A fast solver for the gyrokinetic field equation with adiabatic electrons

M.Borchardt, R.Kleiber
Max-Planck-Institut für Plasmaphysik, EURATOM Association
17491 Greifswald, Germany

W.Hackbusch
Max-Planck-Institut für Mathematik in den Naturwissenschaften,
04103 Leipzig, Germany

Describing turbulence and microinstabilities in fusion devices is often modelled with the gyrokinetic equation. During the time evolution of the distribution function a field equation for the electrostatic potential needs to be solved. In the case of adiabatic electrons it contains a flux-surface-average term resulting in an integro-differential equation. Its numerical solution is time and memory intensive for three-dimensional configurations. Here a new algorithm is presented which only requires the numerical inversion of a simpler differential operator and a subsequent addition of a correction term. This new procedure is as fast as solving the equation without the surface average.

I. INTRODUCTION

Understanding microinstabilities and turbulence in toroidal geometries is of crucial importance for present day fusion devices. For their physical modelling the gyrokinetic equation is a widespread first principles theory [1]. Gyrokinetic simulations are often carried out using particle-in-cell (PIC) methods because they are relatively easy to parallelise. In such PIC methods the trajectories of the particles (e.g. ions and electrons) is described by the equations of motion in a five-dimensional phase space. At each time step a density is calculated from the particle positions by projection onto a space grid. The electrostatic potential, which follows from this density by solving a field equation, then affects the trajectories of the particles again.
A common simplification is to only simulate the ions and use so-called adiabatic electrons. In this case the field equation consists of a Helmholtz operator and an additional term representing the flux-surface average of the potential. In various cases (e.g. linear simulations of high mode number perturbations) this averaging part can be neglected. For nonlinear simulations it becomes crucial since it determines the behaviour of the zonal flow, which strongly influences the transport level.
Discretisation of the field equation leads to a system of linear equations. While the matrix resulting from the Helmholtz part, a differential operator, is sparse and can be stored easily, the flux surface averaging part needs attention since it is a non-local integral operation. For axisymmetric domains (like tokamaks) the average term can be simplified considerably by using a Fourier ansatz [2] or an approach where the field equation is replaced by two decoupled equations for the potential and its flux surface average, respectively [3]. In non-axisymmetric cases (stellarators) this does not give any advantage. Discretisation of the equation leads to a non-sparse matrix, which for realistic grid sizes is too large to be stored in computer memory. In spite of being sparse the Helmholtz matrix alone is still so large that a direct solution (like LU-decomposition) is not practical. Therefore the solution needs to be calculated employing an iterative Krylov subspace method. In this paper one possible way to overcome the problems connected with the flux surface averaging term for non-axisymmetric domains is presented for the EUTERPE code [4].
EUTERPE is a global (full radius, full flux surface) gyrokinetic δf [5] PIC code especially designed to treat three-dimensional geometry. It uses a B-spline discretisation for the charge assignment and the field equation. The equations of motion are integrated by a fourth order Runge–Kutta method. Thus the field equation, where the left hand side stays constant but the density is changing with time, needs to be solved four times during each time step and its fast solution is mandatory.
For reasonable grid sizes the whole procedure rapidly becomes time and memory consuming. Therefore two different parallelisation concepts are implemented in EUTERPE: Firstly, domain decomposition in the toroidal direction where at least one poloidal cut is stored on a single core, and secondly, domain cloning where all particles are partitioned to multiple copies of the whole domain. This cloning mechanism needs communication only for updating the density during each Runge–Kutta step while all other parts of the algorithm work without communication between the clones.

II. THEORY

A generalised toroidal coordinate system is used in this derivation, which is the same as in the EUTERPE code. There are one radial component $s \in [0, 1]$ and two angle-like components $\vartheta, \varphi \in [0, 2\pi]$. These are widely used coordinates

to represent toroidal magnetic domains [6]. The metric tensor is given by $(g^{xy})_{x,y\in\{s,\vartheta,\varphi\}}$ with a Jacobian $\sqrt{g}$ and volume element $\mathrm{d}V = \sqrt{g}\,\mathrm{d}s\,\mathrm{d}\vartheta\,\mathrm{d}\varphi$.
The field equation for the electrostatic potential $\phi(s,\vartheta,\varphi)$ is given by

$$-\nabla\cdot\Big(\rho^2\,\nabla_\perp\phi\Big) + \phi - \langle\phi\rangle = n. \tag{1}$$

This equation is solved with a Dirichlet boundary condition for the potential on the outer boundary ($s = 1$). The variable n denotes the density distribution for the whole domain and $\rho(s,\vartheta,\varphi)$ is a non-zero spatial variable of the order of 10^{-2} for a typical toroidal device. The perpendicular Laplace operator defined in the (s,ϑ)-plane can be computed by

$$\nabla\cdot\rho^2\,\nabla_\perp = \sum_{x,y\in\{s,\vartheta\}} \frac{1}{\sqrt{g}}\frac{\partial}{\partial x}(\rho^2\sqrt{g}g^{xy}\frac{\partial}{\partial y}). \tag{2}$$

The averaging operator $\langle\cdot\rangle$ in equation (1) is the so-called flux average on an $s = \text{const}$ surface

$$\langle\phi\rangle(s) = \frac{1}{P(s)}\int\limits_{s=const}\phi\,\mathrm{d}\Gamma \quad \text{with} \quad P(s) = \int\limits_{s=const}\mathrm{d}\Gamma,$$

where $\mathrm{d}\Gamma = \sqrt{g}\,\mathrm{d}\vartheta\,\mathrm{d}\varphi$, i.e. actually a volume average over the region between two neighbouring s-surfaces.
In EUTERPE the potential is represented by a B-spline discretisation $\phi = \sum\limits_{\nu\in\mathcal{I}}\phi_\nu\Lambda_\nu$. Here $\nu = (i,j,k)$ is a multi index and $\Lambda_\nu = \Lambda_i(s)\Lambda_j(\vartheta)\Lambda_k(\varphi)$ is a tensor product of three one-dimensional B-spline functions [7] with the order o. The size of the underlying regular mesh is $N_s \times N_\vartheta \times N_\varphi$. The index set $\mathcal{I}$ is given by $\{1,2,\ldots,N_s\}\times\{1,2,\ldots,N_\vartheta\}\times\{1,2,\ldots,N_\varphi\}$.
With the use of equation (2), introducing the spline approximation, multiplying by $\Lambda_{\nu'}$ and integrating over the whole domain transforms the field equation (1) into a system of equations

$$\sum_{\nu\in\mathcal{I}}\phi_\nu\int\left[\sum_{x,y\in\{s,\vartheta\}}\rho^2 g^{xy}\frac{\partial\Lambda_\nu}{\partial x}\frac{\partial\Lambda_{\nu'}}{\partial y} + \Lambda_\nu\Lambda_{\nu'}\right]\mathrm{d}V - \int\Lambda_{\nu'}\langle\phi\rangle\,\mathrm{d}V = \int\Lambda_{\nu'}n\,\mathrm{d}V = b_{\nu'}.$$

This equation can be written in matrix form,

$$(H - M)\,\phi = b, \tag{3}$$

where H is the Helmholtz matrix with its elements

$$H_{\nu'\nu} = \iiint\left[\sum_{x,y\in\{s,\vartheta\}}\rho^2 g^{xy}\frac{\partial\Lambda_\nu}{\partial x}\frac{\partial\Lambda_{\nu'}}{\partial y} + \Lambda_\nu\Lambda_{\nu'}\right]\sqrt{g}\,\mathrm{d}s\,\mathrm{d}\vartheta\,\mathrm{d}\varphi$$

and M is the averaging operator with the following matrix representation

$$M_{\nu'\nu} = \iiint\Lambda_{\nu'}\langle\Lambda_\nu\rangle\sqrt{g}\,\mathrm{d}s\,\mathrm{d}\vartheta\,\mathrm{d}\varphi. \tag{4}$$

To solve the system of equations (3) one has to compute the matrix $(H - M)$. The number of non-zeros of the Helmholtz matrix H depends on the size of the overlap of two B-spline functions. The term $\Lambda_\nu\Lambda_{\nu'}$ creates multiple bands in the matrix. The overall width of the bands is about $(2o{+}1)^3$. Therefore H, which has about $(2o{+}1)^3 N_s N_\vartheta N_\varphi$ non-zero elements, can be stored easily because the B-spline order o is small (typical two). If one calculates the matrix elements of the averaging operator (4), one gets

$$\begin{aligned} M_{\nu'\nu} &= \iiint\Lambda_{\nu'}\left(\frac{1}{P(s)}\iint\Lambda_\nu\sqrt{g}\,\mathrm{d}\vartheta\,\mathrm{d}\varphi\right)\sqrt{g}\,\mathrm{d}s\,\mathrm{d}\vartheta\,\mathrm{d}\varphi \\ &= \int P^{-1}(s)\Lambda_i(s)G_{jk}(s)\Lambda_{i'}(s)G_{j'k'}(s)\,\mathrm{d}s \end{aligned} \tag{5}$$

with

$$G_{jk}(s) = \iint\Lambda_j(\vartheta)\Lambda_k(\varphi)\sqrt{g}\,\mathrm{d}\vartheta\,\mathrm{d}\varphi.$$

The matrix M has $(2o+1)N_s(N_\vartheta N_\varphi)^2$ non-zeros. Since this number already becomes very large for small cases, M can normally not be stored.
The first ansatz to solve the discretised field equation (3) was to use a standard linear iterative solver and provide a function which calculates the needed matrix-by-vector product $(H-M)\,v$. This was done with a matrix-free representation of M [8]. It works well, but the number of iterations for the iterative solver increases strongly compared to the simplified problem without the average operator, making this method relatively slow.
The new ansatz approximates the integral in (5) in the same manner as other integrals in the EUTERPE code, i.e. Gaussian integration. The matrix elements $M_{\nu'\nu}$ are computed with

$$\begin{aligned} M_{\nu'\nu} &\approx \sum_{r=1}^{n_s} \gamma(s_r) P^{-1}(s_r) \Lambda_i(s_r) G_{jk}(s_r) \Lambda_{i'}(s_r) G_{j'k'}(s_r) \\ &= \sum_{r=1}^{n_s} \alpha_{ijk}(s_r) \gamma(s_r) P^{-1}(s_r) \alpha_{i'j'k'}(s_r) \end{aligned}$$

where $\alpha_{ijk}(s_r) = \Lambda_i(s_r) G_{jk}(s_r)$ and $\gamma(s_r)$ are the weight factors for the approximation of the integral at the n_s (i.e. $N_s \times$number of Gauss points) radial integration points s_r. With this simplification one can write the matrix representation of the averaging operator as a matrix product

$$M = -ADA^T$$

with the following matrix elements

$$\begin{aligned} A &= \big(\alpha_\nu(s_r)\big)_{\nu\in\mathcal{I}, r=1,\dots,n_s} \\ D &= -\operatorname*{diag}_{r=1,\dots,n_s} \big(\gamma(s_r) P^{-1}(s_r)\big). \end{aligned}$$

Now everything is given to compute the matrix inverse of $(H-M)$. With the help of the Sherman–Morrison–Woodbury formula [9]

$$(W + UV^T)^{-1} = W^{-1} - W^{-1}U(I + V^T W^{-1} U)^{-1} V^T W^{-1}$$

the inverse of $(H-M)$ is given by

$$\begin{aligned} (H-M)^{-1} &= (H + ADA^T)^{-1} \\ &= H^{-1} - H^{-1}A(D^{-1} + A^T H^{-1} A)^{-1} A^T H^{-1}. \end{aligned} \tag{6}$$

This means that one only needs to provide a solver for equation (3) without the average part M, and then the solution to the full problem can be computed easily. The inverse of the dense matrix $S = (D^{-1} + A^T H^{-1} A)$ can be calculated directly because its size $n_s \times n_s$ depends only on the number of integration points in radial direction, which is $O(100)$. None of the terms in (6) change their value during a single run, and the matrix can thus calculated in an initialisation step. Therefore, one has to calculate and store some supplementary matrices $(\tilde{A}, S^{-1}, A^T)$ in addition to the Helmholtz matrix H before the actual run:

- solve the systems of equations (taking into account the Dirichlet boundary condition)

$$\sum_{\nu'\in\mathcal{I}} H_{\nu\nu'} \tilde{\alpha}_{\nu'}(s_r) = \alpha_\nu(s_r) \tag{7}$$

  for all Gaussian integrations points s_r and construct the matrix

$$\tilde{A} = H^{-1}A = \big(\tilde{\alpha}_{\nu'}(s_r)\big)_{\nu'\in\mathcal{I}, r=1,\dots,n_s}$$

- compute $S = (D^{-1} + A^T \tilde{A})$ and store its inverse.

With these precalculated matrices solving the field equation (1) during a normal time step is simplified to:

1. solving the discretised field equation without the average operator

$$H\tilde{\phi} = b \tag{8}$$

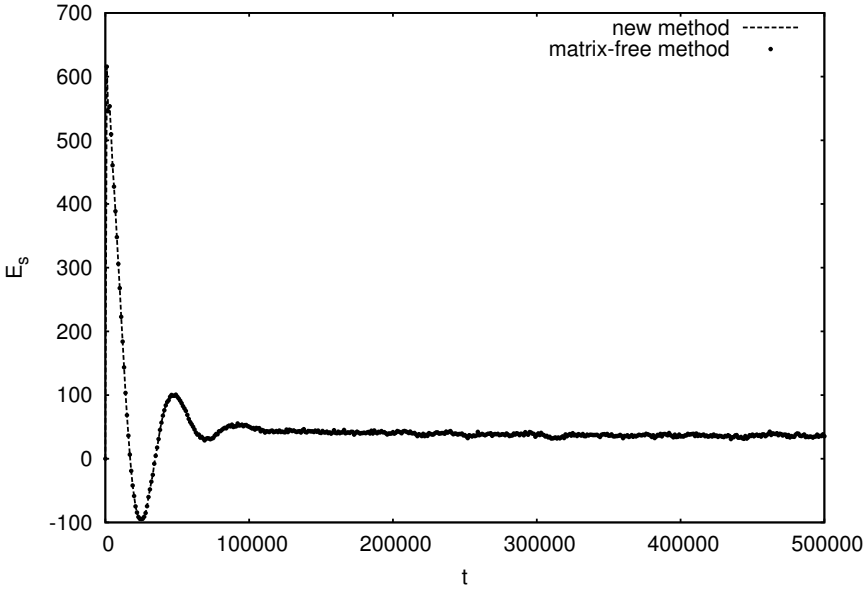

FIG. 1: Rosenbluth-Hinton test: time evolution of the normalised radial electric field at radial position $s = 0.5$ for Wendelstein 7-X, comparison between old matrix-free and new method.

2. computing the correction to the solution to get the result for the complete field equation

$$\phi = \tilde{\phi} - \tilde{A}\ S^{-1} A^T \tilde{\phi}.$$

Thus, in contrast to the old method, the number of iterations for the solution of (3) stays the same with or without the averaging operator M. The only overhead to the problem without the averaging is the triple matrix-by-vector product. Contrary to the older method one has to store three extra matrices $\tilde{A}$, S^{-1} and A^T. The number of non-zero elements of them can easily be calculated. $\tilde{A}$ is a matrix with maximal $n_s N_s N_\vartheta N_\varphi$ entries (depending on the solutions of (7)), S^{-1} is a small $n_s \times n_s$ matrix and A^T has about $n_s(2o+1)N_\vartheta N_\varphi$ non-zeros. This extra storage is required in the new method, whereas the matrix-free solution does not need any extra memory.

III. RESULTS

All test cases from this section were done on the HPC-FF machine at the Jülich Supercomputing Centre. It is a parallel system with a total of 8640 cores (clock rate of 2.93 GHz, 8 cores per computing node). Every node has altogether 24 gigabytes memory available. The realisation in the EUTERPE code is based on the PETSc framework [10] and uses a conjugate gradient method with block Jacobi preconditioner (see e.g. [11]), where on every block an ILU(0) is performed. The block size depends on the number of domains.

To check the correctness and the run time behaviour of the new solver, several different applications of the code were tried. Here we present only results for a standard Wendelstein 7-X configuration [12] with $\beta = 4.8\%$. An important problem is the Rosenbluth–Hinton test [13], where the time evolution of an initial electrostatic field is calculated. The average part of the field equation (1) is substantial: Only if this term is present does one obtain a non-zero asymptote at late time for the radial electric field, otherwise the radial electric field will go down to zero. The behaviour of the solution with its zonal flow oscillations is reproduced very well for both solvers as one can see from Figure 1: The results are nearly identical. In this case a $64 \times 64 \times 64$ grid was used and both runs were done on 256 cores with four clones and eight million particles.

Using the matrix-free ansatz about 49 iterations are necessary to get the solution of equation (3), whereas only 7 iterations (for solving (8)) were required with the new method. This difference in the number of iterations one can see in the behaviour of the run time of a solver step. Figure 2 shows a comparison of a series of runs on different numbers of cores, where this time the grid size was fixed at $64 \times 512 \times 512$ (strong scaling). With an increasing number of cores the solver run time falls nearly linearly for all three cases. One can also see that the new method is significantly faster than the older matrix-free version. In particular, there is no overhead due to the averaging term. The solver run times are sometimes even faster than without the averaging. This can happen because the problem to be solved

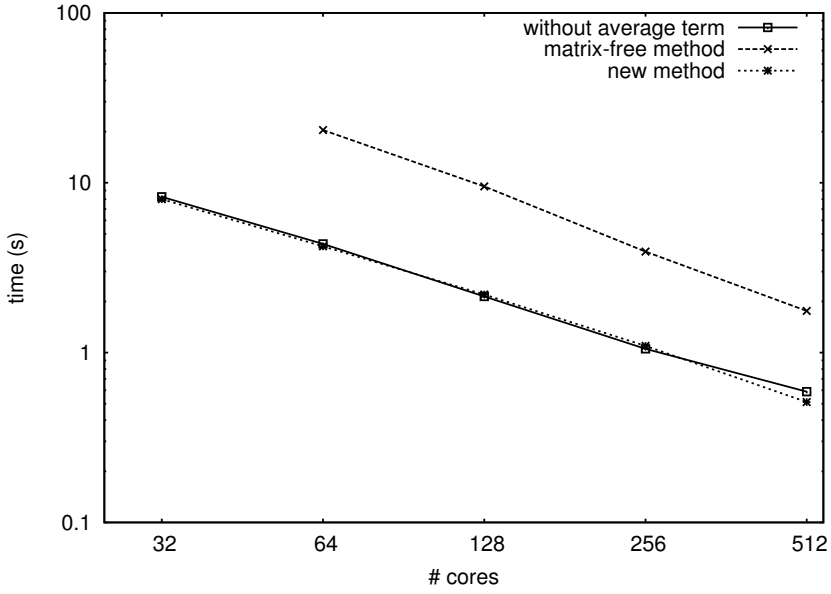

FIG. 2: Solver time for different number of processor cores and for different solution methods with a fixed mesh size of $64 \times 512 \times 512$ (strong scaling).

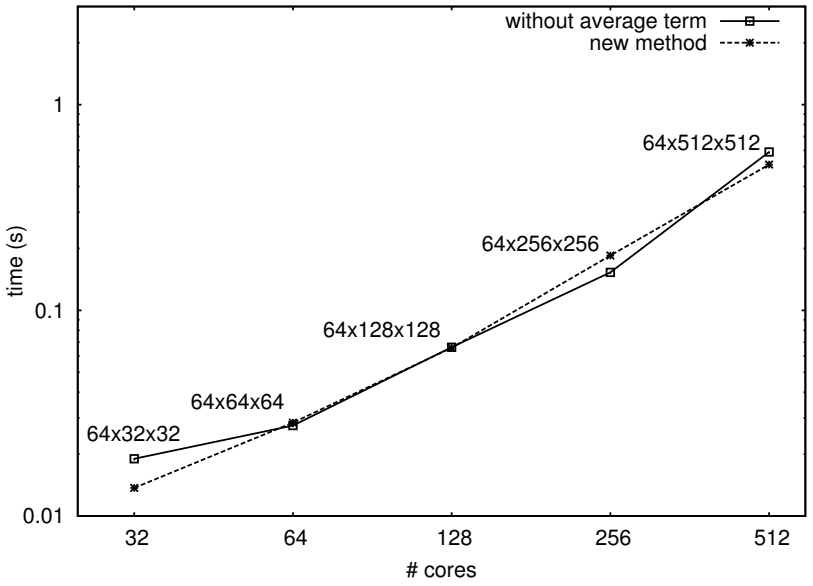

FIG. 3: Solver time for different number of processor cores and increasing mesh size, comparison between solution with and without surface average term (weak scaling).

is time dependent: During each time step, which uses a Runge–Kutta of fourth order, the field equation is solved four times. Therefore the solutions in each step are not equal and the number of iterations is slightly different in each call. If the number of cores and the grid size in ϑ and φ are increased the scaling is again nearly linear. This is shown in Figure 3 where the solver run time is plotted versus the number of cores for this case. Again it can be seen that there is no overhead caused by including the averaging term. The time scaling behaviour showed in this picture depends only on the speed of solving the system of equations (8), i.e., on the parallel iterative solver with its preconditioner.

IV. CONCLUSION

For simulating turbulence and microinstabilities in three-dimensional toroidal devices a gyrokinetic model with adiabatic electrons is often used. The electrostatic field equation then consists of a differential Helmholtz part and an integral flux-surface averaging part. The discretisation of this equation produces a sum of a sparse matrix for the first term and a memory-consuming matrix for the second term. Since already the sparse matrix is very large the use of a preconditioned iterative solver is necessary.
The first method to solve this problem was a method where the matrix-by-vector product needed by the iterative solver is calculated with a matrix-free approach for the second term. This method had the drawback of being very time consuming. With the new ansatz described here it is only necessary to solve the Helmholtz part of the equation and then to obtain the final solution by adding a correction. Thus the number of iterations and the amount of time needed for the problem with the averaging part is the same as without it. Global three dimensional turbulence simulations are only likely to be possible with a fast solver like the one described here.
It was shown that the algorithm scales nearly linearly with the number of cores both for a fixed problem size (strong scaling) and where the number of mesh points together with the number of cores is increased. In contrast to the matrix-free method the new procedure needs extra storage space for some supplementary matrices. If these matrices need too much memory the current strategy to distribute them only over the cores of one clone can be extended by using also the different clones to hold the supplementary matrices in memory. In this way it would be possible to overcome storage bottlenecks.

[1] A. J. Brizard, T. S. Hahm, Foundations of nonlinear gyrokinetic theory, Rev. Mod. Phys. 79 (2007) 421.
[2] S. Jolliet, A. Bottino, P. Angelino, R. Hatzky, T. M. Tran, B. F. Mcmillan, O. Sauter, K. Appert, Y. Idomura, L. Villard, A global collisionless PIC code in magnetic coordinates, Comp. Phys. Commun. 177 (2007) 409.
[3] N. Crouseilles, A. Ratnani, E. Sonnendrücker, An Isogeometric Analysis approach for the study of the gyrokinetic quasi-neutrality equation, J. Comput. Phys. 231 (2012) 373.
[4] V. Kornilov, R. Kleiber, R. Hatzky, L. Villard, G. Jost, Gyrokinetic global three-dimensional simulations of linear ion-temperature-gradient modes in Wendelstein 7-X, Phys. Plasmas 11 (2004) 3196.
[5] M. Kotschenreuther, Numerical simulation, Bull. Am. Phys. Soc. 33 (1988) 2107.
[6] W. D. D'Haeseleer, W. N. Hitchon, J. D. Callen, Flux Coordinates and Magnetic Field Structure, Springer, 1991.
[7] C. DeBoor, A Practical Guide to Splines, Springer, 1978.
[8] R. Kleiber, R. Hatzky, A partly matrix-free solver for the gyrokinetic field equation in three-dimensional geometry, Comp. Phys. Commun. 183 (2012) 305.
[9] G. H. Golub, C. F. van Loan, Matrix Computations, The Johns Hopkins University Press, 1996.
[10] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, H. Zhang, PETSc users manual, Tech. Rep. ANL-95/11 - Revision 3.0.0, Argonne National Laboratory, http://www.mcs.anl.gov/petsc (2008).
[11] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS, 1996.
[12] G. Grieger, C. D. Beidler, H. Maassberg, E. Harmeyer, F. Herrnegger, J. Junker, J. Kisslinger, W. . Lotz, P. Merkel, J. Nührenberg, F. Rau, J. Sapper, A. Schlüter, F. Sardei, H. Wobig, in: Plasma Physics and Controlled Nuclear Fusion Research 1990, Vol. 3, International Atomic Energy Agency, Vienna, 1991, p. 525.
[13] M. N. Rosenbluth, F. L. Hinton, Poloidal flow driven by ion-temperature-gradient turbulence in tokamaks, Phys. Rev. Lett. 80 (1998) 724.