Introduction

The primary goal of this algorithm is to determine the amplitude and phase of the n=1 component of the magnetic field perturbation produced by the plasma. In general, the total field can be decomposed into toroidal harmonics of mode number "n" with the following Fourier representation:

$$B(\phi,t) = \sum_{n=0}^{N} B_n(t)\cos(n\phi - \theta_n(t))$$

In a tokamak with perfect toroidal symmetry, only the n=0 component is present, and all fields are independent of the toroidal angle coordinate ϕ. When non-axisymmetric error fields are present, or other modes such as the resistive wall mode are unstable or amplifying an error field, the n > 0 field components can also become measurable and important. In particular, if the plasma n > 0 field amplitude becomes too large, the plasma can experience confinement degradation and/or disruption. The goal therefore is to minimize the n > 0 components of the total magnetic field. The n=1 component is generally the largest and most damaging to plasma confinement, so we will focus on this component for the near term.

The minimization of the n=1 component of the field can be done either by knowing the source of the n = 1 field and correcting it (pre-programmed error-field correction), or by measuring the n = 1 field in real-time and trying to eliminate it with active feedback by applying an n=1 field with the RWM/EF coils that opposes n=1 field component from the plasma. The primary difficulty of all these efforts is identifying the plasma n=1 field component. Typical n=0 fields at the sensors used for mode identification are several thousand Gauss, and the field we are trying to measure can be as small as 0.5 Gauss. Hence, extensive signal processing is needed to accurately determine the n=1 component of the field.

Further, digitizer bit-resolution limitations also require us to measure the field at different toroidal locations in the NSTX vacuum chamber and determine the field difference using analog electronics before digitization. This technique effectively eliminates the n=0 component of the measured field and makes the n > 0 field measurements much more accurate. For this reason, the field differences are most important for mode identification as described below, since the field sums (which we also digitize) measure primarily measure the n=0 component. Small fractions of the sum signals are needed however to compensate for sensor gain variations.

Algorithm Outline

The NSTX RWM/EF sensor signal processing can be divided into the two major components outlined below. The IDL source code for all parts of this document can be found in:

/p/nstxusr1/nstx-users/jmenard/analysis/mhd/rwmef/routines/rwmef_calibration_routines.pro

The function get_rwmef_compensated_data() is used to calculate the fully compensated data, and an example of how to use this function is given in:

/p/nstxusr1/nstx-users/jmenard/analysis/mhd/rwmef/routines/comp_example

This example reads the calibrated data from the tree, then applies various sensor compensation terms. The "/alpf" keyword forces the IDL code to use the analog low-pass filtered raw data (see below). The goal of the real-time (PCS) RWM/EF sensor signal processing is to match the output of the above IDL compensation function for all 48 BR/BP sensor sums and differences. Then, a mode identification algorithm can be applied to the 24 sensor differences. Both B_P and B_R differences, or a combination of their information, will be used for mode identification.
The outline:

1. Signal processing in the real-time data acquisition program ACQ

    1.1. Raw data acquisition
    1.2. Sensor pair gain calibration
    1.3. PF, TF, and RWM/EF "static" pickup compensation

2. Mode identification algorithm

    2.1. Sensor compensation
        2.1.1. Sensor gain compensation
        2.1.2. PF transient pickup compensation from filtered loop voltages
        2.1.3. TF transient pickup compensation from derivative of TF current
        2.1.4. OH × TF non-linear cross-term compensation
        2.1.5. Current ramp-time and I_P threshold dependent PF transient compensation
        2.1.6. RWM/EF transient compensation from derivative of IRWM currents
        2.1.7. Archiving of compensated sum and difference signals

    2.2. Mode amplitude and phase determination
        2.2.1. Sensor difference base-line zeroing
        2.2.2. User-specified mode decomposition matrix

1. Signal processing in the real-time data acquisition algorithm

1.1. Raw data acquisition

Raw data acquisition here corresponds to the raw in-vessel RWM/EF BR and BP sensor data acquired by PSRTC in ACQ. These signals have units of volts, and all correspond to integrated sensor sums and differences. In addition, the BP difference signals undergo analog low-pass filtering with an f = 2kHz -3dB point to remove aliased high-frequency f > 5kHz MHD mode activity which might otherwise enter the RWM/EF control loop. The filtered BP difference signals will be used for the initial versions of active control.

The corresponding OPERATIONS tag names for these signals are:

Raw BP upper sums	Raw BP upper differences Analog low-pass filtered	Raw BP lower sums	Raw BP lower differences Analog low-pass filtered
\RAW_BBPRWMPPPUS1	\RAW_BFBPRWPPPUD1	\RAW_BBPRWMPPPLS1	\RAW_BFBPRWPPPLD1
\RAW_BBPRWMPPPUS2	\RAW_BFBPRWPPPUD2	\RAW_BBPRWMPPPLS2	\RAW_BFBPRWPPPLD2
\RAW_BBPRWMPPPUS3	\RAW_BFBPRWPPPUD3	\RAW_BBPRWMPPPLS3	\RAW_BFBPRWPPPLD3
\RAW_BBPRWMPPPUS4	\RAW_BFBPRWPPPUD4	\RAW_BBPRWMPPPLS4	\RAW_BFBPRWPPPLD4
\RAW_BBPRWMPPPUS5	\RAW_BFBPRWPPPUD5	\RAW_BBPRWMPPPLS5	\RAW_BFBPRWPPPLD5
\RAW_BBPRWMPPPUS6	\RAW_BFBPRWPPPUD6	\RAW_BBPRWMPPPLS6	\RAW_BFBPRWPPPLD6

Raw BR upper sums	Raw BR upper differences	Raw BR lower sums	Raw BR lower differences
\RAW_BBRRWMPPPUS1	\RAW_BBRRWMPPPUD1	\RAW_BBRRWMPPPLS1	\RAW_BBRRWMPPPLD1
\RAW_BBRRWMPPPUS2	\RAW_BBRRWMPPPUD2	\RAW_BBRRWMPPPLS2	\RAW_BBRRWMPPPLD2
\RAW_BBRRWMPPPUS3	\RAW_BBRRWMPPPUD3	\RAW_BBRRWMPPPLS3	\RAW_BBRRWMPPPLD3
\RAW_BBRRWMPPPUS4	\RAW_BBRRWMPPPUD4	\RAW_BBRRWMPPPLS4	\RAW_BBRRWMPPPLD4
\RAW_BBRRWMPPPUS5	\RAW_BBRRWMPPPUD5	\RAW_BBRRWMPPPLS5	\RAW_BBRRWMPPPLD5
\RAW_BBRRWMPPPUS6	\RAW_BBRRWMPPPUD6	\RAW_BBRRWMPPPLS6	\RAW_BBRRWMPPPLD6

1.2. Sensor pair gain calibration

For this level of calibration, digitizer offset and integrator drift are removed by ACQ by using data from before t = -0.5s when no coils are active. In addition, all raw signals are multiplied by a linear gain coefficient "G0" read from the MDS+ tree to convert from volts to calibrated physical units – in this case units of Tesla.

The corresponding OPERATIONS tag names for these signals are:

Calibrated BP upper sums	Calibrated BP upper diffs Analog low-pass filtered	Calibrated BP lower sums	Calibrated BP lower diffs Analog low-pass filtered
\CAL_BBPRWMPPPUS1	\CAL_BFBPRWPPPUD1	\CAL_BBPRWMPPPLS1	\CAL_BFBPRWPPPLD1
\CAL_BBPRWMPPPUS2	\CAL_BFBPRWPPPUD2	\CAL_BBPRWMPPPLS2	\CAL_BFBPRWPPPLD2
\CAL_BBPRWMPPPUS3	\CAL_BFBPRWPPPUD3	\CAL_BBPRWMPPPLS3	\CAL_BFBPRWPPPLD3
\CAL_BBPRWMPPPUS4	\CAL_BFBPRWPPPUD4	\CAL_BBPRWMPPPLS4	\CAL_BFBPRWPPPLD4
\CAL_BBPRWMPPPUS5	\CAL_BFBPRWPPPUD5	\CAL_BBPRWMPPPLS5	\CAL_BFBPRWPPPLD5
\CAL_BBPRWMPPPUS6	\CAL_BFBPRWPPPUD6	\CAL_BBPRWMPPPLS6	\CAL_BFBPRWPPPLD6

Calibrated BR upper sums	Calibrated BR upper diffs	Calibrated BR lower sums	Calibrated BR lower diffs
\CAL_BBRRWMPPPUS1	\CAL_BBRRWMPPPUD1	\CAL_BBRRWMPPPLS1	\CAL_BBRRWMPPPLD1
\CAL_BBRRWMPPPUS2	\CAL_BBRRWMPPPUD2	\CAL_BBRRWMPPPLS2	\CAL_BBRRWMPPPLD2
\CAL_BBRRWMPPPUS3	\CAL_BBRRWMPPPUD3	\CAL_BBRRWMPPPLS3	\CAL_BBRRWMPPPLD3
\CAL_BBRRWMPPPUS4	\CAL_BBRRWMPPPUD4	\CAL_BBRRWMPPPLS4	\CAL_BBRRWMPPPLD4
\CAL_BBRRWMPPPUS5	\CAL_BBRRWMPPPUD5	\CAL_BBRRWMPPPLS5	\CAL_BBRRWMPPPLD5
\CAL_BBRRWMPPPUS6	\CAL_BBRRWMPPPUD6	\CAL_BBRRWMPPPLS6	\CAL_BBRRWMPPPLD6

An example of the required calibration (simple multiplication) is as follows:

G0 = MDSVALUE('\OPERATIONS::TOP.MAGNETICS.RWMEFSENSOR.B.CALIBRATION:BBPRWMPPPLS6.PARAMETERS:G0')

\CAL_BBRRWMPPPLS6 = G0 × \RAW_BBRRWMPPPLS6

Note that filtered BP difference G0 coefficients have MDS+ tag names that follow the naming convention of the suffixes of the \CAL_ and \RAW_ tag names. For example,

G0 = MDSVALUE('\OPERATIONS::TOP.MAGNETICS.RWMEFSENSOR.B.CALIBRATION:BFBPRWPPPLD2.PARAMETERS:G0')

corresponds to the G0 calibration coefficient for calibrated signal \CAL_BFBPRWPPPLD2

In the OPERATIONS tree, the above calibration is performed using the TDI function: MDSPLUS$:[PPPL]CALIB_BSENSOR_B_RWMEF.FUN

1.3. PF, TF, RWM/EF coil current "static" pickup compensation

The most effective error field and RWM detection and control schemes are likely to be based on measuring and minimizing the plasma non-axisymmetric field and/or response to non-axisymmetric error fields. Isolating the plasma response from other sources of non-axisymmetric field requires extensive sensor compensation in order to obtain a null vacuum response for all difference sensors from all coil systems. In particular, the combination of sensor gain inaccuracies, sensor misalignment, and inherent device non-axisymmetries (error fields) combine to lead to a non-zero sensor BR/BP difference signal when the vacuum sensor response of the ideal system should be identically zero. The dominant "static" compensation terms are from the PF5 coil system (because the coils are close to the sensors) and from the TF coils. All other PF coil systems contribute as well, and the RWM/EF coils obviously produce large vacuum non-axisymmetric field. Thus, for coil current compensation, each BR/BP sum and difference signal has subtracted from it a sum of terms proportional to the PF, TF, and RWM/EF coil currents on NSTX.

The full list of compensation terms in the compensation sum is:

Compensation =	GTF_PCS	× ITF	+
	GOH	× IOH	+
	GPF1AU	× IPF1AU	+
	GPF2U	× IPF2U	+
	GPF3U	× IPF3U	+
	GPF4	× IPF4	+
	GPF5	× IPF5	+
	GPF3L	× IPF3L	+
	GPF2L	× IPF2L	+
	GPF1AL	× IPF1AL	+
	GPF1B	× IPF1B	+
	GRWM1	× IRWM1	+
	GRWM2	× IRWM2	+
	GRWM3	× IRWM3	+
	GRWM4	× IRWM4	+
	GRWM5	× IRWM5	+
	GRWM6	× IRWM6	

Consistent with the nomenclature of the IDL source code (with the exception that RWM/EF current is treated separately in the IDL code), the prefix "CCPC" is used to denote "Coil Current Pickup Compensation" as described above. An example of a CCPC BR lower difference signal is:

```
\CCPC_BBRRWMPPPLD6 = \COR_BBRRWMPPPLD6 + Compensation
```

See Section 2.1.1 for the definition of the \COR_* signals.

Each compensation coefficient can be obtained from the MDS+ tree using tags:

GPF5 = MDSVALUE('\OPERATIONS::TOP.MAGNETICS.RWMEFSENSOR.B.CALIBRATION:BBRRWMPPPLD6.PARAMETERS:GPF5')

Each PF and TF current above is resident in ACQ memory and is assumed to have opposite sign to that in the OPERATIONS and ENGINEERING "Analysis" tree. This is the reason the compensation term is added to the gain corrected data rather than subtracted from it. Finally, the compensation above uses 6 RWM/EF coil currents even though there are presently only 3 SPA power supplies powering the RWM/EF coils. This allows for the coils to be connected with different polarities and also allows for a possible upgrade to 6 SPA units. Retaining the engineering/physics swapped-polarity convention, the RWM/EF currents presently relate to the SPA currents as follows:

IRWM1 = -\ENGINEERING::IRWM1:SIGN × ISPA2 (inside ACQ)
IRWM2 = -\ENGINEERING::IRWM2:SIGN × ISPA3
IRWM3 = -\ENGINEERING::IRWM3:SIGN × ISPA1
IRWM4 = -\ENGINEERING::IRWM4:SIGN × ISPA2
IRWM5 = -\ENGINEERING::IRWM5:SIGN × ISPA3
IRWM6 = -\ENGINEERING::IRWM6:SIGN × ISPA1

2. Mode Identification Algorithm

2.1. Sensor compensation

2.1.1. Sensor pair gain correction

All digitized BR and BP signals correspond to magnetic field sums or differences. The G0 parameter represents a mean gain of the sensor pair. To take into account the slightly different gains of individual sensors in a sum/difference pair, a small fraction of the sum/difference is added to the difference/sum using the "G1" parameter in the MDS tree. These corrected signals are not archived in the MDS tree, and are presently computed only in IDL post-processing routines. However, for the purposes of this document, these signals will be assigned a tag prefix "COR" to distinguish them from "RAW" and CAL".

For a sum signal, the G1 parameter is obtained from the corresponding calibration parameter tag for the sum (not the difference), but the correction is proportional to the corresponding difference signal:

```
G1 = MDSVALUE('\OPERATIONS::TOP.MAGNETICS.RWMEFSENSOR.B.CALIBRATION:BBRRWMPPPLS6.PARAMETERS:G1')

\COR_BBRRWMPPPLS6 = \CAL_BBRRWMPPPLS6 + G1 × \CAL_BBRRWMPPPLD6
```

Similarly, for a difference signal, the G1 parameter is obtained from the corresponding calibration parameter tag for the difference (not the sum), but the correction is proportional to the corresponding sum signal:

```
G1 = MDSVALUE('\OPERATIONS::TOP.MAGNETICS.RWMEFSENSOR.B.CALIBRATION:BBRRWMPPPLD6.PARAMETERS:G1')

\COR_BBRRWMPPPLD6 = \CAL_BBRRWMPPPLD6 + G1 × \CAL_BBRRWMPPPLS6
```

These gain corrections are to be applied to all 48 BR and BP sums and differences, and the corresponding function is get_rwmef_gain_compensated_data() in the IDL source code.

2.1.2. PF transient pickup compensation from filtered loop voltages

PF transient pickup is largest during the plasma current ramp-up and is typically small during flat-top, especially when the toroidal loop voltage is low. This compensation is often turned off late in the discharge to avoid confusion arising from vertical position oscillations and VDEs. Thus, for now, this compensation will be ignored.

2.1.3. TF transient pickup compensation from derivative of TF current

The compensation is only important during TF current ramps. Thus, so long as the TF current is held constant during a plasma discharge, this compensation can be ignored.

2.1.4. OH × TF non-linear cross-term compensation

This compensation is quite important for the BR sensors (and indicates a real non-linear radial field), but is small for the BP sensors. Since the BP sensors will likely be utilized first for RWM/EF control, this compensation term will be ignored for now.

2.1.5. I_P ramp-time and threshold dependent PF transient compensation

In the IDL routines for RWM/EF sensor signal compensation, this routine controls when the PF transient pickup compensation from filtered loop voltages is applied. This will be ignored for now.

2.1.6. RWM/EF transient compensation from derivative of IRWM currents

In the IDL routines for RWM/EF sensor signal compensation, this routine performs both static and transient compensation for RWM/EF coil current pickup. However, in the real-time compensation algorithm, the static RWM/EF coil current pickup is now part of signal processing section 1.3 and need not be repeated here. The transient compensation for RWM/EF coil currents is likely to be important for both BP and BR signals when the coil current frequency (slew-rate) is sufficiently high. This compensation will be implemented during the 2007 run if time permits.

2.1.7. Archiving of compensated sum and difference signals

The IDL routine stores the fully compensated sum and difference signals from the above methods 2.1.1 to 2.1.6 in the MDS+ OPERATIONS tree with the following tag names:

Compensated BP upper sums	Compensated BP upper diffs Analog low-pass filtered	Compensated BP lower sums	Compensated BP lower diffs Analog low-pass filtered
\B_BPRWMPPPUS1	\B_BPRWMPPPUD1_LPF	\B_BPRWMPPPLS1	\B_BPRWMPPPLD1_LPF
\B_BPRWMPPPUS2	\B_BPRWMPPPUD2_LPF	\B_BPRWMPPPLS2	\B_BPRWMPPPLD2_LPF
\B_BPRWMPPPUS3	\B_BPRWMPPPUD3_LPF	\B_BPRWMPPPLS3	\B_BPRWMPPPLD3_LPF
\B_BPRWMPPPUS4	\B_BPRWMPPPUD4_LPF	\B_BPRWMPPPLS4	\B_BPRWMPPPLD4_LPF
\B_BPRWMPPPUS5	\B_BPRWMPPPUD5_LPF	\B_BPRWMPPPLS5	\B_BPRWMPPPLD5_LPF
\B_BPRWMPPPUS6	\B_BPRWMPPPUD6_LPF	\B_BPRWMPPPLS6	\B_BPRWMPPPLD6_LPF

Comp BR upper sums	Comp BR upper diffs	Comp BR lower sums	Comp BR lower diffs
\B_BRRWMPPPUS1	\B_BRRWMPPPUD1	\B_BRRWMPPPLS1	\B_BRRWMPPPLD1
\B_BRRWMPPPUS2	\B_BRRWMPPPUD2	\B_BRRWMPPPLS2	\B_BRRWMPPPLD2
\B_BRRWMPPPUS3	\B_BRRWMPPPUD3	\B_BRRWMPPPLS3	\B_BRRWMPPPLD3
\B_BRRWMPPPUS4	\B_BRRWMPPPUD4	\B_BRRWMPPPLS4	\B_BRRWMPPPLD4
\B_BRRWMPPPUS5	\B_BRRWMPPPUD5	\B_BRRWMPPPLS5	\B_BRRWMPPPLD5
\B_BRRWMPPPUS6	\B_BRRWMPPPUD6	\B_BRRWMPPPLS6	\B_BRRWMPPPLD6

The full signal compensation performed in ACQ should be compared to the above tags in the “pecomp” IDL routine to validate the real-time sensor compensation algorithms.

2.2. Mode amplitude and phase determination

For the second half of the 2006 run, 12 compensated (time-dependent) BP sensor difference signals combined with 12 BR differences signals ≡ $\boldsymbol{D_j(t)}$ will be used for mode identification:

j	D_j
1	\B_BPRWMPPPUD1_LPF
2	\B_BPRWMPPPUD2_LPF
3	\B_BPRWMPPPUD3_LPF
4	\B_BPRWMPPPUD4_LPF
5	\B_BPRWMPPPUD5_LPF
6	\B_BPRWMPPPUD6_LPF
7	\B_BPRWMPPPLD1_LPF
8	\B_BPRWMPPPLD2_LPF
9	\B_BPRWMPPPLD3_LPF
10	\B_BPRWMPPPLD4_LPF
11	\B_BPRWMPPPLD5_LPF
12	\B_BPRWMPPPLD6_LPF
13	\B_BRRWMPPPUD1
14	\B_BRRWMPPPUD2
15	\B_BRRWMPPPUD3
16	\B_BRRWMPPPUD4
17	\B_BRRWMPPPUD5
18	\B_BRRWMPPPUD6
19	\B_BRRWMPPPLD1
20	\B_BRRWMPPPLD2
21	\B_BRRWMPPPLD3
22	\B_BRRWMPPPLD4
23	\B_BRRWMPPPLD5
24	\B_BRRWMPPPLD6

2.2.1. Sensor difference base-line zeroing

Even with accurate vacuum compensation, the BR and BP difference signals will generally not be identically zero in the presence of plasma. Thus, it will be desirable to choose a time-range of interest during the plasma discharge during which the BR and BP difference signals are sampled and averaged. The average value of each difference signal during this time interval will then be subtracted from the same signal starting at the end of the averaging window. Thus, for difference signals which vary slowly following the averaging window, we expect the modified difference signal to be very close to zero. This base-line zeroing is expected to enhance sensitivity to non-axisymmetric plasma modes.

The averaging time range will be specified with an on-off switch programmable wave form, and the same time range will be applied to all difference signals. When the switch transitions from zero to one, averaging will begin, and when the switch back-transitions from one to zero, the averaging will end. For each sensor difference signal $D_j(t)$ of index "j", the average value is computed by simply adding together the signal values at times during the

averaging window and dividing by the number of times = "*N*" within the window using the formula:

$$\overline{D}_j = \sum_i D_j(t_i) / N$$

Obviously at least one sample is needed ($N \geq 1$) for this to work, and after the averaging has ended at "t_{avgend}", each difference signal will have its average value subtracted from it according to:

$$D_j(t > t_{avgend}) \rightarrow D_j(t > t_{avgend}) - \overline{D}_j$$

2.2.2. User-specified mode decomposition matrix

In general on NSTX, the RWM/EF sensor array allows us to compute the n=1, 2 and 3 components of the non-axisymmetric field. The matrix relationship between the sensor difference signals and the mode amplitude and phase is a somewhat complicated function of the toroidal positions of the sensors used, and for some sensor configurations can lead to singular matrix inverses, i.e. will completely fail at mode identification. On NSTX, the sensor difference angles were chosen to eliminate singular inverse matrices even when several difference signals are no longer useable – most often because of sensor failure.

The general matrix equation we are trying to solve is:

$$\mathbf{A_{jk} \cdot x_k(t) = D_j(t)}$$

Here $\mathbf{A_{jk}}$ is a matrix of sensor difference sines and cosines which depends on the toroidal positions of the sensors used, $\mathbf{x_k(t)}$ is the time-varying column vector of quadrature components which are used to determine the amplitude and phase of modes with different toroidal mode number n, and $\mathbf{D_j(t)}$ is the time-varying column vector of 24 compensated BP and BR difference signals. $\mathbf{A_{jk}}$ is not in general a square matrix, so the linear system above can range from overconstrained to underconstrained. In this situation, Singular Value Decomposition (SVD) can be used to solve the set of simultaneous equations Ax=b. The SVD of the m x n matrix A gives three matrices U, W, and V such that $A = UWV^T$. U is an m x n column orthogonal matrix, W is a diagonal matrix of the positive or zero singular values, and V is an n x n orthogonal matrix. The inverse of A can then be written as $A^{-1} = V \cdot \left[diag(1/w_j) \right] \cdot U^T$ where w_j are the singular values from W. In practice, small values of w_j (those close to machine floating point precision) are set to zero and the corresponding values of $1/w_j$ are also set to zero in computing A^{-1}. The solution to the set of equations is then given by: $x = V \cdot \left[diag\left(1/w_j\right) \right] \cdot \left(U^T \cdot b \right)$. If b is not in the range of A, then there is no exact solution to the set of equations, but the solution to x as determined by this method will be the solution which minimizes the residual r defined by $r \equiv \left| A \cdot x - b \right|$.

In the off-line IDL algorithm used for mode identification, the inputs are an index list of working sensors and the range of toroidal mode numbers allowed in the decomposition. Then, the SVD pseudo-inverse A^{-1} is computed and multiplied by the entire time series of sensor difference data. In principle, this general SVD inverse method can also be implemented in the real-time mode-ID algorithm, but is unlikely this can be completed in time for the end of the FY2006 run.

Since the output of the SVD is a simple matrix of coefficients which does not change during a shot, we will simply enter the matrix coefficients into the PCS manually. Also, by restricting ourselves to only computing the n=1 field component, we reduce the number of matrix elements needed. We wish to be able to archive and restore this matrix, and a GUI interface to edit the matrix is highly desirable since 48 parameters will be needed and time-dependent waveforms are not needed. (It may be desirable for the GUI to have 2 matrices – one for BP, another for BR, but the full matrix should be archived and restorable.)

The matrix $\mathbf{Y_{kj}} = \mathbf{A^{-1}_{jk}}$ we wish to enter has 24 columns and 2 rows. The first 12 rows correspond to the B_P sensors and the second twelve rows to the B_R sensors. These separate sections of the matrix will be used to compute n=1 amplitude and phase information based on B_R measurement only, B_P measurements only, and a combination of B_R and B_P measurements. In particular, multiplying parts of $\mathbf{Y_{kj}}$ by parts of $\boldsymbol{D_j(t)}$ will be used to compute the three vectors $\boldsymbol{x_k(t)}$ ***as follows***:

$$\mathbf{x}_{k,B_P}(t) = \mathbf{Y}_{kj} \cdot \mathbf{D}_j(t), \quad j = 0:11$$
$$\mathbf{x}_{k,B_R}(t) = \mathbf{Y}_{kj} \cdot \mathbf{D}_j(t), \quad j = 12:23$$
$$\mathbf{x}_{k,B_{all}}(t) = \mathbf{x}_{k,B_P}(t) + \mathbf{x}_{k,B_R}(t)$$

The components of these vectors $x_{k,S}$, with S={B_R, B_P or B_{all}) are:

$$\mathbf{x_{k,s}} = \begin{bmatrix} \mathbf{x_{1,s}(t)} \\ \mathbf{x_{2,s}(t)} \end{bmatrix} = \begin{bmatrix} \mathbf{B_{1,s}(t)\cos(\theta_{1,s}(t))} \\ \mathbf{B_{1,s}(t)\sin(\theta_{1,s}(t))} \end{bmatrix}$$

where B_1 is the amplitude of the n=1 perturbation and θ_1 is the toroidal angle of the local maximum of the perturbation. B_1 and θ_1 can therefore be determined from the elements of **x** by:

$$B_{1,B_R}(t) = \sqrt{x^2_{1,B_R}(t) + x^2_{2,B_R}(t)}, \quad \theta_{1,B_R}(t) = \arctan\left(\frac{x_{2,B_R}(t)}{x_{1,B_R}(t)}\right)$$
$$B_{1,B_P}(t) = \sqrt{x^2_{1,B_P}(t) + x^2_{2,B_P}(t)}, \quad \theta_{1,B_P}(t) = \arctan\left(\frac{x_{2,B_P}(t)}{x_{1,B_P}(t)}\right)$$
$$B_{1,B_{all}}(t) = \sqrt{x^2_{1,B_{all}}(t) + x^2_{2,B_{all}}(t)}, \quad \theta_{1,B_{all}}(t) = \arctan\left(\frac{x_{2,B_{all}}(t)}{x_{1,B_{all}}(t)}\right)$$

.

To be consistent with all other data presently in the MDS+ tree, for ACQ, this mode-ID algorithm, and the feedback algorithm, the units of the magnetic field

will be **Tesla,** and the units of the phase angle θ_1 will be **Degrees** spanning the range [0,360]. This implies that the θ_1 above should be modified according to:

$$\theta_1(t) \rightarrow \left[(\theta_1(t) \times 180/\pi) + 360\right] \text{MODULUS } 360$$

before they are archived or passed to any other algorithm.

The algorithm should archive all six numbers derived in the above steps, and pass all six of them through to the feedback algorithm.