EEM-RTMデータ作成ライブラリ

取扱説明書(Java版)

株式会社EEM　2011年12月

1. 概要

本ライブラリーはEEM-RTMの入力データを作成するものです。
対応言語はJavaです。
必要な関数を呼ぶことによってEEM-RTMデータが出力されます。
対応するEEM-RTMのバージョンは2.1以降です。

2. 使用法

下記の関数仕様に従ってソースコードを作成し、コンパイルして実行します。
ソースコードのファイル名をsample_rtm.javaとすると、コマンドラインで以下の操作を行います。
> javac sample_rtm.java RtmData.java　　コンパイル
> java sample_rtm　　実行、EEM-RTMデータ出力
出力されたEEM-RTMデータはEEM-RTMで開くことができます。EEM-RTMで小さい修正を行うことも可能です。
なお、上記はRtmData.javaファイルが同じフォルダにあることを仮定しています。違うフォルダにあるときは必要な変更を行って下さい。

3. 関数仕様

本ライブラリ(RtmData)の各関数の仕様は以下の通りです。
実数の引数はすべて倍精度(double)で単位はMKSA、度です。
最初に(1)でインスタンスを作成し、最後に(14)でデータを保存します。その間の関数の呼び出し順は任意ですが、下記の順を推奨します。
同じ関数名で引数の異なるものがありますので注意して下さい。

(1)コンストラクタ

public RtmData()　　インスタンスを作成します。
※最初に一度呼び出すことが必要です。

(2)タイトル

public void setTitle(String title)　　タイトル(オプション)

(3)周波数

```
public void setFreq(double freq)            周波数[Hz]
```

(4)物性値

複数使用可能です。物性値番号は入力した順に2,3,...になります。(1:PECは不要です)

```
public void addMaterial(double epsr,        比誘電率
                        double esgm)        導電率[S/m]
```
※厚さが十分大きく透過波が無視できる場合に使用します。

```
public void addMaterial(double epsr,        比誘電率
                        double esgm,        導電率[S/m]
                        double thick)       厚さ[m]
```

(5)多角柱

断面が多角形で底面と上面が水平な多角柱を指定します。

(5.1)一般の多角柱

```
public void addPillar(int n,                    頂点の数(2以上30以下)
                      const double x[],         頂点のX座標[m]
                      const double y[],         頂点のY座標[m]
                      double z1,                底面の高さ[m]
                      double z2,                上面の高さ[m]
                      int m)                    物性値番号(1以上)
```

(5.2)長方形の柱

```
public void addPillarRect(double x1, double y1, 左下のX,Y座標[m]
                          double x2, double y2; 右上のX,Y座標[m]
                          double z1,            底面の高さ[m]
                          double z2,            上面の高さ[m]
                          int m)                物性値番号(1以上)
```

(5.3)鉛直の壁

```
public void addPillarWall(double x1, double y1, 始点のX,Y座標[m]
                          double x2, double y2; 終点のX,Y座標[m]
                          double z1,            底面の高さ[m]
                          double z2,            上面の高さ[m]
                          int m)                物性値番号(1以上)
```

(6)地面

四角形を指定します。四頂点のX,Y,Z座標は任意です。

(6.1)一般の地面

```
public void addPlane(double x[4],               四頂点のX座標[m]
                     double y[4],               四頂点のY座標[m]
                     double z[4],               四頂点のZ座標[m]
                     int m)                     物性値番号(1以上)
```

(6.2)長方形の水平面

```
public void addPlaneRect(double x1, double y1,   左下のX,Y座標[m]
                         double x2, double y2;   右上のX,Y座標[m]
                         double z0,              高さ[m]
                         int m)                  物性値番号(1以上)
```

(7)アンテナ特性

アンテナ特性を指定します。以後変更されるまで、送信点・観測点・観測線・観測面に適用されます。
初期値は無指向性、垂直偏波です。

(7.1)無指向性

```
public void setAntennaIso(int pol)          偏波(1:垂直,2:水平,3:右旋円偏波,4:左旋円偏波)
```

(7.2)ダイポール

```
public void setAntennaDipole(double theta,  軸方向の θ[度]
                             double phi,    軸方向の φ[度]
                             double wbeam,  3dBビーム幅[度]
                             int pol)       偏波(1:垂直,2:水平,3:右旋円偏波,4:左旋円偏波)
```

(7.3)ビーム

```
public void setAntennaBeam(double theta,    ビーム中心軸の θ[度]
                           double phi,      ビーム中心軸の φ[度]
                           double wtheta,   θ 方向の3dB幅[度]
                           double wphi,     φ 方向の3dB幅[度]
                           int pol)         偏波(1:垂直,2:水平,3:右旋円偏波,4:左旋円偏波)
```

(7.4)ファイル指定

```
public void setAntennaFile(double theta,    回転中心軸の θ[度]
                           double phi,      回転中心軸の φ[度]
                           double rot,      回転角[度]
                           int raw,         0:正規化する、1:正規化しない
                           String filename) アンテナ指向性ファイル名
```

(8)送信点

```
public void setTx(double x,         X座標[m]
                  double y,         Y座標[m]
                  double z,         Z座標[m]
                  double power,     送信電力[W]
                  double phase)     送信位相[度]
```

(9)観測点

```
public void setRx0(double x,        X座標[m]
                   double y,        Y座標[m]
                   double z)        Z座標[m]
```

(10)観測線

```
public void setRx1(double x[2],      始点と終点のX座標[m]
                   double y[2],      始点と終点のY座標[m]
                   double z[2],      始点と終点のZ座標[m]
                   int div)          線分の分割数
```

(11)観測面

```
public void setRx2(double x[4],      4頂点のX座標[m]
                   double y[4],      4頂点のY座標[m]
                   double z[4],      4頂点のZ座標[m]
                   int div12,        頂点1-2方向の分割数
                   int div14)        頂点1-4方向の分割数
```

(12)計算条件

```
public void setSolver(int maxref,          最大反射回数
                      int raydiv,          緯度方向分割数
                      int ndiffr,          回折(0:なし、1:回折のみ、2:回折+反射まで)
                      boolean trans,       透過波を計算するか
                      boolean beam,        レイ放射方向を絞るか
                      boolean pathlog,     path.logを出力するか
                      boolean adiffr,      回折波を近似計算するか
                      double att)          減衰定数[dB/m]
```

(13)その他データ

```
public void setMisc(int size,              画面上の大きさ(ピクセル)、縦・横の大きい方
                    int margin)            画面上の周囲の余白(ピクセル)
```

※既定値はsize=400,margin=0

(14)ファイル出力

```
public void save(String filename)          データをファイル"filename"に出力します。
```

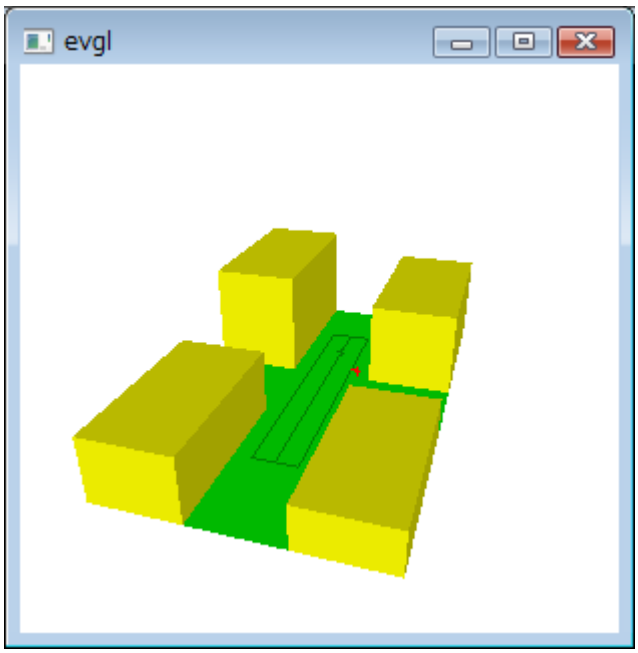
※最後に一度呼び出すことが必要です。

4. サンプルプログラム

```
/*
    sample_rtm.java

    EEM-RTMデータ作成サンプルプログラム

    使い方：
    > javac sample_rtm.java RtmData.java
    > java sample_rtm
*/
```

```
class sample_rtm
{
    public static void main(String args[])
    {
        // new

        RtmData d = new RtmData();

        // title

        d.setTitle("sample");

        // frequency

        d.setFreq(2e9);

        // material

        d.addMaterial(3.0, 0.2);
        d.addMaterial(5.0, 0.1, 0.03);

        // geometry

        d.addPillarRect(+10, +10, +50, +30, 0, 10, 2);
        d.addPillarRect(+10, -10, +50, -30, 0, 15, 2);
        d.addPillarRect(-10, +10, -50, +30, 0, 20, 2);
        d.addPillarRect(-10, -10, -50, -30, 0, 25, 2);

        d.addPlaneRect(-50, -30, +50, +30, 0, 3);

        // tx

        d.setAntennaDipole(0, 0, 90, 1);
        d.setTx(12, 12, 15, 1, 0);

        // rx0

        d.setAntennaIso(1);
        d.setRx0(-20, 0, 1.5);

        // rx1

        double rx1_x[] = {-30, +30};
        double rx1_y[] = {0, 0};
        double rx1_z[] = {1.5, 1.5};
        d.setRx1(rx1_x, rx1_y, rx1_z, 60);

        // rx2

        double rx2_x[] = {-30, -30, +30, +30};
        double rx2_y[] = {-5, +5, +5, -5};
        double rx2_z[] = {1.5, 1.5, 1.5, 1.5};
        d.setRx2(rx2_x, rx2_y, rx2_z, 10, 60);
```

```
        // solver
        d.setSolver(2, 90, 1, false, false, false, true, 0, 10);
        // misc
        d.setMisc(500, 20);
        // output
        d.save("sample.rtm");
    }
}
```



サンプルプログラムの出力データを EEM-RTM で開いた図