General Atomics

DIII–D Physics Memo

Subject: Improvements to TORAY-GA Code **D3DPM No. 0201**

From: R. Prater, R. Harvey, J.E. Kinsey **Date: April 5, 2002**

Some changes to the TORAY-GA code have been made which should be documented.

1. New Ray Pattern Generator

First, a new ray pattern generator has been implemented in version 1.3 of TORAY-GA. One objective for a new pattern generator is to make the rays consistent with a Gaussian distribution, which the older version was not. A second objective is to make the ray pattern more flexible, since the old pattern generator was limited to 1, 6, 18, or 30 rays. In some cases, particularly for ONETWO run with higher radial resolution than 51 grid points, a larger number of rays is needed. For 51 radial grid points, the 30 ray option is satisfactory.

The ray pattern subroutine RAYPAT mimics the propagation of a Gaussian beam in the far field region by propagating an array of straight rays emanating from a single point with divergence appropriate to the Gaussian beam. All rays have the same power, and the density of rays is consistent with the power density of an expanding beam with Gaussian fall-off with radius. This approach is inaccurate in the near field region, where the beam may be focusing, but for the DIII-D installation the focus is outside the plasma so the far field approximation is not bad. Note that the effective single point source lies at the convergence of the far field representation of the rays, not at the focus of the beam or at the waveguide mouth.

The fractional power outside a radius r for a Gaussian distribution is given by

$$f\,(r) = exp\ \{-2\ [r/w(z)]^2\}$$

where the waist radius $w(z)$ as a function of distance z is a linear proportion in the far field,

$$w(z) = D\,z \quad ,$$

where the divergence D is the angular dispersion (half angle for e^{-2} in power), or 2.89 deg. for the DIII-D 110 GHz launchers with inside diameter of 2.375 in. (Note that the inone namelist element hlwec is the half angle at half power, related to D by hlwec = $D*[ln(2)/2]^{1/2}$. For the far field region, D is given by $\lambda/\pi w_1$ = 3.03 deg, where λ is the free space wavelength and w_1 is 16.4 mm for the DIII-D geometry, per John Doane [GA-C21242]. For the DIII-D geometry, we use a slightly smaller value of D, 2.89 deg corresponding to hlwec = 1.7 deg, to better simulate the wave optics in the intermediate distance regime of interest.)

Then for an angle a, the power lying outside that angle is

$$f(a) = exp\,[-2\,(a/D)^2] \quad .$$

The Gaussian distribution can be broken into annular zones, with the boundaries chosen at radii for which the integrated power of each annular zone is proportional to the number of rays in that zone divided by the total number of rays. One central ray is always placed at the center of the distribution, occupying a circular zone. The rays of an annular zone are uniformly spread at the power-weighted center radius for that zone, using an arbitrary number for the orientation of the ray pattern around the central ray. The goal is to find the smallest number of rays for which the power deposition profile is independent of the number of rays, the number of zones, and the orientation of the rays around the central ray.

The Gaussian representation is now the default in the public version of TORAY-GA, identified as version 1.3 or higher in the log file when the code is run. If no toray.in file is found, TORAY-GA will use defaults. Under this condition the value of nray in the inone file may be equal to 1 to get 1 ray, and any other value of nray will give 30 rays with the Gaussian pattern. (Previously, nray could be set to 1, 6, 18, or 30, corresponding to 1, 2, 3, or 4 annular non-Gaussian zones.) Studies have shown that the 6 or 18 ray options are too unreliable to be generally useful so these options have been discontinued, although they can be restored if a toray.in file is used.

With a toray.in file, more control is available. Some new elements of the EDATA namelist are the following, with the default values shown:

```
raypatt = ‘.gauss.’          ! only ‘.km.’ gives old pattern
gauszone = 4                 ! number of annuli in Gaussian pattern
mray = 1 5 12 12 20 24 26    ! array of number of rays in each annulus
cr = 0.0  0.1  0.050, -0.050 ! values for orientation of rays each annulus, in radians
     0.050  –0.025  0.025
cqldat = .false.                     ! set to .true. for running cql3d
```

```
netcdfdat = .false.          ! set to .true. for ncdf file output toray.nc
ar = 0.1                     ! obsolete, no longer used
br = 0.05                    ! obsolete, no longer used
```

Note that if raypatt = ‘.km.’ then nray determines the number of rays in the pattern, as before.

This change was introduced in version 1.3.

2. Netcdf File Output

Setting netcdfdat=.true. in the toray.in file produces a file toray.nc when TORAY is run, This file contains the data generated by toray in the run. The contents of the file can be viewed using the Unix command “ncdump toray.nc”, a command which is most useful when directed to a file. From IDL, the file contents can be restored using a command “s=ncdf_restore(‘toray.nc’)”. The variables available in this file are given by the following list.

```
dimensions:
        neltmax = 53 ;          Largest number of ray elements for any ray
        nrays = 30 ;            Number of rays
        two = 2 ;
        ledge = 51 ;            Number of radial points in ONETWO grid
        ledgem1 = 50 ;
variables:
        int nray ;
                nray:long_name = "Number of rays" ;
        int nharm ;
                nharm:long_name = "First harmonic number" ;
        double freqcy ;
                freqcy:long_name = "Wave frequency" ;
                freqcy:units = "Hz" ;
        double x0(nrays) ;
                x0:long_name = "Ray launch position, for each ray" ;
                x0:units = "cm" ;
        double y0(nrays) ;
                y0:long_name = "Ray launch position, for each ray" ;
                y0:units = "cm" ;
        double z0(nrays) ;
                z0:long_name = "Ray launch position, for each ray" ;
                z0:units = "cm" ;
```

```
double angrid1(nrays) ;
        angrid1:long_name = "Ray launch angle, k-vector wrt z-dirn" ;
        angrid1:units = "deg" ;
double angrid2(nrays) ;
        angrid2:long_name = "Ray launch angle, k-vector wrt R-dirn" ;
        angrid2:units = "deg" ;
int nrayelt(nrays) ;
        nrayelt:long_name = "Number of ray elements for each ray" ;
double ws(nrays, neltmax) ;
        ws:long_name = "poloidal distance along a ray" ;
        ws:units = "cm" ;
double spsi(nrays, neltmax) ;
        spsi:long_name = "radial-like variable (sqrt(norm pol flux))" ;
double wr(nrays, neltmax) ;
        wr:long_name = "major radius" ;
        wr:units = "cm" ;
double wphi(nrays, neltmax) ;
        wphi:long_name = "toroidal angle" ;
        wphi:units = "rad" ;
double wz(nrays, neltmax) ;
        wz:long_name = "vertical height" ;
        wz:units = "cm" ;
double wnpar(nrays, neltmax) ;
        wnpar:long_name = "parallel refractive index" ;
double wnper(nrays, neltmax) ;
        wnper:long_name = "perpendicular refractive index" ;
double delpwr(nrays, neltmax) ;
        delpwr:long_name = "power in ray channel" ;
        delpwr:units = "erg/sec" ;
double sdpwri(nrays, neltmax) ;
        sdpwri:long_name = "power to ions (if incl)" ;
        sdpwri:units = "erg/sec" ;
double cwexde(two, nrays, neltmax) ;
        cwexde:long_name = "Complex Ex/E Polarization" ;
double cweyde(two, nrays, neltmax) ;
        cweyde:long_name = "Complex Ey/E Polarization" ;
double cwezde(two, nrays, neltmax) ;
        cwezde:long_name = "Complex Ez/E Polarization" ;
double fluxn(nrays, neltmax) ;
        fluxn:long_name = "fluxn, Stix norm, |E|=1" ;
        fluxn:units = "erg/sec/cm^2" ;
double sedenfac(nrays, neltmax) ;
        sedenfac:long_name = "Energy density factor rel to free space" ;
double svgrpdc(nrays, neltmax) ;
        svgrpdc:long_name = "Group vel rel to clight" ;
double sbtot(nrays, neltmax) ;
        sbtot:long_name = "Magnetic field strength" ;
        sbtot:units = "gauss" ;
double saspcti(nrays, neltmax) ;
        saspcti:long_name = "Inverse aspect ratio (Bmax-Bmin)/(Bmax+Bmin)" ;
double sene(nrays, neltmax) ;
        sene:long_name = "Density along ray" ;
        sene:units = "particles/cm^3" ;
double ste(nrays, neltmax) ;
        ste:long_name = "Temperature along ray" ;
        ste:units = "keV" ;
double salphac(nrays, neltmax) ;
        salphac:long_name = "Collisional damping wavenumber" ;
        salphac:units = "1/cm" ;
double salphal(nrays, neltmax) ;
        salphal:long_name = "Linear damping wavenumber" ;
        salphal:units = "1/cm" ;
int ledge ;
        ledge:long_name = "Number of radial bin boundaries" ;
double psimag ;
        psimag:long_name = "Psi at magnetic axis" ;
        psimag:units = "Volt-sec" ;
double psilim ;
        psilim:long_name = "Psi at limiting flux surface" ;
        psilim:units = "Volt-sec" ;
double Raxis ;
        Raxis:long_name = "Magnetic axis R-coord" ;
        Raxis:units = "cm" ;
double Zaxis ;
        Zaxis:long_name = "Magnetic axis Z-coord" ;
        Zaxis:units = "cm" ;
double BTaxis ;
        BTaxis:long_name = "Magnetic field at mag axis" ;
        BTaxis:units = "tesla" ;
double volume ;
```

```
        volume:long_name = "Plasma volume" ;
        volume:units = "cm^3" ;
    double xbouni(ledge) ;
        xbouni:long_name = "normalized poloidal flux of bin boundaries" ;
    double bin_vol(ledgem1) ;
        bin_vol:long_name = "normalized bin volumes" ;
    double xmrho(ledgem1) ;
        xmrho:long_name = "normalized rho (sqrt(tor flux) of bin center" ;
    double xmene(ledgem1) ;
        xmene:long_name = "density" ;
        xmene:units = "/cm^3" ;
    double xmete(ledgem1) ;
        xmete:long_name = "Electron temperature" ;
        xmete:units = "keV" ;
    double xmzeff(ledgem1) ;
        xmzeff:long_name = "Z-effective" ;
        xmzeff:units = "charge units" ;
    double weecrh(ledgem1) ;
        weecrh:long_name = "RF power dens in each bin, per incident power" ;
        weecrh:units = "Watts/cm**3 /Incident Watt" ;
    double tpowde(ledgem1) ;
        tpowde:long_name = "Integrated power, starting at center" ;
        tpowde:units = "Watts/(Incident Watt)" ;
    double wiecrt(ledgem1) ;
        wiecrt:long_name = "RFCD per norm bin volume, per incident watt" ;
        wiecrt:units = "A/(norm bin volume*Incident Watt)" ;
    double tidept(ledgem1) ;
        tidept:long_name = "Integrated current, starting at center" ;
        tidept:units = "A/(Incident Watt)" ;
```

To determine the profile of driven current from TORAY-GA, the ONETWO transport code takes the variable array wiecrt, which represents the ECCD in each radial volume bin and has dimension A/W/cm^3 and multiplies by a constant $2\pi R/V$, where R is the major radius of the magnetic axis and V is the plasma volume. Then the result is smoothed, linearly interpolated from the volume-centered grid of TORAY-GA to the boundary-centered grid of ONETWO, integrated to find the total driven current, and then scaled to preserve the total driven current calculated directly by TORAY-GA multiplied by the incident ECH power. The linear interpolation introduces significant errors when the change in the current density is large between adjacent radial grid points. This is frequently the case for ECCD experiments in DIII-D. In this case, higher radial resolution than 50 points is required.

The netcdf file output option was introduced in version 1.3.

3. Improved Integration of Power

The TORAY-GA code calculates the power in a ray as it propagates in the plasma using the expression

$$dP = 2\, k_{\mathrm{i}}\, ds\, cos(\phi)\, P$$

where *dP* is the decrement in the power *P* of a ray in taking a step of size *ds* along its trajectory, k_{i} is the imaginary part of the wavevector, and ϕ is the angle between the wavevector and the magnetic field. The algorithm in TORAY-GA uses the value of k_{i} determined at the end of a step to use in this expression.

In the usual case of strong absorption, which is always the case of interest for current drive applications, the wave is fully attenuated before reaching the maximum of k_{i}, which increases approximately like a Gaussian. This means that the power decrement is always overestimated by this expression since the second derivative is always positive. This may make only a small error in the profile of power deposition, one which is probably unmeasureable in most cases. However, the calculation of current drive is very sensitive to the power profile, since the magnitude of the Doppler shift determines where in velocity space the interaction is taking place. Overestimating the power decrement implies that the interaction between the wave and the electrons is taking place at larger $v_{||}$, hence the overestimation in the integrated driven current can be much larger than the error in the power decrement.

A simple improvement to this problem is to use the average k_{i} rather than the value at the end of the step. A comparision of these approaches is shown in Figs. 1 and 2. Figure 1

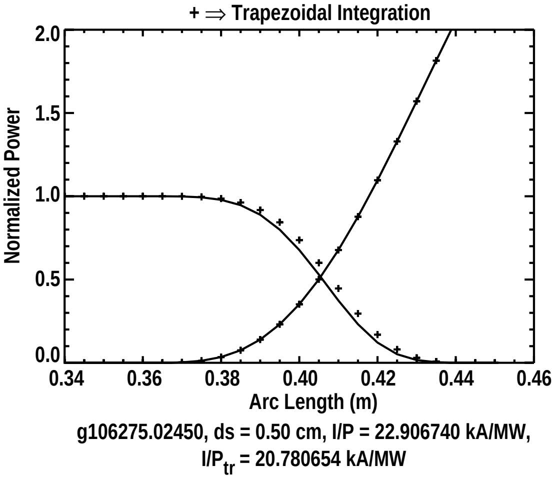

Fig. 1. Normalized power along the arc length of a ray. The equilibrium is 106275.02450, with central temperature of 3.83 keV, central density of $3.99\mathrm{x}10^{19}$ m^{-3} (half the density of the experiment to avoid strong refraction effects), and launch angle for the single ray of polar = 115.869 deg and azimuthal = 196.745 deg. The step size *ds* = 5 mm. The result from TORAY-GA is the solid line, and the + points are for trapezoidal integration.

shows the normalized power in a ray calculated by TORAY-GA compared with that for trapezoidal integration, for a step size of 5 mm. The overestimation of the power decrement can be clearly seen. By comparison, the case for a step size of 1 mm shown in Fig. 2 shows much smaller difference. The current calculated the two ways is shown in Fig. 3. Both cases approach the same value asymptotically as step size is reduced. Assuming that the asymptot is the correct value, the error is shown in Fig. 4. Using a similar process but using 30 rays, as is the usual case for current drive calculations, the calculated current and error are shown in Figs. 5 and 6. This result suggests that step sizes around 1 to 2 mm are needed for 5% accuracy, and that the error can be cut by 40% by using the trapezoidal integration.

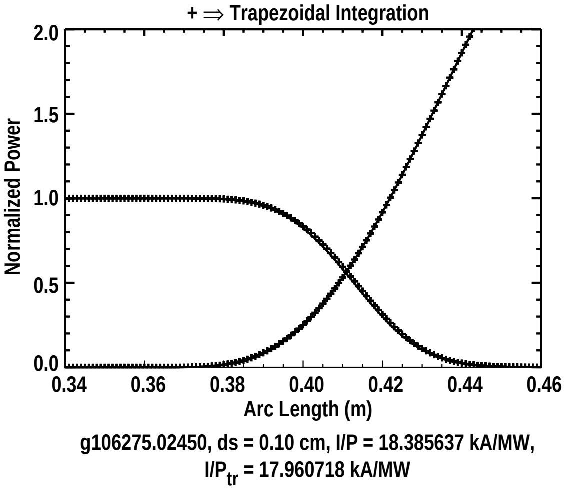

Fig. 2. Same as Fig. 1 but *ds* = 1 mm.

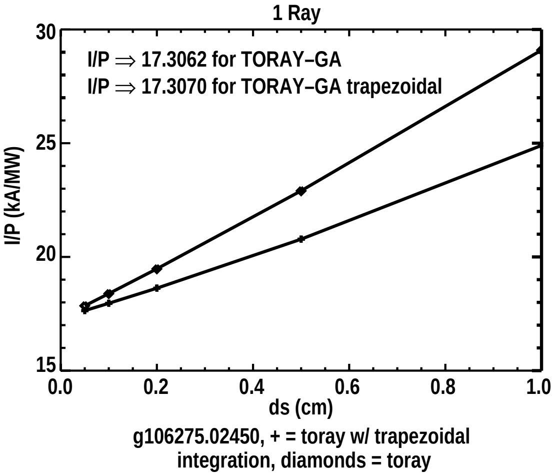

Fig. 3. Integrated current for 1 ray cases, as function of *ds*.

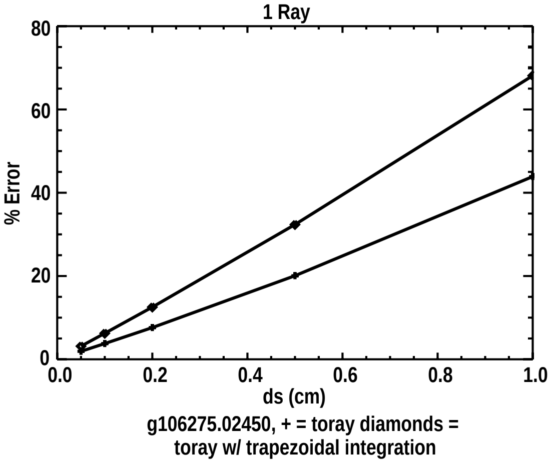

Fig. 4. Error (%) in the calculated current due to nonzero step size.

A study showed that the error introduced by the single-ended integration is larger for more oblique launch and smaller for more normal launch. Also, the larger the plasma β_e, the larger the over-estimation of current. Alternate integration schemes like a three-point integration or assuming a local Gaussian form of the integrand didn’t produce a significant improvement in the accuracy.

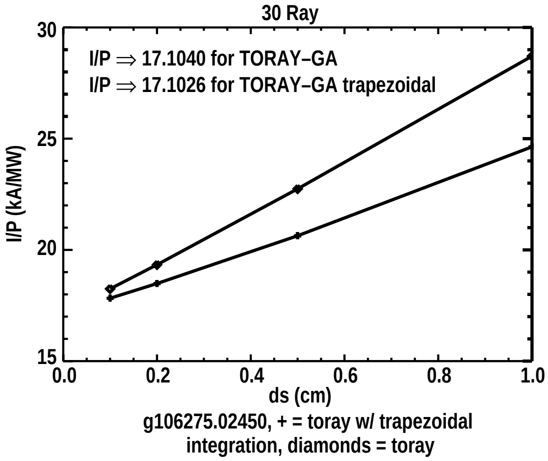

Fig. 5. Same as Fig. 3, but for 30 rays.

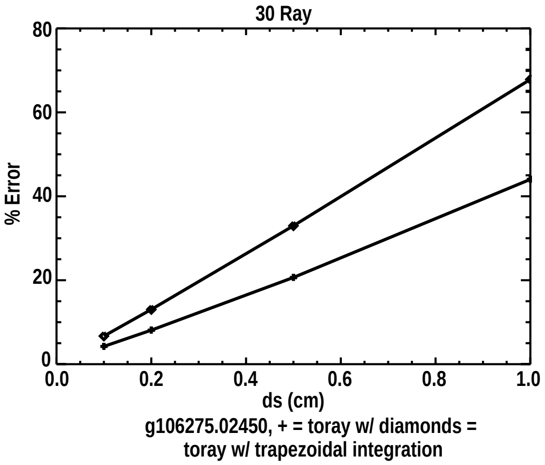

Fig. 6. Same as Fig. 4, but for 30 rays.

4. Improved Normalization of *j/P*

In DIII–D Physics Memo 0102, Y.R. Lin-Liu showed that proper flux surface averaging of the output j/P from the subroutine which calculates the ECCD requires division by $\langle B \rangle$ where TORAY-GA divides by B_{min}. Y.R. Lin-Liu suggests that this can be approximately corrected by dividing j/P by $1+\varepsilon$, where ε is the inverse magnetic aspect ratio $(B_{\mathrm{max}} - B_{\mathrm{min}})/(B_{\mathrm{max}} + B_{\mathrm{min}})$, since $\langle B \rangle$ is not at present available to TORAY-GA. This effect reduces the calculated current.

5. Improved Evaluation of the Poloidal Angle

In the subroutine currn, the quantity θ_p is calculated for input to the routine which calculates the ECCD. θ_p is the “poloidal angle” where ECCD takes place, such that in the circular flux surface and low beta approximation $R_{loc} = R_0 + a \cos \theta_p = R_0 (1 + \varepsilon \cos \theta_p)$. Then

$$\cos \theta_p = \frac{2R_{loc} - (R_{max} + R_{min})}{(R_{max} - R_{min})} \quad .$$

Using B proportional to $1/R$,

$$\cos \theta_p = \frac{(B_{max} + B_{min} - 2B_{max}B_{min}/B_{loc})}{(B_{min} - B_{max})} \quad .$$

However, currn uses

$$\cos \theta_p = \frac{(B_{min} + B_{max} - 2B_{loc})}{(B_{max} - B_{min})} \quad ,$$

which is not correct except in the large aspect ratio limit. Correcting this error introduces a change in θ_p of up to ~10 deg, with the maximum correction at 90 deg and zero correction at $\theta_p = 0$ deg, 180 deg. This correction increases the calculated ECCD.

Improvements 3, 4, and 5 were introduced in version 1.4.