Block-Structured Grids in Full Velocity Space for Eulerian Gyrokinetic Simulations

D. Jarema[a,*], H.J. Bungartz[a], T. Görler[b], F. Jenko[c,b], T. Neckel[a], D. Told[c,b]

[a] *Institut für Informatik, Technische Universität München, Boltzmannstraße 3, 85748 Garching, Germany*
[b] *Max-Planck-Institut für Plasmaphysik, Boltzmannstraße 2, 85748 Garching, Germany*
[c] *Department of Physics and Astronomy, University of California, Los Angeles, CA 90095, USA*

Abstract

Global, i.e., full-torus, gyrokinetic simulations play an important role in exploring plasma microturbulence in magnetic fusion devices with strong radial variations. In the presence of steep temperature profiles, grid-based Eulerian approaches can become quite challenging as the correspondingly varying velocity space structures need to be accommodated and sufficiently resolved. A rigid velocity space grid then requires a very high number of discretization nodes resulting in enormous computational costs. To tackle this issue and reduce the computational demands, we introduce block-structured grids in the all velocity space dimensions. The construction of these grids is based on a general approach, making them suitable for various Eulerian implementations. In the current study, we explain the rationale behind the presented approach, detail the implementation, and provide simulation results obtained with the block-structured grids. The achieved reduction in the number of computational nodes depends on the temperature profile and simulation scenario provided. In the test cases at hand, about ten times fewer grid points are required for nonlinear simulations performed with block-structured grids in the plasma turbulence code GENE (http://genecode.org). With the speed-up found to scale almost exactly reciprocal to the number of grid points, the new implementation greatly reduces the computational costs and therefore opens new possibilities for applications of this software package.

Keywords: Plasma turbulence, gyrokinetic simulation, block-structured grids, thermal speed disparity

[*] Corresponding author

1. Introduction

Gyrokinetics is widely regarded as an efficient and adequate model to simulate microturbulence in magnetically confined plasmas. However, the multiple spatial and temporal scales involved in such simulations render them computationally expensive. One such example is given by the structures in the velocity space. Here, the characteristic scales of the fluctuations of the distribution function are associated with the thermal speed $v_T = \sqrt{2T(x)/m}$, which is a function of species temperature $T(x)$ (x – radial coordinate) and mass m. One reason for the velocity disparities is the inclusion of different species in the simulations; which require different discretization grids for each species and thus lead to challenging numerical treatments of collisions [1, 2, 3, 4]. Another important cause of different velocity scales, which we address in the current study, is the spatial temperature variation $T(x)$.

A particular adaptation of the velocity grid to the background distribution function has already been described in a preceding publication [5] where computational grids consisting of blocks of rectangular regular grids have been introduced. In these grids, the range and resolution of each block is adjusted to a given temperature profile. The proposed computational grids were shown to be accurate and require considerably less computational nodes compared to the original regular grids. Furthermore, these grids reuse a large amount of the regular-grid-specific code, which allows constructing grids and extending the existing physics by different developers simultaneously. The study [5] describes the theoretical background for the whole velocity space (parallel velocity and magnetic moment). However, the main results for the global nonlinear gyrokinetic simulations consider only the parallel velocity direction. In this paper, we extend the approach to include the magnetic moment direction, which further reduces the necessary number of grid points. For instance, in the nonlinear tests presented in this paper, the number of discretization points has been reduced up to ten times without loss of accuracy. The construction of these extended block-structured grids differs significantly from the preceding version that treated the parallel velocity direction only, because we have to ensure accuracy and efficiency of operations such as the gyro-averaging and integrations along the magnetic moment direction.

The full velocity space block-structured grids have been implemented and tested in the Eulerian gyrokinetic code GENE (Gyrokinetic Electromagnetic Numerical Experiment) [6, 7, 8]. For the particle distribution function in a five-dimensional position-velocity phase space, GENE solves a set of nonlinear integro-differential Vlasov coupled to Maxwell equations numerically. In GENE, phase space and time are discretized separately according to the method of lines. The time evolution of the distribution function is solved by a fourth-order explicit Runge-Kutta method. To reduce computational costs, several efficient techniques targeting the mathematical model are applied, such as a δf-splitting of the distribution function and a field-aligned coordinate system (see [9, 10]). Further details on GENE's mathematical model and implementation are provided in [11, 12, 13].

Block-structured grids enable more efficient so-called global simulations by reducing the number of discretization points. In this global simulation regime, the radial extent of the simulation domain may cover up to the full fusion device size, taking into account the radial temperature and density profiles, and investigating the changes in plasma parameters dependent on the radial distance from the magnetic axis. Alternative multi-scale treatments coupling multiple radially local (flux-tube) domain simulations have been suggested in [14, 15]. However, here, cases with turbulence correlation lengths comparable to the length scales of the background density and temperature profiles are addressed. Radially global simulations are thus necessary.

The rest of the paper is structured as follows: The gyrokinetic equations solved in GENE and the mathematical background necessary for the construction of the grids are introduced in Section 2. Section 3 explains the choice of the velocity space resolution, range, and position of the computational nodes according to the radial profiles. Furthermore, we describe the implementation details and include examples of the computational grids in Section 4. Section 5 presents results of linear and nonlinear global gyrokinetic simulations. The contribution of this work is discussed and summarized in Section 6.

2. The gyrokinetic Vlasov-Maxwell system of equations

In this section, we introduce the fundamental notions that lie at the basis of the following sections. To this purpose, we provide a brief introduction to the gyrokinetic system of equations solved numerically by GENE and other similar codes to simulate microturbulence in magnetically confined fusion plasmas. Due to their high temperatures and therefore low collisionalities such plasmas do not thermalize. A fluid description is hence no longer sufficient, and kinetic effects have to be considered. A proper mathematical model is given by the Boltzmann equation or, in case of collisionless limit, the Vlasov equation. The corresponding approach is called kinetic and describes the evolution of a distribution function of all species involved in plasma in a six dimensional phase-space. Three of the coordinates define the position space and another three the velocity space. Due to several reasons, such as the high dimensionality of the model, computer simulations based on the kinetic model are prohibitively expensive for many important physical scenarios.

Fortunately, for such strongly magnetized plasmas the mathematical model can be simplified significantly. The gyration of the charged particles in this type of plasmas is much faster than the dynamics of interest. Therefore, information about the exact position of the particle on its orbit is irrelevant and can be discarded. By means of mathematical operations such as gyro-phase averaging

$$\bar{A}(\mathbf{X}) \equiv \frac{1}{2\pi} \oint \mathrm{d}\theta \; A(\mathbf{X} + \mathbf{r}(\theta)) \tag{1}$$

along the gyrophase angle θ and near identity Lie-transformations (see [16, 17]), we obtain a full-F five dimensional gyrokinetic equation (for species s)

$$\frac{\partial F_s}{\partial t} + \frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} \cdot \boldsymbol{\nabla} F_s + \frac{\mathrm{d}v_\parallel}{\mathrm{d}t}\frac{\partial F_s}{\partial v_\parallel} + \frac{\mathrm{d}\mu}{\mathrm{d}t}\frac{\partial F_s}{\partial \mu} = 0 \tag{2}$$

where the distribution F is a function of the gyrocenter coordinate $\mathbf{X}$, the velocity component parallel to the background magnetic field $v_\parallel$, and the magnetic moment $\mu = mv_\perp^2/2B$.

In Section 5, we provide results obtained from electrostatic simulations. In case of this limit, time derivatives of the introduced coordinates are given by

$$\begin{aligned}
\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} &= v_\parallel \mathbf{b}_0 + \frac{B_0}{B_{0\parallel}^*}\left(\mathbf{v}_E + \mathbf{v}_{\nabla B} + \mathbf{v}_c\right) \\
\frac{\mathrm{d}v_\parallel}{\mathrm{d}t} &= -\frac{\mathrm{d}\mathbf{X}/\mathrm{d}t}{m_s v_\parallel} \cdot \left(q_s \nabla \bar{\phi}_1 + \mu \nabla B_0\right) \\
\frac{\mathrm{d}\mu}{\mathrm{d}t} &= 0
\end{aligned} \tag{3}$$

where B_0 denotes the modulus of the magnetic field vector $\mathbf{B}_0$, $\mathbf{b}_0 = \mathbf{B}_0/B_0$ the corresponding unit vector, $B_{0\parallel}^* = \mathbf{b} \cdot \mathbf{B}_0^*$ the parallel component of $\mathbf{B}_0^* = \mathbf{B}_0 + \nabla \times (\mathbf{B}_0 v_\parallel/\Omega_s)$, and $\Omega_s = q_s B_0/m_s c$ is the gyrofrequency of species s with mass m_s. Moreover, three characteristic drift terms appear: the $\mathbf{E}\times\mathbf{B}$ velocity $\mathbf{v}_E = \frac{c}{B_0^2}\mathbf{B}_0 \times \nabla\bar{\phi}$, the gradient-B drift $\mathbf{v}_{\nabla B_0} = \frac{\mu c}{q_s B_0^2}\mathbf{B}_0 \times \nabla B_0$, and the curvature drift velocity $\mathbf{v}_c = \frac{v_\parallel^2}{\Omega_s}\left(\nabla \times \mathbf{b}_0\right)_\perp$.

In the case of the electrostatic limit, we need to compute the electric potential ϕ, by solving the Poisson equation

$$-\boldsymbol{\nabla}^2 \phi = 4\pi \sum_s q_s n_s \tag{4}$$

where the Coulomb gauge is used, q_s and n_s are the charge and density of species s, respectively. The density is the zeroth order velocity space moment of the distribution function $n(\mathbf{x}) = M_{00}(\mathbf{x})$. A velocity space moment of an arbitrary order is computed by

$$M_{ab}(\mathbf{x}) = \int \delta\left(\mathbf{X} - \mathbf{x} + \boldsymbol{\rho}\right) T^* F\left(\mathbf{X}, v_\parallel, \mu\right) \frac{B_\parallel^*\left(\mathbf{X}, v_\parallel\right)}{m} v_\parallel^a v_\perp^b \,\mathrm{d}^3X\,\mathrm{d}v_\parallel\,\mathrm{d}\mu\,\mathrm{d}\theta \tag{5}$$

where T^* is a pull-back operator transforming the gyrocenter distribution function, whose evolution is described by (2), to the particle distribution function, and $(\mathbf{X}, \mu, v_\parallel, \theta)$ are guiding center coordinates (see [18]). For the exact expression of the pull-back operator used in GENE we refer to [12, 19]. In practice, the velocity space moments can be efficiently computed in three steps: an integration over $v_\parallel$, gyro-phase averaging given by

$$\langle F \rangle = \frac{1}{2\pi}\int \delta\left(\mathbf{X} - \mathbf{x} + \boldsymbol{\rho}\right) F\left(\mathbf{X}\right)\,\mathrm{d}^3\mathbf{X}\,\mathrm{d}\theta \tag{6}$$

and then an integration over the magnetic moment coordinate.

GENE belongs to the class of δf-codes, in which the gyrokinetic Vlasov equation is further simplified by splitting the distribution function F into a local Maxwellian background F_0 and a fluctuating part f_1 (for details on this technique see [20, 21]). The fluctuating part is by one order smaller than the background $f_{1s}/F_{0s} \sim \epsilon$, according to the gyrokinetic ordering described in [16].

The background distribution function exploited in GENE depends on the species density $n_s(x)$ and temperature $T_s(x)$ radial profiles

$$F_{0s}(x, v_\parallel, \mu) = \frac{n_s(x)}{\pi^{3/2} v_T^3(x)} \exp\left[-\frac{m v_\parallel^2/2 + \mu B}{T_s(x)}\right]. \tag{7}$$

This expression describes how the equilibrium temperature, density and thermal speed appear in the gyrokinetic equation. In the following section, we rely on this choice of the background distribution function to construct block-structured grids in the velocity space.

To make grid-based gyrokinetic simulations more efficient, some further simplifications of the underlying equations are realized in the δf-code GENE. To start with, only the first order nonlinearity $\mathbf{E} \times \mathbf{B}$ is retained. Furthermore, the parallel nonlinearity and some other higher order terms can be switched on for testing, but they are usually found to be insignificant, e.g., see [22]. Significant savings in the number of grid points are achieved in the position space by introducing curvilinear field-aligned coordinates. These coordinates help exploit the high anisotropy of plasma turbulence and avoid some derivatives along the magnetic field lines. Other details on the final version of the resulting system of equations, including collisions and electromagnetic effects, can be found, e.g., in [11, 12, 13]. These are, however, not relevant to the following discussion.

3. Velocity space discretizations

3.1. Default grids in GENE

Eulerian gyrokinetic codes employ five-dimensional grids to simulate plasma turbulence. Three dimensions span the position space and two others the velocity space.

GENE exploits field-aligned coordinates in the position space (x, y, z), where x is the radial, y the binormal, and z the direction following the magnetic field line. These coordinates are optimal to resolve strongly anisotropic fluctuations, whose parallel correlation length exceeds the perpendicular one by several orders of magnitude. Details on the position space discretization in GENE can be found in [23], and general information about field-aligned coordinates in [24]. In the rest of the paper, the radial direction will be the most relevant position space coordinate since equilibrium quantities such as temperature and density are constant along y and z in toroidal devices but not in x which affects the equilibrium thermal velocities that need to be covered by the corresponding grids. To discretize the radial distance coordinate, GENE employs a finite difference scheme in combination with Dirichlet and von Neumann boundary conditions. Furthermore, to avoid potential numerical instabilities close to the boundaries due to the Dirichlet condition a Krook-type buffer zone is introduced (cf. [12, 13, 19]).

For the two-dimensional velocity space, a grid with parallel velocity $v_\parallel$ and magnetic moment μ coordinates is chosen. Here, the magnetic moment is defined by $\mu = mv_\perp^2/2B$, where $v_\perp$ is the perpendicular velocity component, m the species dependent mass, and B the strength of magnetic field. In GENE, $v_\parallel$ is discretized via a finite difference scheme on an equidistant mesh with homogeneous Dirichlet boundary conditions $f = 0$. During collisionless plasma simulations, GENE performs only integration-like operations in the μ direction. This is reflected in the type of grids chosen for the magnetic moment coordinate. The default option is Gauss-Laguerre nodes, which are optimal for quadratures of exponential multiplied by polynomial function. The choice is motivated by the exponential shape of the background distribution function and by the fact that the perturbed part approximately retains a similar shape. In many numerical experiments, the Gauss-Laguerre quadrature requires tremendously less (up to 32 times) discretization nodes in the magnetic moment direction than a counterpart equidistant mesh to converge to the same precision results. An example of the default velocity space grid at fixed position space coordinates is provided in Figure 1.

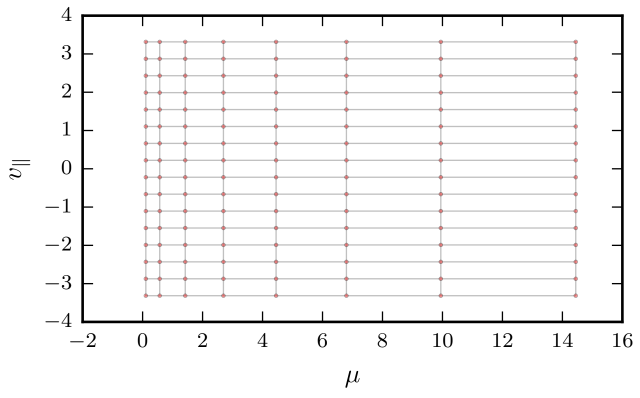

Figure 1: An example of a discretization grid in the velocity space.

3.2. Grid adaptation to temperature profiles

As was shown in [5], adjusting computational grids to the background distribution function leads to a substantial reduction of grid points without a significant loss of precision. The reason for this is that the perturbed distribution function approximately follows the shape of the background distribution function when it is not localized at a certain radial distance.

In GENE, the Maxwellian distribution is chosen as the background distribution by default. It yields the following distribution functions in the $x - v_{\parallel}$ and $x - \mu$ subspaces:

$$F_0\left(x, v_{\parallel}\right) = n(x)\sqrt{\frac{m}{2\pi T(x)}}\exp\left[-\frac{mv_{\parallel}^2}{2T(x)}\right], \tag{8}$$

$$F_0(x, \mu) = n(x)\frac{B}{T(x)}\exp\left[-\frac{\mu B}{T(x)}\right] = n(x)\lambda e^{-\lambda\mu}. \tag{9}$$

Obviously, they become radially dependent through the plasma density and temperature profiles $n(x)$ and $T(x)$. Furthermore, the temperature appears in the exponential factors and, thus, strongly influences the shape of the background distribution function. For example, $F_0(x, v_{\parallel})$ takes a form of a Gaussian bell curve with a wide peak at high temperatures and a sharp peak at low temperatures. Therefore, compared to the case of regions with low temperatures, at high temperatures we need velocity grids with a relatively wide range and low resolution. The velocity range can be computed at each radial distance as a confidence interval of a fixed probability. If there is no data from previous simulations, the simulation may be started with a minimum range of three standard deviations, which corresponds to a probability of 99.7%. Figure 2 (bottom) shows an example of the domain in the $x - v_{\parallel} - \mu$ subspace, obtained by fixing the probability of the confidence interval to 99.7% and using temperature profiles (see Figure 2 (top)) of a particular "Tokamak à Configuration Variable" (TCV) discharge [19].

Furthermore, it is not just the velocity space grid that has to be adjusted due to the temperature radial variations. The thermal gyro-radius, responsible for the scales of the drift wave turbulence, is influenced as well. Therefore, the radial and binormal coordinates require a finer resolution in the low temperature regions. In the current study, we do not adapt the x and y discretization to the temperature variations, and use sufficiently resolved grids in the whole position space in the examples provided. This kind of adjustment is part of our future work.

A simple theoretical way to solve the problem of the radial temperature variations would be the normalization of the velocity coordinates by the thermal speed $v_T = \sqrt{2T(x)/m}$. After the change of variables $v_{\parallel}^{\text{norm}} = v_{\parallel}/v_T$ and $\mu^{\text{norm}} = \mu/v_T^2$, one may remove the explicit temperature dependence from the background distribution function. Furthermore, with the new coordinates, one automatically obtains correct ranges and resolution of the physical velocity coordinates, by keeping the same new velocity grid at each radial position. The simplicity of this approach comes nonetheless at high costs: it demands the change of the mathematical model and introduces a large number of additional terms in the governing equations, some of which are nonlinear. All x-derivatives are changed according to $d/dx \to d/dx + C_1(x) \cdot d/dv_{\parallel} + C_2(x) \cdot d/d\mu$. In addition, many other terms (such as the background Maxwellian and the effective potential χ) pick up additional x dependencies. The combination of these effects leads to a significant increase in the overall complexity of the equations. Moreover,

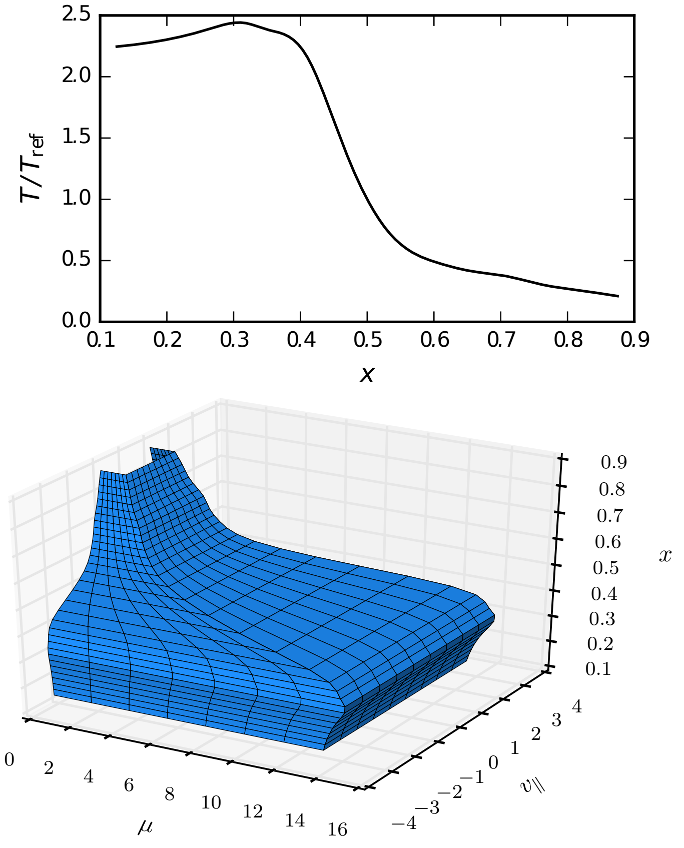

Figure 2: The TCV electron temperature profile (top) and an example of the simulation domain in the $x - v_\parallel - \mu$ subspace (bottom). The radial distance is shown in the minor radius units, the temperature values are shown in units relative to the reference values T_{ref} taken at the reference radial distance ($x_{\mathrm{ref}} = 0.5$).

due to the changes in the set of equations, the whole application would have to be reimplemented from scratch. This prohibits simultaneous improvements of the grids and extensions of the physics by several developers, which is essential for heavily-used codes such as GENE.

3.3. Block-structured grids

To achieve the same effect as the normalization by thermal speed and keep the original mathematical model, we introduce block-structured grids in the velocity space. The idea is to approximate the desired simulation domain in the velocity – radial distance subspace, see Figure 2 (bottom), by rectangular cuboid-shaped blocks, as shown in Figure 3. To obtain an appropriate ratio of resolutions in all cuboids, each block should contain the same number of velocity grid points.

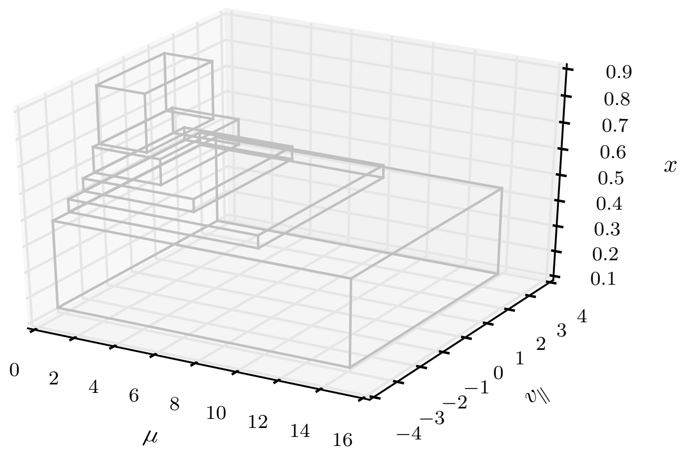

Figure 3: Approximation of a simulation domain in the $x - v_\parallel - \mu$ subspace by rectangular cuboid-shaped blocks.

The effect of the blocking is clearly visible in the examples provided in Figure 4, showing grids in the $x - v_\parallel$ subspace. The four top subplots (`A, B, C, D`) display the $v_\parallel$ coordinate lines of the normalized grid (`A`), the default regular grid (`B`), the block-structured grid with the same resolution in each block (`C`), and the block-structured grid with a fixed number of grid points (`D`); the horizontal axis is the physical (non-normalized) parallel velocity coordinate. The same coordinate lines (but with the horizontal axis corresponding to the normalized parallel velocity) are shown in the four bottom subplots (`E, F, G, H`) of Figure 4. The coordinate lines are shown for the same TCV temperature profile as in the previous examples. From these figures, we observe that if the default regular grid has both a sufficient extent and a resolution fine enough to satisfy both high and low temperature regions, it then consumes more grid points than the grid with the normalized parallel velocity. This happens because we invest too many $v_\parallel$ nodes in the high temperature region and take too wide a range in the low temperature region, which is apparent from Figure 4 (`F`). The problem with the unnecessary wide range in the low temperature region can be solved by cutting away the insignificant grid points, as shown in Figure 4 (`C`). However, the high temperature region is still overresolved, see Figure 4 (`G`). Finally, we get appropriate ranges and resolutions if we keep the same number of $v_\parallel$ grid points in each block as in Figure 4 (`D, H`). Furthermore, in the last example with just six blocks, we obtain practically the same number of $v_\parallel$ grid points as in the case of the normalized velocity grid. The latter approach was first introduced and described for the $v_\parallel$ coordinate in [5]. In the following, we extend this method to include the μ direction and provide theoretical and implementation details.

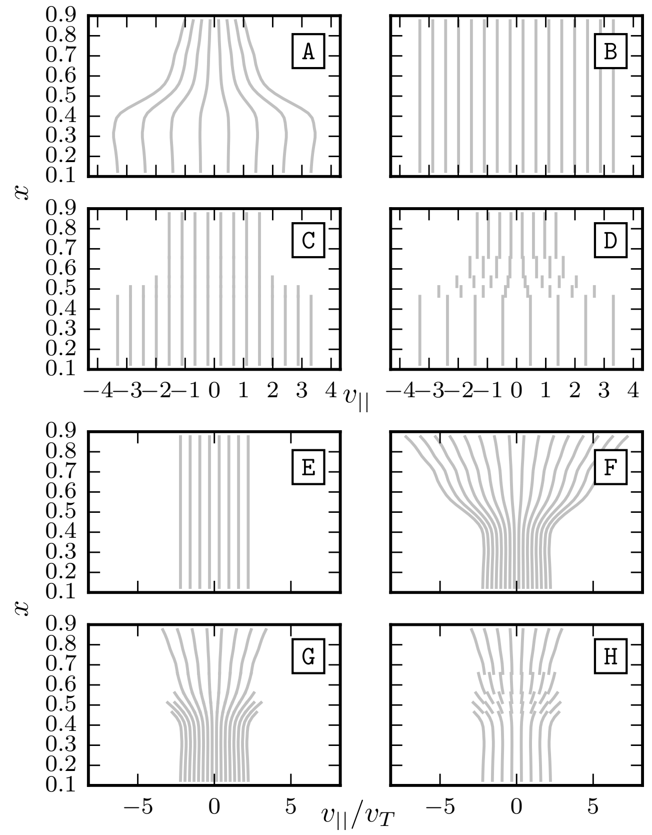

Figure 4: Coordinate lines $v_{\|}$ of the normalized grid (`A, E`), default regular grid (`B, F`), block-structured grid with the same resolution in each block (`C, G`), and block structured grid with a fixed number of grid points (`D, H`). Top subplots (`A, B, C, D`): horizontal axis — physical parallel velocity coordinate. Bottom subplots (`E, F, G, H`): horizontal axis — normalized parallel velocity coordinate.

Magnetic moment discretization

As was mentioned before, the main operations performed in the magnetic moment direction are integration and gyro-averaging and, therefore, we employ the Gauss-Laguerre quadrature rule to efficiently compute integrals with a minimum number of grid points. We fix the position of quadrature nodes (grid points) for all radial distances in the default regular grids in GENE. These nodes μ_m and weights w_m are computed for integrals of the type

$$\int_0^\infty e^{-\mu} p(\mu)\,\mathrm{d}\mu = \sum_{m=1}^{\mathrm{nw0}} w_m p(\mu_m) \tag{10}$$

with the weight function $e^{-\mu}$. Here, nw0 is the number of μ nodes and $p(\mu)$ is a polynomial of maximum degree $2 \cdot \mathrm{nw0} - 1$. For the profiles with significant temperature variations, however, this quadrature rule works fine only at the reference point ($x_{\mathrm{ref}} = 0.5$), due to the temperature normalization T/T_{ref}, which yields the exponential factor $\lambda = 1$ at x_{ref} in (9). At all other radial points, the precision of the quadrature rule deteriorates drastically, as shown in Figure 5 for the dash-dot-line. Here, we plotted the error of the Gauss-Laguerre rules

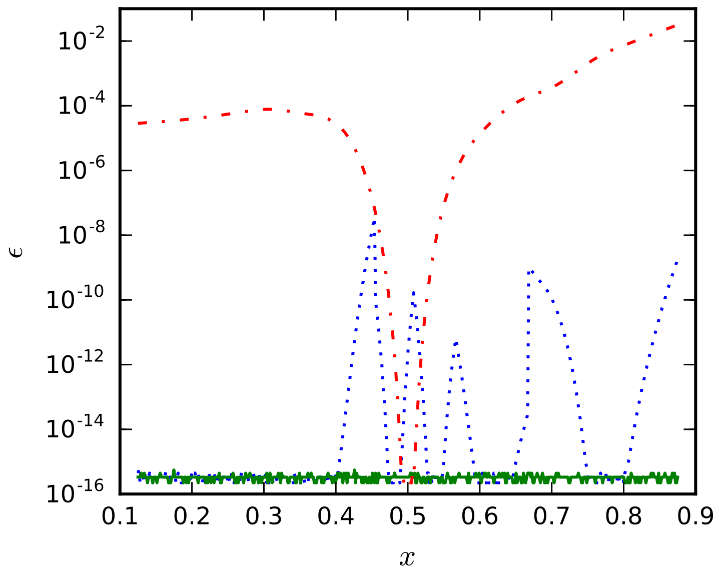

Figure 5: Dependence of Gauss-Laguerre quadrature error on the radial distance coordinate, for quadrature rules with number of nodes $\mathrm{nw0} = 8$. Solid line — ideally adjusted quadrature rule at each radial point. Dash-dot-line — the quadrature rule is fixed for the reference point ($x_{\mathrm{ref}} = 0.5$). Dot-line — the quadrature rule is separately chosen for each block in the radial direction.

at different radial distances for the integration of the background distribution function.

Adjusting the quadrature rule to each radial position makes the integration precise again, see Figure 5 (solid line). However, it leads to different μ nodes at each x and makes the radial derivative computations too complicated. Another solution is to prescribe a different set of μ_m nodes and weights w_m in every block of the block-structured grid described previously. With just six blocks we get quadratures with the precision close to the ideally adjusted rule, as shown in Figure 5 for the dot-line. Furthermore, the radial derivative computations require no modifications within the blocks and only a few additional computations on the boundaries are necessary, see Section 4.

In the case of global gyrokinetic simulations, it is often necessary to take more μ quadrature nodes than it would be required for a quadrature of some analytic function. This happens because of the oscillatory nature of the perturbed distribution function and gyro-averaging, which is a more complicated operation than the mere integration in the magnetic moment direction. A high number of the quadrature nodes yields not only a better resolved grid close to $\mu = 0$ axis, but also large values of the last μ_m points. The background distribution function at large μ_m takes very small values. According to the gyrokinetic ordering, the fluctuating part is by an order smaller than the background. Therefore, due to numerical round-off errors the values of the fluctuating part at particulary high μ do not contribute a lot to the computations of the integrals. Taking all this into consideration, we rescale the given nodes to ensure that they all fit in the prescribed range. Moreover, to keep the Gauss-Laguerre quadrature rule valid, we rescale the weights by the same factor. Such an operation, however, modifies the quadrature rule, which was ideally adjusted for the background distribution at the given radial point.

In this paper, presented results (Section 5) are obtained with rescaled magnetic moment nodes and weights. For the sake of completeness, however, we describe an alternative approach, which keeps the quadrature rule optimal for the given background and the grid nodes within the given magnetic moment range. Instead of integrating from zero to infinity in (10), we take a finite range integral

$$\int_0^{\mu_{\max}} e^{-\mu} p(\mu)\,\mathrm{d}\mu = \sum_{m=1}^{\mathrm{nw0}} w_m p(\mu_m) \tag{11}$$

where $\mu_{\max}$ is sufficiently large and usually computed as a confidence interval of a prescribed probability p ($\mu_{\max} = -\ln(1-p)$). When the exponential decay factor $\lambda \neq 1$, we look for weights and nodes of the following quadrature rule:

$$\int_0^{-\ln(1-p)/\lambda} e^{-\lambda\mu} p(\mu)\,\mathrm{d}\mu = \sum_{m=1}^{\mathrm{nw0}} w_m^{(\lambda)} p(\mu_m^{(\lambda)}) \tag{12}$$

This quadrature rule can be directly obtained from (11) by change of variable $x = \mu/\lambda$ in the integral:

$$\int_0^{-\ln(1-p)/\lambda} e^{-\lambda\mu} p(\mu)\,\mathrm{d}\mu = \frac{1}{\lambda}\int_0^{\ln(1-p)} e^{-x} p(x)\,\mathrm{d}x = \sum_{m=1}^{\mathrm{nw0}} \frac{w_m}{\lambda} p\left(\frac{x_i}{\lambda}\right) \tag{13}$$

Hence, we conclude that $w_m^{(\lambda)} = w_m/\lambda$ and $\mu_m^{(\lambda)} = \mu_m/\lambda$. Efficient computations of the generalized Gauss quadrature are explained in [25], see also Chapter 18 in [26]. The same scaling rule is valid for integrals from zero to infinity.

Besides the integration, the gyro-averaging numerical procedure is also affected by the discretization of the magnetic moment direction. The details on how this is done for the block-structured grids are provided in AppendixA.

4. Implementation details of block-structured grids

The block-structured grids reuse the code already written and tested with the regular grids in GENE. The changes introduced by the new grids are classified in three categories: prefactors, block boundaries, and parallelization.

The prefactors, which have to be modified, correspond to different coefficients in the set of equations. These coefficients depend on the mesh size $\mathrm{d}v_{\|}$, and the nodes and weights from the Gaussian quadrature rule (μ_m and w_m). In the regular grids, the velocity mesh size and the quadrature rule are kept constant for all radial positions. In the block-structured grids, however, these quantities differ in each block. Therefore, they have to be adjusted correspondingly. In most cases, this is a simple and straightforward procedure. The technically most challenging modification is the computation of the gyro-matrix for the modified magnetic moment grids, see AppendixA. The rows of the gyro-matrix $\mathcal{G}$ are blocked in the same way as the radial distance and each block of rows is computed with its own set of the magnetic moment nodes and weights.

4.1. Treatment of block boundaries

Modifications are also necessary on the boundaries of the grid blocks. For regular grids, GENE uses a finite difference scheme to discretize the radial derivative. In our scheme, when all grid points lie inside the blocks, we apply the same derivative computations within the blocks. However, due to the misalignment of the grid points on the block boundaries, additional nodes have to be added for the finite difference scheme. A sketch of such a misalignment is shown in Figure 6, where two velocity grids are taken from neighboring grid blocks.

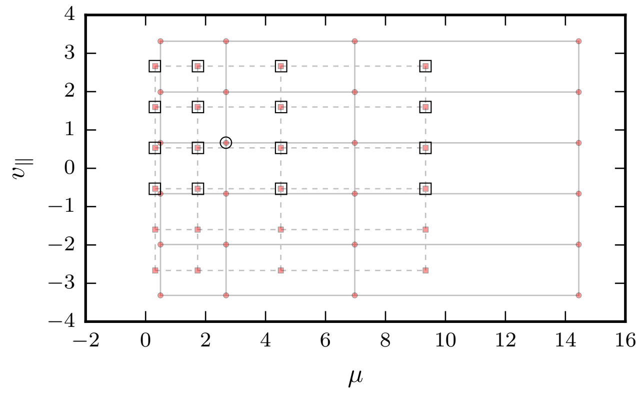

Figure 6: An example of two overlapping grids in the velocity space. One of the grids is marked by solid coordinate lines, another by dashed ones. These two grids are taken from adjacent blocks.

This problem is addressed by interpolating values at the missing locations and using the interpolated values in our finite difference scheme. We apply local polynomial interpolation — the polynomials are defined only around a corresponding interpolation point. For instance, to compute the value at the node surrounded by a circle in Figure 6 on the grid with dashed coordinate lines, for the fourth order interpolation scheme we span a polynomial through those points enclosed by squares. In real life simulations, the resolution of the grid in the velocity space is much finer than in the provided example and the nodes used for the interpolation are more localized.

To interpolate efficiently at missing locations, we introduce ghost grids for saving the interpolated values. The ghost grids are prolongations of the blocks to the neighboring blocks, where the values have to be interpolated for the finite difference scheme. Basic examples of the $x - v_{\|}$ and $x - \mu$ subspace grids with and without ghost layers are shown in Figure 7. The ghost grids are used only to

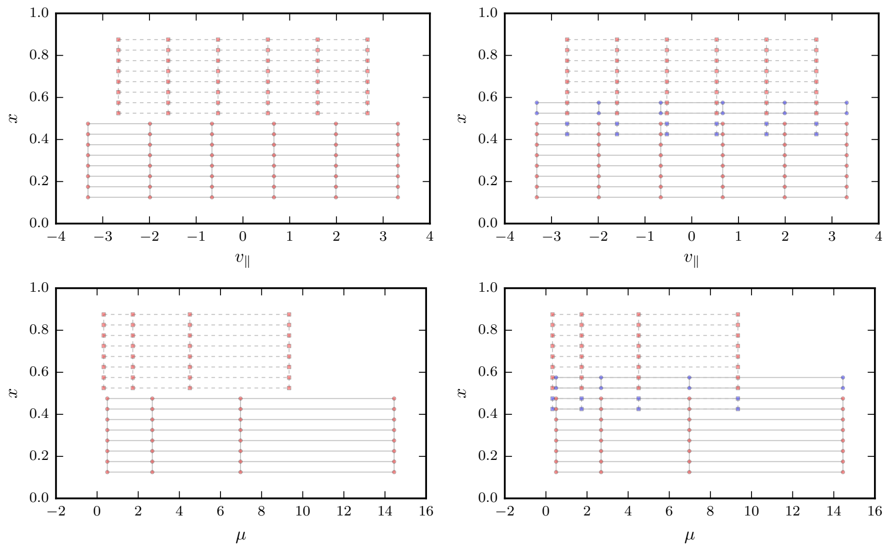

Figure 7: Examples of grids in the $x - v_{\|}$ and $x - \mu$ subspaces, with (right plots) and without (left plots) ghost grids for interpolation.

save the interpolated values and not to have the simulation run on their nodes. Therefore, the ghost grids do not increase the computational complexity, but only the memory requirements to a small extent. For example, in the grids consisting of five blocks used for the nonlinear test in Subsection 5.2, the nodes of the ghost grids amount to only three percent of the total number of the grid points. In view of the final reduction of the number of the grid nodes, which totals up to around ten times less in comparison to the regular grid, the memory additional costs due to the ghost grids are negligible.

The m-order polynomial interpolation leads to errors of order $\mathcal{O}(\Delta v_{\|}^m/\Delta x)+\mathcal{O}(\Delta\mu^m/\Delta x)$ in the radial derivative computations, where $\Delta\mu$, $\Delta v_{\|}$, and Δx are the mesh sizes in the magnetic moment, parallel velocity, and radial distance directions. If the mesh sizes in the velocity space and radial distance are of comparable scales, we consider these inaccuracies of order $m-1$. If necessary, these errors can be eliminated by the method described in [27]. However, in this case, simulations also have to be run on the introduced ghost grids.

4.2. Parallelization with block-structured grids

Most of the parallelization schemes developed for the regular grids remain the same. Only the exchange in the radial direction, which is necessary to compute radial derivatives, undergoes modifications. The radial exchange is schematically represented by arrows in Figure 8 for the $x-v_{\|}$ and $x-\mu$ subspaces. The communication marked by vertical arrows (vertical exchange) is inherited

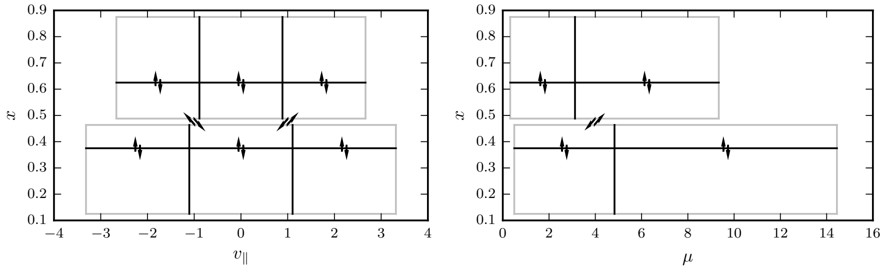

Figure 8: A schematic demonstration of the radial exchange in the $x — v_{\|}$ and $x — \mu$ subspaces of a two-blocks grid. The gray lines represent the outlines of the grid blocks, the black lines are the boundaries of the process grid. The arrows show which processes are communicating.

from the regular grids. The slanted arrows (side exchange) represent the exchange introduced due to the misalignment of the process grids in the different blocks of the discretization grid. The side exchanges typically require less data to be sent than the vertical ones, but are more difficult to implement. There are several possible ways to organize the radial exchange, for instance by combining the side exchanges with the exchanges in the velocity space directions, and then have the vertical exchange performed the same way as in the regular grid. This, nevertheless, does not keep the radial exchange decoupled from the communications in the velocity space directions, nor does it hide the complexity of the radial derivative computations in the specially developed data structures. Therefore, we start the computations of the radial derivative and initialize the side exchanges after the vertical exchange is done. In the ideal case, these computations have already been performed before we start computing derivatives near the block boundaries.

To illustrate the communication during the side exchange, we provide in Figure 9 two grids in the velocity space from adjacent blocks, where the radial direction is orthogonal to the plane of the figure. When we compute the radial derivative on the smaller grid (marked by dash-lines), we take values from the larger grid (marked by solid lines). For example, if we compute the radial derivative in the region marked by a circle, we have to obtain values from the

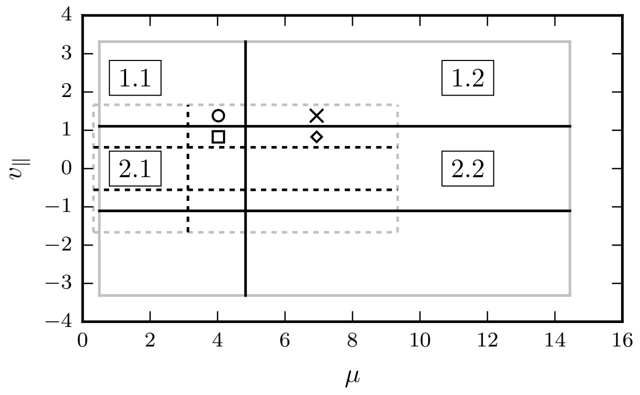

Figure 9: An example of outlines and process grids of two velocity space grids from adjacent blocks. The outline of the smaller grid is plotted with gray dash-lines and the process grid with black dash-lines. The outline and the process grid of the larger grid are plotted with solid lines.

process with index 1.1, in the region marked by the cross from process 1.2, in the region marked by the square from process 2.1, and in the region marked by the diamond from process 2.2. The mentioned processes (1.1, 1.2, 2.1, and 2.2) keep the parts of the ghost grids of the neighbor blocks, which overlap with their local domains. The interpolated values on the ghost grid are precomputed when the computational grid values are available, so they can be directly used for the radial derivative computations. Close to the boundary of the process grid, a process might not possess all computational grid nodes necessary to compute the interpolated values on the corresponding ghost grid. In this case, the process computes only a part of the interpolation sum. Then, the process computing the radial derivative collects and sums up all partly computed interpolation sums.

5. Results

GENE is capable of simulating both linear and nonlinear scenarios. The linear simulations do not include the nonlinear terms of the gyrokinetic set of equations, which results in computationally much cheaper runs. Of course, this type of operation cannot be used to study turbulence but is sufficient to investigate the underlying microinstabilities. The latter are mainly characterized by their linear growth rates and real frequencies which can easily be determined. Both the relatively inexpensive type of simulation and the small and easy accessible number of observables make linear simulations a prime testing tool for the newly developed grids. However, the nonlinear turbulence simulations are the ultimate target of the block-structured grids, because they are extremely demanding in computational resources. Therefore, improving the nonlinear simulations in terms of speed and memory footprint is a prerequisite for many interesting scenarios. The results of the linear and nonlinear simulations obtained with the block-structured grids are provided in the following subsections.

5.1. Linear Simulations

In the linear simulations, we compute the linear growth rate (γ) and frequency (ω) of the dominant mode. There are two main ways to obtain these observables: by finding the eigenvalues of the linearized right hand side of the system of gyrokinetic equations or by simulating the initial value problem. The comparison of these two approaches is described in [28]. The nonlinear initial value problem comprises the linear problem and several additional nonlinear terms. By developing and testing the block-structured grids with the linear initial value solver, we automatically partly address the target nonlinear simulations. Therefore, in this subsection, we present results obtained by simulating the initial value problem. The eigenvalue solver, nevertheless, also benefits from our approach, due to smaller resulting matrices; details about the eigenvalue solver in GENE are provided in [29].

In all the tests involving linear simulations presented in this section, we choose a circular magnetic geometry with concentric flux surfaces and a safety factor profile given by

$$q(r/a) = 2.2(r/a)^2 + 0.868 \tag{14}$$

where a is the minor radius, the aspect ratio at the last flux surface is set to $a/R = 0.35$ (R — major radius), and $\rho^* = \rho_{ref}/a = 1/80$ (ρ_{ref} — reference gyroradius). The choice of the circular magnetic geometry is motivated by the fact that the position space grid requires less discretization nodes for this simple geometry than for realistic configurations based on arbitrary magnetohydrodynamics (MHD) equilibria. As a result, we obtain less expensive simulations. To further reduce the computational costs, we simulate only collisionless electrostatic plasma.

The fluctuating part of the distribution function stemming from linear simulations might appear localized in the direction of the radial distance (see [5]). In such cases, not all the beneficial properties of block-structured grids are apparent, because the simulation results are sensitive mostly to the grid resolution and range in the localization region. The localization of the perturbed distribution function can be controlled by the temperature and density profiles of each of the two species involved in our specific simulations (electrons and protons). For the sake of having a simpler control and also the ability to use the same block-structured grid for each of the two species, in our linear simulations, electrons and protons have the same temperature and density radial profiles. Furthermore, we set the same radial function, which is shown in Figure 10, to compute the temperature and density profiles. The corresponding initial value computation yields a perturbed distribution function with a wide structure in the radial direction as shown in Figure 11. These plots show the absolute values of the distribution function projections on the radial distance – parallel velocity (left plots) and the radial distance – magnetic moment (right plots) subspaces.

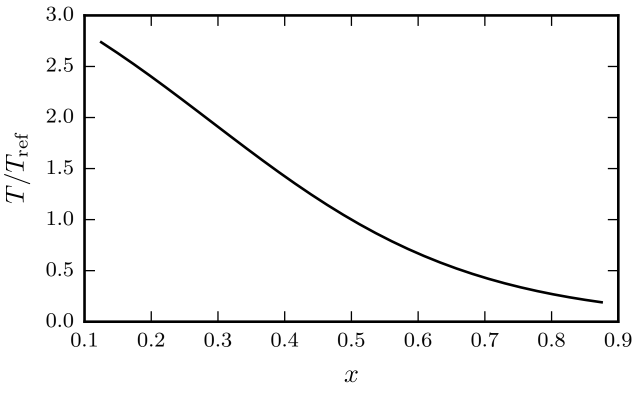

Figure 10: Radial temperature profile used for testing block-structured grids with linear simulations.

The resulting perturbed distributions were obtained using block-structured grids and are shown here for both electrons (top plots) and protons (bottom plots). The color map used in the right plots is logarithmic, to make the shape of the distribution function clearly visible. As a result, the block shapes also become manifest.

To compare block-structured grids with regular ones, we check how quickly γ and ω converge with respect to the numbers of grid points (nv0 and nw0) in the parallel velocity and magnetic moment directions. To minimize the discretization errors stemming from other directions, we fix the number of grid points in these directions to relatively high values $(\mathrm{nx0}, \mathrm{nz0}) = (256, 16)$. Moreover, in this linear test scenario, the toroidal mode number is set to $k_y = 0.3$. The convergence plots are obtained in two stages. First, to find at which nv0 the results converge, we fix the number of magnetic moment nodes to a sufficiently large number $(\mathrm{nw0} = 96)$ and keep the grid in the radial direction – magnetic moment regular. The convergence plots for nv0 are provided in Figure 12. The values of γ and ω are here normalized to the results converged to a precision of three digits after the decimal point, $\gamma_{\mathrm{conv}} = 4.810$ and $\omega_{\mathrm{conv}} = 1.376$. Apparently, $\mathrm{nv0} = 158$ grid points are sufficient in the block-structured grid whereas the regular grid requires $\mathrm{nv0} = 266$. This result demonstrates that we can already remove around 41% of the grid points without a loss of accuracy. Next, we fix the resolutions for the block-structured grid $(\mathrm{nx0}, \mathrm{nz0}, \mathrm{nv0}) = (256, 16, 158)$ and for the regular grid $(\mathrm{nx0}, \mathrm{nz0}, \mathrm{nv0}) = (256, 16, 268)$, and compute the growth rate and frequency for different numbers of grid points in the magnetic moment direction nw0. The convergence plots for nw0 are provided in Figure 13 (left). In this case, the block-structured grid converges to a slightly different growth rate $\gamma = 4.809$ and frequency $\omega = 1.375$, the differences compared to the reference grid being $\Delta\gamma = 0.02\%$ and $\Delta\omega = 0.07\%$. Possible reasons for this deviation are the removal of a significant part of the x — μ subspace from the simulations

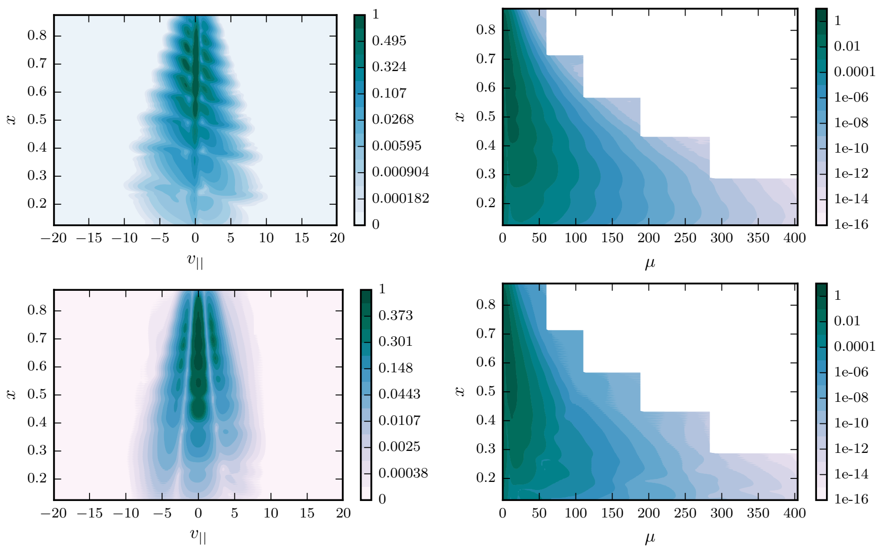

Figure 11: The absolute value of the perturbed distribution function in the radial distance — velocity subspaces. The top plots correspond to electrons, the bottom plots to protons.

with the block-structured grids, see Figure 11 (right), and more significant modifications of the numerical implementation due to the blocking in the magnetic moment direction compared to the modifications due to the parallel velocity blocking. Deviations of similar orders are nevertheless observed in the case of the regular grids as well, when starting the initial value solver with different initial perturbed distribution functions. The deviation is therefore negligible.

From Figure 13 (left), we observe that the results converge very fast for both the block-structured grid (around $8 - 10$ μ grid points) and the regular grid (around $12 - 14$ μ grid points). The important effect of the block-structured grid in this case is that it allows to significantly reduce the error coming from the grids with a very coarse resolution in the magnetic moment direction. In the scenario represented by Figure 13 (left), the regular grid has a lot more grid points in the parallel velocity direction than the block-structured grid. The accuracy gap between these two grids is more evident when we consider the same coarse parallel velocity resolution corresponding to $\mathrm{nv0} = 24$, see Figure 13 (right). This ability of the block-structured grid to preserve accuracy even with very coarse velocity space resolutions is useful for the methods that combine the results from simulations on coarse grids to get an overall accurate result (details on such methods applied in GENE are explained in [30, 31, 32, 33]).

To get more insight into the performance improvement due to the block-structured grids, we compare the simulation time of the regular grid ($(\mathrm{nv0}, \mathrm{nw0}) = (268, 14)$) with that of the block-structured grid ($(\mathrm{nv0}, \mathrm{nw0}) = (158, 10)$). To obtain these measurements, we used the same compiler optimization flags as those

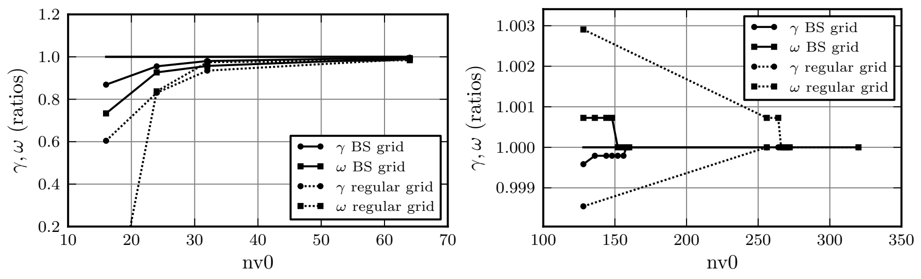

Figure 12: Convergence plots of γ and ω for the number of grid points in the parallel velocity direction nv0. Solid lines correspond to the block-structured grid and dashed lines to the regular grid.

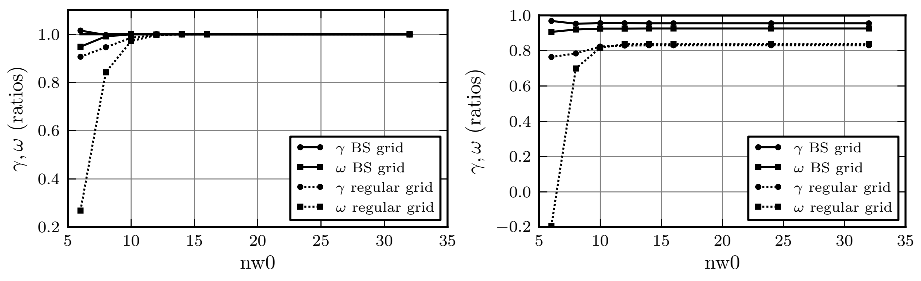

Figure 13: Convergence plots of γ and ω for the number of grid points in the magnetic moment direction nw0. Solid lines correspond to the block-structured grid and dashed lines to the regular grid. The left plot corresponds to simulations with nv0 = 158 for the block-structured grid and nv0 = 268 for the regular grid, the right plot corresponds to simulations with nv0 = 24 for the both types of grids.

applied in GENE for production runs. Hence, we ensure that our measurements are as close as possible to real-life situations. According to the convergence plots, the two grids yield results of the same accuracy. The initial value solver running on 32 CPUs converged in 19 429 s with the block-structured grid and in 39 346 s with the regular grid. This corresponds to a speedup of 2.03, which is slightly smaller than the ratio of the number of grid points (which is 2.36). The difference is caused by the computation and communication overhead on the block boundaries of the block-structured grid, as code optimizations for the newly introduced grids are still part of future work.

5.2. Nonlinear Simulations

The comparison of the nonlinear simulation results is more challenging than for the linear simulations. First, the nonlinear runs are very sensitive to the choice of initial conditions. Therefore, it does not make sense to compare results stemming from different grids at a fixed simulation time. Instead, we are interested in a quasi-stationary state and consider results that are time-averaged over a sufficiently large time interval. Second, there is a big number of observables that can be compared, for example, particle, heat, or momentum fluxes. These observables are computed by integrating in the five-dimensional phase space. As a result, a lot of information is lost and some import effects may remain unnoticed. Therefore, we compare not only the heat fluxes computed from the reference and block-structured grids, but also the fluctuating part of the distribution function. The simplest way to compare the distribution function is to plot its absolute value along a line that spans the entire radial range and goes through the region with the strongest fluctuations. In the case studies provided here, the best choice (due to the most prominent fluctuations) is the line at the middle of the parallel direction interval (outboard midplane position), at the fixed parallel velocity $v_\parallel = 0$ and the smallest magnetic moment μ_{min}, which is determined by the Gauss-Laguerre quadrature rule. For the physics scenario at hand, the toroidal mode number corresponding to the strongest fluctuations is typically $k_y = 0$; the turbulent transport is, however, driven by the dominant component $k_y \neq 0$. Hence, we compare the perturbed distribution function for both $k_y = 0$ and the dominant finite toroidal mode.

As a nonlinear case study, we simulate adiabatic electrons (one-species simulations) and use the TCV electron profiles showed in Figure 2 (top); a comprehensive description of the corresponding TCV discharge can be found in [19]. Furthermore, to reduce computational costs we use the same circular magnetic geometry as described in the linear results subsection. To obtain a spectrum with a dominant toroidal mode, we suppress the electron scale instabilities by adding hyper-diffusion in the x and y directions.

We convey four different numerical experiments to estimate the effect of blocking of the velocity space grid. The outlines of grids for these experiments are shown in Figure 14 and denoted by (`R, B, 1A, 2A`) correspondingly. First, we choose a very well-resolved regular grid as a reference, which has the following number of grid points $(\mathrm{nx0}, \mathrm{nky0}, \mathrm{nz0}, \mathrm{nv0}, \mathrm{nw0}) = (512, 16, 16, 82, 64)$; the outline of this grid is denoted by (`R`). Then, we set up a block-structured grid with five blocks and a coarsened velocity subgrid $(\mathrm{nv0}, \mathrm{nw0}) = (34, 16)$ with the outline marked by (`B`). Moreover, we also run simulations with two alternative regular grids, which have the same number of grid points in the velocity space as the block-structured grid. With the first alternative coarse regular grid, we check whether mere coarsening of the reference grid can yield acceptable results. This grid, therefore, has the same velocity space ranges (the grid outline denoted by (`1A`)) as the reference grid. The second alternative regular grid has the same ranges in $v_\parallel$ and μ directions as the upper block of the block-structured grid (the grid outline is given by (`2A`)). Consequently, it has fine velocity space resolutions — comparable to the resolutions of the reference grid — but narrow velocity grid ranges. This grid, thus, allows to check whether it is sufficient to reduce the velocity space ranges for the regular grid, but without changing the resolution to speed up the proposed simulation without a loss of accuracy.

For the presented four grids, we compare the heat fluxes averaged over a

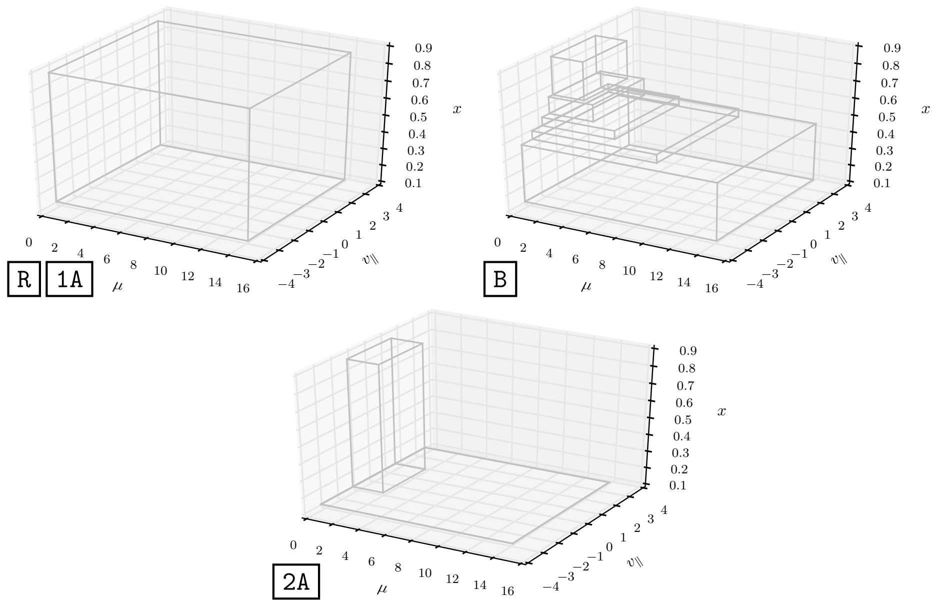

Figure 14: The outlines of grids in the $x - v_\parallel - \mu$, which are used for the comparison of nonlinear results. The top left plot (`R, 1A`) corresponds to the outline of the reference grid and the regular grid with coarse resolution in $v_\parallel$ and μ directions. The top right plot (`B`) corresponds to the block-structured grid and the bottom plot (`2A`) to the regular grid with short ranges in the velocity space directions and fine resolution.

time interval of 35 (R/c_s)[1] and provide the results in Table 1. From the table,

Table 1: Heat fluxes in gyro-Bohm units averaged over a time interval of 35 (R/c_s) stemming from four different grids.

	grid type			
	regular (`R`)	block-structured (`B`)	1st alternative (`1A`)	2nd alternative (`2A`)
Q_{gb}	1.92	1.83	1.64	0.78

we observe that it is the block-structured (`B`) grid that achieves the closest result to the reference grid (`R`). The second best result belongs to the first alternative grid (`1A`), which has a wide range and coarse velocity grid resolutions. The second alternative grid (`2A`) yields too small a heat flux.

The dominant mode of the described simulations corresponds to $k_y = 5.319$. The cross sections of the perturbed distribution function, which was time-averaged over the same interval (with a minimum of 45 samples at different

[1] R — major radius, c_s — ion sound speed

time steps) are shown in Figure 15. The top plot corresponds to the $k_y = 0$

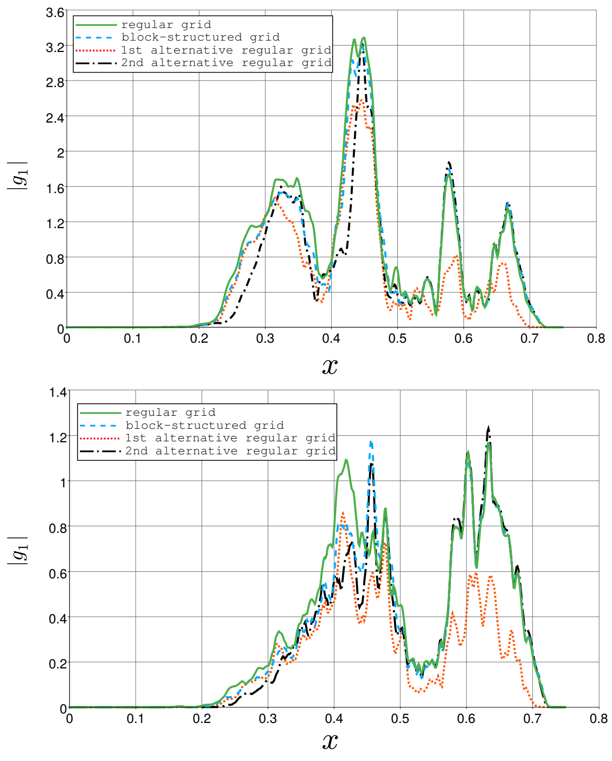

Figure 15: Plots over line of the perturbed distribution function for four different grids. The top figure corresponds to the toroidal mode number $k_y = 0$, the bottom to $k_y = 5.319$. The solid line corresponds to the reference regular grid (R), the dash-line to the block-structured grid (B), the dot-line to the first alternative regular grid (1A) (wide velocity grid ranges, coarse resolutions), and the dash-dot-line to the second alternative grid (2A) (narrow velocity grid ranges, fine resolutions).

toroidal mode, which yields the highest fluctuations. The bottom plot corresponds to the dominant mode with $k_y = 5.319$. From these plots, we observe that the first alternative grid (1A) (dash-line), despite the accurate heat flux result, does not sufficiently resolve fluctuations at the low temperature regions (radial distance $x \in [0.55, 075]$).

From the obtained results we draw the following conclusions: To start with, the first alternative regular (1A) grid with the wide velocity ranges and coarse resolutions yields an acceptable heat flux, but does not resolve the fluctuations in $x \in [0.55, 075]$. Furthermore, the second alternative grid (2A) with the narrow velocity ranges and fine resolutions yields an incorrect heat flux, but the plots over line are quite close to the reference ones. It is only the block-structured grid (B) (having the same number of grid points as the alternative regular grids) which succeeds to both resolve well the perturbed distribution function at all radial distances and give a correct value of the heat flux.

To estimate the speedup gained from the block-structured grid approach, we compare the average time of a time step for one simulation, for both the reference grid (`R`) and its block-structured counterpart (`B`). As for the performance comparison of the linear simulations, we used the compiler optimization flags applied in GENE for production runs. The simulations with the reference grid need 64.8 s and the simulations with the block-structured 6.2 s for one time step. These results were obtained from simulations running on 128 CPUs and using 100 time iterations to compute the averages. The corresponding speedup is 10.5, which is bigger than the ratio of the number of the grid points $(82 \cdot 64)/(34 \cdot 16) \approx 9.4$. The possible reason for the superlinear speedup is a cache effect, which outweighs the computational overhead due to the block boundaries in the simulations using the block-structured grid.

6. Conclusions

The study at hand explained the rationale behind the block-structured grid construction in full velocity space ($v_{\|}$ and μ) and detailed the implementation. The main focus of block-structuring the parallel velocity direction was the correct choice of grid ranges and resolution in each block along the radial direction. In the magnetic moment direction, only quadrature-like numerical operations, such as gyro-phase averaging, were performed. Hence, the main concern of the computational grid in this direction was an accurate integral computation with a minimum number of grid nodes.

We demonstrated the benefits of applying the block-structured grids in full velocity space to a grid-based gyrokinetic code. First of all, we showed that this type of grids allows reducing the number of grid points in the velocity space significantly without a loss of accuracy. The number of floating point operations in the gyrokinetic code is proportional to the number of grid points in the velocity space. As a consequence, reducing the number of grid points by a certain factor leads in the ideal case to a speedup equal to this factor. In the presented examples, the achieved speedups were close to the expected values: for the linear simulations, a 2.36 reduction in the number of grid points lead to a speedup of 2.03, while, in the case of nonlinear simulations, a reduction of 9.4 lead even to a superlinear speedup of 10.5. Moreover, we illustrated how block-structured grids reduce significantly the error of coarse velocity grids and perform reliably at a level where the regular grids are rendered useless. The block-structured grids help reduce the memory footprint, which is especially relevant to nonlinear simulation, as these simulations require usually considerably more grid points than linear simulations do. Furthermore, during the nonlinear simulations, time-averaged values characteristic for the quasi-stationary states are of interest, requiring diagnostic data to be saved at different time steps. The block-structured grids permit reducing the amount of this output tremendously.

Our proposed technique was based on a general approach that did not require the modification of the governing set of equations. This means that porting the code is straightforward in most cases, which is of crucial importance for codes such as GENE, whose regular grid version has been continuously enhanced over the last decade. The block-structured grid is not an alternative to parallelization, but an additional improvement. Furthermore, because the technique was shown to be minimally invasive, the block-structured grid inherits the good parallelization characteristics of the default GENE grids. The additional communication overhead stemming from the side exchanges can be minimized further, since much less data is sent than for the exchanges originating from the regular grid.

In the future, we plan to include collisions between different species, which are simulated with different velocity space block-structured grids. To capitalize further on the advantages of the developed technique, we plan to adjust the grid resolutions in the radial and binormal directions to achieve an extra reduction in the number of grid points.

Acknowledgments

The support of HEPP (International Helmholtz Graduate School for Plasma Physics) is gratefully acknowledged.

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Program (FP7/2007-2013)/ERC Grant Agreement No. 277870.

AppendixA. Gyro-Phase averaging

In GENE, along with a simple quadrature in μ direction, the gyro-phase averaging is computed. In the following, we define the gyro-averaging operations and explain how they can be performed in the case of block-structured grids. There are two types of gyro-averaging that appear in the gyrokinetic set of equations, see [12, 19]. The gyro-phase averaging in the Vlasov equation acts on the fields and is given by

$$\bar{\phi}(\boldsymbol{X}) = \frac{1}{2\pi} \int_0^{2\pi} \phi(\mathbf{X} + \boldsymbol{\rho}(\boldsymbol{X}))\mathrm{d}\theta \tag{A.1}$$

where $\boldsymbol{X} = (X, Y, Z)$ is the gyro-center position, $\boldsymbol{\rho}$ the gyro-radius, and θ the gyro-phase angle.

Another type of gyro-averaging that appears in the field equations and acts on the distribution function and on $\bar{\phi}(\boldsymbol{X})$ is defined by

$$\langle f \rangle (\boldsymbol{x}) = \frac{1}{2\pi} \int \delta(\boldsymbol{X} + \boldsymbol{\rho}(\boldsymbol{X}) - \boldsymbol{x}) f(\boldsymbol{X})\mathrm{d}^3 X \mathrm{d}\theta \tag{A.2}$$

As shown in [12, 19], the discretized versions of both types of the gyro-phase averaging are related. Therefore, we restrict the discussion to the discretization of (A.1) in the following.

The binormal direction Y in GENE is discretized in the Fourier space. Therefore, the potential ϕ is computed by

$$\phi(\boldsymbol{X}) = \frac{1}{2\pi} \sum_{k_y} \phi(X, k_y, Z) e^{\mathrm{i}k_y Y} \tag{A.3}$$

From this follows:

$$\bar{\phi}(\boldsymbol{X}) = \frac{1}{2\pi} \sum_{k_y} \int_0^{2\pi} \phi(X + \rho_x(\boldsymbol{X}), k_y, Z)) e^{\mathrm{i}k_y(Y+\rho_y(\boldsymbol{X}))} \mathrm{d}\theta \tag{A.4}$$

where $\rho_z = 0$ due to the field-aligned position space coordinate system.

Function ϕ is known only at the computational grid points. To be able to carry on the integration in equation (A.4), we approximate this function by using finite-element like base functions with local support. Hence, ϕ is expressed as a linear combination of the base functions:

$$\phi(X, k_y, Z) = \sum_i \phi(X_i, k_y, Z) \Lambda_i(X) = \boldsymbol{\Lambda} \cdot \boldsymbol{\phi} \tag{A.5}$$

where k_y and Z are fixed to some particular values. After inserting (A.5) into (A.4), we obtain the gyro-averaged function at the gyro-center position $\boldsymbol{X}_k = (X_k, Y, Z)$:

$$\begin{aligned} \bar{\phi}(\boldsymbol{X}_k) &= \frac{1}{2\pi} \sum_{i,k_y} \int_0^{2\pi} \phi(X_i, k_y, Z) \Lambda_i(X_k + \rho_x(\boldsymbol{X}_k)) e^{\mathrm{i}k_y(Y+\rho_y(\boldsymbol{X}_k))} \mathrm{d}\theta \\ &= \sum_{i,k_y} e^{\mathrm{i}k_y Y} \mathcal{G}_{k,i} \phi_i = \sum_{k_y} e^{\mathrm{i}k_y Y} \mathcal{G} \cdot \boldsymbol{\phi} \end{aligned} \tag{A.6}$$

where matrix $\mathcal{G}$ is called the gyro-averaging matrix and is defined by

$$\mathcal{G}_{ki}(k_y, z, \mu) = \frac{1}{2\pi} \int_0^{2\pi} \Lambda_i(X_k + \rho_x(\boldsymbol{X}_k)) e^{\mathrm{i}k_y \rho_y(\boldsymbol{X}_k)} \mathrm{d}\theta \tag{A.7}$$

Gyro-radius projections on x and y axis are given by

$$\begin{aligned} \rho_x(\boldsymbol{X}_k) &= \rho(\boldsymbol{X}_k) \sqrt{g^{xx}(\boldsymbol{X}_k)} \cos\theta \\ \rho_y(\boldsymbol{X}_k) &= \frac{\rho(\boldsymbol{X}_k)}{\sqrt{g^{xx}(\boldsymbol{X}_k)}} \left(g^{xy}(\boldsymbol{X}_k) \cos\theta - \sqrt{\gamma_1} \sin\theta\right) \end{aligned} \tag{A.8}$$

According to [12], the choice of base functions relies on an expansion with polynomials of odd degree p:

$$\phi(x) = \sum_{n=1}^{\mathrm{nx0}} \sum_{m=0}^{(p-1)/2} \left. \frac{\partial^m \phi(x)}{\partial x^m} \right|_{x=x_n} P_{m,n}(x) \tag{A.9}$$

where nx0 is the number of grid points in the radial direction. The local support polynomials are defined by

$$\left.\frac{\partial^u P_{m,n}}{\partial x^u}\right|_{x=x_j} = \delta_{jn}\delta_{um}, \quad j = n, n+1, \quad u = 0, \ldots, \frac{p-1}{2} \tag{A.10}$$

We do not know the exact derivatives of the discretized function ϕ. Therefore, we use a finite differences scheme to compute the derivatives. Stencils for derivative computations correspond to those used to discretize the governing equations. If we introduce a vector of polynomials

$$\boldsymbol{P}_m = (P_{m,1}, \ldots, P_{m,\mathrm{nx0}}) \tag{A.11}$$

then the polynomial expansion is

$$\phi(x) = \sum_{m=0}^{(p-1)/2} \boldsymbol{P}_m \mathcal{D}^m \boldsymbol{\phi} \tag{A.12}$$

where $\mathcal{D}^m$ is the m-order derivative band matrix (rows correspond to stencil) and $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_{\mathrm{nx0}})^\intercal$ is a vector with discretized function values.

By comparing expressions (A.5) and (A.12), we conclude that the base functions are

$$\boldsymbol{\Lambda} = \sum_{m=0}^{(p-1)/2} \boldsymbol{P}_m \mathcal{D}^m \tag{A.13}$$

Using the base function expression (A.13) and (A.7), we write down the row of the gyro-averaging matrix:

$$\boldsymbol{\mathcal{G}}_{k*}(k_y, z, \mu) = \frac{1}{2\pi} \sum_{m=0}^{(p-1)/2} \left(\int_0^{2\pi} \boldsymbol{P}_m(X_k + \rho_x(\boldsymbol{X}_k)) e^{\mathrm{i}k_y\rho_y(\boldsymbol{X}_k)} \mathrm{d}\theta \right) \mathcal{D}^m \tag{A.14}$$

Here, we introduce the θ-integral-matrix notation $\mathcal{Q}^m$, with elements computed by

$$\mathcal{Q}^m_{kn} = \frac{1}{2\pi} \int_0^{2\pi} P_{m,n}(X_k + \rho_x(\boldsymbol{X}_k)) e^{\mathrm{i}k_y\rho_y(\boldsymbol{X}_k)} \mathrm{d}\theta \tag{A.15}$$

Then the gyro-matrix can be written simply as

$$\mathcal{G}(k_y, z, \mu) = \sum_{m=0}^{(p-1)/2} \mathcal{Q}^m \mathcal{D}^m \tag{A.16}$$

Due to the band structure of the $\mathcal{Q}$ and $\mathcal{D}$ matrices, $\mathcal{G}$ is also a band matrix.

As shown in [19], the gyro-phase averaging defined by (A.2) can be approximated by $\mathcal{G}^\dagger$.

The physical interpretation of the (A.1) gyro-phase averaging is an integral over a particle trajectory projection in the $x - y$ subspace with a gyro-radius

$\rho = (c/q)\sqrt{2\mu/B}$. In Figure A.16 (left), we show a set of such gyro-trajectories for a given discretization of the magnetic moment μ in the default regular grid. In the case of block-structured grids, see Figure A.16 (right), we cannot compute the (A.1) gyro-phase averaging in the same way as for the regular grid due to different μ nodes at different radial positions. However, this type of gyro-

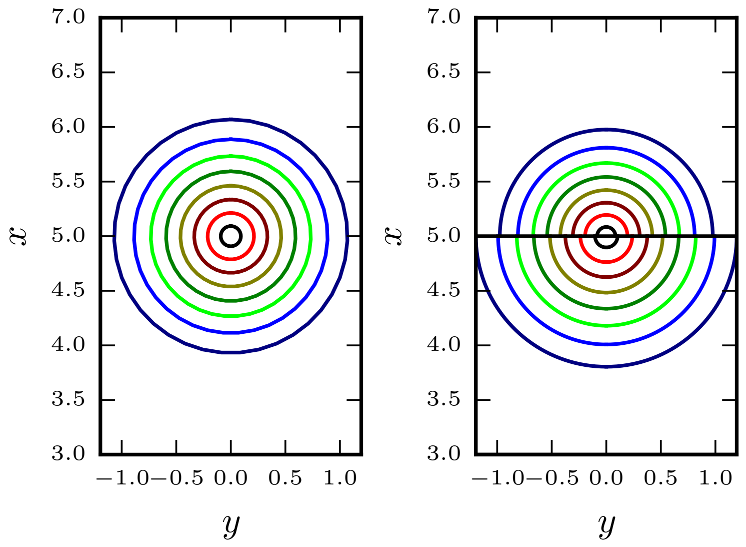

Figure A.16: Examples of gyro-trajectories for a particular set of μ. Left plot — regular rectangular grid, right plot — two-block grid with the block boundary at $x = 5$.

averaging is performed only for the fields, which are functions of the position space coordinates and do not depend on μ. Therefore, one can compute gyro-averaged fields at any magnetic moment coordinate, irrespective of the μ nodes chosen for the discretization of the distribution function.

The second type of gyro-averaging (A.2) is applied on the distribution function. For this reason, we cannot ignore the discretization of the magnetic moment coordinate anymore. Nevertheless, we can still avoid the integration on the closed gyro-trajectories and interpolating in μ direction by combining gyro-phase averaging with μ quadrature. These two operations always appear together in gyro-kinetic simulations. In the case of the regular grid, the combined operation can be written in the schematic way:

$$\int \langle f \rangle (\boldsymbol{x})\, \mathrm{d}\mu = \sum_{m=1}^{\mathrm{nw0}} w_m \mathcal{G}^{\dagger}(\mu_m) \boldsymbol{f}(\mu_m) \tag{A.17}$$

where $\boldsymbol{f} = (f_1, \ldots, f_{\mathrm{nx0}})^{\intercal}$. This operation is valid as long as all elements of $\boldsymbol{f}$ are computed at the same μ_m. It does not hold for the block-structured grids. When we fix the m index, we obtain the following distribution function vector for the case of a two-block grid:

$$\boldsymbol{f}_m = \left(\begin{array}{ccc|ccc} f_1^{(1)} & \cdots & f_k^{(1)} & f_{k+1}^{(2)} & \cdots & f_{\mathrm{nx0}}^{(2)} \end{array} \right)^{\mathsf{T}} \tag{A.18}$$

where the superscript index refers to the corresponding block. For the two-block grid, the operation (A.17) is given by

$$\int \langle f \rangle (\boldsymbol{x}) \, \mathrm{d}\mu =$$

$$\sum_{m=1}^{\mathrm{nw0}} \left(\begin{array}{c|c} w_m^{(1)} \cdot \mathcal{G}^{\dagger(1)}_{*,1\ldots k}(\mu_m^{(1)}) & w_m^{(2)} \cdot \mathcal{G}^{\dagger(2)}_{*,k+1\ldots\mathrm{nx0}}(\mu_m^{(2)}) \end{array} \right) \cdot \left(\begin{array}{c} f_1^{(1)} \\ \vdots \\ f_k^{(1)} \\ \hline f_{k+1}^{(2)} \\ \vdots \\ f_{\mathrm{nx0}}^{(2)} \end{array} \right) \tag{A.19}$$

In the last expression, we denote by $\mathcal{G}^{\dagger(b)}_{*,l\ldots k}(\mu^{(b)})$ a submatrix corresponding to $\mathcal{G}^\dagger$ columns, with indices from l to k. This submatrix is computed with regards to the b-block at $\mu^{(b)}$.

To summarize, in the case of the block-structured grid, we compute for each block columns with corresponding μ nodes and apply (A.19) to compute gyro-phase averaging in combination with quadrature for the magnetic moment coordinate.

To compute the gyro-phase averaging of the fields defined by (A.1) with a correct set of magnetic moments at radial distances corresponding to a certain block, we use the following expression

$$\bar{\phi}(\boldsymbol{X}, \mu_m) = \left(\begin{array}{c} \mathcal{G}^{(1)}_{1\ldots k,*}(\mu_m^{(1)}) \\ \hline \mathcal{G}^{(2)}_{k+1\ldots\mathrm{nx0},*}(\mu_m^{(2)}) \end{array} \right) \cdot \boldsymbol{\phi} \tag{A.20}$$

Here $\mathcal{G}^{(b)}_{l\ldots k,*}(\mu^{(b)})$ is a submatrix corresponding to $\mathcal{G}$ rows, with indices from l to k. Due to the relation $\mathcal{G}^{(b)}_{l\ldots k,*}(\mu^{(b)}) = \mathcal{G}^{\mathsf{T}(b)}_{*,l\ldots k}(\mu^{(b)})$, we do not have to compute separately the $\mathcal{G}$ and $\mathcal{G}^\dagger$ matrices for the block-structured grids. Like for the regular grid, $\mathcal{G}^\dagger$ is the complex conjugate transpose of $\mathcal{G}$.

[1] O. Larroche, An efficient explicit numerical scheme for diffusion-type equations with a highly inhomogeneous and highly anisotropic diffusion tensor, Journal of Computational Physics 223 (1) (2007) 436 – 450.

[2] H. Doerk, Gyrokinetic simulation of microtearing turbulence, Ph.D. thesis, Fakultät für Naturwissenschaften der Universität Ulm (2012).

[3] B. Peigney, O. Larroche, V. Tikhonchuk, Fokker-Planck kinetic modeling of suprathermal α-particles in a fusion plasma, Journal of Computational Physics 278 (2014) 416 – 444.

[4] W. Taitano, L. Chacón, A. Simakov, An adaptive, conservative 0D-2V multispecies Rosenbluth-Fokker-Planck solver for arbitrarily disparate mass and temperature regimes, Journal of Computational Physics (2016) –.

[5] D. Jarema, H. J. Bungartz, T. Görler, F. Jenko, T. Neckel, D. Told, Block-structured grids for Eulerian gyrokinetic simulations, Computer Physics Communications 198 (2016) 105 – 117.

[6] F. Jenko, W. Dorland, M. Kotschenreuther, B. Rogers, Electron temperature gradient driven turbulence, Physics of Plasmas 7 (5) (2000) 1904–1910.

[7] T. Görler, X. Lapillonne, T. Brunner, T. Dannert, F. Jenko, F. Merz, D. Told, The global version of the gyrokinetic turbulence code GENE, Journal of Computational Physics 230 (18) (2011) 7053–7071.

[8] Gyrokinetic Electromagnetic Numerical Experiment, `http://genecode.org`, accessed: 2015-03-03.

[9] J. W. Connor, R. J. Hastie, J. B. Taylor, High mode number stability of an axisymmetric toroidal plasma, Proceedings of the Royal Society A 365 (1720) (1978) 1–17.

[10] P. Xanthopoulos, F. Jenko, Clebsch-type coordinates for nonlinear gyrokinetics in generic toroidal configurations, Physics of Plasmas 13 (092301) (2006) 1–10.

[11] F. Merz, Gyrokinetic simulations of multimode plasma turbulence, Ph.D. thesis, Mathematisch-Naturwissenschaftliche Fakultät der Westfälischen Wilhelms-Universität Münster (2008).

[12] T. Görler, Multiscale effects in plasma microturbulence, Ph.D. thesis, Fakultät für Naturwissenschaften der Universität Ulm (2009).

[13] X. Lapillonne, Local and global Eulerian gyrokinetic simulations of microturbulence in realistic geometry with applications to the TCV tokamak, Ph.D. thesis, École Polytechnique Fédérale de Lausanne (2010).

[14] J. Candy, C. Holland, R. E. Waltz, M. R. Fahey, B. E., Tokamak profile prediction using direct gyrokinetic and neoclassical simulation, Physics of Plasmas 16 (060704) (2009) 1–4.

[15] M. Barnes, I. G. Abel, W. Dorland, T. Görler, G. W. Hammett, F. Jenko, Direct multiscale coupling of a transport code to gyrokinetic turbulence codes, Physics of Plasmas 17 (056109) (2010) 1–11.

[16] A. J. Brizard, T. S. Hahm, Foundations of nonlinear gyrokinetic theory, Reviews of Modern Physics 79 (421) (2007) 421–468.

[17] J. Cary, R. Littlejohn, Noncanonical Hamiltonian mechanics and its application to magnetic field line flow, Annals of Physics 151 (1983) 1–34.

[18] R. G. Littlejohn, Hamiltonian formulation of guiding center motion, Physics of Fluids 24 (9) (1981) 1730–1749.

[19] D. Told, Gyrokinetic microturbulence in transport barriers, Ph.D. thesis, Fakultät für Naturwissenschaften der Universität Ulm (2012).

[20] R. E. Denton, M. Kotschenreuther, δf Algorithm, Journal of Computational Physics 119 (2) (1995) 283–294.

[21] Y. Chen, S. E. Parker, A δf particle method for gyrokinetic simulations with kinetic electrons and electromagnetic perturbations, Journal of Computational Physics 189 (2) (2003) 463–475.

[22] J. Candy, R. E. Waltz, S. E. Parker, Y. Chen, Relevance of the parallel nonlinearity in gyrokinetic simulations of tokamak plasmas, Physics of Plasmas 13 (074501) (2006) 1–11.

[23] P. Xanthopoulos, D. Mikkelsen, F. Jenko, W. Dorland, O. Kelentev, Verification and application of numerically generated magnetic coordinate systems in gyrokinetics, Physics of Plasmas 15 (122108) (2008) 1–8.

[24] W. D'haeseleer, W. Hitchon, J. Callen, J. Shohet, Flux Coordinates and Magnetic Field Structure: A Guide to a Fundamental Tool of Plasma Theory, Springer Verlag, New York, 1991.

[25] W. Gautschi, Algorithm 726: Orthpol–a package of routines for generating orthogonal polynomials and gauss-type quadrature rules, ACM Trans. Math. Softw. 20 (1) (1994) 21–62.

[26] W. Gander, J. Hřebíček, Solving problems in scientific computing using Maple and MATLAB, 3rd Edition, Springer, 1997.

[27] P. Durbin, G. Iaccarino, An approach to local refinement of structured grids, Journal of Computational Physics 181 (2) (2002) 639 – 653.

[28] M. Kotschenreuther, G. Rewoldt, W. M. Tang, Comparison of initial value and eigenvalue codes for kinetic toroidal plasma instabilities, Computer Physics Communications 88 (2–3) (1995) 128–140.

[29] F. Merz, C. Kowitz, E. Romero, J. Roman, F. Jenko, Multi-dimensional gyrokinetic parameter studies based on eigenvalue computations, Computer Physics Communications 183 (4) (2012) 922 – 930.

[30] C. Kowitz, M. Hegland, The sparse grid combination technique for computing eigenvalues in linear gyrokinetics, in: International Conference on Computational Science 2013, Procedia Computer Science, Elsevier, Amsterdam, 2013, pp. 449–458.

[31] D. Pflüger, H.-J. Bungartz, M. Griebel, F. Jenko, T. Dannert, M. Heene, A. Parra Hinojosa, C. Kowitz, P. Zaspel, Exahd: An exa-scalable two-level sparse grid approach for higher-dimensional problems in plasma physics and beyond, in: Euro-Par 2014: Parallel Processing Workshops, Lecture Notes in Computer Science, Springer-Verlag, 2014.

[32] A. Parra Hinojosa, C. Kowitz, M. Heene, D. Pflüger, H.-J. Bungartz, Towards a fault-tolerant, scalable implementation of gene, in: Proceedings of ICCE 2014, Lecture Notes in Computational Science and Engineering, Springer-Verlag, 2015, accepted.

[33] C. Kowitz, Applying the sparse grid combination technique in linear gyrokinetics, Ph.D. thesis, Technische Universitt München (2016).