CORSICA: A Comprehensive Simulation of Toroidal Magnetic-Fusion Devices

Final Report to the LDRD Program

James A. Crotinger
Lynda LoDestro
L. Don Pearlstein
Alfonso Tarditi
Thomas A. Casper
E. Bickford Hooper

June 12, 1997

1 Introduction

In 1992, our group began exploring the requirements for a comprehensive simulation code for toroidal magnetic fusion experiments. There were several motivations for taking this step. First, the new machines being designed were much larger and more expensive than current experiments. Second, these new designs called for much more sophisticated control of the plasma shape and position, as well as the distributions of energy, mass, and current within the plasma. These factors alone made it clear that a comprehensive simulation capability, similar to that provided by the LASNEX code for inertial confinement fusion, would be an extremely valuable tool for machine design. The final motivating factor was that the national Numerical Tokamak Project (NTP) [1] had recently received High Performance Computing and Communications (HPCC) Grand Challenge funding to model turbulent transport in tokamaks, raising the possibility that first-principles simulations of this difficult and important process might be practical in the near future. We felt that the best way to capitalize on this development was to integrate the resulting turbulence simulation codes into a comprehensive simulation.

There are several key problems that have to be solved in order to do comprehensive simulations in an efficient fashion. Such simulations must include the effects of many *microscopic* length- and time-scales—scales that are orders of magnitude shorter than the size of the device and the time-scales that characterize its operation (such as the energy confinement time). These processes include:

- relaxation to lowest-order force balance (ideal magnetohydrodynamics (MHD)), which determines the structure of the magnetic field;
- transport along magnetic field lines; and
- micro-instabilities and the resulting turbulent fluctuations.

Direct global simulation of an experiment, on the fastest time-scale, for the full duration of its operation, had been advocated by some. Estimates of the required computing power indicated that such simulations would not be feasible for quite some time. In order to do a comprehensive simulation efficiently, the length- and time-scale disparities must be exploited. We proposed to do this by coupling the average or quasistatic effects from the fast time-scales to a slow-time-scale transport code for the macroscopic plasma evolution.

In FY93–FY96 we received Director's Initiative (DI) funding to investigate algorithms for computationally coupling such disparate-scale simulations and to implement these algorithms in a prototype simulation code, dubbed CORSICA.[1] Work on algorithms and test cases proceeded in parallel, with the algorithms being incorporated into CORSICA as they became mature. In this report we discuss the methods and algorithms, the CORSICA code, its applications, and our plans for the future.

[1] Corsica's syllables share the acronym CSC with "Comprehensive Simulation Code," which is as close as I can come to explaining why this name was selected.

2 Summary

Three major coupling problems were identified that demonstrate the advantages of multiple-scale coupling: The quasistatic evolution of the magnetic geometry, the coupling of the core and edge transport calculations, and the coupling of the core transport calculation to a turbulence simulation code. In this section we briefly describe these problems, summarize our approach to solving them, and briefly discuss our progress. Detailed discussions follow in Sections 3–7, and a short discussion of our future plans is given in Section 8.

2.1 Quasistatic evolution of magnetic geometry

The first two fast time-scales that we want to avoid in the computation are the Alfvén time-scale and the parallel transport time-scale. If the plasma is MHD stable, it will relax to its lowest-order force balance in an Alfvén time, which is typically on the order of microseconds. We can eliminate this process completely by solving the ideal MHD equilibrium equations to determine the steady-state magnetic geometry. In axisymmetric configurations, which are the only ones that will be considered, the magnetic field lines in the confined region of the plasma will form toroidally nested surfaces, called flux surfaces. The parallel transport time-scale is roughly the time in which energy and particles are transported within a flux surface. For strongly magnetized plasmas, where the cyclotron frequency is much larger than the collision frequency, this time-scale is much faster than the energy confinement time, which is the time-scale that we want to simulate. Thus, we can assume that the plasma densities and temperatures will be constant on the flux surfaces.

In summary, we can describe the transport of energy and mass across the flux surfaces via one-dimensional flux-surface averaged transport equations, written in the curvilinear coordinate system provided by the solution of the MHD equilibrium problem, which now takes the form of the Grad-Shafranov equation for the flux surfaces.

These calculations are coupled in that the solution to the Grad-Shafranov equation provides certain metric coefficients that appear in the transport equations, and the transport equations determine pressure and magnetic flux profiles that appear as inputs to the Grad-Shafranov equation. This approach to coupling the geometry to the transport problem was first described by Grad and Hogan in 1970 [2, 3]. There were difficulties in practice, however, that have led to this approach not being used successfully until recently.

Solving this coupling problem turned out to be more difficult than we had originally anticipated. In addition to dealing with the geometric coupling problems, we also had to properly handle the penetration of skin-current at the plasma edge, to integrate the circuit equation time-advance with the MHD equilibrium solution, and to deal with several numerical problems. Our solutions to these problems are described in Section 3. Our prototype comprehensive simulation code, CORSICA,

implements these ideas by coupling a core transport module to the TEQ ideal MHD equilibrium code and to circuit equations describing the external conductors. CORSICA is described briefly in Section 4. It is now being used for several important MFE applications, which are described in Section 7. CORSICA also forms the base for the core-edge and core-turbulence coupling problems, summarized below.

2.2 Coupling the core and edge regions

As mentioned above, in the confined region of the plasma (also known as the *core*), the magnetic field lines form nested toroidal flux surfaces. This confined region can only have a finite extent. At some point the topology is either broken by direct contact with a material surface (a *limited* plasma) or by diverting field lines away from the plasma using currents in external coils (a *diverted* plasma). In the latter case, a separatrix surface exists outside of which field lines are diverted into a target plate away from the core plasma. In either case the modeling of the unclosed flux surfaces (and of a boundary layer that extends slightly into the core region) requires solving a two-dimensional (2D) transport problem.

This edge transport problem is coupled to the one-dimensional core transport problem at the core-edge boundary, which is located far enough inside the closed-flux-surface region to ensure that the pressure and density are constant on flux surfaces. Thus each region provides the boundary conditions for the other. The challenge is to find an efficient method for coupling these regions. This is important because the edge simulation codes, which must simulate the fast parallel transport time-scale, are very expensive to run. The resulting coupled system must be capable of taking time-steps on the energy confinement time-scale in order to minimize the number of edge transport calculations that are done.

The algorithms that we tested and developed for this portion of the project are described in Section 5. These algorithms have been implemented in a version of CORSICA, known as CORSICA 2, that includes the UEDGE 2-D fluid edge code. Our tests of CORSICA 2 (see Section 7.4) demonstrate code robustness and the ability to deal with experiment-relevant conditions. Despite the preliminary nature of this work, the results obtained so far are credible and in good agreement with experimental data. Work is in progress towards both improving the algorithm performance and refining the physics models used in CORSICA 2 (and in UEDGE). The general idea of coupling existing core and edge simulations is now being pursued by several other groups [4–6].

2.3 Coupling core transport and turbulence calculations

Transport of plasma energy and density across flux surfaces is primarily due to turbulent processes that have small time-scales and space scales. Since the scale lengths are quite disparate, evolution on the long time-scale should be governed by 1D flux-surface-average equations in which anomalous fluxes appear. These anomalous fluxes

are calculated by running the turbulence simulation.

The effects of these fluxes feed back into the turbulence simulation codes via changes in the density and temperature profiles (the turbulence arises out of instabilities that are driven by the free energy available from the density and temperature gradients). The goal is self-consistent evolution of the average profiles with the turbulence. This is a very challenging problem for several reasons:

- The average fluxes are generally nonlinear functions of the density and temperature profiles and their derivatives. This functional dependence can only be calculated by doing large-scale numerical simulations for a given set of profiles, which are very expensive. Furthermore, such nonlinear flux-gradient relationships impose severe time-step restrictions on the the core transport code unless the core-transport time-stepping algorithm is implicit.

- The fluxes are not necessarily diffusive.

- The average fluxes can only be estimated by averaging simulation data over finite spatial and time domains. There will always be some residual turbulent (short-time-scale) noise in these averages, and making this noise sufficiently small to take accurate numerical derivatives is prohibitively expensive. Thus the usual implicit time-stepping schemes are not available.

Our work on this problem is described in Section 6. This portion of the project turned out to be doubly challenging because the state-of-the-art turbulence simulation codes are not yet robust black boxes that can blindly be called by a transport code to get flux estimates. Nevertheless, we have developed a very promising method for solving this coupling problem. Our method has been demonstrated with an application to the 2D Hasegawa-Wakatani model equations for edge-plasma turbulence. We have augmented the basic scheme with two methods, about equally successful, for dealing with locally anti-diffusive behavior. Both local and global implementations of the coupling work well. For the problems where there is at least a moderate separation of spatial scales between the fluctuations and the background, the coupled approach achieves significant savings over the comparison stand-alone simulations, whether or not global effects emerge, while finding, to within expected statistical variations, the same averaged profiles.

For the parameters used in our simulations and for parameters of typical interest for drift-wave-type turbulence in tokamaks, the global implementation of the coupling (in which a single large turbulence code with the same radial domain as the transport code is used) would more efficient than the local one (in which separate copies of the turbulence code are used at each transport mesh-point).

A new version of Corsica, Corsica 3, incorporates the ability to communicate with turbulence simulation codes (via distributed computing, if desired) and the software implementation of our turbulence coupling algorithm. Using the premier 3D tokamak turbulence code, the radially local gyrofluid code Gryffin [7],

initial experiments have been performed coupling CORSICA's transport module to ion-temperature-gradient turbulence simulations. Coupling to a band of eight GRYFFIN's in the outer portion of a tokamak core, converged runs were obtained with a straightforward application of the basic scheme and techniques developed in 2D. Full-core coupled calculations proved much more difficult. The main problem was that GRYFFIN is less robust in the much more unstable inner core. Also, the flux becomes a very sensitive function of the profiles in this regime. Finally, during the iteration the profiles developed extrema that were not expected, and the local codes did not handle this region well. We developed controls that monitored many of the coupled variables and attempted to steer the iteration through difficulties, and in the end we did obtain a converged self-consistent simulation over *much*, but not all, of the core.

Finally, we note that the important issue of multiple-field coupling remains to be addressed before this algorithm could be applied to the physical problems of greatest interest.

3 Coupling core transport and MHD equilibrium calculations

Before describing the details of the coupling algorithms, we must first introduce the basic geometry, the assumptions that are made in this type of analysis, and the basic equations that are being solved. The geometry of the device (and of the plasma) is assumed to be toroidally symmetric, or *axisymmetric*. This is not perfectly true in tokamaks, since there are a finite number of toroidal magnetic field coils, but these effects are generally small and can be handled perturbatively.

3.1 Coordinate systems

Two coordinate systems will be used to describe the plasma. The first is a cylindrical system $\{R, \varphi, z\}$, where z is the coordinate along the axis of symmetry, R is the radial distance from this axis, and φ is the toroidal angle.

The second set of coordinates are the toroidal *flux coordinates*. In devices of interest, the magnetic field lines, produced by currents in the plasma and in external conductors, will lie on toroidally nested surfaces, called *flux surfaces*. These surfaces, which are toroidally symmetric, are typically represented by plotting their intersection with a φ = constant surface as contours in the R-z plane, as shown in Fig. 1. These surfaces are important because transport along the magnetic field lines is extremely fast compared to transport across the flux surfaces. As a result, *on the transport time-scale* many of the physical quantities of interest (plasma densities, temperatures, pressures, etc.) are constant on the flux surfaces, or *flux functions*.[2] Thus we introduce a toroidal coordinate system $\{\rho, \theta, \varphi\}$, where ρ is a flux surface label (such as the volume of the surface, or the toroidal magnetic flux contained within the surface), φ is the toroidal angle, and θ is an angle-like *poloidal* coordinate. The ρ and θ coordinates are not, generally, orthogonal, as shown in the Fig. 2. In this coordinate system, the evolution of the physical variables is described by one-dimensional transport equations. These equations will be summarized below, but first we discuss the calculation of the magnetic geometry. (For a detailed discussion of flux coordinates, see [8].)

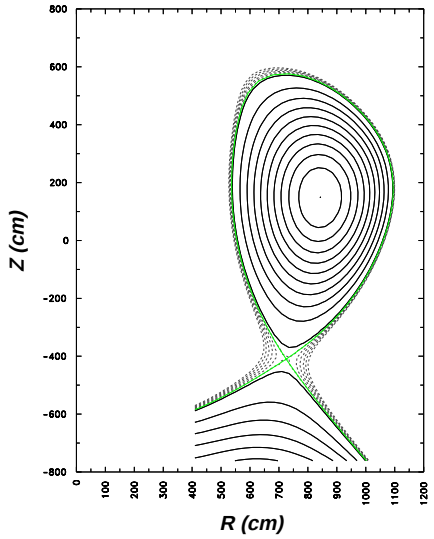

Figure 1: Flux surfaces plotted in the $R - z$ plane.

[2] If the plasma rotates toroidally at near sonic speeds, the mass is pulled outward and the density and pressure are no longer flux functions, although a different set of flux functions can be defined. CORSICA works in the low Mach number limit.

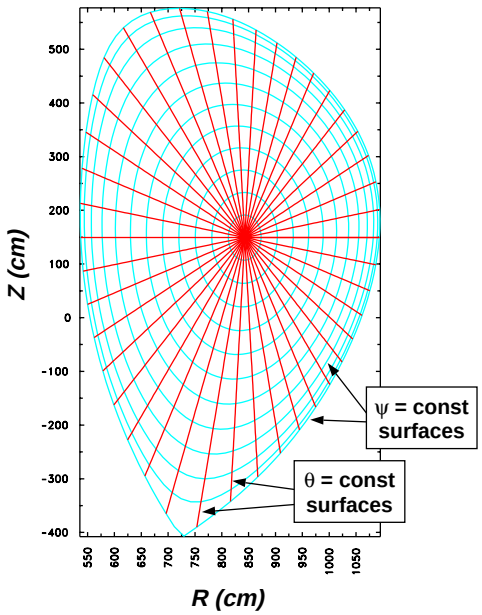

Figure 2: ψ and θ coordinate lines for solution shown in Fig. 1.

3.2 Magnetic Geometry

The magnetic field in such an axisymmetric system can generally be written as

$$\mathbf{B} = \nabla\varphi \times \nabla\psi + F\nabla\varphi \ , \tag{1}$$

where ψ is the stream function for the poloidal component of the magnetic field vector, and $F = RB_{\mathrm{T}}; B_{\mathrm{T}} \equiv \mathbf{B}\cdot\hat{\mathbf{e}}_\varphi$ ($\hat{\mathbf{e}}_\varphi$ is the unit vector in the φ direction). It can be shown that axisymmetry implies $F = F(\psi)$.

It is obvious that ψ is a flux function, and indeed it can be shown that $\psi = \psi_p/2\pi$, where

$$\psi_p \equiv \int \mathbf{B}_{\mathrm{p}} \cdot d\mathbf{A} \tag{2}$$

is the *poloidal flux*—the flux of $\mathbf{B}$ through the washer shaped surface bounded by the intersections of a horizontal plane with the *magnetic axis* (the flux surface with zero enclosed volume) on the inside and the flux surface labeled by ψ on the outside.[3]

For axisymmetric configurations with no flow, it can be shown that the equilibrium ψ satisfies the Grad-Shafranov equation [9]:

$$\Delta^*\psi = -\mu_0 R^2 \frac{\partial p}{\partial \psi} - F\frac{\partial F}{\partial \psi} \ , \tag{3}$$

[3]Note that this is just one possible definition of the poloidal flux, chosen to go to zero at the magnetic axis. In the free boundary calculation described below, the flux is defined to vanish infinitely far from the plasma (and on the z-axis). This corresponds to measuring the flux of $\mathbf{B}_{\mathrm{p}}$ through a disk lying in the mid-plane of the plasma and bounded on the outside by the intersection with a particular flux surface. These two definitions differ only by a constant, which has no physical significance since only $\nabla\psi$ appears in the definition of $\mathbf{B}$.

where the elliptic operator $\Delta^*\psi$ is defined by

$$\Delta^*\psi \equiv R^2 \, \nabla \cdot \frac{1}{R^2} \nabla\psi \; . \tag{4}$$

This equation must be solved iteratively in order to deal with the nonlinear dependencies on ψ from p and F, and to calculate the boundary in the free-boundary case.

There are two classes of boundary conditions that are typically considered. First, one can specify a particular bounding flux surface and solve Eq. 3 inside this domain subject to specifying ψ on the bounding surface, and either $\delta\psi$ between the boundary and the magnetic axis, or the total plasma current. (Again, the particular constant chosen for the boundary condition is not physically important.) This is called the *fixed-boundary* problem.

The second type of problem, the *free-boundary* problem, is much more complicated. In this case the bounding surface for the plasma, outside which the pressure effectively vanishes, is determined either by the intersection of flux surfaces with a physical *limiter* or by the presence of a separatrix, beyond which the magnetic field lines are diverted away from the plasma (by external coils) to intersect with walls or with specially-designed *divertor* targets. In either the diverted or the limited case, the exact shape of the bounding surface is unknown *a priori*. One solves for the total flux due to the plasma current and to currents in the external conductors (including both the poloidal magnetic field coils (*PF coils*) and passive conducting material, such as the vacuum vessel). Then the bounding surface can be determined. This process is done iteratively, in concert with the iterations on the Grad-Shafranov nonlinearities. Finally, the boundary condition for ψ is that it is well-behaved at infinity. (In practice, a finite grid is used and ψ along the boundary of this grid is calculated iteratively from the contributions of all of the currents within the domain.)

3.3 Slow-time-scale plasma evolution

In general, time-dependent plasma evolution is extremely complex. Even the simplified collisional one-fluid MHD model yields an entire zoo of plasma modes and instabilities. However, these motions tend to have relatively fast time-scales and are either catastrophic or are such that they can be treated as perturbations on the axisymmetric evolution of the plasma. Confinement devices are carefully designed to avoid the catastrophic events, such as ideal MHD instabilities. The remaining events are modeled as enhancing transport, which will continue to be calculated in one-dimension.

The geometry is calculated by solving the Grad-Shafranov equation. This ignores the plasma inertia and the (fast time-scale) Alfvén waves that arise when a stable plasma is perturbed away from the ideal force balance. The pressure $p(\psi)$ becomes parameterized by time, and its evolution is provided by solving the flux-surface averaged energy transport equation. The toroidal field function $F(\psi)$ is found by solving

a flux-surface averaged diffusion equation for the magnetic flux. This approach was first discussed by Grad and Hogan in 1970 [2, 3]. Despite its long history, past implementations of this approach in free-boundary simulations have met with many problems. As a result, the work-horse code for performing free-boundary transport simulations in the U.S., the Tokamak Simulation Code (TSC) from Princeton, solves the full two-dimensional time-dependent MHD problem [10]. This requires following artificial Alfvén waves, using an artificially high mass and viscosity, and sub-cycling certain calculations. The TSC approach offers a simpler, more direct implementation, but it typically requires much more computer time than the Grad-Hogan approach that is employed in CORSICA.

The evolution equation for the magnetic flux is derived by combining Faraday's Law for magnetic induction with Ohm's Law:

$$\mathbf{E} + \mathbf{u} \times \mathbf{B} = \eta(\mathbf{J} - \mathbf{J}_{\text{driven}}) \,, \tag{5}$$

where $\mathbf{E}$ is the electric field, $\mathbf{u}$ is the plasma velocity, η is the plasma resistivity, $\mathbf{J}$ is the total plasma current, and $\mathbf{J}_{\text{driven}}$ is the portion of the current driven by non-inductive processes. Substituting this equation into Faraday's Law, one can derive the following evolution equation for the flux function ψ:

$$\frac{\partial \psi}{\partial t} = \frac{V_{\text{loop}}}{2\pi} \,, \tag{6}$$

where V_{loop} is the loop voltage ($\oint \mathbf{E} \cdot d\ell$ along a single toroidal circuit around the plasma). This is referred to as the flux-surface averaged Ohm's law. The loop voltage is given by

$$V_{\text{loop}} = \eta(J - J_{\text{driven}}) \tag{7}$$

$$= \eta \left(\frac{\langle \mathbf{B} \cdot \mathbf{J} \rangle}{\langle \mathbf{B} \cdot \nabla \varphi \rangle} - J_{\text{driven}} \right) \tag{8}$$

$$= \eta \left(F^2 \frac{\partial}{\partial \Phi} \frac{I}{F} - J_{\text{driven}} \right) \,, \tag{9}$$

where

$$J_{\text{driven}} = \frac{\langle \mathbf{B} \cdot \mathbf{J}_{\text{driven}} \rangle}{\langle \mathbf{B} \cdot \nabla \varphi \rangle} \,, \tag{10}$$

and where $I(\psi) = \mu_0^{-1} \oint \mathbf{B} \cdot d\ell$, the integral being taken around a poloidal circuit of the plasma surface, is the toroidal current flowing in the volume enclosed by the flux surface, and where we have now explicitly chosen $\rho = \Phi$, the toroidal magnetic flux enclosed within a flux surface. Also, $\langle \cdot \rangle$ is the flux-surface average operator:

$$\langle A \rangle = \frac{\int A \, \mathcal{J} \, d\varphi \, d\theta}{\int \mathcal{J} \, d\varphi \, d\theta} \,, \tag{11}$$

where $\mathcal{J}$ is the Jacobian for the transformation to flux coordinates. It is more convenient to evolve the rotational transform, ι ("iota-bar"), defined by

$$\iota \equiv \frac{1}{2\pi}\frac{\partial \psi_p}{\partial \Phi} = \frac{\partial \psi}{\partial \Phi} \ . \tag{12}$$

Taking the Φ derivative of Eq. 6, we can write the following diffusion equation for ι:

$$\frac{\partial \iota}{\partial \Phi} = \frac{\partial}{\partial \Phi}\eta\left(F^2\frac{\partial}{\partial \Phi}\frac{Iq}{F}\iota - J_{\text{driven}}\right) \ , \tag{13}$$

where $q \equiv 1/\iota$ (known as the *safety factor*). The quantity Iq/F is actually a metric coefficient and not simply inversely proportional to ι, as the appearance of q indicates. Indeed, for circular flux surfaces of radius r, with $r/R \ll 1$, it can be shown that

$$\frac{Iq}{F} = \frac{2\pi}{\mu_0}\frac{r^2}{R^2} \ . \tag{14}$$

Equation 13 is the desired resistive diffusion equation for the toroidal flux. The solution to this equation can be integrated to give ψ, but only if the value of ψ is known at the magnetic axis. This is found by solving Eq. 6 directly on the magnetic axis.

The quantity Iq/F vanishes at the magnetic axis. Thus Eq. 13 is singular and requires no boundary condition there. At the edge, the boundary condition is given by

$$\iota\big|_{\Phi=\Phi_{\text{edge}}} = \left.\frac{2\pi\mu_0 I_p}{V'\left\langle\frac{|\nabla\Phi|^2}{R^2}\right\rangle}\right|_{\Phi=\Phi_{\text{edge}}} \ , \tag{15}$$

where I_p is the total toroidal plasma current and $V' = \frac{\partial V}{\partial \Phi}$, V being the volume enclosed by the flux surface. Finally, the value of Φ at the edge of the plasma, Φ_{edge}, can vary with time and must be supplied in order to solve the system.

3.4 Particle and energy transport

The flux-surface averaged transport equations for particles and energy are easy to derive in the classical fluid limit, and are discussed in detail in the Corsica Users' Manual [11] and in [12–15]. Here we summarize the equations when written with the toroidal flux as the flux surface label. The ion density transport equation is:

$$\frac{\partial}{\partial t}n_j V' + \frac{\partial}{\partial \Phi}\Gamma_j V' = S_j V' \ , \tag{16}$$

where n_j is the particle density for ion species j, Γ_j is the flux surface average of $\Gamma \cdot \nabla\Phi$ (i.e. of the contravariant "radial" component of the particle flux vector), and S_j is the particle source density.

The energy equations are written in terms of the *entropy density* for the electrons and ions, $\mu_{e,i} = p_{e,i} V'^{5/3}$. (The temperature equilibration time is generally quite fast, and is much faster for different ion species than for ions equilibrating with electrons. Since some sources preferentially heat electrons or ions, substantial electron-ion temperature differences can exist, but differences between the temperatures of different ion species are usually ignorable.) They can be written as:

$$\frac{3}{2}\frac{1}{V'^{2/3}}\frac{\partial \mu_{e,i}}{\partial t} + \frac{\partial}{\partial \Phi}\left(q_{e,i}V' + \frac{5}{2}\Gamma_{e,i}T_{e,i}V'\right) = Q_{Ee,i}V' \,, \tag{17}$$

where q is the flux surface average radial (again, the contravariant radial component) heat flux, $T_{e,i}$ is the temperature, and $Q_{e,i}$ is a source term, including both external sources and collisional sources.

This system is not closed since we don't have equations for the particle and heat fluxes. The experimental observation is that these fluxes are much larger than the predictions of the classical and neoclassical (fully-toroidal) theories of transport in a quiescent plasma. The observed transport levels and the measured fluctuation levels indicate that the transport is due to turbulent processes [16–18]. While there are no exact theories for these turbulent transport fluxes, there are many approximate ones, several of which are available in the CORSICA code. Furthermore, progress has been made on simulating such turbulent plasmas numerically, and we have investigated the possibility of coupling directly to these simulation codes (as will be discussed in Section 6).

3.5 The coupling algorithms

Careful accounting of the inputs and outputs of both the equilibrium calculation and the calculation of the resistive diffusion of flux shows that there is a possible inconsistency between the two calculations. Eliminating this inconsistency is the crux of the coupling problem. Our solutions to this problem and to a couple of other subtle problems are explained below for both the fixed and free boundary cases.

3.5.1 Free boundary

The free boundary MHD equilibrium calculation requires the p and $q = 1/\iota$ profiles,[4] the total poloidal flux within the plasma, $\delta\psi \equiv \psi_{\rm edge} - \psi_{\rm axis}$, and the values of ψ at the external coils, $\{\psi_i\}$. The outputs are the total plasma current, I_p, the total toroidal flux, $\Phi_{\rm edge}$, the magnetic geometry (which is used to calculate various metrics for the flux-coordinate system), and the currents in the external conductors, $\{I_i\}$. Note that since the external currents will be evolved via a set of circuit equations, the problem is not posed in the traditional manner, in which the external currents are input and the values of the flux at the coils are output. Also note that it is F that appears on

[4]For historical reasons, q, rather than ι, is considered the input to this calculation.

the right hand side of Eq. 3, not q. F must be calculated from the geometry and the q profile. This is done internal to the equilibrium code as the geometry is being calculated.[5]

The first problem is that the solution for F may be discontinuous at the plasma boundary. This means that $B_{\rm T}$ is discontinuous, implying the existence of a surface current. While surface currents can exist in an *ideal* ($\eta = 0$) plasma, we assume that the resistivity on the plasma boundary is sufficiently high that the local skin time is *much* shorter than any time-scale of interest. The problem is corrected by taking the skin-current that is required to create the discontinuity in F, and adding it to the plasma current when it is used in the ι boundary condition, Eq. 15. This is equivalent to replacing I_p in that equation by

$$\hat{I}_p = \frac{F(\psi_{\rm edge})}{F_{\rm wall}} I_p \ , \tag{18}$$

where $F_{\rm wall}$ is the value of F in the vacuum region. The q-profile is recalculated to include this current (which will be smeared out through diffusion) and the result is fed back to the equilibrium code. As the solution converges, $F(\psi_{\rm edge}) \to F_{\rm wall}$.

Next consider the inputs and outputs for the resistive diffusion equation (the other transport equations pose no special problems). Solving this equation requires $\hat{I}_p$, $\Phi_{\rm edge}$, and several metric coefficients for the flux-coordinate system. The outputs are the ι profile and the value of $\psi_{\rm axis}$ (the value of $\delta\psi$ is also needed, but this is simply the integral of ι). Unfortunately, the value of $\psi_{\rm axis}$ calculated by the transport code will not necessarily be the same as the value of $\psi_{\rm axis}$ calculated by the equilibrium code! This indicates that the boundary of the plasma is not at the correct location relative to the limiter or separatrix and that it must be moved. Rather than solve for the detailed dynamics of this process, we again make a long-time-scale assumption: The plasma near the boundary is assumed to always be in quasistatic equilibrium and to always terminate *on* the limiter or separatrix (in practice we must stop a very small distance from a separatrix, since the flux-coordinates become singular there). To accomplish this, we use a heuristic feed-back algorithm that modifies the values of q and $\delta\psi$ that are passed to the equilibrium code in proportion to the discrepancy in $\psi_{\rm axis}$.

3.5.2 Fixed boundary

The fixed boundary MHD equilibrium calculation requires the p and q profiles, the total poloidal flux within the plasma, $\delta\psi \equiv \psi_{\rm edge} - \psi_{\rm axis}$, and the shape of the plasma boundary. The outputs are I_p, $\Phi_{\rm edge}$, and the magnetic geometry (and, indirectly, the F profile). The skin current problem arises and is solved as described above. However, it now makes no sense to compare the value of $\psi_{\rm axis}$ calculated in the transport code

[5] Developing an equilibrium solver that could solve such problems (a "q-solver") was a necessary first step that was taken before the CORSICA project got underway [19].

to that calculated in the equilibrium code as there are no coils to set the magnitude of ψ. There are actually many possible procedures for dealing with this extra degree of freedom. What is typically done is to replace the feedback on the error in $\psi_{\rm axis}$ with feedback on the plasma current, in order to force the total plasma current to follow a particular time history. This alternative can also be used in the free-boundary case, where it can be thought of as representing the impact of a virtual infinite solenoid along the symmetry axis, which, in effect, adjusts the overall magnitude of ψ in the equilibrium code to give the desired plasma current. The values of $\psi_{\rm axis}$ in the equilibrium and transport codes will then differ as a function of time by the amount of flux that has been produced by this virtual solenoid.

3.5.3 Numerical subtleties

The above discussion of input and output data was not completely precise. The equilibrium code needs $p(\psi)$ and $q(\psi)$, and the transport code calculates $p(\Phi)$ and $q(\Phi)$. Similarly, the equilibrium code calculates the metric terms as functions of ψ, and the transport code needs these as functions of Φ. In the continuum limit, this would not pose a problem since $\iota(\Phi)$ can be used to transform from one coordinate system to the other. In practice, however, the equilibrium and transport codes have discrete ψ and Φ meshes. The transformations between these grids cannot be done exactly, and this can lead to numerical instabilities. This problem could be avoided by ensuring that the transformations between the coordinate systems are exact inverses of each other, in spite of the grid errors. However, this turns out to be difficult to ensure. It turns out to be sufficient to guarantee that certain integral quantities remain the same on each grid. This is done by numerically calculating a grid-size dependent factor that tends to unity as the grid is refined, and by multiplying or dividing certain geometric terms by this correction factor.

3.5.4 Evolving the coil currents

In general, the currents in the external conductors are coupled through a set of circuit equations. These equations can be written as:

$$\frac{\partial \psi_i}{\partial t} + I_i R_i = V_i \ , \tag{19}$$

where ψ_i is the flux at the coil, I_i is the coil current, R_i is the resistance of the coil, and V_i is the applied voltage on the coil (which is zero for passive conductors like the vacuum vessel). Formally we can write

$$\psi_i = L_{ij} I_j \ , \tag{20}$$

where L_{ij} is an induction matrix, which is generally a complicated nonlinear function of the coil and plasma currents. The eigenvalues of this matrix, which are the "L/R"

times for the system, span many orders of magnitude. Consequently, for efficient numerical evaluations it is necessary to make the coupled system implicit.

To explain how this is done we first need to explain how this part of the free-boundary calculation proceeds. Typically, the flux at all coils is provided and the required coil currents are computed iteratively. To compute the amount by which to correct each coil current, we write

$$\psi_i^k - \psi_i^0 + G_{ij}\ \delta I_j = 0\ , \tag{21}$$

where G_{ij} is the coil Green's function, ψ_i^0 is the desired (input) value of the flux, and

$$\delta I_j = I_i^{k+1} - I_i^k\ , \tag{22}$$

with k being the iteration number. Note that Eq. 21 is not an exact equation as it does not include the perturbations of the plasma current distribution. Such contributions certainly exist, but they cannot be calculated accurately during the equilibrium iterations, and in practice we find that Eq. 21 works satisfactorily.

Next we write out the finite difference representation of Eq. 19. For illustrative purposes we use the following difference scheme:

$$\psi_i^{\rm new} = \psi_i^{\rm old} + \Delta t\ (V_i^{\rm new} - R_i I_i^{\rm new})\ . \tag{23}$$

In order for this to be solved, the right hand side must be iterated concurrently with Eq. 21. Equating the desired flux, ψ_i^0, to $\psi_i^{\rm new}$:

$$\psi_i^0 = \psi_i^k + G_{ij}\delta I_j = \psi_i^{\rm old} + \Delta t\ (V_i^{\rm new} - R_i I_i^{\rm new})\ . \tag{24}$$

Expanding the "new" quantities about the k'th iterate gives:

$$\psi_i^k + G_{ij}\delta I_j = \psi_i^{\rm old} + \Delta t\ \left(V_i^k + \frac{\delta V_i}{\delta I_j}\delta I_j - R_i I_i^k - R_i \delta I_i\right) \tag{25}$$

$$= \psi_i^{\rm old} + \Delta t\ (V_i^k - R_i I_i^k) + \Delta t\ \left(\frac{\delta V_i}{\delta I_j} - R_i\delta_{ij}\right)\delta I_j \tag{26}$$

or

$$\left(G_{ij} - \Delta t\frac{\delta V_i}{\delta I_j} + \Delta t\, R_i\delta_{ij}\right)\delta I_j = \psi_i^{{\rm new},k} - \psi_i^k\ , \tag{27}$$

where the old iterate and old time contributions have been lumped into

$$\psi_i^{{\rm new},k} = \psi_i^{\rm old} + \Delta t\ (V_i^k - R_i I_i^k)\ . \tag{28}$$

Equation 27 is, for this simple example, the equation that the equilibrium code would solve to update the coil currents during its solution iteration. Once the solution converges, the values of $\psi_i^{\rm new}$ are also known and the circuit equations can be advanced to the next time-step. In practice we use a semi-implicit (near-centered) time-differencing scheme. The $\delta V/\delta I$ terms depend on the scheme that is being used to specify the voltage, which is usually some sort of position and shape control algorithm, an example of which will be discussed in Section 7.1.1.

4 The Corsica code

The principle deliverable from this project is the CORSICA code, a flexible and extensible software system for simulating toroidal magnetic fusion devices. This system has the capabilities of doing core transport calculations with fixed- or free-boundary MHD equilibria. A variety of models are available for anomalous transport and for the various heat and particle sources. The code can be configured to include the full UEDGE 2-D edge simulation code, and to use this capability to solve for the plasma evolution outside of the core (and in a narrow boundary layer that extends into the core). The code also has the capability to interact with other processes, possibly on remote machines, using the Portable Virtual Machine (PVM) system. This capability can be used to couple to external simulation codes for other physics, such as plasma turbulence.[6] The CORSICA system is described in detail in the CORSICA Users' Manual [11]. Only a brief overview will be provided here.

The CORSICA program is written using the Basis system [20–25][7], a software framework for developing interactive and programmable (or *steerable*) scientific programs. Basis provides CORSICA with its user interface (an embedded, Fortran-like, interpreted language), with several key software packages (graphics, binary I/O, and time history collection), and with a number of other *system* capabilities (e.g. log files, etc.).

The CORSICA Project added certain extensions to Basis to support the C++ programming language. These modifications allow Basis codes to easily access data and functions written in C++. Not only can these functions and data be accessed from the Basis parser, but software "stubs" are automatically generated to allow both Fortran and C++ routines to access the registered functions and data transparently. Furthermore, `Vector` and `Matrix` classes were developed that mimic Fortran syntax and storage, and that can have their memory managed by Basis. This allows Fortran programmers to easily learn to understand and modify the portions of the code that are written in C++. Note that these modifications do not change the fact that Basis is fundamentally *not* object-oriented. The portions of the code that are written in C++ can, and do, make use of C++'s object-oriented features, but this fact is invisible at the user interface level. See Section 8 for a discussion of our plans to move to a fully object-oriented system.

Since the focus of this project was to develop coupling algorithms and implement them in a prototype code, we tried to minimize the writing of new physics modules as much as possible. The free-boundary MHD equilibrium package, the vertical stability package, a neoclassical transport package, and a variety of diagnostics were inherited from the TEQ equilibrium code. TEQ was already a Basis code and formed the original base on which CORSICA was built. The fixed-boundary equilibrium calculation is based on the Russian POLAR1 code [26], although it has been carefully integrated

[6] The software for turbulence coupling is still under development.

[7] These documents are also available via the World Wide Web at `http://www-phys.llnl.gov/X_Div/htdocs/basis.html`.

in with TEQ so that the user interface for the two MHD equilibrium options is similar. The CORSICA core-transport package is based on the core-transport module from the SUPERCODE [27], a "systems" code that is used for fast steady-state analyses of tokamak designs. From the original SuperCode module, we took the diffusion equation solver (a Galerkin Finite Element solver that uses cubic splines as elements), some of the sources and transport models, and the overall structure for the transport code. We carefully re-derived the time-dependent transport equations and implemented them within this structure. We also added the ability to use linear finite elements, and we added a number of new sources and transport models (again, mostly by borrowing routines from other codes).

The resulting CORSICA code has proven to be a useful tool for many modeling problems. It has been applied to various design tasks, it has been been used to model various aspects of the DIII-D tokamak at General Atomics, and it has been used in the design of a spheromak experiment that is being proposed for LLNL. Its flexibility has also allowed it to be used for investigating new types of transport models, for investigating various core and edge models for the transition from low-confinement to high-confinement modes of tokamak operation (*L-H transitions*), and for investigating the coupling of the core transport to turbulence simulations. Several of these applications are discussed in the next section.

5 Coupling core and edge transport calculations

One of the key problems in designing a practical fusion reactor is finding a method of extracting the heat from the plasma. In all toroidal designs, a large fraction of the heat is carried out of the core by convection and conduction. Once this heat crosses the last closed flux-surface and enters the scrape-off-layer (SOL), it is transmitted to material surfaces (either the limiter or the divertor targets) in a parallel transport time-scale, which is much faster than the perpendicular transport time-scales. Because of this disparity, the radial extent of the SOL is very narrow, and thus the heat flux density on the target surfaces is very high. This has led to a need to design clever SOL and divertor configurations that will reduce the heat load on the target, either by spreading the width of the SOL or by promoting radiation from the SOL (usually via impurity injections).

Modeling the SOL and the boundary layer just inside the separatrix is also of interest because this region is critical to determining the overall confinement properties of the plasma. Many current machines operate in a high-confinement mode (*H-Mode*) that features a suppressed level of turbulence in the edge and SOL. This, in turn, leads to better overall performance, although it also leads to a narrower SOL.

These problems have been, and continue to be, the focus of much research, as plasma performance increases and fusion research approaches the goal of creating an ignited plasma. Most of this research focuses solely on modeling the edge and SOL region, but the core region is strongly coupled to the edge, and both core and edge modeling efforts would benefit from the ability to model the coupled system with a single code.

As mentioned in Section 2, the challenge is to find an efficient method for coupling these regions as the edge simulation codes are very expensive to run. For typical core-modeling problems, it is important that the coupled system be capable of stepping on the core transport time-scale. On this time-scale, the edge is approximately in quasi-static equilibrium, and only the steady-state edge problem need be solved. Other modeling problems, such as stepping through a transition from low confinement (*L-Mode*) to H-Mode, require stepping on the edge transient time-scale. In this case it is important that the core-edge system be capable of stepping together, and that the coupling not slow convergence significantly.

5.1 The iterative coupling scheme

The basic problem is that the coupled fields (density, temperature, momentum, etc.) and the fluxes of these fields must be continuous across the core-edge interface (a flux surface sufficiently far inside of the separatrix that the coupled variables are constant on flux surfaces in the edge calculation). For simplicity, consider a generic conserved quantity u, and its flux across the interface, Γ_u. In this notation, our continuity

requirements are:

$$u_{\text{core}} = u_{\text{edge}} \tag{29}$$

and

$$\Gamma_{u,\text{core}} = \Gamma_{u,\text{edge}} \ , \tag{30}$$

where the "edge" and "core" subscripts denote the quantities as given by the 2-D edge and 1-D core simulations, evaluated on the interface.

Both the core and edge problems can be solved separately by specifying Dirichlet boundary conditions on the core-edge interface. When run in this mode, the flux across the interface is an output that can be considered a function of the input boundary condition; i.e.

$$\Gamma_{u,\text{core}} = \Gamma_{u,\text{core}}(u_{\text{core}}) \tag{31}$$

$$\Gamma_{u,\text{edge}} = \Gamma_{u,\text{edge}}(u_{\text{edge}}) \ . \tag{32}$$

Our iterative coupling scheme starts with a guess at the value of u at the interface (u_I), uses this as the boundary condition for both the core and edge code, takes the time-step, and calculates the flux difference, $\Delta\Gamma$:

$$\Delta\Gamma(u_I) \equiv \Gamma_{u,\text{core}}(u_I) - \Gamma_{u,\text{edge}}(u_I) \ . \tag{33}$$

The problem is solved if $\Delta\Gamma = 0$. We use a Newton method to find a solution to this nonlinear problem within a specified tolerance. At each time-step we use the interface value from the previous time-step to start the iteration. The initial time-step is started by picking a reasonable value.

In practice, multiple fields must be coupled, so the algorithm is implemented for the multi-variable case (the computation of the Jacobian for the Newton iteration is done numerically by solving the fluid equations after having introduced a small perturbation for each variable on the interface).

The above algorithm could also have been formulated by specifying a flux boundary condition (which is generally a mixed-type condition) on the interface, and using the continuity of u as the nonlinear equation to be solved. This has not been attempted.

5.2 The domain-decomposition ("merging") coupling scheme

A different approach from the "classic" iterative coupling has been investigated. The basic idea is to consider the coupled nonlinear system as a whole. The solution of this nonlinear system can be found, as usual, by iterating on a series of linear solutions. A linear domain decomposition algorithm can then be applied to each of these linear problems, allowing effectively to solve each of the coupled domains independently.

In practice the outlined procedure is purely conceptual: the algorithm will never consider the whole system of equations explicitly. Each of the coupled systems is solved independently, according to a domain decomposition scheme that ensures the self-consistency of the solution, and then the procedure is iterated to achieve the convergence of the nonlinear problem. Figure 3 illustrates the scheme schematically.

The advantage of this procedure is to eliminate the iteration loop that is required to couple the different problems (the domain decomposition in fact contains no approximations, then yields the exact solution, i.e. that which would have been found by solving the model in the entire region on a single mesh), at the expense, possibly, of a slower convergence towards the solution of the nonlinear system.

5.2.1 The domain decomposition scheme

The domain decomposition algorithm is based on the "classic" Schur decomposition technique for the solution of linear system. To illustrate the concept, we will describe the simplest case of solving a one-dimensional (1D) single-field problem on uniform grid. The extension to more complex multi-dimensional cases and multiple fields is conceptually straightforward.

A linear equation $L(u) = r$ in one unknown $u = u(x)$, where x is the spatial coordinate, L is a linear operator and $r = r(x)$ is a known term, is to be solved in a system composed of two sub-domains, A and B that share a common interface point x_I and are discretized with a regularly spaced grid (the following discussion actually applies to any sort of discretization and non-uniform mesh choice).

As a result of the domain decomposition there will be two problems $L(u_A) = r_A$ and $L(u_B) = r_B$, each one being completely specified by its left and right boundary conditions (BCs). If the original boundary conditions (supposed of the Dirichlet type), before the decomposition, were $u(x = 0) = u_L$ and $u(x = L) = u_R$, the problem in A will need $u(x = 0) = u_L$ and $u(x = x_I) = u_I$ while the problem in B will need u_I and u_R. Of course at this point u_I is still unknown.

The algorithm proceeds in parallel on the two problems as follows:

1. First solve homogeneous problems with the following BCs:

   (a) Region A with unity right BC and zero left BC:

   $$L(h_{A01}) = 0, \quad h_{A01}(0) = 0, \quad h_{A01}(x_I) = 1 \tag{34}$$

   (b) Region A with zero right BC and unity left BC.

   $$L(h_{A10}) = 0, \quad h_{A10}(0) = 1, \quad h_{A10}(x_I) = 0 \tag{35}$$

   (c) Region B with unity left BC and zero right BC.

   $$L(h_{B01}) = 0, \quad h_{B01}(0) = 0, \quad h_{B01}(x_I) = 1 \tag{36}$$

THE NON-LINEAR MERGING ALGORITHM

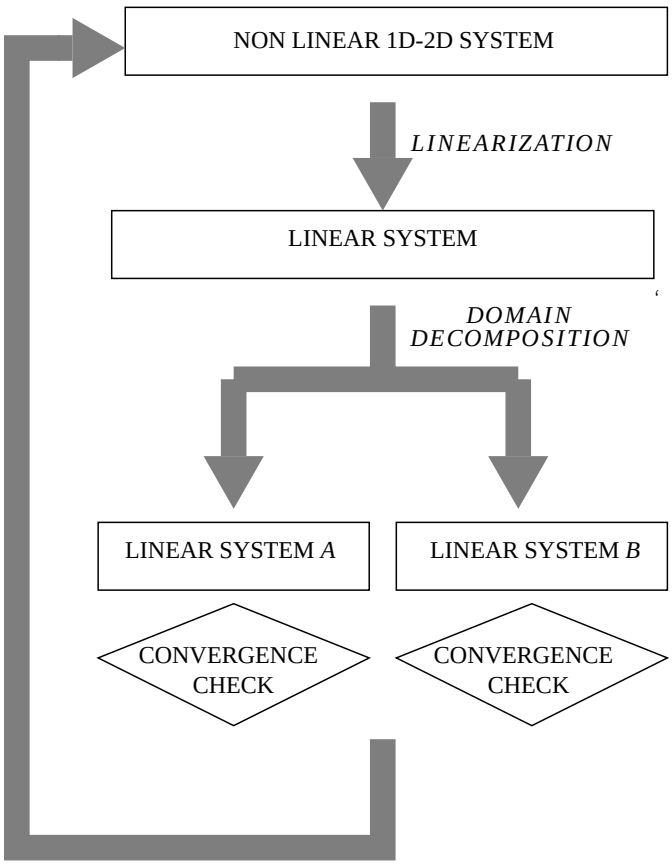

Figure 3: Schematic outline of "merging" algorithm.

(d) Region B with zero right BC and unity left BC.

$$L(h_{B10}) = 0, \quad h_{B10}(0) = 1, \quad h_{B10}(x_I) = 0 \tag{37}$$

2. Next solve inhomogeneous problems with homogeneous BCs:

   (a) Region A:

   $$L(i_A) = r_A, \quad i_A(0) = 0, \quad i_A(x_I) = 0 \tag{38}$$

   (b) Region B:

   $$L(i_B) = r_B, \quad i_B(0) = 0, \quad i_B(x_I) = 0 \tag{39}$$

3. The total solution in each region can be expressed as a superposition of these solutions:

   $$u_A = u_L h_{A10} + u_I h_{A01} + i_A \tag{40}$$
   $$u_B = u_I h_{A10} + u_R h_{A01} + i_B \tag{41}$$

4. Now solve for u_I

   $$u_A = u_A(u_I) \qquad u_B = u_B(u_I) \tag{42}$$

   from Eqs. 40 and 41, the equation $L(u_I) = r$ becomes a linear equation in u_I.

5. Compute the final solution from Eqs. 40 and 41 by substituting the actual value of u_I.

The six solutions required in the first two steps are independent and can be executed in parallel. The next step can be executed after the first two have been completed and the last immediately follows.

In the case grid of N grid points decomposed in two sub-problems, each of size $N/2$, the first step requires the solution of 4 independent problems of $N/2$ simultaneous equations. The second step requires the solution of 2 independent problems of $N/2$ simultaneous equations. The third step requires the solution of one equation, as there is only one interface point in a two-domain 1D problem. The computation of the final solution requires $N/2$ multiplications and $N/2$ additions per each sub-problem. Neglecting this last contribution the total computational cost is that of the solution of 6 systems of $N/2$ equations plus one equation, or about 3 times the effort required to solve the whole problem. If the structure of the linear equation was going to remain the same, and a solution is sought for several different time steps, the method is still essentially as fast as the global solution of N equations because the solutions of the homogeneous equation can be pre-computed and stored (as a sort of Green's functions).

In the present case of coupling two different nonlinear systems, the homogeneous solutions need to be computed at each nonlinear iteration. However the algorithm may still be more efficient than the iterative coupling method because there are no iterations required *solely* to achieve the coupling, and the total number of nonlinear function evaluations may thus be reduced.

In the case of coupling multiple fields there is an extra effort due to the need of computing as many "left" and "right Green's functions" (like the h_{A01}, h_{A10}) per subdomain as there are fields. This is somehow comparable to the effort needed in the Newton iterative coupling scheme for computing the coupling Jacobian matrix (that is computed by perturbing the boundary condition on each field and then solving the system).

5.2.2 Test runs

The described algorithm has been tested by coupling a density transport equation in an early 1D-2D core-edge test code. The purpose was to compare its performances with the standard iterative coupling scheme. For this test problem, it was found that the iterative and the merging algorithms were roughly equivalent in terms of computational speed.

More extensive tests for the more realistic multiple-field coupling problem are required to determine which algorithm is computationally superior. The major drawback to the merging scheme is that the coupling is fairly invasive—it is coupling the solutions together at the level of the internal linear solutions that are performed in the course of iterating to find solutions to the nonlinear systems in each domain. Due to the importance of having an efficient coupling, this scheme deserves further consideration. But in the course of the CORSICA project, the ease of implementing the iterative scheme swung the balance, and that is where our final efforts were focused.

5.3 Corsica 2

The iterative algorithm has been implemented in CORSICA 2 [28], a version of the CORSICA code [29]) (see Section 4) that includes the full CORSICA core transport simulation and the 2-D UEDGE edge and SOL simulation [30, 31]. This merger is illustrated schematically in Fig. 4, which shows the overlapping of the 2-D UEDGE grid with the flux surfaces that form the 1-D core grid.

In the present implementation the code can couple up to six fields: deuterium density (the electron density is determined by imposing quasi-neutrality), electron and ion temperatures, neutral gas density, (thermal) alpha particles and the axial angular momentum (toroidal flow).

The implementation of CORSICA 2 was greatly facilitated by the fact that both CORSICA and UEDGE were Basis codes. Creating a single executable was thus trivial, and the coupling algorithm was implemented using the Basis language. As all of the

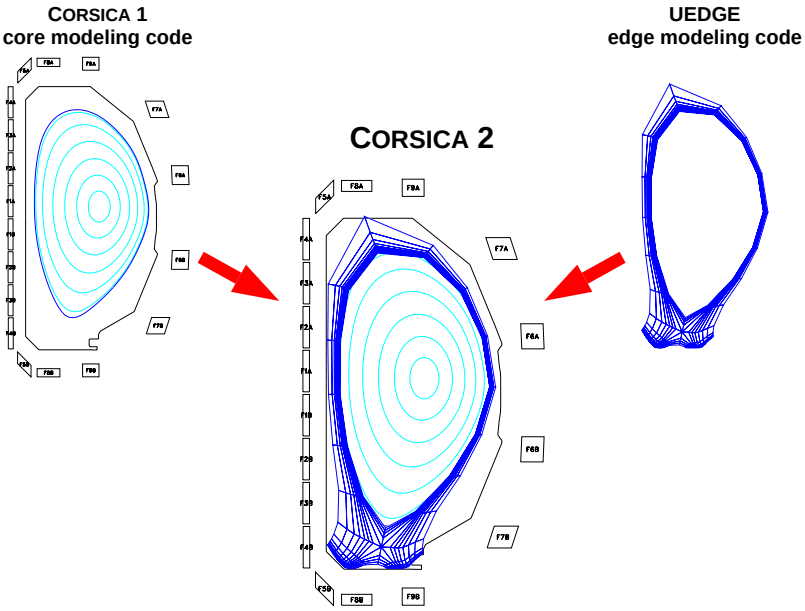

Figure 4: Merger of the 1-D core and 2-D edge/SOL simulations into a single code.

operations involving the coupled fields are done in Basis vector notation, the code can be customized easily by adding to or changing the set of coupled variables.

6 Coupling Core Transport and Turbulence Calculations

In this section we describe our general procedure for evolving slowly varying core temperature and density profiles subject to anomalous transport fluxes calculated via a turbulence simulation. This procedure is fully self-consistent: The profiles as they are calculated by the transport code using the anomalous fluxes are simultaneously used in the turbulence simulation to determine the instability drives. Section 2.3 outlined the difficulties in efficiently coupling these calculations. In this section, we first present the basic scheme and the results of its initial application, using a relatively simple one-parameter, two-dimensional turbulence model. We then conclude with the results of our effort to couple a state-of-the-art tokamak core turbulence code to CORSICA's transport package.

6.1 Basic scheme

We take as our starting point a generic nonlinear system written in conservation form:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{L}(\boldsymbol{u}) + \boldsymbol{\nabla} \cdot \boldsymbol{N}(\boldsymbol{u}) = \boldsymbol{S}\,, \tag{43}$$

where $\boldsymbol{L}$ is a linear operator, $\boldsymbol{N}$ is nonlinear, and $\boldsymbol{u} = [n, \boldsymbol{v}, T, \text{etc.}]$. We use a bar to denote averages (time, ensemble, or over some spatial dimensions), write $\boldsymbol{u} = \bar{\boldsymbol{u}} + \tilde{\boldsymbol{u}}$ where $\bar{\tilde{\boldsymbol{u}}} = 0$, and split the equations into their averaged and remaining parts:

$$\frac{\partial \bar{\boldsymbol{u}}}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{L}(\bar{\boldsymbol{u}}) + \boldsymbol{\nabla} \cdot \bar{\boldsymbol{N}}(\bar{\boldsymbol{u}} + \tilde{\boldsymbol{u}}) = \bar{\boldsymbol{S}} \tag{44a}$$

$$\frac{\partial \tilde{\boldsymbol{u}}}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{L}(\tilde{\boldsymbol{u}}) + \boldsymbol{\nabla} \cdot \boldsymbol{N}(\boldsymbol{u}) - \boldsymbol{\nabla} \cdot \bar{\boldsymbol{N}}(\boldsymbol{u}) = \boldsymbol{S} - \bar{\boldsymbol{S}}. \tag{44b}$$

Our method is predicated on the existence of a separation of time-scales for these two equations. Let τ_s denote the (slow) characteristic time on which the averages evolve, and τ_f, the (fast) characteristic time for the fluctuations in Eq. 44b to reach statistical steady state with fixed $\bar{\boldsymbol{u}}$. The method is designed to address the case $\tau_s/\tau_f \gg 1$, when a solution of Eq. 43 on the long time-scale becomes prohibitively expensive to compute.

Note that no formal separation of τ_s, τ_f scales has yet taken place in writing Eqs. 44. A pure separation-of-scales treatment would solve Eq. 44b treating the slow-scale quantities as constants in time, while Eq. 44a would be solved only using information from Eq. 44b taken in the limit $\tau_f \to \infty$. This approach is not computationally feasible (see discussion of N_f in Section 6.1.1), and our method was developed with more than one eye toward avoiding it.

Since we haven't formally removed the fast scales from Eq. 44a, the evolution of the averaged quantities will still contain fast-time-scale information. The nature of this information will depend on the average employed. For example, if the average were taken over an infinite ensemble of initial conditions, these averages would still

include various linear growth and nonlinear relaxation time-scales. If, on the other hand, we employ spatial averages over periodic dimensions (e.g., y in Section 6.2.2, or the two flux-surface angles in Section 6.3.1), then all fast time-scales remain, albeit at lower amplitudes. (Also, the resulting equation will be independent of the periodic variable(s), and thus Eq. 44a would have reduced dimensionality.) Finally, a time average can be employed, averaging over a finite time period that is long compared to the fastest time-scale (in plasma applications there are often multiple "fast" scales; e.g., in drift-wave turbulence, the real parts of the relevant frequencies are much greater than the growth rates). This average will reduce the high-frequency noise a great deal, but not completely, and the linear growth and nonlinear relaxation times will, of course, still be included.

In summary, $\bar{\boldsymbol{N}}$ has smooth time dependence at least as fast as τ_f and, depending on the statistical quality of the averaging operator, noise at the very fastest scales present in Eq. 43. Furthermore, any global turbulence/transport interactions inherent in Eq. 43 are fully retained in Eqs. 44.

In spite of the fact that Eq. 44a still contains fast-time-scale information, we would like to solve it on the slow time-scale. To do this, it is essential that the time-differencing of Eq. 44a be implicit—otherwise the short time-scales introduced by coupling to Eq. 44b (namely, $\bar{\boldsymbol{N}}$) will cause numerical instability [32]. Motivated by the expected diffusive nature of $\bar{\boldsymbol{N}}(\boldsymbol{u})$ (further discussed in Section 6.1.4), we rewrite each component of Eq. 44a as a diffusion equation:

$$\frac{\partial \bar{u}_i}{\partial t} = \sum_j \frac{\partial}{\partial x_j} D_{ji} \frac{\partial \bar{u}_i}{\partial x_j} - \boldsymbol{\nabla} \cdot \boldsymbol{L}_i(\bar{\boldsymbol{u}}) + \bar{S}_i, \tag{45}$$

where

$$D_{ji} \equiv -\frac{\hat{\boldsymbol{x}}_j \cdot \bar{\boldsymbol{N}}_i(\bar{\boldsymbol{u}} + \tilde{\boldsymbol{u}})}{\partial \bar{u}_i / \partial x_j} \,. \tag{46}$$

The set of $\bar{u}_i$ equations is solved iteratively for $\bar{\boldsymbol{u}}$ at the advanced time $t + \Delta t_s$. Consider the straightforward fully implicit algorithm

$$\bar{u}_i^{(k+1)} = \bar{u}_i^{\text{old}} + \Delta t_s \left[\sum_j \frac{\partial}{\partial x_j} D_{ji}^{(k)} \frac{\partial \bar{u}_i^{(k+1)}}{\partial x_j} - \boldsymbol{\nabla} \cdot \boldsymbol{L}_i(\bar{\boldsymbol{u}}^{(k+1)}) - \bar{S}_i^{(k)} \right], \tag{47}$$

where k is an iteration index, the superscript "old" indicates the previous (k-converged) time-step, and $D_{ji}^{(k)}$ is calculated from Eq. 46 using the latest $\bar{\boldsymbol{u}}$ and $\tilde{\boldsymbol{u}}$. This algorithm is iteratively unstable if D_{ji} depends on gradients of $\bar{\boldsymbol{u}}$, as it typically does for plasma turbulence applications. (This would be true even if the infinite time-average were employed for the bar-operator, or equivalently, if $D_{ji}(\bar{\boldsymbol{u}}, \partial_x \bar{\boldsymbol{u}})$ were known analytically.) To stabilize the iterations, we therefore replace D_{ji} with an iterate-averaged version, $\hat{\bar{D}}_{ji}$, and write our difference equation as

$$\bar{u}_i^{(k+1)} = \bar{u}_i^{\text{old}} + \Delta t_s \left[\sum_j \frac{\partial}{\partial x_j} \hat{\bar{D}}_{ji}^{(k)} \frac{\partial \bar{u}_i^{(k+1)}}{\partial x_j} - \boldsymbol{\nabla} \cdot \boldsymbol{L}_i(\bar{\boldsymbol{u}}^{(k+1)}) - \bar{S}_i^{(k)} \right]. \tag{48}$$

Stability with respect to iterations and the precise form of the iteration average (the $\hat{}$-operator) are discussed in Section 6.1.3. Note that iteration-averaging also helps reduce the noise fed to Eq. 44a through $\bar{\boldsymbol{N}}$.

After each iteration of Eq. 48, the values of $\bar{\boldsymbol{u}}$ used in Eq. 44b are updated to the latest estimates $\bar{\boldsymbol{u}}^{(k)}$, and the fluctuating equations are stepped one Δt_f time-step. After each of these steps, the latest $\tilde{\boldsymbol{u}}$ information is used to construct a new $\bar{\boldsymbol{N}}^{(k)}$ that feeds into the calculation of a new $\hat{\bar{\boldsymbol{D}}}^{(k)}$. Thus, the iterations on Eq. 48 take place simultaneously with the time-stepping of the modified Eq. 44b, which is using a new $\bar{\boldsymbol{u}}$, $\bar{\boldsymbol{u}}^{(k)}$, at each of its time-steps. If the $\bar{\boldsymbol{u}}^{(k)}$ converge smoothly (albeit slowly) to $\bar{\boldsymbol{u}}^{\infty}$, then this process should not take significantly more iterations (Δt_f time-steps) than would be required to calculate a single steady-state solution of Eq. 44b with fixed $\bar{\boldsymbol{u}}$.

6.1.1 Computational advantage

Define $N_s \equiv \tau_s/\Delta t_s$ and $N_f \equiv \tau_f/\Delta t_f$, the number of time-steps required to obtain solutions of Eqs. 44a and 44b for times of the order of their respective characteristic time-scales. If only a steady-state solution of Eq. 44a is sought, then N_s can be as small as 1. N_f, however, is a large number: The fluctuations involved have to be adequately resolved in time and at least a few periods are required for saturation of the spectrum. (Furthermore, there are often important linear or nonlinear frequencies in the problem much greater than τ_f^{-1}. For fixed profiles, plasma turbulence codes in fact typically need N_f on the order of a few thousand steps to saturate, and significantly more to achieve tight statistics requirements.)

The number of time-steps $\Delta t = \Delta t_f$ that Eq. 43 must be advanced to study profile evolution or to reach its statistical steady state is then $t/\Delta t_f = t/\tau_s \times \tau_s/\tau_f \times N_f$. Our coupled method, on the other hand, requires for each time-step Δt_s approximately N_f iterative steps of Eq. 44a and N_f time-steps of Eq. 44b, and so only $N_s N_f$ Δt_f-steps of Eq. 44b to study these same problems. Thus as long as $N_s \ll t/\tau_s \times \tau_s/\tau_f$, the method offers the possibility of a substantial improvement.

If Eq. 44a is of lower dimension than Eq. 44b, no new large matrix solves have been introduced. Nor is there any need for Jacobian derivatives, which would be difficult to come by for this problem. The only overhead is a requirement of N_f solves of Eq. 44a, rather than 1, per Δt_s. (However, in many applications one solve of the differenced Eq. 44a is much cheaper than one of Eq. 44b; for example, in tokamaks core turbulence is 3D whereas the averages, and so Eq. 44a, are 1D. In such cases the overhead is negligible.) Furthermore, our fully implicit differencing of Eq. 44a is unconditionally stable. The only limitations imposed on Δt_s are due to accuracy considerations. In particular, by letting $\Delta t_s \to \infty$ (so $N_s = 1$), we reach the saturated state in approximately N_f steps. That is, we find a saturated turbulent state self-consistent with the profiles driving it for a price comparable to that of one steady-state solution of Eq. 44b with fixed profiles.

6.1.2 Local versus global coupling

The method presented up to this point is the general, "global," version of the coupling problem. Often associated with the separation of time-scales in Eqs. 44, however, is a separation of spatial scales. There is an important limit, called the local approximation, in which the size of the turbulent eddies is sufficiently small that at each point the spectrum depends only on the local field averages and their first derivatives. Many turbulence codes (as well as analytical calculations) employ this approximation. In this case, the background drives, $\bar{u}_i$ in Eq. 44b, are fixed in both time and space, and the turbulence is homogeneous in the direction of the uniform gradient ("radial," in tokamak geometries). In order to utilize such codes we employ a local version of our coupling algorithm, launching a separate, independent turbulence simulation for each mesh-point of Eq. 44a, and manipulating each code's (spatially constant) $\bar{u}_i$ according to Eq. 46. The scheme provides an efficient tool for calculating profile evolution [although here there is no "direct" method (a version of Eq. 43) against which to compare CPU-time]. Any global effects inherent in the physics are of course lost in such calculations. Furthermore, if one's interest is an entire profile, the local approximation is generally more expensive in both CPU and storage requirements: Each local simulation still requires a large radial domain in order to converge with respect to "box size," i.e., to eliminate the effect of radial boundary conditions on the modes. The global implementation will be cheaper as long as $L_{\rm prof}/L_{\rm fluc} < N_{\rm prof}N_{\rm eddy}$, where $L_{\rm prof}$ and $L_{\rm fluc}$ are the background and fluctuation spatial scales, $N_{\rm prof}$ is the number of radial mesh points required for accurate fluxes from the local transport code, and $N_{\rm eddy}$ is the number of eddies in the radial direction per local turbulence simulation necessary for convergence. (This assumed a linear scaling with number of radial mesh points, typical of explicit time-differencing schemes, for the turbulence code.)

6.1.3 Iteration averaging

If the iterations converge, the method is stable in time for arbitrary Δt_s; however, stability of the iterations for a single Δt_s is a serious concern. For general $\bar{\boldsymbol{N}}(\boldsymbol{u})$, some type of average over prior iterates in constructing the D_{ji} is required for stability within a time-step. Here we consider two possible schemes. We temporarily suppress the subscripts on D.

One averaging scheme smoothly relaxes D to the average of the iterates starting from the previous converged answer. We choose a large number N (=100 by default) and define

$$\hat{\bar{D}}^{(k)} = \begin{cases} \max\{0,\ \frac{1}{N}\sum_{j=1}^{k} D^{(j)} + \frac{N-k}{N}\hat{\bar{D}}^{\rm old}\} & \text{if } k < N\ , \\ \max\{0,\ \frac{1}{k}\sum_{j=1}^{k} D^{(j)}\} & \text{if } k \geq N\ , \end{cases} \tag{49}$$

where $\hat{\bar{D}}^{\rm old}$ is the converged $\hat{\bar{D}}$ from the previous transport step. We do not check for convergence until $k \geq N$. Once $k \geq N$, Eq. 49 gives equal weight to all of the $D^{(k)}$.

Since the transport iterates correspond to turbulence-code time increments Δt_f, the early $D^{(k)}$ (low k), which are statistical fluctuations about some transient, may differ significantly from the $D^{(k)}$ with large k. The later ones are fluctuations about the desired mean $\hat{\bar{D}}$. Thus, it may be better to weigh them more heavily. To this end, we use an alternate scheme. For some large N with $k \geq N$ and $a = 1 - 1/N$, set

$$\hat{\bar{D}}^{(k)} \;=\; \max\{0,\, A_k \sum_{j=0}^{k} a^{k-j} D^{(j)}\}\;,\quad A_k = (1-a)/(1-a^{k+1})\;. \tag{50}$$

It is easy to show that Eq. 50 is a form of relaxation (or exponential averaging):

$$\hat{\bar{D}}^{(k)} \;=\; A_k D^{(k)} + (1 - A_k)\hat{\bar{D}}^{(k-1)}\;,$$

with $\lim_{k\to\infty} A_k = 1 - a$. Typically the iterations are begun as in Eqs. 49 for $k < N$.

In Ref. [32], a linear stability analysis of these two schemes is carried out assuming a functional dependence for the diffusion coefficient, $D = D_0(\partial_x n)^p n^q$. It is shown that the scheme in Eq. 49 is stable for $p > -1$, and Eq. 50 is stable provided that $p > -1$ and $N \geq \min\{1, (1+p)/2\}$. However, one should be cautious in applying this result to the actual coupled code algorithms since the analysis effectively assumes a converged turbulence calculation, so that D is uniquely a function of the present profile and not of the iteration history, whereas in practice one iterates the transport equation while simultaneously advancing the turbulence equations.

Finally, we remark that having defined these two averaging schemes, one could apply them instead to the flux and to the profiles, and then compute an average diffusion coefficient according to

$$\hat{\bar{D}}^{(k)} = -\hat{\bar{N}}^{(k)} / \widehat{\partial_x \bar{u}}^{(k)}\;.$$

Since the division is now by an averaged quantity, $\hat{\bar{D}}$ is less subject to large changing iterates occasioned by $\partial_x \bar{u}^{(k)}$ passing through zero. For cases with gradients that are either small or change sign in x, this can be important. The stability analyses referred to in the preceding paragraph apply without alteration to this modified definition of the average diffusion coefficient.

6.1.4 Further remarks

The numerical solution of the coupled Eqs. 44 that we have outlined is equivalent to a formal separation of scales, and so, apart from the usual errors associated with numerical derivatives, it introduces an additional error of order τ_f/τ_s in the time-dependence of $\bar{\boldsymbol{u}}$.

Robustness of the iteration-averaging schemes just described is a critical issue. In general, if $\boldsymbol{D}$ has strong dependence on $\boldsymbol{u}$ and/or its derivatives (i.e., if the flux does not obey a standard Fick's Law and so lead to a standard linear, possibly

spatially varying, diffusion equation), one has little reliable guidance from theory about what form it does take; finding the best way to rewrite Eq. 44a or improve the iteration averages to cure such instability should it emerge for a particular Eq. 43 of interest could then be difficult. In particular, the algorithm conceivably fails in global applications if the flux is non-local (i.e., if $\boldsymbol{D}$ at x depends on the whole profile, not just on the values of $\boldsymbol{u}$ and its derivatives at x); and, with our particular transport-equation solver, which requires $D > 0$, it will fail if the flux tries to run up hill. Also there are problems if $\boldsymbol{D}$ gets too big (i.e., because of local zero gradient). We have devised two extensions to handle the latter problems: One can assign a portion of $\bar{\boldsymbol{N}}$ to a convective term, similarly averaged, if $\hat{\bar{\boldsymbol{D}}}$ gets too big or goes negative; the convective term is assigned enough of the flux to keep $\hat{\bar{\boldsymbol{D}}}$ within bounds. Or, one can introduce an "adaptive alias": add a fake function (the "alias") to $\boldsymbol{u}$ so that the flux runs down the gradient of the sum (with a $\hat{\bar{\boldsymbol{D}}}$ that is within bounds). This alias is changed from iterate to iterate if needed. In our 1D applications, we find both work. These extensions are further discussed in Section 6.2 below.

Given a stable iteration algorithm, the rate of convergence can also be an issue. Since the turbulence simulation (the solution of Eq. 44b) typically takes a few hundred to a few thousand time-steps to compute a reasonably well averaged flux, there is no requirement of a rapid convergence rate for the iteration algorithm. It must not, however, significantly prolong saturation of the turbulence code, or the method loses its advantage. Feeding the turbulence code $\bar{\boldsymbol{u}}$'s that will be, to a certain extent, noisy can also be a problem.

Finally we remark briefly on code-coupling mechanics. A turbulence code whose dependent variable is $\boldsymbol{u}$ must be modified to solve Eq. 44b rather than Eq. 43. Many low-amplitude turbulence codes, however, are written for fluctuating quantities that are small in some sense; that is, "background" or "equilibrium" values have already been subtracted out. These fluctuating quantities are not necessarily the same as our $\tilde{\boldsymbol{u}}$, and indeed they may even evolve average components over time. (These average terms are removed in some codes.) Such averages are, however, properly part of our $\bar{\boldsymbol{u}}$, and, if $\bar{\tilde{\boldsymbol{u}}} = 0$ at $t = 0$, cannot in fact develop from Eq. 44b (since this equation averaged is $\partial\bar{\tilde{\boldsymbol{u}}}/\partial t = 0$). In order to employ our coupling algorithm, it is essential that a proper $\tilde{\boldsymbol{u}}$ be used, and that provision be made in Eq. 44b for $\bar{\boldsymbol{u}}$, normally a fixed input quantity, to be changed every Δt_f time-step.

6.2 Pilot project with two-dimensional turbulence simulations

In order to develop and explore the performance of our coupling techniques, we applied it first to a relatively simple turbulence model. This work is described in full detail in Ref. [32]. The plasma turbulence model used therein is a set of 2D (x, y) equations derived by Hasegawa and Wakatani [33] for modeling electrostatic drift-wave turbulence in a tokamak edge. The two field equations for a perturbed density

variable N_p and the z-component of the vorticity ζ are:

$$\begin{aligned} \partial_t \zeta + \nabla \cdot (\boldsymbol{v}\zeta) &= \alpha(\phi - N_p) + \mu \nabla^2 \zeta \\ \partial_t N_p + \nabla \cdot (\boldsymbol{v} N_p) &= \alpha(\phi - N_p) - \kappa \partial_y \phi + \nu \nabla^2 N_p \ . \end{aligned} \tag{51}$$

Here the electric-field-drift velocity $\boldsymbol{v} = \hat{\boldsymbol{z}} \times \nabla\phi$, ϕ is the electrostatic potential normalized to T_e/e, the vorticity ζ is related to ϕ via $\zeta = \nabla^2\phi$, and $\kappa \equiv -\partial_x \log N_b$. Time is in units of inverse ion cyclotron frequency $\Omega_{ci} = eB/m_i c$, and distance is in units of $\rho_s \equiv c_s/\Omega_{ci}$, where the sound speed $c_s^2 = T_e/m_i$. The density variable N_p is related to the total physical density N and a specified stationary background N_b by:

$$N(x,y,t) = N_b(x)\,(1 + N_p(x,y,t)) \ . \tag{52}$$

Many people have studied this system since it was originally developed by Hasegawa and Wakatani. We note here that the model follows from the two-fluid Braginskii equations [34] by assuming a uniform, constant magnetic field, using drifts for perpendicular velocities (low-frequency approximation), neglecting electron inertia, assuming a spatially and temporally constant electron temperature, neglecting the ion parallel velocity, and assuming $N_p \ll 1$ but $\nabla_\perp N_b \sim \nabla_\perp N_p$.

The fluctuating density N_p itself can have an average, and hence we separate it into its averaged and fluctuating parts:

$$N_p(x,y,t) = n(x,t) + \tilde{n}(x,y,t) \ , \tag{53}$$

with $\langle \tilde{n} \rangle = 0$, where $\langle \cdot \rangle$ will be used to denote the averaging operator for the remainder of this section.

In 2D, the coefficient $\alpha(x) = k_\parallel^2 T_e/m_e\nu_{ei} \propto 1/N_b$, where ν_{ei} is the electron-ion collision frequency and $k_\parallel$ is a chosen parallel wavenumber. As $\alpha \to \infty$, the departure of the electrons from a fully adiabatic response, $N_p - \phi$, goes to zero. The term proportional to κ in Eq. 51 is responsible for the growing modes, and is a source of energy (e.g., $\langle \phi^2 \rangle$) in the equations. For local calculations and ordinary profiles, $\kappa > 0$; in general, though, $\kappa(x)$ can be of either sign. The small diffusive terms proportional to μ and ν represent very fine-scale dissipative processes and are customarily included in simulations in order to remove energy from the highest wave numbers, where it arrives via mode-coupling. Without the μ and ν terms, energy accumulates at high wave numbers; these terms are necessary to achieve (fluctuating) steady states.

Periodic boundary conditions in y are applied to the system, and averages are defined by

$$\langle f \rangle \equiv \frac{1}{L_y} \int_0^{L_y} dy\, f \ . \tag{54}$$

6.2.1 Local coupling

In a local implementation of the Hasegawa-Wakatani equations, the transport code evolves the single field N_b, according to

$$\partial_t N_b + \partial_x \Gamma = S \ , \tag{55}$$

with a specified $S(x)$. (Recall that our starting Eqs. 51 were derived assuming a stationary N_b, leaving the transport time-scale evolution unspecified.) At each point that the flux Γ is needed, an independent homogeneous turbulence calculation that knows *only* about the local value and gradient of N_b is performed. The x-domains of the transport and turbulence simulations thus differ, and to avoid confusion we use X and Y for the latter. In a given simulation of the turbulent Eqs. 51, α and κ are now spatial constants; boundary conditions are periodic in X as well as Y; solutions are translationally invariant, i.e., the turbulence is homogeneous; and averages can therefore be taken over X and Y (which is helpful in reducing noise):

$$\langle f \rangle_{_L} \equiv \frac{1}{L_X L_Y} \int_0^{L_Y} dY \int_0^{L_X} dX\, f \qquad \text{(local approximation)} \; . \tag{56}$$

Performing an X-Y average on Eqs. 51 shows that for the local version (uniform $\alpha,\ \kappa$), due to periodicity in X, if $\langle n \rangle_{_L}$ and $\langle \phi \rangle_{_L}$ vanish at $t = 0$, they remain zero (though the instantaneous n and $\langle \phi \rangle$ are nonzero); then $\langle \zeta \rangle_{_L}$ must also vanish. Thus, the local version of Eqs. 51 provides a suitable candidate for coupling, as discussed in Section 6.1.

We use the HAWC code [35] to solve Eqs. 51 in this constant-background-gradient limit. HAWC makes use of a scale-invariance transformation that exists within the local approximation: If we transform to new variables $\hat{t} = \kappa t,\ \hat{x} = x,\ \hat{\phi} = \phi/\kappa,\ \hat{N}_p = N_p/\kappa$, and $\hat{\alpha} = \alpha/\kappa$, then κ drops out of the equations, and there is only a 1-parameter ($\hat{\alpha}$) family of solutions to Eqs. 51. So we proceed with Eqs. 51 written in scaled variables, i.e., with $\kappa = 1$.

Each HAWC problem is given random but small initial conditions, and each problem has its own constant (in space) value of $\hat{\alpha}$. At each iteration with transport (each HAWC time-step Δt_f), α and κ are recomputed from N_b on the transport grid $[\alpha = \alpha_0(x) N_b(x,0)/N_b]$ and all the $\hat{\alpha}$'s are updated. It is the x-dependence of α at $t = 0$, together with S and the boundary conditions on N_b, that distinguishes one steady-state solution from another.

The flux Γ actually used in the transport equation is obtained from the space average $-c_s N_b \langle N_p \partial_Y \phi \rangle_{_L} = -c_s N_b \kappa^2 \langle \hat{N}_p \partial_Y \hat{\phi} \rangle_{_L}$ of the HAWC solutions. Defining $D_H = \langle \hat{n} \partial_Y \hat{\phi} \rangle_{_L}$, we use

$$\Gamma^{(k)} = -c_s \frac{\hat{\bar{D}}_H^{(k-1)}}{\hat{\bar{\ell}}_n^{(k-1)}} \frac{\partial N_b^{(k)}}{\partial x} \; , \tag{57}$$

where the $\hat{}$bars on the right-hand side refer to the iteration-averaging scheme, Eq. 49, and $\ell_n \equiv -\kappa^{-1}$.

At each time-step Δt_s, sufficient iterations are taken such that all the HAWC's as well as Eq. 48 (i.e., the differencing of Eq. 55) have converged.

In Fig. 5 we present the results of a coupled calculation. The source is a step function,

$$S(x) = \begin{cases} S_0 & x < \delta \\ 0 & \text{otherwise} \end{cases} \quad ; \quad S_0 = 1, \ \ \delta = 0.1 \; .$$

The initial density $N_b(x,0) = 0.1$. The boundary conditions are $\partial_x N_b = 0$ on the left-hand side, and $N_b = .001$ on the right. (Note that the left-hand boundary points, at $x = 0$, are not plotted in the figure.)

In the figure, N_b is plotted vs. grid number $(x/\Delta x)$ at various times. Initially, we expect the evolution to consist of a density increase for $x < 0.1$ (because of the source) and a decay near $x = 1$ (as the effect of the small boundary condition propagates in). For a time, the two regions (of decreasing and increasing density) are independent of each other, since they are separated by a central region where $\partial_x N_b = 0$, so that the turbulence drives, and thus D_H, vanish. By the time the effect of the source on the initial left-hand-side density has just become visible in the figure, the cooling wave has already (at $t \approx 0.016$) reached and begun to interact with the source region. Eventually a steady state is reached, curves N = 0–4 in the figure.

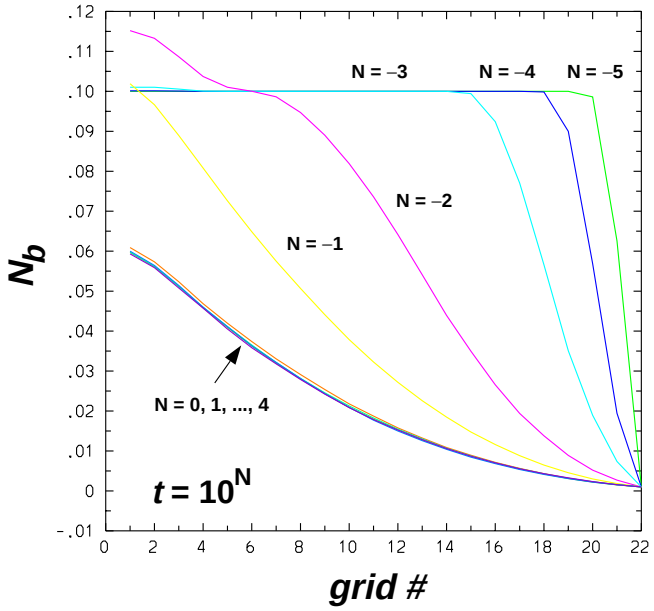

Figure 5: Average density vs. $x/\Delta x$ at various times for local turbulence simulations coupled through a diffusion equation. A separate copy of the HAWC code is run at each grid-point on the x-mesh.

In the interest of speed, the HAWC runs were done with a limited 32×32 grid. At this same resolution, a series of (uncoupled) HAWC runs was performed, and an analytic fit $D_A(\hat{\alpha})$ was found that well approximates the (time- or ensemble-) averaged HAWC fluxes $\bar{D}_H(\hat{\alpha})$. As a test of our code-coupling mechanics and of convergence-control in the presence of noisy data, the same transport problem was solved as with the coupled HAWC's, but with $\hat{\bar{D}}_H$ replaced by D_A in Eq. 57. The resulting $N_b(x,t)$ was nearly identical to the run coupled to the turbulence simulations.

In applying our coupling ideas to this local problem, little modification of the basic scheme was necessary. The transport solver we employed requires a positive diffusion coefficient (thus the "max" function in Eqs. 49 and 50). This issue is a concern (and

is treated in the next section), but here, although we did see realizations for some $\hat{\alpha}$ and k for which $D_H^{(k)}(\hat{\alpha}) \leq 0$, the average $\hat{\bar{D}}_H^{(k)}(\hat{\alpha})$ (for the $\hat{\alpha}$ considered) remained positive. One adjustment was made in the interface between the transport solver and HAWC: The $\hat{\alpha}$ handed to HAWC were modified according to

$$\hat{\alpha}_H = \max\{\, 10^{-2},\, \min(\, 1,\, \hat{\alpha})\} \;,$$

in order to avoid wild fluxes and/or tiny time-steps in HAWC. Since the final profiles fell within this range, this had no effect on the steady-state transport solutions we obtained.

Further results, illustrating the convergence properties of the local-coupling algorithm, and a discussion and tests of the numerical transport-equation solver, are presented in [32].

6.2.2 Global coupling

In order to use the Hasegawa-Wakatani equations as a test-bed for a 1-field global transport/turbulence coupling application, it was necessary to develop an appropriate version of the non-local Eqs. 51. As given, these equations lead to the evolution of averaged quantities $\langle N_p \rangle$ and $\langle \zeta \rangle$. (This is in spite of the implication in the derivation [33] that N_p and ζ are perturbations from averaged quantities.) We work instead with a modified version of Eqs. 51 in which there is no evolution of $\langle \zeta \rangle$, but $\langle N_p \rangle$ still evolves. This modified system provides a numerical test of both the accuracy and the efficiency gain of our coupling algorithm, as it can be solved in two ways: (1) direct solution using the global turbulence code HAWCX [36]; and (2) a coupled solution, in which Eq. 51 is further modified so that neither $\langle N_p \rangle$ nor $\langle \zeta \rangle$ evolves, and an equation for the evolution of $\langle N_p \rangle$ is solved by the transport code (HAWCX is again used to solve the modified fluctuating equations). In each case, in contrast to the local calculation, only a single run of the turbulence code is required.

Analogously to Eq. 53, we introduce the averaged and fluctuating parts of the potential and velocity:

$$\begin{aligned} \phi(x,y,t) &= \langle \phi \rangle + \tilde{\phi} \\ \zeta(x,y,t) &= \langle \zeta \rangle + \tilde{\zeta} \;, \end{aligned}$$

where the averages $\langle \cdot \rangle$, defined in Eq. 54, are functions of x and t. Similarly, Eqs. 51 are themselves separated into two sets, one for the averaged variables n, $\langle \phi \rangle$, and $\langle \zeta \rangle$, and another for the fluctuating variables $\tilde{n}$, $\tilde{\phi}$, and $\tilde{\zeta}$. The two systems are coupled (mathematically) in a manner similar to that of the local case: Fluxes of the averaged variables are determined by $\tilde{n}$ and $\tilde{\phi}$ while certain coefficients of the fluctuating system depend on gradients of n.

In Eqs. 51, we replace the density diffusion term by $\partial_x[\nu \partial_x (N_p - n)]$, where $\nu(x)$ is defined below, and we set $\mu = 0$. (The HAWCX code has intrinsic dissipation in its vorticity equation due to its differencing scheme.) Our modified version of Eq. 51,

which does not evolve $\langle\zeta\rangle$, is then given by subtracting the averaged ζ equation from the ζ equation itself and substituting $\tilde{\zeta} \to \zeta$, to yield

$$\partial_t \zeta + \nabla \cdot (\boldsymbol{v}\zeta) - \partial_x \langle v_x \zeta \rangle = \alpha(\phi - \tilde{n}) \tag{58}$$

$$\partial_t N_p + \nabla \cdot (\boldsymbol{v} N_p) = \alpha(\phi - N_p) - \kappa \partial_y \phi + \partial_x [\nu \partial_x (N_p - n)] \ . \tag{59}$$

In writing Eq. 58, we assumed the initial condition $\langle\phi\rangle = 0$; Eq. 58 implies that $\langle\zeta\rangle$ and $\langle\phi\rangle$ are both zero for all time, and then $\boldsymbol{v} = \tilde{\boldsymbol{v}}$.

If the equation for N_p is averaged, one obtains

$$\partial_t n + \partial_x \Gamma = -\alpha n \ , \quad \text{with} \ \ \Gamma = -\langle \tilde{n} \, \partial_y \phi \rangle \ . \tag{60}$$

Subtracting this from the equation for N_p yields the evolution equation for $\tilde{n}$:

$$\partial_t \tilde{n} + \nabla \cdot (\boldsymbol{v}\tilde{n}) + \partial_x \langle \tilde{n} \, \partial_y \phi \rangle = \alpha(\phi - \tilde{n}) - (\kappa - \partial_x n)\partial_y \phi + \partial_x (\nu \partial_x \tilde{n}) \ . \tag{61}$$

In Eq. 60, the flux is an average of the product of fluctuating variables. In Eq. 61, $\partial_x n$ modifies the coefficient responsible for the growth of the turbulence, i.e.,

$$\kappa \to \kappa' \equiv \kappa - \partial_x n \ . \tag{62}$$

As in the local-coupling implementation, we set $\alpha \propto 1/N_b$; in the global case, though, α is the *only* input (apart from the boundary conditions) that determines the steady-state profiles, and N_b does not change in time.

In HAWCX, the boundary conditions are

$$\zeta, \, \phi, \, \tilde{n} = 0 \text{ at } \ x = 0, \, L_x \ ,$$

and periodic in y. It was found in running coupled or uncoupled problems that convergence was impeded or prevented by behavior driven by the boundaries. In particular, without artificial damping (ν), Eqs. 58–59 plus these boundary conditions yield zero flux at the boundary. The x-dependence in the dissipative function ν is introduced to avoid complications at the boundaries. Its effect is to add diffusion at the endpoints but not disturb the evolution in the interior. In the examples,

$$\nu(x) = \nu_0 [\exp(-x^2/\Delta_\nu^2) + \exp(-(x - L_x)^2/\Delta_\nu^2)] \ . \tag{63}$$

Equation 60 is the desired transport equation. Its flux cannot be assumed to be of opposite sign to ∇n, either initially or at saturation. For small times, $n \approx 0$, and Γ tends to move down the gradient of N_b, which, over a large portion of the domain, is not in the same direction as the gradient of n. As a further complication, inasmuch as n is not a physical density it can take on both positive and negative values. The transport solver, on the other hand, requires a $D > 0$ and is guaranteed to return $n > 0$. We have devised two techniques, about equally successful, to handle the difficulties.

The first approach treats a portion of the flux as convective, rather than diffusive. The flux at iteration $k+1$ of the transport equation is written as

$$\Gamma^{(k+1)} = -\theta \hat{\bar{D}}^{(k)} \partial_x n^{(k+1)} + (1-\theta) \hat{\bar{c}}^{(k)} n^{(k+1)} \,, \tag{64}$$

where $D = -\Gamma/\partial_x n$ and $c = \Gamma/n$ (cf. Eq. 46). A simple prescription for setting θ is:

$$\theta = \begin{cases} 0 \,, & \text{if } D < D_{min} \quad \text{or} \quad D > D_{max} \,, \\ (D_{max} - D)/(D_{max} - D_{min}) \,, & \text{otherwise} \,, \end{cases}$$

where D_{min} and D_{max} are two non-negative numbers limiting the diffusion. Clearly, $0 \leq \theta \leq 1$. It is also easy to let θ have a non-linear dependence on D or to prescribe a range for D for which $\theta = 1$; these choices all work for our application.

The second approach employs a (time-dependent) change of variables. An "adaptive alias" density n_{al} is added to n in such a way that the flux Γ runs down the hill of the "total density" $n_{tot} = n + n_{al}$ with a diffusion coefficient $D = -\Gamma/n_{tot}$ that is within prescribed bounds. In practice, it was found that the particular values of the bounds and the form of the mapping employed in calculating n_{al} had little impact on the convergence rate of the coupling algorithm.

The details of the modification to the differencing due to the convective term, which is treated in a manner equivalent to upwinding with care taken to retain the good stability and positivity properties of the solver, and of our procedure for determining n_{al}, are given in Ref. [32].

We now present results of our coupled global calculations. As mentioned above, we perform each calculation two ways: stand-alone HAWCX runs (S) solve Eqs. 58–59, retaining whatever average N_p $(= n)$ develops; in the coupled calculations (C), HAWCX solves Eqs. 58 and 61, and does not evolve n; instead, it uses n as evolved by Eq. 60. Each iteration of the transport requires a time advancement of Eqs. 58 and 61 using a κ' modified by the transport according to Eq. 62.

We look for steady-state solutions, and compare the saturated value of n obtained the two ways; we also compare the number of time-steps for the S runs with the number of iterations (= number of HAWCX time-advances) for the C runs. To check for steady state in the S runs, HAWCX is run for batches of 1000 cycles each, and the batch-averaged profiles are examined. It can be difficult to determine when HAWCX has saturated, since there are often waves with low frequencies traveling across the domain, even after saturation.

In our first example, we set $\alpha(x) = 10^{-4}/N_b(x)$. A very simple variable change (similar in concept to a fixed alias) proved adequate for this problem: The transport Eq. 60 was used to solve for a new variable $N^+ \equiv \langle N(x,y,t) \rangle$, from which n and then κ' can be constructed (recall Eq. 52). In Eq. 63, we set $\nu_0 = 5$ and $\Delta_\nu = 5L_x/N_x$, so that 5 [10] grid points away from the boundary, ν decreases by a factor of e [e^4]. We apply the iteration-averaging scheme Eq. 50 with $N = 500$. In Fig. 6 we plot the total density N^+ vs. $x/\Delta x$ from the coupled run at $t = 0$, and after 200 and 2200 iterations. It is clear that convergence with the coupled scheme (C) is reached within 200 iteration

cycles. The S run, on the other hand, took well over 10,000 time-steps to reach this same state. In Ref. [32], further plots are presented for this case, exhibiting the approach to steady state of the S run, which was particularly slow in the middle of the domain. We believe the time-scale separation for this problem, and therefore the potential gain in efficiency for the coupled approach, to depend on the ratio of the gradient scale of N_b to the normalizing distance ρ_s (defined below Eq. 51).

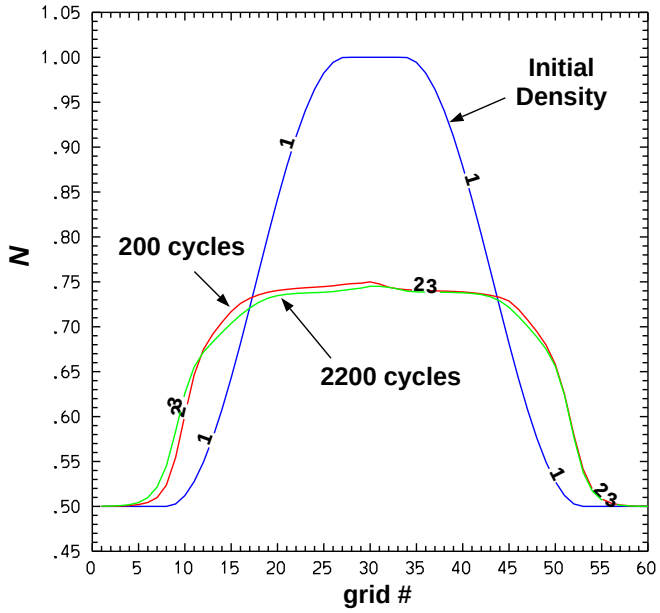

Figure 6: Average total density vs. $x/\Delta x$ at various iteration cycles of a steady-state (one large time-step Δt_s), global, coupled transport/turbulence calculation. The uncoupled global simulation code took well over 10,000 steps to reach this answer.

Finally we show an example of strongly non-local transport. Formally, Eqs. 51 follow from a multiple-length-scale expansion of the Braginskii equations, with the parametric dependence on the (long) equilibrium spatial scale retained. In our numerical example, however, we choose α such that the equilibrium and fluctuation scale lengths are comparable and $N_p \sim 1$, in violation of the assumptions under which the equation set correctly represents the physics, in order to test how well the coupling algorithm works on a strongly non-local problem.

For this example we chose $\alpha(x) = 2.5 \cdot 10^{-4}/N_b(x)$, with N_b shown as the initial N in Fig. 7. To keep the dependent transport-equation variable positive, we used $N^+ \equiv \langle N(x,y,t)\rangle + 1$; and to keep the diffusion coefficient for this variable positive, we introduced a convective term as in Eq. 64.

The figure shows that the coupled run has converged fairly well within 8200 iterations, and gives a measure of the statistical fluctuations associated with the parameters of our various averaging operations. The uncoupled HAWCX run was just approaching, after 30,000 time-steps, a saturated profile within the C-profiles shown. The figure also displays the characteristic eddy size of the HAWCX (C) saturated

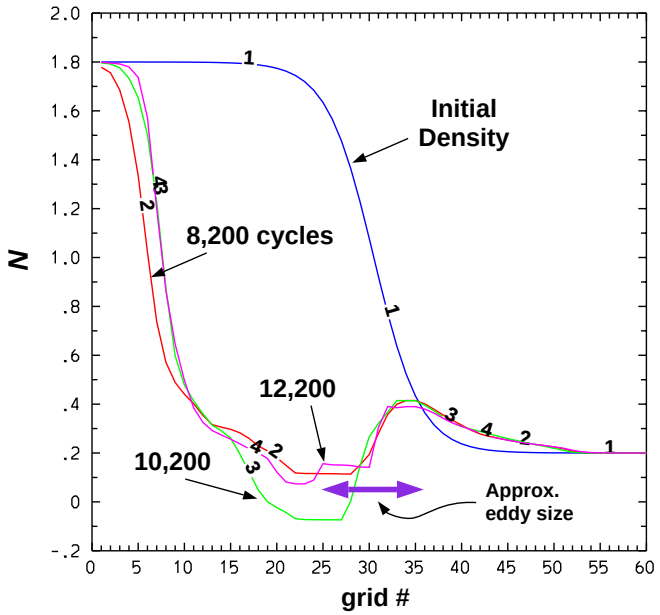

Figure 7: Average total density vs. x at various iteration cycles of a steady-state global coupled transport/turbulence calculation with strongly non-local transport. The uncoupled simulation was just approaching saturation after 30,000 steps.

solution; it is comparable to the average density scale-length, as is characteristic of non-local transport regimes. This is a hard problem. The average flux (not shown) is positive everywhere, so that in the vicinity of grid-cell #30 in the figure, the flux flows up the gradient of the density N^+. Nevertheless, our method is robust enough to find the saturated value. As shown in [32], the adaptive alias scheme was even more efficient for this problem, obtaining a reasonably well converged solution within ~ 2500 iterations.

6.3 3D tokamak turbulence codes coupled to Corsica

Our motivating application for the development of techniques to calculate the evolution, self-consistently and efficiently, of macroscopic profiles under the influence of turbulence is the problem of transport in toroidal magnetic-fusion plasmas. Microturbulence in tokamaks has long been considered to have a significant impact on the evolution of the averaged fields, and in particular to be the primary mechanism for transport of energy from the interior of the plasma to the surrounding structures. The turbulent fluctuations in the plasma density, temperature, electrostatic potential, etc., are themselves driven by gradients of the averages of these fields. Quantitative modeling of this turbulence-transport system is thus essential to predict the performance of future large machines.

The GRYFFIN code is currently the leading simulation code in the international community for studying tokamak core turbulence. This code was developed as a

collaboration between scientists at the Institute for Fusion Studies (IFS) and at the Princeton Plasma Physics Laboratory (PPPL). The bulk of our effort, described in the following sections, was addressed to coupling CORSICA's transport module to GRYFFIN.

We also initiated an effort to couple to a global fluid code for studying tokamak edge turbulence driven by electrostatic resistive ballooning instabilities (modes driven unstable by magnetic field-line curvature in the presence of pressure gradients). This code, FLUEDGE, proceeded from an extension to three dimensions [37] of the 2D code HAWCX, used in the global-coupling pilot project described in the previous section, and was under development for application to core geometries (HAWCX was primarily addressed to open-field-line instabilities, though it includes an outer core region; modifications were necessary to run with closed magnetic flux surfaces only). The CORSICA 3 code-coupling mechanics were generalized to handle coupling to a global code, and the interface routines to couple to FLUEDGE completed and tested. No coupled calculations were attempted, however, as FLUEDGE itself was not yet performing well enough (although now it is) to proceed before the CORSICA 3 project ended.

6.3.1 Gyrofluid turbulence model

The term "gyrofluid" refers to a schema for closing the set of moments of the fundamental magnetized plasma kinetic equations to develop plasma fluid equations valid at low frequency ($\partial/\partial t \ll \Omega_{ci}$), weak collisionality, and low amplitudes for time-dependent phenomena on the fastest scale retained (shear Alfvén and drift-wave time-scales). Associated with the low-amplitude assumption is the approximation of short fluctuation cf. background-gradient scale-lengths in the direction perpendicular to the magnetic field. The basic idea is to expand the kinetic response about the low-velocity limit and to set the coefficients of the resultant (frequency- and wave-number-dependent) terms in the moment equations by matching to a best fit with local linear kinetic theory. Thus the resultant equation set is radially local.

The most fully developed version of this approach is presented in detail by Beer [7], and forms the basis for the six-field (ten complex closure coefficients), electrostatic, fluid-equation set at the heart of the GRYFFIN code. The code itself is also described in Ref. [7]. The volume simulated is a single magnetic flux-tube followed sufficiently long that an entire flux surface is sampled. Background gradients, magnetic geometric parameters, and other field and scalar quantities, are fixed input parameters to the code. A single nonvanishing magnetic flux-surface average, the ion heat flux across the surface, driven by turbulent fluctuations of the fields, is obtained from the solutions at saturation.

Results of exercising this model and comparing against present tokamak experiments have received wide attention since first presented by Kotschenreuther [38]. To model or compare against experiment, a transport calculation of the radial cross-section of the entire tokamak is required. The IFS/PPPL authors proceed by as-

sembling a database of GRYFFIN runs, parameterizing the results (in a well chosen way: as a multiplier on a cheaper, linear, fully kinetic calculation which is intended to capture the bulk of the parametric variation of χ_i), and using the resulting analytic formula in conjunction with the kinetic calculation in their transport calculation. Since Ref. [38]'s appearance, there have been significant additions to the physics in the model (not available in the version of the code offered to us), and parameter space has grown large. Since GRYFFIN simulations are expensive, construction of the database required for the IFS/PPPL approach has rapidly become infeasible (it has not been redone with the better physics included or new parameters added to the fit), and there is significant debate concerning how much of the gyrofluid turbulence physics, as opposed to the simple nonlinear "mixing length" model physics constructed from the linear kinetic code, is making its way into the advertised "first-principles" transport results.

Our coupled turbulence/transport technique, although it still requires a number of GRYFFIN runs equal to the number of radial mesh points, requires many fewer runs than the construction of a credible database, and guarantees that the transport profiles are self-consistent with the turbulence.

6.3.2 Code-coupling issues

Since GRYFFIN calculations themselves are computationally expensive, our first goal was a single, converged, coupled, infinite transport-time-step; i.e., a steady-state with transport profiles self-consistent with the turbulence, carried out at low numerical resolution in the interests of speed. In the CORSICA-to-GRYFFIN direction, a fairly large number of time-dependent parameters are communicated: density, T_e/T_i, $\nabla T_i/\nabla n_i$,, ratio of magnetic curvature to density scale length, impurity concentration, collisional parameters, etc.; equilibrium geometry parameters (local aspect-ratio, shear, "safety factor" q, etc.) are made as self-consistent as possible (the version of GRYFFIN made available to us assumed a variable-but-large-aspect-ratio device with circular flux-surfaces) and kept fixed throughout the run; the equilibrium geometry was also kept fixed for the transport equations. Because MFE plasmas have some highly nonlinear source terms, characteristic source profiles were used and kept frozen. From GRYFFIN, a single number, the ion heat flux, is sent to and used by CORSICA.

Communications between CORSICA and GRYFFIN were handled using PVM, with all codes running on the NERSC C90.

The basic infrastructure for implementing coupling ("wiring" into the CORSICA 1 time-step loop; various convective/diffusive split options; iteration-averaging schemes; etc.) was tested using CORSICA 1 transport models as "black-box" electron and ion heat-flux providers. Coupled problems with fixed sources converged to the right answer (i.e., the same answer as obtained by CORSICA 1 run in its usual manner).

In support of the GRYFFIN/CORSICA coupling effort, a number enhancements to CORSICA itself were made:

- A new relaxation scheme was added to enable large transport time-steps, in order to be able to calculate steady-state profiles in one time-step.

- The IFS/PPPL algebraic ion-thermal-diffusivity model [38] (in which the strongly varying "mixing length" factor discussed above is also parameterized as a simple algebraic expression) was added to CORSICA 1 as one of its standard transport-model options. In CORSICA 3 applications, this is used to set up the initial guess for the profiles for coupled calculations, and to provide a comparison with the steady-state coupled answer.

- Options were introduced to enable switching transport models over portions of the radial grid. The turbulent transport drives simulated by GRYFFIN are significantly reduced or vanish both near the magnetic axis and as the edge is approached. If GRYFFIN were used throughout the radial domain, unrealistic profiles would result, so other transport models were used in those regions.

6.3.3 Code-coupling: GRYFFIN *development*

Although coupling to a turbulence code as has been described is fairly non invasive, and in our 2D application was straightforward, some GRYFFIN-specific development work was required. These tasks included:

- Modifications to enable changing input parameters on the fly. It was necessary to identify all pre-calculations based upon the inputs that we were interested in having self-consistent with our time- (or iteration-) dependent profiles, move them inside the GRYFFIN time-stepping loop as mandated by our coupling procedure, and devise and carry out checks that GRYFFIN calculates correctly this way (i.e., finds the same growth rates and saturated states as if the last values of the "inputs" were the given, fixed inputs to a stand-alone original GRYFFIN run).

- Courant-condition time-step control. GRYFFIN is an explicit code, subject to conditions on its time-step to avoid numerical instability. Its time-step control algorithm treated only the convective nonlinearities, assuming the users' input Δt_f did not violate the stability conditions associated with the linear terms (whose coefficients are fixed in time for fixed inputs). To prevent instability arising from these terms in the course of a coupled problem, it was necessary to install dynamic time-step control based on all the linear conditions.

- I/O modifications to handle running multiple copies of GRYFFIN in a single directory.

- Port to the T3D. When running numerically well resolved problems, both the number of GRYFFIN copies required and the expense of each goes up. In order

to keep such coupled problems in the realm of practicality, a measure of parallelization is indicated. To keep the port from becoming a massive project on its own, GRYFFIN itself was not parallelized. The various copies each run on a single processor of the T3D, and interact with CORSICA running on a workstation. The port entailed:

- Code changes in GRYFFIN to accommodate compiler differences.
- Development of a library of routines supplied by the system only on the C90 (Bessel functions, random numbers, etc.).
- Development of MPP-versions (as opposed to the network versions used on the C90) of selected PVM operations, which greatly enhances otherwise terrible performance on the T3D.
- Work (unfinished) on the T3D system-supplied FFT's, which dominate the running time (as they should) and had very poor performance at the resolutions of interest to us.

The port to the T3D was completed, and successful test runs of the coupling mechanics in a distributed environment were carried out. However, the very slow execution time (due to the FFT problems) precluded this as a useful mode of operation for general work on the coupling.

6.3.4 Results and status

Our initial experiments coupled a band of 8 flux surfaces towards the outer portion of an ITER-like discharge to 8 GRYFFIN's. Results were encouraging: Convergence was achieved at a rate essentially the same as our test cases with "black-box" analytic models, even though (it turned out) the prepared initial condition was far from being self-consistent with GRYFFIN turbulent diffusion rates.

Extending this success to the full-core (excluding a few surfaces at the axis and edge, as explained above), however, proved difficult. Although there was significant variation of parameters across the coupled band problem, none of the GRYFFIN's there were in the strongly unstable regime characteristic of the inner core. In the latter case, strong instability results in very large transport, flattening the profiles, which tends to turn off the instability. Thus profiles hover near marginal stability, while the fluxes remain very sensitive functions, with deviations in profiles producing large changes. As a result, with initial conditions only moderately far from the answer, full-core coupled runs crash, due to single GRYFFIN's crashing; field amplitudes at some point typically develop explosive instability.

Another source of difficulty derives from the nature of local coupling. Intermediate iteration cycles as well as converged problems can have extrema in either the density or temperature profile. For normal diffusive problems, this means a flux of opposite sign (and if it is a density extremum, the magnitude of the predicted GRYFFIN flux is large). A global turbulence calculation would keep itself coherent across the extremum

and experience no problem; but in a local calculation, in the vicinity of an extremum, a very large change in the turbulence problem (but very small change in the transport profile) can be requested from iteration to iteration.

To treat these difficulties, a collection of checks, controls, caps, and methods for dealing with troublesome intermediate-to-solution contingent cases and extrema in profiles (as our first GRYFFIN steady-state solution seems to have, although the initial guess, a steady-state solution with the same sources and the IFS/PPPL model, did not) has been developed. Each has gotten us a bit further before a GRYFFIN crashes, and we have now very nearly succeeded in obtaining a converged solution. In our best case (as the project concluded), the most unstable portions in radius are well converged and yield a very flat temperature profile; the most stable region (near the edge) is near-converged throughout the run; and the intervening portion was heading towards a converged solution when the run blew up. The electron heat-flux, treated via the coupling algorithm but with an analytic model χ_e, was well behaved throughout the run, even though, due to the very large central χ_i, there was a significant difference between T_e and T_i.

Were this project to be revived, there remain a few important coupling issues (apart from the need for ever better turbulence codes) to be addressed:

- Multiple-field coupling. To do couplings for physical problems of the greatest interest (including the application to tokamaks), it remains to address the coupling of multiple fields. Here off-diagonal contributions of the gradients to the fluxes, which can be significant and of either sign, and whose analytic form in general will not be known, can have a serious impact on the stability of the method. Stability of the coupling algorithm near critical gradients (that is, values of the gradients at which the turbulence-driven diffusion coefficient vanishes) is also a potential issue. We have carried out initial analysis and some testing of various algorithms in this regard. (In fact the electron heat-flux in the full-core run described above was treated as if anomalous, i.e., via the coupling algorithm, but with an analytic model χ_e. Although normally the large electron/ion collisional drag term in the energy equations tightly couples T_i and T_e, in this case, due to the turbulence-driven very large central χ_i, there was a significant difference between T_e and T_i. That the electron heat-flux was nevertheless well behaved throughout the run can thus be taken as a first modest two-field success.) Although turbulence-driven particle transport is at present much less well established with respect to experiment than energy transport, various models exist, from simple analytic prescriptions to full non-adiabatic-electron simulations (as in the latest GRYFFIN version). Thus, after further attention to two-field algorithm development, we would like to demonstrate 2-field coupling of energy and particle transport in a realistic tokamak context. A third very important field to treat self-consistently in tokamaks is the toroidal momentum; shear in the $\boldsymbol{E}\times\boldsymbol{B}$ velocity, including flows generated by the instabilities, has a large effect on both linear and nonlinear gyrofluid instability.

- Coupling to a global turbulence code should alleviate many difficulties experienced with the local calculation, and it should be both cheaper, as explained in Section 6.1.2, and have the better physics. It remains to complete a demonstration with FLUEDGE.
- The coupling control must be made adequately robust (enough so that new cases do not in general require more fixes).
- The transport-equation sources must be included in the iteration scheme, to make them self-consistent with the profiles. So far they are frozen.
- A fully coupled core turbulence simulation should of course also couple the equilibrium magnetic geometry into the calculation.

7 Applications

7.1 Tokamak core simulations

7.1.1 Active control in ITER

Modern tokamak designs, such as the International Thermonuclear Experimental Reactor (ITER), have highly elongated plasmas (i.e. plasmas that are taller than they are wide). Such designs are chosen because highly elongated plasmas can carry more current, and this improves transport and increases the maximum pressure that can be confined. Unlike nearly-circular plasmas, elongated plasmas are unstable to vertical displacements. Fortunately, this instability is fairly slow and can be controlled by the system of poloidal field coils that is used to shape and induce current in the plasma.

One of the research applications of the CORSICA code has been to help design the shape and position control system for ITER.[8] This control system monitors the position of six points, known as *gaps*, along the separatrix and feeds back on the magnet voltages to drive these points to their desired position. The points that are monitored are the two points where the separatrix intersects the divertor targets (the *strike points*), the outboard mid-plane, halfway up from the mid-plane to the top of the plasma, the top of the plasma, and the inboard mid-plane. The ITER design specifies that the gaps must be controlled to within 1-10 cm of their desired values on a time-scale of 5-20 seconds. Figure 8 illustrates the ITER plasma and one of the proposed coil sets. The complete coil set is used for control, including the central solenoid. Thus the control system must control the plasma shape and current, and it must also control the vertical instability.

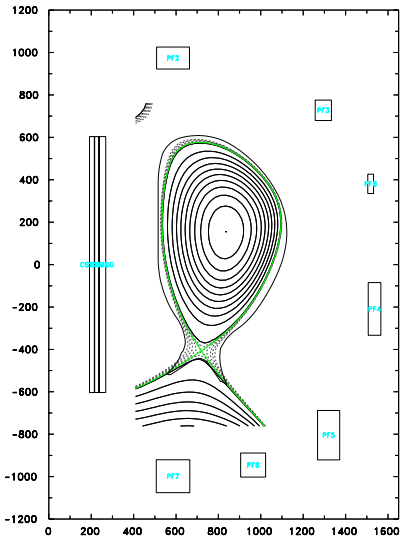

Figure 8: The ITER plasma, first wall, and PF coil set.

The control algorithm employed is a "robust" multi-variable shape and stability controller designed by General Atomics (GA) [39, 40]. This algorithm uses the following mathematical representation:

$$\mathbf{V} = \mathbf{G}_{\mathbf{X}} \cdot \mathbf{X} + \left(\mathbf{G}_p + \mathbf{G}_d \frac{d}{dt} \right) \cdot \mathbf{Y} \tag{65}$$

$$\frac{d\mathbf{X}}{dt} = \mathbf{M} \cdot \mathbf{X} + \left(\mathbf{H}_p + \mathbf{H}_d \frac{d}{dt} \right) \cdot \mathbf{Y} \ , \tag{66}$$

where $\mathbf{Y}$ is an array of measured errors, $\mathbf{X}$ is an estimate of the induced currents in the discretized representation of the passive electrical structure, and the $\mathbf{G}$ and $\mathbf{H}$

[8] This work was performed as a funded "ITER home task."

matrices are the control matrices. The $\mathbf{Y}$ array includes the gap errors, the plasma current error, and the departures of the eight active coils from their equilibrium values. The "d" and "p" subscripts refer to proportional and derivative gain (typically, only the proportional matrices are used—derivative gain is essentially built into this type of controller, even if the "d" matrices are not used). The control matrices were obtained by the GA team using Matlab and CORSICA. The equations are linear, and the solution for the applied voltage can be written as:

$$\mathbf{V} = \mathbf{U}(\Delta t) \cdot \mathbf{Y} \ . \tag{67}$$

This array of voltages is then used in the evolution of the circuit equations, as described in Section 3.5.4.

A typical application of CORSICA control systems modeling is controlling various vertical "events." For example, tokamak plasmas occasionally experience *minor disruptions*, which result in the near-instantaneous drop in the plasma's stored energy and a broadening of the internal current distribution. During such events the plasma is displaced from its desired position, and the control system must be capable of bringing the plasma back. Figure 9 shows the time evolution of the gap errors and the PF coil currents and applied voltages for a minor disruption simulation in which 15% of the total stored energy is instantaneously removed from the plasma at $t = 20$ ms. The peak displacement is about 14 cm, but it is quickly controlled. By 20 seconds, all gap errors are under 2 cm.

7.1.2 Modeling negative central shear discharges in DIII-D

The CORSICA code is being used by LLNL's MFE experimental group to model negative central shear (NCS) discharges created in the DIII-D tokamak experiment at GA.[9] In these experiments, discharges are created by varying the onset, timing and injected power of the 20 MW neutral beam heating and fueling system. The resulting current density profiles are peaked off axis to produce a safety factor profile, q, that is either flat or *inverted*, with the minimum of q located outward from the magnetic axis. (The magnetic shear, $dq/d\psi$ is thus negative in the center, which explains the name of this mode of operation.) Due to improved stability characteristics, this advanced tokamak configuration produces some of the highest performance discharges in DIII-D and is considered to be promising for steady-state and reactor applications. Current interest lies in techniques for controlling and sustaining these discharges. The current modeling effort includes four distinct efforts:

- Benchmarking CORSICA equilibria and time-dependent resistive current diffusion with experimental measurements.
- Validating and developing transport models (conductivity and diffusion) for the NCS configuration.

[9]This work was funded by the LLNL collaboration with GA.

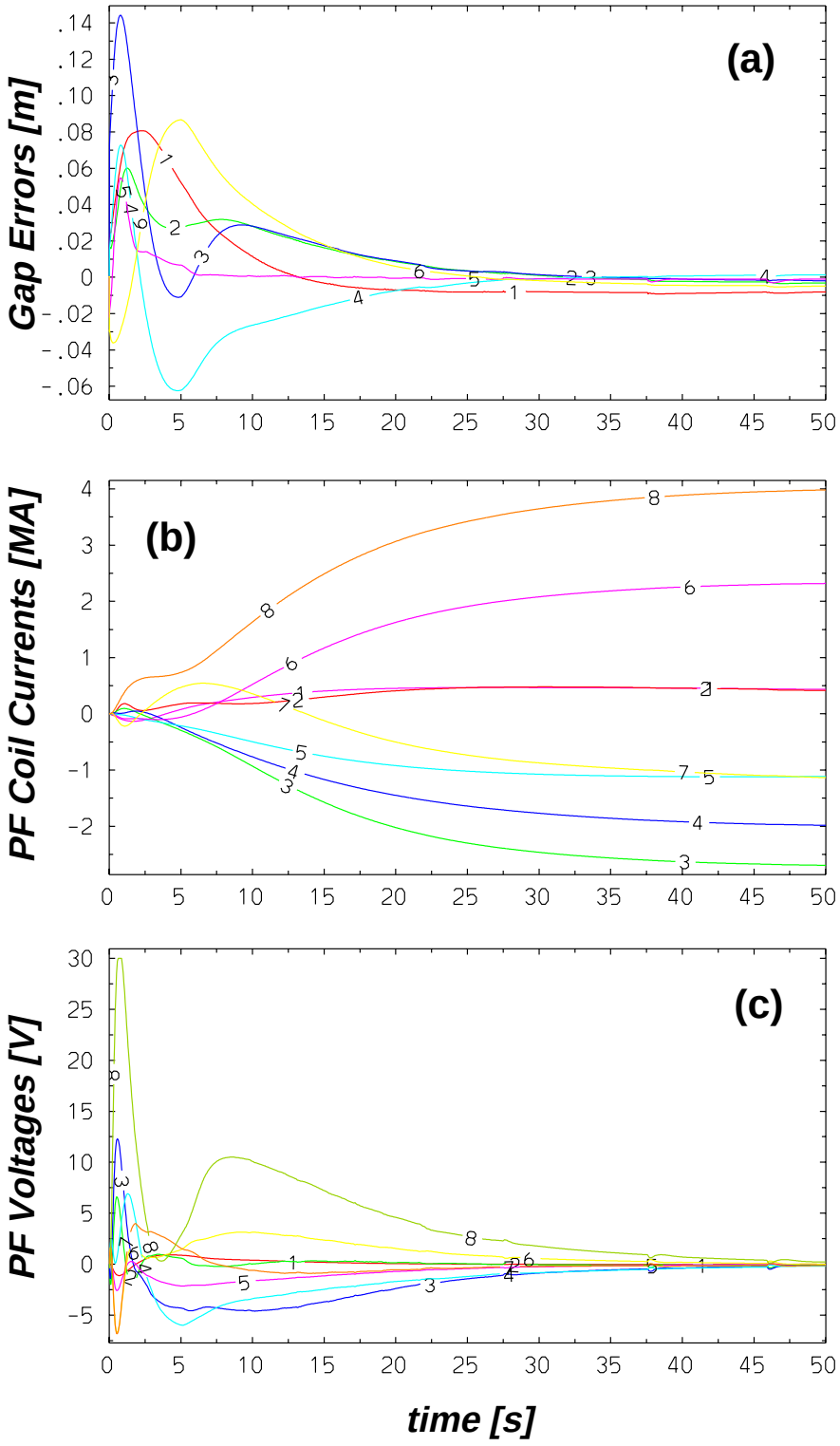

Figure 9: Plasma control example: (a) evolution of the plasma gaps; (b) currents in the active coils; (c) applied voltages for the active coils.

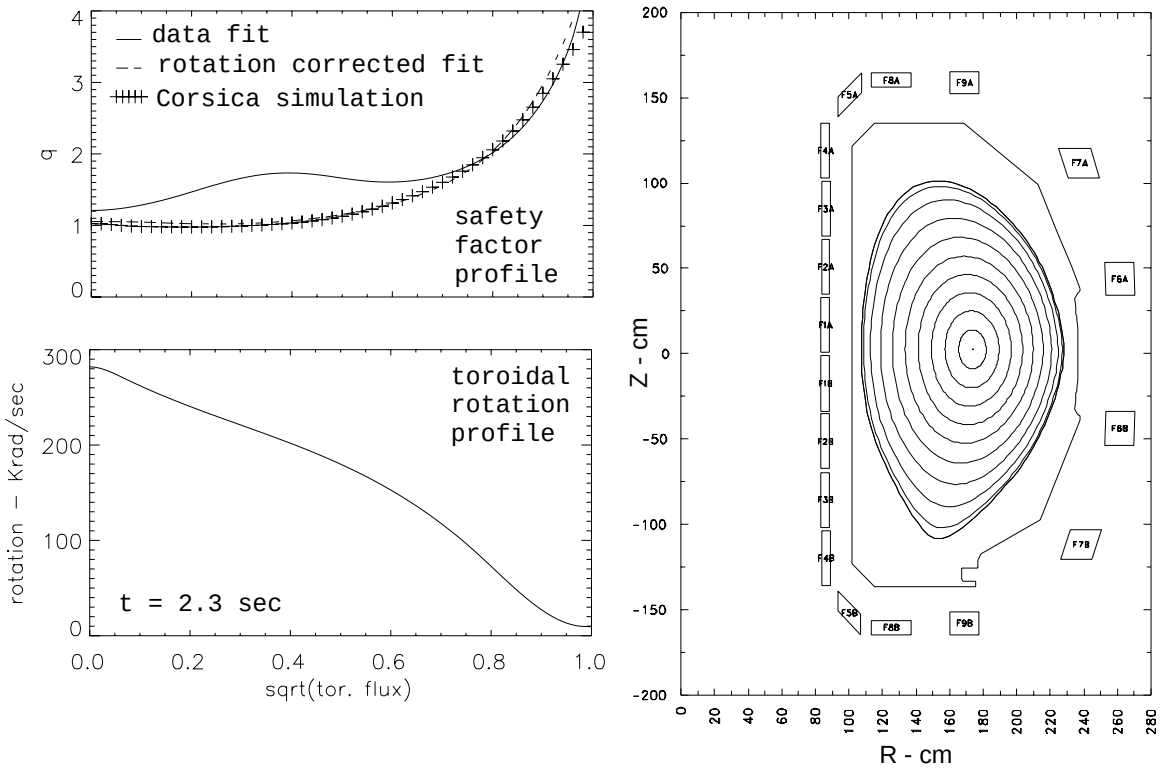

Figure 10: CORSICA simulation of the DIII-D NCS shot #88954.

- Exploring plasma shaping methods to optimize NCS creation.
- Exploring current drive methods to control and sustain the inverted q profile for long duration.

Depending on the nature of the work, CORSICA simulations can be done in either full simulation mode with models used for all plasma quantities or in a hybrid simulation mode where experimentally measured profiles are used to constrain the simulation.

As examples of CORSICA applications in modeling these experiments, we include results from the first two efforts. Using experimental values for density, temperature, and impurity concentration, CORSICA equilibrium solutions and the time-dependent neoclassical resistive current diffusion model have been favorably benchmarked with the analysis of experimental data [41–43]. A typical comparison of the simulated q profile with those produced by fitting experimental data is shown in Fig 10 for discharge #88964. Good agreement requires the new radial electric field (toroidal rotation) corrections to the motional Stark effect measurements for the experimental data. An example of the spatial-temporal evolution of the driven current profile and the resulting q profile for DIII-D discharge #84682 is shown in Fig. 11. Using this DIII-D discharge, we are also validating several existing transport models for the NCS configuration. Time-dependent temperature profiles have been evolved using four

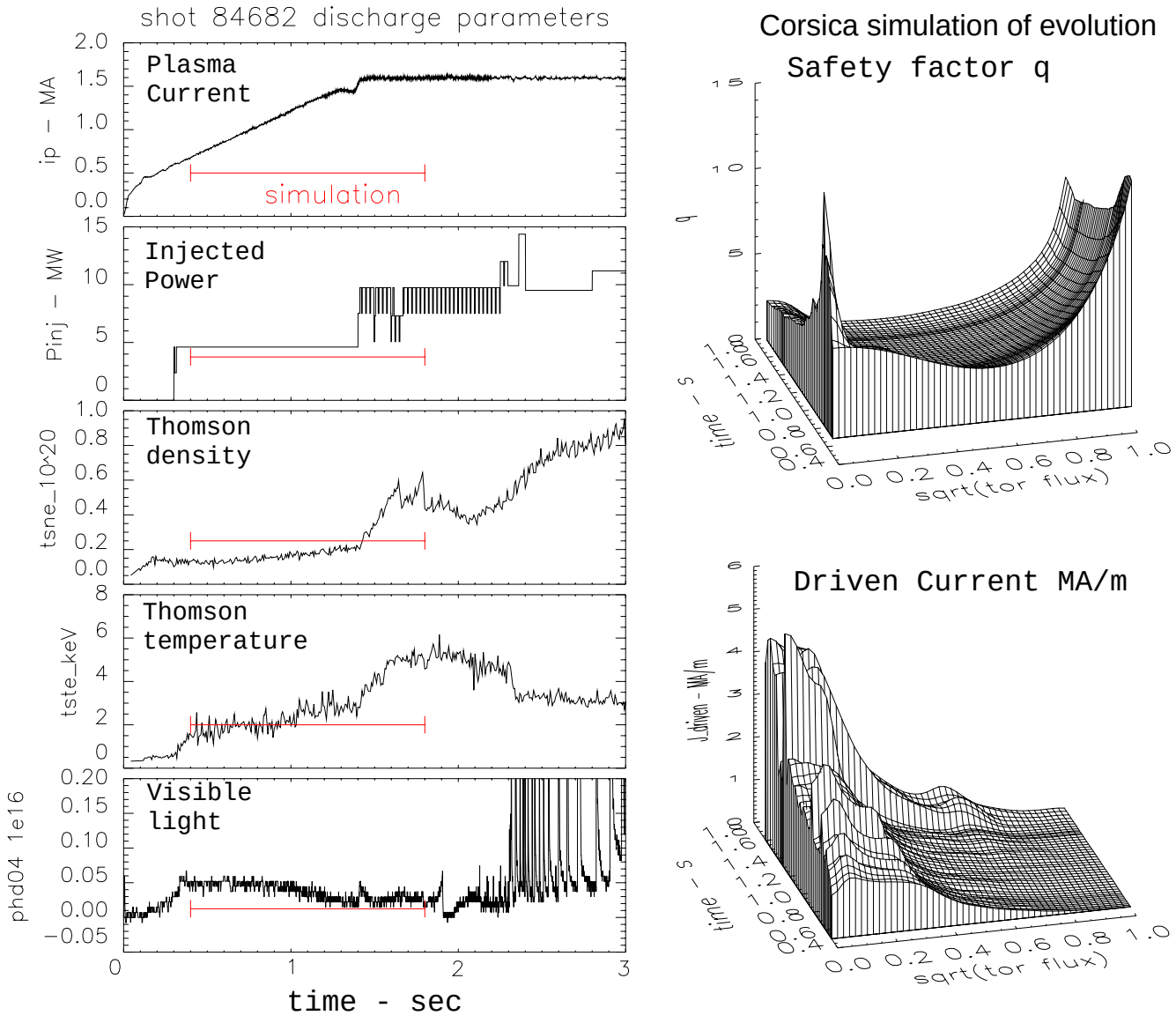

Figure 11: DIII-D shot #84682 measurements and CORSICA simulation of current diffusion.

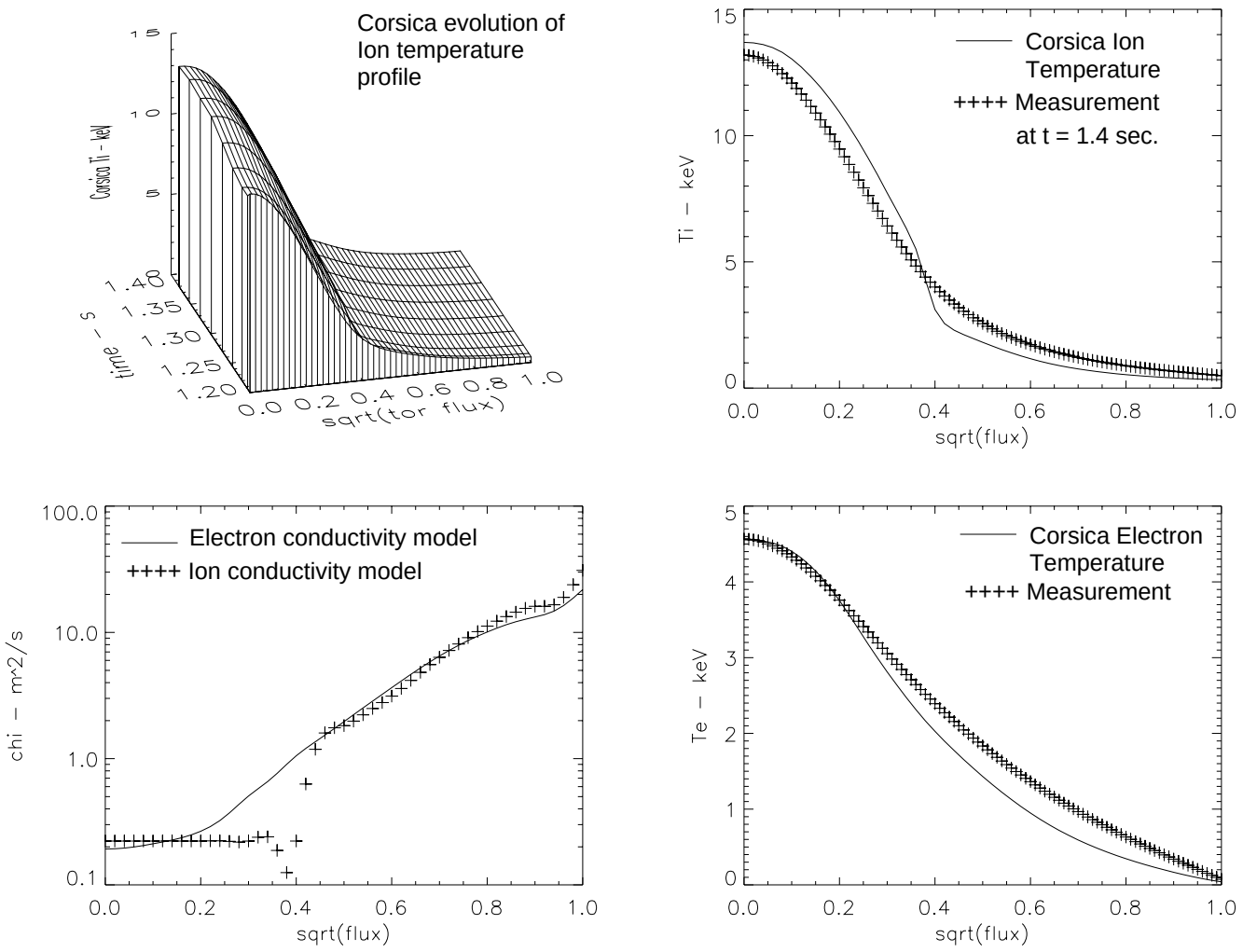

Figure 12: Comparison of transport simulation results and experimental data for DIII-D NCS shot using IFS/PPPL transport model.

different thermal conductivity models: the Chang-Hinton neoclassical model [44, 45], the Rebut-Lallia-Watkins model [46, 47], the IFS/PPPL model for ion temperature gradient turbulence [38], and a data fit based on experimental profile data and the TRANSP analysis code [48]. This modeling uses densities fixed by the experimentally measured profiles and neoclassical conductivity for current diffusion. Results for the IFS/PPPL model are shown in Fig. 12. In addition, variations of the plasma elongation and vertical position are being studied to investigate increasing the region of negative shear during plasma buildup by changing the current driven by the neutral beam injection.

7.2 Other core modeling tasks

CORSICA has been used for a variety of other tokamak core simulation applications that will not be discussed in depth here. These applications include, in rough chronological order:

- Modeling fusion power control for ITER.
- Modeling advanced operating scenarios (including NCS) for ITER and for the now-defunct Tokamak Physics Experiment (TPX).
- Modeling ignition performance of ITER and several variant ITER designs in order to assess sensitivity of predictions to impurities and to differing levels of auxiliary power.
- Modeling high-β_p DIII-D transient discharges observed during negative current ramps.
- Modeling vertical control of ELM-ing plasmas for ITER.
- Modeling the sensitivity of ITER ignition predictions to certain aspects of the ITER turbulent transport model.
- Modeling shape and position control for DIII-D.
- Modeling modifications to the ITER design to provide for studying steady-state operating scenarios.
- Modeling low-aspect ratio bootstrap driven tokamaks.
- In collaboration with magnet designers, using ITER position and shape control simulations to calculate the heating of the cold structure.
- Modeling current profile evolution in enhanced reversed shear (ERS, a cousin of the NCS discharges performed on DIII-D) scenarios in the Tokamak Fusion Test Reactor (TFTR) at Princeton [49].

7.3 Modeling for the LLNL SSPX spheromak

The LLNL spheromak group has been collaborating with the CORSICA group to make specific improvements in CORSICA, and to use CORSICA for design tasks for the new Sustained Spheromak Physics Experiment (SSPX). Magnetic field equilibria for the SSPX have been explored with CORSICA. These calculations were undertaken to compare the effects of conducting walls with that of external poloidal field coils, to evaluate the effects of the separatrix and x-point location due to the coaxial helicity injector, and to determine where to locate a divertor to handle particle and impurity losses in the experiment. The final result of this work is the determination of the geometry of a conducting flux conserving wall for SSPX. Figure 13 illustrates one of the designs for the coil set and equilibrium surfaces, as calculated by CORSICA. This geometry was tested for stability to the dominant MHD modes by transferring the results from CORSICA to the GATO stability code used at General Atomics.

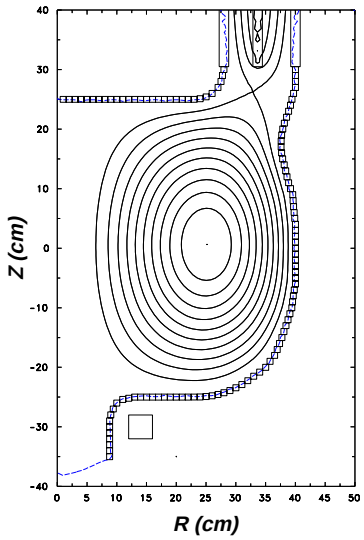

Figure 13: Magnetic configuration for the proposed SSPX spheromak, including the gun and divertor.

In addition to the determination of an optimized geometry for the experiment, several physics effects were considered. The maximum plasma β (the ratio of plasma pressure to magnetic pressure) for ideal MHD stability was determined as a function of the current profile in the plasma. For the case in which no electric currents flow on open magnetic flux surfaces surrounding the plasma, stable solutions were found with $\beta > 0.2$, consistent with previous results. The spheromak modifications to CORSICA allow consideration of equilibria with plasma current outside the separatrix. With such equilibria, only low values of β (a few percent) were generated for conventional current profiles. However, shaping the current profile to have a maximum, allowed higher values ($\beta > 0.1$) to be obtained. The implications for possible current drive are being considered.

In addition, energy transport was considered for the case of losses due to magnetic

fluctuations. If the scaling of the fluctuations drops rapidly enough with the magnetic Lundquist number (the ratio of resistive to Alfvén times) the plasma temperature was found to run away into the fusion regime. The implications for the experiment are that non-ideal effects, such as impurity radiation, may limit the temperature. If so, the experimental effort being applied to minimize the production of impurities may result in significant improvements over previous results.

The CORSICA group plans to expand our SSPX support. Possible applications, depending on funding, include:

- Using CORSICA to analyze experimental data.
- Developing and adding a $n = 1$ linear stability module for modeling the control of the tilt instability, which must be controlled for a spheromak to make a practical reactor.
- Beginning development of a coupled transport-3D MHD equilibrium code, which would allow fully nonlinear control simulations (similar to our current ability to model axisymmetric ($n = 0$) instabilities.

If the proposed LLNL experiment is successful, CORSICA will play an important role in design of a larger follow-on experiment.

7.4 Core-edge applications

7.4.1 Gas puffing test

An initial simple experiment was done to examine the response of the coupled codes to a gas puffing source in the SOL region. The purpose was to see how the core responds to an edge perturbation. A gas puff current was switched on in the SOL and ramped from 0 to 2000 Amps in 1 ms. Thereafter the gas flow was left constant and the profiles evolved toward a steady state. This time evolution is summarized in Fig. 14, and the behavior appears qualitatively consistent with observations in DIII-D. This simulation demonstrated that the coupled system converged properly (usually requiring only a small number of coupling iterations) and that reasonable sized time-steps could be taken.

7.4.2 L-H transition: Basic test

As a more thorough test of the CORSICA 2 code, an L-H transition in the DIII-D tokamak has been considered [50]. The initial conditions were taken from L-mode data for the DIII-D shot #86586 at 1630 ms. The transport coefficients in the core were chosen such that the initial profiles were in a quasi-steady state and approximately fit the experimental data, the corresponding edge values were set to the values used in the UEDGE simulation of the shot at the same time. The core coefficients remained

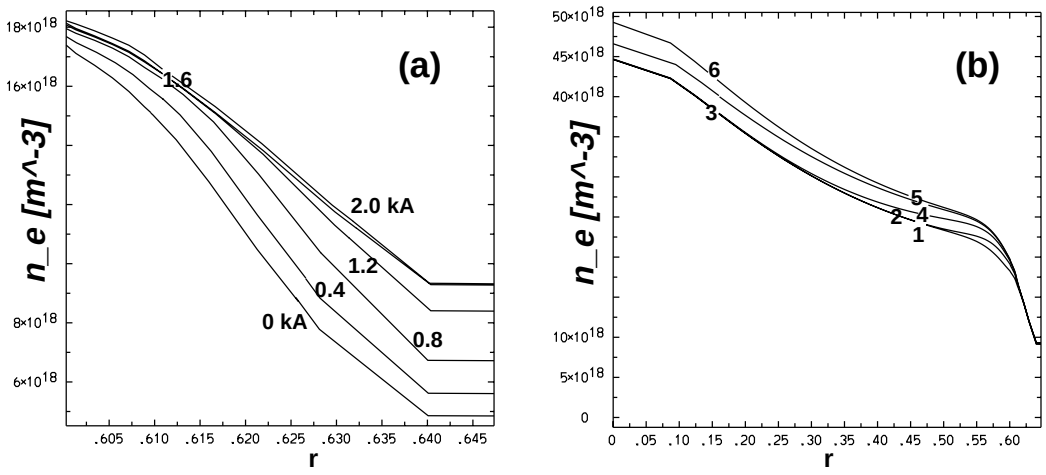

Figure 14: Gas puffing example. (a) Edge density profile during the initial ramp of 1 ms (0 to 2000 A in 400 A increments). (b) Core-edge density at t = 0, 1 ms, 10 ms, 100 ms, 1 s and 100 s for a constant gas flow at 2000 A (curves 1-6).

fixed throughout the remaining evolution. Next the core-edge coupling was enabled. Following a short transient, a new, consistent, steady state was reached.

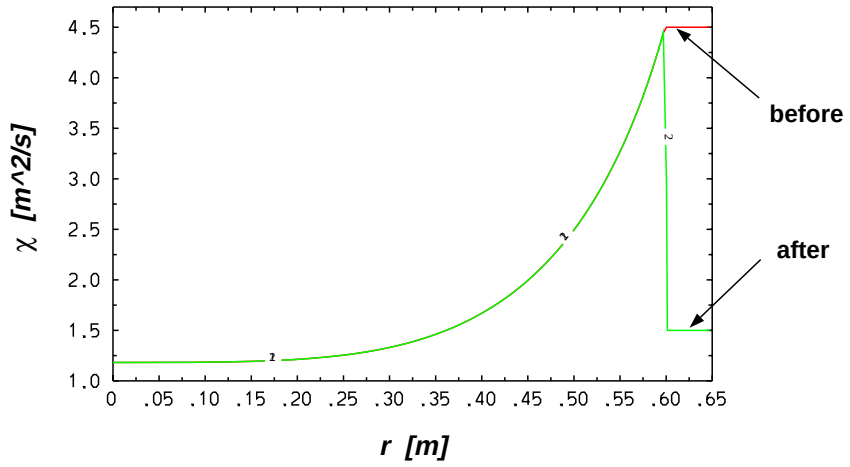

Figure 15: Change in χ profile used to force L-H transition.

In order to simulate an L-H transition as in the experiment, the fueling and the neutral beam heating were tripled. In the experiment, after 20 ms of higher beam power, the transition occurs. In the simulation this transition was modeled by decreasing the diffusion coefficient (D) and the thermal conductivities (χ_e, χ_i) in the edge (Fig. 15). The new values for D, χ_e and χ_i were chosen from a UEDGE run that fit the H-mode measurements at the divertor plate for the same shot at a later time (2550 ms). In this case, the transport coefficients do not exhibit a very strong variation: D drops from 0.4 to 0.3 m^2/s, while the χ's dropped from 4.5 to 1.5 m^2/s. The decreased edge transport coefficients set up a transport barrier that results in

the pedestal-type temperature profiles as observed in the experiment. Fig. 16 shows the time histories of the temperature and density at several positions in the plasma. Fig. 17 shows the initial electron temperature profile and the profile at 2550 ms, along with the experimental data at the same times.

This example demonstrates that the code is sufficiently robust to handle dramatic transitions, and that it produces results that are consistent with the experimental data.

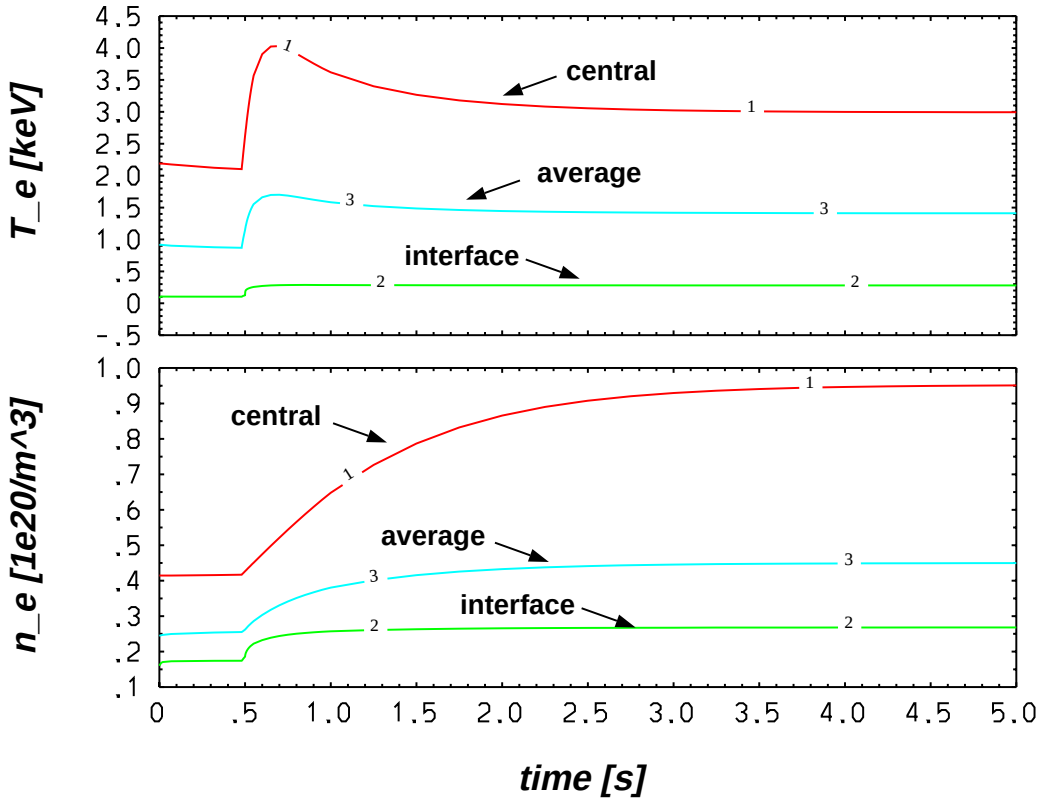

Figure 16: Time histories for the density and temperature at several locations in the plasma.

7.4.3 A semi-empirical, self-consistent L-H transition model

A self-consistent semi-empirical simulation of the L-H transition has recently been performed. This simulation is similar to that described above. However, the drop of the transport coefficients in the H-mode is now a result of changes in the SOL χ and D coefficients as the system undergoes a first order phase transition. The transition is triggered by raising the power in the core (simulating neutral beam injection), which results in an increase in the SOL temperature gradient. The model transport coefficients, which are based on a SOL turbulence model by Cohen, et al., [51], depend nonlinearly on the temperature gradient, and once a critical power flux is achieved, the system undergoes a bifurcation to a new state.

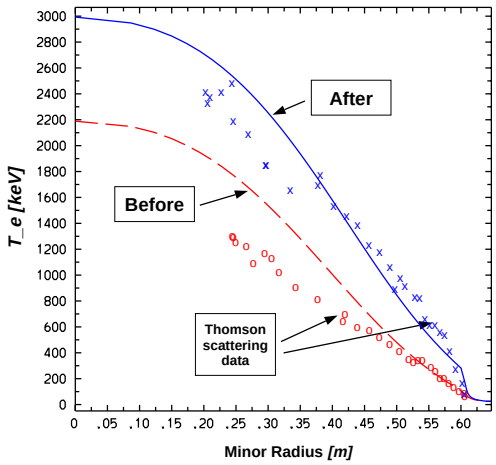

Figure 17: Forced L-H Example: L-Mode and H-Mode profiles and comparisons with experimental measurements.

The SOL turbulence model prescribes the particle and energy diffusion coefficients as proportional to the same turbulent diffusivity. This turbulent diffusivity is expressed as:

$$D_{\text{turb}} = (\rho_s^2 c_s / L_T)\nu^{-1/3} \frac{1 - C(\rho_s/L_T)^2}{1 + K^2} \ , \tag{68}$$

where ρ_s is the ion gyroradius at the divertor plate electron temperature T_{eD}, c_s the ion sound speed at the divertor plate, $L_T = T_{eD}/|\nabla T_e|_{MP}$, where the gradient of T_e is taken at the midplane, $\nu = L_T/L_\parallel$, $K = \beta^{1/2}\nu^{-2/3}$, and C is an adjustable coefficient.

The particle and energy diffusion coefficients in the scrape-off layer are modeled by

$$D_{SOL} = D_0 + k_D D_{\text{turb}} \tag{69}$$

$$\chi_{SOL} = \chi_0 + k_\chi D_{\text{turb}} \ , \tag{70}$$

where D_0 and χ_0 are the H-mode coefficients (held constant in the present treatment). Inside the separatrix, D and χ are fixed at values that initially matches the SOL values. In the core, D and χ have a linear profile in Φ, and so are roughly parabolic in the minor radius r. They smoothly match on to the edge values.

The model for D_{turb} is "non-local" because L_T is a function of the temperature at the divertor plate (a local model would depend on the value of L_T at the point where the D_{turb} is evaluated).

The transition occurs when the function $1 - C(\rho_s/L_T)^2$ goes to zero. (To prevent the diffusion coefficients from becoming negative, D_{turb} is set to zero whenever its value should go negative.) The coefficient C has been chosen empirically in order to make the transition occur at the proper power level. The other parameters are set based on the physics of the experiment. Determination of C is not trivial due to

the nonlinear character of the SOL turbulence model—an initialization procedure is required to compute a diffusivity consistent with the initial plasma profiles. This is done via an iteration loop that is executed at the beginning of the run. The profiles are evolved for one small time step from an initial null value of D_{turb}; then the actual value of D_{turb} is computed from the advanced profiles and the procedure is iterated to convergence.

The results for the simulation of DIII-D shot #86586 are shown Figs. 18 and 19. This simulation starts from the same initial state as the earlier simulation, but the model coefficients are chosen to give smaller H-Mode transport coefficients (D and χ were both dropped to 0.1 m^2/s, values that are more typical of H-Mode shots in DIII-D). Figure 18 shows the history of D_{turb} during the simulation. In the early stages we see it increasing, but at approximately 13 ms it begins to drop slightly, and at around 16 ms it rapidly drops to zero. Figure 19 shows the electron temperature profile before and after the transition. Note that the smaller final transport coefficients have resulted in a sharper temperature pedestal than seen in Fig. 17. Also, note that this simulation was not successfully run to steady state (due to numerical problems in the edge simulation that may be related to the discontinuous transport coefficients that exist after the transition). The H-Mode curve shown in Fig. 19 is at 15 ms past the L-H transition, whereas the H-Mode curve shown in Fig. 17 is at 2 seconds past the transition. One can infer from Fig. 16 that the simulation still has quite a long way to go.

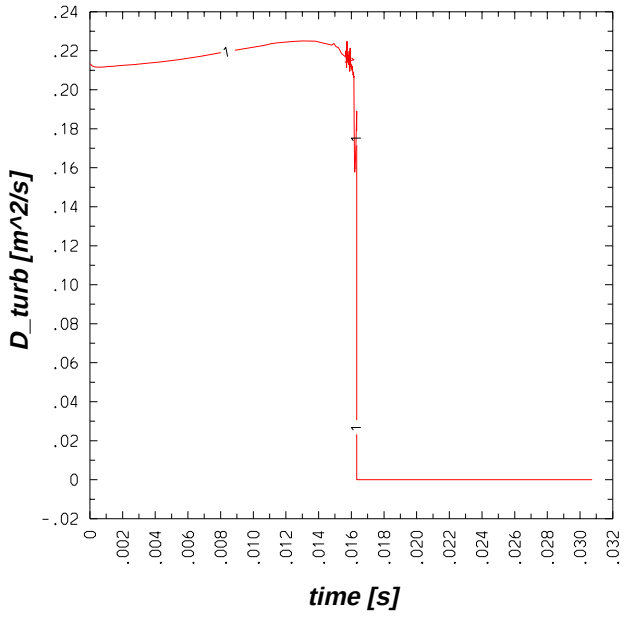

Figure 18: Evolution of the turbulent diffusivity during the L-H transition.

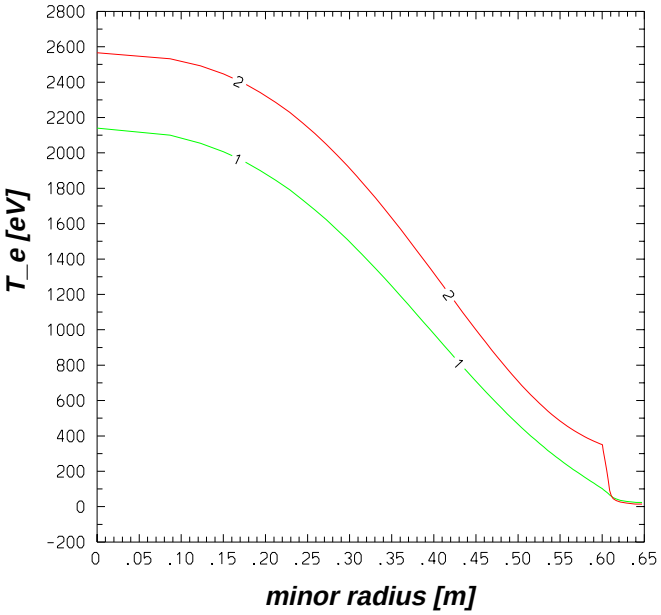

Figure 19: Electon temperature profiles: (1) Initial L-Mode profile, (2) Profile after the transition (at $t = 30$ ms).

8 Future plans

The CORSICA project has succeeded in its goals of developing algorithms for multiple scale coupling, and has successfully implemented most of these in the prototype code. We have been ambitiously marketing these capabilities to our collaborators at General Atomics (GA) and elsewhere, as well as to our sponsors at DOE. While we have been fairly successful with the former, recent drastic cuts in the MFE program have made the latter much more difficult. In spite of these cuts, there is some cause for modest optimism:

The SSPX Spheromak: Our experimental group has received joint funding from LLNL's LDRD program and DOE's Office of Fusion Energy Sciences (OFES) to build a small spheromak experiment at LLNL. As described in Section 7.3, the spheromak group makes heavy use of CORSICA for equilibrium and stability studies. We have requested funding to give more support to this project and to work toward full spheromak transport simulations.

ITER Obligations: CORSICA is considered to be a crucial tool for PF design for ITER. Using CORSICA, our engineers are able to study new designs in a fraction of the time that it takes the Japanese to do similar analyses.

GA Interest: There is increasing interest at GA in using CORSICA to analyze and model the DIII-D experiment, which will be the U.S.'s largest tokamak experiment by the end of FY97. This has recently led us to propose ...

GA Collaboration: General Atomics has a production core-transport code, OneTwo [52]. This code has a great deal of overlap with CORSICA, but it is not as easy to use nor can it do PF design, core-edge coupling, etc. Both GA and LLNL have recognized that it would be mutually beneficial to merge the CORSICA effort with GA's effort, and to produce a new code. Our proposed plan will be discussed in the next section.

The first three items require support and improvement of the existing CORSICA code. We are pressing to receive increased funding to help cover these costs, which are currently being covered by deferring some of our existing MFE money. We will be receiving some computer support from the DIII-D experimental group this year, which is a step in the right direction.

8.1 The LLNL-GA proposal

The needs of the LLNL and GA physicists to have a better tool to analyze and model the DIII-D experiment, coupled with the CORSICA group's desire to build on our present capabilities, had led us to propose a collaboration between the two efforts. This collaboration will build a new code using physics and algorithms from the existing codes, but based on a modern, object-oriented, scripting framework based on

the Python system. This project has the strong support of both LLNL and GA management, and we believe it has a good chance of receiving additional DOE funding (possibly even this fiscal year). Our work plan for the next two years is composed of the following tasks:

1. Develop a graphical user interface (GUI) that will work with both Python and Basis.

2. Design an object-oriented (OO) framework for constructing integrated simulation codes from "physics plugins."

3. Port OneTwo to run under Python (with the GUI, but not the modern framework)

4. Implement the OO framework in Python.

5. Develop tools to aid in writing new plugins and in porting existing physics modules into the plugin framework.

6. Port CORSICA and OneTwo physics modules into the framework.

This approach will get an easier-to-use tool into the hands of physicists working on DIII-D in the fairly near future, and the tool will be directly comparable to OneTwo results. The code structure will then be modernized and the physics improved while maintaining the same graphical user interface.

The near-term physics code application (a merger of CORSICA and OneTwo) is much more moderate than the previous proposals that we have made to DOE, which aimed at expanding our LDRD project to a national collaboration to do comprehensive computing. However, we still have a long-term goal of comprehensive computing in a flexible, programmable, environment, and we plan to design the framework with this goal in mind.

One of our motivations in pursuing this project is our desire to upgrade the programmable framework on which CORSICA is built. The use of Basis has made CORSICA a much more powerful tool than it would otherwise have been, and it has also increased the efficiency of the CORSICA group in developing, testing, and implementing our algorithms. However, a more modern, object-oriented framework would be more powerful, not to mention portable. This same line of reasoning has led several of the ASCI projects to choose Python as the base of their software development efforts, and we plan to collaborate with these groups. Furthermore, we believe that the proposed project will be a good candidate for FY98 DOE2000 toolkit development funding, and will pursue such funding aggressively.

References

[1] B. I. Cohen, D. C. Barnes, J. M. Dawson, G. W. Hammett, W. W. Lee, G. D. Kerbel, J.-N. Leboeuf, P. C. Liewer, T. Tajima, and R. E. Waltz. The Numerical Tokamak Project: Simulation of turbulent transport. *Comp. Phys. Commun.*, 87(1):1–15, January 1995.

[2] Harold Grad and John Hogan. Classical diffusion in a tokamak. *Phys. Rev. Lett.*, 24(24):1337–1340, June 1970.

[3] H. Grad, P. N. Hu, and D. C. Stevens. Adiabatic evolution of plasma equilibrium. *Proc. Nat. Acad. Sci. USA*, 72(10):3789–3793, October 1975.

[4] M. Baelmans et al. Presented at The 22rd European Physical Society Conference on Controlled Fusion and Plasma Physics, Bournemouth, U.K., 1995.

[5] M. Baelmans, D. Reiter, M. Tokar, and P. Börner. Consistent core-edge plasma modelling for TEXTOR plasmas with carbon impurities. Presented at the Fifth Plasma Edge Theory Meeting, Asilomar, California, December 4-6, 1995.

[6] A. Taroni. Global codes for edge plasmas. Presented at the Fifth Plasma Edge Theory Meeting, Asilomar, California, December 4-6, 1995.

[7] M. A. Beer. *Gyrofluid models of turbulent transport in tokamaks*. PhD thesis, Princeton University, January 1995.

[8] W. D. D'haeseleer, W. N. G. Hitchon, J. D. Callen, and J. L. Shohet. *Flux Coordinates and Magnetic Field Structure*. Springer-Verlag, 1991.

[9] H. Grad and H. Rubin. In *Proc. of the Second United Nations Conference on the Peaceful Uses of Atomic Energy*, volume 31, page 190, Geneva, 1958. United Nations.

[10] S. C. Jardin, N. Pomphrey, and J. DeLucia. Dynamic modeling of transport and position control of tokamaks. *J. Comput. Phys.*, 66(2):481–507, October 1986.

[11] James A. Crotinger. CORSICA Users' Manual. Technical Report UCRL-MA-126273 Dr, Lawrence Livermore National Manual, 1997. DRAFT.

[12] F. L. Hinton and R. D. Hazeltine. Theory of plasma transport in toroidal confinement systems. *Rev. Mod. Phys.*, 48(2):239–308, 1976.

[13] S. P. Hirshman and S. C. Jardin. Two-dimensional transport of tokamak plasmas. *Phys. Fluids*, 22(4):731–742, April 1979.

[14] R. Balescu. *Transport Processes in Plasmas: 1. Classical Transport Theory*, volume 1. North-Holland, 1988.

[15] R. Balescu. *Transport Processes in Plasmas: 2. Neoclassical Transport Theory*, volume 2. North-Holland, 1988.

[16] C. M. Surko and R. E. Slusher. Waves and turbulence in a tokamak fusion plasma. *Science*, 221:817–822, August 1983.

[17] Paulett C. Liewer. Measurements of microturbulence in tokamaks and comparison with theories of turbulence and anomalous transport. *Nucl. Fusion*, 25(5):543–621, 1985.

[18] W. M. Tang. Microinstability-based model for anomalous thermal confinement in tokamaks. *Nucl. Fusion*, 26(12):1605–1618, 1986.

[19] L. L. LoDestro and L. D. Pearlstein. On the Grad-Shafranov equation as an eigenvalue problem, with applications to q solvers. *Phys. Plasmas*, 1(1):90–95, January 1994.

[20] Kelly A. Barrett, Yu-Hsing Chiu, Paul F. Dubois, Jeff F. Painter, and Zane C. Motteler. Running a Basis Program; A Tutorial for Beginners. Technical Report UCRL-MA-118543 Part I, Lawrence Livermore National Laboratory, Livermore, CA, 1995.

[21] Paul F. Dubois and Zane C. Motteler. Basis Language Reference Manual. Technical Report UCRL-MA-118543 Part II, Lawrence Livermore National Laboratory, Livermore, CA, 1994.

[22] Yu-Hsing Chin and Paul F. Dubois. EZN User Manual. Technical Report UCRL-MA-118543 Part III, Lawrence Livermore National Laboratory, Livermore, CA, 1994.

[23] Yu-Hsing Chin and Paul F. Dubois. EZD User Manual. Technical Report UCRL-MA-118543 Part IV, Lawrence Livermore National Laboratory, Livermore, CA, 1994.

[24] Paul F. Dubois and Zane C. Motteler. Writing Basis Programs; A Manual for Program Authors. Technical Report UCRL-MA-118543 Part V, Lawrence Livermore National Laboratory, Livermore, CA, 1994.

[25] Paul F. Dubois. The Basis Package Library; A Manual for Program Authors. Technical Report UCRL-MA-118543 Part VI, Lawrence Livermore National Laboratory, Livermore, CA, 1994.

[26] R. R. Khayrutdinov and V. E. Lukash. Studies of plasma equilibrium and transport in a tokamak fusion device with the inverse-variable technique. *J. Comput. Phys.*, 109(2):193–201, December 1993.

[27] S. W. Haney, W. L. Barr, J. A. Crotinger, L. J. Perkins, C. J. Solomon, E. A. Chaniotakis, J. P. Freidberg, J. Wei, J. D. Galambos, and J. Mandrekas. A "SuperCode" for systems analysis of tokamak experiments and reactors. *Fus. Technol.*, 21(3):1749–1758, May 1992.

[28] A. Tarditi, R. H. Cohen, T. D. Rognlien, and G. R. Smith. Coupling core and edge non-linear transport fluid codes. In *Bull. Am. Phys. Soc.*, volume 38, page 2103. The American Physical Society, 1993.

[29] J. A. Crotinger, R. H. Cohen, S. W. Haney, L. L. LoDestro, A. I. Shestakov, G. R. Smith, L. D. Pearlstein, T. D. Rognlien, and A. G. Tarditi. Corsica: A comprehensive tokamak simulation code. In *Bull. Am. Phys. Soc.*, volume 38, page 2016. The American Physical Society, 1993.

[30] T. D. Rognlien, M. E. Milovich, J. L.and Rensink, and G. D. Porter. A fully implicit, time dependent 2-D fluid code for modeling tokamak edge plasmas. *J. Nucl. Mater.*, 196-198:347–51, 1992.

[31] T. D. Rognlien, P. N. Brown, R. B. Cambell, T. B. Kaiser, et al. 2-D fluid transport simulations of gaseous/radiative divertors. *Contrib. Plasma Phys.*, 34:362–367, 1994.

[32] A. I. Shestakov, R. H. Cohen, J. A. Crotinger, L. L. LoDestro, A. Tarditi, and X.-Q. Xu. Self-Consistent Modeling of Turbulence and Transport. Submitted to *J. Comp. Phys.*, March 1997.

[33] M. Wakatani and A. Hasegawa. A collisional drift wave description of plasma edge turbulence. *Phys. Fluids*, 27(3):611–618, March 1984.

[34] S. I. Braginskii. Transport processes in a plasma. In M. A. Leontovich, editor, *Reviews of Plasma Physics*, volume 1, pages 205–311. Consultants Bureau, New York, June 1965.

[35] A. E. Koniges, J. A. Crotinger, and P. H. Diamond. Structure formation and transport in dissipative drift-wave turbulence. *Phys. Fluids B*, 4(9):2785–2793, September 1992.

[36] X.-Q. Xu, R. H. Cohen, J. A. Crotinger, and A. I. Shestakov. Fluid simulations of nonlocal dissipative drift-wave turbulence. *Phys. Plasmas*, 2(3):686–701, March 1995.

[37] X.-Q. Xu and R. H. Cohen. 3D Fluid simulations of turbulence in detached scrape-off-layer plasmas. *Contrib. Plasma Phys.*, 36(2/3):202–206, 1996.

[38] M. Kotschenreuther, W. Dorland, M. A. Beer, and G. W. Hammett. Quantitative predictions of tokamak energy confinement from first-principles simulations with kinetic effects. *Phys. Plasmas*, 2(6):2381–2389, June 1995.

[39] D. A. Humphreys, J. A. Leuer, A. G. Kellman, S. W. Haney, R. H. Bulmer, L. D. Pearlstein, and A. Portone. Toward a design for the ITER plasma shape and stability control system. *Fus. Technol.*, 26(3):331–339, November 1994.

[40] A. Portone, Y. Gribov, M. Huguet, P. L. Mondins, R. Albanese, D. Ciscato, D. A. Humphreys, C. E. Kessel, L. D. Pearlstein, and D. J. Ward. Plasma position and shape control for ITER. In G. H. Miley and C. Elliot, editors, *16th IEEE/NPSS Symposium on Fusion Engineering*, volume 1, pages 345–348, New York, NY, 1995. IEEE.

[41] T. A. Casper, J. Crotinger, S. Haney, J. M. Moller, L. D. Pearlstein, B. W. Rice, and B. W. Stallard. Milestone report: Status report on time-dependent modeling for current profile feedback control. Technical Report UCRL-ID-122226, Lawrence Livermore National Laboratory, September 1995.

[42] T. A. Casper, J. Crotinger, J. Moller, L. D. Pearlstein, B. Rice, B. Stallard, L. Lao, and T. Taylor. Modeling of current profile evolution and equilibria in negative central shear discharges in the DIII-D experiment. In *23rd European Conference on Controlled Fusion and Plasma Physics; Kiev, Ukraine, June 24-28, 1996*, volume 20C, part I of *Europhysics Conference Abstracts*, page 295. European Physical Society, 1996.

[43] T. A. Casper, M. C. Spang, J. Crotinger, L. D. Pearlstein, , B. W. Stallard, and T. Taylor. Corsica time-dependent modeling of DIII-D discharges. In *Bull. Am. Phys. Soc.*, volume 41, page 1571. The American Physical Society, November 1996.

[44] C. S. Chang and F. L. Hinton. Effect of finite aspect ratio on the neoclassical ion thermal conductivity in the banana regime. *Phys. Fluids*, 25(9):1493–1494, September 1982.

[45] C. S. Chang and F. L. Hinton. Effect of impurity particles on the finite-aspect ratio neoclassical ion thermal conductivity in a tokamak. *Phys. Fluids*, 29(10):3314–3316, October 1986.

[46] P. H. Rebut, P. P. Lallia, and M. L. Watkins. The critical temperature gradient model of plasma transport: applications to jet and future tokamaks. In *Plasma Physics and Controlled Nuclear Fusion Research 1988 (Proc. 12th Int. Conf., Nice, France)*, volume 2, pages 191–200. IAEA, 1988.

[47] M. N. Rosenbluth, J. Hogan, D. Boucher, et al. ITER plasma modelling and MHD stability limits. In *Plasma Physics and Controlled Nuclear Fusion Research 1994 (Proc. 15th Int. Conf., Seville, Spain)*, volume 2, pages 517–524. IAEA, 1994.

[48] S. M. Kaye, R. J. Goldston, M. Bell, K. Bol, et al. Thermal energy confinement scaling in PDX limiter discharges. *Nucl. Fusion*, 24(10):1303–1334, October 1984.

[49] E. B. Hooper, L. D. Pearlstein, and R. H. Bulmer. Modeling of MHD equilibria and current profile evolution during the ERS mode in TFTR. Technical Report UCRL-ID-124818, Lawrence Livermore National Laboratory, July 1996.

[50] A. Tarditi, R. H. Cohen, J. A. Crotinger, G. D. Porter, T. D. Rognlien, A. I. Shestakov, and G. R. Smith. Self-consistent core-edge nonlinear transport simulation with CORSICA 2. *Contrib. Plasma Phys.*, 36:132, 1996.

[51] Ronald H. Cohen and Xueqiao Xu. A model for scrape-off-layer low-high (L-H) mode transition. *Phys. Plasmas*, 2(9):3374–3383, September 1995.

[52] W. Pfeiffer, F. B. Marcus, C. J. Armentrout, G. L. Jahns, T. W. Petrie, and R. E. Stockdale. Giant sawtooth oscillations in the Doublet III tokamak. *Nucl. Fusion*, 25(6):655–671, 1985.