Electric and Magnetic field macros in EGSnrc

V. N. Malkov and D. W. O. Rogers

Carleton Laboratory for Radiotherapy Physics, Physics Dept, Carleton University,

E-mail: victormalkov@cmail.carleton.ca and drogers@physics.carleton.ca

CLRP-16-02

Latexed January 13, 2017

Contents

1 Introduction

This report in intended to aid the user in the implementation of the electric and magnetic fields (EMFs) macros in EGSnrc. These macros were developed based on the initial theory proposed by Bielajew[1], and several improvements on that initial implementation are introduced in the code presented here (which will be referred to as EEMF for enhanced electric and magnetic field transport code). In the case of the magnetic field, these improvements and the overall algorithms have been documented and well tested[2]. At present this code addresses transport in constant or very slowly varying fields.

The first order solution to the effect of the EMF on the charged particle transport was available in EGSnrc since the transition from EGS4. However a few user changes in the code were required to make that set of macros functional. With the development of MRgRT technologies, these macros began to be tested by several groups, and became the default EMF transport code in EGSnrc. In EGSnrc code versions dated after October, 2015, input of the electric and magnetic fields could be performed directly though the input files instead of having to recompile each time a new field was needed. The EMF macros described in this report were introduced into the EGSnrc system in January, 2017, through the GitHub system. Some of the main differences between the new code and the original EMF macros are:

- Introduction of a three point integration algorithm (3PI) for PRESTA-II and two point integration for PRESTA-I to take into account scattering during condensed history (CH) steps.
- Addition of a specialized single scatter (SS) algorithm which uses an analytical expression to transport the particles. This permits accurate transport with a precisely known transport distance, larger step sizes to be taken in low density media, and proper boundary crossing.
- Inclusion of a general boundary crossing algorithm (BCA) that works with any properly implemented geometry definition (`howfar` and `hownear` accurately report distances to the geometrical boundaries).

There are several additional changes that were introduced to accommodate for the limi-

tations of the algorithm and for efficiency improvements. The specifics of these changes will be described below in the 'parameters' section. Further, description on how to incorporate the new EMF code into an existing EGSnrc installation is given, and details regarding the use of the macros for all installations is also described.

2 Supported transport

The EEMF macros provide support for magnetic and electric field transport under the assumption that the magnetic field remains constant during individual electron steps (*i.e.*, a new field strength is acquired at the start of each CH or SS step). The magnetic field additions have been extensively tested in the PRESTA-II and SS modes, and it is recommended that these algorithms are used with the default settings listed below. The electric fields have yet to be properly tested and verified for use with larger step sizes. The PRESTA-II algorithm has the capability of running in 1PI and 3PI modes (the magnetic field algorithm optimizes for efficiency by using both), and PRESTA-I is capable of performing EMF transport in 1PI and 2PI. The SS mode supports the 1PI approximation for both electric and magnetic fields or an analytical solution mode for the magnetic field only.

3 Implementing in older versions of EGSnrc

EGSnrc versions installed prior to the introduction of the EEMF package need a few changes to be made to the `egsnrc.mortran` and `egsnrc.macros` files. These two files are located in `$HEN_HOUSE/src` of the EGSnrc installation directory[1]. The necessary changes to the individuals files are describe below, and the line numbers (LN) provided are approximate since variations between EGSnrc versions can occur (LNs are based on the develop branch of the EGSnrc code as of December, 2016). If you have have macro definitions for the EMF transport from the previous EMF implementation, these can be left in the code or removed since they do not interact with the current code so long as they are not both active at the same time (*i.e.*, if the definitions of the other macros are nulls).

[1] If you don't have write permission on these files, copy them to your application's area and adjust the Makefile and egspp1.spec files (the latter will also need to be copied from `$HEN_HOUSE/specs` to your application's area)

3.1 Changes to egsnrc.mortran

Listed below is a description of the required changed in the egsnrc.mortran file. Line number are given and the corresponding subroutine is listed in the brackets.

1. Add ;COMIN/EM/; to the following locations:
   - LN 997 (subroutine ELECTR): Below or part of the ;COMIN/EII-DATA/; expression.
   - LN 4120 (subroutine msdist_pii): Below or part of the ;COMIN/ELECIN, THRESH,UPHIOT, RANDOM,CH-Steps/; expression. Some code versions may have EMF-INPUTS as one of the COMMONS available in this expression already, and is related to the implementation of field parameter inputs via input files. If the EMF-INPUTS is present, it should be left as-is.
   - LN 4368 (subroutine msdist_pi): Below the ;COMIN/ELECIN,THRESH,UPHIOT, RANDOM/; expression.
2. LN 1191 (subroutine ELECTR): Add the expression $EMFIELD_INITIATE_SET_TUSTEP; right below the expression tustep = min(tustep,max(tperp,skindepth));.
3. LN 1381 (subroutine ELECTR): Add (a few lines past the above major change) $EM_FIELD_SS; just after vstep = ustep;.
4. In the ELECTR subroutine, a change to the vacuum transport definition is required by replacing the code block starting on LN 1365:

```
IF(ustep ~= 0) [
  "Step in vacuum vstep  = ustep; tvstep = vstep; "( vstep is ustep
  truncated (possibly) by howfar " tvstep is the total curved path
  associated with vstep) edep = pzero; "no energy loss in vacuum
  $VACUUM-TRANSPORT-EM-FIELD;
      "additional vacuum transport in em field
  e_range = vacdst; $AUSCALL($TRANAUSB); "Transport the particle x(np) =
  x(np) + u(np)*vstep; y(np) = y(np) + v(np)*vstep; z(np) = z(np) +
  w(np)*vstep; dnear(np) = dnear(np) - vstep;
     "(dnear is distance to the nearest boundary " that goes along with
```

```
     particle stack and " which the user's howfar can supply (option)
  irold = ir(np); "save previous region $SET-ANGLES-EM-FIELD;
     "default for $SET-ANGLES-EM-FIELD; is ; (null)
      "(allows for EM field deflection
]"end of vacuum step
```

With the following code:

```
 IF(ustep ~= 0)
 [
 edep = pzero; "no energy loss in vacuum
 IF $EM_MACROS_ACTIVE [;
  "transport in EMF in vacuum:"
  "only a B or and E field can be active"
  "(not both at the same time)"
   $EMFieldInVacuum;
 ]ELSE[;
  "Step in vacuum vstep  = ustep; tvstep = vstep; e_range = vacdst;
  $AUSCALL($TRANAUSB); "Transport the particle
  x(np) = x(np) + u(np)*vstep;
  y(np) = y(np) + v(np)*vstep;
  z(np) = z(np) + w(np)*vstep;
  dnear(np) = dnear(np) - vstep;
    "(dnear is distance to the nearest boundary
    " that goes along with particle stack and
    " which the user's howfar can supply (option)
  irold = ir(np); "save previous region
 ]"end vacuum step
]"end of non-EMF block"
```

The above is done between the comment expression `"Do fast step` and `"end of vacuum step`.

5. LN 1440 (subroutine ELECTR): Add $ADD_WORK_EM_FIELD; right below edep = de;. If it is there, replace $ADD-WORK-EM-FIELD; which is part of the previous EMF implementation (note - vs _).

6. LN 1563 (subroutine ELECTR): Replace the following code
```
x_final = x(np) + u(np)*vstep;
y_final = y(np) + v(np)*vstep;
z_final = z(np) + w(np)*vstep;
```
with,
```
 IF ~($EM_MACROS_ACTIVE)[;
 x_final = x(np) + u(np)*vstep;
 y_final = y(np) + v(np)*vstep;
 z_final = z(np) + w(np)*vstep;
]
```

7. LN 4248, about 9 lines from the end of subroutine msdist_pii: Add $EMFIELD_PII; above the line with the comment "Transport.

8. LN 4470, about 9 lines from the end of subroutine msdist_pi: Add $EMFIELD_PI; above the line with the comment "Transport.

3.2 Changes to egsnrc.macros

The goal of the changes here is to introduce macros which will make any of the EMF changes introduced to egsnrc.mortran negligible when the EMF macros are not being used. The user should ensure that the following code block is inserted into the egsnrc.macros file.

The exact location of this definition is not crucial, but the user may locate any of these REPLACE calls already made in their file (from previous EGSnrc version) and append any of the missing ones listed here. The previous ones, if there, are all together so the above can just be added right after them (search for $ADD_WORK_EM_FIELD;)

```
"TEMPLATES FOR PERFORMING CHARGED PARTICLE TRANSPORT IN EM FIELD"
REPLACE {$SET-TUSTEP-EM-FIELD;} WITH {;}
REPLACE {$SET-USTEP-EM-FIELD;} WITH {;}
```

```
REPLACE {$VACUUM-TRANSPORT-EM-FIELD;} WITH {;}
REPLACE {$SET-ANGLES-EM-FIELD;} WITH {;}
REPLACE {$SET-TVSTEP-EM-FIELD;} WITH {;}
REPLACE {$ADD-WORK-EM-FIELD;} WITH {;}
REPLACE {$EMFIELD_INITIATE_SET_TUSTEP;} WITH {;}
REPLACE {;COMIN/EM/;} WITH {;}
REPLACE {$EMFIELD_PII;} WITH {;}
REPLACE{$EMFIELD_PI;}WITH{;}
REPLACE{$EM_FIELD_SS;}WITH{;}
REPLACE{$ADD_WORK_EM_FIELD;}WITH{;}
REPLACE{$EMFieldInVacuum;}WITH{;}
REPLACE{$EM_MACROS_ACTIVE}WITH{.false.}
```

Once all of the above has been done, the application can be compiled just as a check that everything is properly in place.

4 Using the EEMF macros

Once all of the code changes have been made as given in the previous sections or a new version of the EGSnrc files has been downloaded, turning on the EEMF is fairly simple. The user needs to point the `make` files to the file containing the EEMF macros. This is different for the different types of applications and details are given in section 3.14.1 of PIRS701(2016 version). But briefly:

4.1 egs++ applications

In the application's local `Makefile`, add the explicit path to the `EEMF_macros.mortran` to the line defining `EGSPP_USER_MACROS`, e.g., for the application `cavity` change:

```
EGSPP_USER_MACROS = cavity.macros
```

to:

```
EGSPP_USER_MACROS = cavity.macros  $(EGS_SOURCEDIR)EEMF_macros.mortran
```

where in this example it is assumed the `EEMF_macros.mortran` file has been placed on the `$HEN_HOUSE` with other source code, but it could be on the local area as well (like the `cavity.macros` in this case).

4.1.1 Mortran applications

In this case, add the pointer to the `EEMF_macros.mortran` file anywhere above the `$(USER_CODE).mortran` line in the `user_code.make` file.

4.2 Defining electric and magnetic fields

In more recent versions of the EGSnrc code, definition of electric and magnetic fields can be done in the user's input file. The user can check if their version supports this type of input by searching the `egsnrc.macros` file for the definition of the COMIN: `EMF-INPUTS` (located roughly on LN 626 of the file, if it is present, near the REPLACE call for `$COMIN-EII-INIT;`).

Additionally, an error during code compilation which states that the variables BxIN, ByIN, BzIN, EzIN, etc. are not defined indicates that input file EMF definition is not supported.

If input file submission is not supported the user will have to define the electric and magnetic fields using macro definitions as described in section 4.2.2. If the input file definitions are supported the following section described their use.

4.2.1 Definitions in the input files

If the installation supports this type of input, i.e. if you started from one of the more recent EGSnrc releases, the user can define the field by defining the fields in the `MC transport parameter` section of the input file (as shown in section 3.14.1 PIRS-701 - 2016 version). For magnetic fields (other parameters are included in this section of the input file, but omitted for the examples):

```
:start MC transport parameter:
...
# B-field defined in the X , Y,  Z directions
```

```
Magnetic Field = 0, 1.5, 0 # magnetic field in tesla
EM ESTEPE = 0.2
...
:stop MC transport parameter:
```

and for electric fields:

```
:start MC transport parameter:
...
# E-field defined in the X , Y,  Z directions
Electric Field = 0, 3, 0 # electric field in V/cm
EM ESTEPE = 0.2
...
:stop MC transport parameter:
```

The EM ESTEPE variable is a step size control parameter which is set to be 0.02 by default, and should be set to 0.2 for magnetic field simulations with the EEMF macros. If both electric and magnetic fields are defined, then a 3PI technique will be used for the CH transport, and a 1PI will be applied for the SS.

4.2.2 Defining electric and magnetic fields using macros

If the user's EGSnrc version does not support input file EMF input or varying fields are needed, a macro definition by the user is necessary. If EMF input is not supported in the input file, the macro `$GET_EM_FIELD(#, #, #);` must be defined in the application specific macros file, the EEMF macros file directly, or in a user specific macros file that is called by the `make` file. The three parameters (#'s) are the x, y, and z positions of the electron at the beginning of a step. In this macro the vectors of the electric and magnetic field should be defined (if any are omitted, they will default to zero). The value of the `delta_u` variable should also be set (recommended to be 0.2 for magnetic fields, and set to 0.02 by default). The following is a sample definition of a constant magnetic field:

```
REPLACE{$GET_EM_FIELD(#, #, #);}WITH{;
  "Magnetic field:"
```

```
  Bx = 0.;    By = 1.5;    Bz = 0.;
  "Electric field:"
  Ex = 0.;    Ey = 0.;     Ez = 0.;
  "delta_u definition from the user"
  delta_u = 0.2;
}
```

The electric field and `delta_u` definitions could have been omitted (with a default of no E-field and delta_u of 0.02). The choice of `delta_u` can significantly impact simulation efficiency and a value of 0.2 is recommended for magnetic field simulation with the EEMF package. Definition of electric fields can be done by including non-zero electric field components in the above macro.

Defining a varying magnetic field can be easily accomplished by using the position variables. Below is a sample definition of a radially decreasing magnetic field in the y-direction confined to a 5 cm radius about the origin:

```
REPLACE{$GET_EM_FIELD(#, #, #);}WITH{;
  "Magnetic field:"
  Bx = 0.;
  IF(sqrt({P1}*{P1}+{P2}*{P2}+{P3}*{P3}) < 5.)[;.
    By = 1.5*(1-sqrt({P1}*{P1}+{P2}*{P2}+{P3}*{P3}));
  ];
  Bz = 0.;
  "delta_u definition from the user"
  delta_u = 0.2;
}
```

If defining an electric field, the default settings in the EEMF macros apply electric field energy change based on a constant electric field over each step. Instead, the user may want to use an electric potential definition to determine the energy change due to the E-field over the step. This can be done by defining the macro `$setEFPotential(#, #, #, #);`, where the first three inputs are the x, y, and z coordinates at the start of the step, and the fourth input is the potential that is returned by the function (in V/cm). Here a sample

electric field definition is provided, and this is the electric potential used by Rawlinson *et al.* for determining the effect of charge build up in plastic phantoms on the response of an ion chamber[3]. Inside and at the surface of the ion chamber (which is simulated as an cylinder with a radius of 0.35 cm) the electric potential is set to zero. The potential field is defined in terms of the maximum electric field $E_o = 1100$ kV/cm, and therefore requires a multiplication by 1000 to obtain units of volts for the potential.

```
REPLACE{$setEFPotential(#, #, #, #);}WITH{
"{P1}, {P2}, {P3}, are the x, y, z positions"
"{P4} is the potential to be returned"
r0_em =sqrt({P2}*{P2} + ({P3} + 1.5)**2);
IF(r0_em>0.35)[;
"calculate the potential at this point (the 1000 is a conversion to Volts)"
"E0=1100kV/cm"
{P4} = -1000*1100.*.35*log(r0_em/.35)*(1-.2273*{P3}/r0_em-.409*({P3}/r0_em)**2);
]ELSE[
"potential is zero at the surface and inside the chamber"
{P4} = 0.;
]
}
```

The definition of these macros can be done in a new user macros file which will then have to be included in the Makefile as described in section 4, or they can be included in the application specific macros files when available. For example, this would be the `egs_chamber.macros` files for `egs_chamber`, or `dosxyznrc_user_macros.mortran` in dosxyznrc. Alternatively, the existing macro definition can be modified directly in the `EEMF_macros.mortran` file, but this type of change would have an effect on all codes using this file.

The disadvantage of defining the EMF using this macros approach is that the application code needs to be recompiled after each change to the field. This is fine if studies are being done with a single magnitude field, but can become cumbersome if a variety of fields is needed.

5 Code options and parameters

There are several aspects to the EEMF code that can be changed by the user if needed. These options deal with transport options, efficiency improvements, and code output. These options are described below, and can be edited by either changing the macro's value directly in the `EEMF_macros.mortran` file or by creating another file containing user defined macros and calling on this file in the Makefile of the application code (as described in section 4.2.2. Many of these values should not be changed by the user, but are listed here for completeness. The `delta_u` parameter can be changed for step size effect studies, and some of the boolean value can be changed to suit the user's preferences.

- **$MINS_ANG**: This parameter is the upper bound on the `delta_u` value used. Above this value 3PI is used in PRESTA-II steps and below this value 1PI is used. This 1PI override of 3PI occurs only if the `$orderAdaptive` variable is set to be true. Default value of `$MINS_ANG` is 0.05.

- **$deltaUDefault**: This is the default value value of `delta_u` if input file EMF definition is not supported on the user's EGSnrc version. Default value of 0.02.

- **$MAXS_eLoss**: Similar to the `delta_u` parameter, this variable is a step size restriction parameter that is applied to the electric field calculations. This step size restriction limits the energy loss due to the electric field over the step to the fractional value set by this parameter. Default of 0.1.

- **$IntegrationAlg**: This parameter determines the use of the integration algorithm for the CH mode used. For either PRESTA-I and II, setting this value to '0' will make the algorithm use 1PI with a normalization of the direction vector that simply adds the effect of the EMF and normalizes to unity (this ignores proper momentum conservation for the magnetic field). Setting the value to '1', will use the 1PI technique with correct normalization. IN PRESTA-I, setting the value to '2' will use 2PI, and for PRESTA-II setting it to '3' will use 3PI. Default is '3'.

- **$ssFieldAlg**: Selection of the single scatter algorithm. Setting it to '1' will use 1PI

for both magnetic and electric fields, and a setting of '2' will permit the use of the analytical transport equation for magnetic field transport only. Default is '2' (defaults to 1PI for electric fields).

- **$BCA_lin_buffer**: Distance near the boundary in which EMF effects are ignored, and linear transport is performed. Default of the minimum of the skindepth and `1E-6` cm.

- **$BCA_lin_dist**: Maximum distance that a particle can travel linearly within the linear buffer region. Default is the minimum of the skindepth and `1E-5` cm.

- **$orderBoundary**: Boolean variable that allows the CH algorithm to default back to 1PI if the effect of the B-field using 2PI or 3PI may potentially cause an accidental step over a boundary (step size restrictions are based on 1PI effects). Default is .true., and should not be changed as this will cause geometry errors.

- **$orderAdaptive**: Boolean variable that works with the `$MINS_ANG` variable, and allow the algorithm to switch from 3PI to 1PI if the step size is too small. This is to gain efficiency benefits from using 3PI. Default of .true., and turning off will increase computational time.

- **$duAdaptive**: Boolean variable that turns on scaling of the step size determined by the `delta_u` from the current magnetic field to a 1.5 T field. Default of .true., and turning off may cause numerical errors.

- **$energyLossViaPotential**: Boolean for turning on calculation of energy change in an electric field using the electric potential definition. Default of .false., and the potential should be properly turned on if it is turned on.

- **$EMInternalOutput**: Boolean to control the output of the EEMF parameters during the simulation. Default of .true., and turning this off will silence the output of the algorithm during simulation (this is a one time output to show the field and parameter values at the beginning).

6 Test cases

A test case for determining if the EEMF code is properly implemented is given here. For recently installed EGSnrc versions, the input file can be found in the `egsnrc/dosrznrc/examples` folder and users with older versions can obtain a copy at (...). This test case is performed with the magnetic field in the DOSRZnrc application.

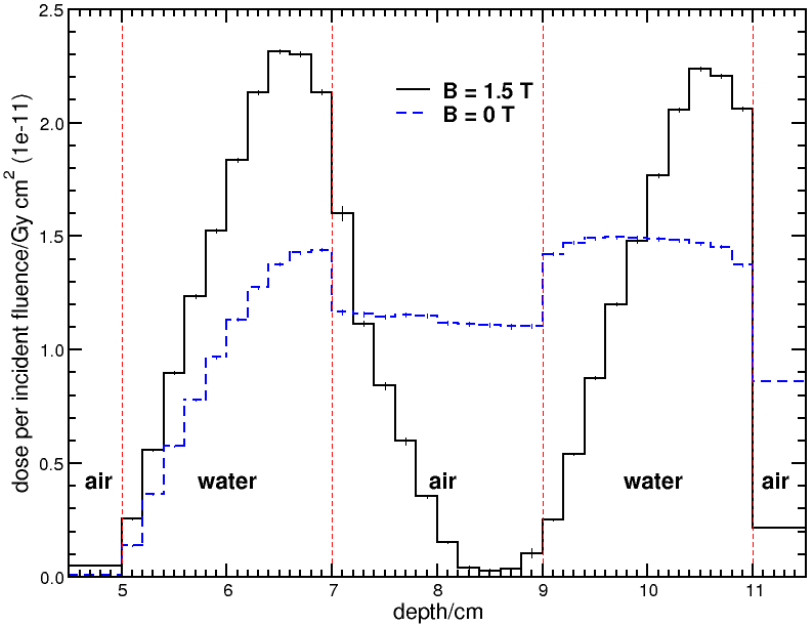

Figure 1: Central axis depth-dose curve for a segmented cylindrical water and air phantom in a 1.5 T or 0 T magnetic field perpendicular to the incoming 5 cm radius, 5 MeV monoenergetic photon beam. The vertical dashed lines represent the water-air interfaces.

A 10 cm radius, 12 cm long, segmented cylindrical geometry is simulated. The first 5 cm are composed of air, followed by 30 0.2 cm thick slabs (the first 10 are water, then 10 of air, and 10 more of water), and the phantom is terminated with another 5 cm section of air. A 5 cm radius parallel (source 0 in DOSRZnrc) 5 MeV photon beam is simulated incoming along the z-direction (front surface of the phantom). The dose is scored along the central axis in a 0.5 cm radial region. Range rejection with an ESAVE of 2.0 MeV is turned on, ECUT is set to 0.700 MeV, and PCUT is set to be 0.01 MeV. The simulation can be performed within an hour on single CPU core (depending on system performance) to achieve 0.5 % statical uncertainty in the water region (16×10^6 histories). The icru700 pegs4 file is used with the materials AIR700ICRU and H2O700ICRU. The user may wish to alter the simulation to run with lower particle creation and cut-off energies by using a different pegs4 file or the icru521

file, but testing showed little difference to justify the efficiency loss for this simple test case.

The results of the simulation with a 1.5 T magnetic field perpendicular to the incoming photon beam and those for a no magnetic field simulation are provided in Figure 1. The expected sharp dose increase, caused by the magnetic field's electron return effect, is present near the exit surfaces of the water regions.

For more complex testing, ion chamber simulation can be run and the results compared to literature values[2]. The egs++ geometry package is extremely adaptive and allows production of fairly complex geometries, but there are cases when boundary crossing error occur. In the case that errors, such as negative ustep, appear during simulation, the user should determine if these errors occur when the EEMF package is turned off. If the error persists with the absence of the magnetic field code, the error is linked to the geometry package. In instances when errors appear at an extremely low rate, the user may increase the allowed number of errors from the default of one by inserting `geometry error limit = n` in the run control section of the input file (where n is the number of permitted error in the simulation).

References

[1] A. F. Bielajew, The effect of strong longitudinal magnetic fields on dose deposition from electron and photon beams, Med. Phys. **20**, 1171 – 1179 (1993). (p 3)

[2] V. N. Malkov and D. W. O. Rogers, Charged particle transport in magnetic fields in EGSnrc, Med. Phys. **43**, 4447 – 4458 (2016). (pp 3 and 15)

[3] J. A. Rawlinson, A. F. Bielajew, D. M. Galbraith, and P. Munro, Theoretical investigation of dose effects due to charge storage in insulating phantoms, Med. Phys. (abstract) **11**, 383 (1984). (p 12)