}{\partial y}\right)+c_{xx}\left(\frac{\partial^2 g}{\partial x^2}\right)+2c_{xy}\left(\frac{\partial^2 g}{\partial x\partial y}\right)+c_{yy}\left(\frac{\partial^2 g}{\partial y^2}\right)\right]^2 \mathrm{d}x\,\mathrm{d}y, \tag{55}$$

where the coefficients *c* are complicated functions of the diffusion coefficients and the flux function (see below). Note that they are functions of *x* and *y* because the diffusion tensor is a function of the position. It is interesting to note that the requirement of minimum *diffusion* in flux coordinates (i.e. defined by a second derivative operator) requires *flatness* and smoothness in Cartesian coordinates (because of the nonzero c_x and c_y terms). There is a danger involved in using Cartesian coordinates. Because derivatives of the flux function are taken with respect to the Cartesian coordinates on a relatively coarse grid one can expect that numerical errors could occur if either ***n*** and ***t*** are almost parallel to the Cartesian axes (i.e. there is an averaging over the grid size, and the error of averaging could be large if there is much change over one pixel). From experience this seems not to be a large problem, except at the magnetic axis where precautions have to be taken (near the magnetic axis $D_\perp = D_{//}$ is taken). The alternative would have been to describe the entire algorithm in flux coordinates, which, however, would complicate the evaluation of line integrals (along curved paths) and the geometric matrix.

Note that the *c* coefficients are functions of the position. Therefore, and because of the square in the integrand of Eqs. (53–55) which generates 25 terms instead of five, the filling of the geometric matrix is somewhat more complicated than before, but can conveniently be written in a similar form as Eq. (48):

$$\boldsymbol{\Omega}=\left(\boldsymbol{C}^{\partial/\partial x}\boldsymbol{D}^{\partial/\partial x}+\boldsymbol{C}^{\partial/\partial y}\boldsymbol{D}^{\partial/\partial y}+\boldsymbol{C}^{\partial^2/\partial x^2}\boldsymbol{D}^{\partial^2/\partial x^2}+2\boldsymbol{C}^{\partial^2/\partial x\partial y}\boldsymbol{D}^{\partial^2/\partial x\partial y}+\boldsymbol{C}^{\partial^2/\partial y^2}\boldsymbol{D}^{\partial^2/\partial y^2}\right)^{\mathsf{T}}\bullet$$
$$\left(\boldsymbol{C}^{\partial/\partial x}\boldsymbol{D}^{\partial/\partial x}+\boldsymbol{C}^{\partial/\partial y}\boldsymbol{D}^{\partial/\partial y}+\boldsymbol{C}^{\partial^2/\partial x^2}\boldsymbol{D}^{\partial^2/\partial x^2}+2\boldsymbol{C}^{\partial^2/\partial x\partial y}\boldsymbol{D}^{\partial^2/\partial x\partial y}+\boldsymbol{C}^{\partial^2/\partial y^2}\boldsymbol{D}^{\partial^2/\partial y^2}\right)\times$$
$$\Delta x\Delta y, \tag{56}$$

where the ***D*** are derived from Eqs. (37–41) and the ***C*** are diagonal matrices with the values *c* from Eqs. (59) below for each grid point on the diagonal.

Next, the position-dependent coefficients *c* are derived. The coordinates chosen are the poloidal Cartesian coordinates major radius *R* and vertical distance *Z*, and the toroidal angle *ϕ*.[13] To be consistent with the general notation, *R* is replaced by *x* and *Z* by *y*, and *ϕ* does not appear explicitly. The unit vectors normal and tangential to the flux surface are

$$\boldsymbol{n}=\frac{\nabla\psi}{\|\nabla\psi\|}=\frac{1}{\|\nabla\psi\|}\left(\frac{\partial\psi}{\partial x}\boldsymbol{e}_x+\frac{\partial\psi}{\partial y}\boldsymbol{e}_y\right)\quad\text{and}\quad\boldsymbol{t}=\frac{1}{\|\nabla\psi\|}\left(\frac{\partial\psi}{\partial y}\boldsymbol{e}_x-\frac{\partial\psi}{\partial x}\boldsymbol{e}_y\right), \tag{57}$$

13 The nabla operator is $\nabla = \boldsymbol{e}_R(\partial/\partial R) + \boldsymbol{e}_Z(\partial/\partial Z) + \boldsymbol{e}_\phi(1/R)(\partial/\partial\phi)$, or $\nabla = \boldsymbol{e}_x(\partial/\partial x) + \boldsymbol{e}_y(\partial/\partial y)$ because the function is toroidally symmetric.

respectively, where ψ is the flux function and ***e*** the axis unit vector, and the normalization factor is simply:

$$\|\nabla\psi\|^2 = \left(\frac{\partial\psi}{\partial x}\right)^2 + \left(\frac{\partial\psi}{\partial y}\right)^2 . \tag{58}$$

The signs in the direction of ***n*** and ***t*** is not important in the following. The coefficients *c* (in each coordinate point) in Eq. (55) follow from substitution of these expressions for ***n*** and ***t*** into Eq. (54) and a cumbersome rearrangement terms:

$$c_{xx} = \frac{1}{\|\nabla\psi\|^2}\left[D_\perp\left(\frac{\partial\psi}{\partial x}\right)^2 + D_{//}\left(\frac{\partial\psi}{\partial y}\right)^2\right], \tag{59a}$$

$$c_{yy} = \frac{1}{\|\nabla\psi\|^2}\left[D_\perp\left(\frac{\partial\psi}{\partial y}\right)^2 + D_{//}\left(\frac{\partial\psi}{\partial x}\right)^2\right], \tag{59b}$$

$$c_{xy} = \frac{1}{\|\nabla\psi\|^2}\left(D_\perp - D_{//}\right)\left(\frac{\partial\psi}{\partial x}\frac{\partial\psi}{\partial y}\right), \tag{59c}$$

$$\begin{aligned} c_x = \frac{1}{\|\psi\|^2}\Bigg[& 2D_\perp\frac{\partial^2\psi}{\partial x^2}\frac{\partial\psi}{\partial x} + 2D_{//}\frac{\partial^2\psi}{\partial x\partial y}\frac{\partial\psi}{\partial y} + \left(D_\perp - D_{//}\right)\left(\frac{\partial^2\psi}{\partial x\partial y}\frac{\partial\psi}{\partial y} + \frac{\partial^2\psi}{\partial y^2}\frac{\partial\psi}{\partial x}\right) \\ & + \left(\partial D \text{ term } c_x\right) + \left(\partial\left(1/\|\nabla\psi\|^2\right) \text{ term } c_x\right) + \left(\text{toroidal term } c_x\right)\Bigg], \end{aligned} \tag{59d}$$

and

$$\begin{aligned} c_y = \frac{1}{\|\psi\|^2}\Bigg[& 2D_\perp\frac{\partial^2\psi}{\partial y^2}\frac{\partial\psi}{\partial y} + 2D_{//}\frac{\partial^2\psi}{\partial x\partial y}\frac{\partial\psi}{\partial x} + \left(D_\perp - D_{//}\right)\left(\frac{\partial^2\psi}{\partial x\partial y}\frac{\partial\psi}{\partial x} + \frac{\partial^2\psi}{\partial x^2}\frac{\partial\psi}{\partial y}\right) \\ & + \left(\partial D \text{ term } c_y\right) + \left(\partial\left(1/\|\nabla\psi\|^2\right) \text{ term } c_y\right) + \left(\text{toroidal term } c_y\right)\Bigg]. \end{aligned} \tag{59e}$$

The derivative terms with *D* and $\nabla\psi$ in Eqs. (59d–e) are:

$$\left(\partial D \text{ term } c_x\right) = \left(\frac{\partial\psi}{\partial x}\right)^2\frac{\partial D_\perp}{\partial x} + \left(\frac{\partial\psi}{\partial y}\right)^2\frac{\partial D_{//}}{\partial x} + \left(\frac{\partial\psi}{\partial x}\frac{\partial\psi}{\partial y}\right)\left(\frac{\partial D_\perp}{\partial y} - \frac{\partial D_{//}}{\partial y}\right), \tag{60a}$$

$$\begin{aligned} \left(\partial\left(1/\|\nabla\psi\|^2\right) \text{ term } c_x\right) = \frac{-2}{\|\nabla\psi\|^2}\Bigg\{ & \left[D_\perp\left(\frac{\partial\psi}{\partial x}\right)^2 + D_{//}\left(\frac{\partial\psi}{\partial y}\right)^2\right]\left(\frac{\partial\psi}{\partial x}\frac{\partial^2\psi}{\partial x^2} + \frac{\partial\psi}{\partial y}\frac{\partial^2\psi}{\partial x\partial y}\right) + \\ & \left(D_\perp - D_{//}\right)\left(\frac{\partial\psi}{\partial x}\frac{\partial\psi}{\partial y}\right)\left(\frac{\partial\psi}{\partial x}\frac{\partial^2\psi}{\partial x\partial y} + \frac{\partial\psi}{\partial y}\frac{\partial^2\psi}{\partial y^2}\right)\Bigg\}, \end{aligned} \tag{60b}$$

$$(\partial D \text{ term } c_y) = \left(\frac{\partial\psi}{\partial y}\right)^2 \frac{\partial D_\perp}{\partial y} + \left(\frac{\partial\psi}{\partial x}\right)^2 \frac{\partial D_{//}}{\partial y} + \left(\frac{\partial\psi}{\partial x}\frac{\partial\psi}{\partial y}\right)\left(\frac{\partial D_\perp}{\partial x} - \frac{\partial D_{//}}{\partial x}\right), \tag{60c}$$

and

$$\left(\partial\left(1/\|\nabla\psi\|^2\right) \text{ term } c_y\right) = \frac{-2}{\|\nabla\psi\|^2}\left\{\left[D_\perp\left(\frac{\partial\psi}{\partial y}\right)^2 + D_{//}\left(\frac{\partial\psi}{\partial x}\right)^2\right]\left(\frac{\partial\psi}{\partial x}\frac{\partial^2\psi}{\partial x\partial y} + \frac{\partial\psi}{\partial y}\frac{\partial^2\psi}{\partial y^2}\right) + \left(D_\perp - D_{//}\right)\left(\frac{\partial\psi}{\partial x}\frac{\partial\psi}{\partial y}\right)\left(\frac{\partial\psi}{\partial x}\frac{\partial^2\psi}{\partial x^2} + \frac{\partial\psi}{\partial y}\frac{\partial^2\psi}{\partial x\partial y}\right)\right\}. \tag{60d}$$

The toroidal terms in Eqs. (59d–e) are due to the change of $\boldsymbol{e}_R$ as a function of toroidal angle ϕ[14]. They are given by

$$\left(\text{toroidal term } c_x\right) = \frac{c_{xx}}{R}\|\nabla\psi\|^2 \text{ and } \left(\text{toroidal term } c_y\right) = \frac{c_{xy}}{R}\|\nabla\psi\|^2, \tag{61}$$

where the major radius R is written explicitly (it is equivalent to x in the chosen coordinates). Note that $\|\nabla\psi\|^2$ in Eq. (61) is to compensate for the normalization in Eqs. (59d) and (59e).

The derivatives in Eqs. (59a–e) can be numerically implemented by the derivative operators derived in Sec. 2.5.1. To do this, the vectors ψ, $\boldsymbol{D}_\perp$ and $\boldsymbol{D}_{//}$ in the grid points are formed with the same numbering of elements as those of $\boldsymbol{g}$ in Eqs. (37)–(41). The derivative $\partial^2\boldsymbol{\psi}/\partial x^2$ in grid point i, for example, is given by $(\boldsymbol{D}^{\partial^2/\partial x^2}\boldsymbol{\psi})_i$.

2.7 Analytical solution of constrained-optimization problem

As said in Sec. 2.1, Eqs. (8) and (10) can be solved by iteratively varying the Lagrange multiplier λ and solving the matrix equation. It is also possible to construct an analytic solution as a function of λ [14,15], which is discussed in this section. In this framework it is also possible to implement additional constraints on the solution $\boldsymbol{g}$, for example that it must be positive $g_j \geq 0$; additional Lagrange multipliers $\boldsymbol{l}$ are introduced for each constraint. Thus, this is a quadratic optimization problem with a quadratic constraint and bounds on the variables $\boldsymbol{g}$. More detailed derivations can be found in Ref. 14 and 15.

Equation (8) or (10) and the Lagrange multipliers should be solved, given the constraints. The following shorthand notation will be used: $\boldsymbol{C} = \boldsymbol{K}^\mathrm{T}\boldsymbol{W}\boldsymbol{K}$, $\boldsymbol{c} = -\boldsymbol{K}^\mathrm{T}\boldsymbol{W}\boldsymbol{f}$ and $\boldsymbol{B} = \boldsymbol{\Omega}$. An additional vector $\boldsymbol{b}$ can be introduced in the object function as a generalization [14,15] to give $O(\boldsymbol{g}) = \langle \boldsymbol{g}|\boldsymbol{B}|\boldsymbol{g}\rangle + 2\langle \boldsymbol{b}|\boldsymbol{g}\rangle$, in the present application $\boldsymbol{b} = \boldsymbol{0}$. Equation (8) or (10), including the constraints on $\boldsymbol{g}$, can now be written as

14 In cylindrical coordinates around the major axis: $\nabla\cdot\boldsymbol{h} = (1/R)(\partial/\partial R)(R\,h_R) + (\partial/\partial Z)h_Z + (1/R)(\partial/\partial\phi)h_\phi$.

$$(\boldsymbol{B}+\lambda\boldsymbol{C})\boldsymbol{g}=\boldsymbol{l}-\boldsymbol{b}-\lambda\boldsymbol{c}. \tag{62}$$

It is possible to solve Eq. (62) formally by considering a generalized eigenvalue problem [14,15]. A solution exists for $\lambda > 0$ if $\boldsymbol{B}$ and $\boldsymbol{C}$ are both positive semidefinite. The solution is given for three different cases in the following three subsections: $\boldsymbol{B}$ and $\boldsymbol{C}$ symmetric and positive semidefinite, the matrix $\boldsymbol{C}$ is positive definite, and the matrix $\boldsymbol{B}$ is positive definite. The first is the general case (which will generally be true for the tomography problem), the other ones are simplifications if either of the matrices is positive definite. If the object function does not only include the smoothness property, but also forces values to be small on the reconstruction boundary [i.e. it has the term $c_0\boldsymbol{I}$ in Eq. (48) which penalizes large values], $\boldsymbol{B}$ is likely to be positive definite. This is what is actually used at present (i.e. the formulae of Sec. 2.7.3).

2.7.1 Both matrices $\boldsymbol{C}$ and $\boldsymbol{B}$ are positive semidefinite

This section is based on Ref. 14. It is required that a solution exists and that $\ker(\boldsymbol{B})\cap\ker(\boldsymbol{C})=\{\boldsymbol{0}\}$, i.e. the subspace of solution space that goes unnoticed by the experiment [ker($\boldsymbol{C}$)] does not overlap with the subspace towards which the *a priori* information is indifferent [ker($\boldsymbol{B}$)]. If the matrices $\boldsymbol{C}$ and $\boldsymbol{B}$ are positive semidefinite, $\boldsymbol{C}+\boldsymbol{B}$ is positive definite and $(\boldsymbol{C}+\boldsymbol{B})^{-1/2}$ exists and the inverse of the matrix in Eq. (62) can be written as

$$(\boldsymbol{B}+\lambda\boldsymbol{C})^{-1}=(\boldsymbol{B}+\boldsymbol{C})^{-1/2}\left[\boldsymbol{I}+(\lambda-1)(\boldsymbol{B}+\boldsymbol{C})^{-1/2}\boldsymbol{C}(\boldsymbol{B}+\boldsymbol{C})^{-1/2}\right]^{-1}(\boldsymbol{B}+\boldsymbol{C})^{-1/2}, \tag{63}$$

where $\boldsymbol{I}$ is the identity matrix and by definition $(\boldsymbol{C}+\boldsymbol{B})^{1/2}(\boldsymbol{C}+\boldsymbol{B})^{1/2}=\boldsymbol{C}+\boldsymbol{B}$.

Defining the generalized eigenvalue problem

$$\boldsymbol{C}|\boldsymbol{y}_k\rangle=\mu_k(\boldsymbol{B}+\boldsymbol{C})|\boldsymbol{y}_k\rangle, \tag{64}$$

and

$$|\boldsymbol{y}_k\rangle=(\boldsymbol{B}+\boldsymbol{C})^{-1/2}|\boldsymbol{u}_k\rangle, \tag{65}$$

where $|\boldsymbol{u}_k\rangle$ are the eigenvectors of

$$(\boldsymbol{B}+\boldsymbol{C})^{-1/2}\boldsymbol{C}(\boldsymbol{B}+\boldsymbol{C})^{-1/2}|\boldsymbol{u}_k\rangle=\mu_k|\boldsymbol{u}_k\rangle, \tag{66}$$

one can show that

$$\left[\boldsymbol{I}+(\lambda-1)(\boldsymbol{B}+\boldsymbol{C})^{-1/2}\boldsymbol{C}(\boldsymbol{B}+\boldsymbol{C})^{-1/2}\right]^{-1}=\sum_{k=1}^{J}\frac{|\boldsymbol{u}_k\rangle\langle\boldsymbol{u}_k|}{1+(\lambda-1)\mu_k}, \tag{67}$$

where the summation is over the grid points. With Eq. (63), Eq. (67) gives

$$|\boldsymbol{g}(\lambda)\rangle=\sum_{k=1}^{J}\frac{\langle\boldsymbol{y}_k|\boldsymbol{l}-\boldsymbol{b}\rangle-\lambda\langle\boldsymbol{y}_k|\boldsymbol{c}\rangle}{1+(\lambda-1)\mu_k}|\boldsymbol{y}_k\rangle. \tag{68}$$

This is the solution without imposed constraints, i.e. a function of λ. Given the $\boldsymbol{l}$, the λ that solves the *constrained* optimization problem is the root of

$$C(\boldsymbol{g}(\lambda)) = \langle \boldsymbol{f}|\boldsymbol{W}|\boldsymbol{f}\rangle - \langle \varepsilon|\boldsymbol{W}|\varepsilon\rangle + \sum_{k=1}^{J} \frac{[\lambda\langle \boldsymbol{c}|\boldsymbol{y}_k\rangle + \langle \boldsymbol{b}-\boldsymbol{l}|\boldsymbol{y}_k\rangle][(2\mu_k - 2 - \lambda\mu_k)\langle \mathrm{c}|\mathrm{y}_k\rangle + \mu_k\langle \boldsymbol{b}-\boldsymbol{l}|\boldsymbol{y}_k\rangle]}{\left(1+(\lambda-1)\mu_k\right)^2} = 0. \tag{69}$$

An equivalent solution can be derived by generalized singular value decomposition [14]. It can be shown that, for $\lambda > 0$, $C(\boldsymbol{g}(\lambda))$ is a monotonically decreasing function with λ [14], i.e. there is a unique solution, which means that Eq. (69) can be solved numerically in a straightforward way. To obtain the solution of Eq. (8) or (10) it is thus necessary to solve the generalized eigenvalue problem Eq. (64) numerically, and use the eigenvectors $|\boldsymbol{y}_k\rangle$ and eigenvalues μ_k in Eq. (69) to find λ.

*2.7.2 The matrix **C** is positive definite*

This section is based on Ref. 15. If the matrix $\boldsymbol{B}$ is positive semidefinite, and $\boldsymbol{C}$ is positive definite, $\boldsymbol{C}^{-1/2}$ exists and the inverse of the matrix in Eq. (62) can be written as

$$(\boldsymbol{B}+\lambda\boldsymbol{C})^{-1} = \boldsymbol{C}^{-1/2}\left[\lambda\boldsymbol{I}+\boldsymbol{C}^{-1/2}\boldsymbol{B}\boldsymbol{C}^{-1/2}\right]^{-1}\boldsymbol{C}^{-1/2}. \tag{70}$$

Equations (64)–(69) in this case become

$$\boldsymbol{B}|\boldsymbol{y}_k\rangle = \mu_k\boldsymbol{C}|\boldsymbol{y}_k\rangle, \tag{71}$$

$$|\boldsymbol{y}_k\rangle = \boldsymbol{C}^{-1/2}|\boldsymbol{u}_k\rangle, \tag{72}$$

$$\boldsymbol{C}^{-1/2}\boldsymbol{B}\boldsymbol{C}^{-1/2}|\boldsymbol{u}_k\rangle = \mu_k|\boldsymbol{u}_k\rangle, \tag{73}$$

$$\left[\lambda\boldsymbol{I}+\boldsymbol{C}^{-1/2}\boldsymbol{B}\boldsymbol{C}^{-1/2}\right]^{-1} = \sum_{k=1}^{J}\frac{|\boldsymbol{u}_k\rangle\langle\boldsymbol{u}_k|}{\lambda+\mu_k}, \tag{74}$$

$$|\boldsymbol{g}(\lambda)\rangle = -\sum_{k=1}^{J}\frac{\langle\boldsymbol{y}_k|\boldsymbol{l}-\boldsymbol{b}\rangle + \lambda\langle\boldsymbol{y}_k|\boldsymbol{c}\rangle}{\lambda+\mu_k}|\boldsymbol{y}_k\rangle, \tag{75}$$

and

$$C(\boldsymbol{g}(\lambda)) = \langle \boldsymbol{f}|\boldsymbol{W}|\boldsymbol{f}\rangle - \langle \varepsilon|\boldsymbol{W}|\varepsilon\rangle + \sum_{k=1}^{J}\frac{-(\lambda^2+2\mu_k\lambda)\langle\boldsymbol{c}|\boldsymbol{y}_k\rangle^2 + \langle\boldsymbol{b}-\boldsymbol{l}|\boldsymbol{y}_k\rangle^2 - 2\mu_k\langle\boldsymbol{c}|\boldsymbol{y}_k\rangle\langle\boldsymbol{b}-\boldsymbol{l}|\boldsymbol{y}_k\rangle}{(\lambda+\mu_k)^2} = 0. \tag{76}$$

Thus, if $\boldsymbol{C}$ is positive definite, the solution of Eq. (8) or (10) can be found by solving the generalized eigenvalue problem Eq. (71) numerically and substituting the eigenvalues and eigenvectors in Eq. (76).

*2.7.3 The matrix **B** is positive definite*

This section is based on Refs. 15 and 1. If the matrix $\boldsymbol{C}$ is positive semidefinite, and $\boldsymbol{B}$ is positive definite, $\boldsymbol{B}^{-1/2}$ exists and the inverse of the matrix in Eq. (62) can be written as

$$(\boldsymbol{B}+\lambda\boldsymbol{C})^{-1}=\boldsymbol{B}^{-1/2}\left[\boldsymbol{I}+\lambda\boldsymbol{B}^{-1/2}\boldsymbol{C}\boldsymbol{B}^{-1/2}\right]^{-1}\boldsymbol{B}^{-1/2}. \tag{77}$$

Equations (64)–(69) in this case become

$$\boldsymbol{C}|\boldsymbol{y}_k\rangle=\mu_k\boldsymbol{B}|\boldsymbol{y}_k\rangle, \tag{78}$$

$$|\boldsymbol{y}_k\rangle=\boldsymbol{B}^{-1/2}|\boldsymbol{u}_k\rangle, \tag{79}$$

$$\boldsymbol{B}^{-1/2}\boldsymbol{C}\boldsymbol{B}^{-1/2}|\boldsymbol{u}_k\rangle=\mu_k|\boldsymbol{u}_k\rangle, \tag{80}$$

$$\left[\boldsymbol{I}+\lambda\boldsymbol{B}^{-1/2}\boldsymbol{C}\boldsymbol{B}^{-1/2}\right]^{-1}=\sum_{k=1}^{J}\frac{|\boldsymbol{u}_k\rangle\langle\boldsymbol{u}_k|}{1+\lambda\mu_k}, \tag{81}$$

$$|\boldsymbol{g}(\lambda)\rangle=-\sum_{k=1}^{J}\frac{\langle\boldsymbol{y}_k|\boldsymbol{l}-\boldsymbol{b}\rangle+\lambda\langle\boldsymbol{y}_k|\boldsymbol{c}\rangle}{1+\lambda\mu_k}|\boldsymbol{y}_k\rangle, \tag{82}$$

and

$$C(\boldsymbol{g}(\lambda))=\langle\boldsymbol{f}|\boldsymbol{W}|\boldsymbol{f}\rangle-\langle\varepsilon|\boldsymbol{W}|\varepsilon\rangle+ \\ \sum_{k=1}^{J}\frac{-(\mu_k\lambda^2+2\lambda)\langle\boldsymbol{c}|\boldsymbol{y}_k\rangle^2+\mu_k\langle\boldsymbol{b}-\boldsymbol{l}|\boldsymbol{y}_k\rangle^2-2\langle\boldsymbol{c}|\boldsymbol{y}_k\rangle\langle\boldsymbol{b}-\boldsymbol{l}|\boldsymbol{y}_k\rangle}{(1+\lambda\mu_k)^2}=0. \tag{83}$$

Thus, if $\boldsymbol{B}$ is positive definite, the solution of Eq. (8) or (10) can be found by solving the generalized eigenvalue problem Eq. (78) numerically and substituting the eigenvalues and eigenvectors in Eq. (83).

2.7.4 Constraints

The bounds (constraints) on $\boldsymbol{g}$ can be set to any arbitrary value and can be an upper bound or a lower bound. Usually, a lower zero bound is required in all grid points (i.e. a non-negativity constraint). In special cases, however, also other bounds can be set for certain grid points to prevent artefacts from occurring there.

It would be possible to require the constraint to be “active” in all (required) grid points and solve Eq. (62) with all Lagrange multipliers. This is a non-linear problem and requires special techniques to be solved. However, not all constraints will be violated, and furthermore, because of the object function grid points violating the constraint are likely to occur in clusters. One can therefore try only to have active constraints for grid points that violate the constraint most (negative local minima for instance). The bounds on $\boldsymbol{g}$ can be implemented in an iterative way by means of the analytical expressions in the previous subsections [14,15]. First one solves the problem with $\boldsymbol{l} = \mathbf{0}$. Only the grid points where this initial solution violates the constraints need to be considered, i.e. one only uses a limited set of the Lagrange multipliers l_j. In the grid points *j* that are chosen to have active constraints (local minima for example), one sets Δg_j so that the constraint is not violated any more, in the case of non-negativity constraints: $\Delta g_j = -g_j$. This correction corresponds to Δl_j, which is zero in non-active constraints, which because of linearity follows from Eq. (62):

$$\Delta \boldsymbol{g} = (\boldsymbol{B} + \lambda \boldsymbol{C})^{-1} \Delta \boldsymbol{l}\,, \tag{84}$$

where only the active constraints contribute. One can thus solve Eq. (68), (75) or (82), depending on which solution method one is using, for just the active constraints. These Δl_j give new l_j used to find a new root λ of Eq. (69), (76) or (83). This possibly results in a solution $\boldsymbol{g}$ that violates some other constraints. This process is iterated until a solution is found that satisfies all constraints. Although convergence cannot be guaranteed, especially when an attempt is made to reconstruct inconsistent data for which no positive solution might exist, the method generally works well. However, when the method has difficulties converging and the set of active constraints only grows, sometimes other criteria have to be used to choose new active constraints. This method is not iterative in solution space, but is an iterative search for active constraints in the space of Lagrange multipliers, which can be much faster if the number of active constraints is small.

2.8 Sparse matrix storage

The geometric and smoothness matrices used in tomography are usually very sparse: typically only a few percent of elements are non-zero. This is easy to understand for the geometric matrix: only the grid points close to the line of sight contribute to the line integral along that line. Also smoothness matrices have only a few elements in diagonal bands that are non-zero, as can be seen from Eqs. (49) and (51). The matrices are also very large, of the order of the number of grid points (>1000). Storing only the non-zero elements of the matrices can therefore save a great deal of memory space. Furthermore, an additional benefit is that, if the non-zero elements are suitably stored, in the evaluation of matrix multiplications only terms are considered that do not lead to zero, which can speed up the computation by several orders of magnitude.

To achieve the benefit of the efficient matrix multiplications it was important to choose a storage scheme that can take advantage of the order in which multiplications are done. This means that the non-zero elements should not be heaped at random onto one stack. The possibility that was implemented was storage by row: i.e. for each row the indices and values of non-zero elements are stored. For the matrix $\boldsymbol{B}$: β_m is number of non-zero elements of row *m*, α_{mn} is the index of *n*th non-zero element on row *m*, and $\gamma_{mn} = B_{m,\alpha_{mn}}$ is the value of the *n*th non-zero element on row *m*. In practice, β is a one-dimensional integer array, and α and γ arrays of linked pointer lists, integer and real, respectively.

The following low-level operations were required on this data structure: initialize the matrix $\boldsymbol{B}$ for storage, delete $\boldsymbol{B}$ from memory, add element B_{ij} to $\boldsymbol{B}$, add a value to a particular element, and check for the existence (non-zero value) of a particular element. The mathematical operations that were implemented include: vector multiplication, inner product of vector and matrix, the transpose of a matrix, and various types of matrix multiplications. These can be described as follows. Vector multiplications:

$$(\boldsymbol{B}\boldsymbol{v})_i = \sum_k B_{ik} v_k = \sum_{k=1}^{\beta_i} \gamma_{ik} v_{\alpha_{ik}}, \tag{85}$$

$$\langle \boldsymbol{v} | \boldsymbol{B} \boldsymbol{v} \rangle = \sum_i \sum_k v_i B_{ik} v_k = \sum_i \sum_{k=1}^{\beta_i} v_i \gamma_{ik} v_{\alpha_{ik}}. \tag{86}$$

The transpose of a matrix is obtained by a loop through all non-zero elements of all the rows and putting these in the correct place of the transposed matrix. For the matrix multiplication $\boldsymbol{A} = \boldsymbol{BC}$ it is most efficient to first obtain the sparse storage of the transpose of $\boldsymbol{B}$. The multiplication is done by three nested loops over the rows i, and k and l over the non-zero elements of each row of both matrices (non-primed symbols for $\boldsymbol{B}^{\mathrm{T}}$ and primed for $\boldsymbol{C}$, respectively): each multiplication is added to the relevant element of the new matrix $\boldsymbol{A}$:

$$A_{\alpha_{ik}\alpha'_{il}} \leftarrow A_{\alpha_{ik}\alpha'_{il}} + \gamma_{ik}\gamma'_{il}. \tag{87}$$

If $\boldsymbol{C}$ is a diagonal matrix with diagonal elements C_i, the matrix multiplication $\boldsymbol{A} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{CB}$ is obtained in a similar way (note that one of the two summations disappears because $\boldsymbol{C}$ is diagonal)

$$A_{\alpha_{ik}\alpha_{il}} \leftarrow A_{\alpha_{ik}\alpha_{il}} + \gamma_{ik} C_i \gamma_{il}. \tag{88}$$

3. SOME MATHEMATICAL DETAILS OF OTHER METHODS

This section describes some expressions that are used in other tomography methods. Section 3.1 introduces two types of projection-space coordinates and gives the relationships between these. Section 3.2 shows how the Cormack method can be applied with elliptical coordinates and Sec. 3.3 describes so-called interpolative generalized natural basis functions. Some details of an implementation of the filtered backprojection method are given in Sec. 3.4: explicit expressions for filters are listed, the backprojection operator is discussed and the inverse filter operation is described.

3.1 Projection space coordinates

A line of sight can be parametrized by its angle and a distance to a reference point, see Fig. 4. In tomography, usually the shortest distance p to a chosen origin is used. Coordinates of a so-called projection at angle ϕ, i.e. a function that is given by the line integrals along parallel lines parametrized by p, is much used in tomography. In this case, ϕ is the angle of the normal of the parallel lines and the horizontal axis, and p is signed (i.e. continuous from a negative to a positive value). Sometimes, in particular when the measurements are not along parallel lines of sight, it is more convenient to choose the angle ξ between the line and the horizontal axis. The parameters of the lines of sight form the coordinates of the so-called projection space.

Because the measurement along a pure line of sight does not depend on the direction of the measurement if the reconstruction region is convex, the point (p,ϕ) in projection space is identical to $(-p,\phi+\pi)$. It is therefore sufficient to consider the ranges $p=[-a,a]$, where a is the maximum radius of the reconstruction region, and $\phi=[0,\pi]$. This space can be seen as a Möbius band with a "Möbius periodicity" of π. Alternatively, one can consider the absolute distance $|p|$ and the range $\phi=[0,2\pi]$. In this case one loses information about the continuity of projection space at $p=0$. The same ranges and remarks are valid for (p,ξ) coordinates. In the following the origin (x_0,y_0) was chosen.

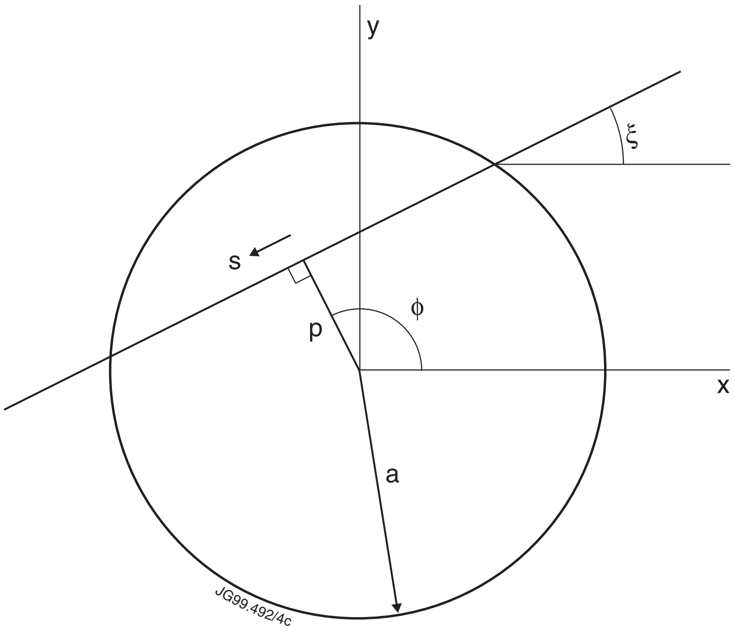

Fig.4: Various definitions of the parametrization of a line of sight.

The definition of the (p,ξ) coordinates of a line is as follows. Given two points on the line (x_1,y_1) and (x_2,y_2):

$$\xi=\arctan\left(\frac{y_2-y_1}{x_2-x_1}\right), \tag{89}$$

and p from the equation for the line:

$$p+(x-x_0)\sin\xi-(y-y_0)\cos\xi=0. \tag{90}$$

In polar coordinates around the origin Eq. (90) is

$$p=r\sin(\theta-\xi). \tag{91}$$

The line can be parametrized with parameter s as

$$x = x_0 - p\sin\xi + s\cos\xi, \tag{92a}$$

$$y = y_0 + p\cos\xi + s\sin\xi. \tag{92b}$$

The ϕ are related to ξ by $\phi = \xi + \pi/2$, and thus $\sin\xi = -\cos\phi$ and $\cos\xi = \sin\phi$.

For (p,ϕ) coordinates the same relations as above are then:

$$\phi = \arctan\left(-\frac{x_2 - x_1}{y_2 - y_1}\right), \tag{93}$$

and p from the equation for the line:

$$p - (x - x_0)\cos\phi - (y - y_0)\sin\phi = 0. \tag{94}$$

In polar coordinates

$$p = r\cos(\theta - \phi). \tag{95}$$

The line parametrized with parameter s:

$$x = x_0 + p\cos\phi + s\sin\phi, \tag{96a}$$

$$y = y_0 + p\sin\phi - s\cos\phi. \tag{96b}$$

It is easy to translate (p,ξ) to (p,ϕ) coordinates and vice versa with $\phi = \xi + \pi/2$ and the Möbius periodicity. Regularly, the coordinates $(|p|,\phi)$ are used, for example in the Cormack method. In that case, provided the ranges specified above are used, the translation is:

$$\begin{aligned} \phi &= \begin{cases} \xi + \pi/2 & \text{if } p \geq 0, \\ \xi - \pi/2 \bmod 2\pi & \text{if } p < 0, \end{cases} \\ |p| &= |p|, \end{aligned} \tag{97}$$

and

$$(p,\xi) = \begin{cases} -|p| & \phi - 3\pi/2 & \text{for } \phi > 3\pi/2, \\ -|p| & \phi + \pi/2 & \text{for } \phi < \pi/2, \\ |p| & \phi - \pi/2 & \text{for } \pi/2 \leq \phi \leq 3\pi/2. \end{cases} \tag{98}$$

If the reconstruction region is not convex, some lines of sight may be blocked by the boundary, which usually will be a solid wall. This is the case, for example, in tokamaks with a closed divertor. In that case the measurement is not independent of the direction and the simple picture of projection space breaks down. One has to extend projection space to the ranges $p = [-a,a]$ and $\phi = [0,2\pi]$; for most lines of sight the point $(-p,\phi+\pi)$ will still be identical to (p,ϕ), but in "shadow regions" it will not and both points need to be considered separately. Projection-space coverage when the reconstruction region is concave is discussed in detail in

Ref. 21. If measurements exist from both directions, they can be summed and used in conventional tomographic reconstruction methods; if such measurements do not exist one may have to resort to reconstruction algorithms that do not operate in projection space. If beam-width effects are important, the measurements from opposite directions are also likely to be different. That fact can be handled in projection space with the technique described in Ref. 9.

3.2 Cormack method in elliptical coordinates

The Cormack method [6,7] connects so-called orthogonal spherical-harmonic functions in object space, with polar coordinates (r,θ), with their analytical Radon transform (also orthogonal) in projection space, with coordinates (p,ϕ). The angular functions are simple $\exp(\mathrm{i}m\theta)$ and $\exp(\mathrm{i}m\phi)$ functions.

The spherical-harmonic functions can be extended to elliptical coordinates in object space. Assume the elliptical coordinates to be

$$\begin{aligned} x &= x_0 + r\cos\theta, \\ y &= y_0 + \varepsilon r\sin\theta, \end{aligned} \tag{99}$$

where (x_0, y_0) is the chosen origin or the polar coordinates and the projection-space coordinates. To use the Cormack method, one can simply apply the linear transformation

$$\begin{aligned} x' &= x, \\ y' &= y_0 + \frac{y - y_0}{\varepsilon}, \end{aligned} \tag{100}$$

to obtain polar coordinates. Because this transformation is linear, a straight line (p,ϕ) will be transformed to another straight line (p',ϕ'):

$$\begin{aligned} \phi' &= \arctan(\varepsilon\tan\phi), \\ p' &= \frac{p\sin\phi'}{\varepsilon\sin\phi} = \frac{p\cos\phi'}{\cos\phi}. \end{aligned} \tag{101}$$

Note that the limits for $\phi,\phi' \to 0,\pi/2,\pi,3\pi/2$ exist. In principle, this is the only transformation that is required to apply the Cormack method to elliptical coordinates: the lines of sight are transformed, the reconstruction is done in polar coordinates, and the reconstruction in real coordinates is back-transformed. To depict the orthogonal functions in projection space for elliptical coordinates (see Fig.5), one can use the inverse transformation:

$$\begin{aligned} \phi &= \arctan\left(\frac{\tan\phi}{\varepsilon}\right), \\ p &= \frac{\varepsilon p'\sin\phi}{\sin\phi'} = \frac{p\cos\phi}{\cos\phi'}. \end{aligned} \tag{102}$$

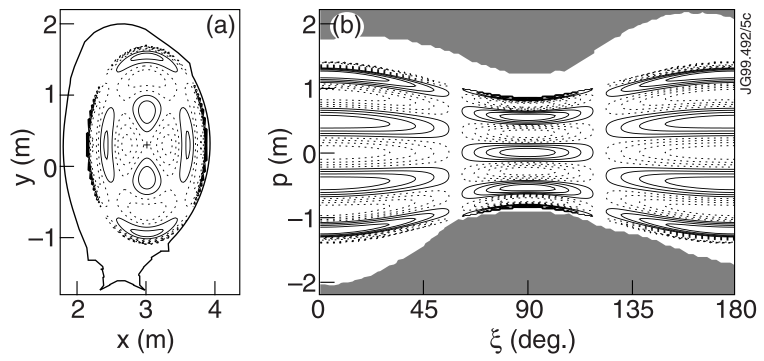

Fig.5: Illustration of a Cormack function in elliptical coordinates suitable for elongated tokamak plasmas, (a) in object space (the JET first wall outline is shown) and (b) in projection space (the grey area indicates lines of sight that pass outside the JET first wall). Note that the angle ξ is used in projection space (see Sec. 3.1). The cosine component of the Cormack function with parameters $n = 2$ and $m = 3$ [6] is shown. This figure was extracted from Figs.2(e) and (f) of Ref. 4.

The above expressions were derived by considering the transformation of two points on the line

$$\begin{aligned}(x_0, y_0 + \tfrac{p}{\sin\phi}) &\rightarrow (x_0, y_0 + \tfrac{p}{\varepsilon\sin\phi}),\\ (x_0 + \tfrac{p}{\cos\phi}, y_0) &\rightarrow (x_0 + \tfrac{p}{\cos\phi}, y_0),\end{aligned} \tag{103}$$

and applying Eqs. (93) and (94).

For (p,ξ) coordinates the above expressions are

$$\begin{aligned}\tan\xi' &= \frac{\tan\xi}{\varepsilon},\\ \frac{p'}{\sin\xi'} &= \frac{p}{\sin\xi}.\end{aligned} \tag{104}$$

3.3 Interpolative generalized natural basis function in object space

The interpolative generalized natural basis functions (NBF) are used for a tomographic reconstruction method described in Ref. 4. These basis functions are pyramid-shaped in projection space [see Fig.1(h) of Ref. 4]. Here it is derived how the corresponding basis functions in object space can be calculated: they are the backprojection of the pyramid functions in projection space.

The most efficient way to determine the interpolative generalized natural basis function in object space is to loop over all grid points in object space and to determine the values of all such basis functions that are nonzero for those points. Thus, for each given point (x,y), one solves intersections of the curve (90) with the grid in p and ξ as a function of ξ, and stores these intersections as ξ_n. The array ξ_n thus contains all grid coordinates ξ_j, and all intersections where

$$p_j = -(x - x_0)\sin\xi + (y - y_0)\cos\xi. \tag{105}$$

The additional solutions ξ_n from Eq. (105) are found by substituting $\tan(\xi/2)=t$, and thus $\sin\xi = 2t/(1+t^2)$ and $\cos\xi = (1-t^2)/(1+t^2)$. Note that the number of solutions depends on the quantity Δ for given p_j by $\Delta = (x-x_0)^2 + (y-y_0)^2 - p_j{}^2$: if $\Delta < 0$ there are no solutions, if $\Delta = 0$ there is a degenerate solution, and if $\Delta > 0$. For $\Delta \geq 0$ the solutions are given by

$$\begin{aligned} t &= \frac{-(x-x_0) \pm \sqrt{\Delta}}{p+(y-y_0)} && \text{and} \quad \xi = 2\arctan t && \text{for } (y-y_0) \neq -p, \\ t &= -\frac{p}{(x-x_0)} && \text{and} \quad \xi = \pi && \text{for } (y-y_0) = -p. \end{aligned} \tag{106}$$

Next, the solutions ξ_n are sorted and it is checked in which pixel, i.e. the region between four grid points, the middle point of each pair lies. The integral along curve (90) for each pixel is then evaluated over the bi-linear interpolative basis functions and the resulting values assigned to each of the four grid points. If the values of one particular natural basis function are needed, only the possible intersections ξ_n of curve (90) for each point (x,y) with the four pixels surrounding the grid point in question need to be considered: the present method however will efficiently give the values for all grid point with non-zero contribution (i.e. skipping all the grid points with zero contribution).

The calculation technique is very similar to the integrals in Sec. 2.4. If h_j, h_{j+1}, h_{j+J} and h_{j+J+1} are the values contained in the grid points at the lower left, upper left, lower right and upper right corners of the pixel, respectively, the bi-linearly interpolated value in any internal point $(p(\xi),\xi)$ on curve (90) is given by (see the definition (13) and (14) of the bi-linearly interpolative basis function in Sec. 2.3)

$$\begin{aligned} h(p(\xi),\xi) = &\left(1 - \frac{p(\xi)}{\Delta p} + \frac{p_j}{\Delta p}\right)\left(1 - \frac{\xi}{\Delta\xi} + \frac{\xi_j}{\Delta\xi}\right) h_j + \\ &\left(\frac{p(\xi)}{\Delta p} - \frac{p_j}{\Delta p}\right)\left(1 - \frac{\xi}{\Delta\xi} + \frac{\xi_j}{\Delta\xi}\right) h_{j+1} + \\ &\left(\frac{p(\xi)}{\Delta p} - \frac{p_j}{\Delta p}\right)\left(\frac{\xi}{\Delta\xi} - \frac{\xi_j}{\Delta\xi}\right) h_{j+J+1} + \\ &\left(1 - \frac{p(\xi)}{\Delta p} + \frac{p_j}{\Delta p}\right)\left(\frac{\xi}{\Delta\xi} - \frac{\xi_j}{\Delta\xi}\right) h_{j+J} \\ = &\, Ap(\xi) + B\xi + Cp(\xi)\xi + D. \end{aligned} \tag{107}$$

Carrying out the backprojection integral for the interval ξ_n to ξ_{n+1} (which corresponds to the pixel j) over the function $h(p(\xi),\xi)$ gives

$$\int_{\xi_n}^{\xi_{n+1}} h(p(\xi),\xi)\,\mathrm{d}\xi = AI_1 + BI_2 + CI_3 + DI_4, \tag{108}$$

where

$$I_1 = \int_{\xi_n}^{\xi_{n+1}} p(\xi)\,\mathrm{d}\xi = (x - x_0)(\cos\xi_{n+1} - \cos\xi_n) + (y - y_0)(\sin\xi_{n+1} - \sin\xi_n), \qquad (109a)$$

$$I_2 = \int_{\xi_n}^{\xi_{n+1}} \xi\,\mathrm{d}\xi = \tfrac{1}{2}(\xi_{n+1}^2 - \xi_n^2), \qquad (109b)$$

$$\begin{aligned} I_3 = \int_{\xi_n}^{\xi_{n+1}} p(\xi)\xi\,\mathrm{d}\xi = &-(x - x_0)(\sin\xi_{n+1} - \xi_{n+1}\cos\xi_{n+1} - \sin\xi_n + \xi_n\cos\xi_n) + \\ &(y - y_0)(\cos\xi_{n+1} - \xi_{n+1}\sin\xi_{n+1} - \cos\xi_n + \xi_n\sin\xi_n) \end{aligned}, \qquad (109c)$$

$$I_4 = \int_{\xi_n}^{\xi_{n+1}} \mathrm{d}\xi = \xi_{n+1} - \xi_n. \qquad (109d)$$

Furthermore,

$$A = a_j h_j + a_{j+1} h_{j+1} + a_{j+J+1} h_{j+J+1} + a_{j+J} h_{j+J}, \qquad (110)$$

and similarly *B*, *C* and *D*. The components are

$$a_j = -\frac{1}{\Delta p} + \frac{\xi_j}{\Delta p \Delta\xi}, a_{j+1} = \frac{1}{\Delta p} + \frac{\xi_j}{\Delta p \Delta\xi}, a_{j+J+1} = -\frac{\xi_j}{\Delta p \Delta\xi}, a_{j+J} = +\frac{\xi_j}{\Delta p \Delta\xi}, \qquad (111a)$$

$$b_j = -\frac{1}{\Delta\xi} - \frac{p_j}{\Delta p \Delta\xi}, b_{j+1} = \frac{p_j}{\Delta p \Delta\xi}, b_{j+J+1} = -\frac{p_j}{\Delta p \Delta\xi}, b_{j+J} = \frac{1}{\Delta\xi} + \frac{p_j}{\Delta p \Delta\xi}, \qquad (111b)$$

$$c_j = \frac{1}{\Delta p \Delta\xi}, c_{j+1} = -\frac{1}{\Delta p \Delta\xi}, c_{j+J+1} = \frac{1}{\Delta p \Delta\xi}, c_{j+J} = -\frac{1}{\Delta p \Delta\xi}, \qquad (111c)$$

$$d_j = 1 + \frac{\xi_j}{\Delta\xi} + \frac{p_j}{\Delta p} + \frac{p_j\xi_j}{\Delta p \Delta\xi}, d_{j+1} = -\frac{p_j}{\Delta p} - \frac{p_j\xi_j}{\Delta p \Delta\xi}, d_{j+J+1} = \frac{p_j\xi_j}{\Delta p \Delta\xi}, d_{j+J} = -\frac{\xi_j}{\Delta\xi} - \frac{p_j\xi_j}{\Delta p \Delta\xi}.. \qquad (111d)$$

By re-arranging the terms and factoring out the h_js, one finds that the backprojection integral contributes the following value to corner point j

$$a_j I_1 + b_j I_2 + c_j I_3 + d_j I_4, \qquad (112)$$

and similarly for the other grid points. The sum over the contributions to the four corner points of the pixel must be equal to $\xi_{n+1} - \xi_n$.

Figure 6 shows an example of the resulting interpolative generalized natural basis functions in object space for one particular grid point of projection space.

3.4 Details of filtered backprojection

The most popular tomographic reconstruction algorithm in medical tomography is the filtered-backprojection (FBP) or convolution-backprojection algorithm [8]. This algorithm is robust and easy to implement, and relatively fast. However, it cannot be used directly to measuring systems

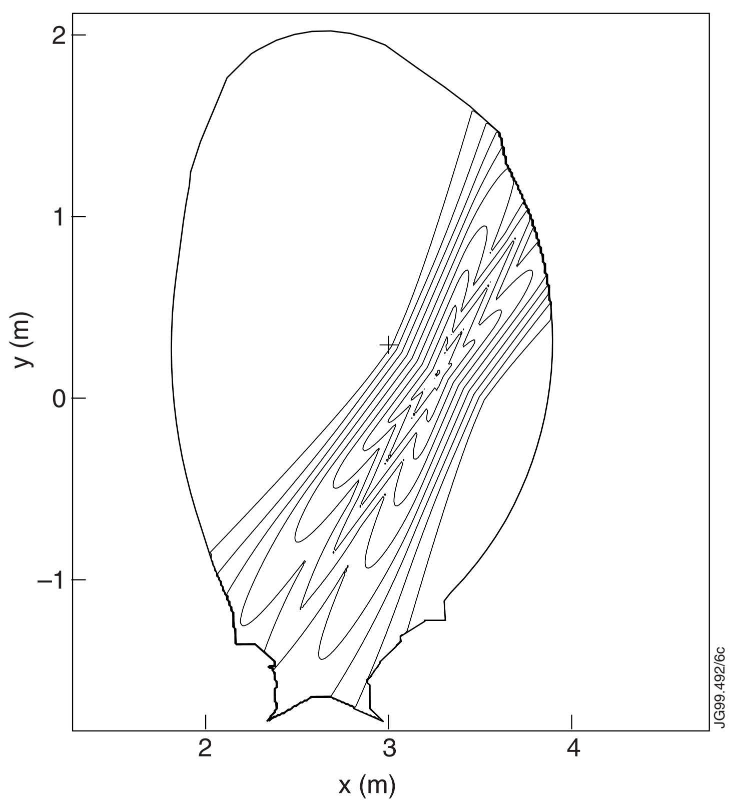

Fig.6: Example of an interpolative generalized natural basis function as contour plot in object space (the basis function is exaggerated in size). This figure was extracted from Fig. 1(g) of Ref. 4.

with irregular coverage. For this purpose, the iterative projection-space reconstruction method, which uses FBP in its iterations, was developed [5]. It is well known that [8,17] the tomographic inversion of the function $f(p,\xi)$ can be obtained by the following two steps: (1) the filtering by means of a convolution

$$h(p,\xi) = \int_{-\infty}^{\infty} f(p',\xi)\,\eta(p-p')\,\mathrm{d}p' \tag{113}$$

followed by (2) the backprojection

$$g(x,y) = \int_0^{\pi} h(-x\sin\xi + y\cos\xi,\xi)\,\mathrm{d}\xi. \tag{114}$$

The filtering function $\eta(p)$ is discussed in Sec. 3.4.1, the backprojection operator in 3.4.2, and the inverse of the filter operation in 3.4.3.

3.4.1 FBP filters

Commonly used filtering functions $\eta(p)$ can be found in the literature [8,17], where usually the function $H(P)$ in Fourier space is given, P being the spatial frequency corresponding to p. For small P, the Fourier transform of the filtering function must follow $|P|$, which is the function required in the analytical Radon inversion. Furthermore, in order to give a good regularization, it must (1) properly take into account the discrete nature of the measurements (at discrete p) and (2) suppress noise at high spatial frequencies. The first requirement is achieved by band-limiting

Table I Filter functions for the FBP method in Fourier domain [$|P|H(P)$] and p *domain [$\eta(p)$].*

Name	$\lvert P\rvert H(P)$	$\eta(p)$
Band-limiting	$\lvert P\rvert$	$\frac{\cos(2\pi P_0 p)+2\pi P_0 p\sin(2\pi P_0 p)-1}{2\pi^2 p^2}$ if $p=0$: P_0^2
Cosine	$\lvert P\rvert\cos\left(\frac{\pi P}{2P_0}\right)$	$4P_0^2\frac{t_1+t_2+t_3}{\pi^2(1-32P_0^2p^2+256P_0^4p^4)}$ with $t_1=(\pi-16\pi P_0^2p^2)\cos(2\pi P_0 p)$, $t_2=16P_0p\sin(2\pi P_0 p)$ and $t_3=-2-32P_0^2p^2$. if $p=\pm\frac{1}{4P_0}$: $P_0^2\left(\frac{1}{2}-\frac{2}{\pi^2}\right)$
Sinc or Shepp-Logan	$\frac{2P_0}{\pi P}\lvert P\rvert\sin\left(\frac{\pi P}{2P_0}\right)$	$8P_0^2\frac{4P_0p\sin(2\pi P_0p)-1}{\pi^2(16P_0^2p^2-1)}$ if $p=\pm\frac{1}{4P_0}$: $\frac{4P_0^2}{\pi^2}$
Generalized Hamming (with α as parameter, $0.5\le\alpha\le1$): Hanning: $\alpha=0.5$; Hamming: $\alpha=0.54$.	$\lvert P\rvert\left[\alpha+(1-\alpha)\cos\left(\frac{\pi P}{P_0}\right)\right]$	$\frac{1}{2\pi^2p^2(1-8P_0^2p^2+16P_0^4p^4)}\sum_{i=1}^{11}t_i$ with $t_1=\alpha\cos(2\pi P_0p)$, $t_2=64\alpha\pi P_0^5p^5\sin(2\pi P_0p)$, $t_3=-24\alpha\pi P_0^3p^3\sin(2\pi P_0p)$, $t_4=32\alpha P_0^4p^4\cos(2\pi P_0p)$, $t_5=-4\alpha P_0^2p^2\cos(2\pi P_0p)$, $t_6=-16P_0^4p^4\cos(2\pi P_0p)$, $t_7=2\alpha\pi P_0p\sin(2\pi P_0p)$, $t_8=-4P_0^2p^2\cos(2\pi P_0p)$, $t_9=8\pi P_0^3p^3\sin(2\pi P_0p)$, $t_{10}=-32\pi P_0^5p^5\sin(2\pi P_0p)$ and $t_{11}=-\alpha+(12\alpha-4)P_0^2p^2-16P_0^4p^4$. if $p=\pm\frac{1}{2P_0}$: $-\frac{P_0^2[\alpha(\pi^2+8)-\pi^2]}{2\pi^2}$ if $p=0$: $\frac{P_0^2[\alpha(\pi^2+4)-4]}{\pi^2}$

the function to half the Nyquist frequency P_0 of the measurements, i.e. $P_0=1/(2\Delta p)$, where Δp

is the distance between adjacent measurements. If the Fourier transform of the function $f(p,\xi)$ for fixed ξ does not have frequency components higher than P_0, no aliasing will arise. The second requirement is achieved by either choosing P_0 smaller than the Nyquist frequency or by a suppressing function $H(P)$ at high P. The amount of suppression is the essential differencebetween the various functions $\eta(p)$ described in the literature. The inverse Fourier transform $\eta(p)$ of the band-limited and suppressing function $|P|H(P)$ can be found analytically for several of the common functions by means of the inverse Fourier cosine transform (because $|P|H(P)$ and $\eta(p)$ are even functions). Some pairs are given in Table I and in Fig. 7.

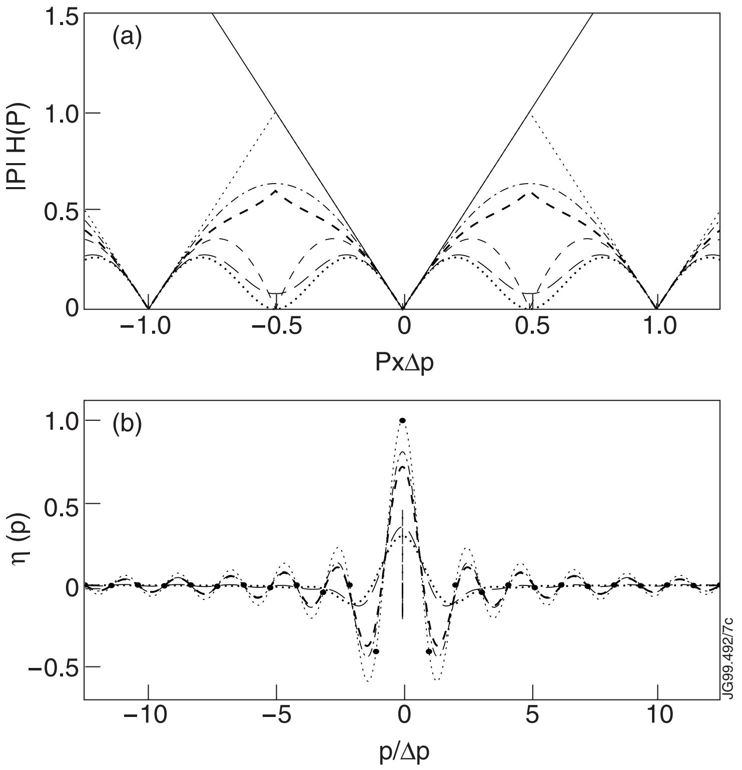

Fig.7: The various FBP filter functions of Table I in Fourier space (a) and direct space (b). Solid line: not band-limited; thin dotted line: band-limited; thin dashed line: cosine filter; dot-dashed line: Shepp-Logan (sinc) filter; thick dotted line: Hanning filter ($\alpha = 0.5$); long-dashed line: Hamming filter ($\alpha = 0.54$); thick dashed line: generalized Hamming filter ($\alpha = 0.8$). The solid circles in (b) show the points at which the filter functions are actually used in the numerical implementation if the band limit is chosen to coincide with two times the Nyquist frequency and no interpolation is applied between sample points.

In the numerical implementation it can be advantageous to interpolate $f(p,\xi)$ for fixed ξ to many more points p, while retaining P_0 of the original sampling in the function $\eta(p)$: otherwise discontinuities in higher derivatives of the interpolated function can give rise to noise in the $h(p,\xi)$. In practice there is not particularly much difference between the results obtained with the various filters, although pure band-limitation may be too rough. Of the filter functions given in Table I, the generalized Hamming window with small α gives most suppression of high frequencies. In practice, one can also choose a smaller cut-off frequency than the Nyquist frequency for a given Δp.

3.4.2 Backprojection

The numerical implementation of the backprojection operator Eq. (114) is usually a *numerical* integral over interpolated values of the function $h(p,\xi)$ between grid points in projection space, typically by nearest-neighbour or *linear* interpolation in p [8]. With the expressions of Sec. 3.3, a higher accuracy can be achieved efficiently by *analytical* expressions for *bi-linear* interpolation, i.e. interpolation in both p and ξ direction. For the backprojection in point (x,y) the ξ_n are solved as before from $\xi = \xi_j$ and Eq. (105). For each interval in ξ, the backprojection is given by Eq. (108), using the expressions (109)–(111).

3.4.3 Inverse filter operation

The inverse operation of Eq. (113) can be expressed as [4]:

$$f(p,\xi) = \int_0^{\pi}\int_{-\infty}^{\infty} h(p',\xi')\ A(p,\xi \mid p',\xi')\ \mathrm{d}p'\ \mathrm{d}\xi', \tag{115}$$

where the function $A(p,\xi \mid p',\xi')$ is given in terms of the geometric function K and natural basis functions B by

$$A(p,\xi \mid p',\xi') = \iint K(p,\xi \mid x,y) B(x,y \mid p',\xi')\, \mathrm{d}x\, \mathrm{d}y. \tag{116}$$

Normally, the functions K and B will be chosen to have finite widths and in that case describe the series-expansion tomography method based on natural basis functions [4]. For theoretical purposes, it is of interest to calculate $A(p,\xi \mid p',\xi')$ for line integrals. This is applied in Ref. 4. Thus, one chooses

$$B(x,y \mid p,\xi) = \delta(p + x\sin\xi - y\cos\xi) \tag{117}$$

and

$$K(p,\xi \mid x,y) = \delta(p + x\sin\xi - y\cos\xi)\, \Pi_{0,a^2}(x^2+y^2). \tag{118}$$

The band-pass function Π in Eq. (118) is defined by

$$\Pi_{z_\mathrm{c},r}(z) = \begin{cases} 1 & \text{for } z_\mathrm{c} - r < z < z_\mathrm{c} + r \\ 0 & \text{otherwise} \end{cases} \tag{119}$$

and can be introduced without affecting the outcome of the Radon transform as g has finite support (i.e. is zero outside a circle with radius a). Substitution of $B(x,y \mid p,\xi)$ and $K(p,\xi \mid x,y)$ into equation (116) gives

$$\begin{aligned} &A(p,\xi \mid p',\xi') \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(p + x\sin\xi - y\cos\xi)\delta(p' + x\sin\xi' - y\cos\xi')\, \Pi_{0,a^2}(x^2+y^2)\, \mathrm{d}x\, \mathrm{d}y. \end{aligned} \tag{120}$$

The following explicit expressions for $A(p,\xi \mid p',\xi')$ were used. By integrating over the delta functions in Eq. (120) one can show that

$$A(p,\xi \mid p',\xi') = \frac{1}{\left|\sin(\xi - \xi')\right|} \Pi_{p'\cos(\xi-\xi'),\sin(\xi-\xi')\sqrt{a^2-p'^2}}(p). \tag{121}$$

In the singularity at $\xi = \xi'$ one can show that

$$\int_{-\infty}^{\infty} h(p',\xi) A(p,\xi \mid p',\xi) \mathrm{d}p' = 2\sqrt{a^2 - p^2}\, h(p,\xi). \tag{122}$$

At first sight it may seem surprising that an equation with two integrals [Eq. (115)] is the inverse of an equation with one integral [equation (113)]. This inverse is however only valid for functions $f(p,\xi)$ that are Radon transforms of arbitrary functions $g(x,y)$; i.e. the functions $f(p,\xi)$ have to satisfy the consistency conditions [1] of projection space [4]. One can further note that the inversion formula of the Radon transform, a single integral, also has a double integration.

ACKNOWLEDGEMENTS

I am very much indebted to Christoph Fuchs and Gijs Fehmers, who have done much of the initial work on the constrained-optimization method used at JET. Christoph Fuchs initially derived alternative expressions for the line integrals with pyramid basis functions and initiated the smoothness matrix for flux coordinates; furthermore, he left a fully functioning tomography code at JET, which has been very useful. Section 2.7 is entirely based on Gijs Fehmers' work.

REFERENCES

[1] L.C. Ingesson, B. Alper, H. Chen, A.W. Edwards, G.C. Fehmers, J.C. Fuchs, R. Giannella, R.D. Gill, L. Lauro-Taroni, M. Romanelli, "Soft x-ray tomography during ELMs and impurity injection in JET," Nucl. Fusion **38**, 1675–1694 (1998)

[2] L.C. Ingesson, R. Reichle, G.C. Fehmers, H. Guo, L. Lauro-Taroni, A. Loarte, R. Simonini, "Radiation distribution and neutral-particle loss in the JET MkI and MkIIA divertors," in *Proc. 24th EPS Conference on Controlled Fusion and Plasma Physics (Berchtesgaden, 1997)*, Ed. M. Schittenhelm *et al.*, Europhysics Conference Abstracts Vol. 21A (EPS, 1997), Part I, pp. 113

[3] L.C. Ingesson, H. Chen, P. Helander and M.J. Mantsinen, "Comparison of basis functions in soft x-ray tomography and observation of poloidal asymmetries in impurity density," Plasma Phys. Control. Fusion **42**, 161-180 (2000); see also L.C. Ingesson, *Application of natural basis functions to soft x-ray tomography,* JET Report JET–R(99)07

[4] L.C. Ingesson, "A tomographic reconstruction method with generalized natural basis functions and *a priori* information," submitted to Inverse Problems

[5] L.C. Ingesson and V.V. Pickalov, "An iterative projection-space reconstruction algorithm for tomography systems with irregular coverage," J. Phys. D: Appl. Phys. **29**, 3009–3016 (1996)

[6] A.M. Cormack, “Representation of a function by its line integrals, with some radiological applications. II,” J. Appl. Phys. **35**, 2908 (1964)

[7] R.S. Granetz and P. Smeulders, “X-ray tomography on JET,” Nucl. Fusion **28**, 457 (1988)

[8] R.M. Lewitt, “Reconstruction algorithms: transform methods,” Proc. IEEE **71**, 390–408 (1983)

[9] L.C. Ingesson, P.J. Böcker, R. Reichle, M. Romanelli, and P. Smeulders, “Projection-space methods to take into account finite beam-width effects in two-dimensional tomography algorithms,” J. Opt. Soc. Am. A **16**, 17–27 (1999); see also Ref. 10 for more details

[10] L.C. Ingesson, P.J. Böcker, R. Reichle, M. Romanelli, and P. Smeulders, “Projection-space methods to take into account finite beam-width effects in two-dimensional tomography algorithms,” JET Report JET-R(98)02

[11] L.C. Ingesson, C.F. Maggi and R. Reichle, “Characterization of geometrical detection-system properties for two-dimensional tomography,” Rev. Sci. Instrum. **71**, 1370-1378 (2000)

[12] J.C. Fuchs, K.F. Mast, A. Hermann and K. Lackner., “Twodimensional reconstruction of the radiation power density in ASDEX Upgrade,” in *Proceedings of the 21st EPS Conference on Controlled Fusion and Plasma Physics (Montpellier, 1994)*, Ed. E. Joffrin *et al.*, Europhysics Conference Abstracts Vol 18B (EPS, 1994), Part III, pp. 1308–1311

[13] M. Bertero, C. de Mol and E.R. Pike, “Linear inverse problems with discrete data. II: Stability and regularisation,” Inverse Problems **4**, 573–594 (1988)

[14] G.C. Fehmers, L.P.J. Kamp, F.W. Sluijter, “An algorithm for quadratic optimization with one quadratic constraint and bounds on the variables,” Inverse Problems **14**, 893–901 (1998)

[15] G.C. Fehmers, *Tomography of the ionosphere*, PhD Thesis (Technische Universitieit Eindhoven, Eindhoven, The Netherlands, 1996)

[16] K.M. Hanson, G.W. Wecksung, “Local basis-function approach to computed tomography,” Appl. Opt. **24** (1985) 4028

[17] G.T. Herman, Image reconstruction from projections (Academic Press, New York, 1980), pp. 122

[18] S.R. Deans, *The Radon transform and some of its applications* (Wiley, New York, 1983), pp. 63

[19] I.S. Gradshteyn, I.M. Ryzhik, Edited by A. Jeffrey, *Table of integrals, series, and products, 5th edition* (Academic Press, Boston, 1994), p. 99

[20] S. Twomey, *Introduction to the mathematics of inversion in remote sensing and indirect measurement* (Elsevier, Amsterdam and New York, 1977)

[21] L.C. Ingesson and R. Reichle, *Lines of sight for ITER bolometers*, JET Report JET-R(98)03