M Wiesenberger et al.

Streamline integration as a method for two-dimensional elliptic grid generation.

Streamline integration as a method for two-dimensional elliptic grid generation

M. Wiesenberger, M. Held

Institute for Ion Physics and Applied Physics, Universität Innsbruck, A-6020 Innsbruck, Austria

L. Einkemmer

Numerical Analysis group, Universität Innsbruck, A-6020 Innsbruck, Austria

Abstract

We propose a new numerical algorithm to construct a structured numerical elliptic grid of a doubly connected domain. Our method is applicable to domains with boundaries defined by two contour lines of a two-dimensional function. The resulting grids are orthogonal to the boundary. Grid points as well as the elements of the Jacobian matrix can be computed efficiently and up to machine precision. In the simplest case we construct conformal grids, yet with the help of weight functions and monitor metrics we can control the distribution of cells across the domain. Our algorithm is parallelizable and easy to implement with elementary numerical methods. We assess the quality of grids by considering both the distribution of cell sizes and the accuracy of the solution to elliptic problems. Among the tested grids these key properties are best fulfilled by the grid constructed with the monitor metric approach.

Keywords: elliptic grid generation, discontinuous Galerkin, conformal grid generation, orthogonal grids

1. Introduction

A structured numerical grid is generated by the (numerical) coordinate transformation of a rectangular "computational domain" to the "physical do-

main" of interest [1, 2, 3]. Compared to unstructured grids a structured grid allows for an easier and computationally cheaper implementation of derivatives since the computational domain is a tensor product grid (a rectangle). Generally, the same numerical techniques as in a Cartesian coordinate system can be used. Still, a structured grid must fulfill certain qualities in order to be useful for practical numerical computations. Among others are smooth and nonoverlapping coordinate lines, boundary orthogonality and a homogeneous distributions of cells across the domain. The latter condition is important in advection schemes. Too small cells deteriorate the CFL condition in explicit schemes and the condition number of the discretization matrix in implicit schemes, while large cells deteriorate the accuracy. Boundary orthogonality is important in order to implement Neumann boundary conditions. As Reference [1] pointed out it is desirable for the grid to be at least near-orthogonal to the boundary. Although it is possible to represent Neumann boundary conditions in a curvilinear grid, the accuracy of the discretization deteriorates and the implementation is more involved than in a boundary-orthogonal grid.

In the literature on grid generation it has been established that elliptic grids are the ones that best fulfill these qualities [1, 2, 3]. These grids are generated by a coordinate transformation that fulfills an elliptic equation. However, the generation of these grids based on the algorithm proposed by Thompson, Thames and Mastin (TTM) is numerically involved [4, 3]. A special class of elliptic grids is generated by conformal mappings. The coordinates obey the Cauchy–Riemann differential equations

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \quad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x} \tag{1}$$

They are orthogonal, smooth, and preserve the aspect ratio of the cells from the computational into the physical space. There are many methods to numerically construct a conformal mapping [5], e.g. boundary integral element methods, however their implementation can be tricky. Among further improvements to the original TTM method are grid adaption methods and the monitor metric approach [6, 7, 3]. Both of these approaches modify the elliptic equation used to construct the coordinate system to a more suitable form. By choosing specific forms the distribution of grid cells is controlled, which makes the coordinate transformation more flexible than conformal mappings. The main numerical difficulty in these techniques is the solution of the nonlinear inverted Beltrami equation. This is required because a (not yet constructed) numerical grid is needed in order to discretize and solve the

elliptic equation directly. If the boundary lines are e.g. given by a parametric representation, this conundrum cannot be easily solved. In this contribution we show how this problem can be avoided if the boundary lines of the physical domain are given by a two-dimensional function.

One important application, where this is indeed the case, are tokamaks. These are magnetic fusion devices that use nearly axisymmetric magnetic fields to confine a plasma [8]. In a two-dimensional plane of fixed toroidal angle the magnetic field-lines, in the idealized case, lie within so-called flux-surfaces. These surfaces are in fact the isosurfaces of the poloidal magnetic flux $\psi(x, y)$ described by the Grad-Shafranov equation. A typical numerical modelling scenario involves the study of plasma dynamics between two flux-surfaces ψ_0 and ψ_1. Structured and unstructured numerical grids have been proposed for the numerical discretization of this region [9, 10, 11, 12, 13, 14, 15, 16]. Let us emphasize, however, that the algorithms developed are in no way restricted to magnetic fusion applications. In particular, the monitor metric approach gives sufficient flexibility to handle large classes of problems. For example, in certain pollution models a system of diffusion-advection-reaction equations have to be solved for a given velocity field (the velocity field is determined by analytic modeling, measurements, or computer simulation). In this case the stream function corresponding to the velocity field takes on the role of ψ [17]. A popular choice for tokamak magnetic fields are so-called flux or magnetic coordinates [18], in which the magnetic flux ψ or a function of it is the first coordinate. One particular class of such coordinate systems is known as PEST coordinates [12]. In this case, the geometric toroidal angle φ is the third coordinate, while a poloidal angle-like coordinate is constructed implicitly by choosing the volume form of the coordinate transformation. Other choices of flux coordinates exist, where the geometric toroidal angle φ differs from the toroidal flux angle φ_f. For example, Boozer [10, 11] and Hamada [9] coordinates. In general, a flux aligned coordinate system is not orthogonal however. Flux aligned grids are commonly constructed by integrating the streamlines of the vector field tangential to the isosurfaces of ψ [18, 19]. Reference [16] constructed a near-conformal coordinate system that is aligned to the magnetic flux-surfaces in this way. The coordinates are near-conformal in the sense that the grid-deformation is small. Unfortunately, the coordinate lines are not orthogonal.

The previous examples show that it is possible to construct a structured grid of a domain defined by the contour lines of a two-dimensional function

$\psi(x, y)$ (the flux). Now recall that this is exactly the requirement for the discretization of an elliptic equation on this domain. The inverted Beltrami equation in the TTM and related methods is no longer needed.

We therefore propose to construct an elliptic grid in three steps. First, a flux aligned grid is constructed by one of the methods that have been proposed in the literature. Then in a second step a suitably chosen elliptic equation is transformed to, discretized and solved in this coordinate system. We investigate the simple Laplace equation, the adapted equation and the elliptic equation with monitor metric suggested by References [3, 6, 7]. Finally, we treat the solution of the elliptic equation as a new flux function. We can therefore use an adapted version of the algorithm in the first step to construct a second coordinate transformation from the ψ-aligned coordinates to coordinates aligned to the solution of the second step. The final transformation then consists of the two consecutive coordinate transformations from the first and the third step.

In Section 2 we derive our algorithm by using methods from differential geometry. We first explain the approach of streamline integration by exemplarily constructing an orthogonal flux aligned grid. Then we show how to construct the conformal grid, the adapted grid and the grid using a monitor metric with our new method. In Section 3 we numerically show that with our algorithm the coordinate map and its derivatives can be computed efficiently and up to machine precision. Furthermore, we plot example grids for a domain relevant in magnetic confinement fusion, which are suitable for edge turbulence simulations. We compare the grids to flux aligned grids and assess the quality of our grids with two different solutions of an elliptic equation and the computation of maximal and minimal cell sizes. We conclude in Section 4.

2. High precision elliptic grid generation

Given is a function $\psi(x, y)$ in Cartesian coordinates. We want to construct a grid on the region bounded by the two lines $\psi(x, y) = \psi_0$ and $\psi(x, y) = \psi_1$ with $\psi_0 \neq \psi_1$. The derivative of the function ψ may not vanish within this region and on the boundary and we further assume that the region is topologically a ring. Note here that this excludes the description of domains with an X-point or O-point. The numerical grid is described by a mapping of the discretization of the rectangular computational domain $(u, v) \in [0, u_1] \times$

$[0, 2\pi]$ to the physical domain (x, y). u_1 is an unknown, which the grid generation process has to provide.
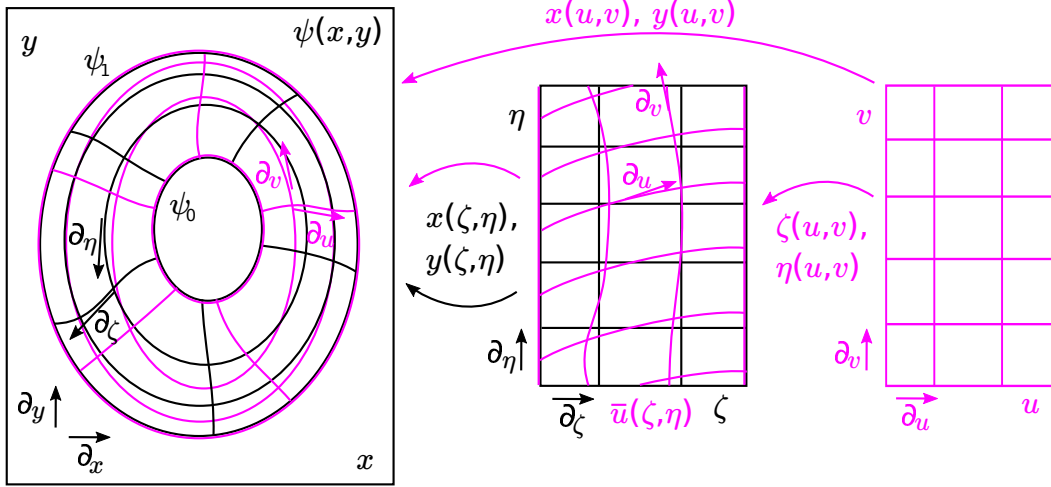

Figure 1: Sketch of the coordinate systems, coordinate lines and basis vector fields involved in our method.

As mentioned in the introduction and illustrated in Fig. 1 the idea of our algorithm are two consecutive coordinate transformations constructed by streamline integration. We denote the first transformation with $x(\zeta, \eta)$, $y(\zeta, \eta)$, where the ζ coordinate is aligned with ψ and η is an angle-like coordinate. The solution of the elliptic equation on this flux-aligned coordinate system is denoted with $\bar{u}(\zeta, \eta)$. The second coordinate transformation is then denoted $\zeta(u, v)$, $\eta(u, v)$ with u aligned to $\bar{u}$.

When deriving the algorithm we use basic methods and notational conventions of differential geometry. Here, we recommend the excellent Reference [20] as an introduction to the topic. We do so since in this approach the separate roles of the metric tensor, the coordinate system and its base vector fields are very clear. This is paramount for a concise description of our method.

In this Section we first show how the basis vector fields can be integrated to construct coordinate lines with the example of orthogonal coordinates in Section 2.2. This follows an introduction of some helpful quantities, our notation and streamlines in Section 2.1. In the next step we transform, discretize and solve an elliptic equation on the flux aligned coordinate system. The solution $\bar{u}(\zeta, \eta)$ then takes the role of a new flux function in the flux aligned

5

coordinates. We can therefore repeat the streamline integration method in the flux aligned coordinate system in order to construct coordinate lines of the final u, v coordinates. We discuss three examples of this method. In Section 2.3 we consider the simple Laplace and the Cauchy-Riemann equations (1) in order to construct conformal coordinates. In Section 2.4 we modifiy the elliptic equation to allow grid adaption, before we introduce the monitor metric in Section 2.5. Finally, we present our algorithm in Section 2.6.

2.1. Preliminaries

In a curvilinear coordinate system (ζ, η) the components of the metric tensor and its inverse take the form

$$\mathbf{g}(\zeta, \eta) = \begin{pmatrix} g_{\zeta\zeta} & g_{\zeta\eta} \\ g_{\zeta\eta} & g_{\eta\eta} \end{pmatrix} \quad \mathbf{g}^{-1}(\zeta, \eta) = \begin{pmatrix} g^{\zeta\zeta} & g^{\zeta\eta} \\ g^{\zeta\eta} & g^{\eta\eta} \end{pmatrix} \tag{2}$$

We denote $g = \det \mathbf{g} = g_{\zeta\zeta}g_{\eta\eta} - g_{\zeta\eta}^2$. For Cartesian coordinates (x, y) the elements of the inverse metric tensor are transformed by

$$g^{\zeta\zeta} = \zeta_x^2 + \zeta_y^2 \tag{3a}$$

$$g^{\zeta\eta} = \zeta_x\eta_x + \zeta_y\eta_y \tag{3b}$$

$$g^{\eta\eta} = \eta_x^2 + \eta_y^2 \tag{3c}$$

since $g^{xx} = g^{yy} = 1$ and $g^{xy} = 0$. Given $\zeta(x, y)$, $\eta(x, y)$ and its inverse $x(\zeta, \eta)$, $y(\zeta, \eta)$ recall that their Jacobian matrices are related by

$$\begin{pmatrix} x_\zeta & x_\eta \\ y_\zeta & y_\eta \end{pmatrix}\Bigg|_{\zeta(x,y),\eta(x,y)} = \frac{1}{\zeta_x\eta_y - \zeta_y\eta_x} \begin{pmatrix} \eta_y & -\zeta_y \\ -\eta_x & \zeta_x \end{pmatrix}\Bigg|_{x,y} \tag{4}$$

With the rules of tensor transformation it is easy to prove that the element of the volume form $\sqrt{g}$ is related to the Jacobian via

$$\sqrt{g} = (x_\zeta y_\eta - y_\zeta x_\eta) = (\eta_y\zeta_x - \eta_x\zeta_y)^{-1} \tag{5}$$

The components of the gradient operator and the divergence in an arbitrary coordinate system read

$$(\nabla f)^i = g^{ij}\partial_j f \tag{6a}$$

$$\nabla \cdot \mathbf{A} = \frac{1}{\sqrt{g}}\partial_i\left(\sqrt{g}A^i\right), \tag{6b}$$

6

where we sum over repeated indices $i, j \in \{\zeta, \eta\}$ and define $\partial_\zeta \equiv \partial/\partial\zeta$, $\partial_\eta \equiv \partial/\partial\eta$. Finally, we introduce the geometrical poloidal angle

$$
\theta(x, y) = \begin{cases} + \arccos\left(\frac{x - x_0}{\sqrt{(x-x_0)^2 + (y-y_0)^2}} \right) & \text{for } y \geq y_0 \\[3mm] - \arccos\left(\frac{x - x_0}{\sqrt{(x-x_0)^2 + (y-y_0)^2}} \right) & \text{for } y < y_0 \end{cases} \tag{7}
$$

such that the differential 1-form

$$
\mathrm{d}\theta = -\frac{y - y_0}{(x - x_0)^2 + (y - y_0)^2}\mathrm{d}x + \frac{x - x_0}{(x - x_0)^2 + (y - y_0)^2}\mathrm{d}y, \tag{8}
$$

where (x_0, y_0) is any point inside the region bounded by ψ_0.

Streamline integration is the central part of our algorithm. Recall that given a vector field $v(x, y) = v^x(x, y)\partial_x + v^y(x, y)\partial_y$ its streamlines are given by the equation

$$
\frac{\mathrm{d}x}{\mathrm{d}t} = v^x(x, y)|_{x(t), y(t)} \tag{9a}
$$

$$
\frac{\mathrm{d}y}{\mathrm{d}t} = v^y(x, y)|_{x(t), y(t)} \tag{9b}
$$

where t is a parameter. Recall here that in differential geometry the directional derivatives ∂_x and ∂_y are the base vector fields of the coordinate system[1]. Now recall that if we have a function $f(x, y)$ such that $f(x(t), y(t))$ is a one-to-one map from t to f we can re-parameterize Eq. (9) by

$$
\frac{\mathrm{d}x}{\mathrm{d}f} = \frac{\mathrm{d}x/\mathrm{d}t|_{t(f)}}{\mathrm{d}f/\mathrm{d}t|_{t(f)}} = \frac{v^x(x, y)}{(v^x\partial_x f + v^y\partial_y f)(x, y)}\bigg|_{x(f), y(f)} \tag{10a}
$$

$$
\frac{\mathrm{d}y}{\mathrm{d}f} = \frac{\mathrm{d}y/\mathrm{d}t|_{t(f)}}{\mathrm{d}f/\mathrm{d}t|_{t(f)}} = \frac{v^y(x, y)}{(v^x\partial_x f + v^y\partial_y f)(x, y)}\bigg|_{x(f), y(f)} \tag{10b}
$$

Furthermore, the derivative of any function $g(x, y)$ along the streamlines of v parameterized by f reads

$$
\frac{\mathrm{d}g}{\mathrm{d}f}\bigg|_{x(f), y(f)} = \frac{(v^x\partial_x g + v^y\partial_y g)(x, y)}{(v^x\partial_x f + v^y\partial_y f)(x, y)}\bigg|_{x(f), y(f)} \tag{11}
$$

---

[1]Some textbooks (e.g. [18]) introduce the notation $\mathbf{e}_i := \partial\mathbf{x}/\partial x^i$ and $\mathbf{e}^i := \nabla x^i$. While this formulation is suitable in many situations we refrain from using it since it mixes the metric into the basis vectors through the use of the gradient (cf. Eq. (6)), which is unpractical for our purposes.

7

Figuratively, in any coordinate system (ζ, η) the 1-forms $\mathrm{d}\zeta$ and $\mathrm{d}\eta$ (the contravariant basis) are visualized by the lines (surfaces in higher dimensions) of constant ζ and η. At the same time the streamlines of the vector fields ∂_ζ and ∂_η (the covariant basis) give the coordinate lines of ζ and η (cf. Fig. 1). For example, if we hold η constant and vary ζ, we go along a streamline of ∂_ζ. This implies that in two dimensions $\mathrm{d}\eta$ and ∂_ζ trace the same line. The vector fields $\nabla\zeta$ and $\nabla\eta$ are associated to $\mathrm{d}\zeta$ and $\mathrm{d}\eta$ through the metric tensor by Eq. (6) and are the vector fields that are everywhere perpendicular to the lines of constant ζ and η, respectively. It is important to realize that in general curvilinear coordinates the vector fields $\nabla\zeta$ and $\nabla\eta$ point in different directions than ∂_ζ and ∂_η. The central point in our algorithm is the realization that once we can express ∂_ζ and ∂_η in terms of ∂_x and ∂_y we can immediately construct the coordinate transformation by integrating streamlines of ∂_ζ and ∂_η using Eq. (9). This holds true even if (x, y) were curvilinear coordinates. The components of the one-forms $\mathrm{d}\zeta$ and $\mathrm{d}\eta$ in terms of $\mathrm{d}x$ and $\mathrm{d}y$ form the elements of the Jacobian matrix of the transformation. These are also necessary in order to transform any tensor (including the metric) from the old to the new coordinate system.

2.2. Orthogonal coordinates

In general, orthogonal coordinates ζ, η with ζ aligned to ψ are described by

$$\mathrm{d}\zeta = \zeta_x \mathrm{d}x + \zeta_y \mathrm{d}y = f(\psi)(\psi_x \mathrm{d}x + \psi_y \mathrm{d}y) \tag{12a}$$

$$\mathrm{d}\eta = \eta_x \mathrm{d}x + \eta_y \mathrm{d}y = h(x, y)(-\psi_y \mathrm{d}x + \psi_x \mathrm{d}y) \tag{12b}$$

With Eq. (12) we have $g^{\zeta\eta} = \zeta_x \eta_x + \zeta_y \eta_y = 0$, $g^{\zeta\zeta} = (\nabla\psi)^2 f^2$, $g^{\eta\eta} = (\nabla\psi)^2 h^2$ and $\sqrt{g}^{-1} = (\nabla\psi)^2 h f$. From Eq. (4) we directly see that the basis vector fields are

$$\partial_\zeta = x_\zeta \partial_x + y_\zeta \partial_y = \frac{1}{(\nabla\psi)^2 f}(\psi_x \partial_x + \psi_y \partial_y) \tag{13a}$$

$$\partial_\eta = x_\eta \partial_x + y_\eta \partial_y = \frac{1}{(\nabla\psi)^2 h}(-\psi_y \partial_x + \psi_x \partial_y) \tag{13b}$$

i.e. ∂_ζ points into the direction of the gradient of ψ and ∂_η into the direction of constant $\psi = $ const surfaces. Now, the coordinate system is defined up to the functions $f(\psi)$ and $h(x, y)$. Note that f must be a function of ψ only since the restriction $\mathrm{d}(\mathrm{d}\zeta) = 0$ must hold. Furthermore, $f(\psi) = \mathrm{d}\zeta/\mathrm{d}\psi \neq 0$

8

is in principle an arbitrary function, yet we choose $f(\psi) = f_0 = \text{const}$. With this choice we directly get

$$\zeta(x, y) = f_0(\psi(x, y) - \psi_0) \tag{14}$$

Note that $\zeta_0 = 0$ and $\zeta_1 = f_0(\psi_1 - \psi_0)$. The up to now undefined function $h(x, y)$ is not arbitrary since $d(d\eta) = 0$ must hold. This is the requirement that $d\eta$ must be a closed form in order for the potential η to exist. We can express this as

$$(\psi_x \partial_x + \psi_y \partial_y)h = f(\nabla \psi)^2 \partial_\zeta h = -h\Delta\psi \tag{15}$$

where $\Delta\psi = \psi_{xx} + \psi_{yy}$ is the two-dimensional Laplacian. Let us remark here that Eq. (15) can be written as $\nabla \cdot (h\nabla\psi) = 0$, which makes the orthogonal grid an elliptic grid with adaption function h as becomes evident later. In order to integrate this equation we need an initial condition for h. We choose to first discretize the line given by $\psi(x, y) = \psi_0$.

As already mentioned ∂_η is the vector field the streamlines of which give the coordinate lines for η. We choose $h(x, y) = \text{const}$ on ψ_0. To this end we parameterize the coordinate line by θ (cf. Eq. (10))

$$\left.\frac{\mathrm{d}x}{\mathrm{d}\theta}\right|_{\zeta=0} = \frac{x_\eta}{\theta_\eta} = \frac{-\psi_y}{\psi_x\theta_y - \psi_y\theta_x} \tag{16a}$$

$$\left.\frac{\mathrm{d}y}{\mathrm{d}\theta}\right|_{\zeta=0} = \frac{y_\eta}{\theta_\eta} = \frac{\psi_x}{\psi_x\theta_y - \psi_y\theta_x} \tag{16b}$$

$$\left.\frac{\mathrm{d}\eta}{\mathrm{d}\theta}\right|_{\zeta=0} = \frac{1}{\theta_\eta} = \frac{(\nabla\psi)^2 h(\psi_0)}{\psi_x\theta_y - \psi_y\theta_x} \tag{16c}$$

Let us define $h(\psi_0)$ such that $\eta \in [0, 2\pi]$, that is,

$$2\pi = \oint_{\psi=\psi_0} \mathrm{d}\eta = \oint_0^{2\pi} \left.\frac{\mathrm{d}\eta}{\mathrm{d}\theta}\right|_{\zeta=0} \mathrm{d}\theta$$

or

$$f_0 := h(\psi_0) = \frac{2\pi}{\int_0^{2\pi} \mathrm{d}\theta \frac{(\nabla\psi)^2}{\psi_x\theta_y - \psi_y\theta_x}} \tag{17}$$

Here, we also fixed the constant f_0 such that our coordinate system ζ, η fulfills the Cauchy–Riemann condition on the boundary line ψ_0. As initial point for

9

the integration of Eq. (16) we can use any point with $\psi(x, y) = \psi_0$. We then use $h(\psi_0)$ on the flux-surface ψ_0 as initial condition for the integration of Eq. (15).

We obtain coordinate lines by integrating the vector fields ∂_ζ and ∂_η given in (13). We start the construction by integrating ∂_η for $\psi = \psi_0$, i.e. $\zeta = 0$. This can be done since $h|_{\psi_0}$ is known. The obtained points serve as starting points for the integration of $\partial_\zeta = f_0^{-1}\partial_\psi$. In order to get h we simply integrate Eq. (15)

$$\frac{\mathrm{d}x}{\mathrm{d}\zeta}\bigg|_{\eta=\mathrm{const}} = \frac{\psi_x}{f_0(\nabla\psi)^2} \tag{18a}$$

$$\frac{\mathrm{d}y}{\mathrm{d}\zeta}\bigg|_{\eta=\mathrm{const}} = \frac{\psi_y}{f_0(\nabla\psi)^2} \tag{18b}$$

$$\frac{\mathrm{d}h}{\mathrm{d}\zeta}\bigg|_{\eta=\mathrm{const}} = -\frac{\Delta\psi}{f_0(\nabla\psi)^2}h \tag{18c}$$

Note that if $\Delta\psi = 0$, we directly get a conformal grid with this algorithm. This can be seen as then $h(\zeta, \eta) = f_0$. In Appendix A we briefly study the class of functions ψ that are solutions of the Grad–Shafranov equation and satisfy $\Delta\psi = 0$. This, however, is not true in general.

Let us further remark on the sign of f_0. It is our goal to construct a right handed coordinate system and to have $\zeta_1 > \zeta_0 = 0$. The curves of constant θ surround x_0, y_0 in a mathematically positive direction. That means that Eq. (17) implies that $f_0 > 0$ if $\nabla\psi$ points away from x_0, y_0. If this is not the case, we obtain $f_0 < 0$. On the other hand if ζ should increase from ψ_0 to ψ_1, we need $f_0 < 0$ for $\psi_1 < \psi_0$ and $f_0 > 0$ for $\psi_1 > \psi_0$. We thus simply take the absolute value of Eq. (17) and multiply by -1 if $\psi_1 < \psi_0$. For ease of notation we do so also in the following without further notice.

Let us finally summarize the grid generation in the following algorithm; we assume that the ζ space is discretized by a list of not necessarily equidistant values ζ_i with $i = 0, 1, \ldots N_\zeta - 1$ and η is discretized by a list of η_j with $j = 0, 1, \ldots N_\eta - 1$:

1. Find an arbitrary point (x, y) with $\psi(x, y) = \psi_0$ and a point x_0, y_0 within the region bound by $\psi(x, y) = \psi_0$ for the definition of θ in Eq. (7)

2. Integrate Eq. (16) with $h = 1$ over $\Theta = [0, 2\pi]$ and use Eq. (17) to compute $f \equiv f_0$ and $h(\psi_0)$. Use any convenient method for the integration of ordinary differential equations.

3. Integrate one streamline of Eq. (13b) with $h = f_0$ from $\eta = 0..\eta_j$ for all j. The result is a list of N_η coordinates x_j, y_j on the ψ_0 surface.
4. Using this list as starting values integrate Eq. (18) from $\zeta = 0 \ldots \zeta_i$ for all i. This gives the map $x(\zeta_i, \eta_j), y(\zeta_i, \eta_j)$ for all i and j.
5. Last, using the resulting list of coordinates and Eq. (12) evaluate the derivatives $\zeta_x(\zeta_i, \eta_j)$, $\zeta_y(\zeta_i, \eta_j)$, $\eta_x(\zeta_i, \eta_j)$, and $\eta_y(\zeta_i, \eta_j)$ for all i and j.

2.3. Conformal coordinates

A conformal mapping $u(x, y), v(x, y)$ has to satisfy the Cauchy–Riemann equations given in Eq. (1). A direct consequence is that u and v are harmonic functions

$$\Delta u = \Delta v = 0 \tag{19}$$

Here, $\Delta = \nabla^2$ is the two-dimensional Laplacian with the divergence and gradient operators defined in (6). First, we note that Eq. (19) holds in every coordinate system. Let us assume that we have constructed flux aligned coordinates (ζ, η). These can be, but not necessarily have to be, the orthogonal coordinates introduced in the last section. Now, in order to construct conformal coordinates u, v we first define

$$u(\zeta, \eta) := c_0(\bar{u}(\zeta, \eta) - \psi_0) \tag{20}$$

and thus

$$\Delta \bar{u}(\zeta, \eta) = 0 \tag{21}$$

where $\bar{u}(0, \eta) = \psi_0$ and $\bar{u}(\zeta_1, \eta) = \psi_1$ fulfills Dirichlet boundary conditions in ζ. In η we have periodic boundary conditions.

Note the analogy between Eq. (20) and Eq. (14). Now, $\bar{u}$ is equal to ψ at the boundaries and its Laplacian vanishes in between. In fact, $\bar{u}$ takes the role of ψ in the following coordinate transformation. We introduce c_0 as a normalization constant with the same role as f_0 in the orthogonal coordinate transformation. Having $\bar{u}(\zeta, \eta)$, our idea is to construct the basis one-forms $\mathrm{d}u$ and $\mathrm{d}v$ in terms of $\mathrm{d}\zeta$ and $\mathrm{d}\eta$ by transforming the Cauchy-Riemann equations to the ζ, η coordinate system. Analogues to the algorithm in Section 2.2 we can then construct the basis vector fields ∂_u and ∂_v, appropriately choose a normalization and then use streamline integration in the ζ, η coordinate system to construct the coordinates. From the basis one-forms $\mathrm{d}u$ and $\mathrm{d}v$ we get the elements of the Jacobian matrix of the transformation. Before we do this in detail however, let us first discuss some alternative elliptic equations to the simple Eq. (19).

2.4. Grid adaption

Although the conformal grid is advantageous for elliptic equations (due to the vanishing metric coefficients) the cell distribution is not very flexible; once the boundary is set the conformal map is unique. We therefore have little control over the distribution of cells. We can use grid adaption techniques to overcome this restriction. The idea is to modify the elliptic equations that u and v have to fulfill. That is, we choose

$$\nabla \cdot \left(\frac{\nabla u}{w} \right) = \nabla \cdot (w \nabla v) = 0 \tag{22}$$

where w is an appropriately chosen weight function. The cell size will be small in regions where w is large and spread out in regions where w is small. The Cauchy–Riemann equations (1) are changed accordingly to

$$v_x = -\frac{u_y}{w} \quad v_y = \frac{u_x}{w} \tag{23}$$

Let us remark here that it is straightforward to implement the weight function in the orthogonal grid generation. In Section 2.2 we simply replace the function h by h/w. Then we have

$$\partial_\zeta = \frac{1}{f_0 (\nabla \psi)^2} (\psi_x \partial_x + \psi_y \partial_y) \tag{24a}$$

$$\partial_\eta = \frac{w}{h (\nabla \psi)^2} (-\psi_y \partial_x + \psi_x \partial_y) \tag{24b}$$

and $\nabla \psi \cdot \nabla (h/w) = -h/w \Delta \psi$. A suitable choice for w is

$$w = |\nabla \psi| \tag{25}$$

as then the angle-like coordinate η becomes the arc length on the ψ_0 line.

2.5. Monitor metric and the heat conduction tensor

We follow Reference [3, 6, 7] and replace the canonical metric tensor $\mathbf{g}$ by a specifically tailored tensor $\mathbf{G}$ that takes the form

$$\mathbf{G}(x, y) = \mathbf{TT} + k^2 \mathbf{NN} + \varepsilon(x, y) \mathbf{I} \tag{26}$$

with $\mathbf{T} = (-\psi_y, \psi_x)$ and $\mathbf{N} = -(\psi_x, \psi_y)$. The vector $\mathbf{T}$ is tangential to the contour lines of ψ while $\mathbf{N}$ is normal to it. We have

$$\sqrt{G} = \left[(\varepsilon + k^2 (\psi_x^2 + \psi_y^2))(\varepsilon + (\psi_x^2 + \psi_y^2)) \right]^{-1/2} \tag{27}$$

12

where $k < 1$ is a constant and ε is a function that is nonzero in the neighborhood of singularities, i.e. where $\nabla \psi(x, y) = 0$. In our work we choose $k = 0.1$ and $\varepsilon(x, y) \equiv \varepsilon = 0.001$. Note that the scalar product induced by $\mathbf{G}$ conserves perpendicularity with respect to $\mathbf{T}$, i.e. if and only if any vector $\mathbf{v} \perp \mathbf{T}$ in the canonical metric, then it is also perpendicular to $\mathbf{T}$ in $\mathbf{G}$. In our application this is important at the boundary.

Now, we consider the elliptic equation

$$\nabla \cdot (\sqrt{G}\mathbf{G}\nabla u) = 0$$
$$\partial_x(\sqrt{G}(G^{xx}\partial_x u + G^{xy}\partial_y u)) + \partial_y(\sqrt{G}(G^{yx}\partial_x u + G^{yy}\partial_y u)) = 0 \qquad (28)$$

with Dirichlet boundary conditions. The resulting grid coordinate u is almost perfectly aligned in regions far away from singularities and breaks the alignment in regions where $|\nabla\psi|$ is small or vanishes.

We now take a slightly more general approach and rewrite Eq. (28)

$$\nabla \cdot (\chi \nabla u) = 0 \qquad (29)$$

where $\chi(x, y)$ is a symmetric positive-definite contravariant tensor. Then the conformal grid from Section 2.3, the grid adaption from Section 2.4 as well as the monitor metric can be considered special cases. The grid adaption is recovered by setting $\chi = 1/w\mathbf{I}$, while the monitor metric is simply $\chi = \sqrt{G}\mathbf{G}$. The true conformal case is, of course, recovered by setting $\chi = \mathbf{I}$.

This allows us to provide a commonly known physical interpretation of Eq. (29). If u is a temperature, then the Dirichlet boundary condition fixes a temperature at the boundary of our domain. The tensor $\chi(x, y)$ is then the anisotropic heat conduction tensor and the coordinate lines for u are the isothermal lines of the steady state solution to the heat diffusion problem. This interpretation allows us to intuitively estimate how the coordinate lines look like when a specific χ is chosen. In the case of grid adaption, if the weight function is large, the heat conduction is low resulting in small temperature gradients and thus closely spaced grid cells. On the other hand, let us reconsider Eq. (26) for the case $\varepsilon = 0$. Then we can write

$$\chi = \frac{1}{k}\hat{t}\hat{t} + k\hat{n}\hat{n} = \chi_{\parallel}\hat{t}\hat{t} + \chi_{\perp}\hat{n}\hat{n} \qquad (30)$$

which for $k < 1$ simply means that the heat conduction parallel to the magnetic field is far stronger than perpendicular to it, which is in fact the case

in an actual fusion reactor. The coordinate lines will thus tend to align with the magnetic flux surfaces with the degree of alignment given by k resulting in an almost aligned grid. In the limit of vanishing k the alignment should be perfect. If $|\nabla\psi|$ vanishes, the tensor (26) reduces to $\chi = \mathbf{I}$.

2.6. The elliptic grids

Suppose that we have constructed a boundary aligned grid (ζ, η), which may but not necessarily has to be the orthogonal grid from Section 2.2. Now, we solve the general elliptic equation

$$\nabla \cdot (\chi \nabla \bar{u}) = \partial_i(\sqrt{g}\chi^{ij}\partial_j\bar{u}) = 0 \tag{31}$$

in the transformed coordinate system with boundary conditions $\bar{u}|_{\partial\Omega} = \psi$. We set $u = c_0(\bar{u} - \psi_0)$. This means that we have to transform the conduction tensor χ from Cartesian to flux coordinates, which is done by the well known rules of tensor transformation

$$\chi^{\zeta\zeta}(\zeta, \eta) = (\zeta_x\zeta_x\chi^{xx} + 2\zeta_x\zeta_y\chi^{xy} + \zeta_y\zeta_y\chi^{yy})|_{x(\zeta,\eta),y(\zeta,\eta)} \tag{32a}$$

$$\chi^{\zeta\eta}(\zeta, \eta) = (\zeta_x\eta_x\chi^{xx} + (\zeta_x\eta_y + \eta_x\zeta_y)\chi^{xy} + \zeta_y\eta_y\chi^{yy})|_{x(\zeta,\eta),y(\zeta,\eta)} \tag{32b}$$

$$\chi^{\eta\eta}(\zeta, \eta) = (\eta_x\eta_x\chi^{xx} + 2\eta_x\eta_y\chi^{xy} + \eta_y\eta_y\chi^{yy}))|_{x(\zeta,\eta),y(\zeta,\eta)} \tag{32c}$$

The equivalent of the Cauchy–Riemann equations in this formulation reads

$$v_\zeta = -\sqrt{g}(\chi^{\eta\zeta}u_\zeta + \chi^{\eta\eta}u_\eta) \tag{33a}$$

$$v_\eta = +\sqrt{g}(\chi^{\zeta\zeta}u_\zeta + \chi^{\zeta\eta}u_\eta) \tag{33b}$$

These are constructed such that $\nabla \cdot ((\chi/\det\chi)\nabla v) = 0$. The interested reader might notice that Eq. (33) just defines the components of the Hodge dual $dv = \star du$ if χ is interpreted as a metric. We note that these equations are now valid for any grid that we use to solve Eq. (31). If we find a boundary aligned grid analytically, we can start the grid construction directly with the solution of Eq. (31) and then proceed with the conformal grid generation.

The relevant equations for the streamline integration now read

$$du = c_0(\bar{u}_\zeta d\zeta + \bar{u}_\eta d\eta) \tag{34a}$$

$$dv = c_0\sqrt{g}(-(\chi^{\eta\zeta}\bar{u}_\zeta + \chi^{\eta\eta}\bar{u}_\eta)d\zeta + (\chi^{\zeta\zeta}\bar{u}_\zeta + \chi^{\zeta\eta}\bar{u}_\eta)d\eta) \tag{34b}$$

which just means that v is orthogonal to u in the scalar product generated by the symmetric tensor χ, which we denote by $\langle .,. \rangle$. We have $J = c_0^2 \sqrt{g} \langle \nabla \bar{u}, \nabla \bar{u} \rangle := c_0^2 \sqrt{g} (\bar{u}_\zeta^2 \chi^{\zeta\zeta} + 2\bar{u}_\zeta \bar{u}_\eta \chi^{\zeta\eta} + \bar{u}_\eta^2 \chi^{\eta\eta})$ and

$$\partial_u = \frac{1}{c_0 \langle \nabla \bar{u}, \nabla \bar{u} \rangle} (\chi^{\zeta\zeta} \bar{u}_\zeta + \chi^{\zeta\eta} \bar{u}_\eta)\partial_\zeta + (\chi^{\eta\zeta} \bar{u}_\zeta + \chi^{\eta\eta} \bar{u}_\eta)\partial_\eta \tag{35a}$$

$$\partial_v = \frac{1}{c_0 \sqrt{g} \langle \nabla \bar{u}, \nabla \bar{u} \rangle} (-\bar{u}_\eta \partial_\zeta + \bar{u}_\zeta \partial_\eta) \tag{35b}$$

As for the orthogonal coordinates we have to integrate these two vector fields to construct our coordinates. We begin with the integration of ∂_v along the $\zeta = 0$ line. It is important to note that $\bar{u}_\eta|_{\zeta=0} = 0$ and thus

$$\eta_v(0,\eta) = \left(c_0 \sqrt{g} \bar{u}_\zeta \chi^{\zeta\zeta}\right)|_{\zeta=0,\eta} \tag{36}$$

We can use Eq. (36) to define c_0 such that $v \in [0, 2\pi]$. In order to do so we simply integrate

$$v_1 = \int_0^{2\pi} \frac{dv}{d\eta}\bigg|_{\zeta=0} d\eta = \int_0^{2\pi} \frac{1}{\eta_v} d\eta = c_0 \int_0^{2\pi} \sqrt{g} \chi^{\zeta\zeta} \bar{u}_\zeta(0,\eta) d\eta := 2\pi \tag{37}$$

with $v_0 = 0$ and choose c_0 such that $v_1 = 2\pi$. This is the analogues equation to Eq. (17), with the difference that we can integrate Eq. (37) directly using numerical quadrature. Having done this we integrate, analog to Section 2.2, ∂_v on $\zeta = 0$ to get starting points for the integration of ∂_u from $\bar{u}_0 = 0$ to $\bar{u}_1 = c_0 \zeta_1$. Note, that we can compute the components of du and dv in terms of dx and dy by using the transformation

$$u_x(\zeta,\eta) = u_\zeta \zeta_x + u_\eta \eta_x, \ \ u_y(\zeta,\eta) = u_\zeta \zeta_y + u_\eta \eta_y \tag{38a}$$

$$v_x(\zeta,\eta) = v_\zeta \zeta_x + v_\eta \eta_x, \ \ v_y(\zeta,\eta) = v_\zeta \zeta_y + v_\eta \eta_y \tag{38b}$$

as soon as the constant c_0 becomes available.

Note that the resulting grid will, in general, not be orthogonal in the Euclidean metric. However, for all the cases we discuss in this paper, the grid is orthogonal at the boundary.

The final algorithm now reads, assuming that u is discretized by not necessarily equidistant points u_i for $i = 0, 1, \ldots N_u - 1$ and v by v_j with $j = 0, 1, \ldots N_v - 1$ and that $x(\zeta,\eta)$, $y(\zeta,\eta)$ as well as the components of the Jacobian $\zeta_x(\zeta,\eta)$, $\zeta_y(\zeta,\eta)$, $\eta_x(\zeta,\eta)$ and $\eta_y(\zeta,\eta)$ are available from the first coordinate transformation (e.g. Section 2.2):

15

1. Choose either $\chi = \mathbf{I}$, $\chi = 1/w\mathbf{I}$ or $\chi = \sqrt{G}\mathbf{G}$ depening on whether a conformal, adapted or monitor grid is desired.
2. Discretize and solve the elliptic equation (31) on the ζ, η grid for $\bar{u}(\zeta, \eta)$ with any method that converges. Use Eq. (32) to transform χ from Cartesian to the ζ, η coordinate system. The chosen resolution determines the accuracy of the subsequent streamline integration.
3. Numerically compute the derivatives $\bar{u}_\zeta$ and $\bar{u}_\eta$ and construct $\eta_v^{-1}(0, \eta)$ using Eq. (36) for $c_0 = 1$
4. Integrate Eq. (37) to determine c_0.
5. On the ζ, η grid compute η_v, ζ_u and ζ_v according to Eq. (35) as well as u_ζ, u_η, v_ζ and v_η according to Eq. (34). Use Eq. (38) and the Jacobian of the ζ, η coordinates to compute u_x, u_y, v_x and v_y.
6. Integrate the streamline of ∂_v for $\zeta = 0$ from $v = 0 \ldots v_j$ for all j on the ζ, η grid using the normalized component η_v. Interpolate $\eta_v(\zeta, \eta)$ when necessary.
7. Using the resulting points as start values integrate ∂_u from $u = 0 \ldots u_i$ for all i and all points. The result is the list of coordinates $\zeta(u_i, v_j)$, $\eta(u_i, v_j)$ for all i and j.
8. Interpolate $x(\zeta, \eta)$, $y(\zeta, \eta)$, $u_x(\zeta, \eta)$, $u_y(\zeta, \eta)$, $v_x(\zeta, \eta)$ and $v_y(\zeta, \eta)$ on this list.

There are two differences in this algorithm from the one presented in Section 2.2: For the integration of the vector fields Eq. (35) and the evaluation of derivatives Eq. (34) we need to evaluate its components at arbitrary points. An interpolation method is thus needed. In order to avoid out-of-bound errors we can artificially make the ζ, η box periodic. Second, the existence of the coordinate v is guaranteed by the Cauchy–Riemann equations and thus the h function does not appear.

A suitable test for the implementation is e.g. the volume/area of the domain. Being an invariant the volume must be the same regardless of the coordinate system in use.

3. Numerical tests

We extend our numerical library FELTOR (`www.github.com/feltor-dev/feltor`) [21] with a geometry package that can handle the various grids discussed in Section 2. For our implementation we choose high order explicit

Runge–Kutta methods (see, for example, [22]) to integrate the necessary ordinary differential equations and discontinuous Galerkin (dG) methods [23, 24] for the elliptic equations and the interpolation. The grid points and metric elements are thus available up to machine precision. Note here that each point of the grids we discussed in Section 2 can be computed independently from each other, which provides a trivial parallelization option.

We perform the computationally intensive inversion of the elliptic equation (31) on a GPU with a conjugate gradient method. On a single CPU a direct solver might be preferable but the assembly of the elliptic operator is not straightforward due to the presence of the metric elements. The advantage in an iterative method is that we do not need to assemble the whole matrix we only need to implement the application to a vector.

As already discussed in the introduction in all structured grids derivatives are pulled back to a rectangular grid (the computational space). The computational space is a product space, i.e. the discretization of the derivatives is one-dimensional (cf. [25, 24]). This reduces the stencil of the discretization compared to a discretization using an unstructured two-dimensional grid and simplifies the communication pattern in a parallel implementation. Furthermore, the matrices have a block-diagonal form with two side bands, where each block has the size $P \times P$ with P the number of polynomial coefficients in each cell. Apart from the corner entries all blocks on the diagonals are equal. This further reduces the storage requirements and in our experience increases the performance of matrix-vector multiplications by a factor of $2-3$ compared to a sparse matrix format (like the compressed sparse row format) that stores all non-zero elements explicitly.

In the following we will use the function ψ that is given by Reference [26] as an analytical solution to the Grad-Shafranov equation. This function is used in practice for turbulence simulations in tokamaks. The values of the 12 coefficients in ψ can be found in Appendix B. Furthermore, all programs used for this work can be found in the latest release of FELTOR [21].

3.1. Convergence of $\bar{u}$

We first check that the numerical solution of Eq. (31) converges as expected. As we do not have an analytical solution we compute the difference in the $L2$ norm of the computational space between two consecutive solutions

$\bar{u}^1_{\text{num}}$ and $\bar{u}^2_{\text{num}}$. We thus define the numerical error

$$\varepsilon_{\bar{u}} = \left(\frac{\int_{\zeta_0}^{\zeta_1} d\zeta \int_0^{2\pi} d\eta (\bar{u}^1_{\text{num}} - \bar{u}^2_{\text{num}})^2}{\int_{\zeta_0}^{\zeta_1} d\zeta \int_0^{2\pi} d\eta (\bar{u}^1_{\text{num}})^2} \right)^{1/2}. \tag{39}$$

In Table 1 we show the results for different high order polynomials to test the

$N_\eta = 10 N_\zeta$	$P = 9$	$P = 11$	$P = 13$	$P = 15$
20	-	-	-	-
40	5.98E-05	8.69E-06	5.86E-06	2.87E-07
80	3.48E-06	3.40E-07	3.75E-08	1.89E-08
160	4.80E-08	3.58E-09	2.89E-10	2.64E-11
320	7.43E-10	3.02E-11	1.71E-12	3.07E-12

Table 1: Comparison of the $L2$ errors (39) for the solution of the elliptic equation with monitor metric (31) on flux grid. The number of cells is denoted by N_η and N_ζ, respectively, and we employ polynomials of degree $P-1$ in each cell.

convergence of the general elliptic equation with monitor metric (31). The region is bounded by $\psi_0 = -20$ and $\psi_1 = -1$. We use a fluxgrid with the angle-like coordinate defined as the arc length to discretize this region [12]. The advantage is that this grid is quickly generated and has, other than the orthogonal grids discussed in Section 2.2, a fairly homogeneous distribution of cells throughout the domain. As we use Dirichlet boundary conditions the nonorthogonality is not an issue. We observe a quick convergence until a relative error of around 10^{-12} is reached. This apparent upper bound is due to the accuracy of the residuum in the conjugate gradient solver at 10^{-11}. Let us note that we observe similar convergence rates for the conformal and adapted grids.

3.2. Grid quality

Now we construct our elliptic grids and compare them to the near-conformal grid suggested by Reference [16] as a reference grid for existing flux aligned grids. Note that in this section by orthogonal grid we always mean the adapted orthogonal grid discussed in Section 2.4. In Fig. 2 we plot the orthogonal grid described in Section 2 with the function h chosen once on the

(a) orthogonal $\psi_0 = -20$ (b) orthogonal $\psi_0 = -1$ (c) near conformal
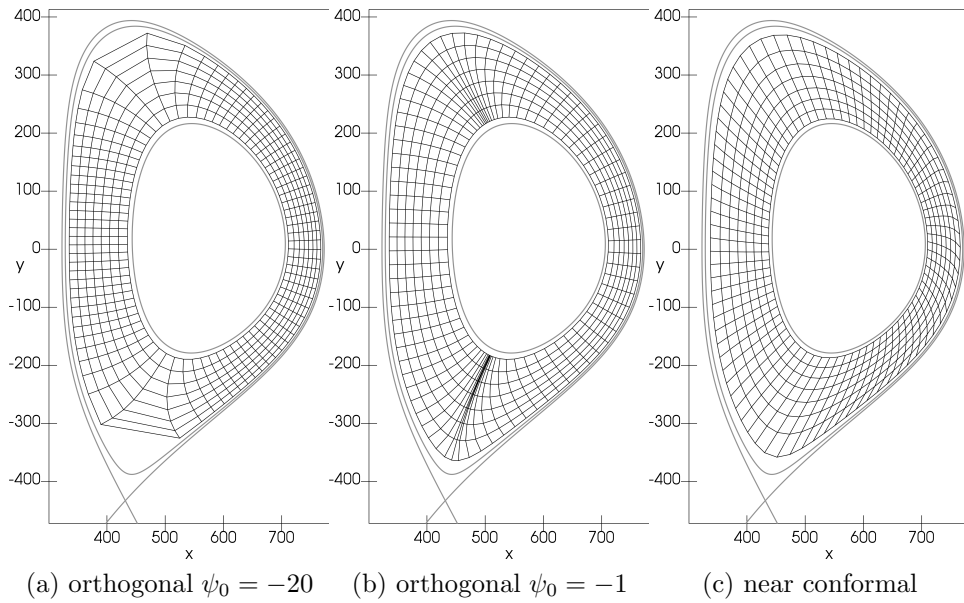
Figure 2: Orthogonal grid with $\psi_0 = -20$, $\psi_1 = -1$ (a), orthogonal grid with $\psi_0 = -1$, $\psi_1 = -20$ (b) both with $w = |\nabla\psi|$ and near conformal grid with $\psi_0 = -20$, $\psi_1 = -1$ (c) with equal number of grid points $P = 1$, $N_\zeta = 8$, $N_\eta = 80$. Note that the nodes represent cell centers and not the actual cell boundaries. The grey lines denote the actual grid boundaries at $\psi = -20$ and $\psi = -1$. For the sake of orientation we also plot the line $\psi = 0$ to indicate the X-point.

inner boundary 2a and once on the outer boundary 2b (in both cases adaption is used), and the near conformal grid proposed by Reference [16] in 2c. In the orthogonal grids, we have an equidistant discretization at the inner respectively outer boundary. Each of these grids is aligned with the magnetic flux function ψ. The orthogonal grids 2a and 2b distribute equally spaced cells evenly in the region around $Z = 0$, where the contour lines are almost straight. The cell size in the radial direction decreases as we go from the inner to the outer boundary. However, when the curvature of the contour lines increases (this is the case in the region around $R = 500$), the cells are either prolongated or compressed in the direction of the angle-like coordinate η. This effect results in cells with very large aspect ratio of the cells in the orthogonal grids.

The near-conformal grid 2c is near-conformal in the sense that the conformal deformation $g^{\zeta\zeta}/g^{\eta\eta}$ is small. Note, however, that this grid is completely aligned to the function ψ. This means that the aspect ratio of the cells remains as constant as possible. The downside of this grid is that it is clearly non-orthogonal especially close to the boundary and thus Neumann boundary conditions are difficult to implement.

In Fig. 3 we plot the results from our elliptic grid generation processes. We show a true conformal grid 3a, an elliptic grid with adaption 3b and one constructed with a monitor metric 3c. The last grid is clearly non-orthogonal while the conformal and adapted grids are orthogonal in the whole domain. All three grids are orthogonal at the boundary (which facilitates the implementation of Neumann boundary conditions). The conformal grid 3a suffers from large cells in the upper and lower regions of the domain. This is cured by either performing grid adaption 3b or by employing a monitor metric 3c. Both result in cells of similar size. The difference between these grids is the degree of flux alignment, which is clearly superior in the monitor grid in the region close to the X-point (Reference [6] provides a more quantitative investigation of the degree of alignment).

We assess the quality of our grids using an analytic solution of the following elliptic equation

$$\nabla \cdot (\chi \nabla f) = \rho, \tag{40}$$

where we either choose a flux aligned solution

$$f_{\text{ana}}(x, y) = 0.1(\psi - \psi_0)(\psi - 2\psi_1 + \psi_0) \tag{41a}$$

$$\chi(x, y) = \frac{x_0}{x}\sqrt{1 + (\nabla\psi)^2} \tag{41b}$$

20

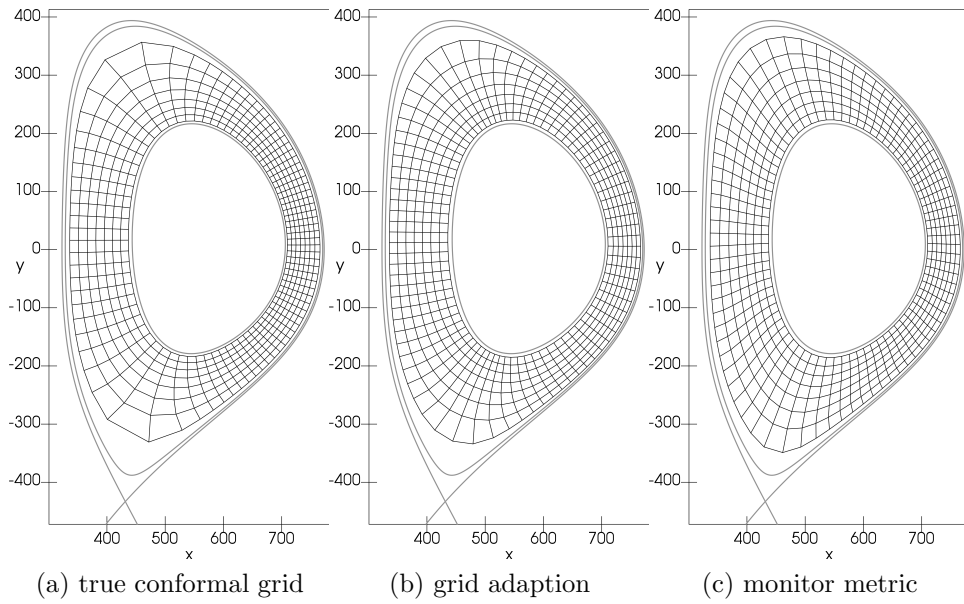(a) true conformal grid   (b) grid adaption   (c) monitor metric

Figure 3: Truly conformal grid (a), adapted grid with $w = |\nabla\psi|$ (b) and monitor grid with $k = 0.1$ and $\varepsilon = 0.001$ (c) with equal number of grid points $P = 1$, $N_\zeta = 8$, $N_\eta = 80$. We choose $\psi_0 = -20$ and $\psi_1 = -1$. Note that the nodes represent cell centers and not the actual cell boundaries. The grey lines denote the actual grid boundaries at $\psi = -20$ and $\psi = -1$. For the sake of orientation we also plot the line $\psi = 0$ to indicate the X-point.

or a localized solution

$$f_{\text{ana}}(x,y) = \begin{cases} 0 \text{ if } \left(\frac{x-x_b}{\sigma}\right)^2 + \left(\frac{y-y_b}{\sigma}\right)^2 > 1 \\ \exp\left[1 + \left(\left(\frac{x-x_b}{\sigma}\right)^2 + \left(\frac{y-y_b}{\sigma}\right)^2 - 1\right)^{-1}\right] \text{ else} \end{cases} \quad (42\text{a})$$

$$\chi(x,y) = \frac{x_0}{x}\sqrt{1 + (\nabla\psi)^2}(1 + 0.5\sin(\theta(x,y))). \quad (42\text{b})$$

The corresponding parameters are as follows: $x_0 = 550$, $\psi_0 = -20$ and $\psi_1 = -1$, $x_b = 440$, $y_b = -220$ and $\sigma = 40$. The position of the blob corresponds to the lower left corner of the domain. In this region the grids differ the most. Also note that the flux aligned solution (41) has a Neumann boundary condition at $\psi = \psi_1$ and a Dirichlet boundary at $\psi = \psi_0$. For the localized solution (42) we choose Dirichlet conditions at both boundaries. In order to numerically solve Eq. (40) we first have to transform the metric and the involved functions to the coordinate system that we use (cf. Reference [25, 24] on how the derivatives are discretized). It might seem that the conformal and orthogonal grids have a performance advantage over the curvilinear grids because the non-diagonal elements of the metric vanish. However, the number of necessary matrix-vector multiplications is four in all cases and only the number of vector-vector multiplications is different. Consequently, we only observe minor differences in performance.

The relative errors are computed in the $L2$ norm via

$$\varepsilon = \left(\frac{\int_{u_0}^{u_1} \mathrm{d}u \int_0^{2\pi} \mathrm{d}v \sqrt{g}(f_{\text{num}} - f_{\text{ana}})^2}{\int_{u_0}^{u_1} \mathrm{d}u \int_0^{2\pi} \mathrm{d}v \sqrt{g}f_{\text{ana}}^2}\right)^{1/2} \quad (43)$$

where $\sqrt{g}\mathrm{d}u\mathrm{d}v$ is the correct volume form in the u,v coordinate system. The ratio of N_v/N_u is chosen such that the aspect ratio of the resulting cells is approximately unity.

Let us point out first that if we choose a function $f_{\text{ana}} = f_{\text{ana}}(\psi)$ the flux aligned grids produce errors that can be orders of magnitude lower than those of the non-aligned grids. This is observed in Table 2. The orthogonal grid has much better errors than the elliptic grids. However, while the conformal and adapted grids have similar errors the monitor grid shows smaller errors for all resolutions. This is due to the fact that the degree of alignment is higher in case of the monitor grid (see, for example, [6] for a discussion). If due to some physical reasons the solution is expected to be flux aligned, the aligned grids are thus preferable. From the dG method we expect convergence of

22

N_u	N_v	Conformal		Adapted		Monitor		Orthogonal	
		P=3							
2	20	3.10E-02	-	1.37E-02	-	8.15E-03	-	7.34E-03	-
4	40	9.58E-03	1.69	4.80E-03	1.51	1.86E-03	2.13	4.67E-04	3.97
8	80	1.32E-03	2.86	1.17E-03	2.03	2.70E-04	2.78	4.28E-05	3.45
16	160	7.76E-04	0.76	2.74E-04	2.10	4.42E-05	2.61	4.26E-06	3.33
32	320	5.17E-05	3.91	5.02E-05	2.45	6.14E-06	2.85	3.58E-07	3.57
64	640	4.57E-06	3.50	7.41E-06	2.76	7.85E-07	2.97	3.23E-08	3.47
		P=4							
2	20	5.30E-03	-	8.72E-03	-	2.72E-03	-	3.01E-04	-
4	40	1.48E-03	1.84	1.61E-03	2.43	2.21E-04	3.62	1.32E-05	4.51
8	80	9.30E-04	0.67	3.87E-04	2.06	5.47E-05	2.01	1.10E-06	3.59
16	160	1.31E-04	2.83	6.96E-05	2.48	7.57E-06	2.85	4.14E-08	4.73
32	320	4.57E-06	4.84	1.00E-05	2.80	7.83E-07	3.27	2.01E-09	4.37
64	640	5.66E-07	3.01	1.07E-06	3.22	7.93E-08	3.30	9.61E-11	4.39

Table 2: Convergence table for Eq. (40) with the flux aligned solution given in Eq. (41) on the different grids proposed. We show the relative error (43) and the corresponding order for various numbers of polynomial coefficients P in the dG discretization. The orthogonal grid uses $\psi_0 = -20$ as the first line.

order P [27]. This is achieved only in ideal cases, however. We observe more irregular behaviour. The orthogonal grid shows approximate orders of 3.5 and 4.5, while the remaining grids show orders mostly between 2.0 and 3.5. Since the discretization method itself is equal in all four cases this irregularity is most likely due to the different metric elements in the four coordinate systems.

In actual turbulence simulations we expect localized structures and eddies as exemplified by the localized solution given in Eq. (42). In Table 3 we show

N_u	N_v	Orthogonal		Conformal		Near Conformal		Monitor	
		P=3							
4	40	1.24E+00	-	1.07E+00	-	6.67E+00	-	2.37E+00	-
8	80	3.38E-01	1.87	7.39E-01	0.54	5.35E-01	3.64	6.53E-01	1.86
16	160	2.57E-01	0.39	3.20E-01	1.21	2.92E-01	0.87	1.15E-01	2.51
32	320	3.68E-02	2.80	6.98E-02	2.20	1.76E-02	4.06	1.38E-02	3.06
64	640	3.15E-03	3.55	6.70E-04	6.70	1.58E-03	3.47	3.78E-04	5.18
		P=4							
4	40	2.39E+00	-	1.49E+00	-	1.84E+00	-	1.19E+00	-
8	80	6.41E-01	1.90	8.11E-01	0.88	7.59E-01	1.28	4.09E-01	1.54
16	160	1.18E-01	2.44	1.04E-01	2.96	9.81E-02	2.95	5.56E-02	2.88
32	320	1.56E-02	2.92	2.14E-02	2.28	7.66E-03	3.68	1.93E-03	4.85
64	640	6.78E-04	4.52	3.66E-04	5.87	2.62E-04	4.87	5.15E-05	5.23

Table 3: Convergence table for the solution of Eq. (40) with the localized solution given by Eq. (42) on the different grids proposed. We show the relative error (43) and the corresponding order for two different numbers of polynomial coefficients P in the dG discretization. The orthogonal grid uses $\psi_0 = -20$ as the first line.

the resulting convergence for a forward discretization on the orthogonal grid, the truly conformal grid, the near conformal grid, and the elliptic grid with monitor metric.

Note that we again observe irregular convergence behaviour for all grids. Nonetheless we ordered the grids from mostly high to lower errors from left to right. The grid constructed with the monitor metric has the lowest error. The worst grids are the orthogonal and the true conformal grids, which have the largest cell sizes at the location, where the blob was placed. The errors from the adapted grid are approximately between the near conformal grid and the grid with monitor metric.

The performance of these grids can be more clearly understood by comparing their maximal and minimal cell sizes for a fixed number of cells. Recall that the arc length of a curve is the integral over the norm of the tangent vector of the curve. In the case of the coordinate lines the tangent vectors are just the basis vectors ∂_u and ∂_v. If we assume that the computational space is discretized in equidistant cells with cell size h_u and h_v respectively, we define

$$l_u := \sqrt{g_{uu}}h_u = \sqrt{g}\sqrt{g^{vv}}h_u \tag{44a}$$
$$l_v := \sqrt{g_{vv}}h_v = \sqrt{g}\sqrt{g^{uu}}h_v \tag{44b}$$

Furthermore we define the ratio of maximal to minimal cell size by

$$a_u := \frac{\max l_u}{\min l_u} \quad a_v := \frac{\max l_v}{\min l_v} \tag{45}$$

	$\max l_u$	$\max l_v$	$\min l_u$	$\min l_v$	a_u	a_v
Orthogonal ($\psi_0 = -1$)	14.23	6.03	1.53	0.06	9.28	94.56
Orthogonal ($\psi_0 = -20$)	9.50	154.22	1.53	3.28	6.20	47.09
Conformal	57.02	79.51	1.75	2.44	32.53	32.53
Near Conformal	21.99	31.07	1.62	2.28	13.59	13.62
Adapted	58.91	20.05	1.68	2.04	34.98	9.82
Monitor	33.14	15.42	1.96	3.04	16.91	5.07

Table 4: Comparison of minimal and maximal cell sizes as defined in Eq. (44) and the ratios defined in Eq. (45) of the various grids with equal number of gridpoints $N_u = 32$, $N_v = 320$. The grids are ordered according to the ratio of maximum to minimum cell size in v from top to bottom.

In Table 4 we compute the maximum and minimum cell sizes of the various grids. For completeness we also computed the values for the orthogonal grid with the first line being the $\psi_0 = -1$ line, which is plotted in Fig. 2. We expect that if the maximal cell size is small, then the error committed by this grid is also small. In fact, this explains the results that were obtained in Table 3. That is, the monitor grid and the near conformal grid are better in terms of accuracy compared to the conformal and orthogonal grids. Note that the orthogonal grid with $\psi_0 = -20$ as the fist line has quite long cells in the v direction but smaller cells than all the other grids in the u direction.

25

The accuracy obtained is an important consideration. However, it has to be seen in the context that this elliptic equation will eventually be coupled to an evolution equation. In this case the minimal grid size is an important characteristics as it determines the CFL condition of explicit solvers and the condition number of the discretization matrix in implicit schemes and thus the computational efficiency of the grid used. Consider the minimal grid size in the v direction of the orthogonal grid with $\psi_0 = -1$ as the first line. It is 50 times smaller than the cell in the monitor grid. Thus, this grid is clearly not efficient for a practical numerical simulation.

The values of a_u and a_v are a measure of how well a grid performs. Small values indicate small errors in the elliptic equation and a large minimal cell size, which is advantageous for advection schemes. Overall we thus find that the monitor grid gives the best accuracy as well as the least stringent CFL condition among the grids we have considered here. This is expressed by a minimal value of a_v and the a_u value second only to the near conformal grid.

4. Conclusion

All in all, we show that we are able to construct various elliptic grids by the method of streamline integration combined with the solution of a suitably chosen elliptic equation. Compared to the TTM and related methods [4, 6, 7] we significantly simplify the construction and accuracy of the method. Compared to the flux-aligned grids suggested in the literature on magnetic fusion problems [16] we improve the distribution of cells across the domain and enable boundary orthogonality.

In our example implementation we discretize the approach laid out in Section 2 with high order Runge Kutta and dG methods. Our grids are suitable but not limited to the discretization of the edge region of magnetically confined plasmas as proved in Section 3. We explicitly show convergence of two solutions for a general elliptic equation that appears in typical physical models of plasma turbulence. Our orthogonal grids are flux aligned but show the largest variation in cell size of all the grids we investigated. The conformal grid to a lesser extent suffers from the same problem. In conclusion we find that the adaptive grid and even more the monitor metric approach yield the best grids in terms of small error and homogeneous cell distribution.

In the future we intend to investigate the possibility to include the X-point into the domain of interest. The difficulty encountered with our algorithm at an X-point or O-point is the vanishing gradient $\nabla\psi$. The Jacobian of any

flux aligned coordinate system thus has a singularity at this point and the convergence of elliptic equations significantly deteriorates or even vanishes. Numerical methods that can handle singularities might solve the problem. Furthermore, a generalization of our algorithm to three dimensions might be feasible.

Appendix A. Conformal and field aligned coordinates

The magnetic flux ψ does in general not satisfy the Laplace equation and thus is not able to serve as a conformal coordinate directly. It is therefore unclear whether it is possible to find a field aligned conformal coordinate system. Thus, we ask if it is possible to find a function u such that $u(\psi)$ satisfies the Laplace equation

$$\Delta u(\psi) = 0$$

which is equivalent to

$$(\nabla \psi)^2 u''(\psi) + (\Delta \psi) u'(\psi) = 0$$

Now we can rewrite this equation as

$$u''(\psi) = -\frac{\Delta \psi}{(\nabla \psi)^2} u'(\psi).$$

Since the left-hand side only depends on ψ, the same must hold true for the right-hand side. This implies that

$$\frac{\Delta \psi}{(\nabla \psi)^2} = g(\psi), \tag{A.1}$$

where g is an arbitrary function of ψ. Note that strictly speaking it is also possible to have $u'' = u' = 0$ which implies that $u(\psi) = $ const. This is certainly a solution but can be discarded for the purpose of constructing a coordinate system.

Equation (A.1) gives a condition on ψ that, if satisfied, allows us to construct a conformal field aligned coordinate system. Unfortunately, this is not a property that holds for the solutions of the Grad–Shafranov equation in general:

$$R\frac{\partial}{\partial R}\left(\frac{1}{R}\frac{\partial \psi}{\partial R}\right) + \frac{\partial \psi}{\partial Z^2} = -\mu_0 R\frac{\mathrm{d}p(\psi)}{\mathrm{d}\psi} - F(\psi)\frac{\mathrm{d}F(\psi)}{\mathrm{d}\psi} \tag{A.2}$$

27

For example, $\psi(R, Z) = R^2 Z^2$ is an equilibrium solution such that

$$\frac{\Delta \psi}{(\nabla \psi)^2} = \frac{1}{2\psi}\left(1 + \frac{1}{1 + R^2/Z^2}\right)$$

which clearly does not satisfy condition (A.1).

Note, however, that for specific equilibria the condition given by equation (A.1) can be satisfied. For example, $\psi(R, Z) = R^4$ satisfies the Grad–Shafranov equation and we have

$$\frac{\Delta \psi}{(\nabla \psi)^2} = \frac{1}{\psi}.$$

Both this and the equilibrium considered above are particular cases of the class of Solovév equilibria. Although the example is uninteresting for the purpose of turbulence simulations it is noteworthy that solutions that satisfy both (A.1) and (A.2) do exist.

Appendix B. Coefficients

For ease of reproduction we print the coefficients of the solovév equilibrium solution for ψ in Reference [26]

$$
\begin{aligned}
A ={}& 0.0, \\
c_{1..12} ={}& [0.07350114445500399706, -0.08662417436317227513, \\
& -0.14639315434011026207, -0.07631237100536276213, \\
& 0.09031790113794227394, -0.09157541239018724584, \\
& -0.00389228297983 7564482, 0.04271891225076417603, \\
& 0.22755456460027913117, -0.13047241360177695448, \\
& -0.03006974108476955225, 0.00421267189210393 1173], \\
R_0 ={}& 547.891714877869, \\
a/R_0 ={}& 0.41071428571428575, \\
\kappa ={}& 1.75, \\
\delta ={}& 0.47,
\end{aligned}
$$

These coefficients are also contained in the dataset for this paper [21].

Acknowledgements

References

[1] J. E. Thompson, Z. Warsi, C. W. Mastin, Numerical Grid Generation, Joe F. Thompson, 1997.

[2] N. Weatherill, B. Soni, J. Thompson, Handbook of Grid Generation, Informa UK Limited, 1998.

[3] V. D. Liseikin, A Computational Differential Geometry Approach to Grid Generation, Springer-Verlag, second edition, 2007.

[4] J. F. Thompson, F. C. Thames, C. W. Mastin, Tomcat - code for numerical generation of boundary-fitted curvilinear coordinate systems on fields containing any number of arbitrary 2-dimensional bodies, J. Comput. Phys. 24 (1977) 274–302.

[5] N. Papamichael, N. Stylianopoulos, Numerical Conformal Mapping, World Scientific Publishing, 2010.

[6] A. H. Glasser, V. D. Liseikin, I. A. Vaseva, Y. V. Likhanova, Some computational aspects on generating numerical grids, Russ. J. Numer. Anal. Math. Modelling 21 (2006) 481–505.

[7] I. A. Vaseva, V. D. Liseikin, Y. V. Likhanova, Y. N. Morokov, An elliptic method for construction of adaptive spatial grids, Russ. J. Numer. Anal. Math. Modelling 24 (2009) 65–78.

[8] J. Wesson, Tokamaks, Oxford University Press, 4th edition, 2011.

[9] S. Hamada, Hydromagnetic equilibria and their proper coordinates, Nucl. Fusion 2 (1962).

[10] A. H. Boozer, Guiding center drift equations, Phys. Fluids (1980) 904.

[11] A. H. Boozer, Plasma equilibrium with rational magnetic surfaces, Phys. Fluids (1981) 1999.

[12] R. Grimm, R. Dewar, M. J., Ideal MHD stability calculations in axisymmetric toroidal coordinate systems, J. Comput. Phys. 48 (1983) 94–117.

[13] C. Z. Cheng, Kinetic extensions of magnetohydrodynamics for axisymmetric toroidal plasmas , Physics Reports 211 (1992) 1–51.

[14] J.-k. Park, A. H. Boozer, J. E. Menard, Spectral asymmetry due to magnetic coordinates, Phys. Plasmas (2008) 064501.

[15] O. Czarny, G. Huysmans, Bezier surfaces and finite elements for mhd simulations, J. Comput. Phys. 227 (2008) 7423–7445.

[16] T. T. Ribeiro, B. D. Scott, Conformal tokamak geometry for turbulence computations, IEEE Transactions on Plasma Science 38 (2010) 2159–2168.

[17] A. Oliver, G. Montero, R. Montenegro, E. Rodríguez, J. M. Escobar, A. Perez-Foguet, Finite element simulation of a local scale air quality model over complex terrain, Adv. Sci. Res. 8 (2012) 105–113.

[18] W. D'haeseleer, W. Hitchon, J. Callen, J. Shohet, Flux Coordinates and Magnetic Field Structure, Springer Series in Computational Physics, Springer-Verlag, 1991.

[19] B. Scott, Shifted metric procedure for flux tube treatments of toroidal geometry: Avoiding grid deformation, Phys. Plasmas 8 (2001) 447–458.

[20] T. Frankel, The geometry of physics: an introduction, Cambridge University Press, second edition, 2004.

[21] M. Wiesenberger, M. Held, Feltor v3.1, Zenodo `http://doi.org/10.5281/zenodo.162407`, 2016.

[22] E. Hairer, S. Nørsett, G. Wanner, Solving Ordinary Differential Equations I, Nonstiff Problems, Springer-Verlag Berlin Heidelberg, 2nd edition, 1993.

[23] B. Cockburn, G. Kanschat, I. Perugia, D. Schotzau, Superconvergence of the local discontinuous Galerkin method for elliptic problems on Cartesian grids, SIAM J. Numer. Anal. 39 (2001) 264–285.

[24] M. Held, M. Wiesenberger, A. Stegmeir, Three discontinuous galerkin schemes for the anisotropic heat conduction equation on non-aligned grids, Comput. Phys. Commun. 199 (2016) 29–39.

[25] L. Einkemmer, M. Wiesenberger, A conservative discontinuous galerkin scheme for the 2d incompressible navier–stokes equations, Comput. Phys. Commun. 185 (2014) 2865–2873.

[26] A. J. Cerfon, J. P. Freidberg, ”one size fits all” analytic solutions to the grad-shafranov equation, Phys. Plasmas 17 (2010) 032502.

[27] D. N. Arnold, F. Brezzi, B. Cockburn, L. D. Marini, Unified analysis of discontinuous galerkin methods for elliptic problems, SIAM J. Numer. Anal. 39 (2002) 1749–1779.