Modified GEANT4 processes for JUNO Simulation

60th Geant4 Technical Forum

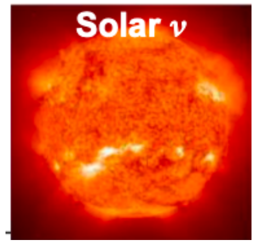
Simon Blyth (IHEP), Guofu Cao (IHEP), Ziyan Deng (IHEP), Cécile Jollet (Bordeaux university, LP2iB - CNRS/IN2P3), Lin Tao (IHEP)

JUNO physics program

- JUNO is a multipurpose Neutrino Observatory and it has a rich program in neutrino physics and astrophysics studying neutrinos in a large energy range.

10^4 evts at 10 kpc
DSNB : 2-4 evts/year

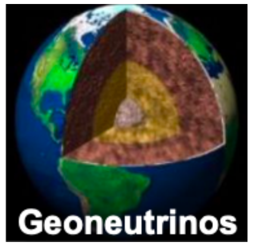
>~100 evts/day



^{8}B : 16 evts/day
^{7}Be : 490 evts/day/kton

Proton decays : $p \to \bar{\nu} + K^+$
Indirect Dark Matter Searches

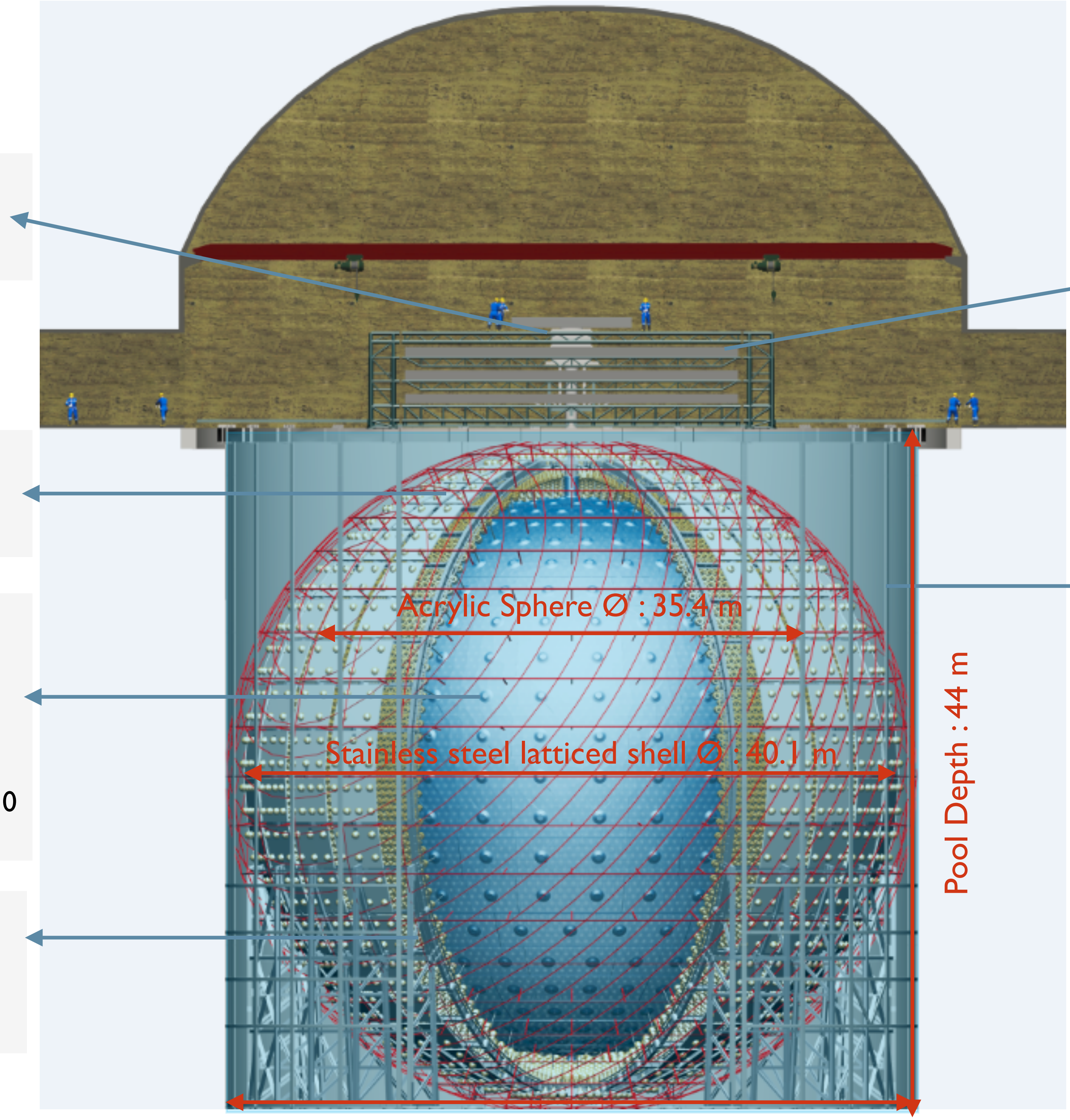
45 evts/day



400 evts/year

→Neutrino mass ordering
→Precision measurement of solar oscillation parameters

JUNO detector



High energy precision

Calibration room

multi-dimension calibration systems

Central Detector (CD)

SS latticed shell
Acrylic sphere

Liquid Scintillator (LS)

20 kilo-ton based LAB LS
→High light yield : ~ 10 000 photons/MeV
→ High transparency : ~ 20 meters attenuation length at 430 nm

Photomultipliers (PMTs)

17 612 20" PMTs
25 600 3" PMTs
→~ 78% coverage

Backgrounds reduction

Top Tracker

3 layers of plastic scintillator (cover ~60% of Water Pool)
→Precise muon tracker

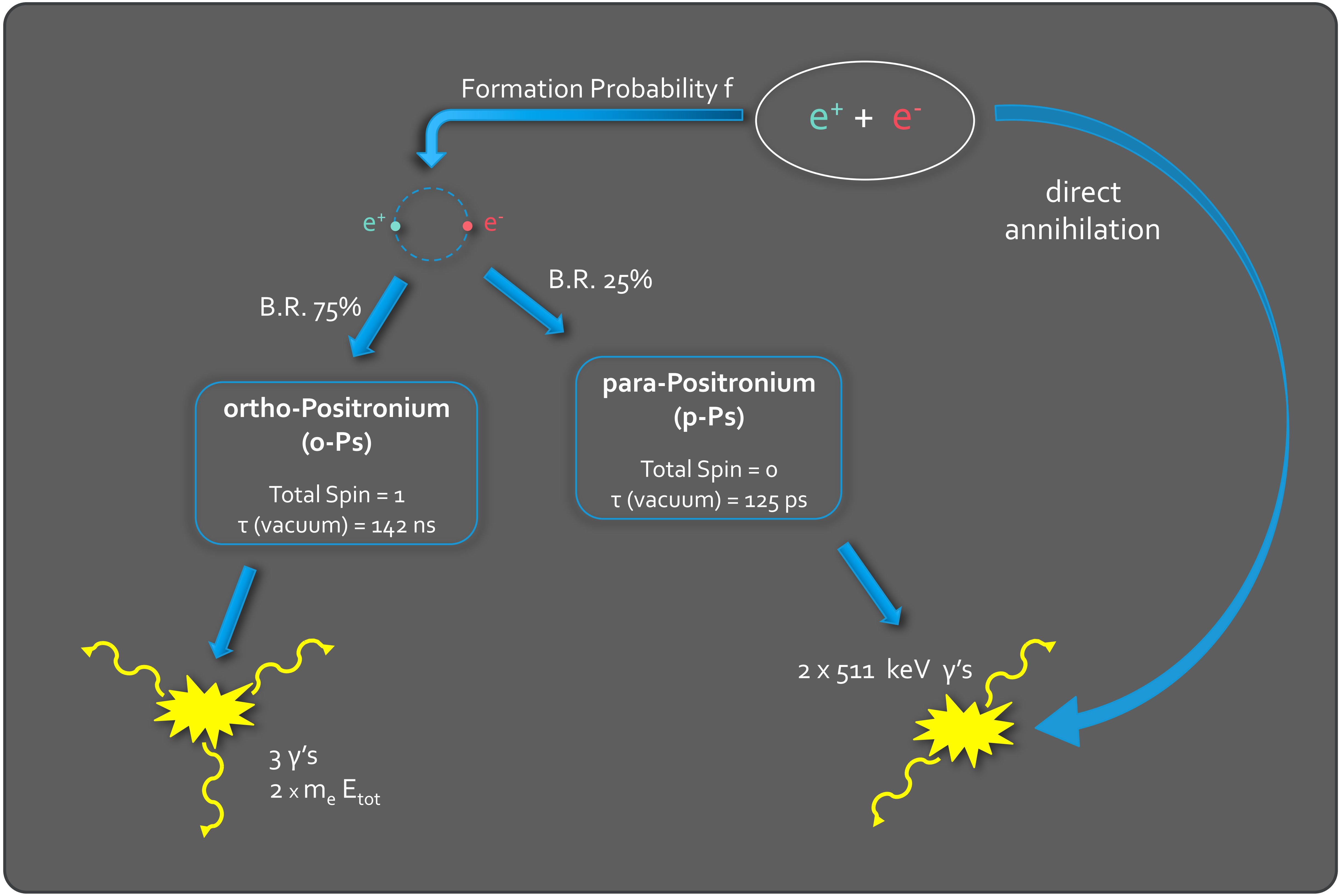
Water Pool (WP)

35 kilo-ton pure water
2400 20" PMTs on CD surface
→High muon detection efficiency
→Protects CD against external radioactivity

JUNO Simulation

- JUNO Simulation software is based on the Geant4 toolkit and the SNiPER (Software for Non-collider Physics Experiment) framework (*Eur.Phys.J.C* 83 (2023) 5, 382, *Eur.Phys.J.C* 83 (2023) 7, 660 (erratum)).
- Several processes have been changed in GEANT4 in order to better fit with the requirements of the experiment:
    - Implementation of positronium decay in G4EmLivermorePhysics.
    - Modifications of gamma generation in neutron capture processes.
    - Modification of refraction index in G4Cerenkov.
    - Modifications of G4OpBoundaryProcess to take into account of the PMT optical model.
    - Radioactive decay of cosmogenic nuclei.

Positronium formation (I)



Formation Probability f

$e^+ + e^-$

direct annihilation

e^+ e^-

B.R. 75%

B.R. 25%

ortho-Positronium (o-Ps)

Total Spin = 1

τ (vacuum) = 142 ns

para-Positronium (p-Ps)

Total Spin = 0

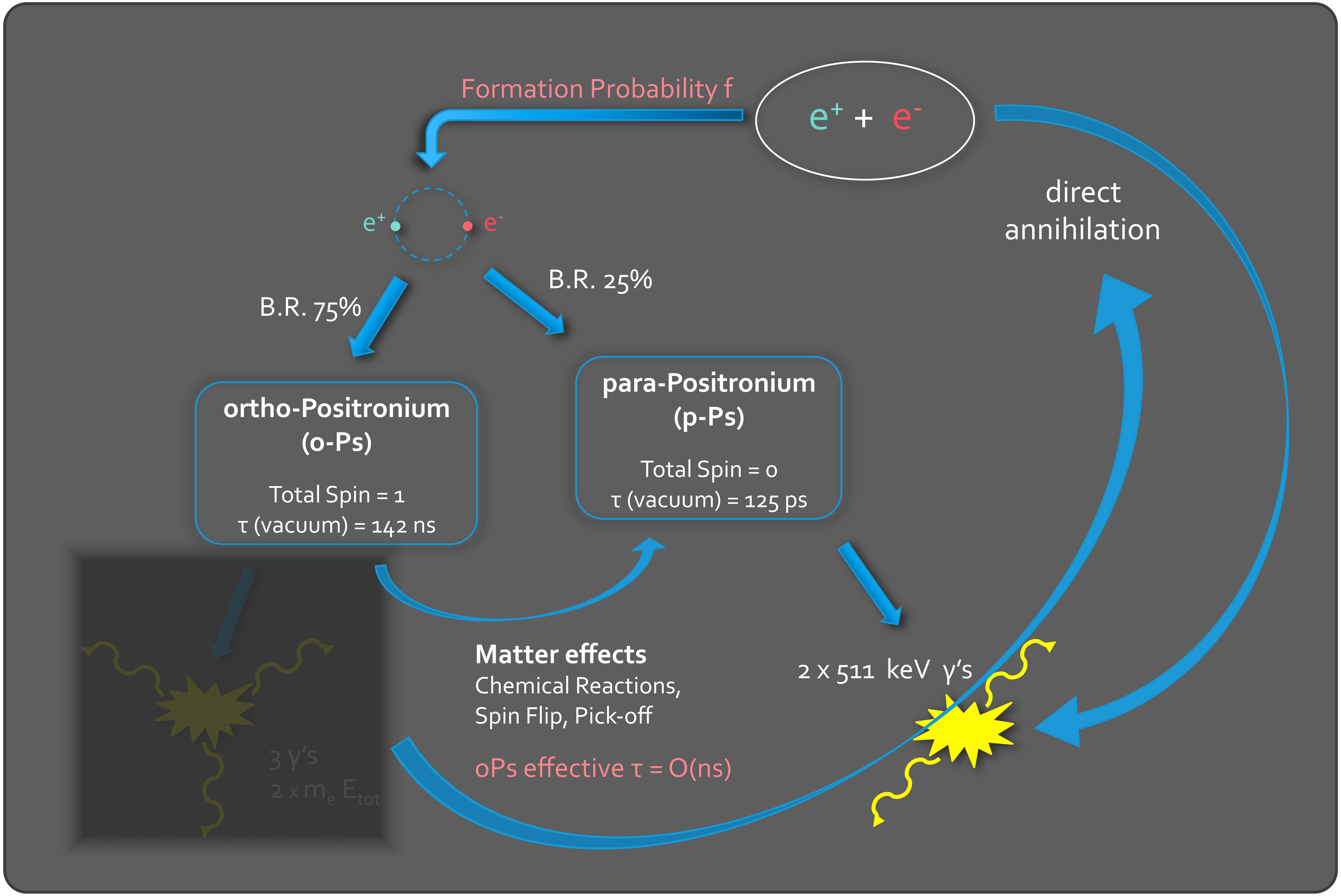
τ (vacuum) = 125 ps

3 γ's

2 x m_e E_{tot}

2 x 511 keV γ's

Positronium formation (2)



Formation Probability f

$e^+ + e^-$

direct annihilation

e^+ e^-

B.R. 75%

B.R. 25%

ortho-Positronium (o-Ps)

Total Spin = 1
τ (vacuum) = 142 ns

para-Positronium (p-Ps)

Total Spin = 0
τ (vacuum) = 125 ps

3 γ's
2 x m_e E_{tot}

Matter effects
Chemical Reactions,
Spin Flip, Pick-off

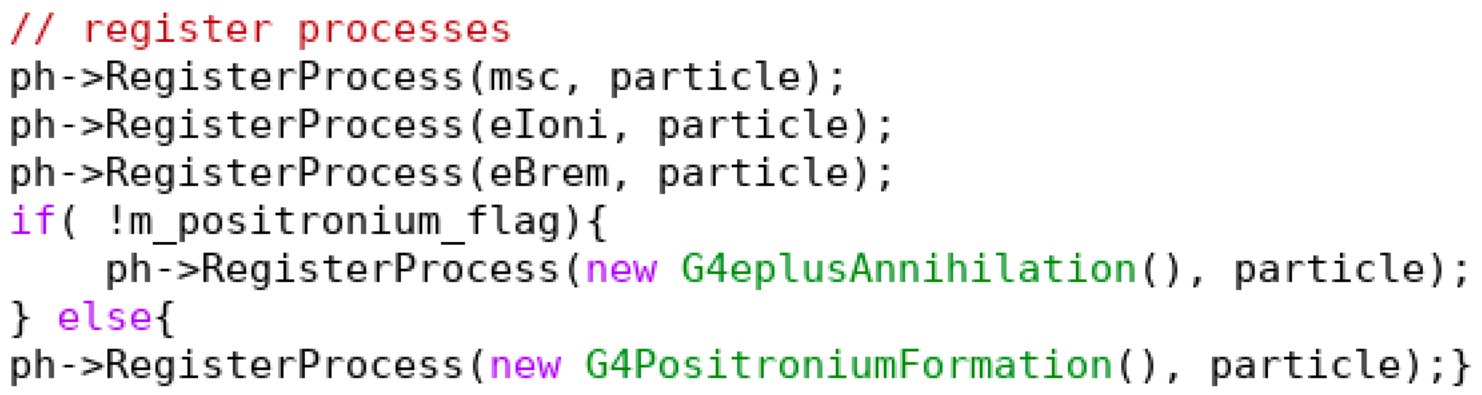
oPs effective τ = O(ns)

2 x 511 keV γ's

Positronium generator (1)

From Paolo Crivelli (ETH Zurich): paolo.crivelli@cern.ch, Cécile Jollet (LP2i Bordeaux): cecile.jollet@cern.ch, Anselmo Meregaglia (LP2i Bordeaux): anselmo.meregaglia@cern.ch

- The positronium generator is made with the addition of 6 files (*.cc and *.hh): G4PositroniumFormation, G4Positronium, G4PositroniumDecayChannel2G, G4PositroniumDecayChannel3G.
- In G4EmLivermorePhysics.cc, we replace the « G4eplusAnnihilation » process by the « G4PositroniumFormation » process.

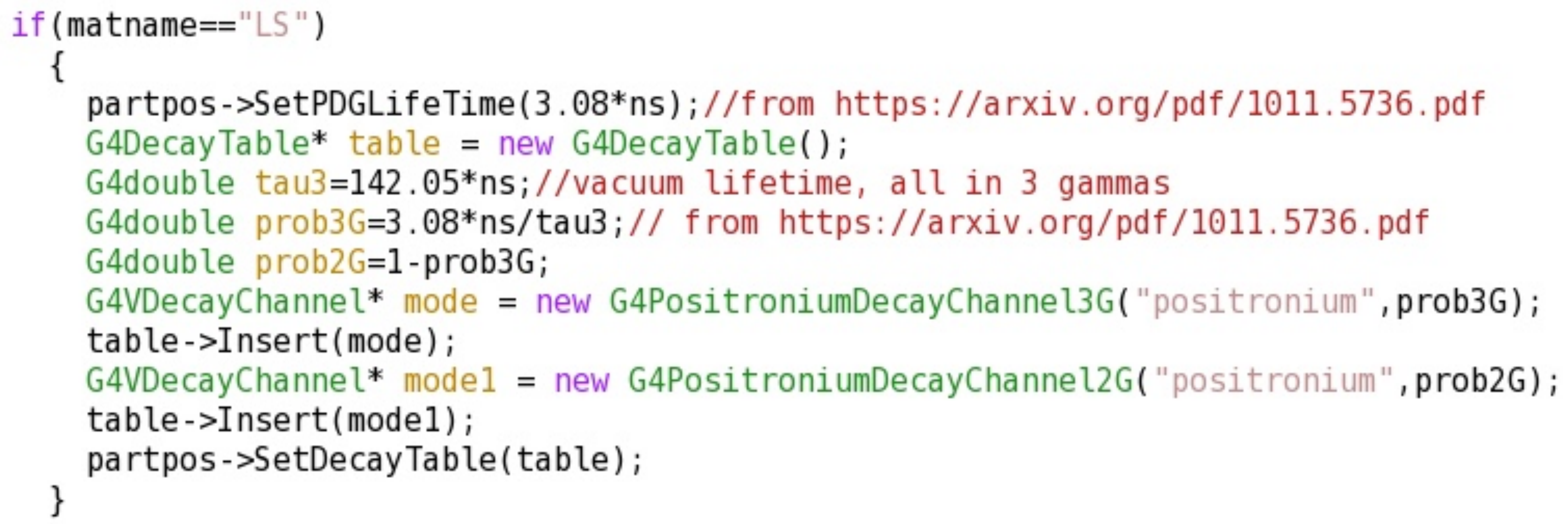


```
// register processes
ph->RegisterProcess(msc, particle);
ph->RegisterProcess(eIoni, particle);
ph->RegisterProcess(eBrem, particle);
if( !m_positronium_flag){
    ph->RegisterProcess(new G4eplusAnnihilation(), particle);
} else{
ph->RegisterProcess(new G4PositroniumFormation(), particle);}
```

- In G4PositroniumFormation, we set for the volume considered the positronium formation probability and its lifetime. In the case of JUNO, the material is Liquid Scintillator and the values are hard-coded (formation probability of 0.545 and lifetime of 3.08 ns from *Phys.Rev.C 88 (2013) 065502*)

Positronium generator (2)

- A random number is shooted, if it is lower than this oPs formation probability, the positron undergoes annihilation, otherwise positronium is formed and the lifetimes are set:



```
if(matname=="LS")
  {
    partpos->SetPDGLifeTime(3.08*ns);//from https://arxiv.org/pdf/1011.5736.pdf
    G4DecayTable* table = new G4DecayTable();
    G4double tau3=142.05*ns;//vacuum lifetime, all in 3 gammas
    G4double prob3G=3.08*ns/tau3;// from https://arxiv.org/pdf/1011.5736.pdf
    G4double prob2G=1-prob3G;
    G4VDecayChannel* mode = new G4PositroniumDecayChannel3G("positronium",prob3G);
    table->Insert(mode);
    G4VDecayChannel* mode1 = new G4PositroniumDecayChannel2G("positronium",prob2G);
    table->Insert(mode1);
    partpos->SetDecayTable(table);
  }
```

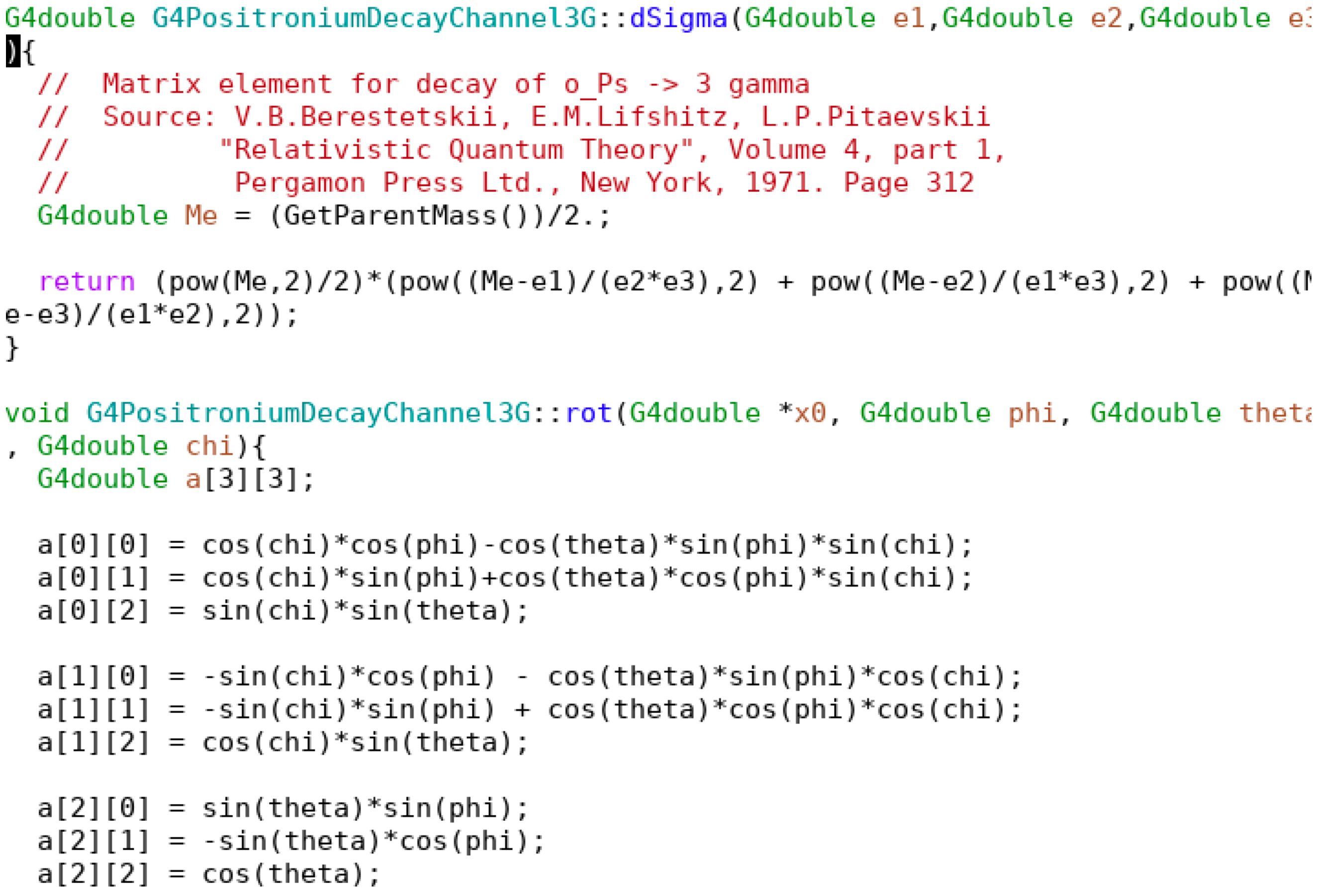
- According to the probability, the annihilation into 3 gammas (2% probability) or 2 gammas is taken into account.

Positronium generator (3)

G4PositroniumDecayChannel2G.cc

```
  //Generating Energy
 G4double cosTeta = 2.*G4UniformRand()-1.  , sinTeta = sqrt(1.-cosTeta*cosTeta);
 G4double phi     = twopi * G4UniformRand();
 G4ThreeVector direction (sinTeta*cos(phi), sinTeta*sin(phi), cosTeta);

 //gamma 1
 G4DynamicParticle * daughterparticle = new G4DynamicParticle(G4MT_daughters[0]
,direction, parentmass/2);
 products->PushProducts(daughterparticle);

 // gamma 2

 G4DynamicParticle * daughterparticle1 = new G4DynamicParticle(G4MT_daughters[1
],-direction, parentmass/2);
 products->PushProducts(daughterparticle1);

 G4cout<<" decay 2 gammas "<<parentmass/2<<G4endl;
```

G4PositroniumDecayChannel3G.cc

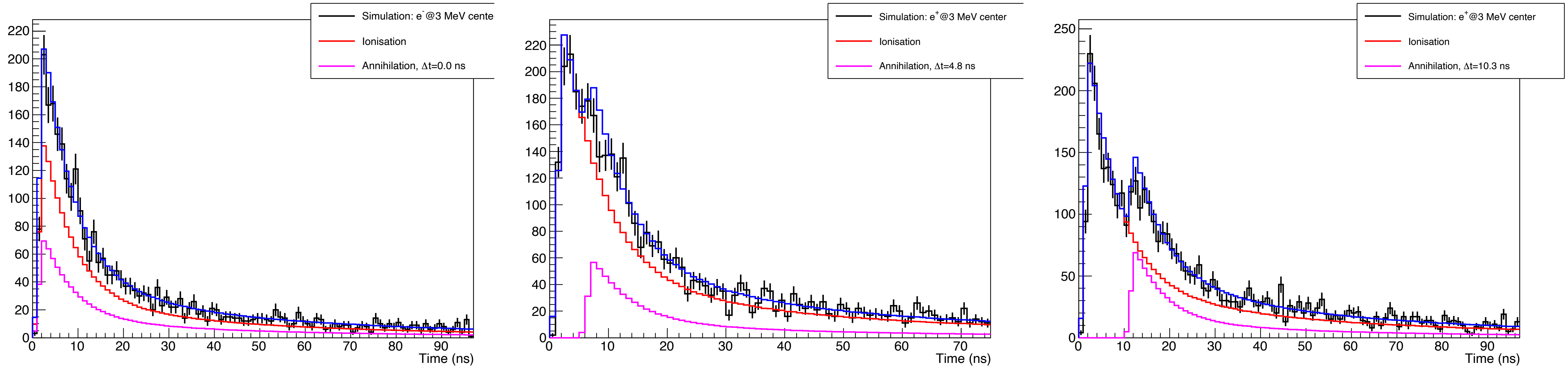


```
G4double G4PositroniumDecayChannel3G::dSigma(G4double e1,G4double e2,G4double e
){
  //  Matrix element for decay of o_Ps -> 3 gamma
  //  Source: V.B.Berestetskii, E.M.Lifshitz, L.P.Pitaevskii
  //          "Relativistic Quantum Theory", Volume 4, part 1,
  //           Pergamon Press Ltd., New York, 1971. Page 312
  G4double Me = (GetParentMass())/2.;

  return (pow(Me,2)/2)*(pow((Me-e1)/(e2*e3),2) + pow((Me-e2)/(e1*e3),2) + pow((
e-e3)/(e1*e2),2));
}

void G4PositroniumDecayChannel3G::rot(G4double *x0, G4double phi, G4double thet
, G4double chi){
  G4double a[3][3];

  a[0][0] = cos(chi)*cos(phi)-cos(theta)*sin(phi)*sin(chi);
  a[0][1] = cos(chi)*sin(phi)+cos(theta)*cos(phi)*sin(chi);
  a[0][2] = sin(chi)*sin(theta);

  a[1][0] = -sin(chi)*cos(phi)  - cos(theta)*sin(phi)*cos(chi);
  a[1][1] = -sin(chi)*sin(phi)  + cos(theta)*cos(phi)*cos(chi);
  a[1][2] = cos(chi)*sin(theta);

  a[2][0] = sin(theta)*sin(phi);
  a[2][1] = -sin(theta)*cos(phi);
  a[2][2] = cos(theta);
```

Positronium generator : results

- The time emission is consequently different for a positron (which forms positronium) and an electron, and this could be very useful for particle identification and e+/e- discrimination.

Time distribution of the p.e. on the PMT



e+ without positronium

e+ with positronium and annihilation delayed by ~4.8 ns

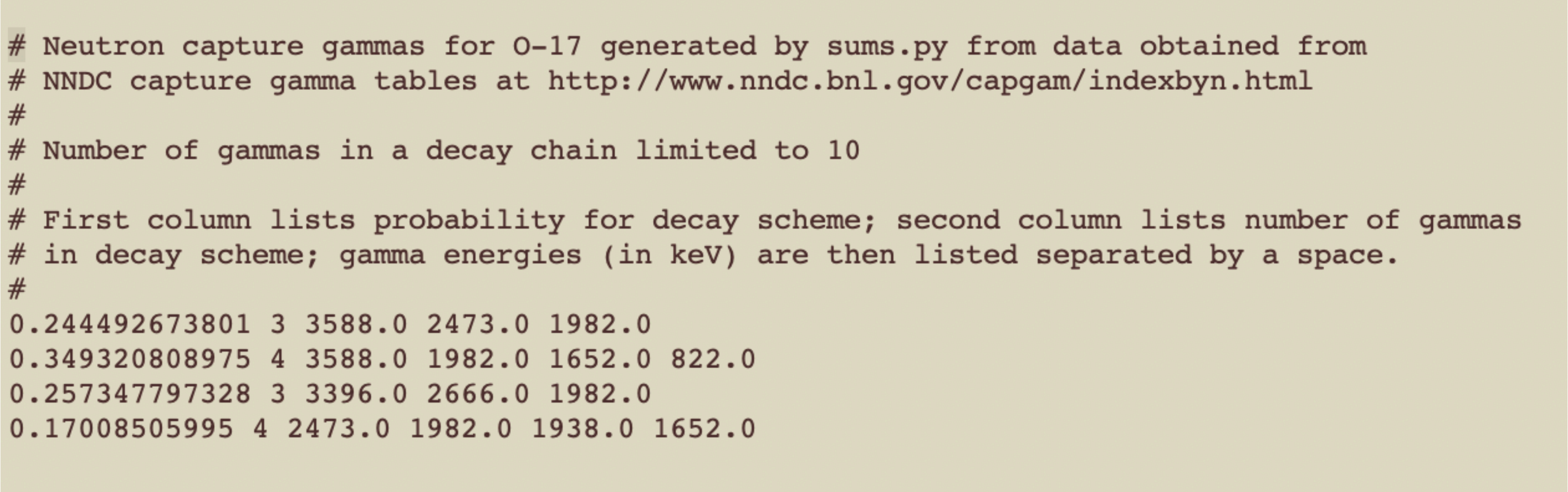
e+ with positronium and annihilation delayed by ~10.5 ns

- In data (Double Chooz experiment *JHEP* 10 (2014) 032) we observed such events and we considered that it is important to simulate them.

Neutron capture

From Guofu Cao (IHEP): caogf@ihep.ac.cn

- The multiplicity and energy of gammas emitted after neutron capture is important for reactor anti-neutrinos experiment.
- Using Geant4.9 , we found some issues in the gamma emission for neutron capture on these different nuclei : Gd, Fe, Ni, Si, P, Mn, S, Cr, O, N and C.
- For all these nuclei we generated *.txt files with the gamma lines from NNDC. In total, 32 files have been created.



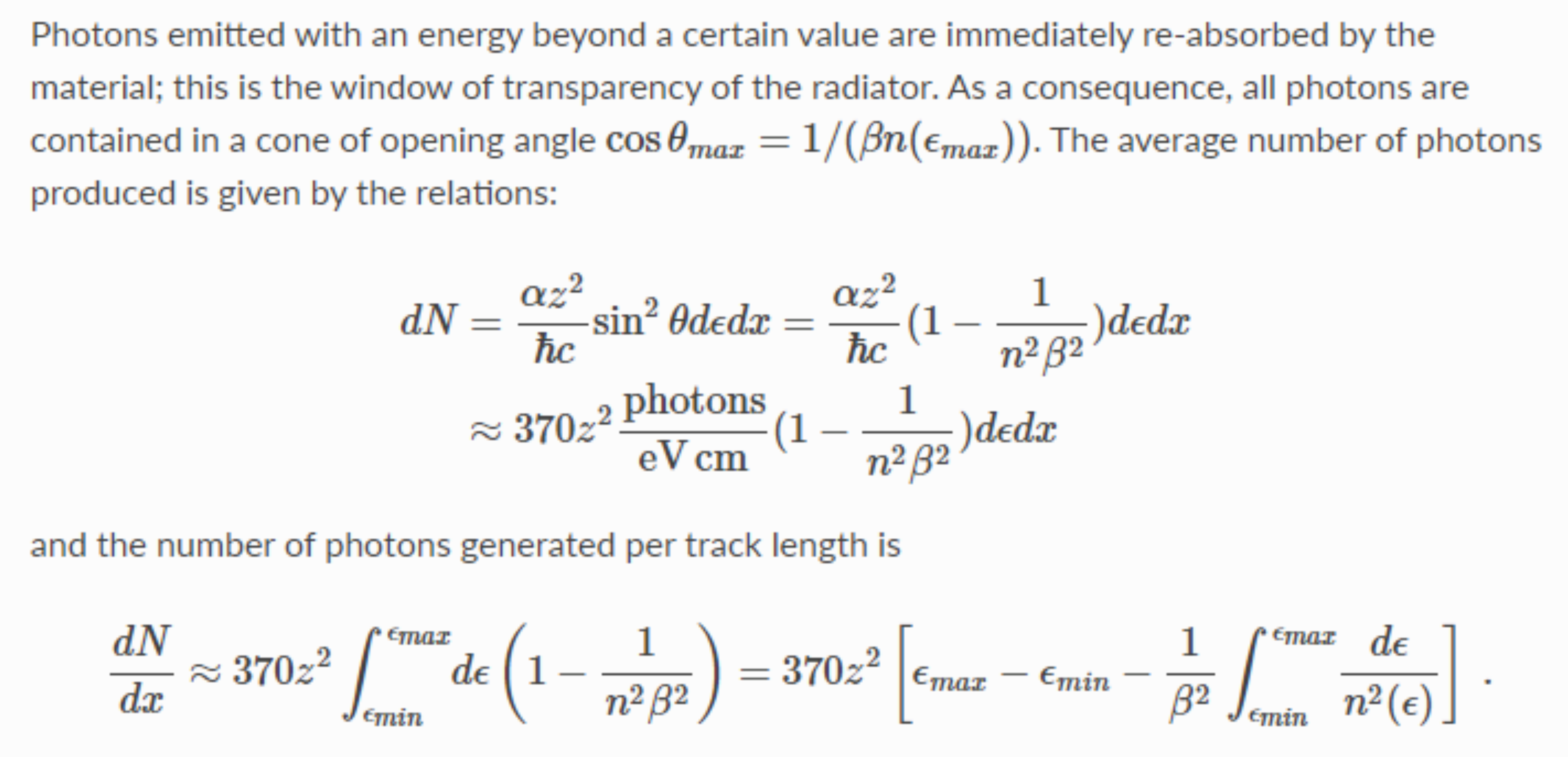
```
# Neutron capture gammas for O-17 generated by sums.py from data obtained from
# NNDC capture gamma tables at http://www.nndc.bnl.gov/capgam/indexbyn.html
#
# Number of gammas in a decay chain limited to 10
#
# First column lists probability for decay scheme; second column lists number of gammas
# in decay scheme; gamma energies (in keV) are then listed separated by a space.
#
0.244492673801 3 3588.0 2473.0 1982.0
0.349320808975 4 3588.0 1982.0 1652.0 822.0
0.257347797328 3 3396.0 2666.0 1982.0
0.17008505995 4 2473.0 1982.0 1938.0 1652.0
```

- We created DsG4NNDCCaptureGammas.cc which read these files and generate the gammas.
- We did not check if modifications have been made with more recent Geant4 versions.

Cerenkov process modifications (1)

From Lin Tao (IHEP): lintao@ihep.ac.cn

https://geant4-userdoc.web.cern.ch/UsersGuides/PhysicsReferenceManual/html/electromagnetic/xray_production/cerenkov.html



Photons emitted with an energy beyond a certain value are immediately re-absorbed by the material; this is the window of transparency of the radiator. As a consequence, all photons are contained in a cone of opening angle $\cos\theta_{max} = 1/(\beta n(\epsilon_{max}))$. The average number of photons produced is given by the relations:

$$dN = \frac{\alpha z^2}{\hbar c}\sin^2\theta d\epsilon dx = \frac{\alpha z^2}{\hbar c}(1-\frac{1}{n^2\beta^2})d\epsilon dx$$

$$\approx 370 z^2 \frac{\text{photons}}{\text{eV cm}}(1-\frac{1}{n^2\beta^2})d\epsilon dx$$

and the number of photons generated per track length is

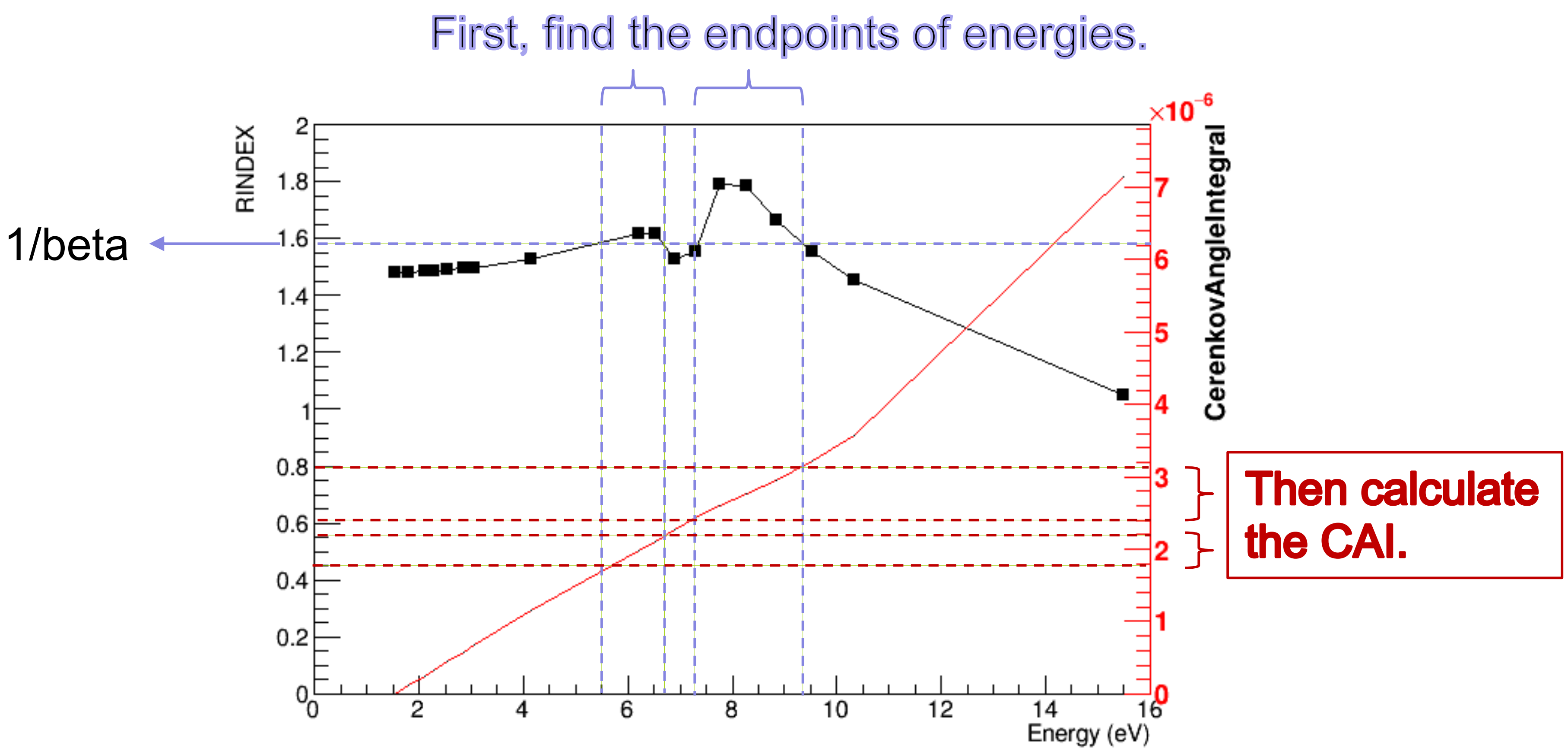
$$\frac{dN}{dx} \approx 370 z^2 \int_{\epsilon_{min}}^{\epsilon_{max}} d\epsilon \left(1-\frac{1}{n^2\beta^2}\right) = 370 z^2 \left[\epsilon_{max}-\epsilon_{min}-\frac{1}{\beta^2}\int_{\epsilon_{min}}^{\epsilon_{max}}\frac{d\epsilon}{n^2(\epsilon)}\right].$$

where $n(E)>1/\beta$

In GEANT4, n(E) is assumed as an increasing function of energy.

Cerenkov process modifications (2)

- In JUNO, the RINDEX of the Liquid Scintillator is not a monotonic function.
- Consequently, we modified G4Cerenkov.cc in order to calculate the Cherenkov Angle Integral (CAI) according to the energy ranges where RINDEX>1/β.


First, find the endpoints of energies.

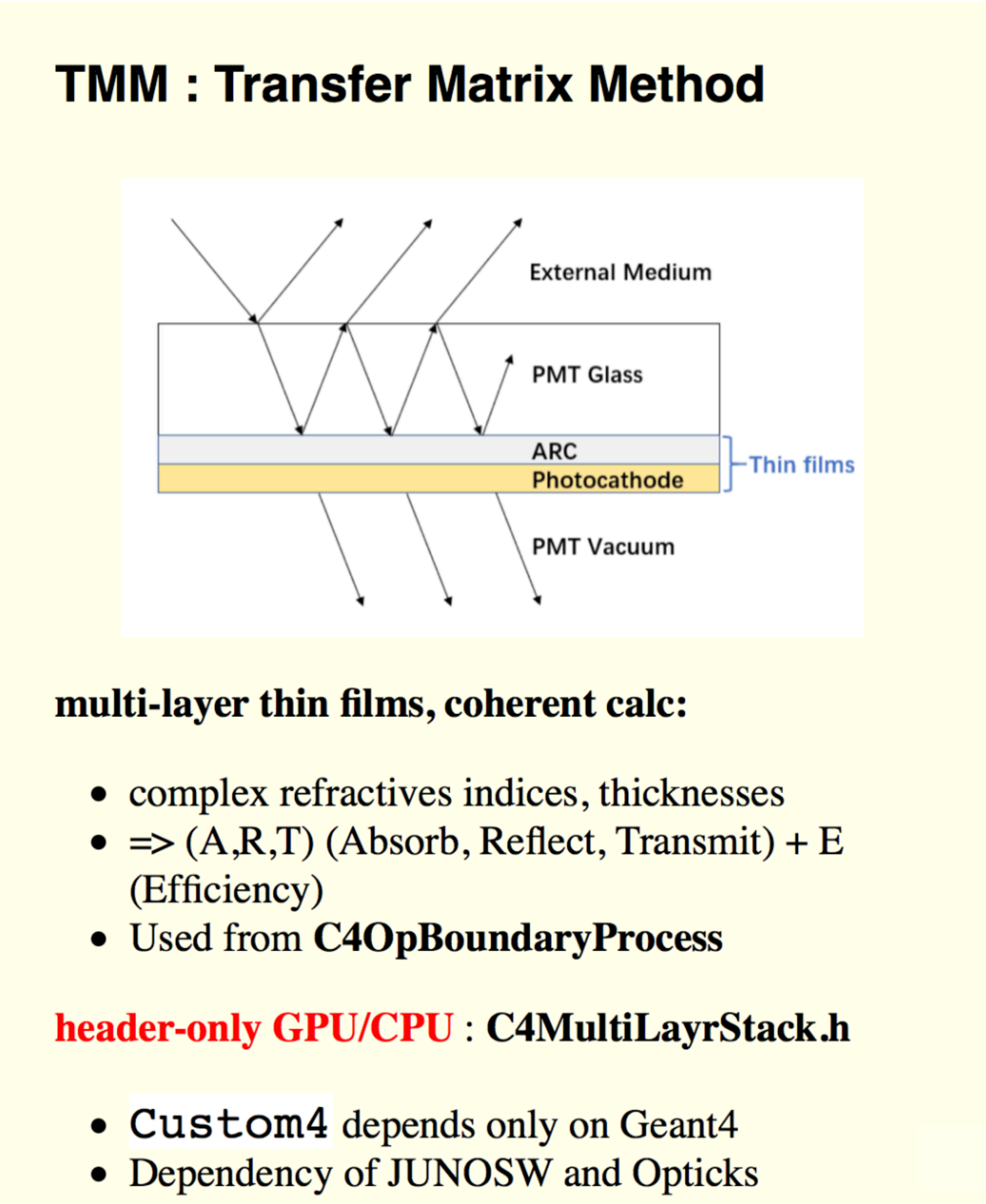
1/beta

Then calculate the CAI.

G4OpBoundaryProcess modifications

From Simon Blyth (IHEP): simon.c.blyth@gmail.com

- The JUNO PMT simulation uses a developed PMT optical model (POM) that allows photons to enter the PMT and bounce around inside and sometimes refract out again to possibly continue on to other PMTs.
- Also the POM accounts for thin film interference effects on the stack of layers: Pyrex, Anti reflection coating (ARC), Photocathode (PHC), Vacuum.
- The calculation yields A,R,T (absorption, reflection, transmission) probabilities uses as input the thicknesses of the ARC and PHC layers and refractive indices for the 4 layers. The two inner layers can have a complex refractive index, allowing absorption.
- The transverse matrix method (TMM) is used to sum up the contribution of interference using complex refractive indices for ARC and PHC.



TMM : Transfer Matrix Method

multi-layer thin films, coherent calc:

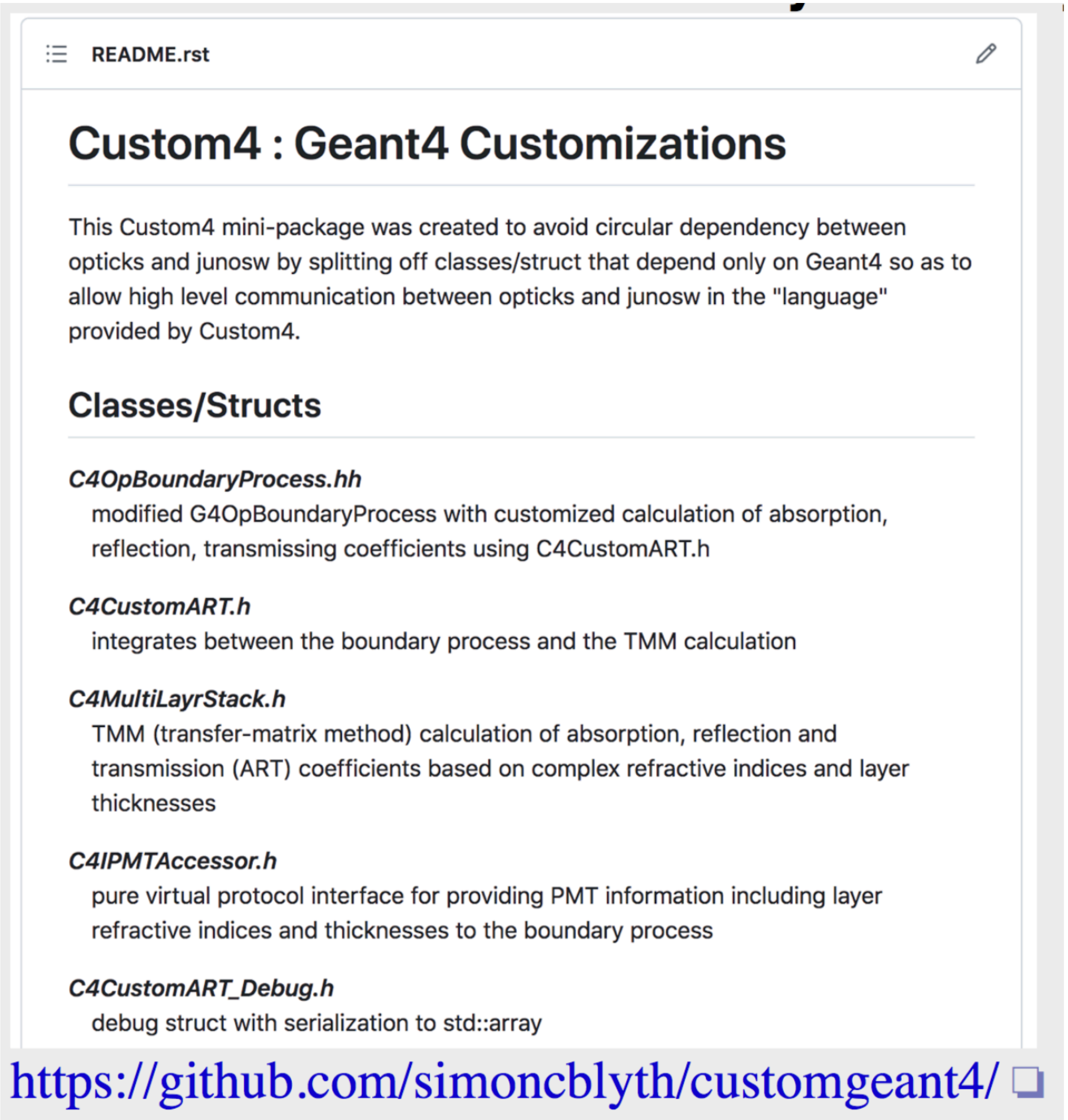
- complex refractives indices, thicknesses
- => (A,R,T) (Absorb, Reflect, Transmit) + E (Efficiency)
- Used from **C4OpBoundaryProcess**

header-only GPU/CPU : **C4MultiLayrStack.h**

- `Custom4` depends only on Geant4
- Dependency of JUNOSW and Opticks

G4OpBoundaryProcess modifications

- Thus with A,R,T, E the standard boundary is modified in a way that reuses as much of standard Geant4 as possible. The customisation allows the minimum change to be "injected" to implement the POM.



Custom4 : Geant4 Customizations

This Custom4 mini-package was created to avoid circular dependency between opticks and junosw by splitting off classes/struct that depend only on Geant4 so as to allow high level communication between opticks and junosw in the "language" provided by Custom4.

Classes/Structs

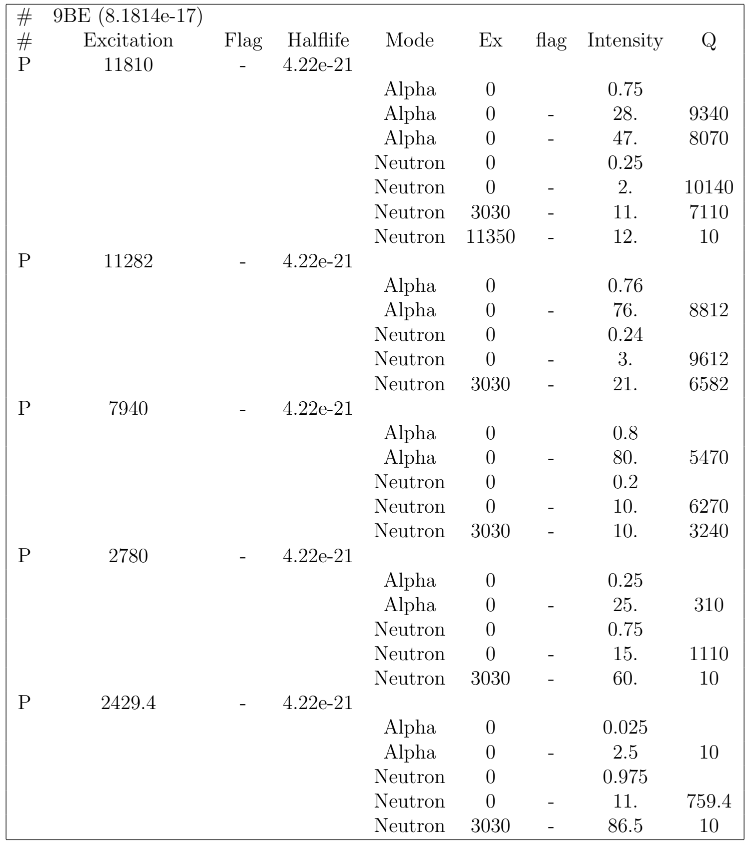
C4OpBoundaryProcess.hh	modified G4OpBoundaryProcess with customized calculation of absorption, reflection, transmissing coefficients using C4CustomART.h
C4CustomART.h	integrates between the boundary process and the TMM calculation
C4MultiLayrStack.h	TMM (transfer-matrix method) calculation of absorption, reflection and transmission (ART) coefficients based on complex refractive indices and layer thicknesses
C4IPMTAccessor.h	pure virtual protocol interface for providing PMT information including layer refractive indices and thicknesses to the boundary process
C4CustomART_Debug.h	debug struct with serialization to std::array

https://github.com/simoncblyth/customgeant4/ ❏

Nuclei radioactive decay

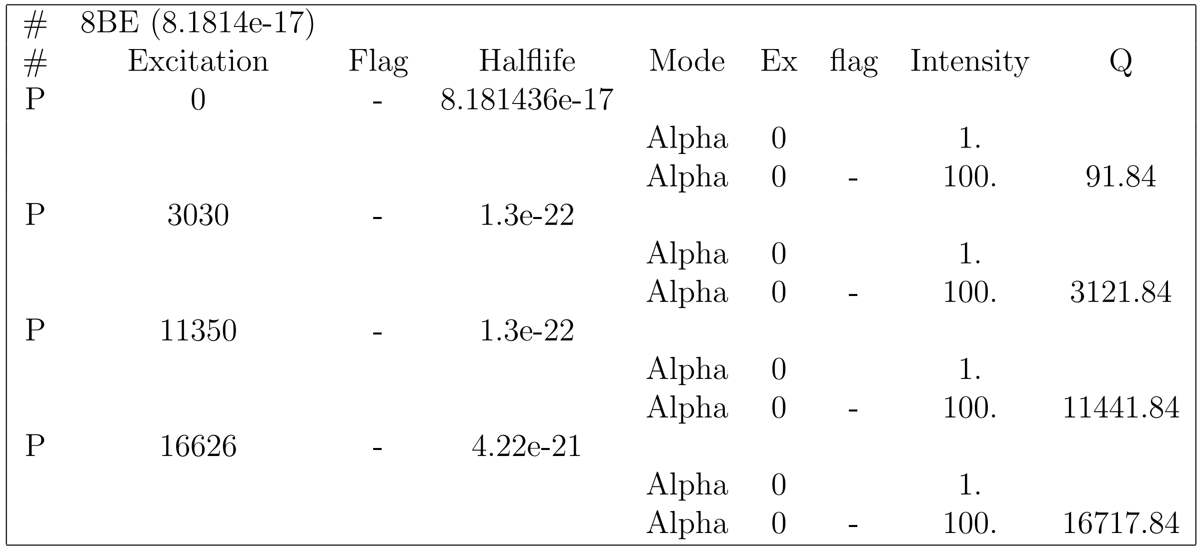
- One important background of reactor anti-neutrinos experiment is the decay of two cosmogenic isotopes: ^{9}Li and ^{8}He.
- Their spectra have been measured by nuclear spectroscopy and their decay is accompanied by neutrons and alpha emission. In previous Geant4 versions the excited states decayed by gamma emission.
- We modified the RadioactiveDecay files and we added the decay with the emission of triton (NIM A 949, 162904 (2020)).
- These modifications have been implemented officially in the Geant4.

z4.a9



#	9BE (8.1814e-17)							
#	Excitation	Flag	Halflife	Mode	Ex	flag	Intensity	Q
P	11810	-	4.22e-21					
				Alpha	0		0.75	
				Alpha	0	-	28.	9340
				Alpha	0	-	47.	8070
				Neutron	0		0.25	
				Neutron	0	-	2.	10140
				Neutron	3030	-	11.	7110
				Neutron	11350	-	12.	10
P	11282	-	4.22e-21					
				Alpha	0		0.76	
				Alpha	0	-	76.	8812
				Neutron	0		0.24	
				Neutron	0	-	3.	9612
				Neutron	3030	-	21.	6582
P	7940	-	4.22e-21					
				Alpha	0		0.8	
				Alpha	0	-	80.	5470
				Neutron	0		0.2	
				Neutron	0	-	10.	6270
				Neutron	3030	-	10.	3240
P	2780	-	4.22e-21					
				Alpha	0		0.25	
				Alpha	0	-	25.	310
				Neutron	0		0.75	
				Neutron	0	-	15.	1110
				Neutron	3030	-	60.	10
P	2429.4	-	4.22e-21					
				Alpha	0		0.025	
				Alpha	0	-	2.5	10
				Neutron	0		0.975	
				Neutron	0	-	11.	759.4
				Neutron	3030	-	86.5	10

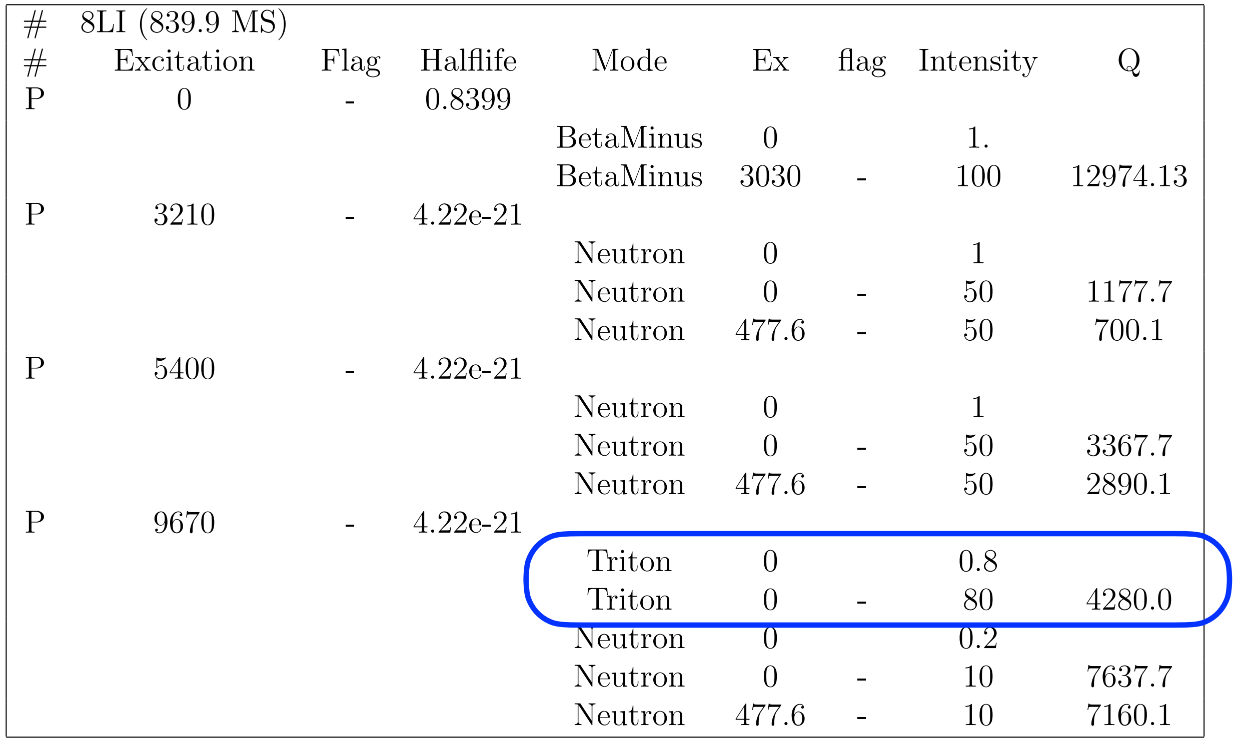
z4.a8



#	8BE (8.1814e-17)							
#	Excitation	Flag	Halflife	Mode	Ex	flag	Intensity	Q
P	0	-	8.181436e-17					
				Alpha	0		1.	
				Alpha	0	-	100.	91.84
P	3030	-	1.3e-22					
				Alpha	0		1.	
				Alpha	0	-	100.	3121.84
P	11350	-	1.3e-22					
				Alpha	0		1.	
				Alpha	0	-	100.	11441.84
P	16626	-	4.22e-21					
				Alpha	0		1.	
				Alpha	0	-	100.	16717.84

Nuclei radioactive decay

z3.a8



# 8LI (839.9 MS)								
#	Excitation	Flag	Halflife	Mode	Ex	flag	Intensity	Q
P	0	-	0.8399					
				BetaMinus	0		1.	
				BetaMinus	3030	-	100	12974.13
P	3210	-	4.22e-21					
				Neutron	0		1	
				Neutron	0	-	50	1177.7
				Neutron	477.6	-	50	700.1
P	5400	-	4.22e-21					
				Neutron	0		1	
				Neutron	0	-	50	3367.7
				Neutron	477.6	-	50	2890.1
P	9670	-	4.22e-21					
				Triton	0		0.8	
				Triton	0	-	80	4280.0
				Neutron	0		0.2	
				Neutron	0	-	10	7637.7
				Neutron	477.6	-	10	7160.1

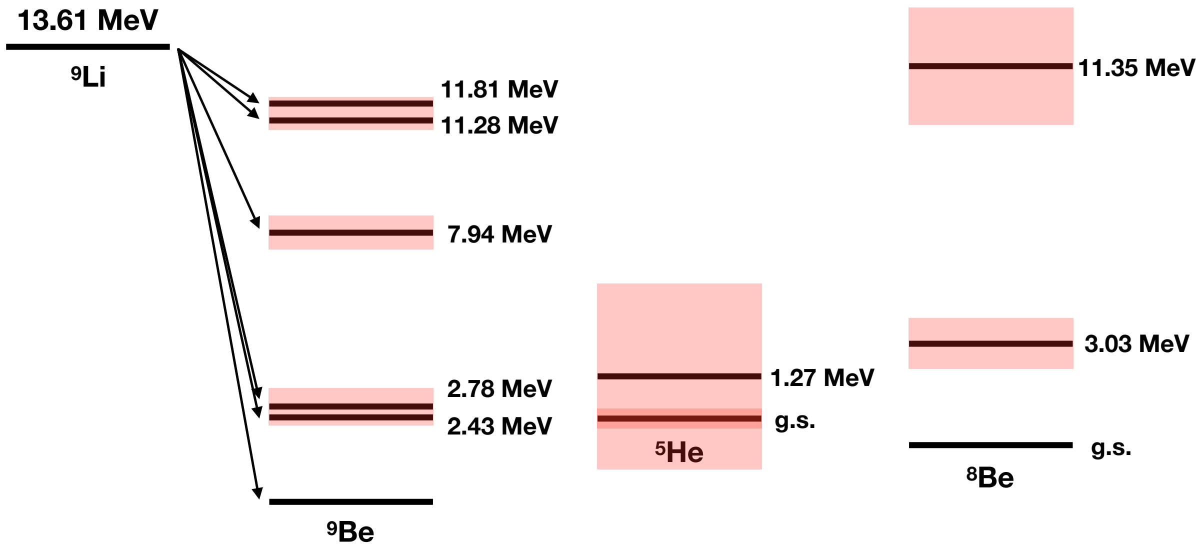
We added a triton decay class based on the alpha decay process:

G4TritonDecay.cc and G4TritonDecay.hh in /processes/hadronic/models/radioactive_decay/

Nuclei radioactive decay

- However all these nuclear states have an important energy uncertainties (normal width in particular for neutron emission states and spectroscopy uncertainties).
- Consequently the emitted particles are not mono-energetic.
- To reproduce better the data, it would be interesting to add the width of the states. Would it be a project to add the uncertainties of the width for the decay of nuclei?

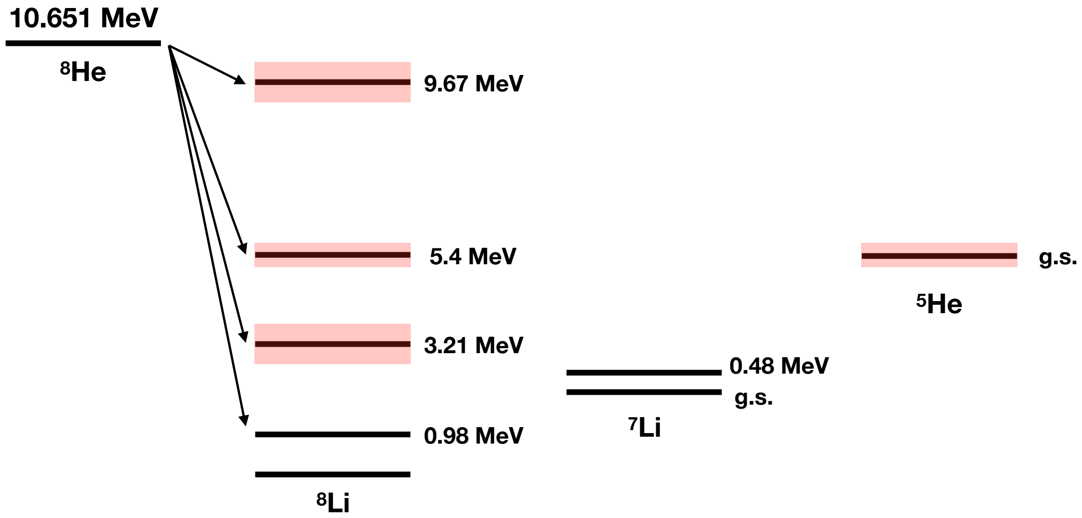
^{9}Li



$^{9}\mathrm{Li} \longrightarrow {}^{9}\mathrm{Be} + e^{-} + \bar{\nu}_e$
$\hookrightarrow {}^{8}\mathrm{Be} + n$
$\hookrightarrow \alpha + \alpha$

$^{9}\mathrm{Li} \longrightarrow {}^{9}\mathrm{Be} + e^{-} + \bar{\nu}_e$
$\hookrightarrow {}^{5}\mathrm{He} + \alpha$
$\hookrightarrow \alpha + n$

^{8}He



$^{8}\mathrm{He} \longrightarrow {}^{8}\mathrm{Li} + e^{-} + \bar{\nu}_e$
$\hookrightarrow {}^{7}\mathrm{Li} + n$

For ^{8}Li of 9.67 MeV only*

$^{8}\mathrm{He} \longrightarrow {}^{8}\mathrm{Li} + e^{-} + \bar{\nu}_e$
$\hookrightarrow {}^{5}\mathrm{He} + {}^{3}\mathrm{H}$
$\hookrightarrow \alpha + n$

Conclusions

- We implemented the positronium generator instead of standard e+ annihilation. The lifetime and the probability of positronium are dependent of the material.
- We created several files to reproduce the multiplicity and the energy of gammas for neutron radiative captures.
- We customized the Cerenkov process in order to take into account the dependance of the refractive index with the photons energy.
- We developed a complex optical model to better reproduce the photons interactions with the PMTS. This has been accompanied by a customisation of the boundary processes.
- We would be also interested in a more precise description of radioactive nuclei decay taking into account of the width of the energy states.

Backup slides

G4OpBoundaryProcess : customized for JUNO PMT Optical Model (POM)

Custom Boundary Process : Advantages

- natural geometry, no fakes
- standard Geant4 polarization, propagation, time
- less code, simpler code
- simpler Geant4 step history (no fakes)
- same geometry on GPU+CPU, easier Opticks validation
- half the geometry objects to model PMT (4->2)

Old FastSim POM	4 Solid, 4 LV, 4 PV
Custom Boundary POM	2 Solid, 2 LV, 2 PV

Disadvantages

- maintain Custom4 C4OpBoundaryProcess
- updating Geant4 needs care if G4OpBoundaryProcess changed

Advantages far outweigh disadvantages

- **JUNOSW MERGED May 25, 2023**

https://github.com/simoncblyth/customgeant4/

Custom Boundary POM

```
+---------------pmt-Pyrex----------------+
|                                        |
|                                        |
|                                        |
|       +~inner~Vacuum~~~~~~~~~~~+       |
|       !                        !       |
|       !                        !       |
|       !                        !       |
|       !                        !       |
|       !                        !       |
|       +                        +       |
|       |                        |       |
|       |                        |       |
|       |                        |       |
|       |                        |       |
|       |                        |       |
|       +------------------------+       |
|                                        |
|                                        |
|                                        |
+----------------------------------------+
```

`OpSurfaceName[0] == '@'`

- `local_z>0` : does MultiLayer ART calc
- `!(local_z>0)` : standard *mirror_opsurf*

C4OpBoundaryProcess::PostStepDoIt : 3-way (A,R,T) Customization

G4OpBoundaryProcess unless OpticalSurfaceNam[0] == '@'/'#'

```
if( OpticalSurfaceName0 == '@' || OpticalSurfaceName0 == '#' )
{
    if( m_custom_art->local_z(aTrack) < 0. ) // Lower hemi : Standard
    {
        m_custom_status = 'Z' ;
    }
    else if( OpticalSurfaceName0 == '@') // MultiFilm ART POM
    {
        m_custom_status = 'Y' ;
        m_custom_art->doIt(aTrack, aStep) ;

        type = dielectric_dielectric ;
        theModel = glisur ;
        theFinish = polished ;
    }
    else if( OpticalSurfaceName0 == '#' ) // Traditional POM
    {
        m_custom_status = '-' ;

        type = dielectric_metal ;
        theModel = glisur ;
        theReflectivity = 0. ;
        theTransmittance = 0. ;
        theEfficiency = 1. ;
    }
}
```

PostStepDoIt type switch

```
if( m_custom_status == 'Y' ) // CustomART handling
{
    G4double rand = G4UniformRand();
    if ( rand < theAbsorption )
    {
        DoAbsorption();      // A
    }
    else
    {
        DielectricDielectric();  // R or T
    }
}
else if (type == dielectric_metal)
{
      DielectricMetal();
}
...
```

Standard DoAbsorption DielectricDielectric reused

DielectricDielectric expects 2-way (**theReflectivity,theTransmittance**) => so `C4CustomART.h` rescales 3-way (**A,R,T**)

`(theAbsorption,theReflectivity,theTransmittance) <= (A, R/(1-A), T/(1-A))`

custom4/C4CustomART.h : include into C4OpBoundaryProcess.cc

```
#include "C4IPMTAccessor.h"
#include "C4MultiLayrStack.h"
#include "C4Touchable.h"

struct C4CustomART {
    const C4IPMTAccessor* accessor ;
    G4double& theAbsorption ;    // doIt sets these
    G4double& theReflectivity ;
    G4double& theTransmittance ;
    G4double& theEfficiency ;

    const G4ThreeVector& theGlobalPoint ;
    const G4ThreeVector& OldMomentum ;
    const G4ThreeVector& OldPolarization ;
    const G4ThreeVector& theRecoveredNormal ;
    const G4double& thePhotonMomentum ;

    C4CustomART(
        const C4IPMTAccessor* accessor,
        G4double& theAbsorption,
        G4double& theReflectivity,
        G4double& theTransmittance,
        G4double& theEfficiency,
        const G4ThreeVector& theGlobalPoint,
        const G4ThreeVector& OldMomentum,
        const G4ThreeVector& OldPolarization,
        const G4ThreeVector& theRecoveredNormal,
        const G4double& thePhotonMomentum
    );
    double local_z( const G4Track& aTrack );
    void doIt(const G4Track& aTrack, const G4Step& aStep );
};
```

https://github.com/simoncblyth/customgeant4/blob/main/C4CustomART.h ❏

C4CustomART customizations

doIt C4MultiLayrStack.h TMM calc, only for:

- OpticalSurfaceName[0]=='@' && local_z > 0
- -> **Absorption, Transmittance, Reflectivity, Efficiency**
- changed via references collected by ctor

reuse standard G4OpBoundaryProcess

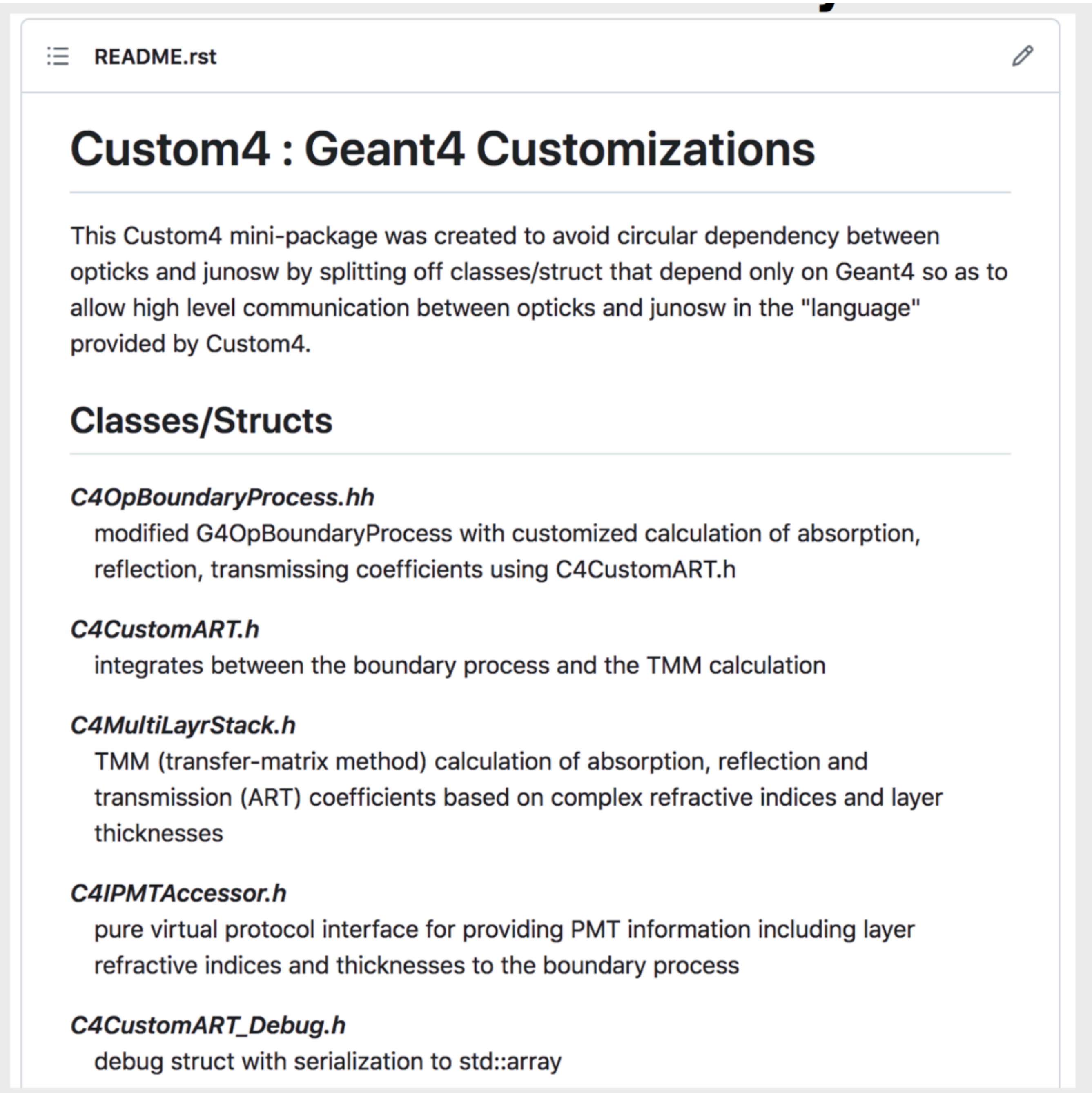
- standard reflect, refract, absorb, detect
- standard polarization
- standard time, speed
- **"inject" minimal change needed for POM**

G4OpBoundaryProcess modifications

From Simon Blyth (IHEP): simon.c.blyth@gmail.com

- The JUNO PMT simulation uses a developed PMT optical model (POM) that allows photons to enter the PMT and bounce around inside and sometimes refract out again to possibly continue on to other PMTs.
- Also the POM accounts for thin film interference effects on the stack of layers: Pyrex, Anti reflection coating (ARC), Photocathode (PHC), Vacuum.
- The calculation yields A,R,T (absorption, reflection, transmission) probabilities uses as input the thicknesses of the ARC and PHC layers and refractive indices for the 4 layers. The two inner layers can have a complex refractive index, allowing absorption.
- The transverse matrix method (TMM) is used to sum up the contribution of interference using complex refractive indices for ARC and PHC.
- Thus with A,R,T, E the standard boundary is modified in a way that reuses as much of standard Geant4 as possible. The customisation allows the minimum change to be "injected" to implement the POM.

G4OpBoundaryProcess modifications



README.rst

Custom4 : Geant4 Customizations

This Custom4 mini-package was created to avoid circular dependency between opticks and junosw by splitting off classes/struct that depend only on Geant4 so as to allow high level communication between opticks and junosw in the "language" provided by Custom4.

Classes/Structs

C4OpBoundaryProcess.hh
modified G4OpBoundaryProcess with customized calculation of absorption, reflection, transmissing coefficients using C4CustomART.h

C4CustomART.h
integrates between the boundary process and the TMM calculation

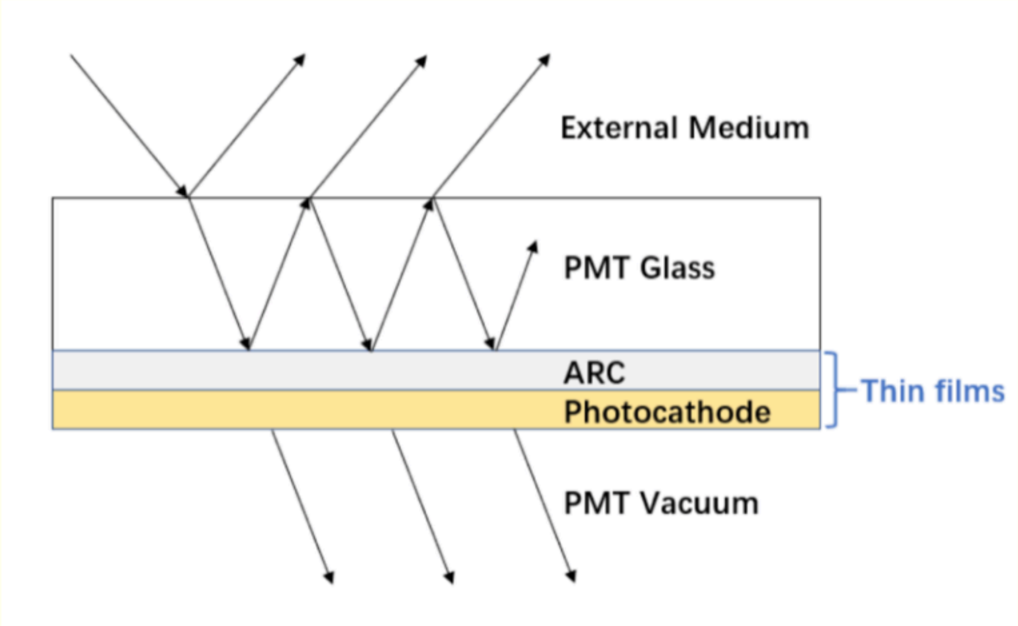
C4MultiLayrStack.h
TMM (transfer-matrix method) calculation of absorption, reflection and transmission (ART) coefficients based on complex refractive indices and layer thicknesses

C4IPMTAccessor.h
pure virtual protocol interface for providing PMT information including layer refractive indices and thicknesses to the boundary process

C4CustomART_Debug.h
debug struct with serialization to std::array

https://github.com/simoncblyth/customgeant4/ ❏

TMM : Transfer Matrix Method



multi-layer thin films, coherent calc:

- complex refractives indices, thicknesses
- => (A,R,T) (Absorb, Reflect, Transmit) + E (Efficiency)
- Used from **C4OpBoundaryProcess**

header-only GPU/CPU : **C4MultiLayrStack.h**

- `Custom4` depends only on Geant4
- Dependency of JUNOSW and Opticks