# Summary of the Recent Updates of EPICS

March 15, 2013

## Contents

This manual describes the recent updates (for versions 9.10 and newer) as well as some summary including old stuff and some obsolete parts of EPICS[1].

The new stuff not included in the previous release of this manual is marked with ♠.

# 1 Remarks

**Emin and tracking scheme** In v9.10, we introduced automatic Emin setting depending on the size of the detector component, special treatment of such setting (Emin and/or quenching factor) for a specified component, etc. To stop tracking of a particle, v9.08 used only Emin information but v9.10 could use residual range information. Results by v9.10 have been compared with those by v9.08 to find that there is no statistically significant difference between them (as to the energy deposit, $< 0.5\%$ ). If we use residual range information, the execution seed is improved by about 20∼30%.

**New volume-shape** As to the treatment of a detector component such as described by "pipe_y", there is some subtle difference between 9.08 or earlier and 9.10. v9.10 introduced new shapes (octagon and honeycomb which permit "_y" type notation)[2]. This cannot be treated by v9.08 without some modification of the source code. (The maximum character length of the shape name was changed from 8 to 12). v9.081 is a version that can understand this new feature with a minimum change of v9.08.

♣New volume-shapes are described in
`http://cosmos.n.kanagawa-u.ac.jp/EPICSHome/NewVol.pdf`

**Cosmos version** The following combination is recommended (rather must be used). Cosmos7.581 is little bit different from 7.58; mis-conversion of $\eta$ and $\Lambda^0$ code from

Table 1: Recommended combination of Cosmos and Epics

| | EPICS | Cosmos |
|---|---|---|
| | 9.081 | 7.581 |
| | 9.10 | 7.60 |
| ♣ | 9.131 | 7.631 |
| ♣ | 9.15 | 7.633/♠7.634 |

QGSJET-II was corrected. Completely stopped anti-proton cannot annihilate in the Jam code and this leads to infinite loop. Although very rare, $\pi^0$ makes a collision in Jam which cannot treat it. These were corrected. Cosmos7.60 could use the "sofia" code for photo-hadron production.

**Jam code** The Jam code in 9.10/9.081 is the same as in older EPICS's (Y. Nara, Nucl. Phys. A 638, 555c (1998));

It has some problem with the treatment of spectator nucleons in heavy ion collisions. Spectator nucleons in a projectile or target, emerge as independent nucleons

[1]Epics9.081 was kept as a minimal update from Epics9.08, since v9.10 contained rather a lot of changes from v9.08 and we were afraid that v9.10 might have serious bugs. Now, we think we need not go back to v9.081

[2]These new components and new treatment of "_y" type specification are described in http://cosmos.n.kanagawa-u.ac.jp/EPICSHome/config.pdf

after collision (even for elastic collisions). That means, for example, when Fe is a projectile, we will never get He secondaries after a collision. The current EPICS treatment of these nucleons is that we accept all nucleons from projectile while discard all spectator nucleons from target. (Let's call this Jam Jam1)

There is another Jam code embedded in the PHITS code[3] In this Jam, the spectator problem has been solved by the PHITS author . Let's call this Jam Jam2. However, Jam2 inherits some defect existed in the original Jam, say, K0 cannot be a projectile particle and other minor bugs. Such problems have been corrected in Jam1. **The implementation of Jam2 is under investigation**.

♣Using Jam2 at high energies was found to be problematic (needs very long computation time for breaking nucleus). Now we introduced a spectator break-up scheme for Jam1. JamFragment=1 (default) in the param file ($HPARAM part) will use this scheme. To disable this, JamFragment=0 may be given. However, the treatment here is still far from satisfactory one (especially for elements heavier than Fe).

**Jam1 and PHITS combination** In the current versions, combination of
IntModel='"phits 2.5 "jam" 5 "dpmjet3"'
was expected to improve the proton primary case at low energies (< 200GeV). However, the current tendency is rather opposite and contradicts earlier observation. This is under investigation.

♣It was found that the jam code gives $P_t$ which is much smaller than other codes (espcially for heavy targets) and cannot explain SPS beam test results for the shower spread. So the better interaction model is probably
IntModel='"phits" 2 "dpmjet3"'

**Intel compiler vs VAX extension** This is very much annoying stuff. Cosmos/EPICS are old and use structure construct based on the so called VAX extension[4]. The compiler seems to have a bug in dealing with the VAX style structure and in some case we encounter quit strange phenomena. Suppose a code fragment like

```
structure /epPos/
     real(8):: x, y, z
end structure

record /epPos/  p(100)
integer n
real(8):: d
...
...
p(n).x = 0.
p(n).y = d
```

[3]PHITS is a one complete package (K. Niita et al., Radiation Measurements 41, 1080 (2006)) for particle transport (at low energies). PHITS includes several interaction models including Jam. The interaction model specified by IntModel='"phits"' implies such models but does not include Jam. An appropriate model inside PHITS is selected depending on the energy and projectile type.

[4]Before Fortran90, there was no structure construct formally in Fortran. However, the C-language style structure has been used long time and it is called VAX extension. This extension is supported by the Intel Fortran compiler but Intel seems not serious about its support and the recent compiler says it will become obsolete in the future versions.

```
p(n).z = 0.
```

In some case, even if "d" is non zero, p(n).y becomes 0. This dose not happening always, but seems to depend on other environment. So far we could not detect the condition for such happening.

One workaround is to write

```
p(n) = epPos(0.d0, d, 0.d0)
```

Coding similar to `p(n).y = d` should be avoided. The users are recommended to use epPos( ..) style coding in their UserHook.

## 2 dE/dx of heavy ions

The ionization energy loss rate ($-dE/dx$; hereafter we regards $dE/dx$ has a positive value) of heavy ions at low energies has been treated by an effective charge method and is fairly accurate down to a few hundred MeV/n where accelerator test experiments are usually performed. However, at lower energies, we need a more accurate treatment. We introduced two things:

- A better effective charge method (Pierce and Blann. Phys. Rev. 1968 vol.173,No2. pp.390-404. With later erratum). Some modification has been done for the He case.
- Incorporation of the SRIM data (`http://www.srim.org/#SRIM`). At present the data for plastic scintillator and SciFi are available. (Both are currently regarded as the same media).

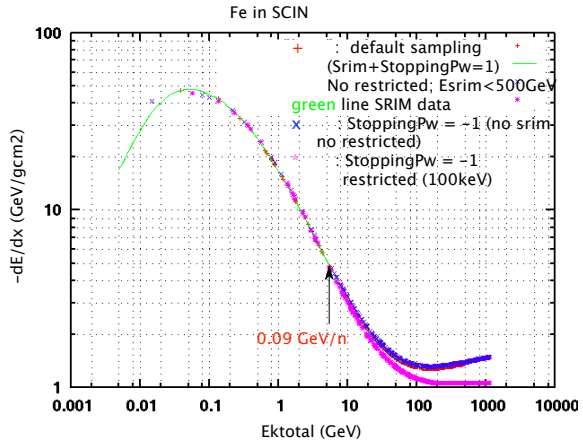


Figure 1: dE/dx of Fe in SCIN. Green line is by SRIM. Lower dots are the restricted energy loss rate with RecoilKeMin=100 keV. Upper one the full $dE/dx$. They coincide below 0.09 GeV/n. Blue crosses (x) by StoppingPw=-1 (new effective charge method) agree with the green line fairly well.

Table 2: Related parameters. D=xx means the default

| variable | in | value | description |
|---|---|---|---|
| StoppingPw | epicsfile | D=1 | When SRIM data is available at low energies (see `SrimEmax` below), use it. At higher energies, the effective charge method shown above is used. |
| | | 2 | Same as above, but the old effective charge method is used. |
| | | -1 or -2 | Even if SRIM data exists, it is not used. The effective charge method corresponding to 1 or 2 is used. |
| SrimEmax | epicsfile | D=0.09 | (GeV/n). Above this energy, SRIM data is not used. In the actual simulation, this value should not be much larger than this since the SRIM data is for the average of the total energy loss including high energy $\delta$-rays from the knock-on process. We use the restricted energy loss and $\delta$-rays above `RecoilKeMin` are randomly generated. |
| MAXHEAVYCHG | ZepMaxdef.h | 30 | In Epics/epics. Maximum charge that can be treated by SRIM. If changed, recompiling of all sources may be needed. |
| `MAX_SRIMMEDIA` | ZepMaxdef.h | 3 | In Epics/epics. Maximum number of media for which SRIM data can be used. |

### 2.1 Creating a new SRIM data

If the user want to add more SRIM data to the existing data or to create new SRIM data for a particular media, Epics/Util/SRIM may be consulted. The Readme there will tell how to do. To see continuation of the SRIM data to the larger energy region, testdEdx.sh in Epics/Util/Elemag/dEdx may be used.

## 3 Minimum Energy

Before v9.10, the minimum energy of particles (gamma, electrons...) during the particle tracking is fixed by parameters given in `epicsfile`. Although it can be dependent on the particle type, it is unique and independent of media thickness. In some case we have a thick PWO and thin Si, or have to transport particles in a very long beam pipe before they reach the detector. In such cases, the minimum energy may be better to be dependent on the detector component.

In v9.10 or later,

- Automatic determination of the minimum energy is possible. The value is fixed for each component by considering its media and thickness.

- When the particle energy becomes lower than the minimum, the particle tracking is not necessarily stopped; in some case further tacking is continued as described in Table below.

- If the value fixed by the automatic way or old way is not satisfactory, there is a mean to fix the minimum for each particular component. See the `Modifier` section.

- The procedure for the automatic determination could be changed by the user. (Probably, such needs will be rare).

Basically, the default `Eabsorb` specifies that the kinetic energy of a charged particle and photon is absorbed at the point where the particle energy becomes lower than the predefined minimum. However, if the particle can decay or annihilate, EPICS may follow the particle down to 0 or some lower energy.

### 3.1 Automatic Emin

The automatic minimum energy is calculated in the subroutine located in

Epics/prog/UserMayChange/epAutoEmin.f

If `AutoEmin` is non 0, this program is called to fix the minimum for a given component. The default value, 2, specifies the following procedure.

- Use the input "minimum" thickness of the component in g/cm$^2$($= t$).

- Compute, max(min($150\sqrt{t}$, 10), 150)$\times 10^{-6}$ (GeV). That is, the value is always between 10 and 150 keV. The value is used for photons (EminG). For electrons, 2 times of this is used and electron mass is added (EminE). Later, the range consideration is applied.

- For AutoEmin=1, we compute max(min($100\sqrt{t}$, 10), 100)$\times 10^{-6}$ (GeV) and assigned to EminG and EminE.

- In both of the above cases, the value of `RecoilKeMin` is fixed by

  max(EminG, $14Z^2 10^{-9}$)

  where $Z$ is the effective atomic number of the media. That is, the rough K-shell energy is considered (GeV).

- The values for `KEmin` is fixed to be the same as `EminG`

- `EminH` is unchanged.

If the user give a value $> 2$ to `AutoEmin` and add a program fragment in epAutoEmin.f, different treatments can be used. If it is 4, the range consideration will be done same as it is 2.

Table 3: Related parameters

| variable | in | value | description |
|---|---|---|---|
| AutoEmin | epicsfile | D=2 | The minimum energy is fixed automatically (see section for `Automatic Emin`). If the particle energy becomes lower than the minimum, the residual range of the particle is computed and compared with the distance to the boundary of the present component. If the range is smaller than the distance to the boundary, the energy is assumed to be absorbed within the range. |
| | | 0 | The old method is used. The values (EminElec, EminGamma, KEmin, EminH) are listed below. |
| | | 1 | Some smaller values of Emin than the AutoEmin=2 case are employed but we don't consider the range and treated as AutoEmin=0 case. |
| | | 3,4 | Reserved for the user. For 4, the range is considered like AutoEmin=2 case. |
| EminElec | epicsfile | D=511e-6 | (GeV) By historical reason, the minimum for electron is always in the total energy. |
| EminGamma | epicsfile | D=100e-6 | (GeV) Photon minimum energy (different from the one for light) |
| KEmin | epicsfile | D=0 | If 0, the minimum kinetic energy for electron is used. This is for the minimum kinetic energy for other particles than electron and neutron |
| EminH | epicsfile | D=0 | If 0, 20 MeV is used for neutron minimum kinetic energy |
| RecoilKeMin | epicsfle | D=0 | If 0, `EminGamma` is used. The restricted energy loss is computed below this energy and $\delta$-rays are randomly generated above this energy. |
| Eabsorb | epicsfile | D=14 | The bit pattern of `Eabsorb` determines how to treat energy of a particle when its energy becomes lower than the predefined minimum. For details, see `epicsfile` in UserHook/Template. See also below. |

# 4 Modifier

The user may need to specify some specific minimum energy or non default quenching effect coefficients for some components[5]. In such cases, the user may give a number in the `modifier` digit for that component in the `config` file (see Fig.). The user must prepare a `ModifyFile` in which the user give that number followed by necessary entries

[5]The quenching coefficients are normally given in the media file and used as the default. If a modifier described here is to specify a change of the coefficients, the quenching treatment is applied even if there is no default specification.

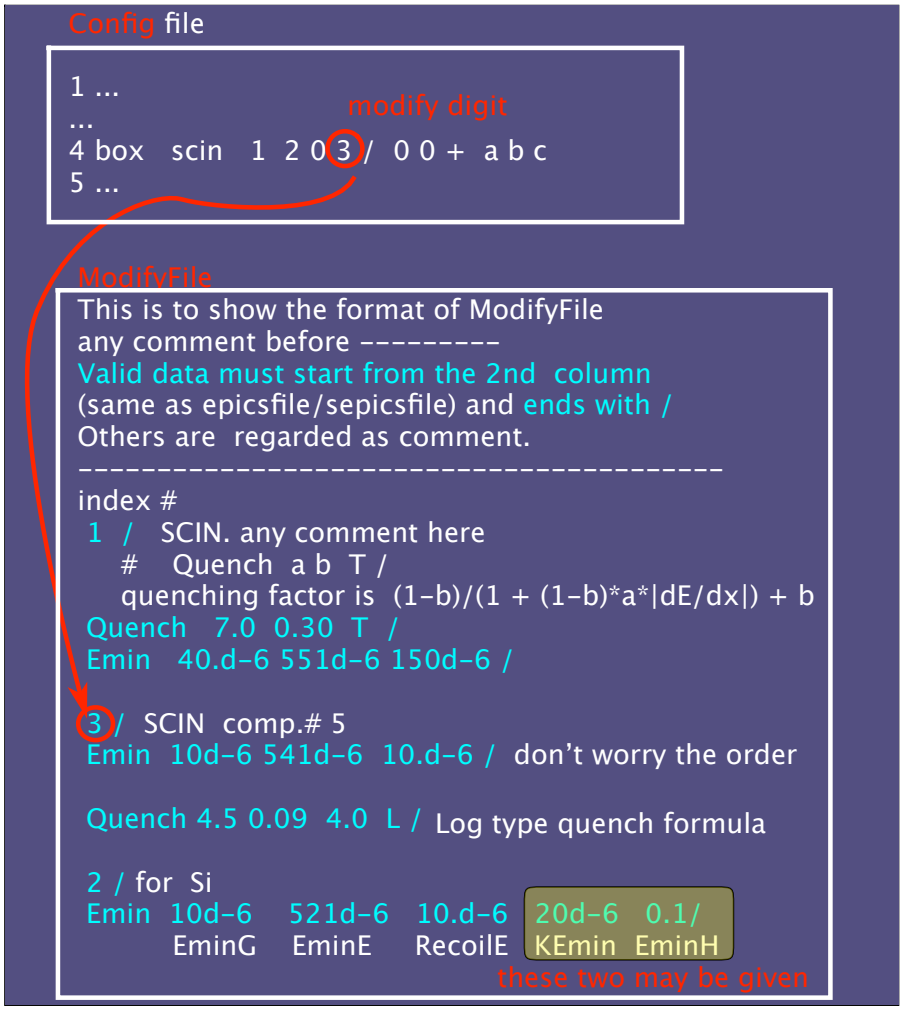


Figure 2: Modifier and ModifyFile

like in Fig. Note that, although we say as if `ModifyFile` were a file, `ModifyFile` itself is not the file name but a variable to contain a path to a file in which modifications are described,
If the modifier field is absent or 0, no modifier is assumed. The modifier number need not be consecutive but it's better to keep it as small as possible to save the memory (must be $< 10^{15} - 1$). The value for the `ModifyFile` must be given in `epicsfile`. The default of `ModifyFile` is ' ' so that no modifier is assumed. The format of `ModifyFile` is similar to `epicsfile`.

The current possible entry is `Quench` and `Emin`. In some case, one may need to change the medium density for the same medium (say, for Air), so `Density` could be an entry candidate. By historical reason, changing density is possible by the notation:

```
2 bos  Air  ...
3 box  Air*1.08  ..
4 box  Air*0.90
```

in the `config` file. Here, 1.08 means 1.08 times higher density than the default given in the media file.

Table 4: Related parameters

| variable | in | value | description |
|---|---|---|---|
| ModifyFile | epicsfile | D=' ' | If the modifier digit is used in the `config` file, a file name here is consulted. However, If this is "blank", all modifiers are neglected. The file should contain the number given in the modifier field and some of the variables shown below. If a modifier number is > the max number in `ModifyFile`, error stop will happen. |
| Quench | ModifyFile | | Coefficients must be given for Tarlè, Birks or Log forrmula (see section for Quenching) . If this is missing for a modifier number, the same action is taken as if the modifier were absent. |
| Emin | ModifyFile | | `EminG, EminE, RecoilE` must be given. `KEmin` and `EminH` may or may not follow them. If last two are not given, the same procedure as `AutoEmin` = 1 case is used. If this is missing for a modifier number, the same action is taken as if the modifier were absent. |

### 4.1 Quenching coefficients

It is assumed that the amount of scintillation light emitted by a heavy ion in a short distance, $\Delta x$, is not proportional to the energy loss (deposit), $\Delta E = \frac{dE}{dx}\Delta x$, in the scintillator but is proportional to $C_f \Delta E$ where $C_f (\leq 1)$ is a $dE/dx$ dependent constant.

**Birks** Before v9.08, the quenching effect is managed by the Birks formula + some corrections. The original Birks formula gives

$$C_f = \frac{1}{1 + a\frac{dE}{dx}} \tag{1}$$

where $a$ is the Birks coefficient. The additional corrections need two more constants, $b$ and $c$. So, for example, the basic media file (Epics/Data/BaseM/SCIN) and media file (Epics/Data/Media/SCIN)[6] contains lines like

```
#  Elem  rho(g/cm^3)    Gas/Solid(1/0)  refl.index     Birks c
     2      1.032              0          1.581      13   9.6  0.5714
```

where the last three numbers are the coefficients, $a, b, c$. However, the correction terms using $b$ and $c$ do not work well and only $a$ has been used in the original formula to get $C_f$. In spite of this fact we will keep 3 numbers for the Birks case; the last two may be any numbers.

Now we may put "B" to express explicitly that these are for the Birks formula:

[6]The `BaseM` file is used only when making the `Media` file. The user may change the quenching coefficients in the media file after creating it. The data in the `BaseM` file need not be changed but it will be better to keep the same value as the `Media` file.

```
#  Elem  rho(g/cm^3)    Gas/Solid(1/0)  refl.index     Birks c
     2      1.032             0           1.581   13  9.6  0.5714 B
```

The unit of $a$ is g/cm$^2$/GeV.

**Talré** A better formula by Talré is now usable (G. Tarlé, S.P . Ahlen and B.G . Cartwright, Astrophys . J. 230 (1979) 607):

$$C_f = \frac{1-b}{1+(1-b)a\frac{dE}{dx}} + b \tag{2}$$

and the (basic) media file format is (e.g., for $a = 8$ and $b = 0.35$)

```
#  Elem  rho(g/cm^3)    Gas/Solid(1/0)  refl.index    Talre c
     2      1.032             0           1.581     8  0.35  T
```

We need two coefficients, $a, b$ and "T". The unit of $a$ is the same as the Birks case, i.e, g/cm$^2$/GeV and $b$ is unitless.

**Log** Another purely empirical formula is a "Log" type[7] which needs three coefficients $a, b$ and $c$:

$$z = a\frac{dE}{dx} + 1 \tag{3}$$

$$C_f = z^{-b\log(cz)} \tag{4}$$

The (basic) media file format would be

```
#  Elem  rho(g/cm^3)    Gas/Solid(1/0)  refl.index    Log quench
     2      1.032             0           1.581     4.6 0.09 5.1  L
```

The three coefficients, $a, b, c$, must be followed by "L". The unit of $a$ is as before (g/cm$^2$/GeV) and $b$ and $c$ are unitless.

So far the format is for the (basic) media file and the values there are used as default for that medium. As mentioned earlier, the *modifier digit* and `ModifyFile` can change these defaults. The format in the `ModifyFile` is one of

```
Quench a b c B  /
Quench a b T /
Quench a b c L /
```

Q... must start from the 2nd column.

## 4.2 User own quenching treatment

If the user wants to use another quenching formula, one solution is to do every thing in `userde` of ephook.f. The coding will look like

[7] Coefficient treatment in v9.08 is different from v9.10 or later so the "Log" formula should not be used in v9.08.

```
real(8):: dedx, Cf
....
if( aTrack.p.charge > 1) then
  call epqElossRate(dedx) ! get dE/dx (GeV/(g/cm2)
   ! get Cf from dedx etc
   ! get effective dE by Cf*Move.dE
   !  use it instead of Move.dEeff
endif
```

Caution: If `info` given to `userde` is 1, the particle is dying, or already dead because its energy is < minimum from the birth. In the latter case, unless charge is > 1, `dedx` is undefined. In some case, even charge 0 particle (e.g, very low energy photons) may come there.

### 4.3 Getting the sum of energy deposit and effective energy deposit

The user can use (in `ue1ev of ephook.f`)
`call epqEloss(i, dEt, dEeff)`
to get true energy deposit (`real(4)::dEt`) and effective deposit (`real(4)::dEeff`) for a component number `i`.

Caution: dEeff may be taken to be proportional to emitted light intensity. However, actual light reaching to sensor could be dependent on the emission position. Such a factor is not taken into account in this dEeff. The user must do such business in `userde` using Move.dEeff and position information, etc

## 5 Seeing the minimum energies and quenching coefficients

To have a look at the minimum energies and quenching coefficients set by `ModifyFile` or `AutoEmin` together with the associated component, the user may go to Epics/Util and issue

- `./testCnf4.sh` for Emin
- `./testCnf5.sh` for quench coef.

The usage will be shown by the command.

## 6 New user interface

In some applications, the user may want to know information of particle interactions (what kind of interaction, where it happened etc). This type of interface is available in Cosmos but not in EPICS. If we add such one, all user must modify the existing ephook.f. This fact delayed implementation of such interface. Now, from v9.10, the user could use such interface[8] while those who don't need such one can use old programs without paying attention to, or without being aware of, the new interface at all.

How to do if the user wants to use the interface ?

1. Edit `Epics/epics/Zepcondc.h` and change the last line to read `#define INTINFO`

[8] If the user needs to know only the first interaction of the incident particle, this interface is not needed. See section 11.

2. `make clean;make` in Epics as usual[9]. This may be done once for all unless the user resets the line to `#undef INTINFO`.

3. In the user's application directory, say, `Epics/UserHook/myApps` or in `Epics/UserHook/xxx/myApps` (see 7. below), issue `manageIntInfo.sh`.

   This command dose two things

   - copies `Epics/UserHook/epUI.f` to the current directory.
   - after `#include "../main.f"` or `#include "../../main.f"` in the ephook.f, adds the next lines

     ```
     #include "Zepcondc.h"
     #if defined (INTINFO)
     #include "epUI.f"
     #endif
     ```

4. If the user dose not do anything more, and `make clean;make`, then the application will run as if the user did not do `manageIntInfo.sh`. The overhead by introducing epUI.f is completely negligible.

5. If the user wants to do something when some particle interacts, the user must edit epUI.f.

   - The usage is explained in the epUI.f
   - Important: Suppose a photon interacts and this program unit is called, then if the user dose not give 0 to `info`, this program will not be called for later photon interactions until the next event simulation starts. For other particles, the same is true.
   - The position information is the one at the local coordinate of the current component.

6. For the applications in `UserHook/` supplied by v9.09, `manageIntInfo.sh` has been done. If the user dose this again, nothing will happen.

7. If the user's application is located in a different directory structure than shown above, the equivalent two things as above must be done by the user.

The epGUI subroutine is included in the epUI.f. This is prepared to cope with future demand of new interfaces.

# 7 New volume shapes

♣See a separate manual for octagon, sqTccl, fpolygon, torus, ciecone; some are not new at all.
`http://cosmos.n.kanagawa-u.ac.jp/EPICSHome/NewVol.pdf`

[9]This should be always ok, but if the user has already made the library before changing `Zepcondc.h`, the safest way is to delete the library in `Epics/lib/.../` once and remake the library.

# 8 ♣Updates in v9.13 and v9.131

Rather updates in Cosmos version 7.62.

1. h-A cross-sections.

   These are normalized to the PDG values at 200 GeV. For heavy material (A>170), there is no change (< 0.1%). For A<170, max difference is 2.5 % for light elements such as Be,C,N except for A=16 for which 5.5 % difference is seen. Note: PDG values have been changing with time.

2. A-A cross-sections.

   A parameter "AAXsec" is introduced, which may be given in the param file ($HPARAM part). AAXsec=0 (default) is to use cross-sections having been used so far. AAXsec=1 will use the ones normalized to Shen's cross-section (Nuclear Physics A491 (1989) 130-146) at 5 GeV/n. Normally, AAXsec=1 gives little bit larger cross-sections than AAXsec=0.

If the user wants to move user's routines placed in a directory (say, xxxx) under UserHook, it may be moved to any directory and the **hookIsOutSide.sh** command may be issued in xxxx to accomplishe necessary changes.

If hookIsOutSide.sh has been executed earlier than **manageIntInfo.sh**, the latter does not work well. This was corrected.

# 9 ♣Updates from v9.131

There is no update which might affect the results more than a % level in usual applications. Many are refinement of utility funcitons.

1. So far Seltzer & Berger's bremsstralung cross-section table has been used from $E_e = 5$ keV to 100 MeV (in eleectron kinetic energy)[10].

   The upper limit is now 10 GeV (which is the max table value). For heavy materials, the cross-section difference over 100 MeV is small, while for light materials the change is not negligible. Therefore, for example, the E-M cascade only in He gas might be affected (Fig.1).

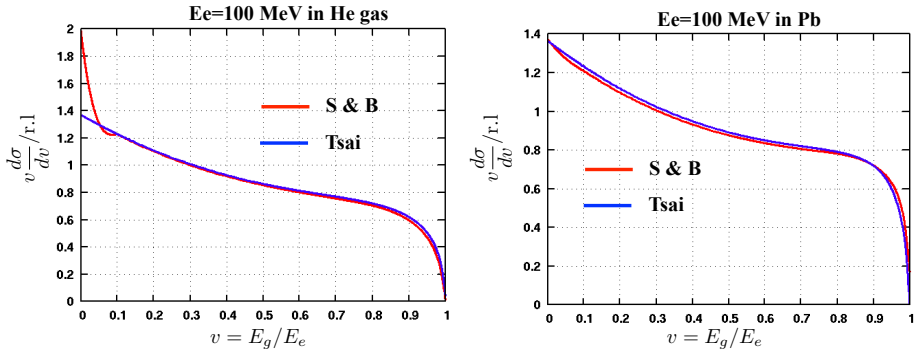


Figure 3: Seltzer & Berger's brems cross-section vs Tsai's one. Left for He gas. Right for Pb. Red line for S & B and blue one for Tsai. Electron energy is 100 MeV.

[10]In the higher energy partial screening region, Tsai's analytical formula has been employed.

A epicsfile parameter "EpartialSC" (0.1 ∼10 GeV; D=10) controls up to which energy the S & B's table is used in actual sampling.

2. There are basically three different bremsstralung cross-sections: Seltzer & Berger's table, Tsai's partial screening formula, and Tsai's complete screening formula [11]. (Besides this, the LPM effect could be imposed). They are used at different enegies but there are small gaps at the connection points.

   Now the differenctial cross-sections are normalized at $v = E_\gamma/E_e = 0.75$. When HowNormBrems=-1, the normalization is performed in such a way that the complete screening cross-section is correct so that Tsai's partial screening cross-section is normalized to that one at some energy where the cross-sections are switched, and then Seltzer & Berger's one is normalized to the normalized partial screening cross-section at 10 GeV. When HowNormBrems=1, S & B is treated as the correct one and the procedure goes inverse way. if HowNormBrems=0, no normalization is performed. The deviation of the normalization factor from 1.0 is normally less than 1 %.

3. The LPM effect so far has been considered only in the complete screening region (above few tens GeV in heavy media). This is enough for many applications where energy deposit is measured.

   However, the effect itself is active at lower energies (e.g, Anthony et al., Phys. Rev. Lett. Vol.75, No.10, 1995).

   Now the effect can be imposed assuming Migdal's formula is applicable (see Appendix A) down to $E_e$ ∼100 MeV in the case of dense media where the supression starts visible for the photon energy of ∼10 keV in log $v$ scale graphs of $v\frac{d\sigma}{dv}$.

   Epicsfile parameters: Besides old "LPMeffect" (t/f. D=t), "Flpm" (>= 1. D=1) is introduced. The minimum energy where the LPM is applied is fixed by $F_{lpm}E_{lpm}$ where $E_{lpm} = \max(0.1, 0.3X0/0.561)$ GeV and X0 is the radiation length of the medium in cm[12].

4. The interaction model, "dpmjet3", needs pre-calculated Glanuber data and they are stored in Data/Media/. So far they could be used for projectiles up to Fe.

   Now the default maximum heavy projectile is Pb.

5. Due to these changes mentioned above, the files in Data/Media/ are now completely different from older ones. If one has media files not listed in Data/Media, they must be recreated.

6. Media file creation: Basically the same as old days.

   - Create a base file in Data/BaseM. In the case of format 2, information for the last line may be obtained by visiting:

     `http://pdg.lbl.gov/2012/AtomicNuclearProperties/index.html`

     and searching a TEXT file. If the medium is not found there, use format 1 without giving the last line. If only some parts are unknown, fill them by -100.
   - Go to Util/Elemag/BremsPair.

[11] Other formulas are also implementable, but currently we don't use them
[12] Don't confuse $E_{lpm}$ with the one in PDB

- Issue ./CreateTab

7. Preparing Glauber files.
   - In any place, issue,
     iniGlauber
     There is an option for the maximum projectile mass (Fe or Pb). Normally, Pb (default) should be selected. If the media have a number of heavy elements, default case takes very much longer time than the Fe case.
   - Two files with .GLB and .inp should be moved to Data/Media/.
8. Drawing Brems/Pair functions and/or testing sampling.
   - Go to Util/Elemag/BremPair [13].
   - Use showBremFunc.sh for drawing brems functions.
   - Use showPairFunc.sh for drawing paircreation functions.
   - Use testBremSamp.sh for brems sampling test.
   - Use testPairSamp.sh for brems sampling test.
9. As new media, Al2024,Al7075,Steel and Au are added.
10. The ingredients of GSO were updated.
11. It's found that PHITS cannot accept $K^{\pm}$ besides $K^0$. This was corrected. They are rather rare at low energies, and hence the effect is very limited.
12. In the PHITS model, knock-on nucleons from a heavy nucleus are almost neutrons. It is said that nucleons from the evaporation mechanism are almost neutrons but it's not clear that the same is true for knock-on nucleons.

    If the number of protons among knock-on nucleons is roughly proportional to the parent $Z/A$, there might be some effect when low energy neutron hits a heavy nucleus.

    To be able to see the effect, a parameter "DoNPadjust" is introduced. It is to be given in the param file(`$HPARAM` part). DoNPadjust=0 (default) is for the PHITS original treatment, DoNPadjust=1 will try to adjust the p/(p+n) ratio be $Z/A$.
13. Config file management. So far many of tasks related to a config file must be performed after going to Util or Util/Geomview. This is inconvenient and now the following commands may be issued in the directory where your target config file is located (the config file may include sub-detectors placed in other directories):

    usenewvol, expandconfig, mkdrawconfig, drawconfig, dispconfigbygeomv, disptracebygeomv

    The list of the commands shown above may be seen by issuing:

    configMenu
14. When the environmental variable are set both for Cosmos and Epics, the FC command in site.config may not work well in some case. This was corrected.
15. In some Linux system, `#!/bin/bash` and `#!/bin/sh` in Scrpt/cppFCPCLinuxIFC64 are not compatible. They are now unified to the former one.

[13] In the directory, there are some obsolete stuffs.

16. $dE/dx$ and target atomic electron brems.

    In the $dE/dx$ calculation of muons, the brems effect by target atomic electrons has been included. Now the same effect can be considered for pi,K,p, too.

    At present we cannot treat it as a stocastic process but only average $< dE/dx >$ is known( D. E. GROOM et al., Muon Stopping Power and Range Atomic Data and Nuclear Data Tables, Vol. 76, No. 2, July 2001). It is dangerous to include the average when the media is not thick enough so that the process can happen many times there.

    So the current resolution is to include it when the material is very thick (say, 1m Fe; this means that the main purpose is to see the muon energy after traversing such a thick medium. In the case of hadrons, they will make hadronic interactions in such a thick medium and considering the effect for them would be almost no meaning.

    The relevant parameter is "TargetElecBrems" in the epicsfile. let's express its last bits as xyz (z is lsb; bit position 0): z is for muon, y for pi,k,p. if each bit and x are on, the effect over $5\sqrt{M_\mu/M}$ GeV is considered ($M$ is the mass of $\mu$, pi,k or p). The default value of the parameter is 0. For thick media, the value of 7 (bit 111) may be given. The effect for 100 GeV muons is order of 3 %.

    If x is off, the effect becomes negligible; it will have meaning when stochastic treatment becomes possible.

17. Two virtual media have been introduced. One is "world" and the other "sp2". (See later).

18. Using "_" in the su-bdetector name is now prohibited.

## 9.1 ♠: Update in Cosmos7.634

1. At high energies, dpmjet3 produces rare particles such as $D_s^+$; they have been simply neglected since their rareness. From 7.634, their decay can be considerd. The relevant parameter is dpmRareDecay[14].

   Default 1 is to treat some of them as older versions: decay of $D^{0,\pm}$, $\Sigma^{0,\pm}$, $\Xi^{0,\pm}$, $\Lambda_c$, $\Omega^-$ is considered (for $\Lambda_c$, only the channel containing muon; 2%) while their collision is treated as a Kaon or proton. Others are forced to decay within dpmjet3 (in older versions, others were neglected).

   0 is to neglect all of such particles.

   2 is to force all of such particles decay within dpmjet3.

   3 is to treat them completely as the older versions.

   No effect will be seen in normal applications.

   The pdg M.C (KF) codes subject to this treatment are:

   $411(D^+)$, $421(D^0)$, $431(D_s^+)$, $441(\eta_c(1S))$, $443(J/\psi(1S))$, $4112(\Sigma_c^0)$, $4122(\Lambda_c^+)$, $4132(\Xi_c^0)$, $4212(\Sigma_c^+)$, $4222(\Sigma_c^{++})$, $4232(\Xi_c^+)$, $3222(\Sigma^+)$, $3212(\Sigma^0)$, $3322(\Xi^0)$, $3312(\Xi^-)$, $3334(\Omega^-)$

[14] It is impossible (?) to force those particles to decay by giving data in dpmjet.inp. So a subroutine "cdpmRareDecay" is added in Cosmos/Particle/Event/Interface/cdpmjet.f

## 9.2 Some details

### 9.2.1 Material of world in a sub-detector

"sp" has been used normally for the material of "world" in sub-detectors. If the world is tight (i.e no gap between the world and contained components), "sp" is good since it is not drawn in default. If the world is not tight, the gap space is treated as 'sp' (almost vacuum). This is good for space experiment simulations. For the ground base experiment, it may be better to use "Air" instead of "sp", though the effect is normally negligible. To replace all "sp" with "Air" is bother.

If we use a new virtual material "world" in stead of "sp" for sub-detector's world, all of them are replaced by the world material of the last world (So if the last one is "H2O", all of sub-detector "world" will become "H2O").

This applies when a sub-detector is simply put in another sub-detector or in the final detector definition, and must be distinguished from the case where a sub-detector is contained by another component explicitly, e.g,

```
#subd abc
1 ...
2 ...
...
n box_w W ...
#end abc

#subd xyz
1 abc ...
2 box W / 1
3 abc ...
4 box W / 3
...
n box_w world ...
#end xyz
```

In this case, it is natural that, if the world of 'abc' is not tight, it is natural to assume that the gap in 'abc' is filled with the medium of the container (W). This has been realized by putting W as the world material of 'abc'. However, if there is another container and its material is Pb, we cannot use 'abc'.

This is inconvenient. If the world is specified as "sp" in this case, the current system will automatically replace it by the container's material. From the point of simulation, this is ok irrespectively of the gap existence.

However, if we display the detector, the gap is drawn even if the gap thickness is 0 (so the inside cannot be seen in default). This is another inconvenience. Therefore, if we are sure that there is no gap, we may use "sp2" in stead of "sp"; then sp2 is not replaced by container's material and eventually replaced by "sp". If we are not sure about the gap existence, it's safe to use "sp".

Recap:

There is clear difference between the cases where a sub-detector is simply put in another sub-detector (or final detector definition) and where a sub-detector is contained by another component. A sub-detector cannot contain another sub-detector or component (instead, simply put them inside).

### 9.3 Parameters summary

Hadronic interactions are managed by Cosmos and hence parameters related them are for Cosmos and better to be placed in the param file. However, some were put in the epicsfile. This is rather confusing both for the user and programing.

Now they are removed from the epicsfile; if they are put in the epicsfile, the program will stop with appropriate messages.

Table 5: Recent new parameters

| variable | to be given in | value | description |
|---|---|---|---|
| ~~JamXs~~ | epicsfile | | not usable now |
| JamXs | param ($HPARAM) | | For Jam interaction model. |
| | | D=0 | only ielastic events are accepted. |
| | | 1 | elastic events are also accepted for non heavy ion projectile. |
| JamFragment | param($HPARAM) | 0 | all spectators are nucleons |
| | | D=1 | heavy fragments may be formed from spectators |
| ~~PhitXs~~ | epicsfile | | not usable now. At present we don't use it. |
| ~~IncGp~~ | epicsfile | | not usable now, instead use next. |
| HowPhotP | param($HPARAM) | | For photo-hadron production |
| | | 0 | no photo-hadron production is considered. |
| | | 1 | use Sofia model (A.Mücke, et al. Comp. Phys. Comm. 124, 290-314) (target is always proton) |
| | | 2 | exp. data at< 2.5GeV. Sofia at > 2.5 GeV |
| | | 3 | Sofia at < 2.5 GeV. At > 2.5 GeV, examine current active model is able to use vector meson ($\rho, \omega, \phi$), pi0, or pi$^{\pm}$ as projectile, in this order, and use one of them as projectile. |
| | | D=4 | Exp. data at < 2.5 GeV. At > 2.5 GeV, the same as in 3. |
| AAXsec | param($HPARAM) | | For AA collision cross-section. |
| | | D=0 | same as older versions |
| | | 1 | cross-section is normalized to Shen's one at 5 GeV/n. 1 gives normally larger $\sigma$ than 0 |
| ♠dpmRareDecay | param($HPARAM) | | Control rare partilcs in dpmjet3 |
| | | D=1 | Those of which decay were treated in older versions (D$^{0,\pm}$, etc) are treated in the same manner. Their collsion is treated as a proton or Kaon. Others are forced to decay in dpmjet3. |
| | | 0 | Neglect all such particles |
| | | 2 | Force all such particles to decay in dpmjet3 |
| | | 3 | Treat them as in the older vesions. |
| Flpm | epicsfile | D=1 | LPM effect is applied when electron energy is >= Flpm*Elpm where Elpm=max(0.1, 0.3X0/0.561) GeV, X0 being r.l in cm. Flpm must be >=1.0 |
| EpartialSC | epicsfile | D=10 | Seltzer & Bergers' numerical brems table is used up to this electron kinetic energy (GeV). must be 1∼10. |

| variable | to be given in | value | description |
|---|---|---|---|
| HowNormBrems | epicsfile | | how to normallzie brems cross-sections. |
| | | D= −1 | complete screening cross-section is taken to be correct. |
| | | 1 | Seltzer & Berger's cross-section is taken to be correct. |
| | | 0 | no normalization is performed. |
| DoNPadjust | parm | D= 0 | Use PHITS default for spectator n/p from target nucleus |
| | | 1 | Adjust n/p ratio so that it is close to target A,Z ratio. |
| TargetElecBrems | epicsfile | | how to treat $dE/dx$ due to target electron's brems effect. should be used if the medium is thick enough. The effect becomes visible when the particle energy $> 5\sqrt{M_\mu/M}$ GeV, where $M$ is the mass of mu, pi, K, or p. |
| | | D=0 | no effect is considered |
| | | bit 0 | If this bit on, effect is considered for muons |
| | | bit 1 | This bit is for pi,K, p |
| | | bit 2 | If this bit is off, only the loss corresponding to restricted energy loss is considered and normally negligible. If the bit is on, average total $dE/dx$ is considered. |

# 10 ♣Recap

### 10.0.1 Case where no world is need in sub-detector

```
In the next example,
#subd  abc
1  ...
2  ...
3  ...
4  box  ..  / ....  / 1 2  3
#end abc
```

the last component (4) contains all other components (1,2,3) and can play a role of a world, and hence world is not needed.

## 10.1 Contain vs Partial Contain

In Fig. 4 left. 'c' is contained by both 'a' and 'b'; some part running off the edge of 'b' is contained by 'a'. By the format with 'NG', 'c' is not recognized by particle 'y', and resembles to the 'partial contain' shown in the right Fig. However, for particle 'x', the running off part is recognized so in this respect, it is not a 'partial contain'. The correct format is the one with 'OK'.

In the 'partial contain' case in the right figure, 'c' is partially contained by 'b' so that the part overflowed from 'b' is treated as non exisistent; the format 'a /c' must not be used. 'c' contains 'd' and if some part of 'd' overflows from 'b', it is regarded

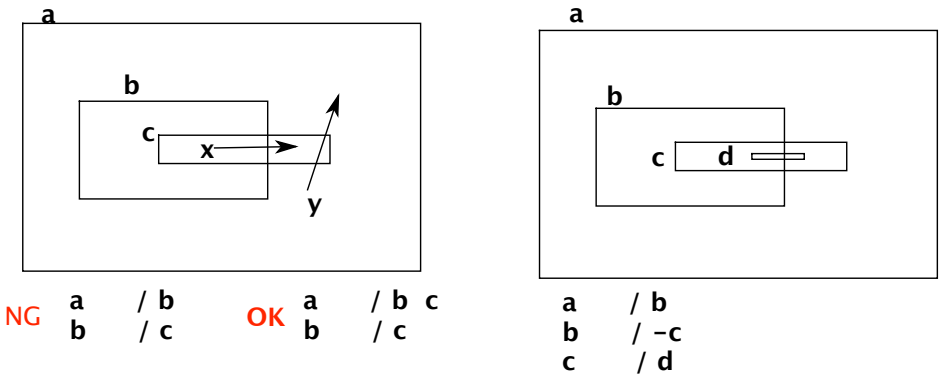


Figure 4: contain and partial contain

as non existent, too. At present, however, non existent part is drawn by the present system.

# 11 Inquiry and other useful subroutines

Here we list inquiry subroutines which may be needed by the user, irrespectively of old or new.

Table 6: Inquiry subroutines

| subroutine name | description |
|---|---|
| epqversion | Get version number of Cosmos and EPICS.<br>Usage:<br>`character(8)::cosv` ! Comos version (say 7.99)<br>`character(8)::epiv` ! EPICS version<br>`call epqversion(cosv, epiv)`<br>Caution: Environmental variable COSMOSTOP and EPICSTOP must have relevant values. |
| epqncp | Get the total number of components.<br>Usage:<br>`integer::ncomp`<br>`call epqncp(ncomp)` |
| epqevn | Get current event number<br>Usage:<br>`integer::eventn`<br>`call epqevn(eventn)`<br>The event number will be updated after `ue1ev` is called. |
| epqinc | Get the incident particle track information<br>Usage:<br>`#include "ZepTrack.h"`<br>`record/eTrack/ aTrack`<br>`call epqinc(aTrack)`<br>aTrack.p.code etc are explained in userde or userbd. E.g, the component number of the incident track is aTrack.cn and its position is aTrack.pos (local coordinate). To get the world coordinate, the user must use `epl2w` (see Table 7)<br>Caution: If multiple particles are incident, only the first one is obtained. |
| epqFirstI | Get the first interaction point of the incident particle<br>Usage:<br>`#include "ZepPos.h"`<br>This is not needed if `ZepTrack.h` is used for `epqinc` in the same subroutine.<br>`record/epPos/firstpos`<br>`call epqFirstI(firstpos)`<br>firstpos.x, firstpos.y, firstpos.z are the interaction point in the **world coordinate**.<br>Caution: Except for the electron, the knock-on and elastic collision are not regarded as interaction. A large negative value (-1000000.0) of `firstpos` indicates no interaction happened. |

| subroutine name | description |
|---|---|
| epqFirstM | Get the media information in which the incident made the first interaction.<br>Usage:<br>`#include "Zmedia.h"`<br>`record /epmedia/ firstM`<br>`call epqFirstM(firstM)`<br>Some of the ingredients:<br>`firstM.A`: real(8) ! Average mass number of the media.<br>`firstM.Z`: real(8) ! Average charge of the media.<br>`firstM.name`: character(8) ! media name.<br>`firstM.colElem`: integer<br>The element # within the media at which the first collision took place. For e/$\gamma$/$\mu$, this will be 0 if the number of elements in the media is $> 1$, and the **variables below are undefined**.<br>`firstM.colA`: integer<br>Mass number of the nucleus at which the collision took place.<br>`firstM.colZ`: integer<br>Charge number of the nucleus at which the collision took place.<br>`firatM.colXs`: real(8)<br>The inelastic cross-section (mb) of the target. |
| epqFirstP | Get process name of the first interaction.<br>Usage:<br>`character(8):: proc`<br>`call epqFirstP(proc)`<br>If no interaction happened, `proc` will be ' '. Hadronic collision is 'coll', brems 'brem', pair creation 'pair', Compton 'comp'; more details are seen in Epics/UserHook/epUI.f. |
| epqCn2Media | Get media information of a given component number.<br>Usage:<br>`#include "Zmedia.h"`<br>`integer:: compn` ! give some number<br>`record /epmedia/ mediax`<br>`compn= ...`<br>`call epqCn2Media(compn, mediax)`<br>If `compn` is wrong, stop will happen. The media name is in `mediax.name` (see, `epqmat`), `mediax.Z` the average charge of the media, etc. |
| epqmat | Get media name of a given component number.<br>Usage:<br>`integer:: compn` ! give some number<br>`character(8)::name` ! If `compn` is wrong, stop will happen.<br>`compn= ...`<br>`call epqmat(compn, mat)`<br>Output `mat` will be such as `Pb, SCIN` etc. |

| subroutine name | description |
|---|---|
| epqstruc | Get component structure (box etc) of a given component number.<br>Usage:<br>`integer:: compn` ! give some number<br>`character(12)::struc` ! If `compn` is wrong, stop will happen.<br>`compn= ...`<br>`call epqstruc(compn, struc)`<br>Output `struc` will be such as `box, ciecone` etc. If the length is shorter than actual one, the tail part will be lost. |
| epqSubdName | Get sub-detector name to which a given component number belongs.<br>Usage:<br>`integer::compn` ! give some number<br>`character(16)::name`<br>If `compn` dose not belong to a sub-detector, name will be ' ' .<br>`compn= ...`<br>`call epqSubdName(compn, name)` |
| epqmatrhoc | Get media name (e.g, Air*1.03) of a given component number.<br>Usage:<br>`integer:: compn` ! give some number<br>`character(20)::name`<br>Output will be `Air*1.0032` etc. `Air` is equivalent to `Air*1`. If `compn` is wrong, stop will happen.<br>`real(4)::rhoc` ! output.<br>The relative density of the component to the default.<br>`integer::lc` ! output. Length of the name content.<br>`compn= ...`<br>`call epqmatrhoc(compn, name, lc, rhoc)` |
| epqCount | Get digit information for calling `userbd` and `userde` of a given component number.<br>Usage:<br>`integer::compn` ! give a component number<br>`integer::countio` ! digit for `userbd`<br>`integer::countde` ! digit for `userde`<br>`compn= ...`<br>`call epqCount(compn, countio, countde)`<br>Wrong `compn` will result in a stop. Suppose a component description line like<br>3 box Pb c de / ....<br>here "c" is the digit used to call userbd and "de" the one used to call userde (and routines for light transport). We call the "c" part **countIO** (since it is to specify whether `userbd` is to be called when a particle enters In or goes Out of a component), and the "de" part **countDE** (since it is to specify whether `userde` is to be called when a particle Deposits Energy in a component). |

| subroutine name | description |
|---|---|
| epLightUnpack CountDE | Decompose `countDE` (see above). Decomposition may be needed when it contains information for light generation and transport.<br>`integer(2)::info` ! NOT integer<br>`integer::d` ! digit for energy deposit count<br>`integer::mn` ! digit for flle for Light or sensor<br>`integer::B` ! digit for light generation/transport<br>`info = countde` ! convert to 2 byte integer<br>`call epLightUnpackCountDE(info, d, mn, B)`<br>The input `info` is decomposed into `d, mn, B`. Details will be explained in the Light Transport section. |
| epqvolatr | Get the volume attribute of a given component number.<br>Usage:<br>`integer::compn` ! give a component number<br>`integer::na` ! Output. number of attributes obtained.<br>`real(8)::vol(x)` ! Output.<br>x must be $\geq$ `na` which is dependent on the volume (say, for box 3, cyl 2, octagon 4 ...).<br>`compn= ...`<br>`call epqvolatr(n, na, vol)`<br>Example: for a box, vol(1), vol(2), vo(3) will be `a,b,c` of the canonical form. If `compn` is wrong , stop will happen. |
| epqcmpdircos | Get the direction cosines of a given component number.<br>Usage:<br>`integer::compn` ! give a component number<br>`real(8)::dir(9)` ! Output.<br>`compn= ...`<br>`call epqcmpdircos(compn, dir)`<br>The direction cosines of the component, showing how the canonical form is rotated. Suppose a canonical box surrounding the component. The rotation is expressed by the direction cosines of rotated canonical box's `a, b, c`: `dir(1:3)` are for `a`, `dir(4:6)` for `b` and `dir(7:9)` for `c`. If the component is not rotated `dir(1:9) =(1,0,0, 0,1,0, 0,0,1)`. If `compn` is wrong , stop will happen. |
| epqOrig | Get the origin coordinate value of a given component number.<br>Usage:<br>`#include "ZepPos.h"`<br>`record /epPos/ orgin`<br>`integer::compn` ! give a component number<br>`compn= ...`<br>`call epqorg(compn, origin)`<br>The value of `orign.x, orign.y, orign.z` is the origin of the component in the **world** coordinate. That is, the origin of the component in its canonical form is shifted by this amount. If `compn` is wrong , stop will hapen. |

| subroutine name | description |
|---|---|
| epqElossRate | Get $dE/dx$ of the current particle<br>Usage:<br>`real(8)::dedx` ! GeV/(g/cm$^2$)<br>`call epqElossRate(dedx)`<br>To be used in userde. See caution in 4.2. |

Table 7: Other subroutines

| subroutine name | description |
|---|---|
| epl2w | Convert position in the local coordinate into the world coordinate<br>Usage:<br>`#include "ZepPos.h"`<br>`integer::cn` ! input. component number<br>`record /epPos/ posl` ! input.<br>The position `posl` in the local coordinate of the component specified by the component number `cn`.<br>`record /epPos/ posw` ! output. world coordinate position<br>`call epl2w(cn, posl, posw)` |
| epw2l | Inverse of `epl2w`<br>Usage:<br>`#include "ZepPos.h"`<br>`integer::cn` ! input. component number<br>`record /epPos/ posw` ! input. world coordinate position<br>`record /epPos/ posl` ! output. local coordinate position<br>`call epw2l(cn, posw, posl)` |
| epl2wd | Convert 3 direction cosines in the local coordinate into the world coordinate<br>Usage:<br>`#include "ZepDirec.h"`<br>`integer::cn` ! input. component number<br>`record /epDirec/ dirl` ! input.<br>Direction cosines `dirl` in the local coordinate of the component specified by the component number `cn`. `dirl=(dirl.x, dirl.y, dirl.z)`.<br>`record /epDirec/ dirw` ! output. world coord. dir. cos.<br>`call epl2wd(cn, dirl, dirw)` |
| epw2ld | Inverse of epl2wd.<br>Usage:<br>`#include "ZepDirec.h"`<br>`integer::cn` ! input. component number<br>`record /epDirec/ dirw` ! input.<br>Direction cosines `dirw` in the world coordinate.<br>`record /epDirec/ dirl` ! output. local coord. dir. cos.<br>`call epw2ld(cn, dirw, dirl)` |

| subroutine name | description |
|---|---|
| cgetfname | Convert special characters in a string to create a new string.<br>Usage:<br>`character(x):: fin` ! input<br>`character(y):: fout` ! output<br>`fin='...'`<br>`call cgetfname(fin, fout)` !<br>x,y must be some number.<br>All of % #1 #2 # `@` $ in `fin` are treated as follows.<br>1) `#1` is replaced by the initial seed of the random number (1st one of the two).<br>2) `#2` is replaced by the initial seed of the random number (2nd one of the two).<br>3) `#` (not followed by 1 nor 2) is replaced by the unix process number.<br>4) `%` is replaced by YYMMDDHHMMSS (year month day hour minut second of the time).<br>5) `@` is replaced by the hostname (dropping domain name, if any).<br>6) `$` assumes it is followed by an environmental variable, and is replaced by its value. Three types can be recognizable: for instance, `$USER` (not followed by any character), `$USER/` (followed by /) `$(USER)` (always ok). In either case, it may be preceded by any character. (This one is usable from Cosmos7.59). |
| copenf | Open an existing sequential ascii file. Special characters in the file name is treated by cgetfname.<br>Usage:<br>`integer::ionum` ! input<br>`integer::icon` ! output<br>`call copenf(ionum, filepath, icon)`<br>where `filepath` is a character string defined by, say, character(60)::filepath and contains a string showing the path to an existing file. `ionum` is the logical file unit number. `icon` =0: ok `iocn` !=0: could not be opened. |
| copenfw | Open an sequential ascii file for writing. It may not exist. Special character treatment is the same as copenf.<br>Usage:<br>`integer::ionum` ! input<br>`integer::icon` ! output<br>`call copenfw(ionum, filepath, icon)` ! icon =0: ok. else ng |
| rndc | Uniform random number in (0,1).<br>Usage:<br>`real(8)::u` ! output ($0 < u < 1.0$)<br>`call rndc(u)` !<br>The same random number generator as used in Cosmos/EPICS. 0 and 1.0 are excluded.<br>There are two other generators and the third one is not used in Cosmos/EPICS. See for details in Cosmos/KKlib/rnd.f |

| subroutine name | description |
|---|---|
| Others | Other random number generators are available. See the following:<br>(Those in Cosmos/KKlib)<br>kgauss.f : Gaussian random number.<br>kbetar.f : Random numbers with density of the beta function<br>kbinom.f: Binomial random number.<br>kcosn.f: cos and sin of uniform random number in $(0, 2\pi)$.<br>knbino.f: Negative binomial random number.<br>kpoisn.f Poisson random number.<br>kampLin.f Random variable with density $(a + bx)dx$<br>ksampPEang.f: Random variable with density $(1-x^2)/(a-x)^4 dx$. Related to electron angle at photo-electric effect.<br>ksampPw.f: Random variable from a function consisting of many power functions.<br>ksampRSA.f: Random sampling of $\cos\theta$ from $(1 + \cos^2\theta)d\cos\theta$<br>ksbwig.f: Random sampling from the Breight-Wigner distribution.<br>ksgamd.f: Sampling from the gamma distribution, $(x/a)^s \exp(-x/a)/\Gamma(s + 1)d(x/a)$<br>ksplandau.f: Sampling from a psuedo-Landu distribution: $\exp(-(y + \exp(-y))/2)dy$ where $y = (x - b)/c$.<br>csampAF.f90: (in Cosmos/Module). Sampling from an arbitrary function specified by a numerical table. (see also ksampAF.f in Cosmos/KKlib/)<br>**Those in Epics/prog/KKlib:**<br>ksbeta.f: ksmpintbetaf: similar to kbetar.f<br>ksx2.f: Random sampling from the $\chi^2$ distribution. |

# 12 Other updates and input parameters

- **Use of environmenal variables.** As described in Table 7, for the file name, we can use environmental variables.

- **Multiple scattering treatment.** The parameter `Molier` in the `epicsfile` now takes integer values rather than "t" or "f", although the user can still use t or f; they are mapped to 0 or 1 by

  f→ 0: This specifies the Gaussian multiple scattering.

  t→ 1: This specifies the Molière multiple scattering.

  The new possible value is 2. If 2 is given to `Molier`, Molière's multiple scattering formula is used, but it's implementation is completely different and more rigorous than `Molier=1`. It also includes Bethe's prescription[15] to overcome the small angle approximation assumed in original Molière's theory. (Goudsmit and Saunderson's scattering formula[16] which does not use small angle approximation is well reflected in Bethe's prescription). However, results by `Molier=2` are (statistically) completely the same as `Molier=1` for cascade showers. The difference appears in the cpu time which is 1.6 times longer for `Molier=2` than `Molier=1`. Therefore, the user may use the default, `Molier=1`.

[15] Phys. Rev. Vol.15 (1953) 1256.
[16] Phys. Rev. Vol.57(1940)24, 58(1940)36.

- **Automatic disk space allocation.** In older versions than 9.08, if the user inputs a large number of particles as the incident (using "+primary" file notation), the particle stack area could overflow during particle tracking; the user had to specify a disk file in the +primary file.

  In case of light tracking, the number of light photons becomes huge. Allocating more memory for the stack is not a good solution.

  Now, if the stack area lacks, EPICS automatically creates "scratch disk file" and the unix system deletes it at the end of job (even if the job ab-ends).

  The relevant parameters could be written in `sepicsfile`

`StackDiskFile 'scratch' /`
  The default "scratch" does not mean the file name but it is a scratch file created by the system with some system-determined file name; the path to the file is fixed by the system;`ifort` will create it in `/tmp/$USER/` with the name something like fortVSXGZg. The file will be deleted at end of job (even if abnormal end).
  **Caution**:
  If a large number of jobs is submitted in a distributed system (e.g, pc clusters), depending on the system, many scratch files could be created in a non-local disk (say, in the NFS mounted home) to which network access is needed, then the jobs will almost kill the whole system due to overwhelming net work access. In such a case, the user must specify the path explicitly which does not require the network access. e.g, `/tmp/$USER/stackdisk#`. It should be noted, the user given file will be deleted only if the job ends normally, otherwise the user must delete it.
  For the scratch files, logical device number, 13 and 16, will be used in default.

- In Cosmos, some awkward behaviors in low energy interaction models were absorbed.

# 13 Interaction models

phits must be used with jam or nucrin or dpmjet3

# 14 Warnings

- Don't use InputP='fix' which requires Xinp etc. Instead, use InputP='u+z' etc with Xrange=... etc.

- Don't put the incident exactly on the boundary of two consecutive components. EPICS will be buffaloed when judging the component the particle belongs to.

## 15 Light transportation

# Appendices

## A Small modifcation of the LPM formula

If we use Migdal's prescription straightforward way, there appears an unnatural bump especially for low energy electrons (Fig.5 left) just below the "starting" photon energy ($v = X_c = 1/(1 + const/E_e)$), so we increased "const" value 2 times[17]. Though, this modification would not be detected in the cascade simulation.

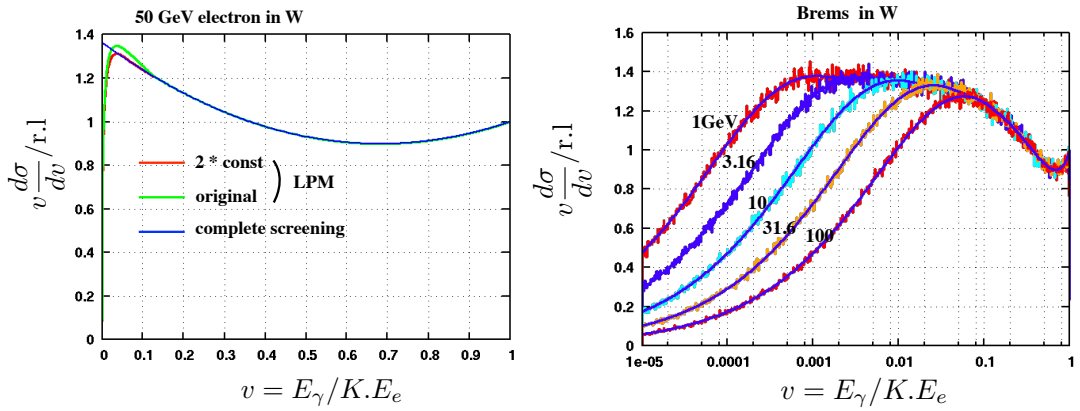


Figure 5: Left. Blue line: complete screening case. When LPM suppression is applied, the cross-section should be less than this line, but actually we get the green line by the Migdal's formula. By adjusting the constant, we get the red line. Right. Some examples of sampling results overlaied with cross-sections with the LPM brems in W. Terr-Michaerian's effect which appears at very small $v$ is neglected.

[17] Util/Elemag/BremPair/epBrgeneric.f