NATIONAL TRANSPORT CODE COLLABORATION (NTCC) PTRANSP

**Final Report to the
U.S. Department of Energy
for the Period
August 1, 2007 through July 31, 2010**

**by
PROJECT STAFF**

AUGUST 2010

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

NATIONAL TRANSPORT CODE COLLABORATION (NTCC) PTRANSP

**Final Report to the
U.S. Department of Energy
for the Period
August 1, 2007 through July 31, 2010**

**by
PROJECT STAFF**

**Work supported by
the U.S. Department of Energy
under DE-FG02-07ER54935**

**GENERAL ATOMICS PROJECT 30288
AUGUST 2010**

ABSTRACT

This report describes the work done under U.S. Department of Energy grant number DE-FG02-07ER54935 for the period ending July 31, 2010. The goal of this project was to provide predictive transport analysis to the PTRANSP code. Our contribution to this effort consisted of three parts: (a) a predictive solver suitable for use with highly non-linear transport models and installation of the turbulent confinement models GLF23 and TGLF, (b) an interface of this solver with the PTRANSP code, and (c) initial development of an EPED1 edge pedestal model interface with PTRANSP. PTRANSP has been installed locally on this cluster by importing a complete PTRANSP build environment that always contains the proper version of the libraries and other object files that PTRANSP requires. The GCNMP package and its interface code have been added to the SVN repository at PPPL.

This is the final project report covering the period of August 1, 2007 to July 31, 2010 for the National Transport Code Collaboration (NTCC) PTRANSP project.

Personnel: H. St. John, L.L. Lao, G. Staebler, J. Kinsey, and P. Snyder

1. TRANSPORT SOLVER MODULE

The GCNMP code generalizes the coupled set of transport equations for evolution of densities, temperature, poloidal B field and toroidal rotation by adding a separate divergence term for the turbulent flux contributions, as shown in Eq. (1).

$$\bar{M} \frac{\partial \bar{u}}{\partial t} - \frac{1}{H\rho} \frac{\partial}{\partial \rho} \left(H\rho \bar{D} \frac{\partial \bar{u}}{\partial \rho} \right) + \frac{1}{H\rho} \frac{\partial}{\partial \rho} \left(H\rho \bar{V} \bar{u} \right) + \frac{1}{H\rho} \frac{\partial}{\partial \rho} \left(H\rho \bar{\Gamma}^T \right) + \bar{W} \bar{u} = \bar{S} \quad . \quad (1)$$

In this equation, the notation is mostly standard. The matrix $\bar{W}$ represents electron-ion energy exchange and the source term $\bar{S}$ represents the external particle, energy, current drive, and momentum source terms and other losses. The time derivatives are taken at constant ρ . The neoclassical part of the transport is represented by the diffusion and convection matrices ($\bar{D}, \bar{V}$), and the turbulent contribution is given by the turbulent flux $\bar{\Gamma}^T$. $\bar{\Gamma}^T$ is assumed to be some implicit function of the field variables and the gradients.

The Newton type linearization of $\bar{\Gamma}^T$ is done by Taylor series expansion in time where θ represent the standard implicitness parameter for taking time derivatives. Hence, we write

$$\bar{\Gamma}^{T(n+\theta)} = (1-\theta) \bar{\Gamma}^{T(n)} + \theta \bar{\Gamma}^{T(n+1)} \quad , \quad (2)$$

$$\bar{\Gamma}^{T(n+1)} = \bar{\Gamma}^{T(n)} + \frac{\partial \bar{\Gamma}^{T(n)}}{\partial \bar{u}} \frac{\partial \bar{u}^{(n)}}{\partial t} \Delta t + \frac{\partial \bar{\Gamma}^{T(n)}}{\partial \bar{u}'} \frac{\partial \bar{u}'^{(n)}}{\partial t} \Delta t \quad , \quad (3)$$

$$\bar{\Gamma}^{T(n+\theta)} = \bar{\Gamma}^{T(n)} + \frac{\partial \bar{\Gamma}^{T(n)}}{\partial \bar{u}} \frac{\partial \bar{u}^{(n)}}{\partial t} \theta \Delta t + \frac{\partial \bar{\Gamma}^{T(n)}}{\partial \bar{u}'} \frac{\partial \bar{u}'^{(n)}}{\partial t} \theta \Delta t \quad . \quad (4)$$

Here, the prime represents differentiation with respect to ρ . In this form the turbulent flux can be incorporated in a finite difference representation of the transport equations [1]. The details are given under a separate document that describes the method used in detail. Note, however, that in the finite difference formulation we commute the time and space differentiation. This is a current approximation made in the code and assumes that the 2D terms are absent. These terms may be added in the future by connection to the PTRANSP XPLASMA interface. The terms are associated primarily with the evolution of the poloidal B field. Although the poloidal B field evolution equation is included in our formulation, we are aware that PTRANSP has its own internal way of dealing with this equation. Hence, current exclusion of the so called 2D source terms is not expected to be an issue. We point out that the derivative of the flux vector as indicated in Eqs. (3) and (4) are the Jacobian matrices associated with the full transport model. The elements of the matrices represent the rate of change of the

particle, energy, poloidal B field, and toroidal rotation fluxes with respect to each field variable and field gradient that is part of the subset of equations to be solved. It is the determination of the elements of these matrices that consume essentially all of the computing time.

The interface to PTRANSP allows selection of an arbitrary subset of equations to be solved from the general prescription given above. The turbulent confinement models (GLF23, TGLF) [1–5] do not necessarily apply all the way out to the plasma edge. Accordingly, an important feature of the solver module is its ability to accept arbitrary boundary locations for each selected profile individually. This feature is also essential for the generalization of the edge pedestal stability calculations described below.

GCNMP actually incorporates three methods of solving the selected subset of the transport equations. The first method assumes that a steady state solution is to be found. Accordingly, all time derivatives are assumed to be zero and the external source terms are constant in time. In this case, a Newton method is used to minimize the residuals of the transport equations. No attempt is made to linearize the turbulent flux separately, as indicated in Eq. (2). Rather, its contribution to the residual of each equation is treated implicitly. This method works quite well for simple models that are not too stiff. However, generation of the required Jacobian matrix can be prohibitively expensive for turbulent transport models and convergence is not guaranteed. We have found this method to be most useful in considering "what if scenarios". For example, given a reasonable representation of a steady state ITER AT type plasma, where the current penetration time is very long, what would a new steady state look like if the rf current were modified? In a first approximation this problem is resolved, yielding a flat toroidal electric field, in less than 1 minute of CPU time (even though it would otherwise take many time steps to reach the current penetration time level).

Two time-dependent solvers are featured in the code. In the first, in an attempt to speed up the code as much as possible, a method similar to a MacCormack type approach to non-linear transport problems is used. For this method, we also introduced a fourth order compact derivative scheme in order to minimize the noise often associated with such an approach. This method relies on evaluating turbulent fluxes only twice per time step, without determination of the Jacobian form of the turbulent transport matrices mentioned above. The chief drawback of such a method is that it is Courant condition limited, which means that the time step has to be quite small for transport grids with more than about 50 points. The final method, and the one which is typically used, is based on expansion of the turbulent flux as indicated above. In this approach the finite difference form of the transport equations is solved using a standard (sparse, tri-diagonal) matrix solver. The implicitness parameter θ is always taken as unity that eliminates the Courant condition stability issue. However, the time step must still be kept sufficiently small so as to avoid over prediction by the linearized derivatives of the turbulent flux. We have experimented with a predictor/corrector approach to optimize the time step for this method, but find that recalculation of the Jacobian elements of the turbulent flux is not advantageous.

1.1. PARALLEL SOLVER METHODS

Whichever method is selected, a parallel computational method is used to generate the turbulent flux contributions. The actual set of equations is solved in a serial manner since the size of the sparse matrices involved is not sufficiently large to warrant a parallel solver. Particularly for TGLF, the computational time involved in generating the transport matrices varies greatly between turbulent and non-turbulent grid points. It is of interest to note that TGLF typically requires about 3 s of CPU time to return the fluxes for a single grid point on a single processor. For the current TGLF model, there are two field variables and two field gradients involved with evolution of the electron and ion temperatures. This means that a total of 5 calls to the TGLF confinement model must be made for each grid point. With a grid of size n , we thus find that, approximately, $15 \times n$ second of CPU time will be required by a single processor on average to resolve the turbulent fluxes over the grid for one time step. Therefore, ideally, using n processors is the best way to decrease this time to its minimum of approximately 15 s.

The user is kept informed by how much time each processor used on its assigned grid points. This will allow limited, manual optimization of the number of processors to use. We originally developed the code with the standard MPI-1 (message passing one) interface. But version 1 of MPI did not allow us to dynamically add and delete processors, hence an automatic optimization for the number of processors to use was not possible. With the availability of an infiniband connected system, it was possible to switch away from the Portland group compiler supplied MPI libraries and instead build a custom MPI2 compliant interface for GCNMP. Accordingly, GCNMP must be built on a system that has MPI2 installed. We have used the open source MVAPICH version available from Ohio State University to achieve this goal [6].

It was found that a domain decomposition type approach was not reasonable for TGLF, because typically there is a sequence of grid points that will have turbulent transport active. If this sequence is done by the same processor, then its accumulated CPU time becomes much more than a corresponding processor that has only partial or no turbulent flux contributions. In GCNMP, using a master-slave type of parallel computational approach mitigates this problem. In this method, a master processor monitors the state of the slave processors. If any slave becomes idle, and more work is to be done, the master will assign another, arbitrary grid point to the slave for processing. The resulting communications scheme between the processor becomes more complex, but an overall gain in efficiency is obtained. Finally, we remark that there is still a significant opportunity to parallelize the TGLF confinement model further by recognizing that the low and high k modes can be done simultaneously on each grid point. The master-slave environment of GCNMP allows this modification to be made by simply passing out a communicator group instead of a single process to each grid point. However, we have not pursued this avenue since we do not have access to a sufficient number of compute nodes to make this possible.

2. STATE FILE

A very important aspect of this project was the PTRANSP interface to GCNMP. The scope of this interface has changed significantly since we first started its development. The state file, described below, now forms the backbone of GCNMP and is under continuous development to embrace new features from other projects and codes. Several starts were required to develop a sufficiently general version of the interface due to various updates in state files, and missing data.

The interface between PTRANSP and GCNMP occurs through state files. A description of the interface and its general use follows. On the PTRANSP side, the interface consists of two SWIM type NETCDF state files, identified by the inclusion of "TG1" and "TG2", respectively, in the name of the NETCDF output files that PTRANSP produces. The actual creation of these state files and some run directives are defined by input parameters to the PTRANSP NAMELIST. In order to solve the coupled set of parabolic equations described in Section 1, we must supply initial and boundary conditions covering the time interval over which the equations are to be solved. It is the function of the TG1 state file to supply the initial conditions. This includes initial profiles for densities, temperatures, poloidal B field and toroidal rotation. It (and TG2) also includes the important boundary location points for each profile as discussed above. Since the boundary point for each profile must be a grid point there may be some interpolation involved in setting the actual boundary used. Furthermore, since there is no dynamic gridding occurring during the time interval over which the equations are solved, we are currently limited to not changing the boundary point location over a single (TG1, TG2) simulation. Since GCNMP does not currently couple to external source, a second, equally important, function of TG1 is to supply all external source of particle, energy, momentum, and current drive. These sources are assumed to be constant during the time interval represented by (TG1, TG2).

The function of the second state file, TG2, is to supply single time point boundary conditions at the end of the time interval of interest. Linear interpolation in time, between the times given in TG1 and TG2 is used to generate a boundary condition at the specified boundary grid point for each profile that is evolved. This is the basic scheme used to make the transport information available to GCNMP.

The two state files TG1 and TG2 are mapped into a single GCNMP state file that includes all of the necessary information that the user is not likely to change. One exception is that the boundary location points may be adjusted without starting from the beginning in PTRANSP to recreate the state files. This can be accomplished only in a unidirectional way at the present time, due to the lack of values in TG2 that precede the boundary location point. GCNMP will accept a translated state file either in NETCDF or text format. It is customary to use the text format as the initial GCNMP input file so that the above editing can be easily

accomplished. Once GCNMP has completed the calculations at the TG2 time, it can write either a text or a NETCDF file. Generally, we specify the output to be in NETCDF form for further processing.

The translation from TG1 and TG2 to the GCNMP state file is done by a separate program, which was developed for this project. Similarly the output from GCNMP is translated to a copy of the TG2 state file using the same program. All quantities changed during the GCNMP calculations are stored and made accessible to subsequent PTRANSP processing in this modified TG2 file.

One final aspect of the interface is a run directives file. This is a NAMELIST type file read by GCNMP that supplies all necessary information for the type of run to be done. This includes modification of the solution time interval (but within the TG1, TG2 constrained level), specification of transport model to use, and various other items required to define a run. A separate manual describes the available inputs.

At the present time, the swim state file has 368 variables. Many included profiles have their own grids and most are integrated quantities that must be converted to densities for our use. We try to transform profiles in a consistent way using known boundary conditions, but generally, for many profiles, there is not enough information to uniquely translate from the SWIM state file to the GCNMP state file and back again without some loss of information. Therefore, some interpolation issues will most likely emerge as we get used to using this interface.

3. EPED1 EDGE PEDESTAL MODEL INTERFACE

A preliminary EPED1 [7] interface with PTRANSP has been constructed. Only very limited testing has been done. The method generates a sequence of parametrically scanned equilibria involving several global plasma parameters. The equilibria are then used in the ELITE peeling-ballooning stability code to obtain the stability boundaries. A unique stability point is obtained by matching the stability boundary to a width constraint. Presently a tedious approach is required to generate the desired results. More work and tests are needed to complete this interface and develop an automated procedure for transport modeling applications.

4. BENCHMARK RESULTS

The most fundamental check one can make on transport code calculations is to reproduce analytic solutions in certain limiting cases. Such a check is part of the basic elements of GCNMP and verifies that a parabolic solution can be achieved to about 4 digits of accuracy under appropriate circumstances. The details are provided in an example case included in the distribution.

Checking codes for turbulent transport consistency typically involves comparing output from independent codes for the same case. For highly non-linear confinement models where multiple branch paths (bifurcations) can occur (depending on often unknown or vague assumptions made), it is generally not expected that exact matches can be made. In what follows we show how GCNMP GLF23 calculations compare with the PTRANSP results using another solver and how GCNMP TGLF results compare with XPTOR results. In matching experimental results, it should be kept in mind that the boundary conditions can influence turbulent transport significantly. Typically a rapid rise in the predicted temperature will influence the entire profile and lead to larger than expected central temperatures. Hence, the ability to adjust the range over which the turbulent transport models act remains a critical input in this modeling. All TGLF simulations shown below used the modified Miller geometry [8].

In Fig. 1 we show a typical ITER simulation run with another solver in PTRANSP and GCNMP. The figure shows that GCNMP simulations can adequately match PTRANSP results. The length of the simulation was about 5 energy confinement times. The GCNMP results took $\rho = 0.8$ as the boundary and hence values for larger ρ will be exactly identical.

Figure 2 shows the TGLF predictions for an L-mode DIII-D discharge together with the results obtained by the independent XPTOR code [4]. It is seen that the assumed neoclassical background diffusivities can influence the results substantially, particularly in regions where the turbulent transport is off. The GCNMP results show that with everything else held constant, a different central temperature is obtained due to the use of different neoclassical formulations. These neoclassical models differ most significantly near the magnetic axis and explain the differences with XPTOR.

An example of the improvement in matching experimental temperature profiles using TGLF instead of GLF23 is shown in Fig. 3. The same boundary values at $\rho = 0.9$ were used for both cases. In Fig. 4 we compare DIII-D L- and H-mode discharges using TGLF confinement. The L-mode discharge was run with XPTOR and GCNMP using version 1.81 of TGLF. The H-mode discharge was run with XPTOR using an updated TGLF with a new collision model (version 1.85). It is seen that substantial improvement in the match to the electron temperature is made with the newer model. This model has however not yet been released for inclusion in this work.

Fig. 1. Comparison of temperature profiles simulated using the GLF23 confinement model with GCNMP and another solver in PTRANSP for an ITER case.

Fig. 2. Electron (LEFT) and ion (RIGHT) temperature profiles computed using TGLF for the DIII-D L-mode discharge 101391. The blue curves are the XPTOR results. The red curves are the experimental measurements. The black and green curves are GCNMP results using the flow equation approach with Chang-Hinton and Rawls-Chu-Hinton neoclassical diffusivities.

Fig. 3. Comparison of simulated electron and ion temperature profiles using TGLF and GLF23 against experimental measurements for DIII-D discharge 132003 at 4.5 s.

Fig. 4. Comparison of temperature profiles for DIII-D L- and H-mode discharges 90761 and 98889. TGLF results for 98889 using XPTOR were based on TGLF version 1.85 with a new collision model. All GCNMP and XPTOR results for 90761 were obtained with the older TGLF version 1.81.

As a final example, DIII-D discharge 136853 is shown in Fig. 5. This graph shows that GLF23 confinement can do quite well. As shown in the second part of the figure, the ETG modes lead to a sharp peak in the electron energy flux near the magnetic axis holding the electron temperature down.

Fig. 5. GLF23 predicted temperature and energy fluxes for DIII-D discharge 136853.

5. SUMMARY

The work described above consists of the GCNMP package, its state file translator, and a preliminary interface to the EPED1 pedestal model. Both the GCNMP package and its state file translator have been added to the PTRANSP SVN repository at PPPL. Updates to these modules will continue to be made in the future, both to improve the models and to continue fleshing out the GCNMP/PTRANSP interface. One primary objective is to extend the methods developed here to include the latest version of TGLF that has toroidal rotation and a number of other improvements not available in the current version.

These packages will be embedded in PTRANSP in the future by the PPPL TRANSP team. We have made a significant effort to ensure that such an embedding takes place seamlessly by using appropriate Fortran 95 object based modular techniques. Thus, for example, every variable is explicitly included or excluded in all subroutines, avoiding future clashes with PTRANSP variables.

REFERENCES

- [1] R.E. Waltz, G.M. Staebler, J. Candy and F.L. Hinton, Phys. Plasmas **14**, 122507 (2007).
- [2] E. Kinsey, G. M. Staebler, and R. E. Waltz, Phys. Plasmas **1**, 055908 (2008).
- [3] R.E. Waltz, J. Candy, F.L. Hinton, C. Estrada-Mila, and J.E. Kinsey, Nucl. Fusion **45**, 741 (2005).
- [4] G.M. Staebler, J.E. Kinsey and R.E. Waltz, Phys. Plasmas **14**, 0055909 (2007)
- [5] R.E. Waltz, G.M. Staebler, W. Dorland, G.W. Hammett, M. Kotschenreuther, and J.A. Konings, Phys. Plasmas **4**, 2483 (1997).
- [6] MVAPICH Team <http://nowlab.cse.phio-state-edu>
- [7] P.B. Snyder, R.J. Groebner, A.W. Leonard, T.H. Osborne, and H.R. Wilson, Phys. Plasmas **16**, 056118 (2009).
- [8] R.L. Miller, M.S. Chu, J.M. Greene, Y.R. Lin-Liu and R.E. Waltz, Phys. Plasmas **5**, 973 (1998).

ACKNOWLEDGMENT

This work supported by the U.S. Department of Energy under DE-FG02-07ER54935.

GENERAL ATOMICS

P.O. BOX 85608 SAN DIEGO, CA 92186-5608 (858) 455-3000