A stable interface to read and write IAEA phase-space files in Geant4

M. A. Cortés-Giraldo[1], R. Capote[2], J. M. Quesada[1]

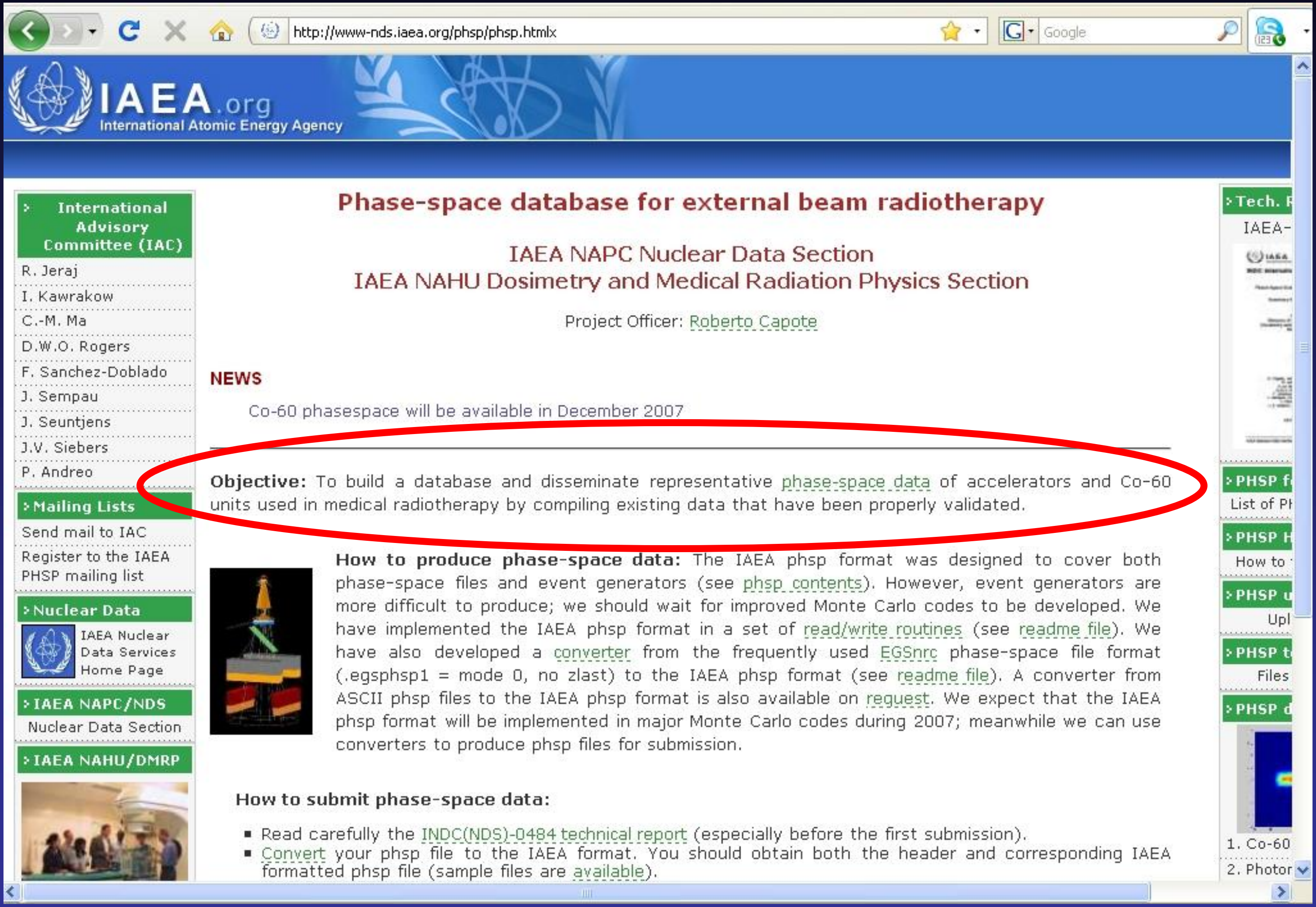
[1]Dep. Física Atómica, Molecular y Nuclear, University of Sevilla, Spain.
[2]Nuclear Data Section, IAEA Headquarters, Austria.

14th Geant4 Collaboration and Users' Workshop
LNS-INFN (Catania, Italy)

October 21st, 2009

Contents

The IAEA Nuclear Data Section Project



Phase-space database for external beam radiotherapy

IAEA NAPC Nuclear Data Section
IAEA NAHU Dosimetry and Medical Radiation Physics Section

Project Officer: Roberto Capote

NEWS

Co-60 phasespace will be available in December 2007

Objective: To build a database and disseminate representative phase-space data of accelerators and Co-60 units used in medical radiotherapy by compiling existing data that have been properly validated.

How to produce phase-space data: The IAEA phsp format was designed to cover both phase-space files and event generators (see phsp contents). However, event generators are more difficult to produce; we should wait for improved Monte Carlo codes to be developed. We have implemented the IAEA phsp format in a set of read/write routines (see readme file). We have also developed a converter from the frequently used EGSnrc phase-space file format (.egsphsp1 = mode 0, no zlast) to the IAEA phsp format (see readme file). A converter from ASCII phsp files to the IAEA phsp format is also available on request. We expect that the IAEA phsp format will be implemented in major Monte Carlo codes during 2007; meanwhile we can use converters to produce phsp files for submission.

How to submit phase-space data:

- Read carefully the INDC(NDS)-0484 technical report (especially before the first submission).
- Convert your phsp file to the IAEA format. You should obtain both the header and corresponding IAEA formatted phsp file (sample files are available).

(http://www-nds.iaea.org/phsp)

The IAEA Nuclear Data Section Project

- **The IAEA format:**
  - The IAEA has created a standardized format to unify files made by different codes.
  - The complete IAEA format consists of two files:
    - `*.IAEAphsp`: binary file where all the positions, momenta and extra-variables are stored.
    - `*.IAEAheader`: ASCII file with miscellaneous information (statistical information, references, variables stored...)
  - In addition, routines to convert EGSnrc files to this format are provided.
  - More information at http://www-nds.iaea.org/phsp

IAEA phase-space file interface in Geant4

- A stable interface to read and write IAEA phase-space files has been developed to be used in Geant4 applications.
- "Stable" means that this interface is not affected by internal changes in the IAEA format that may happen in the future.

- The files involved are:
  - The IAEA routines published on their web site.
  - Our new classes:
    - *G4IAEAphspReader*
      - Derived from *G4VPrimaryGenerator*.
    - *G4IAEAphspWriter*
      - Just a singleton class.
  - Optional class for analysis with ROOT.

IAEA phase-space file interface in Geant4

- **IAEA Routines:**

`iaea_config.h`: declares types of variables.

`iaea_header.h(.cc)`: defines a ‘struct’ which manages the *.IAEAheader file.

`iaea_phsp.h(.cc)`: defines all the methods to get/store the information from/to the IAEA phase-space files.

`iaea_record.h(.cc)`: this is an structure that defines the format used to store the information.

`utilities.h(.cc)`: miscellaneous definitions and functions.

Available at http://www-nds.iaea.org/phsp

G4IAEAphspWriter class

- **Writer class properties:**
  - Singleton. A messenger class can be associated easily.
  - Extra integer variable of type "incremental history number" is stored by default for statistical purposes.
  - Compatible with executions composed by several runs.
- **To use it:**
  - Three *user action* classes are needed: *UserRunAction*, *UserEventAction* and *UserSteppingAction*.
  - Call *SetZStop(double)* method for each phase space plane definition in *UserRunAction* constructor.
  - *BeginOfRunAction()*, *EndOfRunAction()*, *BeginOfEventAction()* and *UserSteppingAcion()* methods must be invoked in their suitable user action class.

G4IAEAphspReader class

- **Reader class properties:**
  - Particles sharing the same original history are created in the same event in Geant4 to do statistical analysis properly.
  - Therefore, an event in the Geant4 simulation corresponds to a complete history and not only to a given particle.
  - Options provided for the user:
    - Divide the phase-space file in chunks (parallel runs).
      - `SetTotalChunks(G4int)`
    - Choose a certain chunk.
      - `SetChunk(G4int)`
    - Particle recycling is considered as well.
      - `SetTimesRecycled(G4int)`

G4IAEAphspReader class

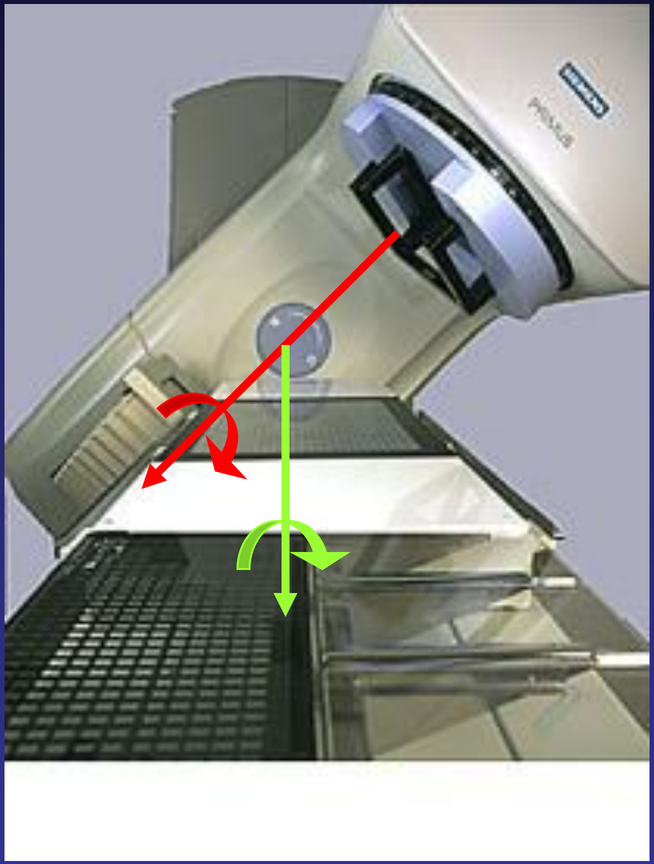
- **Spatial transformations:**
  - In global frame can be done using these methods:

```
SetGlobalPhspTranslation(const G4ThreeVector &);
SetRotationX(G4double alpha);
SetRotationY(G4double beta);
SetRotationZ(G4double gamma);
SetRotationOrder(G4int order);
```

  - `order` is a 3-digit integer number which combines 1, 2 and 3, without repetitions, meaning 1 = X, 2 = Y and 3 = Z axis.
    For example: `order = 132` means first rotate around X, and second around Z axis.

G4IAEAphspReader class

- **Isocentric rotations:**

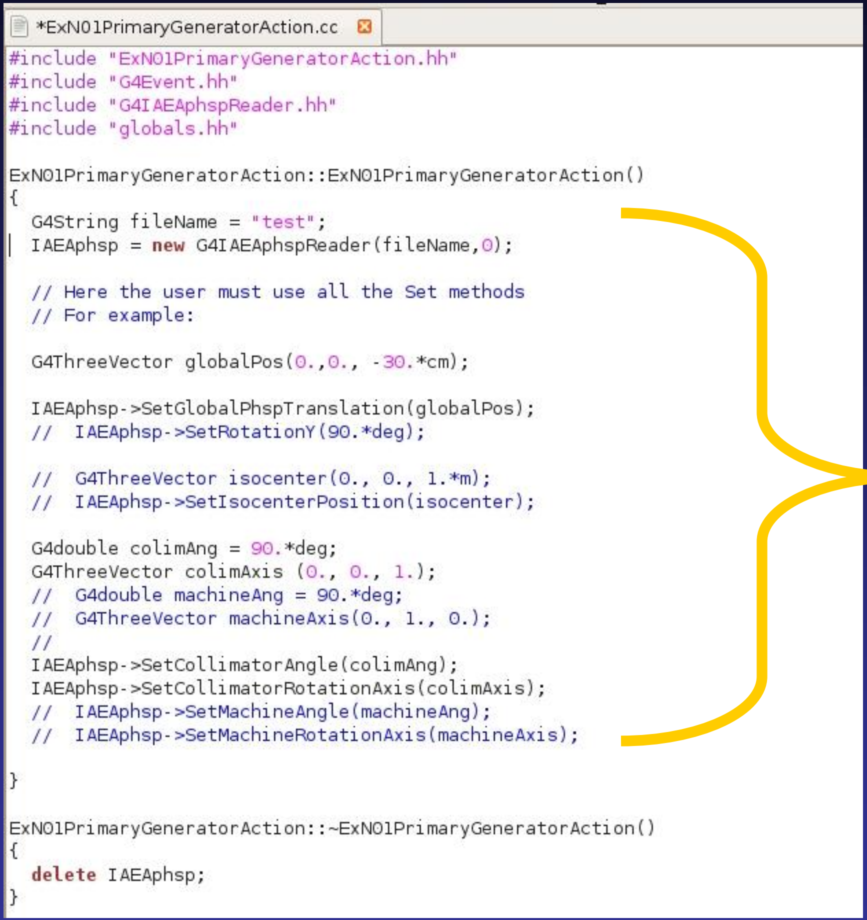


```
SetIsocenterPosition(const
G4ThreeVector & );

SetCollimatorRotationAxis(const
G4ThreeVector & );

SetCollimatorAngle( G4double );

SetMachineRotationAxis(const
G4ThreeVector & );

SetMachineAngle( G4double );
```

  – Configured to rotate the collimator first.

G4IAEAphspReader class

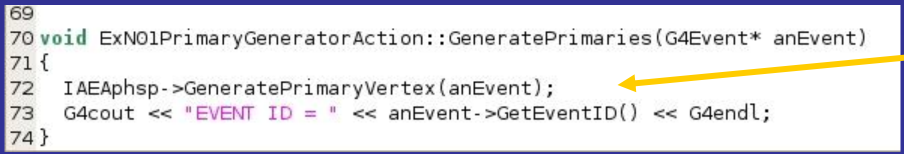
- **How to use the reader class:**



```
#include "ExN01PrimaryGeneratorAction.hh"
#include "G4Event.hh"
#include "G4IAEAphspReader.hh"
#include "globals.hh"

ExN01PrimaryGeneratorAction::ExN01PrimaryGeneratorAction()
{
  G4String fileName = "test";
  IAEAphsp = new G4IAEAphspReader(fileName,0);

  // Here the user must use all the Set methods
  // For example:

  G4ThreeVector globalPos(0.,0., -30.*cm);

  IAEAphsp->SetGlobalPhspTranslation(globalPos);
  //  IAEAphsp->SetRotationY(90.*deg);

  //  G4ThreeVector isocenter(0., 0., 1.*m);
  //  IAEAphsp->SetIsocenterPosition(isocenter);

  G4double colimAng = 90.*deg;
  G4ThreeVector colimAxis (0., 0., 1.);
  //  G4double machineAng = 90.*deg;
  //  G4ThreeVector machineAxis(0., 1., 0.);
  //
  IAEAphsp->SetCollimatorAngle(colimAng);
  IAEAphsp->SetCollimatorRotationAxis(colimAxis);
  //  IAEAphsp->SetMachineAngle(machineAng);
  //  IAEAphsp->SetMachineRotationAxis(machineAxis);

}

ExN01PrimaryGeneratorAction::~ExN01PrimaryGeneratorAction()
{
  delete IAEAphsp;
}
```

In UserPrimaryGeneratorAction constructor the G4IAEAphspReader* pointer must be created, and all the 'Set' methods the user needs also must be invoked.

'Set' methods can be associated to a messenger class as well.

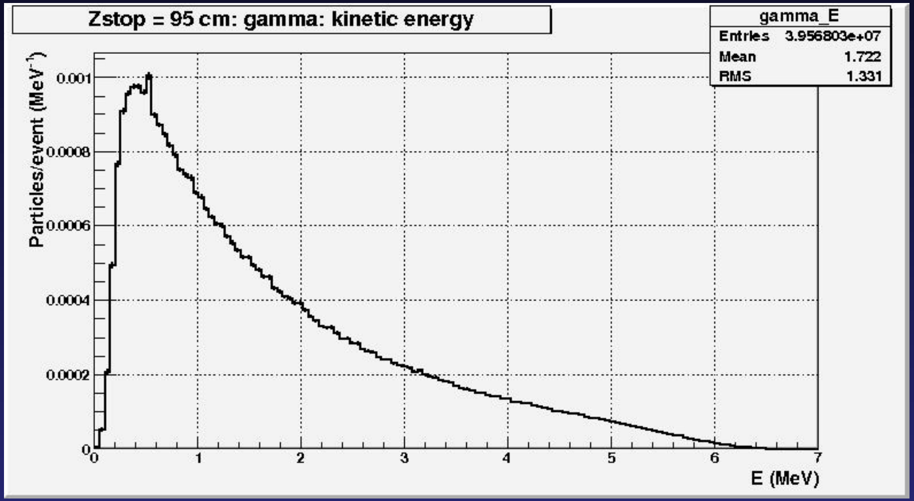


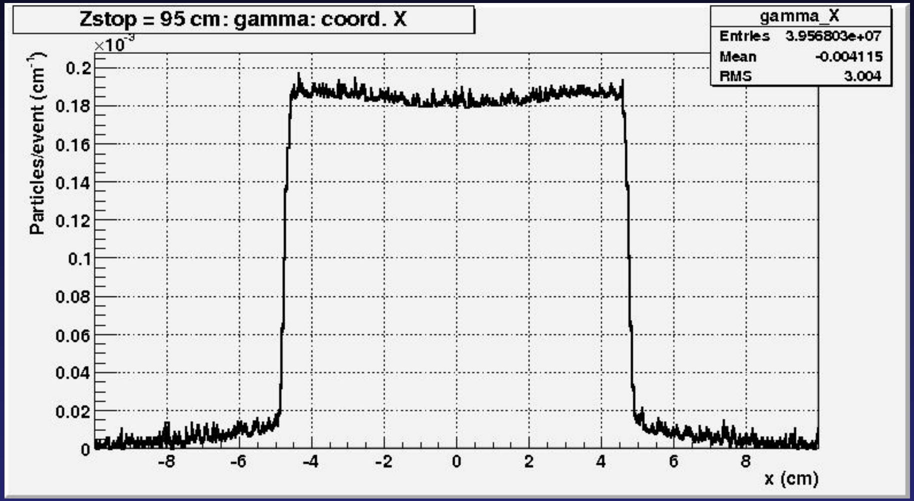
```

void ExN01PrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{
  IAEAphsp->GeneratePrimaryVertex(anEvent);
  G4cout << "EVENT ID = " << anEvent->GetEventID() << G4endl;
}
```

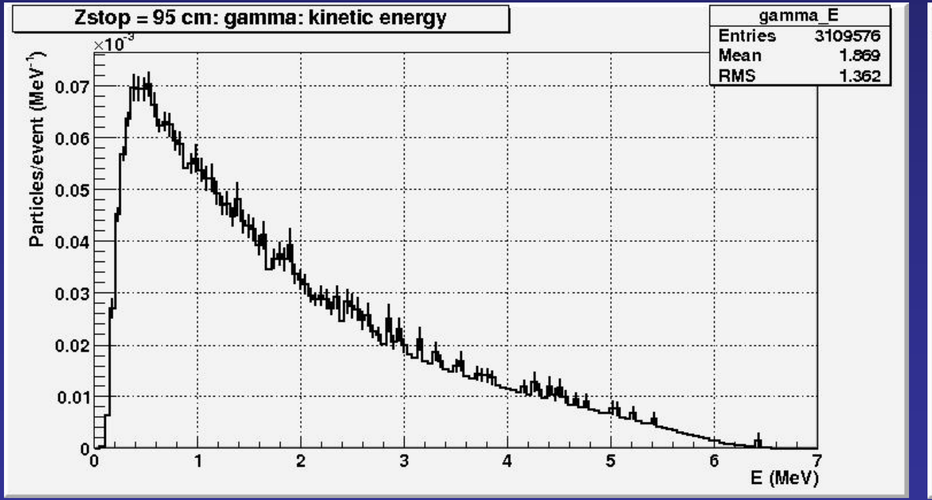
In `GeneratePrimaries` method the user only have to invoke `GeneratePrimaryVertex.`

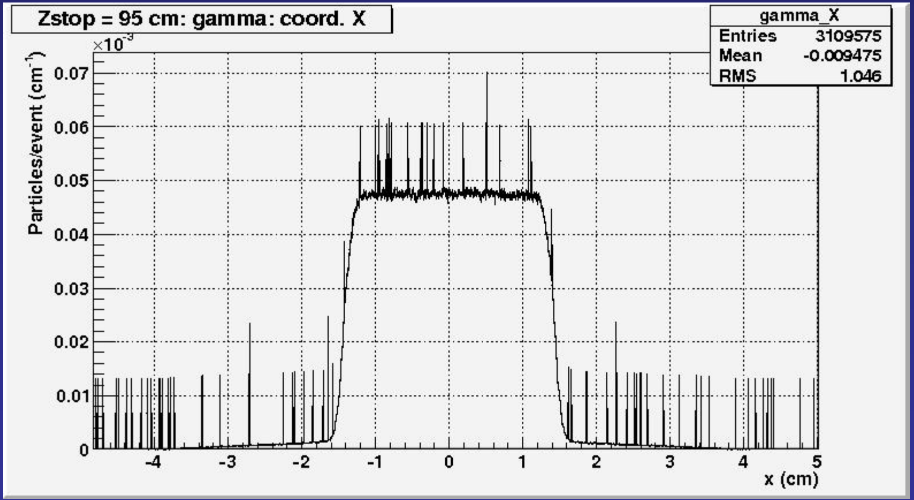
IAEA phase-space file interface in Geant4

- **Results:**





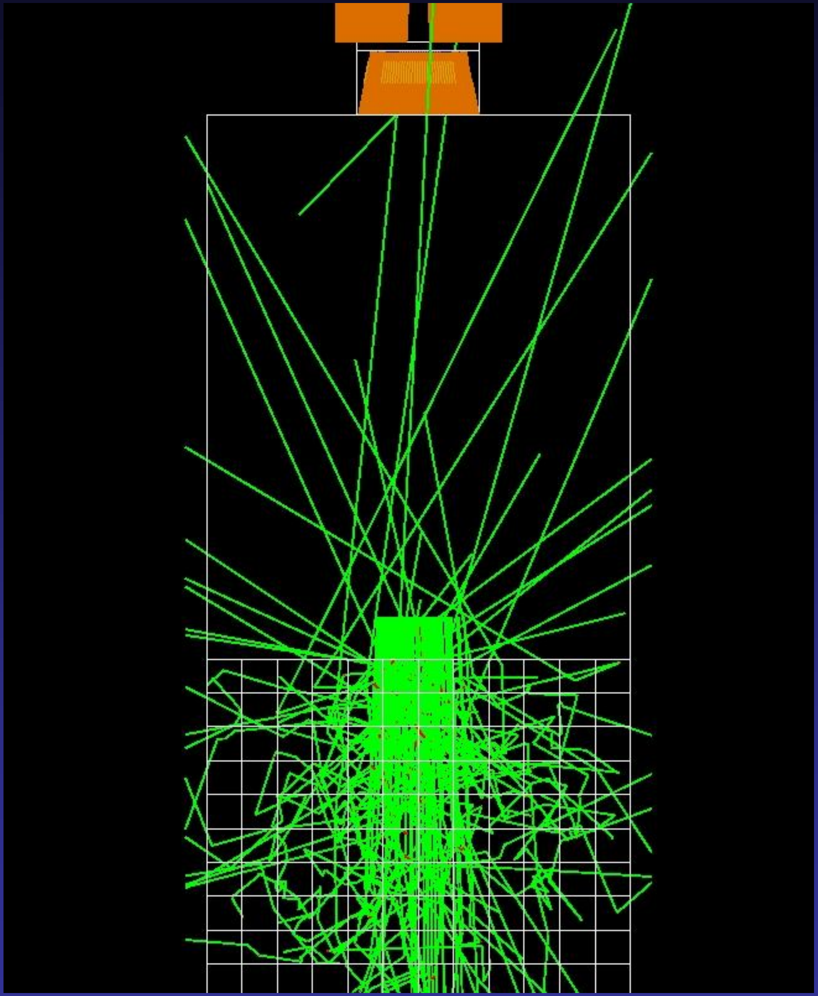




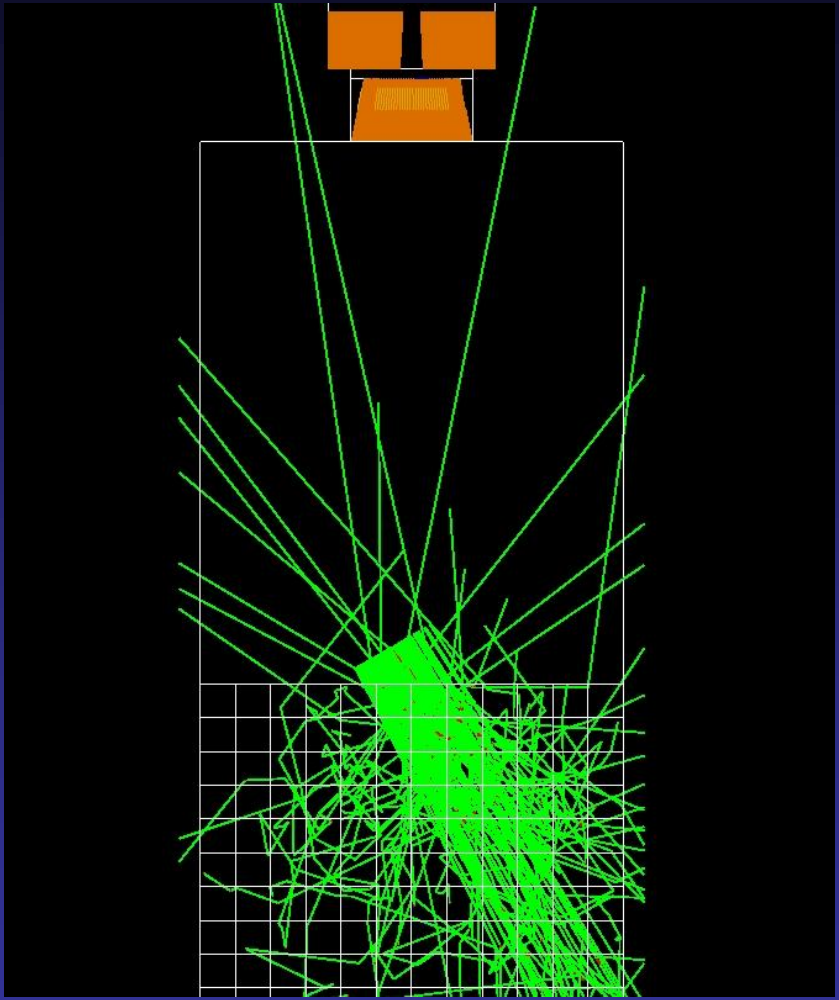
Phase space files taken from the IAEA database.

IAEA phase-space file interface in Geant4

- **Results:**



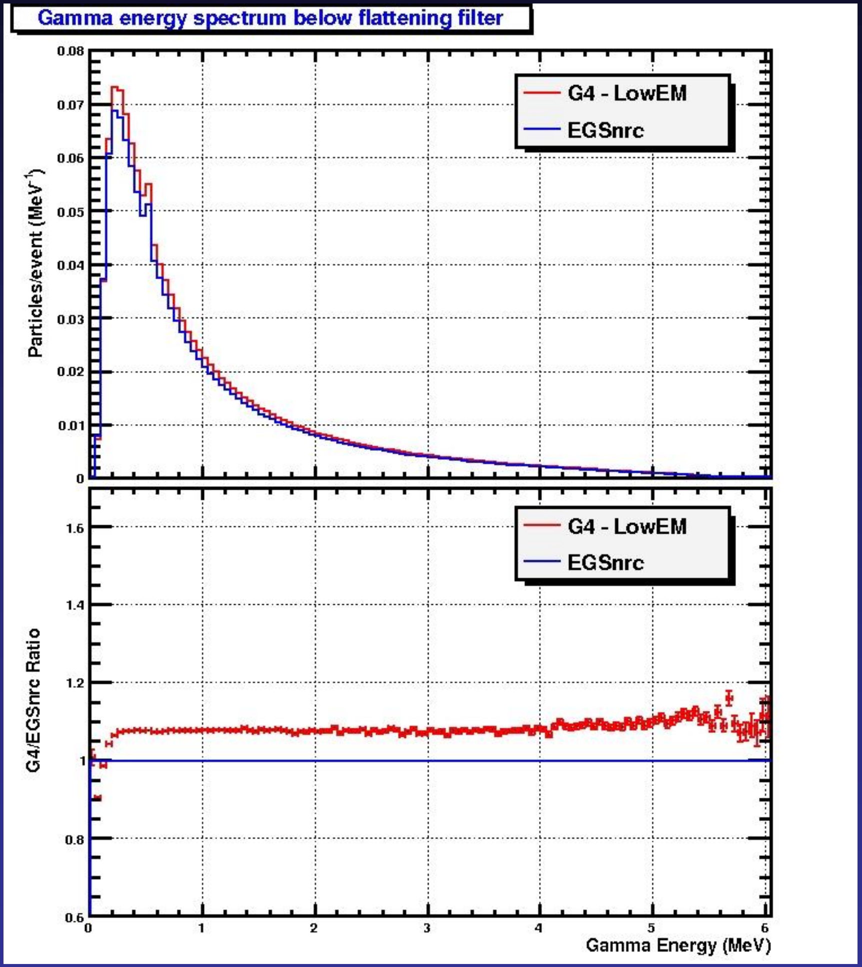
No rotations

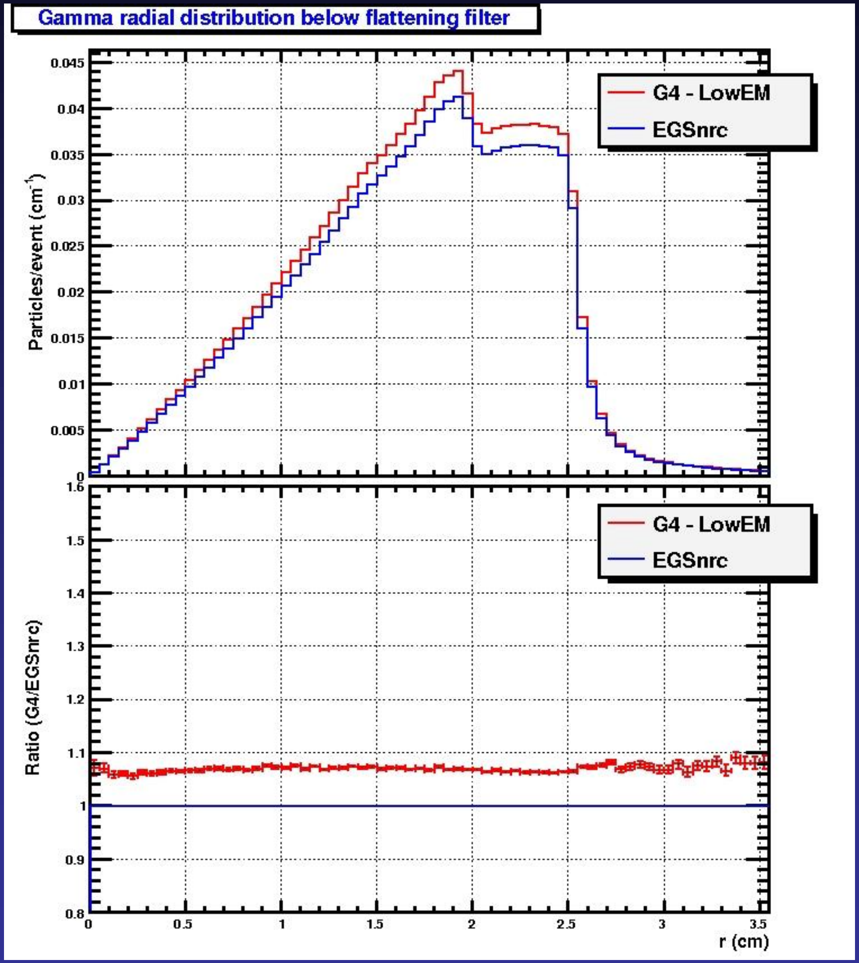


Gantry rotated 30 deg

Geant4 Simulation of a Siemens Primus Linac

- **Results. EGS vs. Geant4 PSF comparison:**





To be repeated with the new LowEnegy EM classes!

Summary and conclusions

- An interface to read and write phase space files in IAEA format has been developed.
  - Easy to use and stable against future changes in the IAEA code.
  - Allows the possibility of creating associated messenger classes to modify the data members through a macro file.
  - Respects the correlations between particles, so it allows the user to do a proper statistical analysis.
  - Possibility of dividing the phase-space files into chunks to perform ‘parallel runs’.

- Publication about to be submitted.
- Available to the community in short time.

And that’s all...

THANKS FOR YOUR ATTENTION