Gyrosynchrotron code documentation, part I: input, output, and calling conventions for IDL users

The source code can be compiled to produce either a Windows dynamic link library (default names are `MWTransferArr32.dll` and `MWTransferArr64.dll`) or a Linux shared object (default name is `MWTransferArr.so`). These libraries contain two IDL-callable functions:

- `GET_MW` — the function that calculates the emission parameters for a single line-of-sight;
- `GET_MW_SLICE` — the function that is able to calculate the emission parameters for multiple lines-of-sight simultaneously using multi-processor parallelization.

The emission parameters are computed by integrating the radiation transfer equation along the specified line(s)-of-sight. The radiation transfer equation includes contributions of the following emission/absorption mechanisms:

- Gyrosynchrotron (magnetobremsstrahlung); the emission and absorption coefficients are computed according to the formulae and algorithms presented in the paper of Fleishman & Kuznetsov (2010). The electron distribution function can be either analytical (factorized, with the energy and pitch-angle distributions selected from lists of pre-defined distributions) or numerically defined (by an array); a combination of the analytical and numerically defined distributions is possible as well.
- Free-free: electron-ion collisions; the emission and absorption coefficients (for the Maxwellian and kappa-distributions) are computed according to the formulae presented in the paper of Fleishman & Kuznetsov (2014).
- Free-free: electron collisions with neutral hydrogen atoms (at the temperatures below 160×10^3 K only); the absorption coefficient is computed according to the formulae presented in the paper of Stallcop (1974).

In the below description, all data types are referred to according to their IDL specifications.

1 Function `GET_MW`

1.1 Calling conventions

This function is called from IDL using the `call_external` function with the following syntax:

```
res = call_external(libname, 'GET_MW', Ndat, Parms, E, mu, f, RL, /d_value)
```

where `libname` is the name of the appropriate library, `Ndat` is the array specifying the model dimensions, `Parms` is the array specifying various parameters of the emission source, `E`, `mu` and `f` are the arrays describing the electron distribution function (if the array-defined function is used), and `RL` is the array specifying the input emission parameters and receiving the output emission parameters.

1.2 General parameters

The input parameter `Ndat` is the 4-element long-integer array specifying the dimensions of the other data arrays, namely:

- `Ndat[0]` $= N_z$ is the number of volume elements along the line-of-sight (i.e., the number of points where the plasma and magnetic field parameters are specified).

- `Ndat[1]` $= N_E$ is the number of elements in the energy array `E`.
- `Ndat[2]` $= N_\mu$ is the number of elements in the pitch-angle array `mu`.
- `Ndat[3]` $= M$ is the number of parameters used to describe each volume element; should be ≥ 29 to provide compatibility with the original *fast gyrosynchrotron codes* by Fleishman & Kuznetsov (2010) or larger (up to 37) to enable additional capabilities, see below.

1.3 Source parameters

The input parameter `Parms` is the two-dimensional ($M \times N_z$) double-precision floating-point array. Each M-element column of this array `Parms[*, j]` $\equiv$ `ParmLocal` corresponds to a single (jth) volume element along the line-of-sight ($0 \leq j < N_z$). For each volume element, the emission source parameters and the numerical code parameters are specified as follows:

- `ParmLocal[0]` $= S$ is the visible source area, in cm^2.
  Note: Only the value corresponding to the first volume element `Parms[0, 0]` is considered.
- `ParmLocal[1]` $= \Delta z$ is the length of the volume element along the line-of-sight, in cm.
- `ParmLocal[2]` $= T_0$ is the background plasma temperature, in K.
  Note: To turn off the free-free contribution, set $T_0 < 0$; in this case, all other quantities depending on the background plasma temperature (ionization degree etc.) will be computed using $|T_0|$.
- `ParmLocal[2]` $= \varepsilon$ is the matching parameter ε in the thermal/nonthermal analytical electron distributions.
- `ParmLocal[4]` $= \varkappa$ is the parameter $\varkappa$ in the kappa-distribution (analytical electron distribution).
- `ParmLocal[5]` $= N_{\rm nod}$ specifies the integration method and accuracy in the continuous gyrosynchrotron code.
  – If $N_{\rm nod} \geq 1$, then integration over energy is performed using the trapezoidal method with $N_{\rm nod} + 1$ nodes (rounded downwards to the nearest integer).
  – Otherwise, the Romberg method (with the default accuracy of 10^{-5}) is used.
- `ParmLocal[6]` $= E_{\rm min}$ is the low-energy cutoff of the accelerated electrons in the power-law-related analytical electron distributions, in MeV.
- `ParmLocal[7]` $= E_{\rm max}$ is the high-energy cutoff of the accelerated electrons in the power-law-related analytical electron distributions, in MeV.
- `ParmLocal[8]` $= E_{\rm break}$ is the break energy in the double-power-law analytical electron distributions, in MeV.
- `ParmLocal[9]` $= \delta_1$ is the power-law index in the single-power-law distributions or the low-energy power-law index in the double-power-law distributions (analytical electron distributions).
- `ParmLocal[10]` $= \delta_2$ is the high-energy power-law index in the double-power-law analytical electron distributions.

- `ParmLocal[11]` $= n_0$ is the background (thermal) plasma density, in cm^{-3}.
  Note: This parameter is used only if `ParmLocal[29]` = `ParmLocal[30]` $= 0$ (see below). In this case, it is interpreted as the total (ionized+neutral) hydrogen number density, for a purely hydrogen plasma; number densities of the thermal electrons n_e, protons n_p (assuming $n_\mathrm{e} = n_\mathrm{p}$) and neutral hydrogen atoms n_H are computed using the background plasma temperature $|T_0|$ and the Saha equation for the ionization degree. For a sufficiently high temperature ($\rightarrow$ full ionization), $n_\mathrm{e} \rightarrow n_0$.

- `ParmLocal[12]` $= n_\mathrm{b}$ is the number density of nonthermal electrons in the analytical electron distributions, in cm^{-3}.

- `ParmLocal[13]` $= B$ is the magnetic field strength, in G.

- `ParmLocal[14]` $= \theta$ is the angle between the magnetic field and the line-of-sight, in degrees.

- `ParmLocal[15]` $= f_0$ is the parameter controlling how the frequency nodes of the emission spectrum are determined.
  Note: Only the value corresponding to the first volume element `Parms[15, 0]` is considered.
  – If $f_0 > 0$, this parameter is interpreted as the starting frequency (in Hz) to compute the emission spectrum. The frequency nodes are computed automatically and are logarithmically spaced: $f_1 = f_0 10^{\Delta}$, $f_2 = f_1 10^{\Delta}$, $\ldots$, $f_0 10^{(N_f-1)\Delta}$. (see below)
  – Otherwise, the frequency nodes are taken from the array `RL` (see below).

- `ParmLocal[16]` $= \Delta$ is the logarithmic step in frequency (see above); this parameter is used only if $f_0 > 0$.
  Note: Only the value corresponding to the first volume element `Parms[16, 0]` is considered.

- `ParmLocal[17]` specifies the chosen analytical electron distribution over energy (index of the model distribution function, see the separate document `GScodeII.pdf`); non-integer parameters are rounded downwards to the nearest integer.
  Note: To turn off the contribution of the analytical electron distribution function, set `ParmLocal[17] = 0`, which corresponds to the free-free only model.
  Note: If the kappa-distribution (`ParmLocal[17]` $= 6$) is chosen, the free-free contribution is also computed using the formulae for the kappa-distribution (Fleishman & Kuznetsov 2014); in all other cases, the free-free contribution is computed using the formulae for the Maxwellian distribution. By default (i.e., if `ParmLocal[17]` does not match any of available indices), the free-free only model is used.

- `ParmLocal[18]` $= N_f$ is the number of frequency nodes in the emission spectrum (non-integer parameters are rounded downwards to the nearest integer); the nodes are either computed automatically or taken from the array `RL`, depending on the parameter f_0.
  Note: Only the value corresponding to the first volume element `Parms[18, 0]` is considered. The number of nodes must match the dimensions of the array `RL`.

- `ParmLocal[19]` specifies the chosen analytical electron distribution over pitch-angle (index of the model distribution function, see the separate document `GScodeII.pdf`); non-integer parameters are rounded downwards to the nearest integer.
  Note: This parameter is meaningful only if some analytical energy distribution (different from the free-free only model) is selected. By default (i.e., if `ParmLocal[19]` does not match any of available indices), the isotropic distribution is used.

- `ParmLocal[20]` $= \alpha_c$ is the loss-cone boundary in the loss-cone analytical electron distributions, in degrees.

- `ParmLocal[21]` $= \alpha_0$ is the beam direction in the beam-like analytical electron distributions, in degrees.

- `ParmLocal[22]` $= \Delta\mu$ is the loss-cone boundary width or the beam angular width in the loss-cone or beam-like analytical electron distributions, respectively.

- `ParmLocal[23]` $= a_4$ is the coefficient a_4 in the supergaussian analytical electron distribution.

- `ParmLocal[24]` *is currently unused.*

- `ParmLocal[25]` $= f_{cr}^{C}/f_B$ specifies the boundary frequency f_{cr}^{C} (expressed in units of local gyrofrequency) in the hybrid code.
  – If the emission frequency $f > f_{cr}^{C}$ then the emissivity and absorption coefficient are calculated using the continuous code.
  – If $f < f_{cr}^{C}$ then the exact code with summation over cyclotron harmonics is used.
  – If $f_{cr}^{C} = 0$ then the code becomes purely continuous.
  – If $f_{cr}^{C} < 0$ then the code is continuous. In addition, the continuous spectrum is renormalized using the exact parameters calculated at $f = f_{cr}^{WH}$ (see below) in order to improve accuracy.

- `ParmLocal[26]` $= f_{cr}^{WH}/f_B$ specifies the boundary or renormalization frequency f_{cr}^{WH} (expressed in units of local gyrofrequency) in the optimized hybrid code.
  – If $f_{cr}^{WH} < f < f_{cr}^{C}$ then the exact code with summation over cyclotron harmonics uses the approximated expressions for the Bessel functions.
  – If $f < f_{cr}^{C}$ and $f < f_{cr}^{WH}$ then the exact expressions for the Bessel functions are used.
  – If $f_{cr}^{C} < 0$ and $f_{cr}^{WH} > 0$ then the exact emission parameters calculated at $f = f_{cr}^{WH}$ are used to renormalize the spectrum obtained using the continuous code in order to improve accuracy.

- `ParmLocal[27]` controls the behaviour of the hybrid code at the boundary frequencies:
  – If `ParmLocal[27]` $\neq 0$ then the correction factors are applied to the gyrosynchrotron emissivities and absorption coefficients to remove jumps at the frequencies f_{cr}^{C} and (if necessary) f_{cr}^{WH}.
  – If `ParmLocal[27]` $= 0$ then the emission parameters are uncorrected and the jumps may appear.

- `ParmLocal[28]` controls the so-called Q-optimization of the continuous code (see Fleishman & Kuznetsov 2010):
  – If `ParmLocal[28]` $\neq 0$ then the optimization is turned on which improves accuracy.
  – If `ParmLocal[28]` $= 0$ then the optimization is turned off which improves computation speed.

- `ParmLocal[29]` $= n_H$ is the neutral hydrogen number density, in cm^{-3} (see below).

- `ParmLocal[30]` $= n_p$ is the proton number density, in cm^{-3}.
  Note: If either `ParmLocal[29]` or `ParmLocal[30]` are nonzero, the corresponding neutral hydrogen and proton number densities n_H and n_p are used in further computations, and the thermal electron number density n_e is assumed to be $n_e = n_p$; the parameter `ParmLocal[11]` is ignored.

If `ParmLocal[29]` = `ParmLocal[30]` = 0, the local plasma parameters are computed using the total plasma density n_0, temperature T_0, and Saha equation.

- `ParmLocal[31]` specifies whether the contribution of the array-defined electron distribution function is considered:
  – If `ParmLocal[31]` $\neq 0$, contribution of the array-defined electron distribution function is included.
  – If `ParmLocal[31]` = 0, contribution of the array-defined electron distribution function is not included; the parameters `ParmLocal[32 : 36]` are ignored.

- `ParmLocal[32]` (rounded downwards to the nearest integer) specifies how the pitch-angle dependence of the array-defined electron distribution is treated:
  – If `ParmLocal[32]` = 0, the electron distribution at each energy is replaced by isotropic one (averaged over pitch-angle), and the continuous gyrosynchrotron code (at $f > f_{\rm cr}^{\rm C}$) uses the fast Petrosian-Klein approximation.
  – If `ParmLocal[32]` = 1, the electron distribution at each energy is replaced by isotropic one (averaged over pitch-angle), and the continuous gyrosynchrotron code uses the numerical root-finding algorithm by Fleishman & Kuznetsov (2010).
  – If `ParmLocal[32]` ≥ 2, the exact (possibly anisotropic) electron distribution is used.

- `ParmLocal[33]` controls the assumptions about the energy grid for the array-defined electron distribution (used for integration and interpolation over energy):
  – If `ParmLocal[33]` = 0, the code optimized for the logarithmically-spaced energy nodes (with $E_{i+1}/E_i \simeq$ const) is used. All energy and distribution function values must be positive.
  – If `ParmLocal[33]` $\neq 0$, the code optimized for the equidistant energy nodes (with $E_{i+1} - E_i \simeq$ const) is used.
  Note: The code should match the actual energy spacing – this can improve the calculation accuracy greatly.

- `ParmLocal[34 : 35]` *are currently unused.*

- `ParmLocal[36]` controls the 2D interpolation method (for the array-defined electron distribution):
  – If `ParmLocal[36]` = 0, spline interpolation is used.
  – If `ParmLocal[36]` $\neq 0$, local interpolation over $2 - 3$ adjacent nodes is used.
  Note: Spline interpolation is used by default, because it usually provides higher speed and accuracy. Local linear-quadratic interpolation is an experimental mode.

Since such parameters as the visible source area S, starting frequency f_0, frequency step Δ and number of frequencies N_f refer to the whole emission source rather than to a single volume element, they cannot vary along the line-of-sight. These parameters are taken from the first volume element (`Parms[*, 0]`); the corresponding values in other volume elements are ignored.

Unnecessary parameters at the end of the above list can be omitted; i.e., the dimension M of the array `Parms` is allowed to be < 37. In particular, setting $M = 29$ and using only the parameters `Parms[0 : 28, *]` makes the code similar to the previous releases of the *fast gyrosynchrotron codes*. Any parameter `Parms[i, j]` with $i \geq M$ is assumed to be zero.

1.4 Array-defined electron distribution function

The input parameter `E` is the N_E-element ($N_E \geq 3$) double-precision floating-point array specifying the electron distribution nodes in the energy space E_i, in MeV. These values must be monotonically increasing, i.e. $E_{i+1} > E_i$.

Note: If logarithmic energy spacing is used (`ParmLocal[33]` $= 0$), all energy values must be positive ($E_i > 0$).

The input parameter `mu` is the N_μ-element ($N_\mu \geq 3$) double-precision floating-point array specifying the electron distribution nodes in the pitch-angle space; namely, it contains pitch-angle cosines $\mu_j = \cos\alpha_j$. These values must be monotonically increasing, i.e. $\mu_{j+1} > \mu_j$.

Note: The array `mu` should cover the entire range of possible values from -1 to 1 (i.e., $\mu_0 = -1$ and $\mu_{N_\mu - 1} = 1$); otherwise, missing values of the distribution function will be found using extrapolation which can yield incorrect results.

The input parameter `f` is the three-dimensional ($N_z \times N_E \times N_\mu$) double-precision floating-point array specifying the electron distribution function. Namely, each two-dimensional sub-array `f[k, *, *]` ($0 \leq k < N_z$) contains the values of the electron distribution function $f_{ij}^{(k)} = f^{(k)}(E_i, \mu_j)$ at the grid nodes in the energy/pitch-angle space, in the kth volume element along the line-of-sight. The grid nodes are specified by the above-mentioned arrays `E` and `mu`; they are the same for all volume elements. The distribution function in each volume element is assumed to satisfy the following normalization condition:

$$2\pi \int\limits_{E_{\min}}^{E_{\max}} \mathrm{d}E \int\limits_{\mu_{\min}}^{\mu_{\max}} f(E,\mu)\mathrm{d}\mu = n_{\mathrm{b}}, \tag{1}$$

where n_{b} is the local number density of energetic electrons (in cm^{-3}), and the energy is in MeV.

Note: If logarithmic energy spacing is used (`ParmLocal[33]` $= 0$), all values of the distribution function must be positive ($f_{ij} > 0$).

Note: To compute the distribution function and its derivatives properly, the energy and pitch-angle grids must contain at least three nodes each. Therefore, the contribution of the array-defined electron distribution function is computed if (a) $N_E \geq 3$ and (b) $N_\mu \geq 3$ and (c) the contribution of the array-defined distribution in the current volume element is explicitly turned on (`ParmLocal[31]` $\neq 0$). Otherwise, the contribution of the array-defined electron distribution function is not computed.

If the array-defined electron distribution function is not used, the corresponding parameters in the call to `call_external` can be left undefined, e.g.

```
res = call_external(libname, 'GET_MW', Ndat, Parms, 0, 0, 0, RL, /d_value)
```

1.5 Emission parameters

The input/output parameter `RL` is the two-dimensional ($7 \times N_f$) double-precision floating-point array, where the number of frequencies N_f is given by `Parms[18, 0]`. The array rows contain the following data:

- `RL[0, *]` $= f$ is the emission frequency, in GHz.
- `RL[1, *]` $= I_f^{(\mathrm{Lw})}$ is the left-polarized emission intensity (for the weak mode-coupling model), in sfu.
- `RL[2, *]` $= I_f^{(\mathrm{Rw})}$ is the right-polarized emission intensity (for the weak mode-coupling model), in sfu.
- `RL[3, *]` $= I_f^{(\mathrm{Ls})}$ is the left-polarized emission intensity (for the strong mode-coupling model), in sfu.
- `RL[4, *]` $= I_f^{(\mathrm{Rs})}$ is the right-polarized emission intensity (for the strong mode-coupling model), in sfu.

- `RL[5, *]` $= I_f^{(\mathrm{Le})}$ is the left-polarized emission intensity (for the exact mode-coupling model), in sfu.
- `RL[6, *]` $= I_f^{(\mathrm{Re})}$ is the right-polarized emission intensity (for the exact mode-coupling model), in sfu.

Note: All intensities are calculated under the assumption that the emission source is observed from the distance of one astronomical unit (e.g., it is located at the Sun and observed from the Earth). The weak/strong/exact mode-coupling models are explained, e.g., by Cohen (1960).

On input, if $f_0 =$ `Parms[15, 0]` ≤ 0, the row `RL[0, *]` is assumed to contain the list of frequencies f_i (in GHz) for which the emission parameters should be computed. These values must be monotonically increasing, i.e. $f_{i+1} > f_i$. If $f_0 =$ `Parms[15, 0]` > 0, the values in this row are ignored. Other rows of the array `RL` are assumed to contain the initial emission intensities (at the lower boundary of the simulation region, in sfu) for the respective frequencies and mode-coupling models.

After calculations, the row `RL[0, *]` contains the list of the emission frequencies (either the same as on input or computed automatically according to the above-described algorithm, depending on the parameter `Parms[15, 0]`). Other rows of the array `RL` contain the computed emission intensities.

1.6 Return value

The return value (`res`) indicated the status of computation:

- `res` $= 0$: no errors.
- `res` < 0: incorrect format of the input parameters (e.g., not enough arguments).
- `res` > 0: some error occurred during computations (e.g., invalid parameters of either analytical or array-defined electron distribution function were encountered).
  Note: The parameter checking has not been fully implemented yet, therefore some invalid parameter combinations can pass without notice.

If case of any error (`res` $\neq 0$), the array `RL` remains unchanged.

Note: Since an IDL-callable module is unable to check the type of the supplied arguments, all of them (e.g., the type and size of the arrays `Parms` and `RL`) must match exactly the above specifications; otherwise, the results are completely unpredictable.

2 Function GET_MW_SLICE

2.1 Calling conventions

This function is called from IDL using the `call_external` function with the following syntax:

```
res = call_external(libname, 'GET_MW_SLICE',
                    Ndat_m, Parms_m, E, mu, f_m, RL_m, /d_value)
```

where `libname` is the name of the appropriate library, `Ndat_m` is the array specifying the model dimensions, `Parms_m` is the array specifying various parameters of the emission source, `E`, `mu` and `f_m` are the arrays describing the electron distribution function (if the array-defined function is used), and `RL_m` is the array specifying the input emission parameters and receiving the output emission parameters.

2.2 General parameters

The input parameter `Ndat_m` is the 5-element long-integer array specifying the dimensions of the other data arrays. It is similar to the parameter `Ndat` of the single-thread function `GET_MW` (see Section 1.2), but has one additional element, namely:

- `Ndat_m[0]` $= N_{\rm pix}$ is the number of lines-of-sight (pixels).
- `Ndat_m[1]` $= N_z$ is the number of volume elements along a line-of-sight (i.e., the number of points where the plasma and magnetic field parameters are specified), assumed to be the same for all lines-of-sight.
- `Ndat_m[2]` $= N_E$ is the number of elements in the energy array `E`.
- `Ndat_m[3]` $= N_\mu$ is the number of elements in the pitch-angle array `mu`.
- `Ndat_m[4]` $= M$ is the number of parameters used to describe each volume element.

2.3 Source parameters

The input parameter `Parms_m` is the three-dimensional ($M \times N_z \times N_{\rm pix}$) double-precision floating-point array. Each two-dimensional sub-array `Parms_m[*, *, m]` ($0 \le m < N_{\rm pix}$) is equivalent to the parameter `Parms` of the single-thread function `GET_MW` and represents the parameters of a single (mth) line-of-sight, according to the description in Section 1.3.

Note: The emission frequencies (both the number of frequencies and their exact values) are assumed to be the same for all lines-of-sight. For this reason, the corresponding parameters are taken from the first volume element of the first line-of-sight, i.e.:

- $f_0 =$ `Parms_m[15, 0, 0]`,
- $\Delta =$ `Parms_m[16, 0, 0]`,
- $N_f =$ `Parms_m[18, 0, 0]`,

and the corresponding values for other volume elements and other lines-of-sight are ignored.

2.4 Array-defined electron distribution function

The input parameters `E` and `mu` are the same as those in the single-thread function `GET_MW` (see Section 1.4); they specify the electron distribution nodes in the energy and pitch-angle spaces, respectively. They are the same for all lines-of-sight.

The input parameter `f_m` is the four-dimensional ($N_z \times N_E \times N_\mu \times N_{\rm pix}$) double-precision floating-point array. Each three-dimensional sub-array `f_m[*, *, *, m]` ($0 \le m < N_{\rm pix}$) is equivalent to the parameter `f` of the single-thread function `GET_MW` and contains the values of the electron distribution function for a single (mth) line-of-sight, according to the description in Section 1.4.

2.5 Emission parameters

The input/output parameter `RL_m` is the three-dimensional ($7 \times N_f \times N_{\rm pix}$) double-precision floating-point array, where the number of frequencies N_f is given by `Parms_m[18, 0, 0]`. Each two-dimensional sub-array `RL_m[*, *, m]` ($0 \le m < N_{\rm pix}$) is equivalent to the parameter `RL` of the single-thread function `GET_MW` and contains the input/output emission parameters for a single (mth) line-of-sight, according to the description in Section 1.5.

Note: The emission frequencies are assumed to be the same for all lines-of-sight. For this reason, if $f_0 =$ `Parms_m[15, 0, 0]` ≤ 0, the frequencies are taken from the set-up for the first line-of-sight, i.e. $f =$ `RL_m[0, *, 0]`, and copied to those for other lines-of-sight.

2.6 Return value

The return value (`res`) indicated the status of computation:

- `res` $= 0$: no errors.
- `res` < 0: incorrect format of the input parameters (e.g., not enough arguments).

Note: In contrast to the single-thread function `GET_MW`, the function `GET_MW_SLICE` currently does not report the errors (e.g., incorrect electron distribution parameters) that occurred during computation; however, the separate processor threads can display the respective error messages.

References

Cohen, M. H. 1960, *ApJ*, 131, 664

Fleishman, G. D. & Kuznetsov, A. A. 2010, *ApJ*, 721, 1127

Fleishman, G. D. & Kuznetsov, A. A. 2014, *ApJ*, 781, 77

Stallcop, J. R. 1974, *ApJ*, 187, 179