

CBZ/MulticellBurner の最低限の使用マニュアル¹

2025/10/31 千葉 豪

PWR 燃料集合体の燃焼計算を想定したサンプル入力 main.vera2o.cxx を基に、MulticellBurner モジュールの使用法の解説を行う。基本的な考え方は Burner と同様であるが、複数の異なる燃料棒が配置されること、通常の燃料棒では燃料棒あたり 1 つの燃焼領域を設定する一方で可燃性毒物入りの燃料棒では径方向に分割した複数の燃焼領域を設定することなどが異なる。サンプル入力のヘッダ部分を以下に示す。

Listing 1: ヘッダの情報

```
1 // * VERA depletion benchmark *
2 //
3 // Problem 2O
4 //
5 //-----+
6 //          o o
7 //          o o
8 //          G o o
9 //          X o o o
10 //          o o o o o
11 //          X o o X o o
12 //          o o G o o o o
13 //          o o o o o o o
14 //          X o o X o o X o o
15 //
16 // X : guide tube
17 // o : normal fuel
18 // G : Gd-bearing fuel
19 //
20 //
21 // (Medium-ID map)
22 //
23 //          52
24 //          50 51
25 //          40 48 49
26 //          X 37 38 39
27 //          32 33 34 35 36
28 //          X 28 29 X 30 31
29 //          14 15 16 24 25 26 27
30 //          6 7 8 9 10 11 12 13
31 //          X 0 1 X 2 3 X 4 5
32 //
33 // X : guide tube
34 // o : normal fuel
35 // G : Gd-bearing fuel
36 //
37 // "0" is the center pin of the multicell system.
38 // Medium 0-52 : Fuel
39 //          53 : Cladding
40 //          54 : Moderator
```

ここで対象としている燃料集合体には、通常の燃料棒、可燃性毒物である Gd 入り燃料棒、案内管が規則的に配置される。個々の燃料棒はそれぞれ異なる燃焼媒質として扱われ、さらに Gd 入り燃料棒については径方向に分割した複数の燃焼媒質として扱うことが可能である。Medium ID map ではそれぞれの位置の燃料棒に割り当てられる燃焼媒質の index が示されている。この例では Gd 入り燃料棒は 8 つの燃焼領域として扱うため、Gd 入り燃料棒の位置には燃焼媒質の一番最初の index が示されている。結果的にこの例では 53 の燃焼媒質が定義されている。それ以降の媒質の index は非燃焼媒質に対して用いられ、この例では 53 が「Cladding」、54 が「Moderator」に割り当てられている。

1 入力情報の詳細

ヘッダ部分に続く入力部分を以下に示す。

Listing 2: 計算条件の指定

```
1 // +++ USER INPUT +++++
2
3 bool pc_cal = true; // Default is [true]
4
```

¹Document/CBG_Manual/MBurner

```

5 // (When the following methods are activated, burnup calculations are NOT carried out.)
6 bool geom_plot = false; // If [true], plot data for mesh setting is written on a file [plotdata]
7 bool clip_plot = false; // If [true], plot data for material setting is written on a file [clipdata]
8 bool show_medium_id = false; // If [true], medium indices are printed on the screen.

```

boolean 変数 `pc_cal` では、燃焼計算における predictor-corrector 法（PC 法）適用の有無を指定する。体系が Gd 入り燃料棒を含む場合には PC 法の適用が必須であるが、2 倍の計算時間を要することになるので、試算では適用しないことで計算時間の短縮を図るとよいであろう。なお、Gd 入り燃料棒に対しては AOWPC 法が適用される。

`geom_plot` では、gnuplot で描画可能なドットファイル `plot_data` が生成される。このファイルは空間メッシュ分割情報を格納しており、gnuplot で `plot 'plot_data' w d` とすると描画結果が画面に出力される。`clip_plot` では CLIP で描画可能なファイル `clip_data` が生成される。CLIP に使い方は別稿を参照のこと。`show_medium_id` では、燃料集合体に配置される燃料棒に割り当てられる燃焼媒質の index が画面に出力される。この機能は、Gd 入り燃料棒の径方向分割数を変更する際に使用されることを想定している。なお、これらのいずれかを `true` にした場合、これらの処理が行われ次第、プログラムは終了する。

引き続き入力部分を以下に示す。

Listing 3: 計算体系の指定

```

1 // ... geometry
2
3 int array = 17;
4 int pin_map[]={ // 1:Gd-bearing rod, 2:Guide tube
5     2,0,0,2,0,0,2,0,0,
6     0,0,0,0,0,0,0,0,
7     0,0,1,0,0,0,0,
8     2,0,0,2,0,0,
9     0,0,0,0,0,
10    2,0,0,0,
11    1,0,0,
12    0,0,
13    0,
14 };
15
16 real pin_pitch = 1.26; // [cm]
17
18 real assembly_pitch = pin_pitch * array;
19
20 real radius_pellet = 0.4096; // [cm]
21 real radius_cladding = 0.475; // [cm]
22 real radius_gt_in = 0.561; // [cm] ... inner radius of guide tube
23 real radius_gt_out = 0.602; // [cm] ... outer radius of guide tube
24
25 int div_pellet_gd = 8;
26 // The number of radial divisions of Gd-bearing rod.
27 // If this is changed from [8], the material index assignment is also changed.

```

int 型変数 `array` では、PWR 燃料集合体の縦（もしくは横）の配列数を定義する。ここでは 17 としているので 17x17 の燃料集合体が想定されていることになる。この例では 1/8 対称性を持つ集合体（1/8 集合体）を想定しているが、フル集合体などの計算も可能である。

16 から 23 行目では種々の幾何形状情報を定義する。なお、このサンプルでは被覆管とペレットの間のギャップ領域を定義できないため、ギャップ領域が存在する場合には、ギャップ領域を真空とみなした上で被覆管とギャップ領域を混合させたものとして計算を行う。また、この例では `assembly_pitch` を `pin_pitch` の配列数倍としているため、集合体間のギャップは存在しないことになる。

`div_pellet_gd` では Gd 入り燃料の径方向の燃焼領域分割数を指定する。この例では 8 としているが、これを変更すると燃焼媒質の総数や燃焼媒質 index の割り当てが変わることになるので、慣れるまでこの数値を変更することは推奨しない。

引き続き入力部分を以下に示す。

Listing 4: 空間離散化条件の指定

```

1 #if 0
2 // ... default setting
3 int div_pellet = 4; // [4] ... number of radial divisions of fuel pellet
4 int div_gt_in = 4; // [4] ... number of radial Div. of coolant region inside of guide tube
5 int div_mod_ring = 3; // [3] ... number of radial Div. of coolant regions outside of cladding/GT
6 int div_azim = 8; // [8] (per 2PI) ... number of azimuthal divisions of unit cell

```

```

7 #endif
8
9 #if 1
10 // ... simplified setting
11 int div_pellet = 1;
12 int div_gt_in = 1;
13 int div_mod_ring = 1;
14 int div_azim = 1;
15 #endif

```

ここでは空間離散化（メッシュ分割）条件を指定する。通常は3から6行目の default setting を推奨するが、試計算の段階では計算時間を大幅に短縮できる simplified setting を推奨する。これらの違いについては上で説明した boolean 変数の `geom_plot` を `true` にして可視化することで理解できるであろう。

引き続き入力部分を以下に示す。

Listing 5: 核データライブラリ、燃焼チェーンの指定

```

1 bn.SetLibrary(cbglibdir,"jendl-4.0"); // ... 107-group library
2
3 //bn.SetLibrary_361g(cbglibdir,"jendl-4.0");
4 //bn.SetLibrary_361g(cbglibdir,"jendl-5");
5 //bn.SetLibrary_361g(cbglibdir,"endlf-8");
6 //bn.SetLibrary_361g(cbglibdir,"endlf-7.1");
7 //bn.SetLibrary_361g(cbglibdir,"jeff-3.2");
8 //bn.SetLibrary_361g(cbglibdir,"jeff-3.3");
9
10
11 bu.GetBurnupChain().ReadDecayConstantFromFile(cbglibdir,"srac-org");
12 bu.ReadAtomicMassDataFromFile(cbglibdir,"jeff311");
13 bu.ReadReactionEnergyFromFile("../","sractype");
14 bu.GetBurnupChain().Set21HeavyMetalChain();
15
16 // ... Full(1400)-FP chain
17 //bu.GetBurnupChain().ReadFPYieldDataFromFile(cbglibdir,"j2011t");
18 //bu.GetBurnupChain().OverWritingChainData(cbglibdir,"j2011t");
19
20 // ... 138-FP chain
21 bu.GetBurnupChain().ReadFPYieldDataFromFile(cbglibdir,"fp138.j2011t");
22 bu.GetBurnupChain().OverWritingChainData(cbglibdir,"fp138.j2011t");
23
24 // ... 193-FP chain
25 //bu.GetBurnupChain().ReadFPYieldDataFromFile(cbglibdir,"fp193.j2011t.20201117");
26 //bu.GetBurnupChain().OverWritingChainData(cbglibdir,"fp193.j2011t.20201117");

```

使用する核データライブラリは1から8行目の形式で指定する。`SetLibrary` では107群、`SetLibrary_361g` では361群のライブラリが使用される。

燃焼チェーンについては、アクチニドは14行目の `Set21HeavyMetalChain` 等により指定する。この例で用いているチェーンには Cm-247 や Th 同位体などは含まれないので、それらを計算した場合は別のメソッドを用いる必要がある。FP については16から26行目の形式で指定する。`ReadFPYieldDataFromFile` メソッドではFPの核分裂収率を、`OverWritingChainData` メソッドではFPの崩壊データを定義する。17、18行目は JENDL/FPY-2011、FPD-2011 で定義されている1,400のFP全てを考慮するものであるが、膨大な計算容量を必要とする可能性があるため、計算を実行したときのメモリを確認することを強く推奨する。基本的にこのレベルの詳細チェーンを用いる場合には Burner を用いたピンセル燃焼計算を推奨する。21、21行目は反応度に影響を与える138のFPを選別したチェーンであり、燃料集合体の燃焼計算ではこの使用を推奨する。

引き続き入力部分を以下に示す。

Listing 6: 媒質情報の指定

```

1 // ++++++
2 // Medium data
3
4 // (uo2 data : 3.1wt% enrichment)
5 int num_uo2=5;
6 int mat_uo2[]={922340,922350,922360,922380,80160};
7 real den_uo2[]={6.11846e-6, 7.18132e-4, 3.29861e-6, 2.21546e-2, 4.57642e-2};
8
9 // (Gd-bearing rod)
10 int num_uo2_gd=12;
11 int mat_uo2_gd[]={
12     922340,922350,922360,922380,
13     641520,641540,641550,641560,641570,641580,641600,
14     80160};
15 real den_uo2_gd[]={

```

```

16      3.18096e-6, 3.90500e-4, 1.79300e-6, 2.10299e-2,
17      3.35960e-6, 3.66190e-5, 2.48606e-4, 3.43849e-4, 2.62884e-4, 4.17255e-4, 3.67198e-4,
18      4.53705e-2,};
19
20  real temp_fuel=900.; // [K]
21
22  for(int i=0;i<num_burnmed;i++){
23      if ( bn.IsGdFuel(i) ) { // Gd-bearing fuel
24          bn.PutFuelDataMB(num_uo2_gd, mat_uo2_gd, den_uo2_gd, temp_fuel, i);
25      }else{ // normal fuel
26          bn.PutFuelDataMB(num_uo2, mat_uo2, den_uo2, temp_fuel, i);
27      };
28  };
29
30  // (clad)
31  real vol_org=0.475*0.475-0.4185*0.4185; // gap region is smeard to cladding region
32  real vol_mod=0.475*0.475-0.4096*0.4096;
33  real fact_clad=vol_org/vol_mod;
34  int mat2[]={400000,260000,240000,500000,720000};
35  real den2[]={4.25394e-2*fact_clad, 1.48556e-4*fact_clad, 7.59773e-5*fact_clad,
36      4.19397e-4*fact_clad, 2.21337e-6*fact_clad};
37  bn.PutCladData(5,mat2,den2,600.);
38
39  // (water)
40  int mat3[]={10010,80160,50100,50110};
41  real den3[]={4.67505e-2, 2.33753e-2, 1.00874e-5, 4.06030e-5};
42  bn.PutModeratorData(4,mat3,den3,591.);

```

4 から 7 行目では通常の燃料棒、9 から 18 行目では Gd 入り燃料棒の組成を指定する。また、30 から 36 行目では被覆管の組成を指定しているが、燃料ペレットと被覆管の間のギャップ領域を考慮できないサンプル入力であるため、被覆管をギャップ領域にスミアして扱う必要がある。そのため、与えられた数密度を希釈した数値を与えている。なお、PutCladData、PutModeratorData メソッドの 4 つ目の引数では温度 [K] を与える。

引き続き入力部分を以下に示す。

Listing 7: 照射条件の指定

```

1  // ++++++
2  // Burnup condition
3
4  int med_normalize = -1;
5  // When [-1], neutron flux is normalized by whole assembly power per unit axial length
6
7  int div = 1; // the number of divisions for each step in the array [burn-time]
8  //bn.PutBurnStep(2*div); // [8] by 9.6 GWD/t, [24] by 35.2GWD/t
9  bn.PutBurnStep(16*div); // [8] by 9.6 GWD/t, [24] by 35.2GWD/t
10 real burn_time[]={
11     0.04, 0.36, 1.2,
12     1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6,
13     1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6,
14     1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6,
15     1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6, 1.6,
16 }; // GWD/t
17
18 bn.PutPowerDensity( 40. , "MW.t" ); // [MW/t] (for whole assembly)
19 bn.PutBurnTime( burn_time , true , false , div );
20 // ... second argument : [true/false] = [GWd.t/day]
21 // ... third argument : [true/false] = [accumulate/width]

```

3 行目の med_normalize では出力を規格化する燃焼媒質の index を指定する。これを-1とした場合には計算体系全体の出力で規格化を行う。また、これを-2以下とした場合には、媒質 index が 0 から-med_normalize-1 までで規格化を行うこともできる。例えば med_normalize を-10とした場合には、index が 0 から 9 までの媒質の出力で規格化を行う。

5 行目以降は燃焼ステップ数とステップの幅を指定する。ある燃焼ステップの設定に対して、各ステップをさらに詳細化した計算を行う場合を想定し、int 型変数 div を定義できる。通常はこれを 1 とすればよいが、2 以上にするによって、上記のような計算が可能となる。

PutPowerDensity では出力履歴を指定する。この例では MW/t 単位で与えられているが、2 つ目の引数を「W_cm」とする(か省略する)ことで、W/cm 単位で与えることも可能である。この場合、燃料棒 1 本あたりではなく、計算対象全体あたりの単位高さあたりの出力となる。

2 計算結果として画面に出力される情報

出力の冒頭に、計算に使用した核データ、燃焼チェーンの情報が以下のように出力される、計算結果の出力ファイルを保存したときに有用な情報となる。

Listing 8: 計算に使用した核データ、燃焼チェーンの情報

```
1 # Reading N-ENERGY data at the directory [../../CBGLIB/j4.107g.iwt4/]
2 # Reading N-ENERGY data at the directory [../../CBGLIB/j4.107g.iwt4/]
3 # Reading FP yield data file      : ../../CBGLIB.BURN/CBG.FY/fp138.j2011t
4 # Overwriting with chain data file : ../../CBGLIB.BURN/CBG.Chain/fp138.j2011t
```

いくつかの warning メッセージに続いて、以下のように代表燃料の数密度と温度情報が出力される。

Listing 9: 燃料媒質の情報

```
1 # Fuel number density information
2 # (number density and temperature)
3 # (representative medium in the case of [MulticellBurner]).
4 #
5 #      U234 : 6.118460e-06 9.000000e+02
6 #      U235 : 7.181320e-04 9.000000e+02
7 #      U236 : 3.298610e-06 9.000000e+02
8 #      U238 : 2.215460e-02 9.000000e+02
9 #      O016 : 4.576420e-02 9.000000e+02
```

さらに、燃料棒の各位置毎の Dancoff 係数が外面に表示される。案内管に近い位置や、集合体間にギャップがある場合には集合体外周位置で、大きな値となる。

Listing 10: Dancoff 係数の表示

```
1 # Dancoff factor calculated by [AutoDancoffCalculation]
2 #
3 #      Representative energy group : 55
4 #      Corresponding energy       : 1.000000e+01 eV
5 #
6 # (GT)  0.6942  0.6951  (GT)  0.6932  0.6944  (GT)  0.6933  0.6424
7 #      0.6663  0.6654  0.6944  0.6657  0.6663  0.6946  0.6626  0.6448
8 #      0.6650  0.6943  (Gd)  0.6659  0.6950  0.6640  0.6453
9 #      (GT)  0.6968  0.6938  (GT)  0.6948  0.6429
10 #      0.6853  0.7149  0.7132  0.6639  0.6444
11 #      (GT)  0.6944  0.6452  0.6425
12 #      (Gd)  0.6457  0.6431
13 #      0.6425  0.6417
14 #      0.6441
```

さらに、燃焼媒質毎の燃料重量が表示される。

Listing 11: 初期燃料重量の表示

```
1 # Initial heavy metal weight [g/cm]
2 #
3 #      Target media : 1.56808e+02
4 #
5 #      Medium 0 : 2.38235e+00
6 #      Medium 1 : 2.38333e+00
7 #      Medium 2 : 2.38321e+00
8 #      Medium 3 : 2.38358e+00
9 #      Medium 4 : 2.38338e+00
10 #      Medium 5 : 2.38253e+00
11 #
12 #
13 #      Medium 49 : 4.76619e+00
14 #      Medium 50 : 2.38240e+00
15 #      Medium 51 : 4.76593e+00
16 #      Medium 52 : 2.38366e+00
17 #      Total    : 1.56808e+02
```

さらに、照射条件が表示される。

Listing 12: 照射条件の表示

```

1 # BURNUP CONDITION SETTING
2 #
3 # The number of burnup steps : 16
4 #
5 # In the case of [MulticellBurner], the fuel burnup is calculated for the target.
6 #
7 # Step    Day          BU          Day-width    BU-width    Power density
8 #         [GWD/t]      [GWD/t]      [GWD/t]      [W.cm]
9 #
10 # 0      0.000e+00    0.000e+00    1.000e+00    4.000e-02    6.272e+03
11 # 1      1.000e+00    4.000e-02    9.000e+00    3.600e-01    6.272e+03
12 # 2      1.000e+01    4.000e-01    3.000e+01    1.200e+00    6.272e+03
13 # 3      4.000e+01    1.600e+00    4.000e+01    1.600e+00    6.272e+03
14 # 4      8.000e+01    3.200e+00    4.000e+01    1.600e+00    6.272e+03
15 # 5      1.200e+02    4.800e+00    4.000e+01    1.600e+00    6.272e+03
16 # 6      1.600e+02    6.400e+00    4.000e+01    1.600e+00    6.272e+03
17 # 7      2.000e+02    8.000e+00    4.000e+01    1.600e+00    6.272e+03
18 # 8      2.400e+02    9.600e+00    4.000e+01    1.600e+00    6.272e+03
19 # 9      2.800e+02    1.120e+01    4.000e+01    1.600e+00    6.272e+03
20 # 10     3.200e+02    1.280e+01    4.000e+01    1.600e+00    6.272e+03
21 # 11     3.600e+02    1.440e+01    4.000e+01    1.600e+00    6.272e+03
22 # 12     4.000e+02    1.600e+01    4.000e+01    1.600e+00    6.272e+03
23 # 13     4.400e+02    1.760e+01    4.000e+01    1.600e+00    6.272e+03
24 # 14     4.800e+02    1.920e+01    4.000e+01    1.600e+00    6.272e+03
25 # 15     5.200e+02    2.080e+01    4.000e+01    1.600e+00    6.272e+03
26 #         5.600e+02    2.240e+01
27 #
28 # Accumulated power/flux : 3.513e+06

```

3 計算結果の取り出し

全ての燃焼媒質における燃焼度毎の数密度などの情報は `ShowNumberDensityHistory` メソッドにより取り出すことができる。以下の例は、特定の核種の情報を取り出すためのものである。4つ目の引数は取り出したい量を指定するためのキーワードであり、`Burner` モジュールと同一である。また、「その2」で示されているように、結果を外部ファイルに出力したい場合には6番目の引数にファイル名を指定すればよく、結果の表示桁数を増やしたい場合には7番目の引数に整数を与えればよい（デフォルトは4）。なお、5番目の引数はデータの総和のみを出力させるオプションであり、通常は `false` とすればよい。

Listing 13: 燃焼媒質における数密度等の取り出し例（その1）

```

1 int prt_nuc=2;
2 string prt_nuc_nam[]={ "Gd155", "Gd157" };
3 bn.ShowNumberDensityHistory( prt_nuc , prt_nuc_nam , bu , "nd-per-vol" );

```

Listing 14: 燃焼媒質における数密度等の取り出し例（その2）

```

1 int prt_nuc=2;
2 string prt_nuc_nam[]={ "Gd155", "Gd157" };
3 bn.ShowNumberDensityHistory( prt_nuc , prt_nuc_nam , bu , "nd-per-vol" , false , "file" , 5 );

```

燃焼媒質に含まれる全ての核種の情報について取得したいときには、以下に示すように、これまでの例における最初の2つの引数を省略すればよい。

Listing 15: 燃焼媒質における数密度等の取り出し例（その3）

```

1 bn.ShowNumberDensityHistory( bu , "nd-per-vol" , false , "file" , 5 );

```

各燃焼度点における燃焼媒質の中性子スペクトルは以下の例にある `WriteFileFuelFlux` メソッドにより外部ファイルに出力できる。引数は、1つ目が外部ファイルを出力するディレクトリ、2つ目がファイル名、3つ目が数値の表示桁に対応する。

Listing 16: 燃焼媒質における中性子スペクトルの取り出し例

```

1 int output_digit = 5;
2 bn.WriteFileFuelFlux( "./" , "fuel-flux" , output_digit );

```