13-260609

To: Distribution Date: 26 June 2009

From: R. Woolley Subject: NSTX CSU Poloidal Fields

Reference 1: R. Woolley, Excel file NSTX_CSU_AssumedPFSystem_090617.xls, issued with this memo 26June 2009
Reference 2: R. Woolley, Excel file NSTX_CSU_Poloidal_Field_Flux-5cm-plots.xls, issued with this memo 26 June 2009
Reference 3: J. Menard, Excel file NSTX_CSU_PF1ABC_geometry_Menard_20090616.xls, issued by email 17June2009
Reference 4: J. Menard, Excel file NSTX_CSU_PFcurrents_OHswing_versionE_20090616.xls, issued by email 17June2009
Reference 5 C. Neumeyer, Excel file NSTX_CS_Upgrade_090217.xls, issued 17 February 2009
Reference 6: J. Menard, Excel file NSTX_CSUv3_IPF_20081023.xls, issued 23October2008

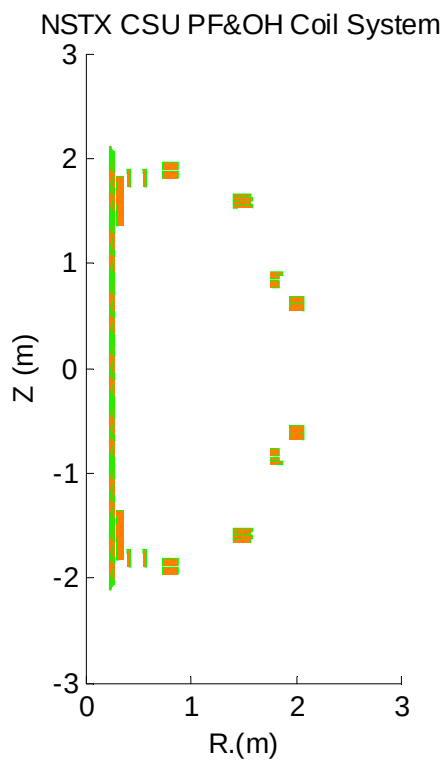
Summary

This memo documents an analysis of poloidal magnetic fields and fluxes in the National Spherical Torus eXperiment Center Stack Upgrade (NSTX CSU). It states the numerical calculation methods, describes the assumed PF and OH coil set (Ref.1), and details approximations made in representing plasma current. Analysis results have been used to create a useful tool, an Excel spreadsheet (Ref.2) which dynamically calculates poloidal magnetic field components and poloidal magnetic flux on a grid of NSTX CSU locations, given operator-input adjustable values of currents in each of the PF coil circuits, in the OH coil, and in the plasma. In addition to stating the computed numerical poloidal magnetic field and flux results for any set of currents the spreadsheet also summarizes the overall spatial pattern of those results via dynamically updated contour plots of magnetic flux, radial magnetic field and vertical magnetic field.

PF and OH Coil Representations

PF and OH coil representations used in the present analysis are detailed in the single table of Ref.1 and are plotted here on this page. The table describes 32 concentric winding packs, each with a winding cross-section configured as a rectangular matrix of rectangular conductor turns. For each winding pack matrix the table states the vertical, radial and total numbers of turns, the winding pack cross-section's width and height, and the (R,Z) coordinates of the winding pack's current center. Turn-to-turn transitions are not described so in this representation the turns are considered axisymmetric. As listed in the table, the 32 windings are configured in 13 series-connected electrical circuits.

The geometric coil representations used were obtained from various different sources. The coil geometry for the PF1ABC coils was extracted from Ref.3. The coil geometry for PF2 and PF5 coils was taken from Ref. 5. Because of its higher resolution details the Ref.6 description of PF3 and PF4 coils was chosen. The Ref.6 approach was also adopted to describe the OH coil as four separate single-layer windings, each with a different number of turns, since that provides more precise poloidal field calculations at TF joints located close to the OH coil ends than would a single rectangular winding model. However, the Ref. 6 OH turn numbers were modified to reflect the latest (17February2009) design point information on the OH coil.



Numerical Methods

The numerical calculation of poloidal field magnetic quantities could have been done in various different ways. The Greens function approach taken here was chosen to minimize human effort, since MATLAB magnetic routines I had developed in the past were available. The latest available MATLAB version, 7.7.0.471 (R2008b) was therefore used for the present analysis.

The Greens function methods employed use exact formulae for the poloidal magnetic field and flux of an idealized circular filament of current, as given, e.g., in formulae from Smythe's textbook on Electromagnetism, employing the Complete Elliptic Integral elementary functions, K and E. MATLAB has its own numerical m-file function subroutines evaluating K and E that are accurate to the full precision of MATLAB's floating point word length, i.e., they are essentially as exact as MATLAB's implementations of other elementary function such as sine and cosine. Since no nonlinear ferromagnetic materials are involved, the magnetic field and flux components of any finite array of coaxial circular current filaments are then calculated exactly as a superposition sum of the contributions from each circular filament.

To use this approach for actual PF coils one must first represent each PF coil by a finite set of coaxial circular filaments. The difference between actual and ideal configurations is that in an actual coil the current is distributed smoothly over a finite conductor volume while the volume of an idealized current filament is zero. Modeling an actual PF coil winding as a single current filament scaled by the number of turns results in quite accurate calculated magnetic field values far from the coil, but the error increases at locations closer to the coil. Indeed, an idealized circular filament's field strength theoretically approaches an infinite limit at a sequence of locations approaching the filament's location, whereas an actual PF coil's field remains finite everywhere, even inside the conductor.

To reduce the error in magnetic field or flux numerically calculated at points close to the powered winding pack, each winding pack is represented by a finite rectangular matrix array of current filaments instead of a single filament. This is done because a filament matrix more closely represents the smooth spatial distribution of the actual coil current. Thus the calculated field using several filaments, exact for that modeled filamentary current distribution, is a better approximation to the actual coil's field than the calculated field of one filament. The calculation method used to locate filaments first tiles the winding pack cross-section with a rectangular matrix of congruent sub rectangles, then places a single filament at the ***center*** of each sub rectangle. In subsequent magnetic calculations these circular filaments are assigned equal currents, thus approximating a constant current density over the winding pack's cross section.

For magnetic field and flux evaluation points located inside the body of a winding pack, an additional modification to the field and flux calculation algorithm is used. First, field and flux is evaluated on the entire finite grid of (R,Z) points located at the ***corners*** of the tiling sub rectangles. Then, this 2D matrix of computed magnetic values is ***interpolated*** via a 2D bilinear algorithm to calculate the resulting field and flux at all specified internal evaluation points. Obviously this additional step avoids ever generating infinite results, and the fact that each calculation location internal to the winding pack is symmetrically surrounded by a grid of filamentary currents makes it plausible that errors due to filamentary current distribution modeling may cancel out. Some ad hoc comparisons with results from other methods also suggest this method is highly accurate. It should be understood that no comprehensive error analysis on this additional step has been performed, however, this additional step is only used for locations inside a winding pack body when evaluating field or flux resulting from powering that same winding pack.

The primary MATLAB m-file subroutine used for field and flux calculation is the following:

```
function [br,bz,flux]=poloidal_fieldy(rho,zeta,r,z);
% Axisymmetric poloidal magnetic field and flux are calculated
% at locations specified in matrices rho and zeta, normalized to a
% total source current of one ampere uniformly distributed between
% circular loop filament locations specified by matrices r and z.
% Matrix sizes must match between r and z which may be 1D or 2D arrays.
% Sizes of rho and zeta must also match each other but they may be ND
% arrays. SI units are used.
%
if size(r)~=size(z)
    error('r and z filament description matrices must be of same size')
end
if size(rho)~=size(zeta)
    error('rho and zeta field calc location description arrays must be of same size')
end
br=zeros(size(rho));bz=br;flux=br;aphi=flux;
%The following uses linear indexing for rho,zeta,br,bz,aphi,flux
%Note that aphi is the vector potential and flux is integrated over two pi radians.
for i=1:numel(br)
        rhot=rho(i);zetat=zeta(i);
% First confirm this evaluation location misses all filaments.
        disc=(r-rhot).^2+(z-zetat).^2;
        if (isempty( find( disc==0 ) ) );
        m=4*rhot*r./((rhot+r).^2+(zetat-z).^2);
        [K,E]=ellipke(m);
        bz(i)= mean(mean((2e-7).*(K+((r.^2-rhot^2-(zetat-z).^2)./disc).*E)...
        ./sqrt((rhot+r).^2.+(zetat-z).^2)  )');
        if rhot==0
            br(i)=0.;
            aphi(i)=0.;
        else
            br(i)=mean(mean(2e-7*(z-zetat)./sqrt((rhot+r).^2+(zetat-z).^2).*(-K+...
            (r.^2+rhot^2+(zetat-z).^2)./disc.*E)/rhot)');
            aphi(i)=mean(mean(4e-7*sqrt(r/rhot).*((1.-m/2).*K-E)./sqrt(m))');
        end
        flux(i)=2*pi*rhot*aphi(i);
        else
            bz(i)=NaN;
            br(i)=NaN;
            aphi(i)=NaN;
            flux(i)=NaN;
        end
end
```

This m-file calculates the magnetic field components, br and bz, and flux, the total magnetic flux included inside the complete 2π radian circle(s) about the z-axis passing through location(s) specified by (rho, zeta). Providing they have the same dimension size as each other, rho and zeta may be specified either as single scalars, as 1-D vectors, as 2-D matrices, or even as ND-arrays having 3 or even more dimensions. The magnetic results variables, br, bz, and flux, are returned by the m-file having the same dimensions size as the field evaluation location specifiers, rho and zeta.

The calculated field is modeled as though it is produced by current shared between a set of coaxial circular loops passing through poloidal half-plane location(s) (r,z), where r and z have the same dimensions size as each other and also may be dimensioned as either single scalars, as 1-D vectors, as 2-D matrices, or even as ND-arrays with 3 or more dimensions. Obviously, r and rho may have completely different size dimensions. The model assumes one ampere of current is equally *divided* between the different loops specified in (r, z), so that the total modeled current flowing is 1 ampere-turn.

SI units are used, i.e., rho, zeta, r, and z must all be expressed in meters, the calculated br and bz are in tesla and flux is in webers.

The set of filaments representing a coil is generated automatically by the following MATLAB m-file subroutine:

```
function [r,z]=filamentize(rmin,rmax,zmin,zmax,m,n)
% Calculates filament matrix to represent rectangular xsection pf coils.
% Rectangle is divided into subrectangles; filaments are located at
% subrectangle centers.
%
r=zeros(m,n);z=r;
for j=1:n;
    r(:,j)=rmin+(rmax-rmin)/n*(j-0.5);
end
for i=1:m;
    z(i,:)=zmin+(zmax-zmin)/m*(i-0.5);
end
```

The first four arguments specify the rectangular cross section dimensions of a PF coil. The other two arguments state the (integer) number if sub rectangle "tiles" to partition it into, with m being the number in the z direction and n the number in the r direction. The rho and zeta quantities it returns are each m-by-n matrices containing the radial and axial coordinates of centers of sub rectangles subdividing the PF coil cross-section.

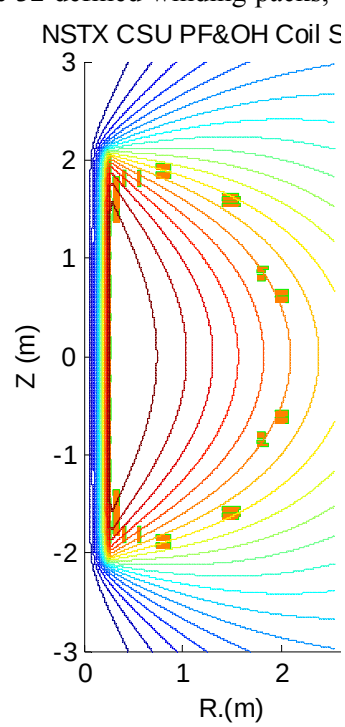
To locate the corners of the tiling subrectangles the folloing m-file was written:

```
function [rho,zeta]=filamentize_corners(rmin,rmax,zmin,zmax,m,n)
% Calculates matrices of field evaluation locations in rectangular xsection pf coils.
% Rectangle is divided into subrectangles; eval. locations are at their
% corners.
%
rpoints=linspace(rmin,rmax,n+1);
zpoints=linspace(zmin,zmax,m+1);
[rho,zeta]=meshgrid(rpoints,zpoints);
```

The MATLAB calculations first evaluated, for one ampere-turn in each of the 32 defined winding packs, the br, bz, and flux values at 601 X 301=180,901 locations representing 1 cm square grid corners covering the following NSTX CSU region:

$$-3\text{ m} \leq Z \leq 3\text{ m} \quad \text{and} \quad 0\text{ m} \leq R \leq 3\text{ m}$$

Weighted sums of these arrays were then taken to determine the spatial patterns of field components and flux resulting from 1 ampere flowing in each of the 13 coil circuits, using weighting factors reflecting the different numbers of turns in each winding pack. Results were saved to disk in a MATLAB file under the data names, BRcoil, BZcoil, and FLUXcoil, each of which is a ND array of numbers dimensioned (601,301,14). However, at this point the 14th "page" of each of these arrays, intended to represent approximate spatial magnetic information for the plasma, still did not contain any meaningful data.

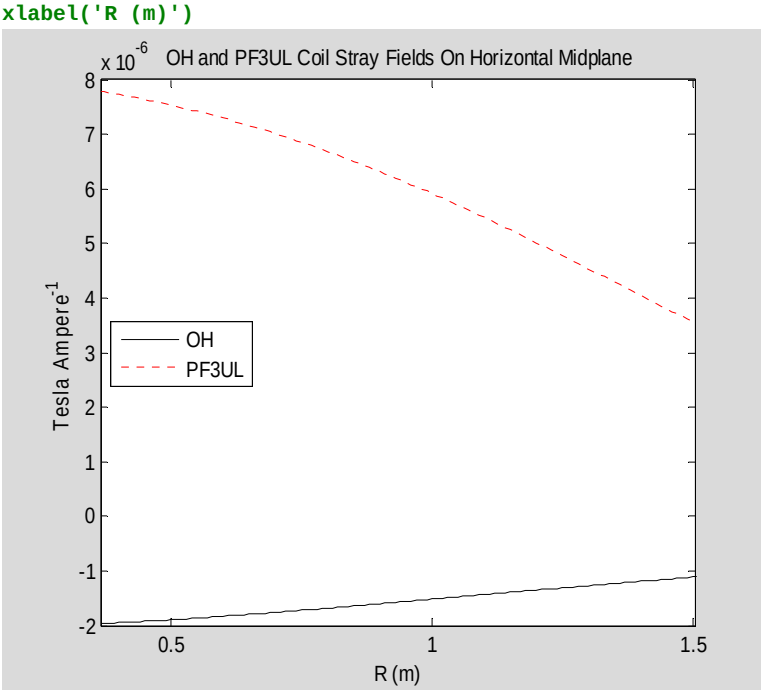


OH Stray Field Compensation For Plasma Startup

The adjacent figure superimposes a spatial contour plot of OH coil magnetic flux onto the coil system diagram. The fact that much of the OH flux returns through the region where the plasma must start up indicates that the stray field must be significant there. Plasma startup

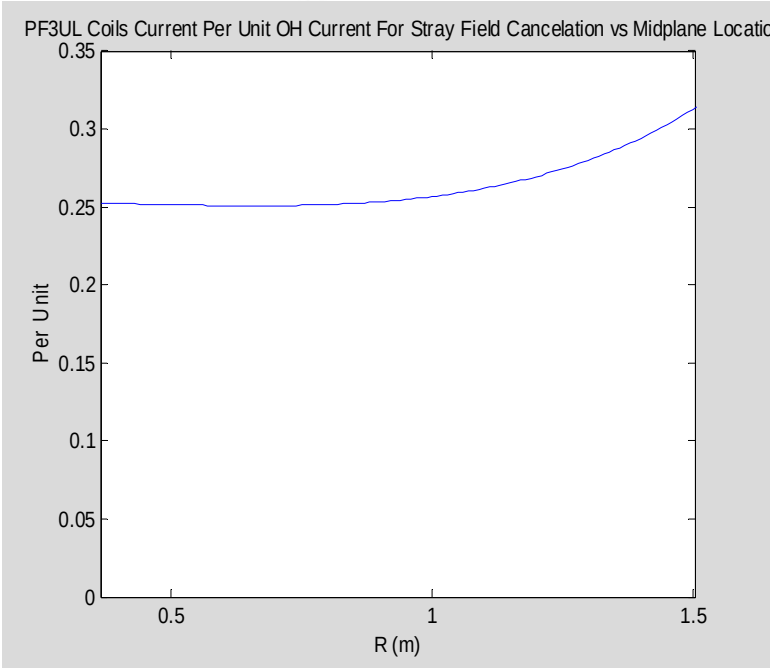
cannot occur with large stray fields, so active compensation is essential for plasma start-up. It may therefore be useful to analyze how to compensate the OH in order to have magnetic flux bypass the plasma region where startup must occur. According to J. Menard the present NSTX practice is to drive PF3 coils to carry currents proportional to the OH coil current for this job. Since these are symmetrical about the horizontal midplane, balanced PF3 currents and any OH current together contribute no radial field on the horizontal midplane, only vertical field there. Therefore it is instructive to plot the midplane vertical fields of PF3UL and OH, versus R, as commanded by the following MATLAB computational cell:

```
plot(RHO(301,:),BZcoil(301,:,1),'k',RHO(301,:),BZcoil(301,:,6)+BZcoil(301,:,9),':r');
xlim([0.365 1.504]);
legend('OH','PF3UL','Location','West');
title('OH and PF3UL Coil Stray Fields On Horizontal Midplane');
ylabel('Tesla Ampere^-^1')
xlabel('R (m)')
```



The negative ratio of the two plotted stray vertical field per ampere is the ratio of PF3 to OH currents needed to cancel midplane stray fields, versus the plasma startup radial locastion. That is plotted below:

```
plot(RHO(301,:),-BZcoil(301,:,1)./(BZcoil(301,:,6)+BZcoil(301,:,9)))
xlim([0.365 1.504]);
title('PF3UL Coils Current Per Unit OH Current For Stray Field Cancelation vs Midplane
Location');
ylim([0 0.35])
ylabel('Per Unit')
xlabel('R (m)')
```



At the official center of the NSTX CSU plasma for which R0=0.934 m, the stray field compensation ratio for IPF3/IOH is as follows:

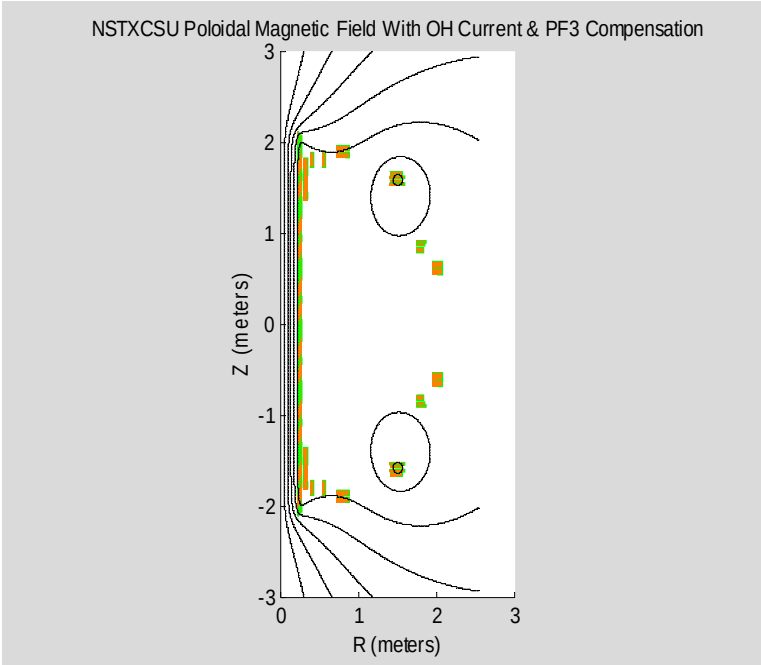
```
interp1(RHO(301,:), -BZcoil(301,:,1)./(BZcoil(301,:,6)+BZcoil(301,:,9)),0.934)

ans =

    0.2543
```

This ratio is used to calculate and contour plot total magnetic flux from the combination of OH plus PF3U&L adjusted to match 0.2543 of the OH current.

```
clf;
ylim([-3 3]);
xlim([0 3]);
for iwinding=1:32;
rwindingmin=winding_r(iwinding)-winding_dr(iwinding)/2;
rwindingmax=winding_r(iwinding)+winding_dr(iwinding)/2;
zwindingmin=winding_z(iwinding)-winding_dz(iwinding)/2;
zwindingmax=winding_z(iwinding)+winding_dz(iwinding)/2;
H(iwinding)=rectangle('Position',[rwindingmin zwindingmin rwindingmax-rwindingmin, zwindingmax-
zwindingmin], 'EdgeColor','g','LineWidth',0.0001,'LineStyle','-.','FaceColor',[1 0.5 0]);
end
hold on
FL=24000*(FLUXcoil(:,:,1)+0.2543*(FLUXcoil(:,:,6)+FLUXcoil(:,:,9)));
contour(RHO,ZETA,sqrt(FL),'k','LineWidth',.1); axis image;xlim([0 3]),ylim([-3 3]);
hold off
xlabel('R (meters)'),ylabel('Z (meters)'),title('NSTXCSU Poloidal Magnetic Field With OH Current
& PF3 Compensation')
```



The above flux contour plot shows that the properly compensated OH startup flux is indeed steered away from the plasma start-up region on the Z=0 m horizontal midplane at R=0.934 m.

Approximate PLASMA Magnetics Representation For Present Engineering Purposes

There are significant issues as to how to represent the spatial distribution of current in the plasma itself, which completely determines its field and flux patterns. The plasma's internal current distribution is not uniform as in PF coil windings but instead varies according to details of the plasma's internal pressure equilibrium, density and temperature. However, it should be remembered that the present purposes are focused mainly on approximately evaluating the forces and stresses on TF coil and PF coil components outside the NSTX vacuum vessel which are involved in CS Upgrade engineering design. These forces are not very sensitive to the detailed distributions of distant currents. Knowledge of the total plasma current and its current center location may by itself be adequate for the present purposes, and some additional very approximate information about the plasma current's spatial distribution may provide enough improvement in the present approximations to be useful for engineering purposes.

I solicited and then followed J.Menard's advice in this matter. His suggested approximation to use herein is to represent the plasma as a current matching the total plasma current, with a constant current density distributed over a rectangular cross section (thus fitting in with the PF coil calculation algorithms used herein) ***but in which the rectangle's width matches the actual plasma width on the midplane and the rectangle's height is chosen so that the rectangle's area matches the plasma cross section's total area***. Now, since the engineering approximation desired here is to use a ***single*** plasma current representation for all expected plasma equilibria, the "matches" criterion must be interpreted loosely in an approximate and statistical sense, for it would not be possible for a single rectangular current distribution to precisely result in equality of cross section areas individually for all plasmas. Fortunately he has provided in Ref.4 a summary of 96 computed equilibria for the CS upgrade, so some statistical calculations were feasible.

For any rectangular uniform current distribution representing the plasma, a computed equilibrium set of specified coil and plasma currents can be used to calculate the corresponding flux function. Saddle x-points in that flux function can be automatically found and used to set levels plotted in a flux contour plot. The innermost such closed contour is then identified as the Last Closed Flux Surface (LCFS) and used further to calculate the plasma's cross sectional area. This procedure was automated and followed for

each of the 96 computed equilibria. In order to do these calculations , two special m-file subroutines first had to be developed, find_xpoint, and closed_contour_parms. These are discussed in the following.

The find_xpoint m-file subroutine is listed below. It returns a vector of row indices, a vector of column indices, and a vector of contour levels, for any saddle x-points in the matrix, A, passed to it. Note that for saddle points that occur in the interior of mesh squares, the indices point to the lowest entries for both row and column.

```
function [I,J,C]=find_xpoint(A);
% Finds indices preceding xpoints & their contour levels.
%
% A is a 2D matrix of reals, suitable for a contour plot.
% If it contains any saddle points (i.e., xpoints) then this
% procedure finds them. I is the vector of row indices preceding
% xpoints, J is the corresponding vector of column indices, and
% C is the corresponding vector of xpoint contour levels.
% R. Woolley, 2009
%-------------------------------------------------------------
%
% The following detects saddle points in interior of mesh squares.
A11=A(1:end-1,1:end-1);
A21=A(2:end,1:end-1);
A12=A(1:end-1,2:end);
A22=A(2:end,2:end);
ALPHA=(A11-A12)./(A11-A12-A21+A22);
BETA=(A11-A21)./(A11-A12-A21+A22);
[I,J]=find(zeros(size(ALPHA))<=ALPHA & ALPHA<ones(size(ALPHA)) & zeros(size(BETA))<=BETA &
BETA<ones(size(BETA)));
IND=sub2ind(size(A11),I,J);
C=(A11(IND).*A22(IND)-A12(IND).*A21(IND))./(A11(IND)-A12(IND)-A21(IND)+A22(IND));
%
%The following finds saddle points coinciding with mesh points
[m,n]=size(A);
B=logical(zeros(size(A)));
for i=2:m-1;
    for j=2:n-1;
        B(i,j)=((A(i,j)>A(i-1,j))&(A(i,j)>A(i+1,j))&(A(i,j)<A(i,j-1))&(A(i,j)<A(i,j+1)) );
        B(i,j)=B(i,j)|((A(i,j)<A(i-1,j))&(A(i,j)<A(i+1,j))&(A(i,j)>A(i,j-1))&(A(i,j)>A(i,j+1)) );
    end
end
[I1,J1]=find(B);
IND1=sub2ind(size(B),I1,J1);
C1=A(IND1);
%Then the lists are concatenated.
I=[I; I1];
J=[J; J1];
C=[C; C1];
```

The other m-file subroutine is closed_contour_parms, as listed below. This procedure is given a contour matrix formatted as it results from MATLAB's contour plotting algorithms, and it selects from that matrix any closed contour having the specified level, clevel, which also encloses a particular point, (xinside, yinside). If there are no such contours then NaN results are output. Otherwise the first contour found that matches these criteria is analyzed to determine its enclosed area and its minimum and maximum intersections with y=yinside.

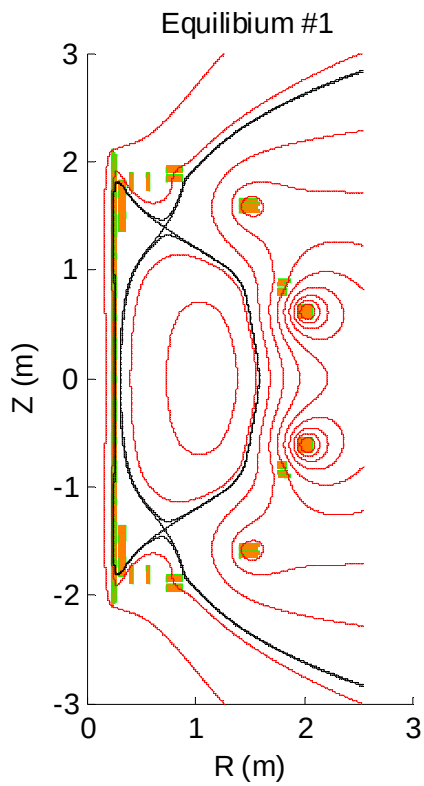
```
function  [Area,xmin,xmax]=closed_contour_parms(Cm,clevel,xinside,yinside)
% Calculates a closed 2D contour's area and its x-values at specified y.
% Cm is the contour matrix as produced by the low-level function, contourc,
% clevel is the scalar value of its particular contour to be analyzed, and
% (xinside, yinside) are coordinates of any point inside that contour. If no
% closed contour specified by Cm matches these inputs then NaN values are
% returned; otherwise, the first closed contour specified by Cm that has
% the stated level and encloses the specified inside point is used.  The
% total (absolute) area enclosed by the contour is calculated, and also the
% two extreme x-values where this contour crosses y=yinside are found and
```

```
% are set equal to xmin and xmax.
% R.Woolley 2009
%
Area=NaN;xmin=NaN;xmax=NaN;
[m,n]=size(Cm);
if (m==2);
    nc=1;
    while (nc<=n);
     np=Cm(2,nc);
%    Check whether contour has the specififed level and is closed:
     if Cm(1,nc)==clevel && Cm(1,nc+1)==Cm(1,nc+np) && Cm(2,nc+1)==Cm(2,nc+np);
         C=Cm(:,(nc+1):(nc+np))-[xinside;yinside]*ones(1,np);
         a=0.;
         for i=1:np-1;
             a=a+atan2(C(1,i)*C(2,i+1)-C(2,i)*C(1,i+1),C(1,i)*C(1,i+1)+C(2,i)*C(2,i+1));
         end
         a=round(abs(a/2/pi));
         if a~=1;
             nc=nc+np+1;
             continue
         else
%  Specified contour has been found, so calculate Area, xmin, xmax
             Area=0.;
             xmin=0;
             xmax=0;
             for i=1:np-1
                 Area=Area+(C(2,i+1)+C(2,i))*(C(1,i)-C(1,i+1))/2.;
                 if (C(2,i)>=0 && C(2,i+1)<=0 || C(2,i)<=0 && C(2,i+1)>=0);
                     if C(2,i)==C(2,i+1);
                         xmin=min(min(xmin,C(1,i)),C(1,i+1));
                         xmax=max(max(xmax,C(1,i)),C(1,i+1));
                     else
                         x=(C(1,i)*C(2,i+1)-C(1,i+1)*C(2,i))/(C(2,i+1)-C(2,i));
                         xmin=min(xmin,x);
                         xmax=max(xmax,x);
                     end
                 end
             end
             xmin=xmin+xinside;
             xmax=xmax+xinside;
             Area=abs(Area);
             return
         end
     else
         nc=nc+np+1;
         continue
     end
    end
end
```

These m-files were used with other computational cells to step through the 96 coil and plasma current combinations listed for the equilibria in Ref.4. Unfortunately 2 of the 96 equilibria failed to yield closed flux surfaces passing through the automatically found saddle x-points. It may be significant that for these two cases J.Menard's computed plasma equilibria plots of Ref. 4 were both quite vertically asymmetric, whereas the plasma current rectangle assumed here was vertically centered on the midplane. However, in the other 94 of the 96 plasma equilibria, for a range of assumed rectangle heights the automatically located saddle x-points robustly yielded closed flux surfaces passing through the same x-points, and 94 values of LCFS area were obtained for each assumed rectangle height. Choosing the median of these areas to compute the next iteration's assumed rectangle height by J.Menard's suggested algorithm resulted in quick convergence to a rectangle half-height of 0.99575m. However, if the median were ignored then the final range of computed half-heights for the 94 equilibrium cases ranged from 0.9377 to 1.1950. Because this is a fairly wide band it was decided to simply set the plasma representation's uniform current density rectangle half-height to exactly 1.0 m. The inner and outer edges of the modeled plasma current rectangle were located at R=0.52 m and R=1.63 m, respectively.

Following the same algorithms as used for OH and PF coils, the br, bz, and flux data per plasma current ampere were then calculated on the spatial (R,Z) grid and saved to disk in the 14th page of the respective BRcoil, BZcoil, FLUXcoil ND arrays.

An example appears below of one of the 94 successful plasma equilibrium poloidal flux plots generated using PF currents from Ref.4 and the final rectangular plasma model. It should be understood that this calculation is approximate since it does not reflect details of the plasma current distribution determined by the plasma equilibrium. However, in spite of that it is sufficiently accurate to predict xpoints in approximately the same locations as Ref.4 which reflected a detailed plasma model.



<u>Excel file: A Poloidal Magnetics Calculation Tool</u>

Although the calculated MATLAB spatial arrays for magnetic field an flux per unit currents in each coil circuit and plasma would in principle provide a useful capability for anyone on the NSTX CSU project to evaluate forces for postulated conditions, the reality is that more personnel are fluent using Excel than MATLAB. Therefore it was decided to export the magnetic data computed in MATLAB into Excel worksheets and to implement Excel algorithms calculating the magnetic field and flux patterns for any input set of currents.

This was done, resulting in Ref.2 which is being distributed by email along with this memo. However, it was not possible to reproduce the 1 cm spatial resolution for the full grid without exceeding the maximum size of email attachments permitted at PPPL. Therefore the resolution was reduced to 5 cm squares. In addition the maximum radial extent was further limited to 2.5 m, which barely includes the TF coil on the horizontal midplane. Using this reduced data volume the computed MATLAB magnetic arrays were automatically copied into 42 worksheet look-up tables, which have subsequently been hidden (although

not password protected) in order to avoid any accidental alteration of their data contents. These 42 worksheets include 14 coil circuit worksheets for each of the spatial patterns of BR, BZ, and FLUX.

Three visible worksheets labeled BR, BZ, and FLUX provide the calculated radial and vertical field components and flux patterns on the same spatial grid. The other worksheet, labeled MAIN, includes 14 cells into which the user can type the coil and plasma currents to be assumed in calculating the total field and flux patterns. The MAIN worksheet also includes for convenience a table, available for "copy and paste" purposes, of the Ref.4 coil and plasma currents for its 96 computed plasma equilibria, and also the pattern of compensated coil currents to use for plasma startup. Finally, the MAIN worksheet also includes three Excel contour plot charts of the spatial magnetic patterns calculated in BR, BZ, and FLUX worksheets. These contour plots automatically update themselves immediately after coil or plasma currents are changed, at least in the MS Office 1999-2003 version. Unfortunately in the MSOffice2007 version of Excel the contour plots do not seem to work correctly. However the MSOffice2007 BR,BZ, and FLUX worksheet numerical results at least are properly updated.

Conclusion

Poloidal magnetic quantities have been calculated for the 16June2009 version of the NSTX CSU PF coil system and for an approximate representation of expected plasmas. Its accuracy is expected to be sufficient for engineering evaluations of TF and PF coil system design changes. The results have been used to create an Excel-based tool (Ref.2) which evaluates poloidal magnetic field and flux at all PF and TF coil system conductors for all static CS upgrade current scenarios.