Informes Técnicos Ciemat

1201
Marzo, 2010

Programas Periféricos de ASTRA para el TJ-II

D. López-Bruna
J. M. Reynolds
Á. Cappa
J. Martinell
J. García
C. Gutiérrez-Tapia

Laboratorio Nacional de Fusión

Toda correspondencia en relación con este trabajo debe dirigirse al Servicio de Información y Documentación, Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas, Ciudad Universitaria, 28040-MADRID, ESPAÑA.

Las solicitudes de ejemplares deben dirigirse a este mismo Servicio.

Los descriptores se han seleccionado del Thesauro del DOE para describir las materias que contiene este informe con vistas a su recuperación. La catalogación se ha hecho utilizando el documento DOE/TIC-4602 (Rev. 1) Descriptive Cataloguing On-Line, y la clasificación de acuerdo con el documento DOE/TIC.4584-R7 Subject Categories and Scope publicados por el Office of Scientific and Technical Information del Departamento de Energía de los Estados Unidos.

Catálogo general de publicaciones oficiales
http://www.060.es

Depósito Legal: M -14226-1995
ISSN: 1135 - 9420
NIPO: 471-10-016-9

Editorial CIEMAT

CLASIFICACIÓN DOE Y DESCRIPTORES

S70
MAGNETIC FIELD CONFIGURATION; MAGNETIC CONFINEMENT;
CALCULATION METHODS; STELLARATORS; CHARGED-PARTICLE TRANSPORT;
TOKAMAK DEVICES; PLASMA; PARTICLE SOURCES

Programas Periféricos de ASTRA para el TJ-II

López-Bruna, D.; Reynolds, J. M.; Cappa, Á.; Martinell, J.; García, J.; Gutiérrez-Tapia, C.
26 pp. 5 fig. 37 ref.

Resumen:

Muchos cálculos de transporte para estudiar los datos del dispositivo TJ-II se realizan con el sistema de transporte ASTRA. Sin embargo, ingredientes fundamentales en estos cálculos tales como las fuentes de partículas o de energía se obtienen mediante complicados códigos independientes. Éstos son accesibles desde ASTRA mediante los procedimientos que se explican.

Peripheral Codes in ASTRA for the TJ-II

López-Bruna, D.; Reynolds, J. M.; Cappa, Á.; Martinell, J.; García, J.; Gutiérrez-Tapia, C.
26 pp. 5 fig. 37 ref.

Abstract:

The study of data from the TJ-II device is often done with transport calculations based on the ASTRA transport system. However, complicated independent codes are used to obtain fundamental ingredients in these calculations, such as the particle and/or energy sources. These codes are accesible from ASTRA through the procedures explained in this report.

Programas periféricos de ASTRA para el TJ-II

D. López-Bruna, J. M. Reynolds, Á. Cappa, J. Martinell,
J. García, C. Gutiérrez-Tapia

Resumen

Muchos cálculos de transporte para estudiar los datos del dispositivo TJ-II se realizan con el sistema de transporte ASTRA. Sin embargo, ingredientes fundamentales en estos cálculos tales como las fuentes de partículas o de energía se obtienen mediante complicados códigos independientes. Éstos son accesibles desde ASTRA mediante los procedimientos que se explican.

1. *Introducción*

Para calcular propiedades del transporte en los dispositivos de fusión por confinamiento magnético se necesitan las fuentes y los coeficientes de transporte. En el TJ-II es particularmente difícil estimar bien las fuentes de calor y partículas y por eso se ha invertido mucho trabajo en desarrollar o adaptar códigos con este propósito. En cuanto a los coeficientes de transporte, bien sabido es que se trata de un problema sin resolver en la actualidad. Por otro lado, sin embargo, en las máquinas del tipo *stellarator* está razonablemente bien establecida la preponderancia de los flujos colisionales para determinar el potencial electrostático macroscópico del plasma. Esto permite considerar que los flujos de transporte de partículas son una suma de flujos intrínsecamente ambipolares –p. ej. provenientes de la turbulencia electrostática–; y flujos colisionales, que determinan el campo eléctrico al imponer la condición de ambipolaridad. De éstos hay abundante literatura y existen formulaciones analíticas para los casos más sencillos. También existen extensos códigos de cálculo numérico para estimar apropiadamente los flujos colisionales en función del campo eléctrico. En todo caso, la estimación del campo eléctrico "ambipolar" y los términos fuente en *stellarators* son piezas clave en la interpretación del transporte.

El interés de los programas de cálculo de las fuentes y los coeficientes de transporte es mucho mayor si pueden acoplarse automáticamente a algún código de transporte. Puesto que el más usado actualmente para analizar el transporte en los plasmas del TJ-II es ASTRA [**1**], en este informe recopilamos los procedimientos de cálculo que hay actualmente disponibles para ejecutar desde ASTRA. Podemos clasificar los códigos que aquí se nombran en

- Códigos principales (en particular EIRENE [**2**], FAFNER [**3**], TRUBA [**4**] y el propio ASTRA).
- Subrutinas de ASTRA (se encuentran en el subdirectorio `$HOME/astra/sbr`, dentro del directorio de ASTRA en el área de usuario).
- Códigos intermediarios (pues normalmente lo son entre códigos principales)

Los códigos principales tienen sus propios manuales mientras que los códigos intermediarios –pero no siempre– cuentan con pequeños documentos de ayuda. El presente informe pretende ayudar al usuario en el empleo de algunas subrutinas de ASTRA y los códigos intermediarios que manejan para compartir información con los códigos principales. Llamamos "programa periférico" a la conjunción de los tres tipos de código arriba enlistados aunque, como

2010 *PACS:* 52.25.Fi, 52.55.Dy,52.65.-y.

veremos, un programa periférico puede consistir sólo en una subrutina de ASTRA. En general, "periférico" alude a ASTRA porque se trata de ampliar la capacidad de cálculo de ASTRA mediante los cálculos realizados por otro código principal.

Cuando un usuario edita y diseña subrutinas para el sistema ASTRA, éste recompila dando lugar a un ejecutable que incluye el objeto asociado a la nueva subrutina. Normalmente se comparten variables en memoria y el ejecutable resulta eficiente. Sin embargo, cuando se dispone de otros códigos principales que hacen cálculos costosos, al usuario no le compensa reescribir el código principal para convertirlo en una subrutina de ASTRA, o ASTRA en un proceso subordinado del código principal. Para acoplar ASTRA a otro código principal utilizamos una subrutina de ASTRA que, normalmente, hace llamadas al sistema para que se ejecute un código intermediario diseñado para lanzar otros códigos principales. El código intermediario suele gestionar el intercambio de datos entre ASTRA y el otro programa principal. Pero, entonces, la comunicación entre el intermediario precompilado y ASTRA no puede hacerse a través de los registros de memoria del ordenador sino a través de la lectura–escritura de archivos, un método muy lento. Esto no es inadecuado cuando el programa principal llamado es muy costoso numéricamente (la lectura–escritura pasa a consumir un tiempo comparativamente despreciable) o bien cuando sólo es preciso llamarlo unas pocas veces por ejecución de ASTRA.

El presente documento es un primer manual descriptivo de los códigos intermediarios y las subrutinas con que se ha completado ASTRA para hacer un modelado integral de los plasmas del TJ-II. En los códigos fuente de las subrutinas de ASTRA y en los intermediarios (que no son sino nuevas subrutinas o pequeños programas precompilados) se ha procurado incluir comentarios suficientes para facilitar la tarea de quien los vaya a editar. A pesar de todo, el usuario que quiera hacer análisis de transporte con ASTRA no puede saber de antemano qué códigos fuente son relevantes ni cuáles son sus funciones o cómo se espera trabajar con ellos. Hemos redactado este informe para informar al respecto.

Usaremos la siguiente notación basada en la propia de ASTRA (se recomienda tener a mano el manual de usuario). Las constantes escalares generales en ASTRA se llaman CFx ó CVx, donde x es un número entre 1 y 16. Aquí nos referiremos a ellas como Cx, Cy, etc para aludir a constantes que el usuario podrá elegir y cambiar interactivamente* durante la ejecución de ASTRA. Lo importante realmente es que se trata de escalares. En cuanto a los arreglos matriciales, ASTRA dispone de funciones llamadas CARx, en este caso numerados desde 1 hasta 32 y los referiremos con esta notación. El resto de notaciones se ha tomado directamente de ASTRA. Desde ASTRA se escriben las subrutinas indicando el intervalo de tiempo de llamada, tiempo inicial, tiempo final y tecla de llamada interactiva. Aquí indicaremos sólo la tecla de llamada, cuya elección es arbitraria.

Los ejemplos que siguen se han desarrollado en la computadora "fenix" del Centro de Cálculo del Ciemat (fenix.ciemat.es) [**5**]. Estos programas, incluido ASTRA, se encuentran instalados también en la computadora "euler" (euler.ciemat.es) del mismo Centro. Suponemos que el usuario corre ASTRA desde un directorio local `$HOME/astra`.

El informe se estructura de la siguiente manera: los primeros apartados (§2–§4) llevan por título el del código principal que se va a usar en el programa periférico, y en sucesivas secciones se describe brevemente la subrutina de ASTRA correspondiente, se informa sobre su uso, sobre las variables usadas y sobre las precauciones necesarias al incorporarla a un modelo y, si existe, se da el nombre del "manualillo" donde se dan detalles sobre el código intermediario que acopla

* Esto no siempre es posible. Si se asignan valores a las constantes dentro del modelo ASTRA, el cambiarlas durante la ejecución no tiene efecto porque se sobreescriben en cada paso temporal con el valor asignado en el modelo.

la subrutina de ASTRA al código principal**. En el §5 habalmos de las subrutinas de ASTRA que obtienen el campo eléctrico ambipolar en base a calcular los flujos neoclásicos de electrones e iones. Cada una de estas secciones §2–§5 acaba con un ejemplo concreto de cálculo. El §6 es un ejemplo de escritura de modelo ASTRA –es decir, escrito en el lenguaje de programación de ASTRA [**1**]– que usa varios de los programas periféricos descritos. Acabamos con una previsión de futuro a corto plazo para ubicar mejor este informe en el curso de las tareas relacionadas.

2. EIRENE: Cálculo de la fuente de partículas por inyección de gas y reciclado de la pared

2.1. Subrutina ASTRA: `eirene.f`

Su función es tomar datos de ASTRA, escribirlos en el directorio apropiado para que EIRENE pueda tomarlos como datos de entrada y, mediante una llamada al sistema, arrancar el programa local `$HOME/drey/driver_eirene`. Como veremos más abajo, en `eirene.f` se definen varios valores que puede interesar modificar.

El código fuente `driver_eirene.f` hay que compilarlo localmente usando
`-L/disco02/fusion/guaspx/EIRENE-2004_new/conexdir -lconex`
(véase el manualillo `conex_eir.help`, que se encuentra normalmente en el mismo directorio `$HOME/drey/`) de manera que enlace con el programa **conex_eir**, que es el que de hecho arranca EIRENE. Desde 2009 hay tres subrutinas en la librería "conexdir" creada por J. Guasp:

1. **conex_eir** para utilizar pared litiada.
2. **conex_eirb** que usa Boro
3. **conex_eirhel** que es para plasmas de He.

Es muy importante recordar que la llamada a una u otra se produce en **driver_eirene**, de manera que si se han hecho modificaciones hay que compilar `driver_eirene.f` para que se cree el nuevo programa ejecutable **driver_eirene**.

2.1.1. Uso de la subrutina

Desde el modelo ASTRA se llama escribiendo
`EIRENE(Cx,CARy)::::S;`
donde:

- Cx: Entrada. Tiempo de confinamiento de las partículas [s]. Lo normal es definir Cx=TAUPB en el modelo ASTRA.
- CARy: Salida. Perfil de distribución de la fuente de electrones [10^{19} $m^{-3}s^{-1}$].

Hay muchos parámetros y datos que se puden mandar a EIRENE para que calcule en base a las opciones que simulan las condiciones experimentales. En la versión que describimos aquí, algunos de esos parámetros se definen dentro de `eirene.f` (ver el manualillo `conex_eir_v1_help`), de manera que si el usuario quiere cambiarlos debe editar la subrutina. Se trata de:

- n_proc – Número de procesadores (normalmente n_proc=24)
- n_tray – Número de trayectorias (se recomienda n_tray=100000)
- n_octantes – Número de octantes en los que se va a hacer el cálculo (normalmente n_octantes = 2).
- malla_eir – Número de puntos radiales de la malla que usará EIRENE (normalmente malla_eir = 36).
- zlim1 – Posición [cm] del limitador 1 (sector A3 del TJ-II). Actualmente se lee de la variable ZRD46 de ASTRA.

** Estos "manualillos", o documentos de ayuda, suelen encontrarse en las propias máquinas de cálculo. Todos los disponibles han sido redactados por J. Guasp, pues es quien ha desarrollado algunos de los códigos intermediarios que se usan en este informe precisamente con el objeto de hacer de uso público los códigos princiales EIRENE y FAFNER.

- zlim2 – Idem limitador 2 (sector C3 del TJ-II). Se toma de la variable ZRD47.
- rcycl(1) – Coeficiente de reciclado en el limitador helicoidal. Fijado en `eirene.f`
- rcycl(2) – Idem limitador poloidal 1 (sector A3 del TJ-II). Id.
- rcycl(3) – Idem limitador poloidal 2 (sector C3 del TJ-II). Id.
- flxpuff(1) – Flujo de inyección de gas frío, válvula 1 [A atómicos]. Definida actualmente como ZRD48/8*n_octantes.
- flxpuff(2) – Id. válvula 2. Actualmente fijado a cero.

Otros parámetros que pueden cambiarse en la llamada a EIRENE son las potencias respectivas de ECRH y de NBI. Las configuraciones disponibles para EIRENE se encuentran en archivos: `/disco02/fusion/guaspx/EIRENE-2004_new/geom_*`
Véase el manual de EIRENE (Ref. [**6**]) para conocer otras posibilidades de cálculo con EIRENE.

A diferencia de los cambios que se hagan en el código `driver_eirene.f`, que exigen compilarlo otra vez, aquí no es necesario porque ASTRA comprueba si ha habido cambios en la carpeta de subrutinas (donde se encuentra `eirene.f`) y compila para actualizar la librería de usuario antes de crear el ejecutable del modelo ASTRA.

2.1.2. Precauciones En primer lugar, hay que asegurarse de que los parámetros relevantes están bien definidos dentro de `eirene.f`, ver arriba. En particular, hay que observar la configuración magnética. También debe prestarse atención al tipo de pared.

El programa protesta en la llamada a **conex_eir** (o las otras dos opciones, ver §2.1) si el fichero experimental no provee algunos datos. Por ejemplo: si no se dan valores ZRD46, ZRD47 y ZRD48 (asignados respectivamente a zlim1, zlim2 y flxpuff) en el archivo experimental de ASTRA, la llamada a EIRENE retorna el mensaje

```
cp: cannot stat ‘deirxx/fort.198’: No such file or directory
Error          -1
Error en Eirene          -1
```

También debe tenerse cuidado con el tiempo de confinamiento de las partículas con el que inicialmente se llama a EIRENE, pues un valor desmedido (como sucede a veces con los valores iniciales) podría dar problemas en el cálculo. Una posibilidad es definir una fuente inicial de partículas (esto se hace en el archivo experimental `$HOME/astra/exp/*`), si bien habrá que cuidar de eliminarla tras el cálculo de EIRENE para que no se añada artificialmente.

Manualillos: `eirene_for_tj2_v11.doc`, `conex_eir.help`, `conex_eir_v1_help`.

2.1.3. Variables en ASTRA Como se ha visto arriba, estamos usando las variables de ASTRA llamadas ZRD46, ZRD47 (posiciones de los limitadores) y ZRD48 (inyección de gas) para definir algunos valores que usa EIRENE. Estas variables deben definirse en los archivos experimentales y pueden ser dependientes del tiempo, algo especialmente útil para simular la inyección de gas.

2.2. Subrutina ASTRA: `perf_neut.f`

Su función es activar el código intermediario **driver_perfn2** (en la carpeta `$HOME/drey/`), encargado de calcular la distribución de la fuente, tras haber hecho al menos una llamada a EIRENE, de manera rápida en tanto los perfiles del plasma no difieran mucho de aquellos con los que ha calculado EIRENE. El programa **driver_perfn2** usa archivos voluminosos generados por EIRENE.

2.2.1. Uso de la subrutina Desde el modelo ASTRA se usa

```
EIRENEPN(Cx,Cy,CARz)::::F
```

donde Cx es, como en la llamada a EIRENE, el tiempo de confinamiento. Por eso conviene definir igualmente Cx=TAUPB. La constante Cy es la densidad de línea, de manera que

normalmente usaremos Cy=NELAC. El arreglo CARz es, como antes, la distribución de la fuente que devuelve la subrutina. Aquí hemos elegido la tecla "F" para activar interactivamente el proceso desde ASTRA.

2.2.2. Precauciones Como se advierte en el manualillo, esta subrutina debe usarse con la precaución de que los perfiles del plasma y los tiempos de confinamiento no difieran mucho de los usados al correr EIRENE previamente para generar los archivos que usa `perf_neut`. Si estas diferencias crecen, conviene repetir la llamada a EIRENE.

Es conveniente que los ficheros de datos `/astra/exp/*` definan valores para ZRD46, ZRD47 y ZRD48. Aunque `perf_neut.f` y los códigos intermediarios llamados no usan esta información, es muy posible que haya que actualizar el cálculo con EIRENE, que sí la usa. La experiencia es que dos descargas distintas, aunque tengan perfiles parecidos, requieren un cálculo con EIRENE antes de poder usar sin problemas `perfil_neutr.f`.

Esta subrutina se construyó y probó en los casos de pared borada. No debería cambiar mucho si se usa para los casos de pared litiada aunque, insistimos, conviene alejarse lo menos posible de las condiciones en las que EIRENE ha calculado.

Manualillo: `conex_eir_v1_help`.

2.3. Ejemplo: evolución de la densidad en plasmas NBI con pared borada

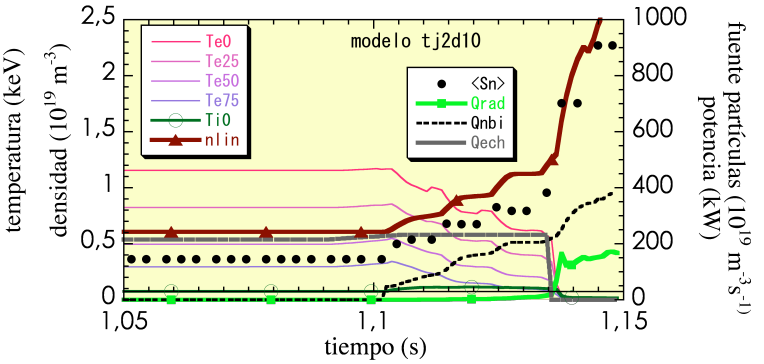

FIGURA 1. *Simulación de un plasma ECRH estacionario sobre el que se inicia un calentamiento NBI ($t = 1100$ ms) dando lugar al crecimiento de la densidad. Al llegar a densidades de corte para el ECRH el plasma se apaga debido a la radiación.*

La figura 1 es el resultado de un cálculo en el que se tomaron datos experimentales de una descarga ECRH del TJ-II en estado estacionario para hacer una prueba de evolución del plasma al entrar la inyección de neutros rápidos. El objetivo del cálculo era estimar el valor de la fuente de reciclado en los plasmas NBI con el interior de la cámara de vacío borada. Especialmente en las primeras descargas de este tipo, era común que el plasma se apagara prácticamente al llegar a la densidades de corte de ECRH, en torno a densidades de línea $\bar{n}_e \approx 1{,}2 \times 10^{19}$ m^{-3}.

Estos cálculos se hicieron combinando las dos llamadas,

```
Particle source from Monte-Carlo code
CV16=TAUPB
CV15=NELAC
EIRENE(CV16,CAR14):.01:0.103::S;
EIRENEPN(CV16,CV15,CAR14):0.005:0.1::F;
```

donde, como se ve, EIRENEPN (la subrutina "rápida") se llama cada 5 ms, mientras que el verdadero cálculo Monte-Carlo con EIRENE se hace cada 10 ms de tiempo simulado. Obsérvese que empiezan a ser llamadas en tiempos algo distintos para intercalarlas, así como que se puede usar el mismo arreglo para almacenar el retorno de las subrutinas.

Para explicar el aumento incontrolado de la densidad en estas condiciones de pared, incluso considerando modelos de transporte –como es el caso– en los que no aumenta el tiempo de confinamiento de las partículas al pasar a la fase NBI, es necesario añadir una fuente de partículas frías de magnitud semejante a las vertidas por el haz en bruto, es decir, considerando un neto de $S_{\mathrm{haz,f}}[10^{19}\ \mathrm{m^{-3}s^{-1}}] \sim 0{,}625 I_{\mathrm{NBI}}$, donde I_{NBI} es la corriente nominal del haz [MA]. El aumento incontrolado de la densidad, según estos cálculos, se debe a la confluencia tanto de una fuente extra de partículas coincidente con la inyección del haz de neutros, como a un alto reciclado de la pared. El rápido apagado del plasma se debe al aumento repentino de la radiación.

3. FAFNER: Cálculo de los perfiles de deposición de energía y partículas por NBI

El cálculo de los perfiles de deposición provenientes de los haces de neutros rápidos (NBI) se ha ido actualizando con el tiempo en el TJ-II. Ya antes de la fase experimental se hicieron varios trabajos, por ejemplo relacionados con la evolución de las impurezas para estudiar escenarios de colapso radiativo [**8**]. Inicialmente, al notar que los perfiles de deposición dependían principalmente del promedio de la densidad electrónica [**9**], se desarrolló un programa de cálculo rápido (**calc_nbi**) basado en datos precalculados [**10**]. Gracias al desarrollo de librerías precompiladas para facilitar la conexión entre códigos, actualmente se puede lanzar directamente FAFNER, si bien hay que hacerlo tras haber obtenido las distribuciones de neutros mediante un cálculo apropiado con EIRENE.

A continuación nos referimos a ambos códigos intermediarios.

3.1. Subrutina ASTRA: `nbi_fafner.f`

Su función es activar el intermediario **calc_nbi** (J. Guasp), encargado de devolver una interpolación de los perfiles de deposición de NBI basada en un barrido de parámetros mediante el código FAFNER adaptado al TJ-II. La subrutina llama al código **fasolt**, que queda enlazado con **calc_nbi** al compilarlo así:

```
ifort -o fasolt fasolt.f /disco02/fusion/guaspx/datos_tj2/calc_nbi.o
```

El manualillo se llama `calc_nbi.help` y puede encontrarse en

```
fenix:/disco02/fusion/guaspx/datos_tj2/calc_nbi.help
```

3.1.1. Uso de la subrutina

Se incorpora al modelo ASTRA escribiendo

```
NBI_FAFNER(Cx,Cy,Cz,CARw,PIBM,SNEBM)::::H;
```

donde Cx, Cy y Cz son, respectivamente, los promedios en volumen de la densidad electrónica [10^{19} $\mathrm{m^{-3}}$], de la temperatura electrónica [keV] y de la temperatura iónica [keV]. En CARw se devuelve el perfil radial de absorción total del plasma (tanto iones como electrones). En vez de usar otros arreglos multiuso de ASTRA, hemos tomado directamente las variables PIBM (perfil radial de absorción de los iones) y SNEBM (perfil radial de nacimiento de los iones rápidos), aunque esto no es obligatorio. El perfil de absorción de los electrones se puede definir entonces como PEBM=CARw-PIBM. Esta manera de proceder sería incorrecta si el usuario quisiera también ejecutar la subrutina NBI propia de ASTRA, pues internamente usa estos mismos arreglos.

3.1.2. Precauciones FAFNER necesita la distribución de neutros para calcular la deposición de NBI porque las pérdidas por intercambio de carga (CX) pueden ser una fracción considerable de las pérdidas por órbitas rápidas [**9**, **11**] dependiendo de la cantidad de neutros.

Hay un modificador que se define, por ahora, dentro de la propia subrutina de ASTRA `nbi_fafner.f`, a saber:

- La configuración magnética

mientras que otros modificadores muy importantes se evalúan usando constantes de ASTRA que deben definirse en el "log" del modelo*:

- La energía principal del haz: CNB2
- La fracción de la energía principal: CNBI1
- La fracción de la energía mitad: CNBI2
- La fracción de la energía 1/3: CNBI3
- El grado de coinyección: CNBI4

También son necesarios los valores de la carga efectiva y de la carga de la impureza principal, que se toman de las variables de ASTRA

- ZEF(0)
- ZIM1(0)

y, por lo tanto, deben quedar definidas en el modelo.

Requisito de los ficheros de datos `$HOME/astra/exp/*` es que contengan la potencia total del haz de neutros: QNBI. Aquí se entiende que QNBI es la potencia de entrada al toro ("port through power") debida a *la suma de los dos inyectores*. Como en otras variables de intercambio con las subrutinas de ASTRA, usamos las unidades convenidas en ASTRA y la potencia se da en [MW].

3.2. Subrutina ASTRA: `nbi_fafner_v0.f`

Su función es llamar al intermediario **conex_faf**, provisto por J. Guasp, que es el verdadero encargado de ejecutar FAFNER. Para ello, el usuario debe compilar localmente el código `fasolt_v0.f`, que actualmente se encuentra siempre en el directorio local `$HOME/drfafner`, y que incluye la llamada a **conex_faf**. Un ejemplo de línea de compilación es

```
ifort -o fasolt_v0 fasolt_v0.f -L/home/guaspx/EIRENE-2004_new/conexdir -lconex
```

donde vemos que es necesario enlazar con la librería `conex` para poder usar **conex_faf**.

El programa **fasolt** se ocupa, además de llamar a **conex_faf**, de escribir el archivo de datos `$HOME/drfafner/fafner.input` que contiene los datos de entrada para FAFNER.

3.2.1. Uso de la subrutina En ASTRA la subrutina se llama `nbi_fafner_v0.f` y la manera de llamarla desde el modelo ASTRA es

```
NBI_FAFNER_v0(Cx,CNB1,j,PEBM,PIBM,SNEBM)::::H
```

donde los argumentos son

- Entrada:
  - Cx – tiempo de confinamiento de las partículas [s]
  - CNB1 – Potencial de aceleración del haz [kV]
  - j – Toma valores 1 ó 2 para escoger el inyector respectivo: NBI-1 (coinyección), NBI-2 (contrainyección)
- Salida:
  - PEBM – Perfil de potencia calórica a los electrones
  - PIBM – Id. iones

---

* Recordemos que las diversas constantes y variables de un modelo en ASTRA pueden editarse durante la ejecución. Si el usuario las salva, quedan almacenadas en un archivo dentro de la carpeta `$HOME/astra/equ/log/`.

- SNEBM – Perfil fuente de nacimiento en el plasma de iones del haz. Se define como `625.*perfHr/(fr1+fr22/2+fr3/3)/V_haz` siendo `perfHr` el perfil de deposición de iones rápidos y `V_haz` el potencial de aceleración del haz (para el que hemos usado CNB1 como variable de entrada).

Puesto que ASTRA admite llamar varias veces a la misma subrutina en cada paso temporal, la manera de tener dos inyectores es, por ejemplo, escribir en el modelo ASTRA:

```
NBI_FAFNER_v0(Cv16,CNB1,1,PEBM,PIBM,SNEBM):0.001:9999::H;
NBI_FAFNER_v0(Cv16,CNB2,2,CAR20,CAR25,CAR29):0.001:9999::I;
```

de manera que las teclas "H" (inyector 1) e "I" (inyector 2) permiten actualizar el cálculo correspondiente a cada inyector *ad libitum*. En este ejemplo, Cv16 contiene el tiempo de confinamiento de las partículas –obviamente común a ambos inyectores– y las constantes CNB1 y CNB2 se han usado para almacenar los respectivos voltajes de aceleración.

3.2.2. Precauciones La potencia de NBI debe definirse mediante la variable ASTRA "QNBI", y los parámetros de cada inyector mediante alguna variable auxiliar como en el ejemplo de antes.

En `nbi_fafner_v0.f` se usa la constante predeclarada CNBI4 de ASTRA como fracción de coinyeción, pues la potencia del haz se define así en la subrutina:

```
Potencia del inyector 1 = CNBI4*QNBI
Potencia del inyector 2 = (1-CNBI4)*QNBI
```

Las fracciones de energía E, E/2 y E/3 también están asignadas a constantes de ASTRA dentro de `nbi_fafner_v0.f`:

```
CNBI1 <-- fracción de la energía principal
CNBI2 <-- fracción de la energía mitad
CNBI3 <-- fracción de la energía 1/3
```

La configuración también se define en la subrutina. Habrá que cambiarla ahí si lo pide el caso editando `nbi_fafner_v0.f`. No hay que compilar porque lo hace ASTRA al generar el ejecutable.

Para hacer los cálculos se leen las distribuciones radiales de átomos y moléculas provenientes de una ejecución de EIRENE. Ojo a esto, pues conviene que sea del mismo plasma (configuración, gas y pared). Los datos se almacenan en los ficheros locales `$HOME/drey/fort.143` y `$HOME/drey/fort.144`.

Manualillo: Se llama `conex_faf_v0.help` y puede encontrarse en los directorios de EIRENE:

```
euler:/home/guaspx/EIRENE-2004_new/conexdir/conex_faf_v0.help
fenix:/disco02/fusion/guaspx/EIRENE-2004_new/conexdir/conex_faf_v0.help
```

3.2.3. Variables en ASTRA Resumimos las variables de ASTRA que se usan internamente al usar este programa periférico:

```
QNBI <-- Potencia total nominal NBI1+NBI2
CNBI1, CNBI2, CNBI3 <-- Fracciones de las energías del haz
CNBI4 <-- Fracción de coinyección
```

3.3. Ejemplo: canalización de la potencia de NBI dependiendo del reparto energético del haz

La figura 2 muestra el cálculo de la potencia transferida a distintos canales (transparencia o "shine through", `Psh`; órbitas rápidas perdidas, `Plos`; calentamiento electrónico, `Pel`; calentamiento iónico, `Pion`) dependiendo de la composición energética, o reparto, del haz para un plasma NBI del TJ-II. Los distintos casos se han obtenido cambiando los coeficientes CNBI1, CNBI2 y CNBI3 para que correspondan a los repartos siguientes:

- `8-1-1`: 30 keV (80 %); 15 keV (10 %); 10 keV (10 %)
- `6-2-2`: 30 keV (60 %); 15 keV (20 %); 10 keV (20 %)

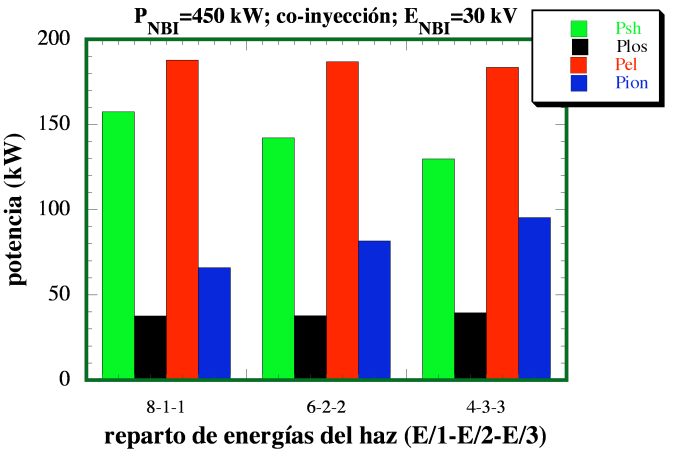

FIGURA 2. *Canalización de la potencia de NBI dependiendo del reparto energético en la composición del haz de neutros: "shine through", `Psh`; pérdidas por órbitas rápidas, `Plos`; calentamiento electrónico, `Pel`; calentamiento iónico, `Pion`.*

- `4-3-3`: 30 keV (40 %); 15 keV (30 %); 10 keV (30 %)

Los parámetros de la descarga se han tomado de la nº 18998, correspondiente a un plasma NBI con densidad central $n(0) \approx 3{,}2 \times 10^{19}$ m^{-3} (densidad de línea $\approx 1{,}7 \times 10^{19}$ m^{-3}), y temperaturas electrónica e iónica centrales $T_e(0) \approx 0{,}28$ keV y $T_i(0) \approx 0{,}13$ keV respectivamente. Como se advierte arriba, se tomó la precaución de calcular previamente la distribución de átomos y moléculas mediante una ejecución de EIRENE.

Se encuentra que las pérdidas por transparencia del plasma frente al haz decrecen conforme es menor la fracción energética alta (30 keV). La potencia cedida a los electrones disminuye ligeramente cuando esta fracción disminuye pero aumenta la cedida a los iones. En todo caso las diferencias no son drásticas. Las formas de los perfiles de deposición de la potencia total (calentamiento electrónico más iónico) no resultan apenas diferentes entre los distintos repartos de energías del haz.

4. *TRUBA: Cálculo de las fuentes de calentamiento por ECRH y EBWH*

4.1. *Subrutina ASTRA:* `truba.f`

Su función es escribir los datos de perfiles del plasma que usará TRUBA, así como ejecutar los procesos `$HOME/drtruba/bsim` y `$HOME/drtruba/bsimP`. Mediante el proceso `bsim` se determinan las condiciones exactas de lanzamiento y distribución de los rayos del haz de microondas. `bsim` necesita también diversos parámetros que se definen en `$HOME/drtruba/bsim.dat`. Algunos de ellos son los siguientes:

- Elección del sistema de calentamiento, ya sea el de 53,2 GHz o el de 28,0 GHz
- Ángulo del espejo interior del sistema elegido
- Frecuencia del haz [GHz]
- Modo de polarización (modo-O, modo-X)
- Cintura del haz [m]
- Posición del foco del haz respecto a un origen dado [m]
- Número de rayos y distribución espacial
- Paso de integración

El programa `bsimP` lee todas las salidas de TRUBA y compone un resultado final en base a todos los rayos independientes. Además prepara archivos de salida listos para su representación gráfica.

Para mayor información, se recomienda acudir al manual del código TRUBA [**12**].

4.1.1. Uso de la subrutina La llamamos escribiendo

```
TRUBA(CARx,CARy)::::T
```

en el correspondiente modelo ASTRA. Los arreglos que devuelve son

- CARx – Perfil normalizado de deposición de potencia ECRH
- CARy – Perfil normalizado de deposición de potencia EBWH

Al estar normalizados, los perfiles deben multiplicarse por las correspondientes potencias. Por ejemplo, en el modelo ASTRA escribiríamos

```
PEECR=QECR*CARx
PEFW=QFW*CARy
```

siendo PEECR, PEFW dos perfiles predeclarados en ASTRA; y QECR y QFW dos variables que pueden estar definidas (y depender del tiempo) en los archivos experimentales de cada descarga.

TRUBA calcula a la vez la contribución electromagnética y la electrostática independientemente de que, dependiendo de las condiciones de lanzamiento se pueda favorecer la aparición de sólo una de ellas. El usuario debe tener claro qué tipo de sistema y con qué condiciones de plasma va a hacer el cálculo para saber qué tipo de calentamiento cabe esperar.

4.1.2. Precauciones Por lo comentado arriba, hay que definir el tipo de cálculo en `$HOME/drtruba/bsim.dat`.

Requisitos de los ficheros de datos `/astra/exp/*`: tener definidas las potencias QECR y QFW, o las variables que correspondan según la escritura del modelo.

4.2. Subrutina ASTRA: `vashra.f`

La subrutina `vashra.f` no es más que una adaptación de `truba.f` para que el cálculo se solicite a recursos distribuidos. Por lo demás, el uso desde ASTRA es idéntico en lo que a precauciones y sentido de los parámetros respecta. Por ejemplo, los archivos que ASTRA –a través de `vashra.f`– escribe para que sirvan de entrada a TRUBA, se escribirán en la carpeta `/$HOME/drvashra/` en vez de en `/$HOME/drtruba/`, pero esto es transparente para el usuario. Otra diferencia con `truba.f` es que la subrutina, en vez de directamente los intermediarios **bsim** y **bsimP** (ver §4.1), pone en marcha el ejecutable **gwcom** que, en resumen, se ocupa de sacar los datos a un servidor –donde se lanzará adecuadamente TRUBA de manera distribuida usando la infraestructura desarrollada para MaRaTra [**13**]– y retornarlos a ASTRA tras la ejecución de TRUBA. Más información sobre los intermediarios que usa `vashra.f` se puede encontrar en las Refs. [**14**, **15**].

4.2.1. Uso de la subrutina Como venimos diciendo, de cara al usuario todo es idéntico excepto que para llamarla desde un modelo ASTRA escribiremos

```
VASHRAT(CARx,CARy)::::T
```

donde los arreglos etc. son como se ha explicado en el §4.1.1.

4.2.2. Precauciones Se aplica lo advertido en el §4.1.2. Además, dado que `vashra.f` usa recursos distribuidos, es necesario disponer de los correspondientes permisos.

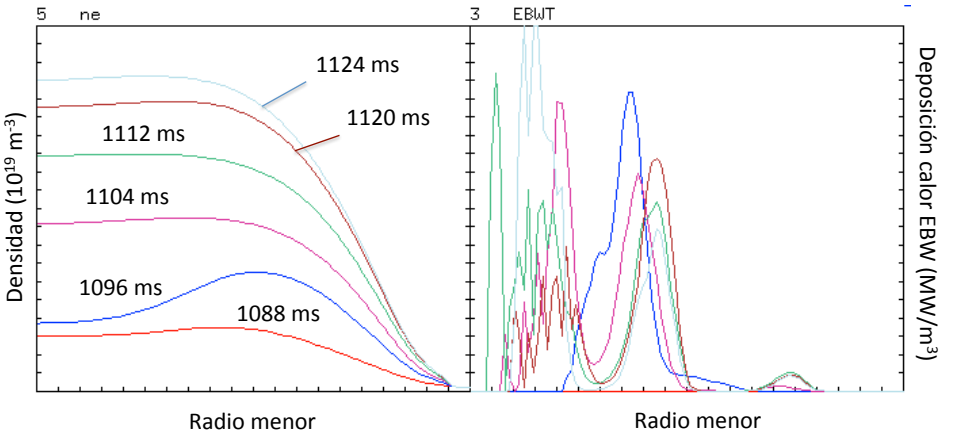

FIGURA 3. *Evolución de los perfiles de densidad electrónica (izda.) y de potencia de calentamiento Bernstein (dcha.) en una simulación donde se aumenta repentinamente el gas inyectado por las válvulas ($t = 1090$ ms) sobre un plasma ECRH estacionario.*

4.3. Ejemplo: evolución del perfil de calentamiento Bernstein con la densidad del plasma

En este caso mostramos un cálculo de evolución del perfil de deposición EBWH obtenido mediante llamadas a TRUBA cada 1 ms de tiempo simulado y usando 97 rayos. Para esto se utilizaron recursos distribuidos accediendo mediante llamadas a la subrutina `vashra.f` en el modelo de ASTRA. Se estudia el cambio del perfil de deposición dependiendo de la densidad del plasma. La figura 3 muestra varios instantes de la evolución, en la que se ha forzado un fuerte aumento de la densidad ($t = 1090$ ms) mediante una inyección repentina de gas frío. Se observa cómo, al alcanzarse una cierta densidad ($t \approx 1104$ ms), se forma un pequeño pico de deposición en la mitad exterior del plasma –las abscisas son el radio menor normalizado– mientras la máxima densidad de potencia se desplaza hacia el interior. Cerca del eje magnético (radio cero) el perfil de densidad de potencia está peor definido y presenta picos que no corresponden al modelo de calentamiento. Para mejorar la definición en esa zona hay que aumentar el número de rayos, si bien afecta poco a la potencia integrada porque el volumen correspondiente es pequeño.

5. TRANSPORTE NEOCLÁSICO: Cálculo del campo eléctrico radial por ambipolaridad en los flujos colisionales

5.1. Subrutina ASTRA: `flux_Efield.f`

Su función es calcular coeficientes de transporte neoclásico y el correspondiente campo eléctrico radial ambipolar. La estructura de la subrutina es tal que los flujos pueden provenir de formulaciones disponibles o de interpolaciones basadas en datos numéricos. La manera de llamarla desde ASTRA es la siguiente

```
FLUX_EFIELD(j,CARx,CARy,CARz)::::X;
```

donde `j` es un *entero* que indica el modelo de flujos neoclásicos que se va a usar. Esto, por supuesto, depende de los que haya disponibles dentro de `flux_Efield.f`. En la actualidad hay dos, ambos analíticos, de manera que

1. j=1 – Modelo de Beidler [**16**, **17**]
2. j=2 – Modelo de Kovrízhnyj [**18**]

Con otros valores la subrutina no devuelve nada. Los arreglos en la llamada son

- `CARx` – Coeficiente neoclásico diagonal de difusión de los electrones [m^2/s]
- `CARy` – Coeficiente neoclásico diagonal de difusión térmica de los electrones [m^2/s]
- `CARz` – Coeficiente neoclásico diagonal de difusión térmica de los iones [m^2/s]

El campo eléctrico se almacena directamente en la variable ER [V/m] del sistema ASTRA. El modelo ASTRA da por hecha la ambipolaridad y hace evolucionar la densidad electrónica, si bien puede haber ecuaciones de evolución para las impurezas. Lo normal, entonces, será usar en el modelo coeficientes de transporte

```
DN = CARx + otros coeficientes
HE = CARy + otros coeficientes
XI = CARz + otros coeficientes
```

y definir la densidad iónica en el modelo, por ejemplo a través de la carga efectiva y algún perfil de impureza del plasma. En el ejemplo que mostraremos luego se considera un plasma cuya única impureza es carbono, tomando como perfil de densidad

$$n_{\mathrm{C}} = \frac{n_e(Z_{\mathrm{ef}} - 1)}{Z_{\mathrm{C}}(Z_{\mathrm{C}} - 1)}.$$

La densidad iónica entonces es

$$n_i = n_e \frac{Z_{\mathrm{C}} - Z_{\mathrm{ef}} + 1}{Z_{\mathrm{C}}},$$

expresión bien aproximada cuando Z_{ef} es pequeña en comparación con la carga de la impureza.

Como se ve, de momento sólo se pasan algunos coeficientes diagonales. Esto puede ampliarse de manera bastante inmediata editando `flux_Efield.f` y adaptando correspondientemente la forma de la llamada desde ASTRA. El archivo fuente de esta subrutina contiene todos los códigos que necesita a excepción del resolvedor de ecuaciones diferenciales **lsode**, incorporado en ASTRA (depende de la versión; ver §5.1.1).

Los dos modelos neoclásicos actualmente disponibles usan los perfiles de rizado toroidal, ϵ_t, y helicoidal, ϵ_h A diferencia de la subrutina del §5.2, en los modelos de `flux_Efield` estos perfiles se encuentran definidos internamente en función de la coordenada normalizada ρ:

$$\epsilon_t(\rho) = 0{,}095\rho^{0{,}55}$$

$$\epsilon_h(\rho) = 0{,}134\rho^{0{,}61}.$$

Aunque esto no es restrictivo salvo que se deseen analizar configuraciones extremas, costaría poco incluir estos perfiles como argumentos de entrada para `flux_Efield`.

5.1.1. Precauciones En `flux_Efield.f`, y dependiendo de la opción de modelo, se usan unas u otras formulaciones para los flujos neoclásicos. Es posible que tales causen el fallo del integrador (**lsode**) si algunas magnitudes no están correctamente definidas en el modelo ASTRA, esto es, en el modelo de transporte. Por ejemplo, ambos modelos [**16**, **18**] usan la transformada rotacional. Si no está bien definida, **lsode** aborta el programa.

Hay que definir ZEJ, pues la subrutina `flux_Efield.f` usa esta variable de ASTRA para incluir en los cálculos el número atómico del ión principal del plasma. Normalmente ZMJ=1.

En las últimas versiones de ASTRA, que trabajan en doble precisión, debe usarse la subrutina **dlsoda**. Por lo tanto, si se desea portar `flux_Efield.f` a una versión reciente de ASTRA hay que editar la subrutina y hacer el cambio.

Importante: esta subrutina está en desarrollo. Es posible que además de la propia subrutina, el usuario deba incluir en su directorio `$HOME/astra` otras funciones auxiliares de `flux_Efield.f`. Por ejemplo, el modelo de la Ref. [**18**] (opción `j=2`) necesita funciones de Bessel que se han escrito en `$HOME/astra/fnc/bessi.f`.

Requisitos de los ficheros de datos `/astra/exp/*`: depende del modelo. Por ahora basta con proporcionar la transformada rotacional (llamada "MU" en ASTRA). En el caso del TJ-II,

lo habitual es definir la transformada rotacional de vacío (llamada "MV" en ASTRA) en el correspondiente archivo de datos experimentales.

5.2. Subrutina ASTRA: `nhripple.f`

Se trata realmente de un conjunto de subrutinas desarrolladas en base a los modelos usados en la tesis doctoral de J. García [**19**] y resumidos en la Ref. [**20**]. En la actualidad se usan por separado porque la manera de alcanzar la condición de ambipolaridad es distinta a la de `Flux_Efield.f`. La formulación de los flujos neoclásicos usada aquí se detalla en la Ref. [**21**] y ha resultado más ajustada a los plasmas del TJ-II [**20**, **22**] que las primeras pruebas con los modelos anteriores [**23**], si bien estos trabajos están en desarrollo.

La subrutina `nhripple.f` es, en realidad, un conjunto de subrutinas que originalmente se llamaban sin argumentos. Sus nombres son

`iniripp.f`
`tnripl.f`
`erstep.f`
`trdneo.f`

Actualmente se deben pasar los perfiles de rizado helicoidal y toroidal como argumentos de la subrutina `tnripl.f`. Por ejemplo, si se desea obtener el campo eléctrico radial usando esta formulación neoclásica, en el modelo de ASTRA habrá que definir la línea

`iniripp():; tnripl(CARx,CARy):; erstep():; trdneo():`

El campo eléctrico radial se almacena directamente en el arreglo *ad hoc* de ASTRA, es decir, ER. Los coeficientes de transporte se encuentran en arreglos internos de las subrutinas:

- `YCAR27` – Difusividad térmica electrónica (parte antisimétrica del cálculo); en `tnripl.f`
- `YCAR31` – Id. (parte simetrica); en `trdne.f`
- `YCAR28` – Difusividad térmica iónica (parte antisimétrica del cálculo); en `tnripl.f`
- `YCAR32` – Id. (parte simetrica); en `trdne.f`

de manera que las respectivas difusividades térmicas neoclásicas son $\chi_e^{\mathrm{NC}} = \mathrm{YCAR27} + \mathrm{YCAR31}$ y $\chi_i^{\mathrm{NC}} = \mathrm{YCAR28} + \mathrm{YCAR32}$. Sería fácil sacar estos coeficientes como argumentos de las llamadas a las correspondientes subrutinas, pero se ha preferido presentar `nripple.f` en el estado actual dado que no está decidido en qué manera actualizarla. También se pueden renombrar –así se encontraba originalmente– los arreglos de manera que automáticamente se tenga $\chi_e^{\mathrm{NC}} = \mathrm{CAR27} + \mathrm{CAR31}$ y $\chi_i^{\mathrm{NC}} = \mathrm{CAR28} + \mathrm{CAR32}$, pero es peligroso hacer que las subrutinas usen arreglos multiuso de ASTRA, pues no son visibles desde el modelo ASTRA y es fácil sobreescribirlos accidentalmente.

5.2.1. Precauciones Las distintas subrutinas usan los perfiles de rizado toroidal y helicoidal. El procedimiento normal es que se encuentren definidos en el archivo experimental de la descarga que se desee analizar. Por ejemplo, si se usan los arreglos `CAR15` y `CAR16` para almacenar respectivamente ϵ_t y ϵ_h, entonces la llamada sería `tnripl(CAR15X,CAR16X):`.

Debe estar definido el arreglo "MVX" de ASTRA, donde se almacena el perfil de vacío de la transformada rotacional. También se usa internamente el arreglo "NIZ1" (densidad de una impureza a decidir), pero no hay problema si es nulo.

Para portar el código periférico a otras máquinas el usuario debe copiar, además de las subrutinas, otras que son llamadas desde ellas y que se encuentran también en el directorio `$HOME/astra/sbr/` agrupadas en el archivo `ssl.f` (se trata de una pequeña recopilación de funciones de uso común).

5.2.2. Variables en ASTRA En resumen, la llamada a las subrutinas que componen `nhripple.f` usa los perfiles MV (transformada rotacional) y NIZ1 (densidad de la impureza principal) de ASTRA. Además, el fichero experimental –o el modelo ASTRA– debe tener definidos los perfiles de rizado helicoidal y toroidal.

5.3. Ejemplo: comparación de los modelos neoclásicos

Veamos una comparación del comportamiento de los tres modelos disponibles: [**16**] (Beidler 1996), [**18**] (Kovrízhnyj 2006) y [**21**] (Hastings 1985); usando perfiles de una descarga del TJ-II en fase de NBI, es decir, con la densidad en corte para el ECRH (densidad de línea $\bar{n} = 2{,}3 \times 10^{19}$ m^{-3}) y temperaturas bajas ($T_e(0) \approx 250$ eV; $T_i(0) \approx 120$ eV). Como muestra la Fig. 4 (izda.), los tres modelos dan lugar a un potencial electrostático negativo en todo el plasma de acuerdo con los datos experimentales (no se muestran), si bien hay notables discrepancias que se deben a la integración de los perfiles de campo eléctrico (Fig. 4, dcha.) en la parte del borde del plasma. Recordemos que la salida de `flux_Efield` es el campo eléctrico.

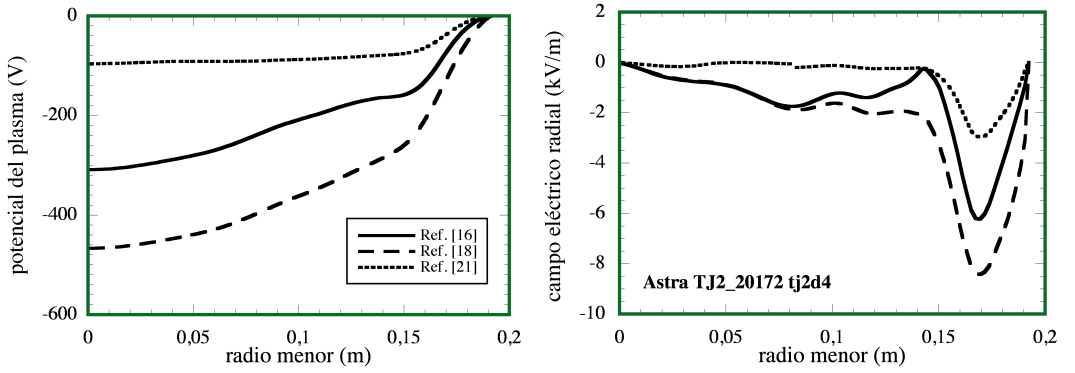

FIGURA 4. *Comparación de los modelos neoclásicos de las referencias mostradas. Se han usado datos de la descarga del TJ-II n^o 20172 en fase NBI.*

Los tres modelos predicen campos positivos, usando perfiles de plasmas ECRH, también de acuerdo con los experimentos. En base a los perfiles de la descarga ECRH n^o 17406 se obtienen campos eléctricos que cambian de signo, de positivos en el interior a negativos en el exterior, en posiciones radiales dependiendo del modelo pero dentro del recorrido mostrado en la Fig. 5. Allí usamos la misma colisionalidad normalizada $\nu^* = 2\pi\nu_e R_0/\iota v_t$ que en la Ref. [**24**], donde ν_e es la colisionalidad electrónica, $R_0 = 1{,}5$ m el radio mayor del TJ-II, ι la transformada rotacional en radianes y v_t la velocidad térmica electrónica. Aunque se trata de primeros cálculos, estos resultados sugieren que el campo eléctrico radial en los plasmas del TJ-II siguen pautas esperables en base al transporte neoclásico. Si las temperaturas electrónicas son suficientemente altas, los tres modelos predicen campos eléctricos y potenciales positivos en todo el plasma. Estos resultados se pueden analizar respecto de las soluciones algebraicas de la ecuación de ambipolaridad (ver [**21**] y [**25**]).

6. Ejemplo de modelo ASTRA

A continuación escribimos un modelo de transporte en el lenguaje de ASTRA que usa varias de las subrutinas anteriores. Tal como se presenta puede correr –con el archivo experimental apropiado– un caso de descarga NBI del TJ-II permitiendo cálculos interactivos de la fuente de partículas (EIRENE) y de NBI para los dos inyectores (FAFNER), además del campo eléctrico según un modelo analítico. Sin embargo, se presenta con otras posibilidades –comentadas anteponiendo una exclamación en las correspondientes líneas– para mejor servir de ejemplos de uso. Además, es cómodo disponer en el modelo de opciones que se pueden "descomentar". Por ejemplo, el modelo que sigue presenta dos funciones gaussianas que pueden comentarse y a cambio activar la llamada a TRUBA, bien directamente, bien usando el cálculo distribuido

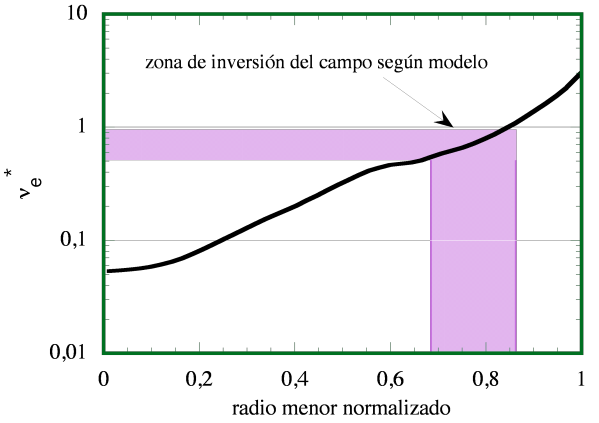

FIGURA 5. *Región radial (sombreada) en la que se invierte el signo del campo eléctrico según los tres modelos neoclásicos de la Fig. 4. Se han usado datos de la descarga del TJ-II nº 17406 en fase ECRH:* $\bar{n} = 0{,}66 \times 10^{19}\ m^{-3}$, *potencia de ECRH 590 kW.*

a través de Vashra-T. Recuérdese que, hasta la fecha (pero ver §7), la geometría magnética que se usa es la de vacío [**26**] y debe escribirse en los correspondientes archivos experimentales guardados en `$HOME/astra/exp/`.

El modelo incorpora versiones simplificadas del nivel promedio de fluctuaciones según teorías que no vienen al caso (véase [**28**] para más detalles) y a las que se destinan las funciones `F1` y `F4`. La primera se presenta activada porque es una contribución importante al transporte en el modelo, mientras que `F4` se deja como ejemplo de posible nueva ecuación de evolución rápida – escalas de la turbulencia. La función `F2` es el flujo poloidal y está desactivada. Corresponde a la preparación de un estudio en el que los flujos poloidales debidos al campo ambipolar neoclásico se incorporan a otras contribuciones, como las debidas a la generacion de flujos macroscópicos a partir de la turbulencia.

Las siguientes líneas corresponden al archivo `$HOME/astra/equ/tj2d10a`.

```
!! MODELO TJ2d10a (D. López-Bruna. CIEMAT, Ene-Feb 2009)

NEQUIL=-1;   ! Forzando la geometría de g3d (J. Guasp)

Constantes (los valores dados son orientativos)

!  CF1=0.1 ! Transporte mínimo en la difusividad De
!  CF2=0.1; ! Coeficiente en la parte Datr=CF2*Xe (modelo K-P) de De
!  CF3=0.1; ! alfa_2 en la cizalladura del modelo de turbulencia/transición
!  CF4=.05; ! Control de escala temporal en las ecuaciones rápidas.
!  CF5=0.5; ! fracción de QECRH en QTL1
!  CF6=.9; ! \Delta \rho: anchura del peldaño (FTANH)
!  CF7=.2; ! delta (FTANHF)
!  CF8=0.; ! enrasado (FTANHF)
!  CF9=100; ! altura (FTANHF)
!  CF10=0.2; ! Anchura deposición ECRH, QTL1
!  CF11=0.1; ! Anchura deposición ECRH, QTL2
!  CF12=0.2; ! Anchura gaussiana pump-out
!  CF13=...; ! Amortiguamiento viscoso, flujo poloidal
```

```
!  CF14=1.0 ! Saturación de la turbulencia (en alfa_1)
!  CF15=0.3; ! Kperp_rhos, o corrección a iota/2*pi en alfa_1
!  CF16; ! Auxiliar (tau_LHD)

!  CV1=10; ! Témino para reducir Xe en el borde
!  CV2=0.1; ! Coef. en la parte X_{e,atr} de colisionalidades bajas (K-P)
!  CV3=1.0; ! Término de supresión en la turbulencia ballooning.
!  CV4=100; ! Coef. en la parte X_{e,atr} de colisionalidades altas (K-P)
!  CV7=10; ! Coef. en la parte D_{e,atr} de colisionalidades altas (K-P)
!  CV10=0.02; ! Centrado ECRH, QTL1
!  CV11=0.00; ! Centrado ECRH, QTL2
!  CV12=0.30; ! Centrado gaussiana pump-out
!  CV13 ! Auxiliar: tiavb
!  CV14 ! Auxiliar: teavb
!  CV15 ! Auxiliar: NELAC
!  CV16 ! Auxiliar: TAUPB
!  CV17 ! Auxiliar: neavb

!  CHE2=1.; ! Nivel mínimo cte. de Xe
!  CHE3=1.; ! Nivel mínimo cte. de Xi
!  CHE4=0.1; ! Intensidad del pump-out por ECRH

!  CNB1=30; ! Energía del NBI-1 en kV
!  CNB2=30; ! Energía del NBI-2 en kV
!  CNB3=60; ! corriente del NBI-1 en kA
!  CNB4=60; ! corriente del NBI-2 en kA
!  CNBI1=0.8 ! fracción de la energía principal
!  CNBI2=0.1 ! fracción de la energía 1/2
!  CNBI3=0.1 ! fracción de la energía 1/3
!  CNBI4=0.5 ! tanto por uno de co-inyección

!  CNEUT1=1 ! Coef. en la fuente exp. de partículas
!  CNEUT2=0.8 ! Coef. en la fuente de partículas Inbi/(e Volumen).
!  CIMP1=1.2 ! Coeficiente para Zef
!  CRAD1=1.; ! Fraccion de QECR absorbida


Funciones auxiliares

! CAR1: Frecuencia de deriva omega*=(k_p rho_s) 3e5 sqrt(Te)/L_n
! CAR2: Gaussiana normalizada para ECRH (QTL1)
! CAR3: Libre. Ver Ec. momento toroidal
! CAR4: Gaussiana normalizada para ECRH (QTL2)
! CAR5: gradiente (-) de la presión electrónica
! CAR6: presión electrónica
! CAR7: grad Te
! CAR8: grad Ne
! CAR9: tanh para la velocidad de fuga de las partículas cerca del borde
! CAR10: crecimiento atrapadas =f_a^2 omega* (rho_s/Ln) sqrt(m_e/m_i)
! CAR11: Er gradPe (kV/m)
! CAR12: Término supresor en F1 =alfa2*(r/q*grad(q/r*Er) )**2 / gamma
```

```
! CAR13: Término de saturación en F1 = (k_perp rho_s)^2 / rho_s^2
! CAR14: Fuente de partículas de Eirene
! CAR15: gradPe experimental
! CAR16: Pe experimental NEX*TEX
! CAR17: Xe neoclásica de FLUX_EFIELD
! CAR18: Xi neoclásica de FLUX_EFIELD
! CAR19: De neoclásica de FLUX_EFIELD
! CAR20: PEBM NBI2
! CAR21: Pech de Truba ó Vashra-T
! CAR22: Pebw de Truba ó Vashra-T
! CAR23: Fafner, Perfil radial de absorción total del plasma
! CAR24: cizallamiento en F4 (turbulencia RB)
! CAR25: PIBM NBI2
! CAR26: Auxiliar para X_{e,atr}; colisionalidades altas modelo K-P
! CAR27: Auxiliar para X_{e,atr}; colisionalidades altas modelo K-P
! CAR28: X_{e,atr}; colisionalidades altas modelo K-P
! CAR29: SNEBM NBI 2
! CAR30: Crecimiento lineal para modelo ballooning (F4), sin definir
! CAR31: Auxiliar fuente de partículas del NBI
! CAR32: fuente de partículas del NBI


! Las llamadas a subrutinas pueden llevar los argumentos:
!     Delta_t:t_inicial:t_final:Tecla_llamada_interactiva

Calentamiento por Electron Cyclotron Heating (dos girotrones posibles)
----------------------------------------------------------------------

! Definir funciones para QTL1 y QTL2...
! Función gaussiana normalizada a 1 MW: exp{-[(r-cv10)/(a*cf10]^2}
FGAUSS(CV10,CF10,CAR2):; ! QTL1
FGAUSS(CV11,CF11,CAR4):; ! QTL2

!TRUBA(car21,car22):.001:0.08::T ! car21=Pech; car22=Pebw
!VASHRAT(car21,car22):.001:1.10::T

CRAD4=VINT(CAR22B)

Calentamiento por Neutral Beam Injection (NBI)
----------------------------------------

!NBI:0.005:91.108::;
!NBI source from Fasolt
!cv17=neavb
!cv14=teavb
!cv13=tiavb
!NBI_FAFNER(cv17,cv14,cv13,car23,PIBM,SNEBM):0.001:1.110::H;
!PEBM=car23-PIBM
NBI_FAFNER_v0(cv16,CNB1,1,PEBM,PIBM,SNEBM):0.001:91.110::H;
NBI_FAFNER_v0(cv16,CNB2,2,CAR20,CAR25,CAR29):0.001:91.111::I;
  !    NBI_FAFNER_v0(ytaup,V_haz,kbeam,d2,d3,d4)
```

```
  !    PIBM (de Astra): Perfil radial de absorción de los iones
  !    SNEBM (idem) : Perfil radial nacimiento iones rápidos (H(r))
CAR23=PEBM+PIBM+CAR20+CAR25

! Fuente de partículas
----------------------

ZEF=1+(CIMP1-1)*sqrt(sqrt(NE/NEB));   ! Carga efectiva plasma
!ZEF=1.4
AIM1=12.;   ! masa de la impureza principal
ZIM1=6.;    ! carga de la impureza principal
NIZ1=NE*(ZEF-1.)/(ZIM1-1.)/ZIM1;  ! densidad impureza
NI=NE*(ZIM1-ZEF+1.)/ZIM1;  ! densidad iónica
ZMAIN=1;      ! carga de la componente iónica ppal.

! Fuente de partículas mediante código Monte-Carlo
CV16=TAUPB
CV15=NELAC
EIRENE(CV16,CAR14):0.005:91.108::S;
EIRENEPN(CV16,CV15,CAR14):0.005:90.001::F;

! Velocidades de fuga
FTANHF(CF6,CF7,CF8,CF9,0.0,CAR9)::::G;

Transporte NC
-------------

! Campo ambipolar NC: FLUX_EFIELD(modelo,DN,HE,XI)
FLUX_EFIELD(1,CAR19,CAR17,CAR18):0.01:90.::X;
! INIRIPP()::::;
! TNRIPL(CAR15X,CAR16X)::::;
! ERSTEP()::::;
! TRDNEO()::::;

! Difusividad térmica:

CAR26=NUEE*BTOR*rho*rho/TE/2000
CAR27=1+CAR26*CAR26
CAR28=CV2*NUEE*RHO*RHO*( (2./3.*0.34*RHO/AB)  )**(1.5)/CAR27
HE=CHE2/NELAC+CAR17+CAR28+CV4*F1+CV1/NELAC**2*(1-FPR)**4;
XI=CHE3+CAR18

! Difusividad partículas:
!DN=constante + NC + frac_atr^2 * Xe;
DN=CF1/NELAC+CAR19+CF2*CAR28*(2./3.*0.34*RHO/AB)+CV7*F1;

Densidad electrónica
--------------------

NE:E;
NEB=NEXB; NE=NEX;
```

```
SN=CAR14+CNEUT1*(1-FJUMP(90.001))*SNX+SNEBM+CAR29 !+CNEUT2*CAR32

! Velocidad de fuga. Se usa CN porque el flujo se calcula como
! la parte difusiva más un término CN*NE.
! Se eliminan las fugas desde dentro.
! Al llegar a corte de ECRH se anula el término de fuga.
 CN=CAR9+CHE4*GAUSS(CV12,CF12)*(STEP(1.74-FRMAX(NE)))

Temperatura electrónica
-----------------------

TE:E; TEB=TEXB; TE=TEX;
! Fuentes de calentamiento y de pérdidas:
! P_ECRH = Gi*gaussiana, donde Gi (=CRAD1) es la eficiencia
! de la deposición. P_ECRH (=PEX) se carga en CAR2 y CAR4...
PEX=CRAD1*CF5*QECR*CAR2 + CRAD1*(1-CF5)*QECR*CAR4;
! ... o se defines con la función gaussiana GAUSS...
!CAR2=CF5*GAUSS(CV10,CF10) + (1-CF5)*GAUSS(CV11,CF11);
!PEX=QECR*CAR2/VINT(CAR2B);
! ... o se toma de los datos experimentales.
!PEX=QECR*CAR2X;
! La potencia radiada experimental es PRADX...
!PRAD=PRADX;
! ... o puede venir de una fórmula de usuario
PRAD=POCHAN*NE*NIZ1 !+PBRAD
PET=-PEI;
PE=-PRAD+PEI*TI+PEX*(STEP(1.74-FRMAX(NE)))*CRAD1+PEBM+car22+car20

Temperatura iónica
------------------

TI:E; TI=TIX; TIB=TIXB
PIT=-PEI
PI= -PICX+PEI*TE+PIBM+car25 ! Intercambio Coulombiano implícito

Flujo poloidal
--------------
! Se congela usando la conf. de vacío
CU:A; CC=.8*CCSP*(1.-sqrt(2./3.*0.34*RHO/AB));
MV=MVX
CU=CC;

-----------------------------------------------------------------

! Perfiles auxiliares (gradPe, etc.)

CAR6=NE*TE;           ! Pe
CAR16=NEX*TEX;        ! Pe experimental
CAR5=-grad(CAR6);     ! gradPe
CAR15=-grad(CAR16);   ! gradPe experimental
```

```
CAR7 = grad(TE)                          ! gradTe
CAR8 = grad(NE)                          ! gradNe

CAR11 = -TE*(CAR7/TE + CAR8/NE)      ! Er gradPe (kV/m)
CAR11C=0.;
CAR11B=0.;

Nivel medio de fluctuaciones
----------------------------

F1=1.e-3;
! F1=F1X  ! Si se dispone de un perfil inicial
F1:E;
! Condiciones de contorno mediante...
!F1B=0.1;  ! función dada o...
!QF1B=0; ! ...flujo dado explícitamente o...
QFF1B=0.; ! ... flujo dado implícitamente

! Auxiliar omega*=(k_p rho_s) 3e5 sqrt(Te)/L_n
CAR1= CF15*3.e5*sqrt(TE)*abs(CAR8/NE);
CAR1C=0;

! Crecimiento
! gamma=f_a^2 omega* (rho_s/Ln) sqrt(m_e/m_i) =
!      =f_a^2 (k_p rho_s) (T_e/(eB))/L_n^2 sqrt(m_e/m_i)
CAR10=(2./3.*0.34*RHO/AB)*CF15*22*TE;
CAR10=CAR10*CAR8*CAR8/NE/NE/BTOR

! Cizallamiento
! shearing=alpha2*( r/q*grad(q/r*Er) )**2 / gamma
! CAR14=CAR11/RHO/MU;
! CAR12=CF3*( RHO*MU*grad(CAR14) )**2 / CAR10;
! CAR12C=0.;

! Saturación no lineal
! alpha1= (k_perp rho_s)^2/rho_s^2
CAR13=CF14*0.9e5*BTOR*BTOR*CF15**2/TE;
CAR13B=CF14*0.9e5*BTOR*BTOR*CF15**2/TEB;

! modelo (gamma-alfa1*eps-alfa2*shearing)*eps
SFF1=(CAR10-CAR13*F1)/CF4;
DF1=1./CF4;

Turbulencia RB

F4:A;
F4=.001

car24=CV3*car30*CAR13**2/(CAR22+1.)  ! cizallamiento para RB
DF4=CF10/CF4
SFF4=(CAR22-CV2*CAR23*F4-CAR24)/CF4
```

```
QFF4B=0;

momento poloidal
----------------

! CF13x - mu - Amortiguamiento viscoso del fluido
! CF14x - alfa3 - Esfuerzos de Reynolds
! CF15x - Difusión de la turbulencia

! F2:A; F2=-CV6*cos(GP*RHO/ROC); QF2B=CV5;
! CAR7=1/VR*grad(CAR3); ! OJO: dividir por algo con dimensiones de longitud.

! difusión
!DF2=CF15x*F1+CF14*CAR6+CFUS1*gauss(1.,.05);
!DF2=CF15x*F1
!DF2=CF15x/CF4

! forzado
! SF2=-CF13x*CAR21*F2-CF14x*CAR7
! CAR9=1.e6*F3/NI*GP2*BTOR*MU*RHO/RTOR ! E_r = V_tor*Bpol

! tau_LHD

CF16=0.17*(.1*NEAVB)**0.69*BTOR**0.84
CF16=CF16*AB*AB*RTOR**0.75/(QTOTB**0.58)   ! tau_LHD

car3=ER/VR

Perfiles radiales
-----------------------------------------------------------------

Perfiles principales

ne\NE\\NEX\5;
Te\TE\\TEX\-2;
Phi\(vint(car3B)-vint(car3))/1000\1; potencial electrostático, kV
Xi\XI\10;

Sn\SN\\SNX !\-3;
Pech\PEECR\\PEX !\-3;
De\DN\1;
Xe\HE\10;

ni\NI\5;
Ti\TI\\TIX\-2;
Er\ER/1000\10;  ! Campo eléctrico radial, kV/m
XiNC\car18;

SeNB\SNEBM+car29;
PeNB\PEBM+car20;
DeNC\car19;
```

```
XeNC\car17;


Fuentes, sumideros, geometria

Zeff\ZEF\3;
PeiC\PEICL !\-3;
n_CV\NIZ1
Dat+\CV7*F1\0.5;
nada\;
Fs\VINT(SNTOT)/GP/G11; ! No es esto el flujo de las fuentes de partÌculas
Fve\NE*CN\-4;
Xat+\CV4*F1\5;

Prad\PRAD !\-3;
Pei\PEI*TE;
PEGN\PEGN !\-3;
Dat-\CF2*HE*(2./3.*0.34*RHO/AB)\0.5;
Pebw\car22;
ro\RHO;
Gn\GN\-4; ! D_n \nabla n_e
Xat-\CAR28\5;

XeX\HEXP\10;


Dexv\DN\1;
Fe\GN\-4;
<gr>\CAR10X;
Prad\PRAD\\PRADX;


iota\MU-3/2\.2
epst\CAR3X
epsh\CAR4X
Eirn\CAR14

gama\CAR10;
alfa\CAR13;
fluc\F1;
ErIs\CAR14/1000\10;
ErNc\ER/1000\10;
omge\CAR1;
Pe\CAR6\-6;
Fatr\sqrt(2./3.*0.34*RHO/AB);
Ln\NE/(abs(CAR8)+.0001)\1;
vfug\CN;
grPe\CAR5\-7;

ErgP\CAR11/1000\10;       ! Er gradPe (kV/m)
Er\(CAR14+CAR11)/1000\10;      ! Er gradPe (kV/m)+Erˆisland
```

```
gama\CAR10;
PeX\CAR16\-6;
etae\ETAE;
kros\.0033*sqrt(TE)/BTOR;
Kper\sqrt(CAR13);
tgh\CAR9;
grPX\CAR15\-7;

! Geometría

shea\shear ; Vol\volum\\CAR1X ; Slat\slat\\slatX ; iota\MV\\MVX
g11\G11/VR\\G11X; g22\G22/rho/RTOR/RTOR\\G22X; g33\G33\\G33X; dVdr\VR\\VRX

------------------------------------------------------------------------
------------------------------------------------------------------------

Señales temporales
------------------------------------------------------------------------

Te0_TEC_-1;
 Ti0_TIC_-1;
Te25_TE(HRO*12)_-1;
 nlin_NELAC_5;
Te50_TE(HRO*24)_-1;
 Snbi_VINT(SNEBMB)_1;
Te75_TE(HRO*36)_-1;
 <Sn>_VINT(SNTOTB)_500;

Qebw_CRAD4*1000_500;
 Qrad_VINT(PRADB)*1000_500;
Qnbi_VINT(car23B)*1000_500;
 Qei_QEICLB*1000_500;
Qech_VINT(PEXB)*(STEP(1.2-NELAC))*CRAD1*1000_500;
 Qibm_vint(PIBMB)*1000_500;
Qicx_QICXB*1000_500
 Qebm_vint(PEBMB)*1000_500;

Snbi_VINT(SNEBMB);
 Wi_WIB*1000_-4;
<Sn>_VINT(SNTOTB);
 QeTo_QETOTB
Cpuf_ZRD48X; ! Inyección de gas frío, en A
 ZefM_FRMAX(ZEF);
Pn2_VINT(PRADB)*1000/NELAC/NELAC
 <Te>_TEAVB;

<Ti>_TIAVB; nncl_NNCL; nnbm_cf16;
 QNB_qnb;
Wpl_WTOTB*1000;
 CV_VINT(CAR25B)*1000
ne0_NEC;
```

```
 We_WEB*1000_-4;
corte_FRMAX(NE)/1.74_1
 dndt_TIMDER(NELAC)/NELAC/1000_.1  ! en aumento relativo por ms

taue_TAUEB*1000_-3;
taui_TAUEIB*1000_-3; ! taux_WTOTB/(QJOULB)_-3;
taup_TAUPB*1000_-3;
ECRH_QECR
EBWH_QFW
NBI_QNBI
nvol_NEAVB
------------
```

7. *Trabajo en desarrollo*

Dado el ritmo al que se desarrollan estos trabajos, resulta oportuno describir las previsiones a corto o medio plazo. Los programas periféricos de ASTRA se han usado ya para la producción científica basada en datos del TJ-II. Algunos ejemplos son los siguientes cálculos (no se indica el nombre del trabajo, sino lo concerniente a lo calculado con ASTRA usando códigos periféricos):

- Fuentes de reciclado en barridos en densidad de las descargas ECRH [**29**].
- Fuentes de partículas y potencia NBI en los modos "L" y "H" de confinamiento [**30**].
- Evolución teórica de los perfiles de deposición por EBWH en descargas con paso de régimen ECRH a régimen NBI [**31**].
- Contribución neoclásica en los plasmas ECRH [**22**].

Sin embargo, cabe esperar actualizaciones de los códigos principales (EIRENE, TRUBA, FAFNER) que requerirán actualizar los códigos intermediarios y, seguramente, también las subrutinas de ASTRA. A los cálculos del campo eléctrico ambipolar les dedicamos un apartado a continuación, igual que a otro importante trabajo en desarrollo: actualizar la geometría de vacío cuando el plasma tenga suficiente presión o cuando haya corrientes del plasma importantes, lo que se puede lograr considerando VMEC como nuevo código principal para acoplar a ASTRA. Previsiblemente, estos desarrollos darán lugar a futuros informes técnicos.

7.1. *Inclusión de DKES*

Se ha visto que los dos modelos de transporte neoclásico incluidos actualmente en `Flux_Efield.f` son analíticos, pero también hemos advertido en el §5.1 que otras elecciones pueden proceder del cálculo numérico. La idea en este caso es crear una base de datos de transporte neoclásico (ejecutando casos con distintas colisionalidades y valores del campo eléctrico) mediante el uso distribuido [**32**, **33**] del código DKES [**34**]. El plan consiste en elaborar una tabla de valores de los coeficientes de transporte neoclásicos según los valores de colisionalidad y campo eléctrico en distintas posiciones "radiales" del TJ-II. De esta tabla se pueden obtener funciones de ajuste o interpolación que harían las veces de "modelo de transporte neoclásico" para calcular los flujos de electrones e iones. Con esta información, el código periférico de cálculo del campo eléctrico ambipolar (§5) permitiría la inclusión de los flujos provenientes del cálculo numérico con DKES. Obsérvese que mediante este proceso se obtiene el campo eléctrico *autoconsistente* según los flujos neoclásicos dados por DKES.

7.2. Inclusión de VMEC

Partiendo de una configuración magnética de vacío, el código de equilibrio magnetohidrodinámico (MHD) VMEC [**27**] toma los perfiles de presión termodinámica y densidad de corriente del plasma para obtener la configuración perturbada. Se está escribiendo un código [**35**] que, tomando la salida de VMEC, obtiene coordenadas magnéticas –las de VMEC no lo son– y calcula el tensor métrico promediado a las correspondientes superficies de flujo magnético constante. Estos elementos métricos son necesarios en las ecuaciones del transporte y pueden incorporarse inmediatamente al cálculo [**26**].

Conviene observar que no sólo es importante la obtención de la geometría del equilibrio MHD, sino también que se trate de coordenadas magnéticas. El cálculo de algunas magnitudes importantes en la física de *stellarators*, como p. ej. la cizalla magnética local [**36**], se basa en el uso de coordenadas magnéticas. Otras magnitudes relacionadas con la evolución de la transformada rotacional también se calculan mucho más fácilmente en este tipo de coordenadas [**26**, **37**].

Agradecimientos

Como se ha ido indicando, la incorporación de los códigos EIRENE y FAFNER se debe a las numerosas librerías, códigos de acceso a ellas y "manualillos" desarrollados por J. Guasp (Laboratorio Nacional de Fusión, Ciemat). El código de solución a la condición de ambipolaridad fue escrito en `flux_Efield.f` por G. V. Pereverzev (Max Planck Institut für Plasmaphysik, Garching, Alemania), a quien agradecemos sinceramente su continuada colaboración con las instalaciones de ASTRA en las computadoras del Ciemat. El modelo de flujos de la Ref. [**16**] fue inicialmente –versión monoenergética– escrito en `flux_Efield.f` por F. Castejón (Laboratorio Nacional de Fusión, Ciemat).

Referencias

1. G. V. Pereverzev, P. N. Yushmanov, *ASTRA Automated System for TRansport Analysis*, Max Planck Institut für Plasmaphysik, Rep. IPP 5/98, Garching, February 2002.
2. D. Reiter, M. Baelmans, P. Börner, Fusion Sci. Technol. **47**, 172 (2001).
3. A. Teubel, F.P. Penningsfeld, Plasma Phys. Control. Fusion **36**, I43 (1994).
4. M. A. Tereshchenko et. al, Proc. 30th EPS Conference on Contr. Fusion and Plasma Phys., 27A, P-1.18 (2003).
5. http://www.ciemat.es/portal.do?TR=C&IDR=32
6. D. Reiter, *The Eirene Code User Manual*, Version: 11/2005; http://www.eirene.de.
7. http://www.eirene.de/html/recent_reports.html
8. J. Guasp, C. Fuentes, M. Liniers, *Dinámica de impurezas durante la inyección de haces neutros en el TJ-II (simulación)*, Informes Técnicos Ciemat 981, octubre 2001.
9. M. Liniers et al., Proc. 15th International Stellarator Workshop, Madrid, 3–7 October 2005, P3-14
10. J. Guasp, C. Fuentes, M. Liniers, *Cálculos de inyección de haces neutros para las descargas del TJ-II*, Informes Técnicos Ciemat 1050, diciembre 2004.
11. F. L. Tabarés et al., Plasma Phys. Control. Fusion **50**, 124051 (2008)
12. M. A. Tereshchenko et al., Informes Técnicos Ciemat 1134, febrero 2008.
13. http://grid.ct.infn.it/egee_applications/application_details.php?ID=88
14. J. L. Vázquez-Poletti et al., *Vashra-T: Grid Ray Tracing for the Fusion Physics ASTRA Code*, OGF25/EGEE 4th User Forum, Catania (Italy), March 2009.
15. F. Castejón et al., *Grid Computing for Fusion Research*, 3rd Iberian Grid Infrastructure Conference, Valencia (Spain), May 2007. IBERGRID Proceedings, pp. 291-302.
16. C. D. Beidler, *Neoclassical transport properties of HSR*, IPP-Report 2/331, p.194 (1996).
17. H. Maaßberg, C. D. Beidler, E. E. Simmet, Plasma Phys. Control. Fusion **41**, 1135–1153 (1999).
18. L. M. Kovrizhnykh, Plasma Phys. Rep. **32**, 988 (2006).
19. Jerónimo García Olaya, *Study of electron heat transport in LHD and TJ-II*, Tesis doctoral, Universitat Politècnica de Catalunya (España), 2006, ISBN: 84-690-1931-7
20. J. García, J. Díes, F. Castejón, K. Yamazaki, Phys. Plasmas **14**, 102511 (2007)
21. D. Hastings, W. Houlberg, K.-C. Shaing, Nucl. Fusion **25**, 445 (1985).

22. V. I. Vargas et al., *ECRH power dependence of electron heat di?usion in ECRH plasmas of the TJ-II stellarator*, 35th EPS Conf. Plasma Phys., Hersonissos, Crete, Greece, 9–13 june (2008): ECA Vol. 32, P5.018
23. J. J. Martinell, D. López-Bruna, F. Castejón, V. I. Vargas, C. Gutiérrez-Tapia, *Radial Electric Field Computations in TJ-II and Comparison with HIBP Measurements*, 35th EPS Conference on Plasma Physics, Sofia, Bulgaria, 29 june – 3 july (2009), ECA Vol. 33, P4.184
24. V. Tribaldos, Phys. Plasmas **8**, 1229 (2001).
25. L. M. Kovrizhnykh, Plasma Phys. Rep. **31**, 17 (2005).
26. D. López-Bruna, J. A. Romero, F. Castejón, *Geometría del TJ-II en Astra 6.0*, Informes Técnicos Ciemat 1035, agosto 2006.
27. S. P. Hirschmann et al, Phys. Fluids **26**, 3553 (1983)
28. D. López-Bruna, F. Castejón, J. M. Fontdecaba, *Transporte con ASTRA en TJ-II*, Informes Técnicos Ciemat 1035, enero 2004.
29. V. I. Vargas et al., *Density dependence of particle transport in ECRH plasmas of the TJ-I I stellarator*, Informe Técnico Ciemat 1162, febrero 2009.
30. T. Estrada et al., Plasma Phys. Control. Fusion **51**, 124015 (2009).
31. A. Cappa et al., *Dynamic simulation of the Electron Bernstein Wave heating under NBI conditions in TJ–II plasmas*, aceptado en Contr. Plasma Phys. (2010).
32. A. J. Rubio-Montero et al., *Executions of the DKES code on the EELA-2 e-Infrastructure*, Proc. 2nd EELA-2 Conf., Choroní (Venezuela) 2009, pp. 27–32.
33. A. J. Rubio-Montero et al., *Drift Kinetic Equation solver for Grid (DKEsG)*, Proc. 1st EELA-2 Conf., Bogotá (Colombia) 2009.
34. V. I. van Rij, S. P. Hirschmann, Phys. Fluids B **1**, 563 (1988).
35. J. M. Reynolds, D. López-Bruna, *Incorporación de VMEC al cálculo dinámico del transporte con Astra*, próximo Informe Técnico Ciemat.
36. J. M. Green, M. S. Chance, Nucl. Fusion **21** 453 (1981).
37. P. I. Strand, W. A. Houlberg, Phys. Plasmas **8** (6), 2782 (2001).

D. López Bruna, Á. Cappa
Laboratorio Nacional de Fusión (Ciemat)
Asociación Euratom-Ciemat
Avda. Complutense 22
28040 Madrid

J. M. Reynolds
Instituto de Biocomputación y Física de Sistemas Complejos
Mariano Esquillor, Edificio I+D
50018 Zaragoza

J. Martinell
Instituto de Ciencias Nucleares
Universidad Nacional Autónoma de México
A.P. 70-543, México D.F., México

J. García
Association Euratom-CEA
Cadarache, 13108 St. Paul-les-Durance
Francia

C. Gutiérrez-Tapia
Instituto Nacional de Investigaciones Nucleares
Cra. México-Toluca s/n
A.P. 18-1027, México D.F., México