Function GET_MW – single-thread version
built-in abundance tables

Calling syntax:
```
res = call_external(libname, 'GET_MW', Lparms, Rparms, Parms, $
                    T_arr, DEM_arr, DDM_arr, RL)
```

Function parameters:

0. Lparms – 5-element long integer array of dimensions and global (for all voxels) integer parameters (see below).
1. Rparms – 3-element double array of global (for all voxels) real parameters (see below).
2. Parms – array of LOS parameters, 15 × Nz elements, double. Parms[*, i] represents the parameters for *i*th voxel (see below).
3. T_arr – array of temperatures where DEM/DDM are specified, NT elements, double, in K. The temperature grid is assumed to be the same in all voxels, and the same for both DEM and DDM.
4. DEM_arr – array of DEMs, NT × Nz, double, in cm^{-6} K^{-1}. DEM_arr[*, i] represents the DEM for *i*th voxel.
5. DDM_arr – array of DDMs, NT × Nz, double, in cm^{-3} K^{-1}. DDM_arr[*, i] represents the DDM for *i*th voxel.
6. RL – input/output array, 7 × Nf, double. RL[*, i] corresponds to *i*th frequency (see below).

Array of dimensions and global integer parameters Lparms:
Lparms = [Nz, Nf, NT, DEM_key, DDM_key]

0. Nz – number of voxels along LOS;
1. Nf – number of frequencies in the spectrum;
2. NT – number of temperatures in the T_arr array; must be ≥ 2 – otherwise DEM/DEM are ignored;
3. DEM_key – global DEM on/off key.
   a. 0: DEM is enabled: it can be used in all or some voxels, depending on the local DEM on/off keys (see below).
   b. ≠ 0: DEM is disabled for all voxels, regardless of the local DEM on/off keys.
4. DDM_key – global DDM on/off key: same as above, but for DDM.

Array of global real parameters Rparms:
Rparms = [S, f_0, Δf]

0. S – visible source area, in cm^2 (is used unless local values in individual voxels are specified by Parms[14, *], see below).
1. f_0 – starting frequency of the spectrum, in Hz:
   a. is used, only if $f_0 > 0$;
   b. if $f_0 \leq 0$, the frequencies are taken from the RL[0, *] array.
2. Δf – logarithmic frequency step (is used only if $f_0 > 0$).

Array of parameters Parms (for a single voxel, 15 parameters):

0. Parms[0] = Δz – voxel length, in cm.
1. Parms[1] = T_0 – plasma temperature, in K (is used if DEM or DDM are not specified).
2. Parms[2] = n_0 – either electron concentration or total atomic concentration (depending on other parameters), in cm^{-3} (is used if DEM or DDM are not specified).
3. Parms[3] = B – magnetic field strength, in G.
4. Parms[4] = θ – viewing angle, in degrees.
5. Parms[5] = φ – magnetic field azimuthal angle, in degrees.
6. Parms[6] – emission mechanism flag (rounded to the nearest integer):
   a. 0: all emission mechanisms (gyroresonance + free-free + contribution of neutrals) are included;
   b. 1: gyroresonance is off;
   c. 2: free-free is off;
   d. 4: contribution of neutrals is off;
   e. 8: even if DEM and/or DDM are present, the free-free and gyroresonance emissions are computed using the isothermal approximation with the electron concentration and temperature derived from the DDM or DEM (from the DDM, if both are specified).

   Several flags can be combined by usual or bitwise summation: e.g., mechanism flag = 2 + 4 turns off both free-free and contribution of neutrals, etc.
7. Parms[7] = $s_{\max}$ – maximum cyclotron harmonic number.
8. Parms[8] = $n_{\rm p}$ – proton concentration, in cm^{-3} (is used if DEM or DDM are not specified, and the temperature is low).
9. Parms[9] = $n_{\rm HI}$ – neutral hydrogen concentration, in cm^{-3}.
10. Parms[10] = $n_{\rm HeI}$ – neutral helium concentration, in cm^{-3}.
11. Parms[11] – local DEM on/off key:
    a. 0: DEM is used (provided that NT ≥ 2 and DEM is enabled by the global key);
    b. ≠ 0: DEM in this voxel is ignored even if it is specified; T_0 and n_0 are used instead.
12. Parms[12] – local DDM on/off key: same as above, but for DDM.
13. Parms[13] – element abundance model:
    a. 0: coronal (*default*);
    b. 1: photospheric (Caffau);
    c. 2: photospheric (Scott).
14. Parms[14] = S – local source area, in cm^2. These values are used only if $S > 0$ in all voxels along the line-of-sight; in such a case, they override the source area specified by Rparms[0], and the radiation transfer equation takes the form:

$$\frac{\mathrm{d}I}{\mathrm{d}z} = j - \kappa I - \frac{I}{S}\frac{\mathrm{d}S}{\mathrm{d}z},$$

where z is the coordinate along the line-of-sight, I is the intensity of the given mode, j is the emissivity, and κ is the absorption coefficient; the visible source area becomes equal to the last value of S. If any one of S values is zero or negative, the source area is assumed to be constant and is specified by Rparms[0], and the refraction term in the radiation transfer equation (the last term in the above formula) is ignored.

Input/output array RL:

0. First row (RL[0, *]) – emission frequencies, in GHz. On input, this array is used if f_0 = Rparms[1] ≤ 0; otherwise, the frequencies are computed using the f_0 and Δf parameters: $f_1 = f_0 10^{\Delta f}$, $f_2 = f_1 10^{\Delta f}$, etc. On output, this array contains the computed or pre-defined emission frequencies.

Other rows – emission intensities, as observed from the Earth, in sfu:

1. RL[1, *] – left polarization, weak mode coupling;
2. RL[2, *] – right polarization, weak mode coupling;
3. RL[3, *] – left polarization, strong mode coupling;
4. RL[4, *] – right polarization, strong mode coupling;
5. RL[5, *] – left polarization, exact mode coupling.
6. RL[6, *] – right polarization, exact mode coupling.

On input, these arrays specify the emission intensities at the start of the line-of-sight; on output, they contain the emission intensities at the end of the line-of-sight.

Return value: currently, -1 if the input was incorrect (incorrect number of parameters); 0 otherwise.

Function GET_MW – single-thread version

user-defined abundance tables

Calling syntax:

```
res = call_external(libname, 'GET_MW', Lparms, Rparms, Parms, $
                    T_arr, DEM_arr, DDM_arr, $
                    fzeta_arr, Tzeta_arr, zeta_arr, RL)
```

Function parameters:

0. Lparms – 8-element long integer array of dimensions and global (for all voxels) integer parameters (see below).

1-5. Rparms, Parms, T_arr, DEM_arr, DDM_arr – same as in the version with built-in abundance tables; for Parms, see a note below.

6. fzeta_arr – array of frequencies where the ζ-function is specified, Nf_zeta elements, double, in Hz. The frequency grid is assumed to be the same in all voxels and for all supplied abundance sets.

7. Tzeta_arr – array of temperatures where the ζ-function is specified, NT_zeta elements, double, in K. The temperature grid is assumed to be the same in all voxels and for all supplied abundance sets.

8. zeta_arr – array of ζ-function values, Nf_zeta × NT_zeta × N_zeta elements, double. This array is the same for all voxels. zeta_arr[*, *, m] represents the 2D ζ-function table for *m*th abundance set.

9. RL – same as in the version with built-in abundance tables.

Array of dimensions and global integer parameters Lparms:

Lparms = [Nz, Nf, NT, DEM_key, DDM_key, Nf_zeta, NT_zeta, N_zeta]

0-4. Nz, Nf, NT, DEM_key, DDM_key – same as in the version with built-in abundance tables.

5. Nf_zeta – number of frequencies where the ζ-function is specified.

6. NT_zeta – number of temperatures where the ζ-function is specified.

7. N_zeta – number of supplied 2D ζ-function tables (abundance sets).

Array of parameters Parms (for a single voxel):

Most of parameters are the same as in the version with built-in abundance tables, except Parms[13] that specifies the element abundance model index *m*, so that the 2D ζ-function table given by zeta_arr[*, *, m] is used in this voxel.

Return value: currently, -1 if the input was incorrect (incorrect number of parameters); 0 otherwise.

Function GET_MW_SLICE – multi-thread version
built-in abundance tables

Calling syntax:
res = call_external(libname, 'GET_MW_SLICE', Lparms_M, Rparms_M, $
Parms_M, T_arr, DEM_arr_M, DDM_arrM, RL_M)

Function parameters:

0. Lparms_M – 6-element long integer array of dimensions and global (for all voxels and all LOSs) integer parameters (see below).
1. Rparms_M – array of real parameters common for all voxels within each LOS, 3 × Npix, double (see below).
2. Parms_M – array of voxel parameters, 15 × Nz × Npix elements, double (see below).
3. T_arr – array of temperatures where DEM/DDM are specified, NT elements, double, in K. This parameter is the same as in the GET_MW function: the temperature grid is assumed to be the same in all voxels and all LOSs, and the same for both DEM and DDM.
4. DEM_arr_M – array of DEMs, NT × Nz × Npix, double, in cm^{-6} K^{-1} (see below).
5. DDM_arr_M – array of DDMs, NT × Nz × Npix, double, in cm^{-3} K^{-1} (see below).
6. RL_M – input/output array, 7 × Nf × Npix, double (see below).

Array of dimensions and global integer parameters Lparms_M:
Lparms_M = [Npix, Nz, Nf, NT, DEM_key, DDM_key]

0. Npix – number of LOSs.

Other elements (1^{st} to 5^{th}) are the same as the 0^{th} to 4^{th} elements of the Lparms array in the GET_MW function. In particular:

a. all LOSs have the same number of voxels Nz;
b. the number of frequencies Nf is the same for all LOSs;
c. the global DEM and DDM on/off keys are related to all voxels within all LOSs.

Other parameters: sub-arrays Rparms_M[*, i], Parms_M[*, *, i], DEM_arr_M[*, *, i], DDM_arr_M[*, *, i] and RL_M[*, *, i] correspond respectively to the parameters Rparms, Parms, DEM_arr, DDM_arr and RL of the single-thread GET_MW function, for ith LOS.

Return value: currently, -1 if the input was incorrect (incorrect number of parameters); 0 otherwise.

Function GET_MW_SLICE – multi-thread version for user-defined abundance tables: not implemented yet.

Function GET_GX_MW – single-thread version

Calling syntax:

```
res = call_external(libname, 'GET_GX_MW',          $
                    Lparms, Rparms, Parms,          $
                    Qrun, Lrun, logTDEM,            $
                    DEM_run, DDM_run, RL)
```

Function parameters:

0. Lparms – 7-element long integer array of dimensions and global keys.
1. Rparms – 3-element double array of global (for all voxels) real parameters.
2. Parms – array of LOS parameters, 15 × Nz elements, double. Parms[*, i] represents the parameters for *i*th voxel.
3. Qrun – the EBTEL Q grid, NQ × NL elements, float, in erg cm^{-3} s^{-1}.
4. Lrun – the EBTEL L grid, NQ × NL elements, float, in cm.
5. logTDEM – the EBTEL temperature grid ($\log_{10}T$, where the temperature T is in K), NT elements, float.
6. DEM_run – the EBTEL DEM table, NT × NQ × NL elements, float, in cm^{-6} K^{-1}.
7. DDM_run – the EBTEL DDM table, NT × NQ × NL elements, float, in cm^{-3} K^{-1}.
8. RL – input/output array, 7 × Nf elements, double.

Array of dimensions and global integer parameters Lparms:
Lparms = [Nz, Nf, NQ, NL, NT, DEM_key, DDM_key]

0. Nz – number of voxels along LOS.
1. Nf – number of frequencies in the spectrum.
2. NQ – size of the EBTEL Q grid.
3. NL – size of the EBTEL L grid.
4. NT – size of the EBTEL temperature grid.
5. DEM_key – global DEM on/off key (same as in the GET_MW function).
6. DDM_key – global DDM on/off key (same as in the GET_MW function).

Array of global real parameters Rparms:
This array is the same as in the GET_MW function.

Array of parameters Parms (for a single voxel, 15 parameters):
This array is the almost the same as in the GET_MW function, with the exception of two parameters:

11. Parms[11] = Q – the EBTEL heating rate, in erg cm^{-3} s^{-1}.
12. Parms[12] = L – the EBTEL loop length, in cm.

Note that the DEM and/or DDM in a given voxel are computed and used if $Q > 0$, $L > 0$, and the (Q, L) pair is located within the EBTEL table. Otherwise, the plasma temperature T_0 = Parms[1] and density n_0 = Parms[2] are used to compute the emission.

Input/output array RL:
This array is the same as in the GET_MW function.

Return value: currently, -1 if the input was incorrect (incorrect number of parameters); 0 otherwise.

Function GET_GX_MW_SLICE – single-thread version

Calling syntax:

```
res = call_external(libname, 'GET_GX_MW_SLICE',         $
                    Lparms_M, Rparms_M, Parms_M,         $
                    Qrun, Lrun, logTDEM,                 $
                    DEM_run, DDM_run, RL_M)
```

Function parameters:

0. Lparms_M – 8-element long integer array of dimensions and global keys. Lparms_M = [Nz, Nf, NQ, NL, NT, DEM_key, DDM_key], where Npix is the number of LOSs, and other elements are the same as in the single-thread function GET_GX_MW (they are assumed to be the same for all LOSs).
1. Rparms – array of real parameters common for all voxels within each LOS, 3 × Npix elements, double. Rparms_M[*, i] represents the parameter Rparms of the single-thread function GET_GX_MW for *i*th LOS.
2. Parms – array of voxel parameters, 15 × Nz × Npix elements, double. Parms_M[*, *, i] represents the parameter Parms of the single-thread function GET_GX_MW for *i*th LOS.
3. Qrun is the same as in the single-thread function GET_GX_MW (this grid is assumed to be the same for all LOSs).
4. Lrun is the same as in the single-thread function GET_GX_MW (this grid is assumed to be the same for all LOSs).
5. logTDEM is the same as in the single-thread function GET_GX_MW (this grid is assumed to be the same for all LOSs).
6. DEM_run is the same as in the single-thread function GET_GX_MW (this table is assumed to be the same for all LOSs).
7. DDM_run is the same as in the single-thread function GET_GX_MW (this table is assumed to be the same for all LOSs).
8. RL_M – input/output array, 7 × Nf × Npix elements, double. RL_M[*, *, i] represents the parameter RL of the single-thread function GET_GX_MW for *i*th LOS.

Return value: currently, -1 if the input was incorrect (incorrect number of parameters); 0 otherwise.