AGATA geant4 simulations for AGATA@Ganil: Features and examples

J. Ljungvall

September 17, 2013

1 Introduction

This is a as-short-as-possible document to explain features available in the geant4 AGATA simulation package found here. All code and examples have been tested using **geant4.9.6.p01**. The idea is to provide the spokespersons of the future AGATA@Ganil campaign with a good idea of the tools available to propose and plan the experiments. Some basic knowledge of the AGATA geant4 simulation is assumed.

Some of these features are also thought of to be used to create data sets that can be analyzed with the same code that will/was used for the experimental data. This will help with understanding both the data and the tools used to analyze the data (the latter mainly an aid for students).

2 New features/modifications

This section outlines some of the features added and or changed for the simulations of the AGATA@Ganil campaign.

2.1 AgataPhysicsList

Description of some added flags that controls the physics used in the AGATA.

2.1.1 -SN

Using this flag triggers the use of the G4ScreenedNuclear instead of multiple scattering for generic ions. This should be used whenever the tracking of heavy ions are important (i.e. RDDS or line shape simulations). However, presently, as used it is very slow because it tracks heavy ions every where, not only in selected regions. To improve on this a class called `AgataSNCuts` has been developed. In this energy cuts for heavy ions can be set. For all heavy ions, for heavy ions created by ScreenedElastic and for heavy ions in sensitive detector volumes. The energy

limits can be controlled with the commands

```
/Agata/ScreenedElastic/SetEnergyLimitCreatedByScreenedElastic
/Agata/ScreenedElastic/SetEnergyLimitHeavyIons
/Agata/ScreenedElastic/SetEnergyLimitHeavyIonsInSensitiveVolume
```

and the details are controlled as follows

/Users/joa/AGATA/agataganil/src/AgataSNCuts.cc

```
G4double
AgataSNCuts::PostStepGetPhysicalInteractionLength(const G4Track& track,
                G4double    previousStepSize,
                G4ForceCondition* condition)
{
  *condition = NotForced;
  //If gamma everything goes
  if(track.GetParticleDefinition()->GetParticleName()=="gamma")
    return DBL_MAX;
  //Everything created by ScreenedElastic is stopped if slow
  if(track.GetCreatorProcess()){
    if(track.GetCreatorProcess()->GetProcessName()=="ScreenedElastic" &&
       track.GetKineticEnergy()/track.GetParticleDefinition()->GetAtomicMass()
       < fEnergyLimitCSN){
      return 0.;
    }
  }
  // In target and degrader everything goes or hall phys
  if(track.GetVolume()){
    if(track.GetVolume()->GetName()){
      if(track.GetVolume()->GetName()=="Target" ||
   track.GetVolume()->GetName()=="fronting" ||
   track.GetVolume()->GetName()=="backing" ||
   track.GetVolume()->GetName()=="Degrader" ||
   track.GetVolume()->GetName()=="hallPhys")
  return DBL_MAX;
    }
  }
  if(track.GetParticleDefinition()->GetParticleType()=="nucleus"){
    //Here we stop slow heavy ions
    if((track.GetKineticEnergy()/
  track.GetParticleDefinition()->GetAtomicMass())<fEnergyLimitHIS){
      return 0.;
    }
    //If not in a senstive volume stop also "fast" ions
    const G4Step *astep = track.GetStep();
    const G4StepPoint *steppoint = astep->GetPreStepPoint();
    const G4TouchableHandle touch = steppoint->GetTouchableHandle();
    const G4VPhysicalVolume *volume = touch->GetVolume();
    const G4LogicalVolume *lvolume = volume->GetLogicalVolume();
    if(!lvolume->GetSensitiveDetector()){
      if((track.GetKineticEnergy()/
    track.GetParticleDefinition()->GetAtomicMass())<fEnergyLimitHI){
  return 0.;
      }
    }
```

```
  }
  return DBL_MAX;
}
```

2.2 AgataGeneratorAction

By using the `-AddOn` flag a class generating the events can be loaded from a shared library, given as argument to the flag. This is done via dlload mechanism of standard c. This is done by creating a class in the same way as if adding a new generator, i.e.

/Users/joa/home2/experiment/E614/geantsim/AgataAddOn/e614AddOn.hh

```
class PrimaryGeneratorAction : public AgataGeneration
{
public:
  PrimaryGeneratorAction();
```

and making sure that in the library the following function is accessible for the dlload function.

/Users/joa/home2/experiment/E614/geantsim/AgataAddOn/e614AddOn.cc

```
extern "C" {
  AgataGeneration *Generator(){
    return dynamic_cast<AgataGeneration*>(new PrimaryGeneratorAction());
  }
}
```

This event generator is then loaded by the agata code by

/Users/joa/AGATA/agataganil/src/AgataGeneratorAction.cc

```
AgataGeneratorAction::AgataGeneratorAction( G4String path, G4int genType, G4bool
     hadr, G4bool polar, G4String name )
{
  G4bool inter = true;
  if( genType ) inter = false;

  AgataDefaultGenerator*      theDefaultGenerator      = NULL;
  AgataAlternativeGenerator* theAlternativeGenerator = NULL;
  AgataTestGenerator* theTestGenerator = NULL;
  G4bool UsingAddOnGenerator = false;
  //The first thing we will do is to see if we can find a
  //genartor in the AddOn libraries, if so we use that first of it that
  //we found
  std::vector<void*>::iterator itAddOns =
    AgataDetectorAncillary::AddOns.begin();
  void*(*Generator)();
  for(; itAddOns!=AgataDetectorAncillary::AddOns.end(); ++itAddOns){
    Generator=(void*(*)())dlsym(*itAddOns,"Generator");
    if(Generator){
      theGeneration = static_cast<AgataGeneration*>(Generator());
      UsingAddOnGenerator = true;
      break;
    }
  }
  if(!UsingAddOnGenerator){
```

```
    switch( genType ) {
    case 0:
      theDefaultGenerator = new AgataDefaultGenerator( path, inter, hadr, polar,
     name );
      theGeneration = (AgataGeneration*)theDefaultGenerator;
      break;
    case 1:
      theDefaultGenerator = new AgataDefaultGenerator( path, inter, hadr, polar,
     name );
      theGeneration = (AgataGeneration*)theDefaultGenerator;
      break;
    case 2:
      theAlternativeGenerator = new AgataAlternativeGenerator(name);
      theGeneration = (AgataGeneration*)theAlternativeGenerator;
      break;
    case 3:
      theTestGenerator = new AgataTestGenerator();
      theGeneration = (AgataGeneration*)theTestGenerator;
      break;
    }
  }
}
```

which also shows that if such a generator is found it is the only one used.

2.3 AgataTestGenerator (`shell> Agata -Test ...`)

An event generator that essentially allows the user to set energy, particle, etc using the /gun/ commands of geant4. This can sometimes be helpful when verifying the simulation, .e.g the physics.

2.4 AgataAlternativeGenerator (`shell> Agata -Gen ...`)

The *AgataAlternativeGenerator* class, has been modified and extended quite extensively. The most important of these changes are outlined here.

2.4.1 Notion of absolute cross section

If needed the simulation can, in a very crude way, include the reaction cross section. This is done " by hand" by estimating then number N of beam particles needed before each reaction, and then making the reaction only taking place each N beam particles that passed the target. This can be useful if one tries to estimate the number of random coincidences between, as an example, γ rays from the β decay of radioctive nuclei that has been stopped in the reaction chamber and Rutherford scattering (of the same radioactive nuclei in the beam) into a silicon detector. This is set with the command:

```
/Agata/generator/emitter/SetNumberOfParticlesPerReaction
```

the part of the code handling this looks like

/Users/joa/AGATA/agataganil/src/Reaction.cc

```
G4RunManager * runManager = G4RunManager::GetRunManager();
if(ReactionEachNEvents>0){
  if(runManager->GetCurrentEvent()->GetEventID()%ReactionEachNEvents!=0){
    return DBL_MAX;
  } else {
    ;
  }
}
```

and is found in the method

/Users/joa/AGATA/agataganil/src/Reaction.cc

```
G4double Reaction::PostStepGetPhysicalInteractionLength(
                              const G4Track& aTrack,
                              G4double,
                              G4ForceCondition* condition
                             )
```

2.4.2 Notion of absolute time

For the simulation of some experiments, if the purpose of the simulations is to create "realistic data" to test or develop analyzing tools there is a need of an absolute time. This to be able to take into account not only coincidences but also background in the time spectra, loss in efficiency from detector pile-up etc.

The time is set to zero in the beginning of each run, and then propagated using either the "beam structure" or, if only sources are available the source strength.

Propagation of time using beam structure:

The user have to give the following information:

1. Particles per Second (pps [1/s]), if set ≤ 0 no time propagation
   `/Agata/generator/emitter/SetParticlePerSeconds`
2. Beam Bunch frequency (HF [s]), if set ≤ 0 Continuous beam
   `/Agata/generator/emitter/SetAccelaratorHF`
3. Bunch length (Bunchtime [s]), should be > 0
   `/Agata/generator/emitter/SetWidthOfBeamPuls`

Examples of "beam structures" with 10^{10} and 10^5 particles per second are shown in figure 1 and 2, respectively. Time is then updated using the code

/Users/joa/AGATA/agataganil/src/AgataAlternativeGenerator.cc

```
oldeventTime = eventTime;
if(pps>0){
  // Here update event time if beam I  not 0
  if(HF<=0){
    //if no structure just add 1/pps seconds
    eventTime+=1./pps*s;
  } else {
```

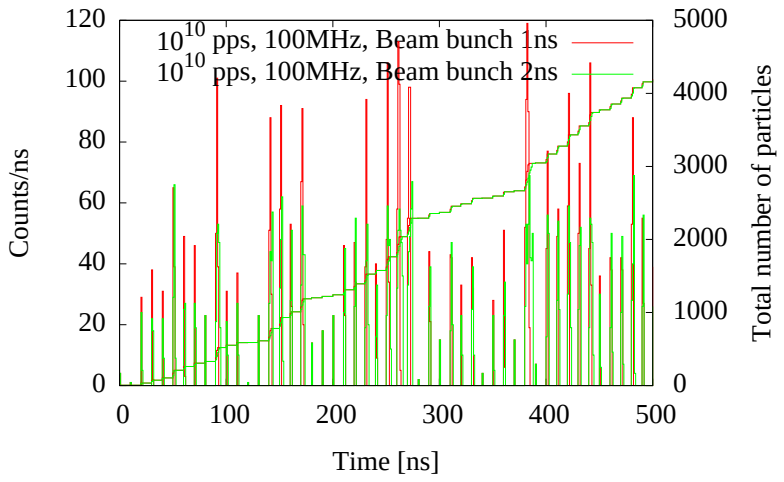

Figure 1: Beam structure for 10^{10} pps, A beam repition rate if 100 MHz and beam bunches of 1 respectively 2 ns.

```
      //Otherwise we have to be smart.
      //Number of particles per bunch can be <1 or >1
      //so the amount of time that has to be added is not always the same
      //First find how many HF we should jump forward
      G4double meanHF=1./particlesperbunch;
      G4int nHFtoJump=CLHEP::RandPoisson::shoot(meanHF);
      if(nHFtoJump>0){
//Now we should move forward eventtime to
//to an hf...
unsigned long int athf = floor(eventTime*HF);
eventTime = athf/HF;
      }
      G4double inbunch=particlesperbunch<1 ? G4UniformRand()*WidthBeamBunch
: CLHEP::RandExponential::shoot(WidthBeamBunch/particlesperbunch);
      eventTime+=(nHFtoJump/(HF*s))*s+inbunch;
      while(eventTime<oldeventTime) eventTime+=1/HF;
    }
}
```

Propagation of time using beam source strength:

In the case of a simulation using only sources, the event time is propagated using the weakest source strength of the given sources. The corresponding code is

/Users/joa/AGATA/agataganil/src/AgataAlternativeGenerator.cc

```
if(pps<=0){
  //Here I have to use the decay of gammas as the thing that moves
  //the event time forward. What I do is that I set the event time to
  //the "latest" gamma decay, decay the other ones the number of required
  //times to get it right. If there is a large number of sources
  //maybe this sort should be replaced by something else...
```

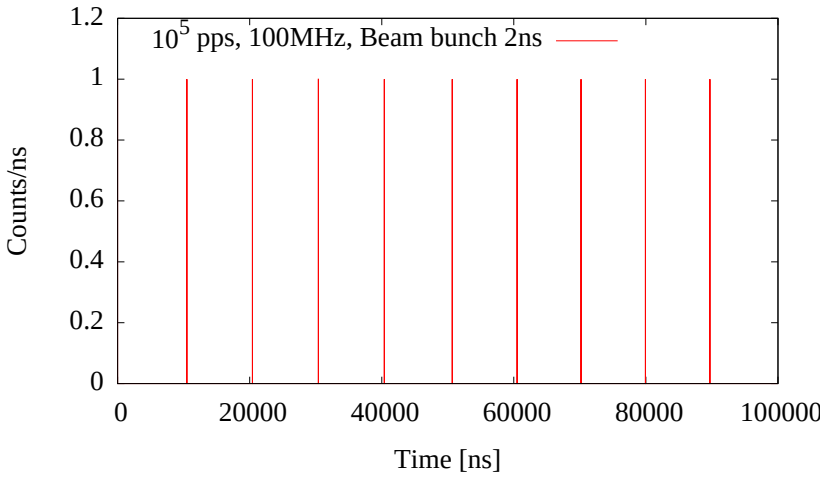

Figure 2: Beam structure for 10^5 pps, A beam repition rate if 100 MHz and beam bunches of 2 ns.

```
    if(thesources.size()>0){
      std::sort(thesources.begin(),thesources.end(),SortSources);
      thesources.back().decay_at_time=CLHEP::RandExponential::
shoot(1./(1e3*thesources.back().activity)*s);
      eventTime+=thesources.back().decay_at_time;
      thesources.back().CreateParticles(anEvent,particleGun,eventTime);
    }
  }
```

Storing the event time:

In the simulation the event time is stored in four variables

```
  G4int day,hour,min;
  G4double eventTime,oldeventTime;
```

that are updated before time propagation of each event according to

/Users/joa/AGATA/agataganil/src/AgataAlternativeGenerator.cc

```
  while(eventTime>=60.*s){
    eventTime-=60.*s;
    min++;
  }
  while(min>=60){
    min-=60;
    hours++;
  }
  while(hours>=24){
    hours-=24;
```

```
    days++;
  }
  oldeventTime = eventTime;
```

It is done like this because of when the AGATA simulation calls the method writing out the event header as relative to when the primary particles are generated.

Information found in output file (GammaEvents.????):

If the user has given enough information to propagate the time the following code is used to create the event headers in the AGATA list mode output files:

/Users/joa/AGATA/agataganil/src/AgataAlternativeGenerator.cc

```
  // Finally writes out the particles
  if(pps<=0 && thesources.size()==0){
    sprintf(aLine, "%5d %10.3f %8.5f %8.5f %8.5f %d\n",
      -partType, epart/unitEnergy,
      direction.x(), direction.y(), direction.z(), evt->GetEventID());
  } else {
    sprintf(aLine, "%5d %10.3f %8.5f %8.5f %8.5f %5d %5d %5d %5d %8.5f\n",
      -partType, epart/unitEnergy,
      direction.x(), direction.y(), direction.z(),
      evt->GetEventID(),days,hours,min,oldeventTime);
  }
```

If the user has also asked for the particle interaction times to be written out, an example of a part of a list mode file could be (here for a 10kBq ^{60}Co source):

```
-100              4
 -101     0.00000   0.00000   0.00000   0.00000
 -102       0.000     0.000     0.000
   -1   1172.000  -0.30927   0.94212  -0.12947      4      0      0      0 400388.00000
-100              5
 -101     0.00000   0.00000   0.00000   0.00000
 -102       0.000     0.000     0.000
   -1   1172.000   0.96552  -0.25318   0.06059      5      0      0      0 500675.00000
    3     1.255     2.300    35.174  -250.957 13 99881.845
    3     1.255    -6.953    47.771  -290.203 43 99881.986
    3     0.015   -21.822    44.881  -285.548 34 99882.039
    3    11.067   -20.953    45.105  -291.317 44 99882.059
    3     1.223   -21.239    45.706  -291.302 44 99882.061
    3   122.775   -21.235    45.715  -291.299 44 99882.061
    3    32.717   -20.952    45.104  -291.318 44 99882.059
    3   179.328   -21.837    44.878  -285.537 34 99882.039
    3   826.234    -6.488    47.316  -290.429 43 99881.987
    3    29.363    -6.488    47.316  -290.429 43 99881.987
    3   117.924     2.307    35.170  -250.962 13 99881.845
    3     1.255   -21.216    45.715  -291.319 44 99882.061
    3     8.589   -21.216    45.715  -291.319 44 99882.061
```

2.4.3 Reaction mechanisms

Some additional reaction mechanisms have been added. Details of there use are given in the examples below.

Coulomb excitation
Allows for Coulex to any initial level in the nucleus. Relativistic kinematics taking into account the energy given to the nucleus in the excitation is used. An angular distribution (in C.O.M system) has to be given, and a file describing the decay of the nucleus. The second file can be auto generated with the command:

```
shell> Agata -decay zZZ.aAAA
```

given that the nucleus is in the geant4 data base. If not, the user has to write her/his own.

Fusion Fission
Beam and target are fused, after which the compound nucleus emits some neutrons, and then fission into the wanted fragments (out of which the second one only has a known Z). User has again to give a file describing the decay of simulated nucleus.

General source
A source can be defined using a [nucleus].geant4src file, with a specified format. The strength, in kBq, should also be given. The source be of a point, a line, a plane, or a spherical shell. It can be used together with other reaction mechanisms. Angular correlations between particles can be given using Legendre polynomial expansions.

2.4.4 Addition of γ-ray backgrounds

Discrete γ rays
Any number of discrete γ rays that should be emitted, with a given probablity, each event can be added using the command

```
/Agata/generator/emitter/AddDiscreteGamma
```

and cleared with the command

```
/Agata/generator/emitter/ClearListOfDiscreteGammas
```

The code to generate these γ rays is

/Users/joa/AGATA/agataganil/src/AgataAlternativeGenerator.cc

```
std::vector<std::pair<G4double,G4double> >::iterator
  iteg = fDiscreteExtraGammas.begin();
for(;iteg!=fDiscreteExtraGammas.end(); ++iteg){
  int nbgammas =
    ((int)floor(iteg->second)+
     (G4UniformRand()<(iteg->second-floor(iteg->second)) ? 1:0));
```

```
    for(int i=0; i<nbgammas; i++){
      energy   = iteg->first;
      position = G4ThreeVector();
      G4double costheta = 2.0 * G4UniformRand() - 1.0;
      G4double sintheta = sqrt( 1 - costheta * costheta );
      G4double phi      = 360. * G4UniformRand() * deg;
      direction.set( sintheta*cos(phi), sintheta*sin(phi), costheta );
      particleGun->SetParticlePosition(position);
      particleGun->SetParticleMomentumDirection(direction);
      particleGun->SetParticleEnergy(energy);
      particleGun->GeneratePrimaryVertex(anEvent);
    }
  }
```

Exponential distribution

Gamma rays from an exponential distribution can also be added each event, although only with a fixed number. The relevent commands are

```
/Agata/generator/emitter/setSlopeGammaBackground
/Agata/generator/emitter/setMaxEGammaBackground
/Agata/generator/emitter/setNumberOfGammaBackground
```

and the γ rays are generated with

/Users/joa/AGATA/agataganil/src/AgataAlternativeGenerator.cc

```
  double A = exp(-SlopeExpBkg*MaxEBkg);
  for(int i=0; i<nbextragammasbkg; i++){
    double r = A+(1-A)*G4UniformRand();
    while(r<=0) r = A+(1-A)*G4UniformRand();
    energy   = -1/SlopeExpBkg*log(r);
    position = G4ThreeVector();
    if(energy>MaxEBkg) std::cout << energy << " " << MaxEBkg << std::endl;
    G4double costheta = 2.0 * G4UniformRand() - 1.0;
    G4double sintheta = sqrt( 1 - costheta * costheta );
    G4double phi      = 360. * G4UniformRand() * deg;
    direction.set( sintheta*cos(phi), sintheta*sin(phi), costheta );
    particleGun->SetParticlePosition(position);
    particleGun->SetParticleMomentumDirection(direction);
    particleGun->SetParticleEnergy(energy);
    particleGun->GeneratePrimaryVertex(anEvent);
  }
```

2.4.5 Multiple reaction products

It is now possible to add several particles to the list of particles produced in a "reaction". This is done via modifications in the classes `Outgoing_beam.cc` and `Reaction.cc`. This is presently only used to create a target recoil for Coulomb excitation reactions.

2.5 AgataDetectorAncillary

A mechanism similar to that in AgataGeneratorAction has also been added to the AgataDetectorAncillary class. It is also invoked using the `-AddOn` flag. An ancillary class is defined as

usual in the AGATA package, i.e.

/Users/joa/home2/experiment/E614/geantsim/AgataAddOn/e614AddOn.hh

```
class Galotte  : public AgataAncillaryScheme
{
public:
  Galotte(G4String,G4String);
  ~Galotte();
```

In the shared library the following functions should also be defined

/Users/joa/home2/experiment/E614/geantsim/AgataAddOn/e614AddOn.cc

```
Galotte *agalotte;

extern "C" {
  AgataAncillaryScheme *Constructor(G4String path , G4String name){
    agalotte = new Galotte(path,name);
    return dynamic_cast<AgataAncillaryScheme*>(agalotte);
  }
}

extern "C" {
  void Destructor(){
    delete agalotte;
  }
}
```

The ancillary is then added to the list of ancillaries like

/Users/joa/AGATA/agataganil/src/AgataDetectorAncillary.cc

```
  //Here we should check for addons...
  std::vector<void*>::iterator itAddOns = AddOns.begin();
  void*(*Constructor)(G4String,G4String);
  for(; itAddOns!=AddOns.end(); ++itAddOns){
    Constructor=(void*(*)(G4String,G4String))dlsym(*itAddOns,"Constructor");
    if(Constructor){
      AgataAncillaryScheme *anc = static_cast<AgataAncillaryScheme*>
  (Constructor(path,name));
      numAnc++;
      whichAnc.push_back(1000000+numAnc);
      theAncillary.push_back((AgataAncillaryScheme*)anc);
      theConstructed.push_back((AgataDetectorConstructed*)anc);
    }
  }
```

2.6 New Detector geometry, Orgam (`shell> Agata -g 4 ...`)

As a fourth geometry has been added. It correspondes to the Eurogam array with its frame and phase I coaxial HPGe detectors. This is a feature mainly used for experiments at the IPN Orsay Tandem, and will therefor not be described in detail.

3 Output to named pipe: Compressing or Analyzing on the fly

The user has the option, using the command

`/Agata/file/WriteToFifo`

which then creates a fifo named `GammaEvent.fifo`. The simplest way to profit from this to compress the list mode file while it is created. To do this run the Agata geant4 simulations in one terminal, and compress in another terminal window with the command:

`shell> cat GammaEvents.fifo | bzip2 -c GammaEvents.bz2`[1]

A more complex example follows later in this text.

[1] From memory, did not try while writing this...

4 Different simulations

4.1 ^{62}Fe DIS RDDS

In this section the example of a simulation of one distance for a plunger measurement of multi-nucleon transfer (^{238}U on ^{64}Ni target). In order to setup such a simulation a set of files are needed. First of all, in this example, two macro files. The first one, `62FeSetup.mac` sets up the physics of the simulation:

Listing 1: 62FeSetup.mac

```
################## Reaktion
/Agata/generator/emitter/BeamIn/Z 92
/Agata/generator/emitter/BeamIn/A 238
/Agata/generator/emitter/BeamIn/KE 1547 MeV
/Agata/generator/emitter/BeamIn/fcZ -5 cm
/Agata/generator/emitter/BeamIn/bDir 45 180
/Agata/generator/emitter/BeamOut/DZ -66
/Agata/generator/emitter/BeamOut/DA -176
#target in the following
/Agata/generator/emitter/BeamOut/Z 28
/Agata/generator/emitter/BeamOut/A 64
#starts at a level E*
/Agata/generator/emitter/BeamOut/ProjectileExcitation 2.1759e+03 1.2 8.768e+02 1
/grdm/setRadioactiveDecayFile 26 62 decay62Fe
############### Geometria
/Agata/detector/targetMaterial G4_Ni
/Agata/detector/targetSize 20 20 1
/Agata/detector/degraderMaterial G4_Mg
/Agata/detector/degraderSize 20 20 5.7
/Agata/detector/separation .1 mm
/grdm/allVolumes
/grdm/verbose 0
/Agata/detector/SetBuildSpectrometer true
/Agata/detector/angleFile MNTeuler.list
/Agata/detector/rotateArray 0 20
#distribution of reaction points according to "./profile.dat"
/Agata/generator/emitter/disableFixDepth
/Agata/generator/emitter/enableUniformDistr
/Agata/generator/emitter/BeamOut/setPtr 1.
/Agata/generator/emitter/BeamOut/setPfe  0.
/Agata/generator/emitter/BeamOut/setPclx 0.
/Agata/generator/emitter/BeamOut/setPff 0.
/Agata/generator/emitter/BeamOut/adistFile aadist_45
#to "hit" VAMOS and not simulate a lot of junk
/Agata/generator/emitter/BeamOut/phi 170 190
/Agata/generator/emitter/setNumberOfGammaBackground 0
/Agata/generator/emitter/SetEventsPerReaction 10
```

It sets the beam energy to 1548 MeV, tells the simulation to create iron (DZ=-66) of mass 62 (DA=-176). Further more, the recoil will have to different states populated (corresponding to

the 2+ and 4+) populated with almost equal probability. An important command is

```
/grdm/setRadioactiveDecayFile 26 62 decay62Fe
```

which sets the datafile for the decay of recoil. Such a file can be created using
`shell> Agata -decay z26.a062`

and modified if needed. In this example we have:

Listing 2: decay62Fe

```
#  62FE (68 s)
#  Excitation    Halflife Mode        Daughter Ex  Intensity           Q
#
# File sanitized by Vanderbilt Decay-o-matic
# $Id: repair_decay_files.py,v 1.20 2006/04/24 14:53:36 marcus Exp $
# Fri Oct  6 15:26:18 2006
#
#P        0.0000    6.8000e+01
#                          BetaMinus        0.0000    1.0000e+00
#                          BetaMinus      506.1000    1.0000e+02    2023.9000
P           876.8    5.1e-12
                          IT           876.8       100.
P          1818.8    1.45e-12
                          IT          1818.8       100.
P            2016    1.45e-12
                          IT            2016       100.
P          2175.9    8.8e-12
                          IT          2175.9       100.
P            3011    5e-12
                          IT            3011       100.
P            3017    1e-14
                          IT            3017       100.
P            3312    1e-14
                          IT            3312       100.
P            3390    1e-24
                          IT            3390       100.
P            3606    1e-24
                          IT            3606       100.
P            3631    1e-24
                          IT            3631       100.
P          3633.6    1e-24
                          IT          3633.6       100.
P            4255    1e-24
                          IT            4255       100.
P            5323    1e-24
                          IT            5323       100.
```

We also need a file with the relative differential cross section, in this example called `aadist_45`. It is shown in figure 3 and some of the file is shown below:

Listing 3: aadist_45

```
0 0 6.49935e-20
0.1 0.1 7.13957e-20
0.2 0.2 7.84206e-20
```

```
0.3  0.3  8.61282e-20
0.4  0.4  9.45839e-20
0.5  0.5  1.03859e-19
0.6  0.6  1.14033e-19
0.7  0.7  1.25191e-19
0.8  0.8  1.37426e-19
0.9  0.9  1.50843e-19
.
.
.
179.7  179.7  1.12627e-16
179.8  179.8  1.03372e-16
179.9  179.9  9.48675e-17
180  180  8.70543e-17
```

Finally, there is the macro, `62Fe_AGATA.mac`, that we use to run the simulation:

Listing 4: 62Fe_AGATA.mac

```
/control/execute 62FeSetup.mac
/Agata/file/enableLM
/Agata/detector/separation .250 mm
/Agata/detector/update
/Agata/run/runNumber 3000
/tracking/verbose 0
/run/beamOn 240000
```

with the commando

```
shell> Agata -Gen -n -SN -b 62Fe_AGATA.mac
```

which will generate an output file called `GammaEvents.3000`. The geometry and some 10 events are shown in figure 4. A typical event in this file could be

```
-100              4
 -101     0.12182   0.06566  -0.11458   0.99124
 -102       0.000     0.000    -0.267
 -104 428.00000 57680.47785
   -8 1547000.000  -0.70711   0.00000   0.70711 4
   21      0.038   -38.411   198.765  -129.676 00
   21      0.129   -37.892   197.309  -129.271 00
   21     11.067   -42.484   191.370  -132.929 00
   21    147.441   -42.490   191.358  -132.925 00
   21     38.044   -37.891   197.309  -129.270 00
   21    179.370   -38.627   198.869  -129.727 00
   21    231.744   -38.629   198.887  -129.704 00
   21    199.900   -38.641   198.884  -129.735 00
```

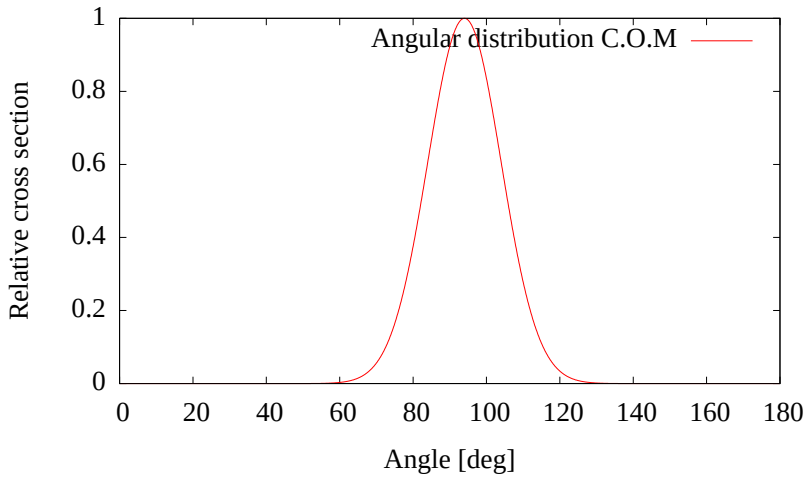

Figure 3: Angular distribution of ^{62}Fe recoils used in the simulation.

This can be decoded as follows:

`-100` - New event followed by event number
`-101` - Velocity and direction of recoil when passing the “spectrometer”
`-102` - Position of reaction in target
`-104` - Energy and mass of particle passing the “spectrometer”

The rest is as described in the AGATA simulation manual. Treating this data in your favorite way should give something like what is shown in figure 5

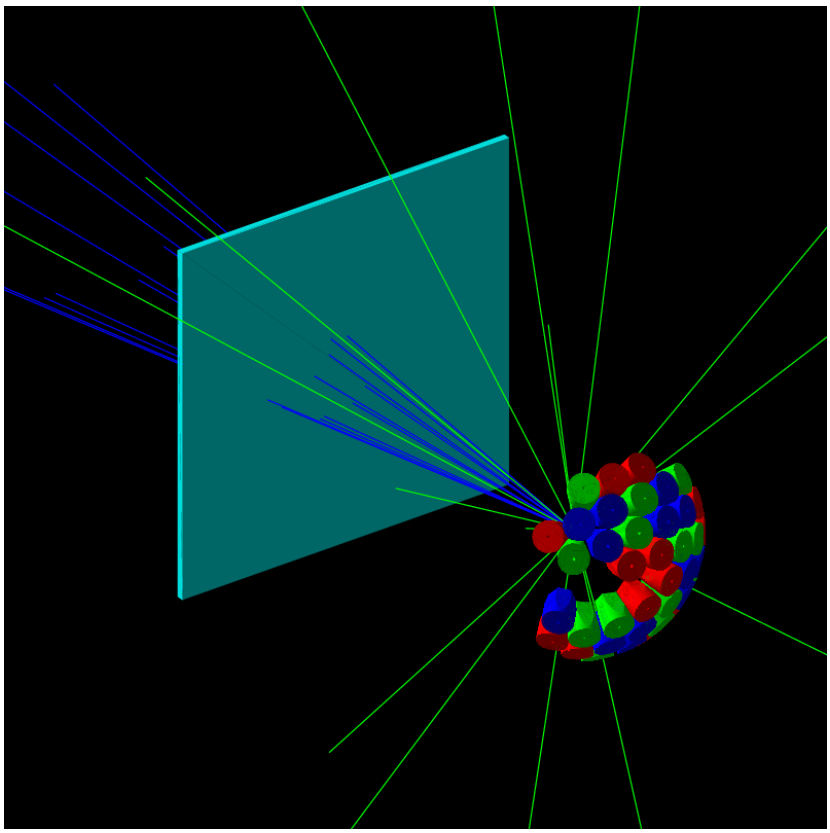

Figure 4: Geant4 simulation of a RDDS measurement of ^{62}Fe after multi-nucleon transfer. Ten events are shown. Blue lines are the recoils, green lines emitted γ rays. The colored wall is a schematic "spectrometer".

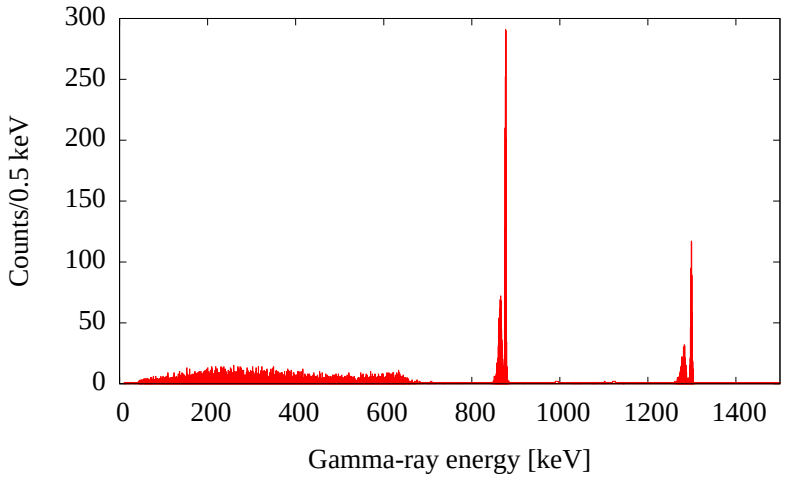

Figure 5: Gamma-ray spectrum for the simulation of a RDDS experiment for ^{62}Fe.

4.2 ^{186}Pb HIFE RDDS

The following example shows the simulation of a heavy-ion fusion evaportion experiment. In this case $^{82}Kr(^{106}Pd, 2n)^{186}Pb$ with a beam energy of 357 MeV. The macro file to set this up is

Listing 5: 186PbSetup.mac

```
################## Reaktion
/Agata/generator/emitter/BeamIn/Z 36
/Agata/generator/emitter/BeamIn/A 83
/Agata/generator/emitter/BeamIn/KE 357 MeV
/Agata/generator/emitter/BeamIn/fcZ -5 cm
/Agata/generator/emitter/BeamOut/DZ 46
/Agata/generator/emitter/BeamOut/DA 103
#target in the following
/Agata/generator/emitter/BeamOut/Z 46
/Agata/generator/emitter/BeamOut/A 106
#starts at a level E* = 3707 keV
/Agata/generator/emitter/BeamOut/ProjectileExcitation 2160.9 1
/grdm/setRadioactiveDecayFile 82 186 decay186Pb
/grdm/setPhotoEvaporationFile 82 186 gammas186Pb
############## Geometry
/Agata/detector/targetMaterial G4_Pd
/Agata/detector/targetSize 20 20 1
/Agata/detector/degraderMaterial G4_Mg
/Agata/detector/degraderSize 20 20 1
/Agata/detector/separation .05 mm
/Agata/detector/SetBuildSpectrometer true
/Agata/detector/angleFile A180/A180euler1P.list
/grdm/allVolumes
/grdm/verbose 0
/Agata/generator/emitter/disableFixDepth
/Agata/generator/emitter/enableUniformDistr
/Agata/generator/emitter/BeamOut/setEvapNumberN 3
/Agata/generator/emitter/BeamOut/disableEvapFromFileP
/Agata/generator/emitter/BeamOut/setPtr 0.
/Agata/generator/emitter/BeamOut/setPfe  1.
/Agata/generator/emitter/BeamOut/setPclx 0.
/Agata/generator/emitter/BeamOut/setPff 0.
/Agata/detector/update
```

Two addtional files are needed, `decay186Pb`, and also `gammas186Pb`. This is because ^{186}Pb is not a part of the geant4 gamma-decay data base. The two files are shown below.

Listing 6: decay186Pb

```
# 186PB (4.79 s)
#  Excitation    Halflife Mode        Daughter Ex  Intensity           Q
#
# File sanitized by Vanderbilt Decay-o-matic
# $Id: repair_decay_files.py,v 1.20 2006/04/24 14:53:36 marcus Exp $
# Fri Oct  6 15:26:26 2006
#
#P       0.0000    4.7900e+00
#                                Alpha        0.0000    1.0000e+00
#                                Alpha        0.0000    5.3780e+01    6471.0000
#                                Alpha      328.0000    1.0800e-01    6143.0000
```

```
#                                   Alpha      351.8000    8.6400e-02     6119.2000
P          662.4    12.5e-12
                          IT          662.4       100.
P            923    12.5e-12
                          IT            923       100.
P           1260    4.16e-12
                          IT           1260       100.
P         1675.1    3.47e-12
                          IT         1675.1       100.
P         2160.9    1e-24
                          IT         2160.9       100.
P         2711.4    1e-24
                          IT         2711.4       100.
P         3313.9    1e-24
                          IT         3313.9       100.
```

Listing 7: gammas186Pb

```
6.624000e+02 6.624000e+02 1.000e+02  1 0.00e+00   2.00 5.520e-03 8.225e-01 1.087
    e-01 1.190e-02 8.207e-03 2.464e-02 3.025e-03 2.174e-03 2.210e-05 3.116e-05
    9.583e-03
9.230000e+02 2.606000e+02 1.000e+02  1 0.00e+00   4.00 1.771e-01 8.067e-01 1.005
    e-01 2.138e-02 1.737e-02 2.265e-02 5.274e-03 4.463e-03 1.119e-04 1.294e-04
    1.033e-02
1.260000e+03 3.370000e+02 1.000e+02  1 0.00e+00   6.00 8.240e-02 8.130e-01 1.039
    e-01 1.848e-02 1.426e-02 2.343e-02 4.609e-03 3.704e-03 7.348e-05 8.652e-05
    1.013e-02
1.675100e+03 4.151000e+02 1.000e+02  1 0.00e+00   8.00 4.680e-02 8.138e-01 1.055
    e-01 1.628e-02 1.207e-02 2.393e-02 4.090e-03 3.152e-03 5.131e-05 6.248e-05
    9.931e-03
2.160900e+03 4.858000e+02 1.000e+02  1 0.00e+00  10.00 3.160e-02 8.144e-01 1.067
    e-01 1.462e-02 1.048e-02 2.404e-02 3.683e-03 2.760e-03 3.856e-05 4.894e-05
    9.808e-03
2.711400e+03 5.505000e+02 1.000e+02  1 0.00e+00  12.00 2.340e-02 8.198e-01 1.071
    e-01 1.339e-02 9.487e-03 2.428e-02 3.404e-03 2.503e-03 3.066e-05 4.080e-05
    9.712e-03
3.313900e+03 6.025000e+02 1.000e+02  1 0.00e+00  14.00 1.900e-02 8.213e-01 1.077
    e-01 1.260e-02 8.799e-03 2.447e-02 3.198e-03 2.327e-03 2.613e-05 3.574e-05
    9.640e-03
```

For the format of this file, see the geant4 documentation/and or slac forum. With the macro file

Listing 8: 186Pb.mac

```
/control/execute 186PbSetup.mac
/Agata/detector/update
/Agata/file/enableLM
/Agata/file/info/enableTime
/Agata/file/verbose 1
/tracking/verbose 0
/Agata/run/runNumber 18602
/Agata/detector/separation .10 mm
/Agata/detector/update
/run/beamOn 10000
```

and the command

```
shell> Agata -Gen -n -SN -b 186Pb.mac
```

an output file called GammaEvents.18602 is created. It has the same format as the output file of the ^{62}Fe simulations of above. The geometry of the simulation is shown in figure 6, and in figure 7 is the γ-ray spectrum produced after tracking and Doppler correction.

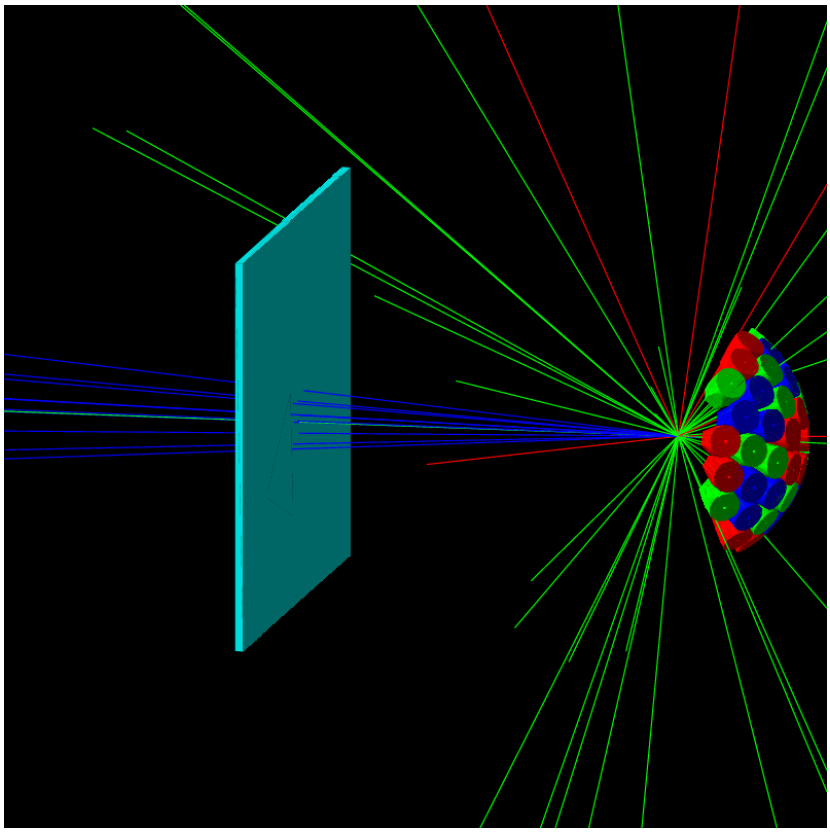

Figure 6: Geant4 simulation of a RDDS measurement of ^{186}Pb after heavy-ion fusion evaporation reaction. Ten events are shown. Blue lines are the recoils, green lines emitted γ rays. Red lines are conversion electrons. The colored wall is a schematic "spectrometer".

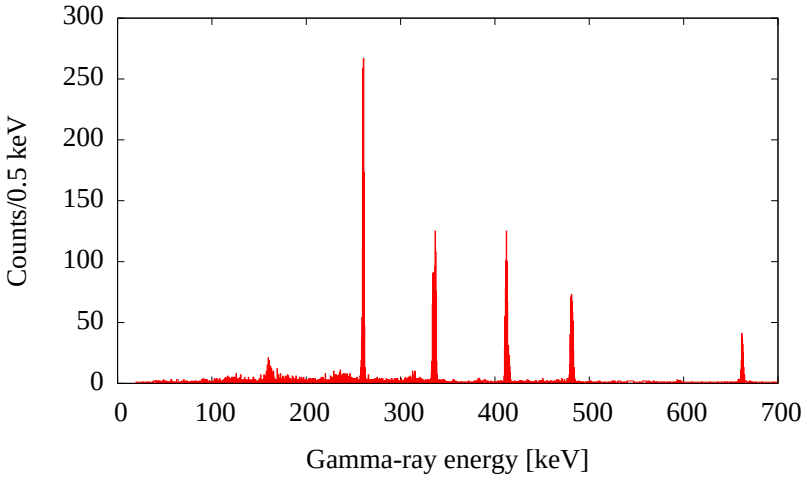

Figure 7: Gamma-ray spectra for the simulation of a RDDS experiment aimed at measuring lifetimes in ^{186}Pb.

4.3 ^{238}U+^{12}C Fusion Fission

Here are the files needed to reproduce the spectrum found in figure 9. Apart from the reaction mechanism this example also shows how to add a γ-ray background.

Listing 9: FFSetup.mac

```
### Beam
/Agata/generator/emitter/BeamIn/Z 92
/Agata/generator/emitter/BeamIn/A 238
/Agata/generator/emitter/BeamIn/KE 1476 MeV
/Agata/generator/emitter/BeamIn/fcZ -5 cm
/Agata/generator/emitter/BeamIn/bDir 20 180
#Give reaction product "counted" from beam, this gives 114Pd
/Agata/generator/emitter/BeamOut/DZ -46
/Agata/generator/emitter/BeamOut/DA -124
#target in the following, OBS this is not the "geant4 target per se",
#but used in the reaction kinematics
/Agata/generator/emitter/BeamOut/Z 4
/Agata/generator/emitter/BeamOut/A 9
#starts at a level E* 3253.4 (12+)
/Agata/generator/emitter/BeamOut/ProjectileExcitation 4147.4 1.
/grdm/setRadioactiveDecayFile 46 114 decay114Pd
############## Geometria
#Here target. G4_Be is natural Be but close enough
/Agata/detector/targetMaterial G4_Be
/Agata/detector/targetSize 20 20 2.3
/Agata/detector/degraderMaterial Vacuum
/grdm/allVolumes
/grdm/verbose 0
#This gives a "spectrometer" giving which ion was detected in the output
/Agata/detector/SetBuildSpectrometer true
#no AGATA detectors hence an empty list of detectors
/Agata/detector/angleFile A180/A180euler1P.list
/Agata/detector/update
#distribution of reaction points according to "./profile.dat"
/Agata/generator/emitter/disableFixDepth
/Agata/generator/emitter/enableUniformDistr
#what kind of reactions...
/Agata/generator/emitter/BeamOut/setPtr 0.
/Agata/generator/emitter/BeamOut/setPfe  0.
/Agata/generator/emitter/BeamOut/setPclx 0.
/Agata/generator/emitter/BeamOut/setPff 1.
#to "hit" VAMOS and not simulate a lot of junk
/Agata/generator/emitter/BeamOut/phi 175 185
/Agata/generator/emitter/BeamOut/theta 15 25
#to produce a background...
/Agata/generator/emitter/setNumberOfGammaBackground 20
/Agata/generator/emitter/SetNumberOfParticlesPerReaction 1
```

Listing 10: decay114Pd

```
# 114PD (2.42 m)
#  Excitation    Halflife Mode       Daughter Ex  Intensity           Q
#
# File sanitized by Vanderbilt Decay-o-matic
# $Id: repair_decay_files.py,v 1.20 2006/04/24 14:53:36 marcus Exp $
# Fri Oct  6 15:26:20 2006
```

```
#
#P         0.0000    1.4520e+02
#                          BetaMinus          0.0000    1.0000e+00
#                          BetaMinus          0.0000    9.2000e+01     1451.0000
#                          BetaMinus        126.7000    8.0000e-01     1324.3000
#                          BetaMinus        136.7000    2.5000e-01     1314.3000
#                          BetaMinus        358.5000    7.5000e+00     1092.5000
P            332.5    2e-10
                           IT           332.5       100.
P            694.4    1e-11
                           IT           694.4       100.
P            852.2    1e-11
                           IT           852.2       100.
P            871.9    1e-11
                           IT           871.9       100.
P           1011.5    1e-11
                           IT          1011.5       100.
P           1115.5    1e-11
                           IT          1115.5       100.
P           1319.8    1e-11
                           IT          1319.8       100.
P           1391.7    1e-11
                           IT          1391.7       100.
P           1500.2    1e-11
                           IT          1500.2       100.
P           1630.3    1e-11
                           IT          1630.3       100.
P           1983.6    1e-11
                           IT          1983.6       100.
P             2065    1e-11
                           IT            2065       100.
P           2183.6    1e-11
                           IT          2183.6       100.
P           2215.8    1e-11
                           IT          2215.8       100.
P           2289.9    1e-11
                           IT          2289.9       100.
P           2520.2    1e-11
                           IT          2520.2       100.
P           2598.7    1e-11
                           IT          2598.7       100.
P           2622.8    1e-11
                           IT          2622.8       100.
P           2654.6    1e-11
                           IT          2654.6       100.
P           2789.1    1e-11
                           IT          2789.1       100.
P           2859.8    1e-11
                           IT          2859.8       100.
P           2905.6    1e-11
                           IT          2905.6       100.
P           3104.6    1e-11
                           IT          3104.6       100.
P           3128.1    1e-11
                           IT          3128.1       100.
P           3337.7    1e-11
                           IT          3337.7       100.
```

```
P       3443.3   1e-11
                         IT       3443.3      100.
P       3503.8   1e-11
                         IT       3503.8      100.
P       3737.9   1e-11
                         IT       3737.9      100.
P       4147.4   1e-11
                         IT       4147.4      100.
P       4472.7   1e-11
                         IT       4472.7      100.
P       5011.7   1e-11
                         IT       5011.7      100.
P       5255.8   1e-11
                         IT       5255.8      100.
```

Listing 11: FF_AGATA.mac

```
/control/execute FFSetup.mac
/Agata/file/enableLM
/Agata/detector/separation 0. mm
/Agata/detector/update
/Agata/run/runNumber 0
/tracking/verbose 0
/Agata/generator/emitter/BeamOut/ProjectileExcitation 3443.3 1.
/run/beamOn 50000
```

```
shell> Agata -Gen -n -SN -b FF_AGATA.mac
```

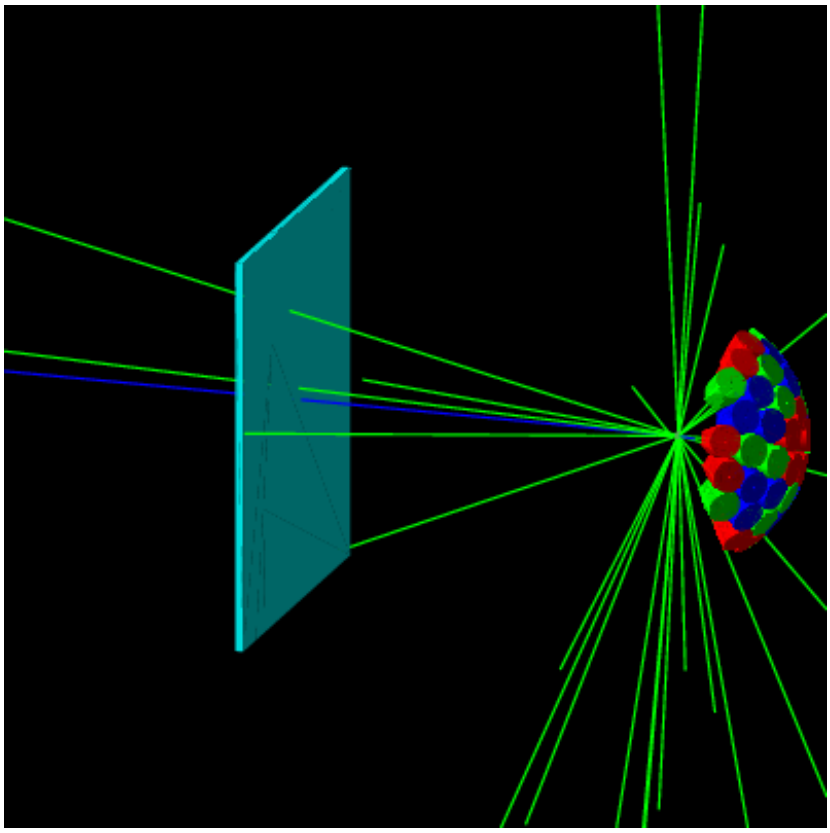

Figure 8: Geant4 simulation of fusion-fission experiment, ^{238}U on Be. One event is shown. Blue lines are the recoils, green lines emitted γ rays. The colored wall is a schematic "spectrometer".

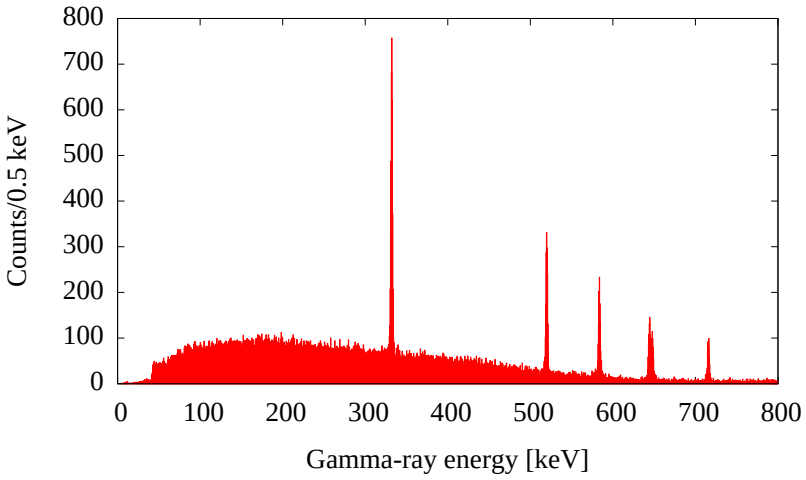

Figure 9: Gamma-ray spectra for the simulation of a fusion fisson experiment producing ^{114}Pd

5 From geant4 to ADF - Analyzing simulated data like experimental data

The following to examples show two advanced features:

1. The use of an absolute time in the simulations
2. An example of how to use this by creating events corresponding to what comes out from the AGATA PSA algorithms when running an experiment.

5.1 ^{60}Co source

The first example shows the simulation of three different ^{60}Co sources, with the activities 1 kBq, 100 kBq and 10 MBq. For all three sources the total number of γ rays emitted where $2 * 10^6$. The simulation also has one EXOGAM clover as an "ancillary". This is however not treated in this example. The geometry is shown in figure 10. To produce such a simulation the following macro is used

Listing 12: sourceruns.mac

```
/Agata/file/enableLM
/Agata/file/info/enableTime
/Agata/file/WriteToFifo true
/Agata/file/info/enableNSeg
/Agata/file/verbose 1
/tracking/verbose 0
/Agata/detector/enableAncillary
/Agata/detector/traslateArray 0. 0. 0.
/Agata/detector/solidFile /Users/joa/AGATA/agataganil/A180/A180solid.list
/Agata/detector/angleFile /Users/joa/AGATA/agataganil/A180/A180euler5T.list
/Agata/detector/wallsFile /Users/joa/AGATA/agataganil/A180/A180wallsS2p.list
/Agata/detector/clustFile /Users/joa/AGATA/agataganil/A180/A180clustS2p.list
/Agata/detector/sliceFile /Users/joa/AGATA/agataganil/A180/A180slice.list
/Agata/detector/enableCapsules
/Agata/detector/wallThickness 4.0
/process/inactivate Reaction
/Agata/detector/update
/grdm/allVolumes
/Agata/generator/emitter/SetGammaRaySource 60Co 10000
/run/beamOn 1000000
```

where in this example a source activity of 10MBq is given (10000 kBq). The simulation will also look for a file called `60Co.geant4src` that defines the γ-ray source. In this example this file looks like

Listing 13: 60Co.geant4src

```
#This is a comment...
#first a line describing the geometry of the source
#Point x y z
#or
#Line x1 y1 z1 x2 y2 z2
#or
#Plane x1 y1 z1 x2 y2 z2 x3 y3 z3
```

```
#or
#Sphere x y z R1 R2
#each branch is separated by a blank line
#first line of each branch is prob
#and each decay in each branch is a line
#
# Particle tau Energy_dist_type Energy_par A_coeff_in_Ang_Corr
#
#Not that for the first particle in each branch the emisson angle
#is isotropic, and lifetime set to zero (decay time is generated)
#

Point 0 0 0

1.
gamma 0      Discrete 1172  1
gamma 8e-13 Discrete 1333  1 0 0.1020 0 0.0091
```

in which the format is explained. To run this simulation, one start the simulation in one terminal

```
shell> Agata -Gen -a 1 9 -b sourceruns.mac
```

The simulation will then continue stop waiting for another process to read on the fifo named `GammaEvents.fifo`. In this example this fifo is both saved and analyzed in such a way that adf files are generated that can be anaylsed using, e.g., the femul emulator. This is done like

```
shell> cat GammaEvents.fifo | tee -a 10000kBqsource | MakeADFEvents
```

and a adf file is generated for each AGATA crystal and for the EXOGAM clover. The details of this is not within the scope of this text, but the source code of `MakeADFEvents` can be found in A. An event in the output of this simulation looks like

```
-100          999852
 -101     0.00000   0.00000   0.00000   0.00000
 -102       0.000     0.000     0.000
   -1   1172.000   0.13682   0.13277  -0.98166 999852      0      0    16
    37287669131.67716
    0      1.255    36.394    35.317  -261.116 24 111823.873
    0      0.006    37.188    36.779  -263.150 24 111823.881
    0      0.006    33.862    38.151  -267.956 24 111823.901
    0      0.129    33.219    38.342  -272.998 34 111823.918
    0     11.067    33.143    28.200  -294.558 44 111823.998
    0    391.495    33.150    28.143  -294.588 44 111823.998
    0     43.249    33.218    38.343  -272.999 34 111823.918
    0     56.380    33.859    38.152  -267.955 24 111823.901
    0    364.100    37.233    36.809  -263.172 24 111823.882
    0    304.311    36.385    35.290  -261.150 24 111823.873
```

As compered to the event shown for the ^{62}Fe simulation the line starting with the token showing the emitted particle, in this case `-1` also contains information about the "wall clock" time. The four last numbers are, in order, day hour seconds nanoseconds. For each interaction point the

last number is the time, in nanoseconds, for the interaction as mesured from the "wall clock" time given in the event header.

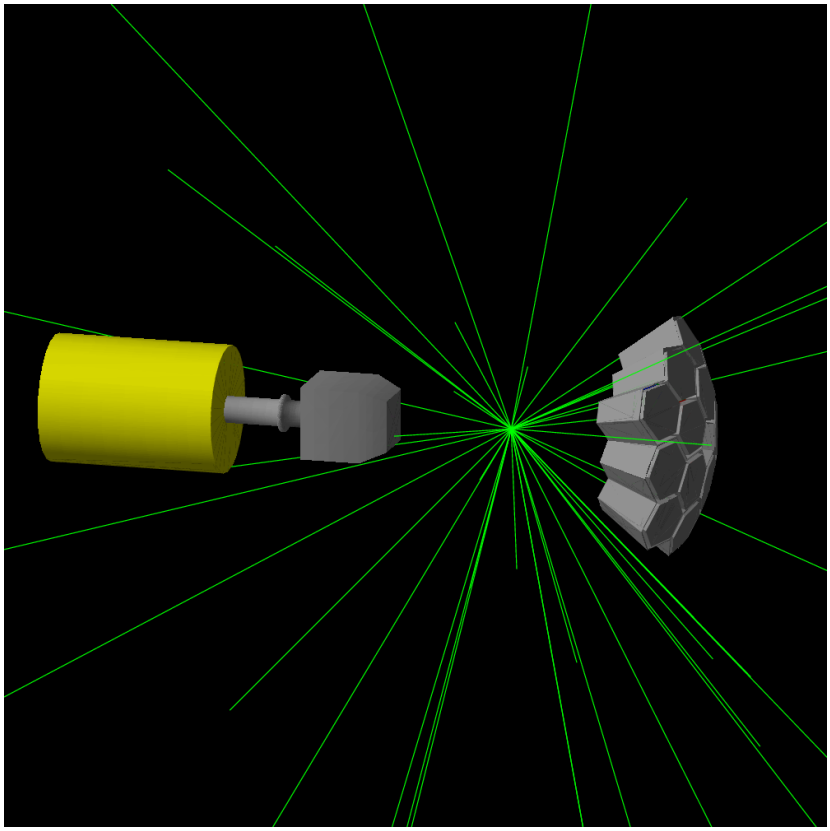

Figure 10: Geant4 simulation of a ^{60}Co source. Green lines are emitted γ rays. The simulation includes the aluminum cans and end caps. A EXOGAM clover is also included.

Figure 11 to 15 shows some results that have been produced using the Gammaware package using the adf file containing tracked data. The distribution for the consecutive timestamps shown in figure 11 should follow exponetial distributions. The probablity to detect at least one γ ray from ^{60}Co is about 15%, which means that the distributions for 1kBq and 100kBq has the expected shape. For 10MBq this is no longer the case, and this is due to pile up in the detectors (I have assumed that all events within 10μs in a detector are considered as one). One can also see the expected count-rates in individual detectors, this by dividing by 15, i.e, 10 Hz, 1000 Hz, and 60kHz, respectively.

Looking at the γ-ray spectra in figure 12 it is again clear that for the 10MBq source, there is a huge number of events with summing. The peak efficiency for for 1kBq, 100kBq, and 10MBq are 4.5%, 4.2%, and 0.2%, respectivley.

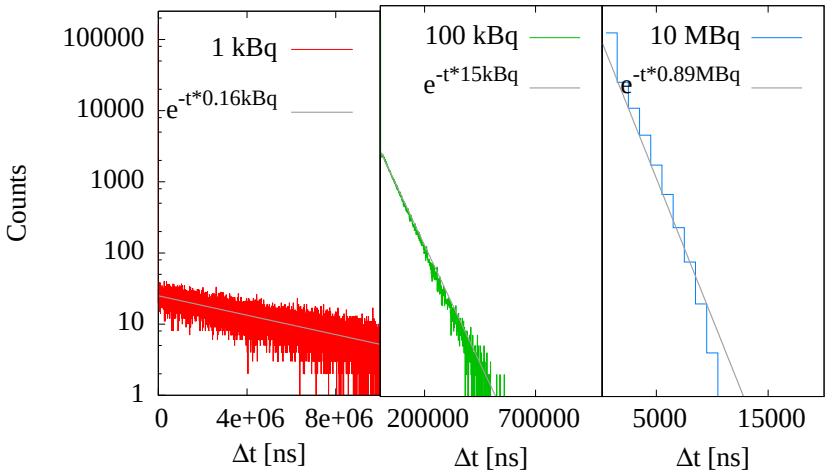

Figure 11: Consecutive timestamps for events following each other for the three different ^{60}Co sources.

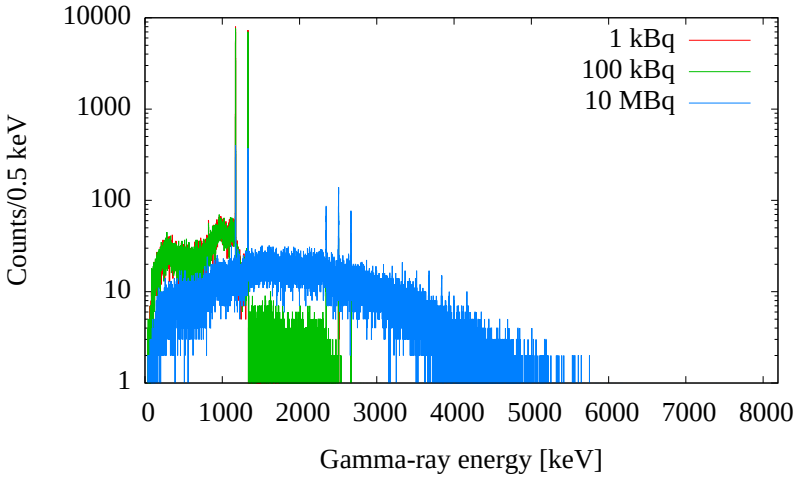

Figure 12: Singles γ-ray spectra for three different ^{60}Co sources.

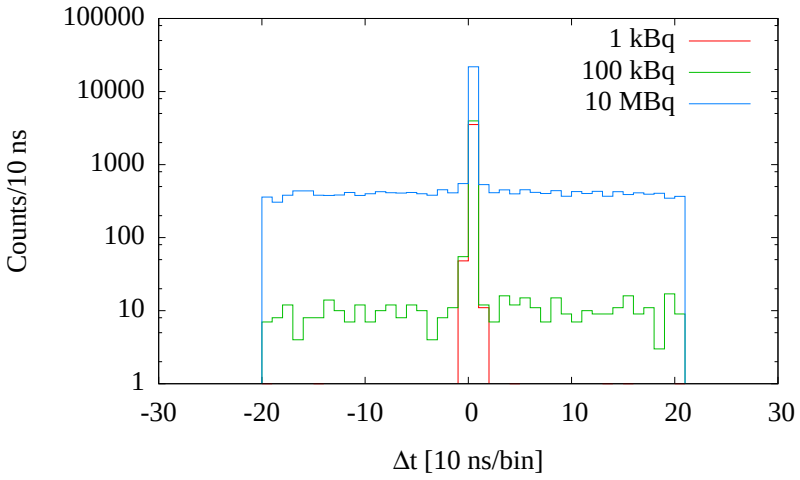

Figure 13: Time spectra for $\gamma\gamma$ coincidences for three different ^{60}Co sources.

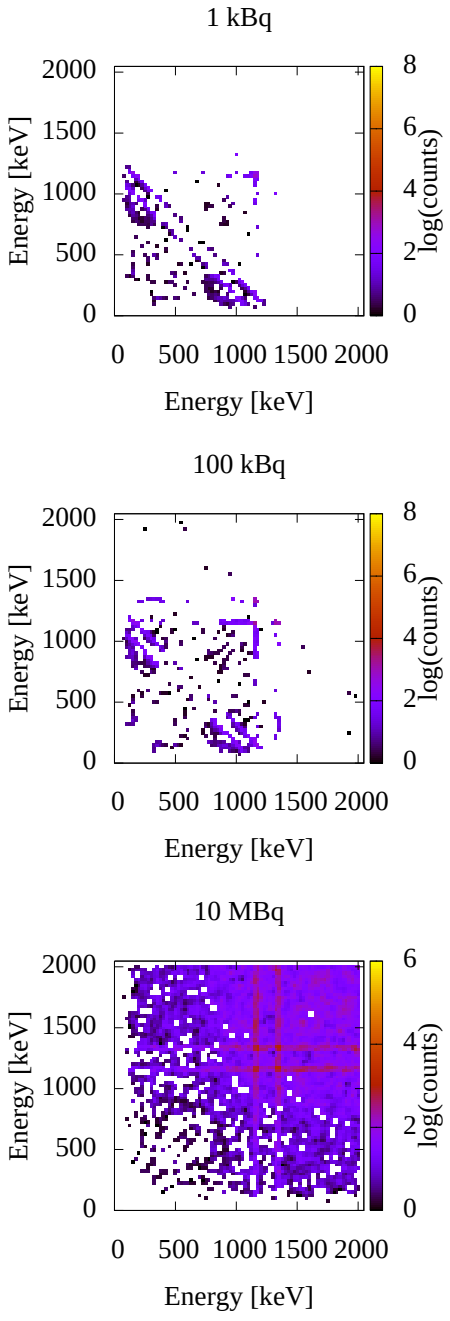

Figure 14: $\gamma\gamma$ matrices for three different ^{60}Co sources.

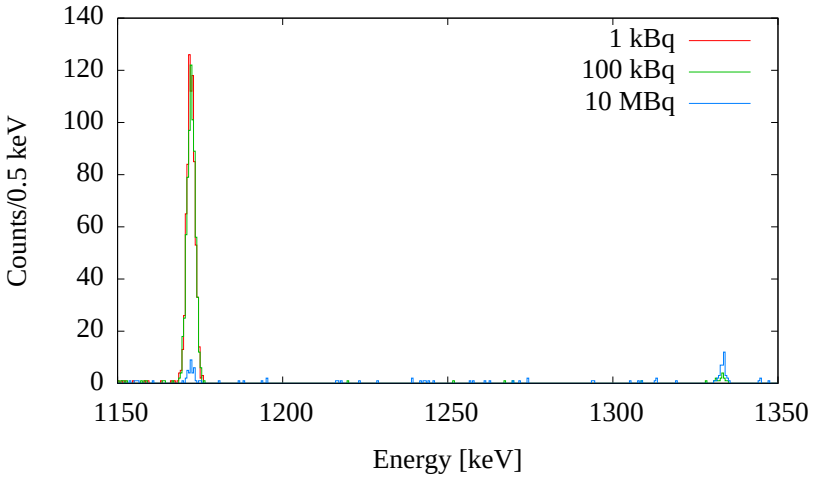

Figure 15: Gate on the 1333 keV γ line in the matrices shown in figure 14. A timegate of $\pm$20 ns has been used.

5.2 Coulex of radioactive beam

This section shows a fairly complex example. Two simulations are run, with the same reaction but different number of particles per second and different number of particles per reaction. The setup consist of 5 ATC and one DSSD detector with 16 segments and 16 rings (this is the KOELN ancillary in the AGATA package). To allow the simulation of the decay of the scattered radioactive beam a target chamber is also included in the simulation. This is all shown in figure 16.

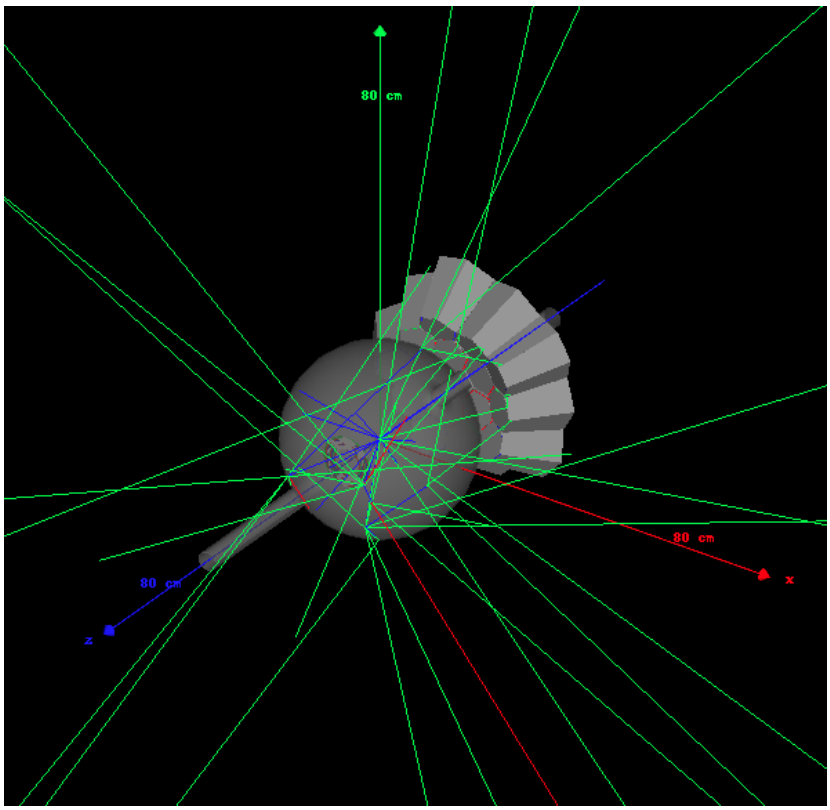

Figure 16: Geant4 simulation of Coloumb excitation of ^{74}Kr, DSSD ancillary and reaction chamber also shown. Beam and recoils are in blue,γ rays are shown as green lines. Positrons are shown as red lines.

The reason for running two simulations is to minimize the computer time needed. The first simulation is run to create events that describes events with Coulomb excitation of the beam. To get descent statistics we need about 10^5 events like that. We therefor simulate events where we have 0.14 particles per second (7 seconds per particle). This way 10^5 events is about a week of beam tiem. If we were to do this based on the cross section we would have to simulate 10^{10} events. On my PC this would correspond to about 2 months of simulation. The second simulation, with 10^4 paricles per second and one reaction/10^4 particles is run for

what corresponds to ten hours of beam time ($3.6 * 10^8$ particles). This data set is then used to create the background. In this example coincidences between AGATA and the DSSD are crucial. Code to handle such a situation is shown in appendix B. The MakeADFEvents code automatically looks for a shared library with the name `lib{ANCNAME}.so`. A trained eye might recognize the experiment that stood model for this simulation.

The needed macros and datafiles are (commented lines in the macros are for the background simulation):

Listing 14: kryptongeometry.mac

```
/Agata/detector/degraderMaterial Vacuum
/grdm/allVolumes
/grdm/verbose 0
/Agata/detector/SetBuildSpectrometer false
/Agata/detector/solidFile A180/A180solid.list
/Agata/detector/wallsFile A180/A180walls.list
/Agata/detector/clustFile A180/A180clust.list
/Agata/detector/sliceFile A180/A180slice.list
/Agata/detector/angleFile A180/A180eulerDemo.list
/Agata/detector/enableCapsules
/Agata/detector/chamberRadius 170
/Agata/detector/chamberMaterial Aluminum
/Agata/detector/enableAncillary
/Agata/detector/ancillary/Koeln/detSize 11 35
/Agata/detector/ancillary/Koeln/detPosition 0 0 25
/Agata/detector/ancillary/Koeln/absoMate Vacuum
/Agata/detector/ancillary/Koeln/targMate Vacuum
/Agata/detector/ancillary/Koeln/detSegments 16 16
/Agata/detector/ancillary/Koeln/detThick 1
/Agata/detector/ancillary/Koeln/detPassThick 0
```

Listing 15: krypton_ruth.mac

```
/control/execute kryptongeometry.mac
/Agata/detector/targetMaterial G4_Pb
/Agata/detector/targetSize 20 20 1
/Agata/generator/emitter/BeamIn/Z 36
/Agata/generator/emitter/BeamIn/A 74
/Agata/generator/emitter/BeamIn/KE 348 MeV
/Agata/generator/emitter/BeamIn/fcZ -65 cm
/Agata/generator/emitter/BeamIn/bDir 0 0
/Agata/generator/emitter/BeamIn/opA .1
/Agata/generator/emitter/BeamIn/spotSize .05 cm
/Agata/generator/emitter/SetParticlePerSeconds 0.14
#/Agata/generator/emitter/SetParticlePerSeconds 10000
/Agata/generator/emitter/SetAccelaratorHF 100 MHz
/Agata/generator/emitter/SetWidthOfBeamPulse 2 ns
/Agata/generator/emitter/BeamOut/DZ -0
/Agata/generator/emitter/BeamOut/DA -0
/Agata/generator/emitter/SetNumberOfParticlesPerReaction 1
#/Agata/generator/emitter/SetNumberOfParticlesPerReaction 70000
/Agata/generator/emitter/BeamOut/Z 82
/Agata/generator/emitter/BeamOut/A 208
/Agata/generator/emitter/BeamOut/ProjectileExcitation 455.8 4 1013.92 1
/grdm/setRadioactiveDecayFile 36 74 decay74Kr
/Agata/generator/emitter/BeamOut/setPtr 0.
/Agata/generator/emitter/BeamOut/setPfe  0.
```

```
/Agata/generator/emitter/BeamOut/setPclx 1.
/Agata/generator/emitter/BeamOut/setPff 0.
/Agata/generator/emitter/BeamOut/adistFile aadist
/Agata/detector/update
/Agata/ScreenedElastic/SetEnergyLimitCreatedByScreenedElastic 0.01 MeV
/Agata/ScreenedElastic/SetEnergyLimitHeavyIons 1 MeV
/Agata/ScreenedElastic/SetEnergyLimitHeavyIonsInSensitiveVolume 1 MeV
/run/initialize
/Agata/file/verbose 1
/Agata/file/enableLM
/Agata/file/info/enableTime
#/Agata/file/WriteToFifo true
/run/beamOn 1000000
#/run/beamOn 360000000
```

Listing 16: decay74Kr

```
#  74KR (11.50 m)
#  Excitation    Halflife Mode        Daughter Ex  Intensity           Q
#
# File sanitized by Vanderbilt Decay-o-matic
# $Id: repair_decay_files.py,v 1.20 2006/04/24 14:53:36 marcus Exp $
# Fri Oct  6 15:26:19 2006
#
P       0.0000    6.9000e+02
                          MshellEC       0.0000    3.5180e-03
                          BetaPlus       0.0000    8.2550e-01
                          KshellEC       0.0000    1.5340e-01
                          LshellEC       0.0000    1.7600e-02
                          MshellEC       9.8400    2.4190e-02    3130.1600
                          MshellEC      72.6200    5.2660e-03    3067.3800
                          MshellEC      89.6200    3.0410e-02    3050.3800
                          MshellEC     132.6000    2.2290e-03    3007.4000
                          MshellEC     179.5000    5.3930e-03    2960.5000
                          MshellEC     212.8600    9.9180e-02    2927.1400
                          MshellEC     239.3200    6.6480e-03    2900.6800
                          MshellEC     306.5500    1.5030e-01    2833.4500
                          MshellEC     390.0600    1.8220e-03    2749.9400
                          MshellEC     534.7000    4.6850e-03    2605.3000
                          MshellEC     609.1100    1.7530e-02    2530.8900
                          MshellEC     612.9000    1.6190e-03    2527.1000
                          MshellEC     701.2800    2.1160e-02    2438.7200
                          MshellEC     831.8000    3.2160e-03    2308.2000
                          MshellEC     970.0000    9.6240e-03    2170.0000
                          MshellEC     978.3000    4.8500e-03    2161.7000
                          BetaPlus       9.8400    8.8000e+00    3130.1600
                          BetaPlus      72.6200    1.7400e+00    3067.3800
                          BetaPlus      89.6200    1.0000e+01    3050.3800
                          BetaPlus     132.6000    6.8000e-01    3007.4000
                          BetaPlus     179.5000    1.5000e+00    2960.5000
                          BetaPlus     212.8600    2.6000e+01    2927.1400
                          BetaPlus     239.3200    1.6700e+00    2900.6800
                          BetaPlus     306.5500    3.4000e+01    2833.4500
                          BetaPlus     390.0600    3.7000e-01    2749.9400
                          BetaPlus     534.7000    6.7000e-01    2605.3000
                          BetaPlus     609.1100    2.2000e+00    2530.8900
                          BetaPlus     612.9000    2.2000e-01    2527.1000
                          BetaPlus     701.2800    2.2000e+00    2438.7200
```

			BetaPlus	831.8000	2.4000e−01	2308.2000
			BetaPlus	970.0000	5.2000e−01	2170.0000
			BetaPlus	978.3000	2.6000e−01	2161.7000
			KshellEC	9.8400	1.0550e+00	3130.1600
			KshellEC	72.6200	2.2940e−01	3067.3800
			KshellEC	89.6200	1.3180e+00	3050.3800
			KshellEC	132.6000	9.6630e−02	3007.4000
			KshellEC	179.5000	2.3730e−01	2960.5000
			KshellEC	212.8600	4.3080e+00	2927.1400
			KshellEC	239.3200	2.9010e−01	2900.6800
			KshellEC	306.5500	6.5940e+00	2833.4500
			KshellEC	390.0600	7.9110e−02	2749.9400
			KshellEC	534.7000	2.0210e−01	2605.3000
			KshellEC	609.1100	7.5640e−01	2530.8900
			KshellEC	612.9000	7.0430e−02	2527.1000
			KshellEC	701.2800	9.1140e−01	2438.7200
			KshellEC	831.8000	1.4070e−01	2308.2000
			KshellEC	970.0000	4.2230e−01	2170.0000
			KshellEC	978.3000	2.1070e−01	2161.7000
			LshellEC	9.8400	1.2050e−01	3130.1600
			LshellEC	72.6200	2.6330e−02	3067.3800
			LshellEC	89.6200	1.5200e−01	3050.3800
			LshellEC	132.6000	1.1140e−02	3007.4000
			LshellEC	179.5000	2.7320e−02	2960.5000
			LshellEC	212.8600	4.9280e−01	2927.1400
			LshellEC	239.3200	3.3240e−02	2900.6800
			LshellEC	306.5500	7.5540e−01	2833.4500
			LshellEC	390.0600	9.0670e−03	2749.9400
			LshellEC	534.7000	2.3240e−02	2605.3000
			LshellEC	609.1100	8.6090e−02	2530.8900
			LshellEC	612.9000	7.9520e−03	2527.1000
			LshellEC	701.2800	1.0740e−01	2438.7200
			LshellEC	831.8000	1.6080e−02	2308.2000
			LshellEC	970.0000	4.8120e−02	2170.0000
			LshellEC	978.3000	2.4490e−02	2161.7000
P	455.8	1.63e−11				
			IT	455.8	100.	
P	1013.92	9.1e−12				
			IT	1013.92	100.	
P	1121.4	0				
			IT	1121.4	100.	
P	1782.03	6.3e−13				
			IT	1782.03	100.	
P	1942.7	0				
			IT	1942.7	100.	
P	2616.07	0				
			IT	2616.07	100.	
P	2657.82	0				
			IT	2657.82	100.	
P	2748.94	1.8e−13				
			IT	2748.94	100.	
P	2814.03	0				
			IT	2814.03	100.	
P	3141.95	0				
			IT	3141.95	100.	
P	3368.86	0				
			IT	3368.86	100.	

```
P        3843.35    0
                        IT        3843.35        100.
P        3893.75    6.9e-14
                        IT        3893.75        100.
P         4134.9    0
                        IT         4134.9        100.
P        4724.85    0
                        IT        4724.85        100.
P         5088.9    0
                        IT         5088.9        100.
P        5180.86    1.2e-13
                        IT        5180.86        100.
P         5767.9    0
                        IT         5767.9        100.
P         6213.9    0
                        IT         6213.9        100.
P         6514.9    1.4e-13
                        IT         6514.9        100.
P         6970.9    0
                        IT         6970.9        100.
P         7493.9    0
                        IT         7493.9        100.
P         7856.9    0
                        IT         7856.9        100.
P         8320.9    0
                        IT         8320.9        100.
P         9302.9    0
                        IT         9302.9        100.
P          10863    0
                        IT          10863        100.
```

In figure 17 is the differential cross section used to sample the angle of the outgoing particles shown. The simulation is then run in the same manner as for the ^{60}Co source:

```
shell> Agata -Gen -n -SN -a 1 1 -b krypton_ruth.mac
```

in one terminal window. In another terminal window (at the same path)

```
shell> cat GammaEvents.fifo | PARTICLEID=-8 MakeADFEvents
```

The "`PARTICLEID=-8`" is a system variable telling `MakeADFEvents` what type of particles to look for. The output from the simulations, 16 ADF files, 15 for the crystals and one for the DSSD detector, are stored in a typical "AGATA" hierarchy (i.e. `./Data/1R ... ./Data/Ancillary`). The data is " tracked" using the AGATA narval emulator femul, and transformed into ROOT trees using the GammaWare package. The details of this, if it is your first time, potentially frustrating process in out of scope of this document.

Figure 18 showw the energy of the particle detected in the DSSD detector. This information is used to supress background using coincidences and to give the information needed for the Doppler correction.

With each beam particle reacting in the target and 0.14 pps time spectra and γ spectra can be generated. In figure 19 the distribution of times between events in the DSSD and tracked γ rays is shown. Only the peak at zero is real coincidences, all other peaks are random

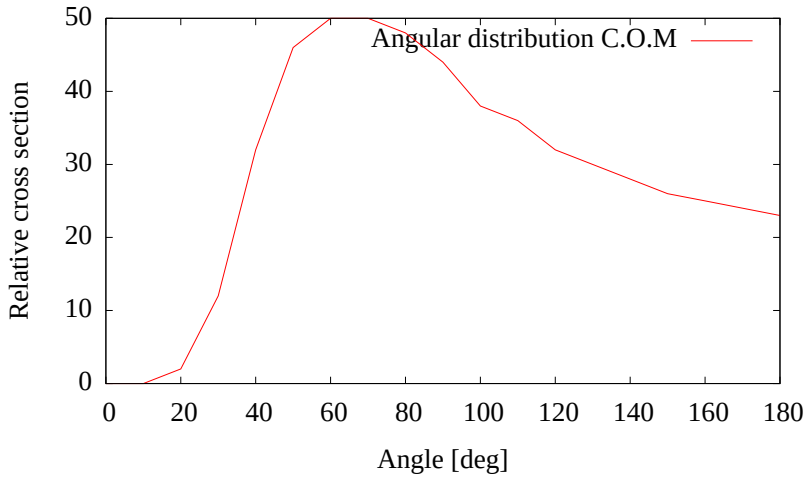

Figure 17: Relative cross section for the Coulomb excitation process used.

coincidences between γ rays from β decay and the beam particles arriving every 7 seconds. Using this information we can gate on events coming from a true Coulomb excitation of a ^{74}Kr event. The effect of this is shown in figure 20 together with the Doppler Correction capabilities of this setup.

However, the above does not correctly represent how a experiment would look like because we miss about 100000 particles that has not passed the target, and hence not do not contribute to neither the β decay background nor to rutherford scattering into the DSSD.

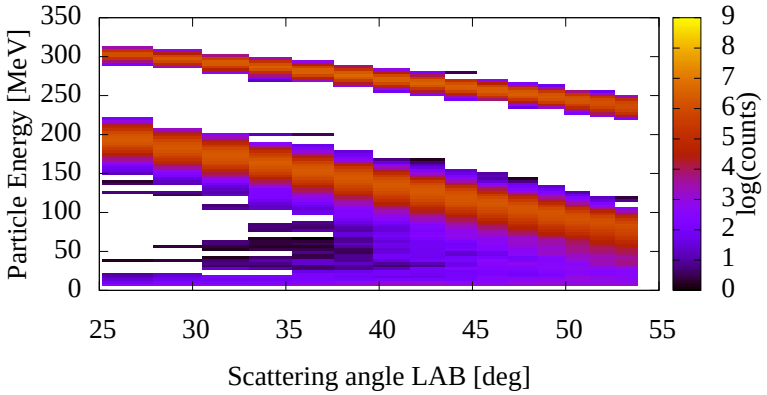

Figure 18: Kinematics for ^{74}Kr (upper band) and ^{208}Pb recoils (lower band).

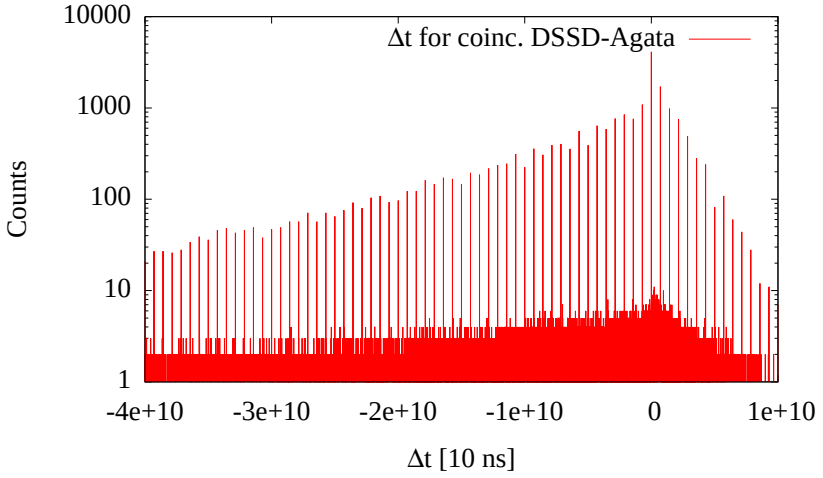

Figure 19: Δt for coincidences between the DSSD and Agata. Notice that only the peak at $\Delta t = 0$ contains real coincidences. The other peaks come from coincidences between γ rays from β decay and rutherford scattered beam particles.

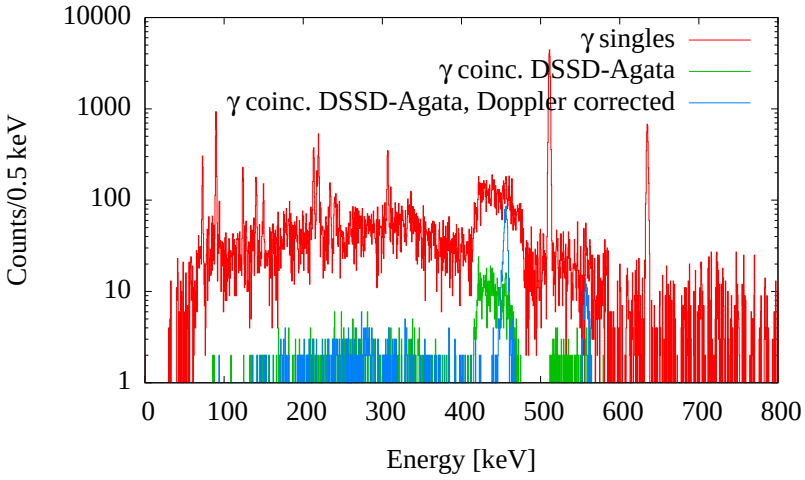

Figure 20: Gamma-ray spectra with and without coincidences condition between the DSSD and Agata.

6 Suggested to-do list

TEST TEST TEST and DEBUG DEBUG DEBUG

once that is done...

Reactions

- Allow for several open channels in HI Fusions reactions
- Target excition in Coulex (there is a primitve variant already implemented since long time)
- Allow competning reaction mechnism with different final products

Detectors

- VAMOS, i.e. particle ray-tracing, with conversion to correct file format to be used at GANIL
- Neutron Wall...
- ...

A MakeADFEvents

/Users/joa/home2/minaROOTklasser/AGATA/MakeADFEvents.cxx

```
/*
  Code to make adf files from AGATA geant4 simulation output.  It
  works either based on abosulte time (if that information is aviable)
  or on event by event basis (event start token in data file - not yet
  implemented!)

  Event are created either by taking data in time windows or in each event.
  These event are stored in temporary structures, and handled in "groups"
  corresponding to "events" in the real world.

  This data is then packed, smeared etc and stored in adf frames.
 */

#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <sstream>
#include <vector>
#include <iomanip>
#include <set>
#include <cmath>
#include <algorithm>

#include "TVector3.h"
```

```
#include "TMath.h"
#include "TGeoMatrix.h"
#include "TRandom.h"

#ifndef __CINT__
//ADF
#include "./include/ConfAgent.h"
#include "./include/DFAgent.h"
#include "./include/AgataKeyFactory.h"
#include "./include/AgataFrameFactory.h"
#include "./include/BufferIO.h"
#include "./include/FrameIO.h"
#include "./include/ConfigurationFrame.h"

#include <dlfcn.h>


#ifndef ADF_CrystalFrame
#include "./include/CrystalFrame.h"
#endif
#ifndef ADF_PSAFrame
#include "./include/PSAFrame.h"
#endif
#ifndef ADF_TrackedFrame
#include "./include/TrackedFrame.h"
#endif
#ifndef ADF_AgataCompositeFrame
#include "./include/AgataCompositeFrame.h"
#endif
#include "./include/Trigger.h"

#endif

struct agatacluster
{
  //euler angles of det in cluster
  double psi[3], theta[3], phi[3];

  //Postistion of det. in cluster
  TVector3 d[3];

  //Which crystals in this cluster
  int crystalnumber[3];
  //What type of crystals they are
  int crystaltype[3];

  // Which cluster
  int clusternumber;
};

struct clusterpos

{
  //Eulerangles of cluster
  double psi, theta, phi;

  // Position of cluster
```

```
  TVector3 d;

  //Which crystals that are in this cluster
  int crystals[3];

  // In case there is more than one typ of cluster
  int clusternumber;
};


struct crystaltransformationinfo

{
  TGeoHMatrix *transformation;
  int crystaltype;
};




class Interactions {
public:
  Interactions(){;}
  ~Interactions(){;}
  void push_back(Double_t x, Double_t y, Double_t z, Double_t e,
     Int_t det, Int_t seg)
  {
    posx.push_back(x);
    posy.push_back(y);
    posz.push_back(z);
    energy.push_back(e);
    detnumber.push_back(det);
    segnumber.push_back(seg);
  }
  void clear()
  {
    posx.clear();
    posy.clear();
    posz.clear();
    energy.clear();
    detnumber.clear();
    segnumber.clear();
  }
  std::vector<Double_t> posx,posy,posz,energy;
  std::vector<Int_t> detnumber, segnumber;
  unsigned int size() {return posx.size();}
};


class TimeInfo {
public:
  TimeInfo(){;}
  TimeInfo(Int_t d,Int_t h,Int_t m,Double_t t) : detT(t),day(d),hour(h),
              min(m) {;}
  ~TimeInfo(){;}
  Double_t detT;
  Int_t day,hour,min;
```

```
  void Set(Int_t d,Int_t h,Int_t m,Double_t t){
    day=d; hour=h; min=m;
    detT=t;
    while(detT>=6e10){
      detT-=6e10;
      min++;
    }
    while(min>=60){
      min-=60;
      hour++;
    }
    while(hour>=24){
      hour-=24;
      day++;
    }
  }
  Double_t operator-(const TimeInfo &S) const {//Return the difference in ns
    Double_t dday=(day-S.day);
    Double_t dhour=(hour-S.hour);
    Double_t dmin=(min-S.min);
    Double_t dT=detT-S.detT;
    return (1e9*(dday*24.*3600.+dhour*3600.+dmin*60.)+dT);
  }
};

class Interaction {
public:
  Interaction(){
    time.Set(0,0,0,0);
  }
  Interaction(Double_t x, Double_t y, Double_t z, Double_t e,
        Int_t det, Int_t seg, Int_t day, Int_t hour, Int_t min,
        Double_t detT)
  {
    posx=x;
    posy=y;
    posz=z;
    energy=e;
    detnumber=det;
    segnumber=seg;
    time.Set(day,hour,min,detT);
  }
  ~Interaction(){;}
  void Set(Double_t x, Double_t y, Double_t z, Double_t e,
     Int_t det, Int_t seg, Int_t day, Int_t hour, Int_t min,
     Double_t detT)
  {
    posx=x;
    posy=y;
    posz=z;
    energy=e;
    detnumber=det;
    segnumber=seg;
    time.Set(day,hour,min,detT);
  }
  bool operator()(const Interaction &lhs, const Interaction &rhs)
  {
```

```
    return (lhs.time-rhs.time)<0;
  }
  Double_t posx,posy,posz,energy;
  Int_t detnumber, segnumber;
  TimeInfo time;
  static void Print(const Interaction &I, std::ostream&str=std::cout) {
    str << I.posx << " " << I.posy << " " << I.posz << " "
  << I.energy  << " " << I.detnumber << " "
  << I.segnumber << " " << I.time.day << " " << I.time.hour
  << " " << I.time.min << " " << I.time.detT << std::endl;
  }
};

template <class T> class event {
public:
  event(){;}
  ~event(){;}
  event(const T &data,Int_t da,Int_t h,Int_t m,Double_t t)
  {detData=data; day=da;hour=h;min=m;detT=t;}
  std::vector<T> detData;
  Double_t detT;
  Int_t day,hour,min;
  ULong64_t GetTimeStamp() const {
    return 1e8*(((day*24+hour)*60+min)*60)+detT/10;
  }
  bool operator()(const event &lhs,const event &rhs)
  {
    Int_t rhsday=rhs.day;
    Int_t rhshour=rhs.hour;
    Int_t rhsmin=rhs.min;
    Double_t rhst=rhs.detT;
    Int_t lhsday=lhs.day;
    Int_t lhshour=lhs.hour;
    Int_t lhsmin=lhs.min;
    Double_t lhst=lhs.detT;
    while(rhst>=6e10){
      rhst-=6e10;
      rhsmin++;
    }
    while(rhsmin>=60){
      rhsmin-=60;
      rhshour++;
    }
    while(rhshour>=24){
      rhshour-=24;
      rhsday++;
    }
    while(lhst>=6e10){
      lhst-=6e10;
      lhsmin++;
    }
    while(lhsmin>=60){
      lhsmin-=60;
      lhshour++;
    }
    while(lhshour>=24){
      lhshour-=24;
```

```
      lhsday++;
    }
    if(lhsday!=rhsday) return lhsday<rhsday;
    if(lhshour!=rhshour) return lhshour<rhshour;
    if(lhsmin!=rhsmin) return lhsmin<rhsmin;
    return lhst<rhst;
  }
  void Print() const {
    std::cout << day << " " << hour << " " << min << " " << detT
        << " ";
  }
};


class SantasLittleHelper {
public:
  SantasLittleHelper(){;}
  ~SantasLittleHelper(){;}
  static Interactions *InteractionsInEvent;
  static std::ostringstream *buffer;
  static void AddData(const Interaction &data){
    InteractionsInEvent->push_back(data.posx,data.posy,data.posz,data.energy,
     data.detnumber,data.segnumber);
  }
  static void StreamToBuffer(const Interaction &data){
    Interaction::Print(data,*buffer);
  }
};

Interactions *SantasLittleHelper::InteractionsInEvent=0;
std::ostringstream *SantasLittleHelper::buffer=0;

int packpoints(int number, double posx[], double posy[], double posz[],
         double energy[], int detnumb[], int segnumb[]);

int packpointsbarycenter(int number, double posx[], double posy[],
       double posz[], double energy[], int detnumb[],
       int segnumb[]);



typedef void (*AncHandler)(std::ostringstream &);

void MakeDetectorTransformations(std::vector<clusterpos>
         &theclusterpostions,
         std::vector<agatacluster> theclusters,
         std::vector<crystaltransformationinfo>
         &thecrystaltransformations)

{
  std::vector<clusterpos>::iterator ittheclusterpostions;
  for (ittheclusterpostions = theclusterpostions.begin();
       ittheclusterpostions!= theclusterpostions.end();
       ittheclusterpostions++){
    clusterpos *cp = &(*(ittheclusterpostions));
    // Here pick the correct cluster number
    std::vector<agatacluster>::iterator ittheclusters;
```

```
    for (ittheclusters = theclusters.begin();
   ittheclusters!=theclusters.end();
   ittheclusters++){
      agatacluster lacluster = *(ittheclusters);
      if (cp->clusternumber == lacluster.clusternumber){
  // First the cluster transformation
  // All rotation matricies are made by hand due too different
  // convention used for the euler angles between ROOT (z x' z'')
  // and geom. input files (z y' z'') :(
  double matrix[9];
  double cosphi,sinphi,costheta,sintheta,cospsi,sinpsi;
  // Rotation of cluster rel. lab
  cosphi =   cos(cp->phi);
  sinphi =   sin(cp->phi);
  costheta = cos(cp->theta);
  sintheta = sin(cp->theta);
  cospsi =   cos(cp->psi);
  sinpsi =   sin(cp->psi);
  matrix[0] = -sinpsi*sinphi + costheta*cosphi*cospsi;
  matrix[3] = sinpsi*cosphi + costheta*sinphi*cospsi;
  matrix[6] = -cospsi*sintheta;
  matrix[1] = -cospsi*sinphi - costheta*cosphi*sinpsi;
  matrix[4] = cospsi*cosphi - costheta*sinphi*sinpsi;
  matrix[7] = sinpsi*sintheta;
  matrix[2] = sintheta*cosphi;
  matrix[5] = sintheta*sinphi;
  matrix[8] = costheta;
  TGeoRotation clusterrotation;
  clusterrotation.SetMatrix(matrix);
  //Translation of cluster rel lab
  TGeoTranslation clustertranslation(cp->d[0],cp->d[1],cp->d[2]);
  TGeoCombiTrans clustercombination(clustertranslation,
           clusterrotation);
  TGeoRotation crystalrotation;
  TGeoTranslation crystaltranslation;
  TGeoCombiTrans crystalcombination;
  TGeoHMatrix m;
  for (int i=0; i<3; i++){
    // Here we actually put the crystals in position
    // Calculate rotation matrix for crystal rel. cluster...
    cosphi   = cos(lacluster.phi[i]);
    sinphi   = sin(lacluster.phi[i]);
    costheta = cos(lacluster.theta[i]);
    sintheta = sin(lacluster.theta[i]);
    cospsi   = cos(lacluster.psi[i]);
    sinpsi   = sin(lacluster.psi[i]);
    matrix[0] = -sinpsi*sinphi + costheta*cosphi*cospsi;
    matrix[3] = sinpsi*cosphi + costheta*sinphi*cospsi;
    matrix[6] = -cospsi*sintheta;
    matrix[1] = -cospsi*sinphi - costheta*cosphi*sinpsi;
    matrix[4] = cospsi*cosphi - costheta*sinphi*sinpsi;
    matrix[7] = sinpsi*sintheta;
    matrix[2] = sintheta*cosphi;
    matrix[5] = sintheta*sinphi;
    matrix[8] = costheta;
    crystalrotation.SetMatrix(matrix);
    // Translation of crystal rel cluster
```

```
    crystaltranslation.SetTranslation((lacluster.d[i])[0],
              (lacluster.d[i])[1],
              (lacluster.d[i])[2]);
    crystalcombination.SetTranslation(crystaltranslation);
    crystalcombination.SetRotation(crystalrotation);
    // Make total transformation for a crystal
    m = clustercombination * crystalcombination;
    TGeoHMatrix *real = new TGeoHMatrix(m);
    crystaltransformationinfo temp;
    temp.transformation = real;
    temp.crystaltype = lacluster.crystaltype[i];
    // Keep the transformation
    thecrystaltransformations.push_back(temp);
    cp->crystals[i] = thecrystaltransformations.size()-1;
  }
  break;
      }
    }
  }
}


class TreatOneEvent {
public:
  TreatOneEvent(ADF::Frame *cframe_,
    ADF::ConfigurationFrame *gconfframe_,
    ADF::APSAFrame      *fFramePSA_,
    ADF::RawFrame *fFrameAnc_,
    ADF::RawFrame *fFrameAnc1_,
    UShort_t &crystal_id_,UShort_t &crystal_status_,
    Float_t &CoreE0_,Float_t &CoreE1_,Float_t &CoreT0_,
    Float_t &CoreT1_,
    std::vector<crystaltransformationinfo>
    &thecrystaltransformations_,
    std::vector<agatacluster> &theclusters_,
    std::vector<clusterpos>   &theclusterpostions_,
    std::map<int,std::vector<int> > &listofinteractions_,
    std::map<int,int> &listofinteractionsperdetector_,
    std::map<int,int> &listoframesperdetector_,
    agatacluster *acluster_,
    clusterpos *apos_,
    std::map<int,std::ofstream *> &outputfiles_,
    std::map<Int_t,ofstream *> &ancoutput_,
    std::map<Int_t,std::string> &ancillarymap_,
    std::map<Int_t,AncHandler> &ancillaryhandlers_,
    Interactions &InteractionsInEvent_,bool addres_,
    bool transformtocrystal_, std::string baseoutput_,
    unsigned long int &evtnb_,
    std::map<int,double> &energy_wrote_out_,
    std::map<int,int> &interactions_out_) :
    cframe(cframe_),gconfframe(gconfframe_),fFramePSA(fFramePSA_),
    fFrameAnc(fFrameAnc_),fFrameAnc1(fFrameAnc1_),crystal_id(crystal_id_),
    crystal_status(crystal_status_),CoreE0(CoreE0_),
    CoreE1(CoreE1_),CoreT0(CoreT0_),CoreT1(CoreT1_),
    thecrystaltransformations(thecrystaltransformations_),
    theclusters(theclusters_) ,theclusterpostions(theclusterpostions_),
    listofinteractions(listofinteractions_),
```

```
    listofinteractionsperdetector(listofinteractionsperdetector_),
    listoframesperdetector(listoframesperdetector_),acluster(acluster_),
    apos(apos_),outputfiles(outputfiles_),ancoutput(ancoutput_),
    ancillarymap(ancillarymap_), ancillaryhandlers(ancillaryhandlers_),
    InteractionsInEvent(InteractionsInEvent_),addres(addres_),
    transformtocrystal(transformtocrystal_),baseoutput(baseoutput_),
    evtnb(evtnb_),energy_wrote_out(energy_wrote_out_),
    interactions_out(interactions_out_)  {;}
  ~TreatOneEvent() {;}
  void Treat(std::multiset<event<Interaction>,event<Interaction> >::iterator
       itStartEventQue);
private:
  ADF::Frame *cframe;
  ADF::ConfigurationFrame *gconfframe;
  ADF::APSAFrame      *fFramePSA;
  ADF::RawFrame *fFrameAnc;
  ADF::RawFrame *fFrameAnc1;
  UShort_t &crystal_id,&crystal_status;
  Float_t &CoreE0,&CoreE1,&CoreT0,&CoreT1;
  std::vector<crystaltransformationinfo> &thecrystaltransformations;
  std::vector<agatacluster> &theclusters;
  std::vector<clusterpos>   &theclusterpostions;
  std::map<int,std::vector<int> > &listofinteractions;
  std::map<int,int> &listofinteractionsperdetector;
  std::map<int,int> &listoframesperdetector;
  agatacluster *acluster;
  clusterpos *apos;
  std::map<int,std::ofstream *> &outputfiles;
  std::map<Int_t,ofstream *> &ancoutput;
  std::map<Int_t,std::string> &ancillarymap;
  std::map<Int_t,AncHandler> &ancillaryhandlers;
  Interactions &InteractionsInEvent;
  bool addres,transformtocrystal;
  std::string baseoutput;
  unsigned long int &evtnb;
  std::map<int,double> &energy_wrote_out;
  std::map<int,int> &interactions_out;
};



void TreatOneEvent::Treat(std::multiset<event<Interaction>,
          event<Interaction> >::iterator
itStartEventQue)

{
  //It is here I should make the adf frames,
  //We start to do it only for AGATA crystals
  if(itStartEventQue->detData.begin()->detnumber<180){
    //Start with packing points
    InteractionsInEvent.clear();
    SantasLittleHelper::InteractionsInEvent = &InteractionsInEvent;
    std::for_each(itStartEventQue->detData.begin(),
      itStartEventQue->detData.end(),
      SantasLittleHelper::AddData);
    //Pack points...
    interactions_out[itStartEventQue->detData.begin()->detnumber]+=
```

```
      InteractionsInEvent.size();
    Int_t numberofinteractions =
      packpointsbarycenter(InteractionsInEvent.size(),
         &InteractionsInEvent.posx[0],
         &InteractionsInEvent.posy[0],
         &InteractionsInEvent.posz[0],
         &InteractionsInEvent.energy[0],
         &InteractionsInEvent.detnumber[0],
         &InteractionsInEvent.segnumber[0]);
    //Smear energy and pos res...
    for(Int_t i=0; i<numberofinteractions; i++){
      /*error_p = 0.5*sqrt(0.1/energy[i])/2.35;
fwhm/2.35 in cm
this is from NIM A 638 PA Sderstrm et al.*/
      double energy=InteractionsInEvent.energy[i];
      if(addres){
Double_t resfactor=0.62/2.35/sqrt(3);
InteractionsInEvent.posx[i]+=
  gRandom->Gaus(0,10.*(.27+resfactor*sqrt(0.1/energy)));
InteractionsInEvent.posy[i]+=
  gRandom->Gaus(0,10.*(.27+resfactor*sqrt(0.1/energy)));
InteractionsInEvent.posz[i]+=
  gRandom->Gaus(0,10.*(.27+resfactor*sqrt(0.1/energy)));
      }
      energy_wrote_out[InteractionsInEvent.detnumber[i]]+=
InteractionsInEvent.energy[i];
      InteractionsInEvent.energy[i]+=
gRandom->Gaus(0,sqrt(5305./5326+energy*21./5326)/2.35);
      //We also tranform to crystal refernce if asked for
      if(transformtocrystal){
double r[3],rp[3];
rp[0]=InteractionsInEvent.posx[i];
rp[1]=InteractionsInEvent.posy[i];
rp[2]=InteractionsInEvent.posz[i];
thecrystaltransformations[InteractionsInEvent.detnumber[i]].
  transformation->MasterToLocal(rp,r);
InteractionsInEvent.posx[i]=r[0];
InteractionsInEvent.posy[i]=r[1];
InteractionsInEvent.posz[i]=r[2];
      }
    }
    //Here we fill the event frame with PSA frames...
    //We need to fill crystal per crystal...
    listofinteractions.clear();
    for(Int_t i=0; i<numberofinteractions; i++){
      if(outputfiles.find(InteractionsInEvent.detnumber[i])==
 outputfiles.end()){//open output file if needed...
std::string det[3]={"R","G","B"};
std::ostringstream filename;
filename << "Data/" << (InteractionsInEvent.detnumber[i]/3)+1
    << det[(InteractionsInEvent.detnumber[i]%3)]
    << "/" << baseoutput
    << (InteractionsInEvent.detnumber[i]/3)+1
    << det[(InteractionsInEvent.detnumber[i]%3)]
    << "_" << std::setw(4)
    << std::setfill('0')
    << 0
```

```
    << ".adf";
outputfiles[InteractionsInEvent.detnumber[i]]=
  new std::ofstream(filename.str().c_str());
//We also dump a config frame on the stream...
outputfiles[InteractionsInEvent.detnumber[i]]->
  write((char*)gconfframe->GetKey()->GetRealBuffer()->
  GetAddress(),
  gconfframe->GetKey()->GetKeyLength());
outputfiles[InteractionsInEvent.detnumber[i]]->
  write((char*)gconfframe->GetRealBuffer()->GetAddress(),
  gconfframe->GetKey()->GetDataLength());
    }
    listofinteractions[InteractionsInEvent.detnumber[i]].
push_back(i);
  }
  std::map<int,std::vector<int> >::iterator itlist
    = listofinteractions.begin();
  for(;itlist!=listofinteractions.end(); ++itlist){
    listofinteractionsperdetector[itlist->first]+=
itlist->second.size();
    listoframesperdetector[itlist->first]++;
    fFramePSA->Reset();
    crystal_id=itlist->first;
    crystal_status=0;
    CoreE0 = 0;
    CoreE1 = 0;
    CoreT0 = 0;
    CoreT1 = 0;
    ((ADF::AgataKey*)fFramePSA->GetKey())->SetEventNumber(evtnb);
    ((ADF::AgataKey*)fFramePSA->GetKey())->
SetTimeStamp(itStartEventQue->GetTimeStamp());
    std::vector<int>::iterator iti = itlist->second.begin();
    for(; iti!=itlist->second.end(); ++iti){
CoreE0+=InteractionsInEvent.energy[*iti];
CoreE1+=InteractionsInEvent.energy[*iti];
ADF::PSAHit *hit = (ADF::PSAHit*)fFramePSA->Data()->NewHit();
hit->Reset();
hit->SetE(InteractionsInEvent.energy[*iti]);
hit->SetXYZ(InteractionsInEvent.posx[*iti],
      InteractionsInEvent.posy[*iti],
      InteractionsInEvent.posz[*iti]);
hit->SetID(InteractionsInEvent.segnumber[*iti]);
hit->SetT(0);
hit->SetDT(1.);  // chi2 temporarily stored in
    }
    fFramePSA->Write();
    //Write this to the correct file
    std::ofstream *fpointer=outputfiles[itlist->first];
    fpointer->write((char*)fFramePSA->GetKey()->GetRealBuffer()->
        GetAddress(),
        fFramePSA->GetKey()->GetKeyLength());
    fpointer->write((char*)fFramePSA->GetRealBuffer()->
        GetAddress(),
        fFramePSA->GetKey()->GetDataLength());

  }
} else {//So this is an ancillary...
```

```
    Int_t ancnb =
      1000*(itStartEventQue->detData.begin()->detnumber/1000);
    if(ancoutput.find(ancnb)==ancoutput.end()){
      std::ostringstream filename;
      filename << "Data/Ancillary/Anc"
         << "_" << ancillarymap[ancnb]
         << "_" << std::setw(4)
         << std::setfill('0')
         << 0
         << ".adf";
      ancoutput[ancnb]=new std::ofstream(filename.str().c_str());
    }
    std::ostringstream tmpbf;
    ADF::RawFrame *frameanc=fFrameAnc;
    SantasLittleHelper::buffer=&tmpbf;
    std::for_each(itStartEventQue->detData.begin(),
      itStartEventQue->detData.end(),
      SantasLittleHelper::StreamToBuffer);
    if(ancillaryhandlers.find(ancnb)!=
       ancillaryhandlers.end()){
      (ancillaryhandlers[ancnb])(tmpbf);
      frameanc=fFrameAnc1;
    }
    ((ADF::AgataKey*)frameanc->GetKey())->SetEventNumber(evtnb);
    ((ADF::AgataKey*)frameanc->GetKey())->
      SetTimeStamp(itStartEventQue->GetTimeStamp());
    if((*(unsigned int*)tmpbf.str().c_str())>0){
      ((ADF::AgataKey*)frameanc->GetKey())->
  SetDataLength((UInt_t)SantasLittleHelper::buffer->
          str().size());
      ancoutput[ancnb]->write((char*)frameanc->GetKey()->
            GetRealBuffer()->
            GetAddress(),
            frameanc->GetKey()->GetKeyLength());
      ancoutput[ancnb]->write((char*)SantasLittleHelper::buffer->
            str().c_str(),
            frameanc->GetKey()->
            GetDataLength());
    }
  }
#ifdef       _DEBUG_
  itStartEventQue->Print();
  std::cout << std::endl;
  std::for_each(itStartEventQue->detData.begin(),
    itStartEventQue->detData.end(),
    Interaction::Print);
  std::cout << "<->\n";
#endif
}



int MakeADFEvents(std::string baseoutput="PSA_",
      bool transformtocrystal=true,bool addres=true,
      Double_t timegate = 10,
      /*micros, summing in a det, if Dt larger, considered as new
        event*/
```

```
      std::string particleid = "-1"
      /* particle to look for when checking event time*/)

{
  //  sleep(5000);
  /**
     Here we will define ADF things...
  */
  // To customise
  // the factory for everything
#ifndef __CINT__
  gRandom->SetSeed(time(0));
  ADF::ConfAgent::theGlobalAgent();
  const char factory[] = "Agata";
  // the two types of embedded frames, the first one should stay data:psa
  const char frame_type1[] = "data:psa";
  // the version of the primary key and the frames to be embedded
  ADF::Version key_v(4,0), frame_v1(65000,1);
  // end to customize ... works without any changes after
  ADF::FactoryItem prikeydef1(factory,frame_type1,key_v);
  ADF::FactoryItem aframedef1(factory,frame_type1,frame_v1);
  // Ask global agent to fill a Configuration frame which is dumped
  ADF::Frame *cframe = NULL;
  ADF::ConfigurationFrame *gconfframe = NULL;
  ADF::APSAFrame      *fFramePSA;
  ADF::AgataFrameTrigger fTrigger("data:psa");
  if(!ADF::MainFrameFactory::theMainFactory().
     IsKnown(aframedef1.GetFactoryName().data()) ){
    std::cout << " Agata Factory not properly loaded " << std::endl;
    return 1;
  }
  fFramePSA=dynamic_cast<ADF::APSAFrame*>
    (ADF::MainFrameFactory::theMainFactory().New(prikeydef1,aframedef1));
  cframe=ADF::MainFrameFactory::theMainFactory().
    New(ADF::FactoryItem("Agata","conf:global",key_v),
  ADF::FactoryItem("Agata","conf:global",frame_v1));
  if (cframe) gconfframe = dynamic_cast<ADF::ConfigurationFrame *>(cframe);
  // change the agent to take into account this configuration
  ADF::ConfAgent::theGlobalAgent()->GetDFAgent()->
    SetComment("from geant4 AGATA sim");
  ADF::ConfAgent::theGlobalAgent()->GetDFAgent()->
    AddKnownFrame(fFramePSA,'+');
  ADF::ConfAgent::theGlobalAgent()->GetDFAgent()->
    Configure(gconfframe,"out");
  UShort_t crystal_id,crystal_status;
  Float_t CoreE0,CoreE1,CoreT0,CoreT1;
  ADF::GObject *glob = fFramePSA->Data()->Global();
  glob->LinkItem("CrystalID",     &crystal_id);
  glob->LinkItem("CrystalStatus", &crystal_status);
  glob->LinkItem("CoreE0",        &CoreE0);
  glob->LinkItem("CoreE1",        &CoreE1);
  glob->LinkItem("CoreT0",        &CoreT0);
  glob->LinkItem("CoreT1",        &CoreT1);
  //Anc frame
  const char frame_type_anc[] = "data:ranc0";
  ADF::Version key_anc(4,0), frame_anc(65000,0);
  // end to customize ... works without any changes after
```

```
  ADF::FactoryItem prikeyanc(factory,frame_type_anc,key_anc);
  ADF::FactoryItem aframeanc(factory,frame_type_anc,frame_anc);
  ADF::RawFrame *fFrameAnc=dynamic_cast<ADF::RawFrame*>
    (ADF::MainFrameFactory::theMainFactory().New(prikeyanc,aframeanc));
  const char frame_type_anc1[] = "data:ranc1";
  ADF::Version key_anc1(4,0), frame_anc1(65000,0);
  // end to customize ... works without any changes after
  ADF::FactoryItem prikeyanc1(factory,frame_type_anc1,key_anc1);
  ADF::FactoryItem aframeanc1(factory,frame_type_anc1,frame_anc1);
  ADF::RawFrame *fFrameAnc1=dynamic_cast<ADF::RawFrame*>
    (ADF::MainFrameFactory::theMainFactory().New(prikeyanc1,aframeanc1));
#endif
  std::vector<crystaltransformationinfo> thecrystaltransformations;
  std::vector<agatacluster> theclusters;
  std::vector<clusterpos>   theclusterpostions;
  theclusterpostions.resize(60);
  std::map<int,std::vector<int> > listofinteractions;
  std::map<int,int> listofinteractionsperdetector;
  std::map<int,int> listoffframesperdetector;
  std::map<int,double> energy_read_in;
  std::map<int,double> energy_wrote_out;
  std::map<int,int> interactions_in;
  std::map<int,int> interactions_out;
  int ctr = 0;
  agatacluster *acluster = 0;
  clusterpos *apos = 0;
  std::map<int,std::ofstream *> outputfiles;
  std::map<Int_t,ofstream *> ancoutput;
  std::map<Int_t,std::string> ancillarymap;
  std::map<Int_t,AncHandler> ancillaryhandlers;

  //Check OUTPUT_MASK
  std::string keyword,value;
  char oneline[1024];
  bool UseTimeForTimestamps = false;
  do{
    std::cin.getline(oneline,1024);
    ctr++;
    std::istringstream ss(oneline);
    ss >> keyword >> value;
  }while(keyword!="OUTPUT_MASK");
  std::cout << keyword << " " << value << "\n";
  if(value[2]!='1'){
    std::cout << "We use abosult positions!!!";
    //    return 1;
  }
  if(value[6]=='1'){
    UseTimeForTimestamps=true;
    std::cout << "We use event time for timestamps!!!\n";
  }
  //Get number of detectors
  int numberofdet = 0;
  double rinner,router;
  do{
    std::cin.getline(oneline,1024);
    ctr++;
    std::istringstream ss(oneline);
```

```
    ss >> keyword >> rinner >> router >> numberofdet;
  }while(keyword!="SUMMARY");
  std::cout << keyword << " " << numberofdet << "\n";
  do{
    std::cin.getline(oneline,1024);
    ctr++;
    std::istringstream ss(oneline);
    ss >> keyword;
    if(keyword=="ANCIL"){
      Int_t ancnb,whichan;std::string ancname;
      ss >> ancname >> whichan >> ancnb;
      ancillarymap[ancnb]=ancname;
      //Here check if we can find a anc library
      void *lib_handel;
      std::ostringstream libname;
      libname << "lib" << ancname << ".so";
      lib_handel = dlopen(libname.str().c_str(),RTLD_LAZY);
      if(lib_handel){
  AncHandler oneh=
    (void(*)(std::ostringstream&))dlsym(lib_handel,ancname.c_str());
  if(oneh) ancillaryhandlers[ancnb]=oneh;
      }
    }
  }while(keyword!="SOLID");
  do{
    std::cin.getline(oneline,1024);
    ctr++;
    std::istringstream ss(oneline);
    ss >> keyword >> value;
  }while(keyword!="CLUSTER");
  std::cout << keyword << " " << value << "\n";
  int i1,i2,i3;
  double ps,th,ph,x,y,z;
  int oclust = -1;
  int lline;
  do{//Here we get the clusters
    std::cin.getline(oneline,1024);
    ctr++;
    std::istringstream ss(oneline);
    ss >> keyword >> value;
    if(keyword!="ENDCLUSTER"){
      lline = strlen(oneline);
      if(lline < 2) continue;
      if(oneline[0] == '#') continue;
      if(sscanf(oneline,"%d %d %d %lf %lf %lf %lf %lf %lf",
    &i1, &i2, &i3, &ps, &th, &ph, &x, &y, &z) != 9){
  break;
      }
      if(oclust != i1){
  oclust = i1;
  theclusters.push_back( agatacluster());
  acluster = &theclusters.back();
  acluster -> clusternumber = i1;
      }
      acluster -> crystaltype[i3] = i2;
      acluster -> crystalnumber[i3] = i3;
      acluster -> psi[i3] =  ps * TMath::DegToRad();
```

```
      acluster -> theta[i3] = th * TMath::DegToRad();
      acluster -> phi[i3] = ph * TMath::DegToRad();
      (acluster -> d[i3])[0] = x;
      (acluster -> d[i3])[1] = y;
      (acluster -> d[i3])[2] = z;
    }
  }while(keyword!="ENDCLUSTER");
  std::cout << keyword << " " << value << "\n";
  do{
    std::cin.getline(oneline,1024);
    ctr++;
    std::istringstream ss(oneline);
    ss >> keyword >> value;
  }while(keyword!="EULER");
  std::cout << keyword << " " << value << "\n";
  int nEuler=0;
  do{
    std::cin.getline(oneline,1024);
    ctr++;
    std::istringstream ss(oneline);
    ss >> keyword >> value;
    if(keyword!="ENDEULER"){
      lline = strlen(oneline);
      if(lline < 2) continue;
      if(oneline[0] == '#') continue;
      if(sscanf(oneline,"%d %d %lf %lf %lf %lf %lf %lf",
    &i1, &i2, &ps, &th, &ph, &x, &y, &z) != 8)
  break;
      //       theclusterpostions.push_back(clusterpos());
      apos = &theclusterpostions[i1];
      apos -> clusternumber = i2;
      apos -> psi = ps*TMath::DegToRad();
      apos -> theta = th*TMath::DegToRad();
      apos -> phi = ph*TMath::DegToRad();
      (apos -> d[0]) = x;
      (apos -> d[1]) = y;
      (apos -> d[2]) = z;
      nEuler++;
    }
  }while(keyword!="ENDEULER");
  std::cout << keyword << " " << value << "\n";
  do{
    std::cin.getline(oneline,1024);
    ctr++;
    std::istringstream ss(oneline);
    ss >> keyword;
  }while(keyword!="$");
  //Make detectortransformations
  MakeDetectorTransformations(theclusterpostions,theclusters,
            thecrystaltransformations);
  std::multiset<event<Interaction>,event<Interaction>> EventQue;
  event<Interaction> AnEvent;
  event<Interaction> StopEvent;
  std::map<int, std::multiset<Interaction,Interaction>> theInteractions;
  Interaction oneinteraction,testinteraction;
  Interactions InteractionsInEvent;
  unsigned long int evtnb=0,readevtnb=0;
```

```
bool new_event=false;
unsigned long int event_with_interactions=0;
std::string codewords;
float e;double t,t_=0;short detnb,segnb;
int day_=0,hour_=0,min_=0;
//Here everything is set, so I make my "event treater"
TreatOneEvent treatit(cframe,gconfframe,fFramePSA,fFrameAnc,fFrameAnc1,
    crystal_id,crystal_status,CoreE0,CoreE1,CoreT0,CoreT1,
    thecrystaltransformations,theclusters,
    theclusterpostions,listofinteractions,
    listofinteractionsperdetector,listoframesperdetector,
    acluster,apos,outputfiles,ancoutput,ancillarymap,
    ancillaryhandlers,InteractionsInEvent,addres,
    transformtocrystal,baseoutput,evtnb,
    energy_wrote_out,interactions_out);
while(!std::cin.eof()){
  std::cin.getline(oneline,1024);
  ctr++;
  std::istringstream ss(oneline);
  if(ss>>codewords){
    if(codewords=="-100"){//new event
new_event=true;
//Track down the particle id
do{
  std::cin.getline(oneline,1024);
  ctr++;
  ss.str(oneline);
  ss.clear();
  ss>>codewords;
}while(codewords!=particleid);
if(ss >> e){
  if(ss >> x){
    if(ss >> y){
      if(ss >> z){
  if(ss >> readevtnb){
    if(UseTimeForTimestamps){
      if(ss >> day_){
        if(ss >> hour_){
    if(ss >> min_){
      if(ss >> t) {;} else exit(-1);
    }
        }  else exit(-1);
      }  else exit(-1);
    } else {;}
  }  else exit(-1);
      }  else exit(-1);
    }  else exit(-1);
  } else exit(-1);
} else exit(-1);
evtnb++;
if(UseTimeForTimestamps){//Here we treat data with times...
  StopEvent.detT=t;
  StopEvent.day=day_;
  StopEvent.hour=hour_;
  StopEvent.min=min_;
  std::multiset<event<Interaction>,event<Interaction> >::iterator
    itStartEventQue=EventQue.begin();
```

```
    std::multiset<event<Interaction>,event<Interaction> >::iterator
      itStopEventQue=EventQue.upper_bound(StopEvent);
    for(;itStartEventQue!=itStopEventQue; ++itStartEventQue){
      treatit.Treat(itStartEventQue);
    }
#ifdef       _DEBUG_
    std::cout << ">--------<\n";
#endif
    EventQue.erase(EventQue.begin(),itStopEventQue);
    StopEvent.detT=0;
    StopEvent.day=100000000;
    StopEvent.hour=100000000;
    StopEvent.min=100000000;
    AnEvent.day=day_;
    AnEvent.hour=hour_;
    AnEvent.min=min_;
    AnEvent.detT=t;
    t_=t;
  } else {//Here if we only got events
    std::multiset<event<Interaction>,event<Interaction> >::iterator
      itStartEventQue=EventQue.begin();
    for(;itStartEventQue!=EventQue.end(); ++itStartEventQue){
    }
    EventQue.erase(EventQue.begin(),EventQue.end());
  }
      }
      detnb = atoi(codewords.c_str());
      if(detnb>=0){
  if(new_event && detnb<180){
    new_event=false;
    event_with_interactions++;
  }
  ss >> e >> x >> y >> z >> segnb;
  if(UseTimeForTimestamps) ss >> t; else {
    std::cout << "Only event number files not yet implemented\n";
    exit(-1);
  }
  energy_read_in[detnb]+=e;
  interactions_in[detnb]++;
  oneinteraction.Set(x,y,z,e,detnb,segnb,day_,hour_,min_,t_+t);
  //Put to the correct detector
  theInteractions[detnb].insert(oneinteraction);
  //Check if we should make an event in this detector
  testinteraction.time.Set(theInteractions[detnb].begin()->time.day,
        theInteractions[detnb].begin()->time.hour,
        theInteractions[detnb].begin()->time.min,
        theInteractions[detnb].begin()->time.detT+
        timegate*1000);
  std::multiset<Interaction,Interaction>::iterator itstart=
    theInteractions[detnb].begin();
  std::multiset<Interaction,Interaction>::iterator itstop=
    theInteractions[detnb].upper_bound(testinteraction);
  testinteraction.time.Set(day_,hour_,min_,t_);
  if(itstop!=theInteractions[detnb].end()){
    bool OldEnough=true;//We only make an event if the youngest
    //interaction is older than present event
    //Make new event
```

```
    AnEvent.detData.clear();
    AnEvent.day=theInteractions[detnb].begin()->time.day;
    AnEvent.hour=theInteractions[detnb].begin()->time.hour;
    AnEvent.min=theInteractions[detnb].begin()->time.min;
    AnEvent.detT=theInteractions[detnb].begin()->time.detT;
    for(;itstart!=itstop;++itstart){
      if(testinteraction.time-itstart->time<0){
        //So if the last interaction we want to use
        //is younfer than event dont do event
        OldEnough=false;
        break;
      }
      AnEvent.detData.push_back(*itstart);
    }
    if(OldEnough){
      EventQue.insert(AnEvent);
      //Now get rid of the entries already used
      theInteractions[detnb].erase(theInteractions[detnb].begin(),
           itstop);
    }
  }
      }
    }
    if (ctr%10000==0){
      //First find largest interaction que
      unsigned int longest=0;
      std::map<int, std::multiset<Interaction,Interaction> >::iterator
  itInteractions=theInteractions.begin();
      for(; itInteractions!=theInteractions.end(); ++itInteractions){
  if(itInteractions->second.size()>longest)
    longest=itInteractions->second.size();
      }
      std::cout << std::setw(80) << " \r";
      std::cout << "At line" << std::setw(10) << ctr
    << " and event" << std::setw(10) << evtnb
    << " max event que" << std::setw(5)
    <<  EventQue.size()
    << " max int. que" << std::setw(5)
    << longest
    << " \r";
      std::cout.flush();
    }
  }
  std::cout << "\nRead " << std::setw(10) << ctr
      << " lines and " << std::setw(10) << evtnb
      << " events and " << std::setw(10)
      <<  event_with_interactions
      << " events with at least 1 interaction.\n";
  //First we have to clear the list of interactions
  std::map<int, std::multiset<Interaction,Interaction> >::iterator
    itInteractions=theInteractions.begin();
  for(; itInteractions!=theInteractions.end(); ++itInteractions){
    std::multiset<Interaction,Interaction>::iterator itstart=
      itInteractions->second.begin();
    std::multiset<Interaction,Interaction>::iterator itstop=
      itInteractions->second.end();
    //Make new event
```

```
    AnEvent.detData.clear();
    AnEvent.day=itInteractions->second.begin()->time.day;
    AnEvent.hour=itInteractions->second.begin()->time.hour;
    AnEvent.min=itInteractions->second.begin()->time.min;
    AnEvent.detT=itInteractions->second.begin()->time.detT;
    for(;itstart!=itstop;++itstart){
      AnEvent.detData.push_back(*itstart);
    }
    EventQue.insert(AnEvent);
    //Now get rid of the entries already used
  }
  //Here we have to get rid of all events in the que that have not been
  //treated...
  std::cout << "\nEmptying event que.\n";
  while(EventQue.size()>0){
    if(EventQue.size()%1000==0){
      std::cout << std::setw(10) << EventQue.size()
    << " events left to treat.\r";
      std::cout.flush();
    }
    std::multiset<event<Interaction>,event<Interaction> >::iterator
      itStartEventQue=EventQue.begin();
    treatit.Treat(itStartEventQue);
    ++itStartEventQue;
    EventQue.erase(EventQue.begin(),itStartEventQue);
  }
  std::map<int,std::ofstream*>::iterator itfiles = outputfiles.begin();
  for(; itfiles!=outputfiles.end(); ++itfiles){
    itfiles->second->close();
    delete itfiles->second;
  }
  itfiles=ancoutput.begin();
  for(; itfiles!=ancoutput.end(); ++itfiles){
    itfiles->second->close();
    delete itfiles->second;
  }
  std::cout << "\n\n\n";
  std::ofstream intperdet("interactionsperdetector.txt");
  intperdet << std::setw(4) << "#Det" << std::setw(10) << "Nb Frames"
      << std::setw(20) << " Nb Interactions"
      << std::setw(10) << " Energy in "
      << std::setw(10) << " Energy out "
      << std::setw(10) << " inter. in "
      << std::setw(10) << " inter. out "<< "\n";
  std::map<int,int>::iterator itipd = listofinteractionsperdetector.begin();
  for(; itipd!=listofinteractionsperdetector.end(); ++itipd){
    intperdet << std::setw(4) << itipd->first << " "
        << std::setw(10) << listoframesperdetector[itipd->first]
        << std::setw(14) << itipd->second
        << std::setprecision(2) << std::setw(14) << std::fixed
        << energy_read_in[itipd->first]
            << std::setprecision(2) << std::setw(14) << std::fixed
        << energy_wrote_out[itipd->first]
        << std::setw(10) << interactions_in[itipd->first]
        << std::setw(10) << interactions_out[itipd->first]
        << "\n";
  }
```

```
  return 0;
}









#define SQ(x) ((x)*(x))
#define resolution 5.0
/* distance below which one cannot distinguish the presence of more than 1
    interaction */

void swap(double v[], int m, int l)
{
  register double temp;
  temp = v[m];
  v[m] = v[l];
  v[l] = temp;
}
void swapi(int v[], int m, int l)
{
  register int temp;
  temp = v[m];
  v[m] = v[l];
  v[l] = temp;
}

int packpoints(int number, double posx[], double posy[], double posz[],
        double energy[], int detnumb[], int segnumb[])
{
  int i, j, n, l, jp[5000], jjp[5000], ip[5000], iip[5000];
  double rpack, esum;
  l = 0;
  /* check which interactions are within precision of each other in
     the same segment */
  for (i = 0; i < number; i++) {
    for (j = i + 1; j < number; j++) {
      rpack = sqrt(SQ(posx[i] - posx[j]) + SQ(posy[i] - posy[j]) +
       SQ(posz[i] - posz[j]));
       if (rpack < resolution && segnumb[i]==segnumb[j] &&
    detnumb[i]==detnumb[j]) {
  l++;
  jp[l] = j;
  jjp[l] = j;
  ip[l] = i;
  iip[l] = i;
       }
    }
  }
  /* check if couples have already been packed by previous couples */
  for (i = 1; i <= l; i++) {
    for (j = i + 1; j <= l; j++) {
```

```
      if (ip[i] == ip[j]) { /*&& ip[i] != -1) { */
  for (n = j + 1; n <= l; n++) {
    if (ip[n] == jp[i] && jp[n] == jp[j]) {
      iip[n] = -1;
      jjp[n] = -1;
    }
    if (ip[n] == jp[j] && jp[n] == jp[i]) {
      iip[n] = -1;
      jjp[n] = -1;
    }
  }
      }
    }
  }

  for (n = 1; n <= l; n++) {
    if (iip[n] == -1)
      ip[n] = -1;
    if (jjp[n] == -1)
      jp[n] = -1;
  }

  for (j = 1; j <= l; j++) {
    for (i = 0; i < number; i++) {
      if (ip[j] == i && jp[j] != i) {
  esum = energy[i] + energy[jp[j]];
  /* new position pondered by energie */
  posx[i] = ((posx[i] * energy[i]) +
       (posx[jp[j]] * energy[jp[j]])) / esum;
  posy[i] = ((posy[i] * energy[i]) +
       (posy[jp[j]] * energy[jp[j]])) / esum;
  posz[i] = ((posz[i] * energy[i]) +
       (posz[jp[j]] * energy[jp[j]])) / esum;
  energy[i] = esum; /* put 2 energies into 1 */
  swap(energy, number - 1, jp[j]);
  /* put unused energy at the end of the list */
  swap(posx, number - 1, jp[j]);
  swap(posy, number - 1, jp[j]);
  swap(posz, number - 1, jp[j]);
  swapi(detnumb, number - 1, jp[j]);
  swapi(segnumb, number - 1, jp[j]);
  for (n = j + 1; n <= l; n++) {
    if (ip[n] == jp[j]) { /* if the one just packed needs to be packed */
      ip[n] = i;
    }
    if (jp[n] == jp[j]) { /* if the one just packed needs to be packed */
      jp[n] = i;
    }
    if (ip[n] == number - 1) {  /* if end of list needs to be packed */
      ip[n] = jp[j];
    }
    if (jp[n] == number - 1) {  /* if end of list needs to be packed */
      jp[n] = jp[j];
    }
  }
  number -= 1;  /* decrement the number of interactions */
      }
```

```
    }
  }
  return number;     /* return new number of interactions */
}



int packpointsbarycenter(int number, double posx[], double posy[],
       double posz[], double energy[], int detnumb[],
       int segnumb[])
{
  int i,j;
  double en[800];
  for(i=0;i<number;i++){
    for(j=i+1;j<number;j++){
      if(energy[i]!=0 && detnumb[i]==detnumb[j] && segnumb[i]==segnumb[j]){
  en[i]=energy[i]+energy[j];
  posx[i]=((energy[i]*posx[i])+(energy[j]*posx[j]))/en[i];
  posy[i]=((energy[i]*posy[i])+(energy[j]*posy[j]))/en[i];
  posz[i]=((energy[i]*posz[i])+(energy[j]*posz[j]))/en[i];
  energy[j]=0;
  energy[i]=en[i];
      }
    }
  }
  j=0;
  for(i=0;i<number;i++){
    if(energy[i]!=0){
      energy[j]=energy[i];
      posx[j]=posx[i];
      posy[j]=posy[i];
      posz[j]=posz[i];
      detnumb[j]=detnumb[i];
      segnumb[j]=segnumb[i];
      j++;
    }
  }
  number = j;
  return number;     /* return new number of interactions */
}



#ifndef __CINT__
int main() {
  std::string baseoutput="PSA_";
  bool transformtocrystal=true;
  bool addres=true;
  Double_t timegate = 10;
  std::string particleid = "-1";
  if(getenv("PARTICLEID")) particleid=std::string(getenv("PARTICLEID"));
  MakeADFEvents(baseoutput,transformtocrystal,addres,timegate,particleid);
  return 0;
}
#endif
```

B AncHandler example

/Users/joa/AGATA/agataganil/testsuite/absolute_time/KOELN.cxx

```
/*
  This codes creates data:ranc1 frames with Legnaro data format, i.e
  one UInt_t telling how many Float_t that follows

  compile with

  g++ -shared -o libKOELN.so KOELN.cxx
*/

#include <iostream>
#include <fstream>
#include <sstream>
#include <map>
#include <string>

extern "C"  {
  void KOELN(std::ostringstream &buff);
}


void KOELN(std::ostringstream &buff)

{
  static std::map<int,float> LUT_theta;
  static std::map<int,float> LUT_phi;
  if(LUT_theta.size()==0){
    std::ifstream input("NUMBERING.S2");
    int ring,sec;
    float theta,phi;
    while(input.good()){
      input >> sec >> ring >> theta >> phi;
      LUT_theta[ring]=theta;
      LUT_phi[sec]=phi;
    }
  }
  //I will sum up everything in the same segments...
  //Get rid of x,y,z
  std::map<int,float> E,T,theta,phi;
  std::istringstream abuffer(buff.str());
  std::istringstream aline;
  buff.str("");
  buff.clear();
  std::string oneline;
  double e,x,y,z,ns;
  int seg,det,day,hour,min;
  //First make some space in buffer
  unsigned int numberoffloats=0;
  buff.write((char*)&numberoffloats,sizeof(unsigned int));
  while(abuffer.good()){
    std::getline(abuffer,oneline);
    if(abuffer.good()){
      aline.clear();
      aline.str(oneline);
```

```
      aline >> x >> y >> z >> e >> det >> seg >> day >> hour >> min >> ns;
      E[seg]+=e;
      T[seg]+=ns*e;
      theta[seg]=LUT_theta.find(seg)!=LUT_theta.end() ? LUT_theta[seg] : -1.;
      phi[seg]=LUT_phi.find(det-1000)!=LUT_phi.end() ? LUT_phi[det-1000] : -1.;
    }
  }
  std::map<int,float>::iterator itE=E.begin();
  //Fill with data floats
  for(; itE!=E.end(); ++itE){
    if(itE->second>5){
      T[itE->first]/=(itE->second>0 ? itE->second : 1);
      buff.write((char*)&(itE->second),sizeof(float));numberoffloats++;
      buff.write((char*)&(T[itE->first]),sizeof(float));numberoffloats++;
      buff.write((char*)&(theta[itE->first]),sizeof(float));numberoffloats++;
      buff.write((char*)&(phi[itE->first]),sizeof(float));numberoffloats++;
    }
  }
  //Tell how many floats that were written
  buff.seekp(0);
  buff.write((char*)&numberoffloats,sizeof(unsigned int));
}
```