

Scintillation Light in LArTPCs: Simulation and Reconstruction

Patrick Green

University of Oxford

UK LArSoft Workshop, Edinburgh, October 2026

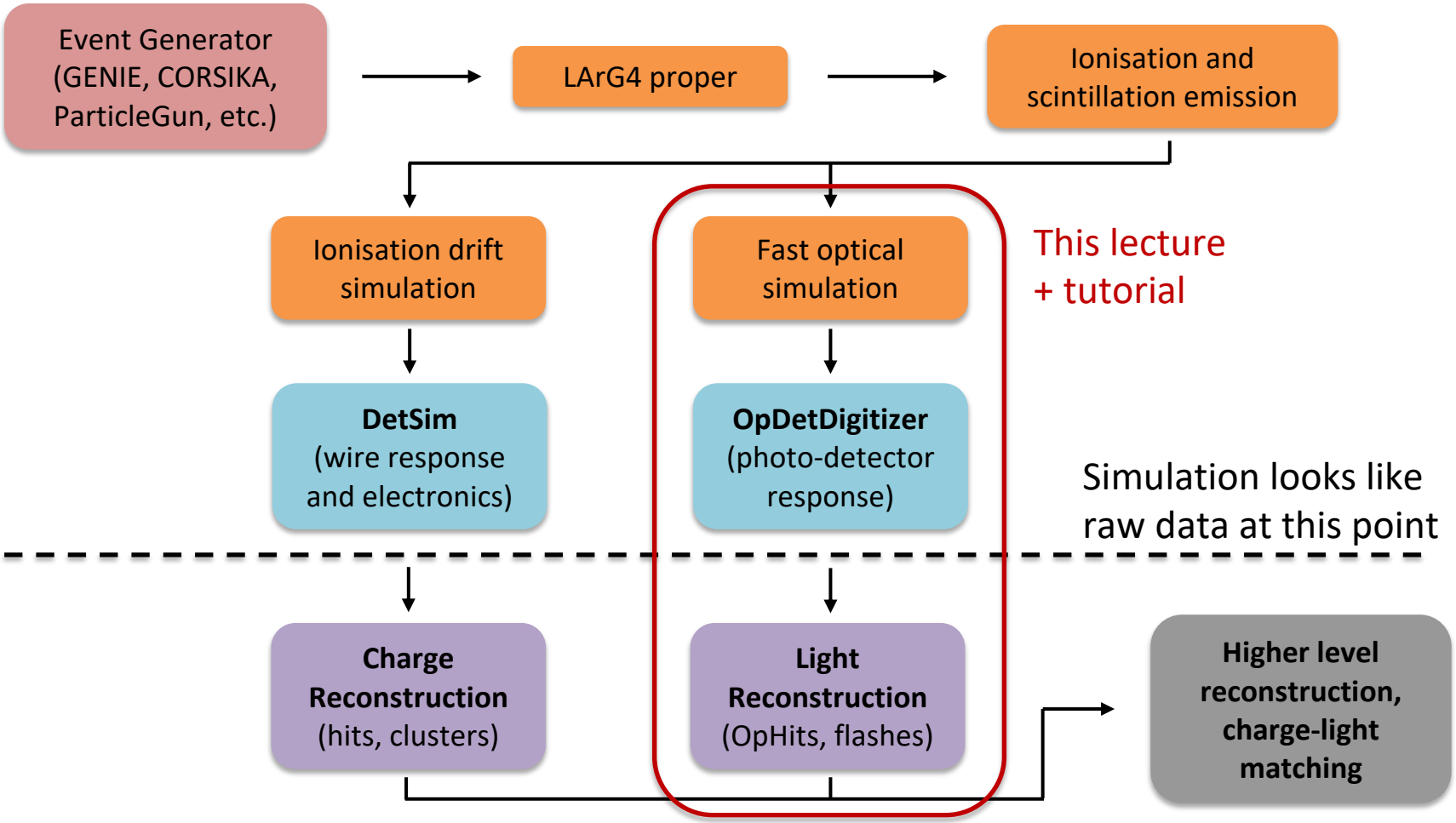
Outline

This talk will cover scintillation light simulation and reconstruction in LArSoft

- physics of scintillation light in LArTPCs
- the different simulation modes available
- scintillation light reconstruction

Next, we will have a hands-on session where you will get to have a go running the scintillation light simulation and reconstruction for yourself

Simulation flowchart



Each stage is a module (or set of modules)

Each stage passes data products, “objects”, to the next stage

Elements of light simulation







Light Source:

- How many photons are produced?
- What is their wavelength?
- What is their time distribution?

Transport:

- How many photons make it to the photon detector?
- How long does it take them?
- Do they scatter / get absorbed / reflected, etc.?

Detection:

- What is our detection efficiency?
- Are there any extra timing effects?

Different modes of simulation

Full optical simulation (extremely slow)

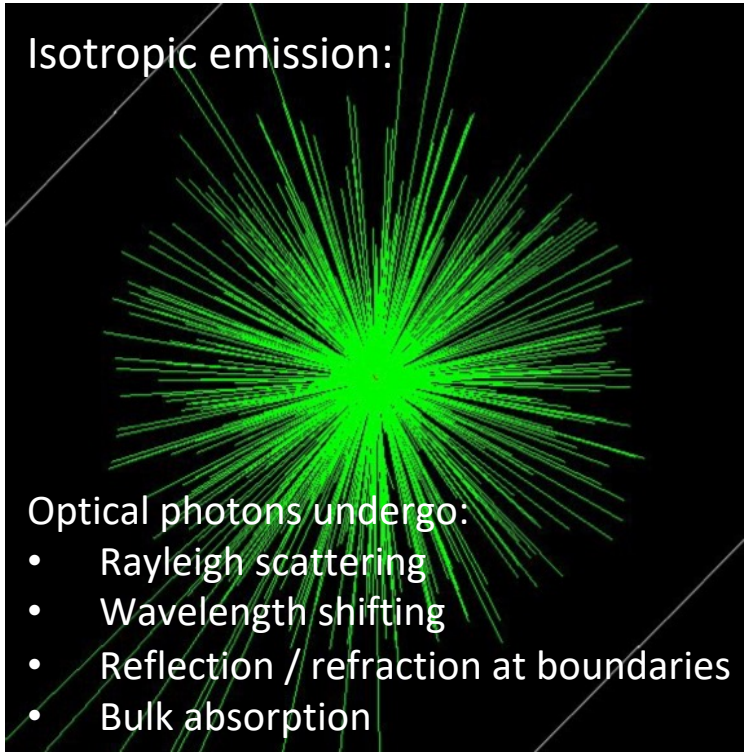
- propagate each scintillation photon individually
- extremely slow, but performance can (and is) enhanced using GPUs

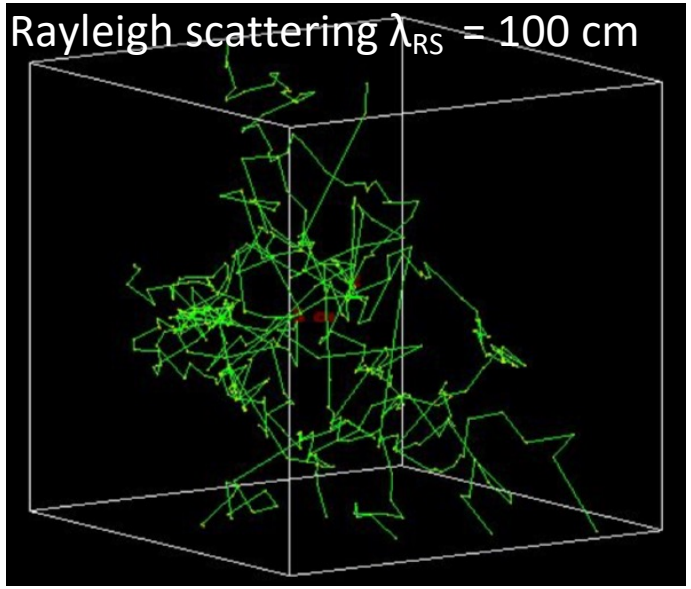
Fast optical simulation (faster, but less precise)

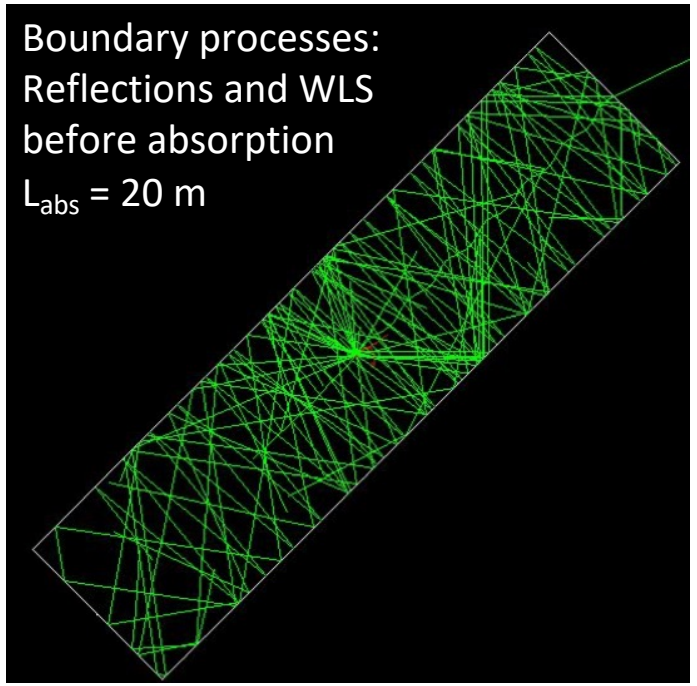
- still need to run full optical at least once
- majority of optical properties "burned in"
- three primary methods exist:
  optical library, semi-analytical, generative neural network

Full optical simulation

Scintillation yield ~24000 photons / MeV — for GeV-scale interactions in large detectors, tracking each individual photon is prohibitively slow -> alternative approaches needed





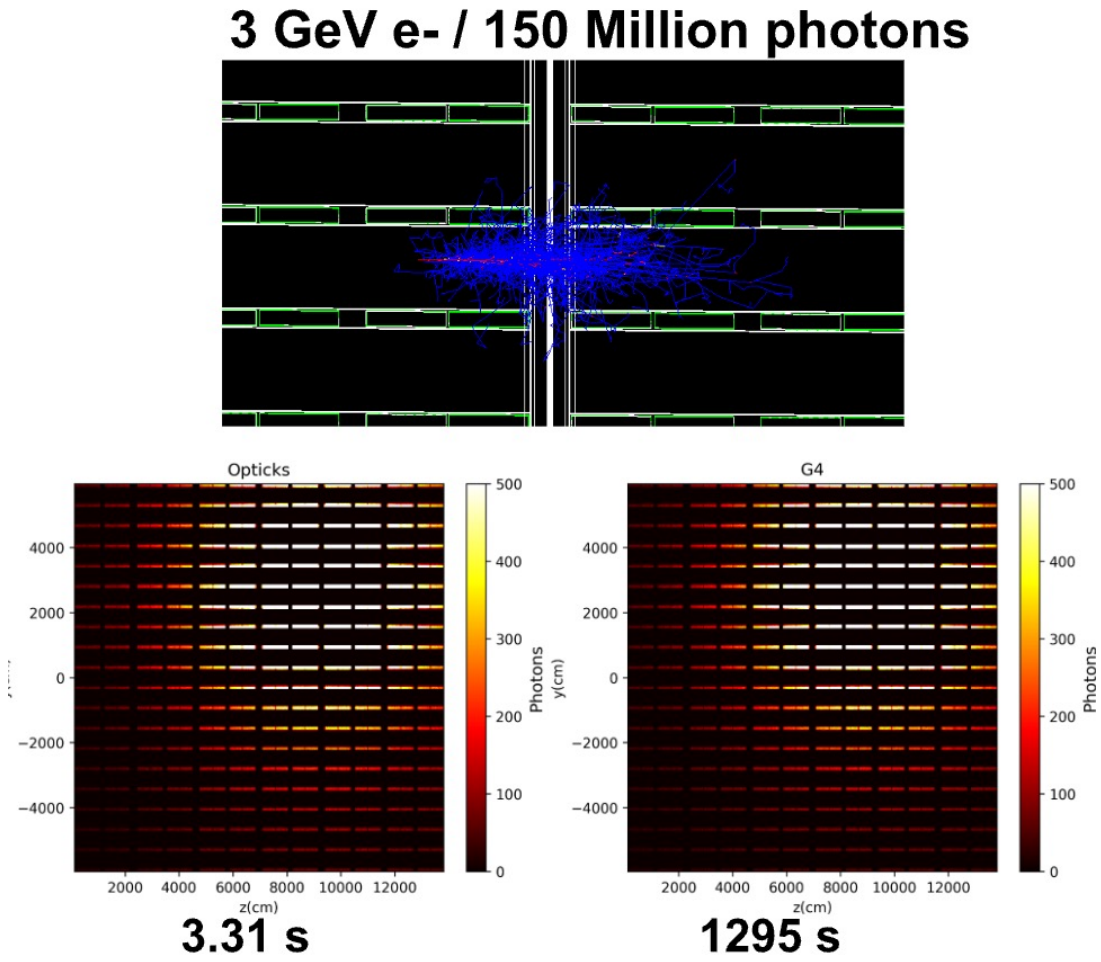


Full optical simulation using GPUs

On-going efforts to use GPUs for full optical simulation in DUNE

1. Opticks, using NVIDIA OptiX
   - developed for JUNO and adapted for DUNE
     EPJ Web Conf. 337 (2025) 01093, arXiv:2608.27306
2. Celeritas
   - developed for LHC experiments (CMS, ATLAS)
     EPJ Web Conf. 295 (2024) 11005

Different computation challenges: GPU resources more limited + in some cases using vendor specific tools (NVIDIA OptiX)



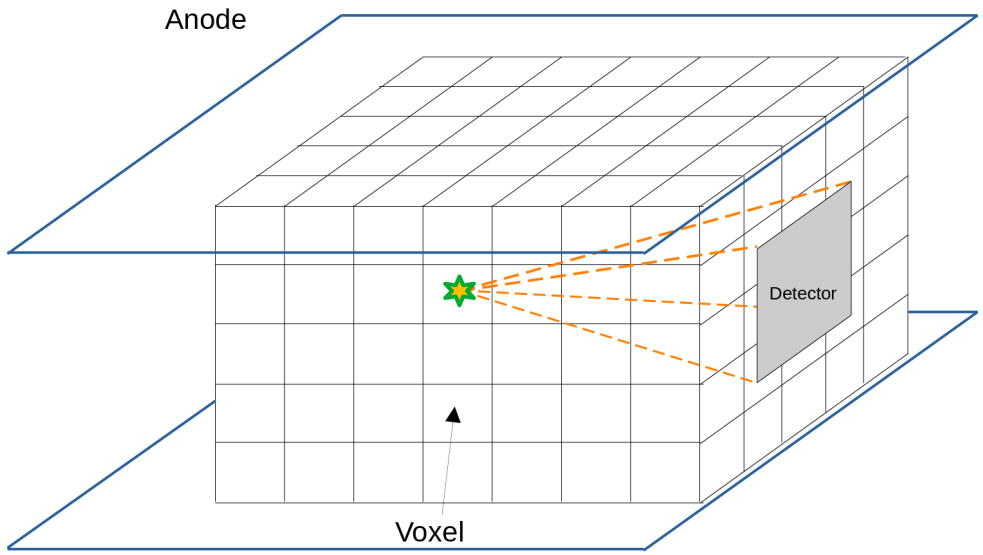
Opticks example, 400× speed-up (Nvidia RTX 5090)
Image: I. Parmaksiz, A. Higuera

Fast optical simulation: optical library

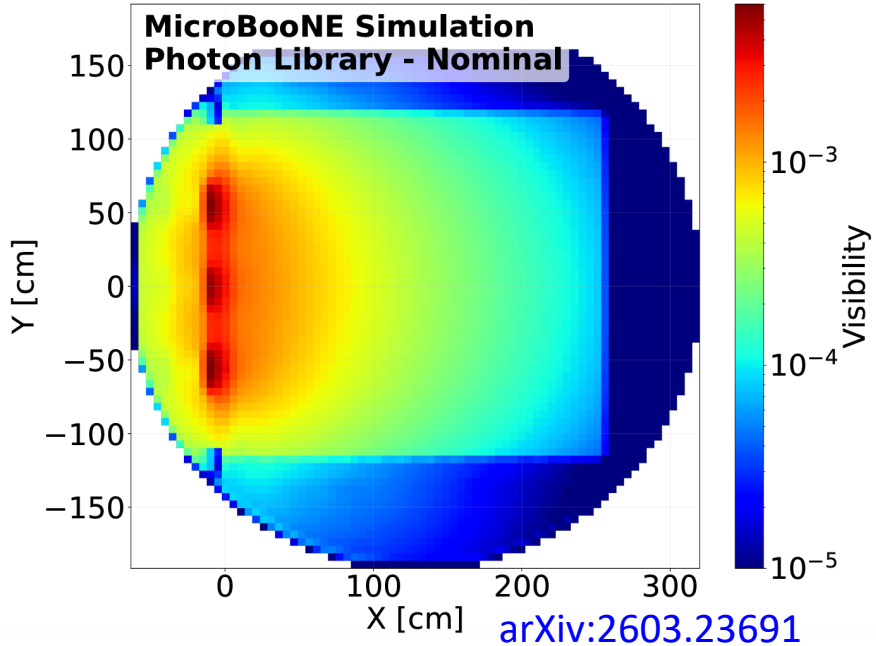
Map of visibilities: probabilities of photons from each voxel reach each photo-detector

Size scales with LAr volume, voxel size (granularity), number of photo-detectors – memory and disk

- used in MicroBooNE, but prohibitive size for larger detectors (DUNE) or many photo-detectors (SBND)



Typical voxel size: 5×5×5 cm^3



$$N_{photons} = \frac{dE}{dx} \times \Delta x \times LY \times \text{Visibility}$$

arXiv:2603.23691

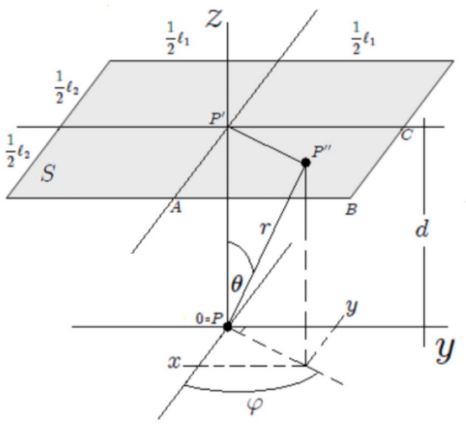
Fast optical simulation: semi-analytical model

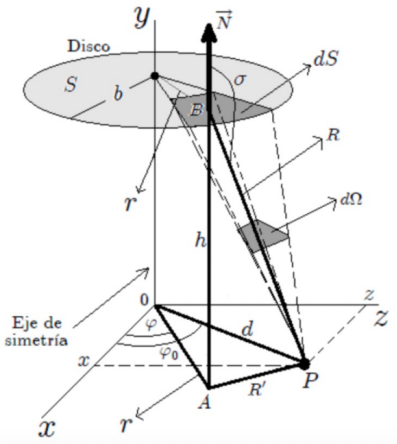
Isotropic emission makes the problem *almost* geometric, can calculate analytically

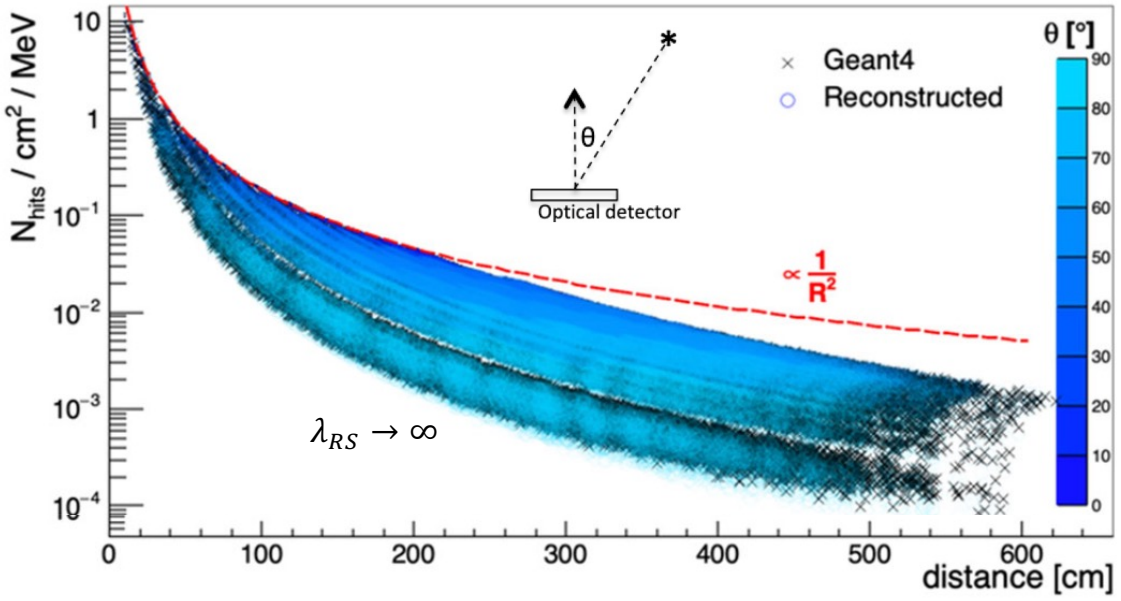
- *almost* because we have Rayleigh scattering and border effects – need to correct for via parametrization

Current approach used in SBND and DUNE (far detectors + some protoDUNE geometries)

$$N_\Omega = e^{-\frac{d}{\lambda_{abs}}} \Delta E \cdot S_\gamma(\mathscr{E}) \frac{\Omega}{4\pi}$$







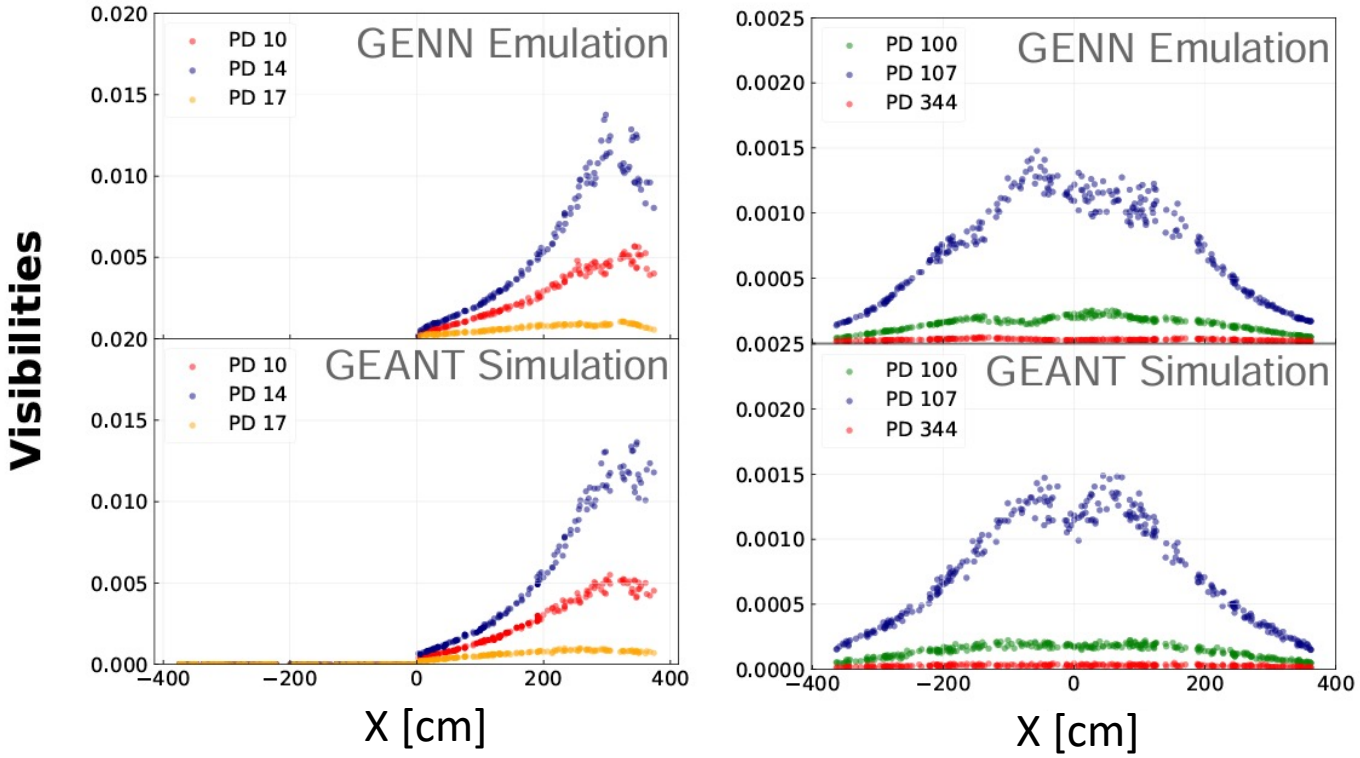
Eur. Phys. J. C 81 (2021) 4, 349

Fast optical simulation: neural networks

Generative neutral network (GENN):

- network trained on Geant4 simulation of voxelized detector, similar to a library
- rather than storing visibilities in look-up table, use network to predict at run-time
- avoids memory limitation of library, at some CPU cost

Available in DUNE (some geometries)



Mach. Learn. Sci. Tech. 3 (2022) 1, 015033



Emission

Scintillation mechanism in LAr

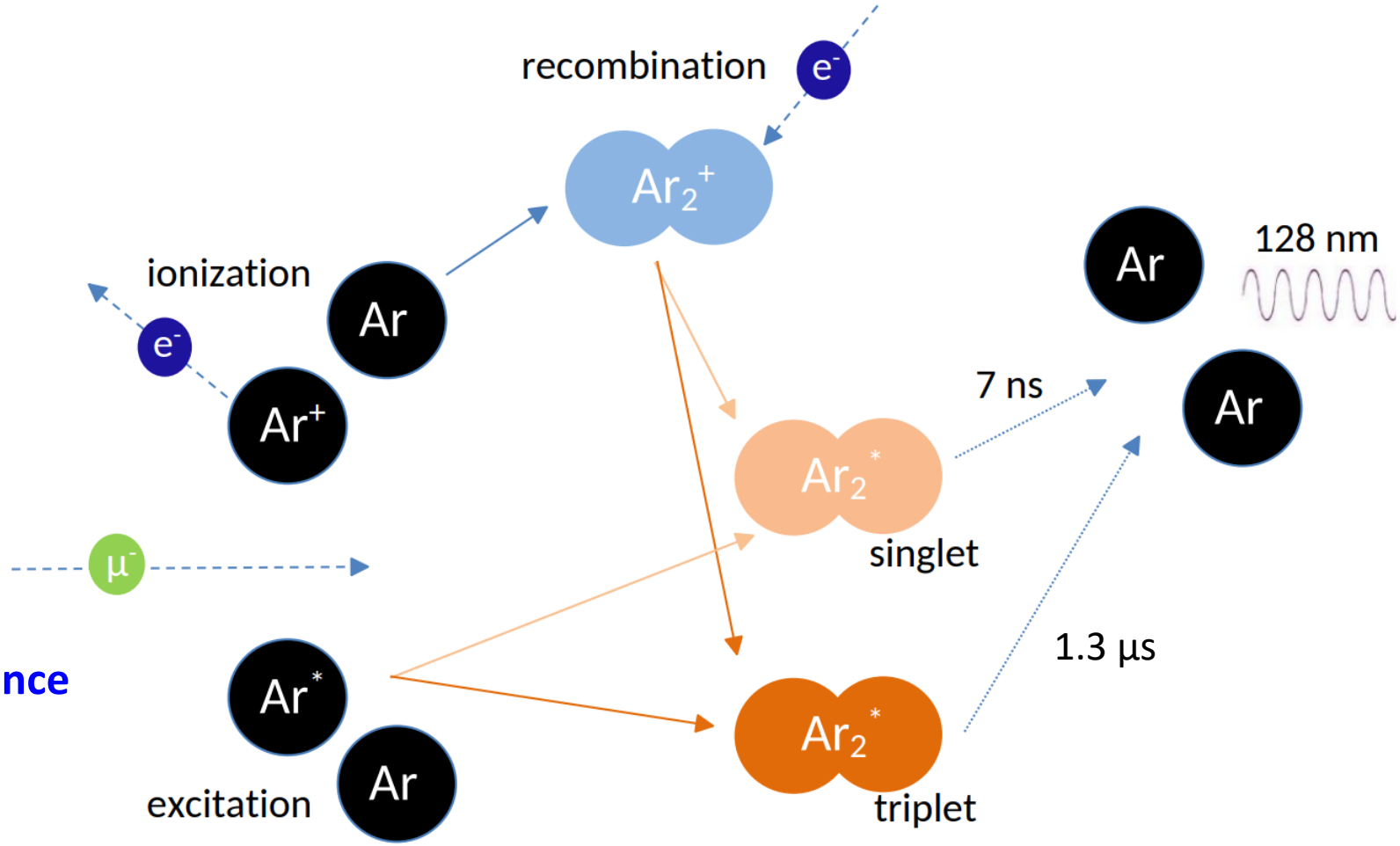
- **Recombination luminescence**

$Ar^+ + Ar \longrightarrow Ar^+_2$

$Ar^+_2 + e \longrightarrow Ar^*_2 \longrightarrow 2Ar + h\nu$

- **Self-trapped excitation luminescence**

$Ar^* + Ar \longrightarrow Ar^*_2 \longrightarrow 2Ar + h\nu$



LAr scintillation light properties

Argon is a prolific scintillator:

~40000 photons/MeV @ zero electric field

Emission is in vacuum-ultraviolet (VUV) spectrum:

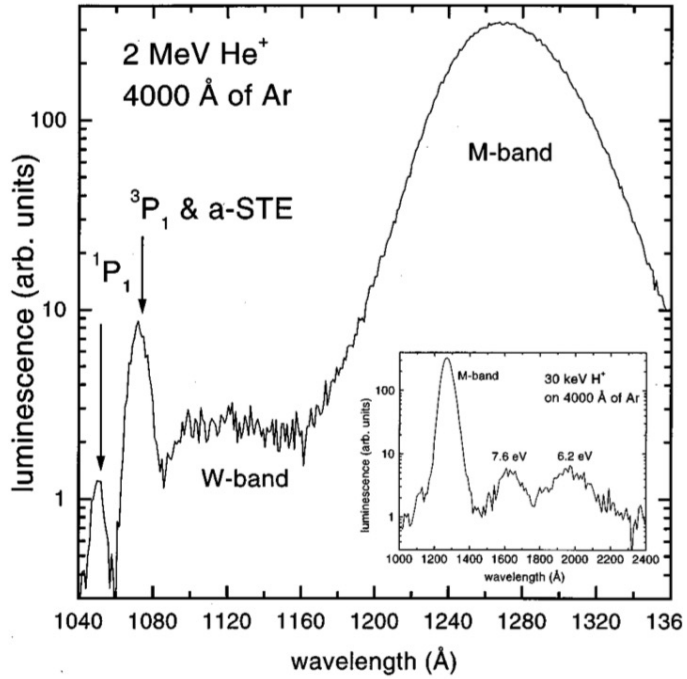
~128 nm (FWHM ≃ 6 nm)

Argon excimer decays characterized by two very different time components (singlet / triplet):

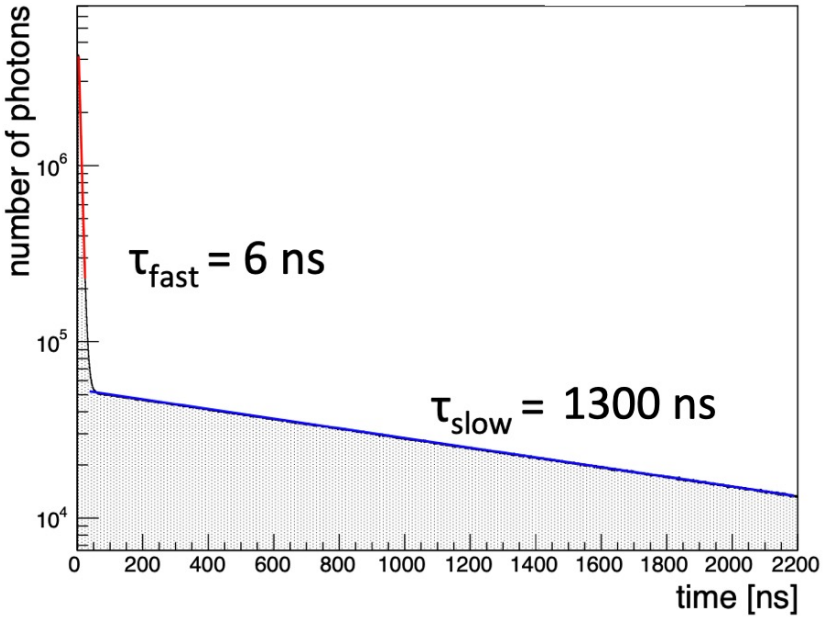
a *fast component*, $\tau_s \sim 6$ ns

and a *slow component*, $\tau_T \sim 1.3$ µs *

*Note: when the slow time constant is convolved with the emission time of a commonly used wavelength-shifter (TPB) this results in a larger value ≈ 1.5-1.6µs



Phys. Rev. B 56 (1997), 6975



LAr scintillation light properties in LArSoft

In LArSoft the scintillation light properties are all parameterised in either:

- old way “legacy” — via fhicl: lardataalg/DetectorInfo/larproperties.fcl [MicroBooNE, DUNE (partial)]
- new way “refactored” — all material properties defined directly in GDML [SBN, DUNE (partial)]
- may then be overwritten in individual experiment repositories

These properties are read when running the *full optical simulation*

GDML (SBND – “_full_simulation.gdml”)

```
<matrix name="FASTSCINT" coldim="2" values="7.2*eV 0.0
                                            7.9*eV 0.04
                                            8.3*eV 0.12
                                            8.6*eV 0.27
                                            8.9*eV 0.44
                                            9.1*eV 0.62
                                            9.3*eV 0.8
                                            9.6*eV 0.91
                                            9.7*eV 0.92
<matrix name="FTC" coldim="1" values="6." unit="ns" />
<matrix name="STC" coldim="1" values="1300." unit="ns"/>
```

lardataalg / lardataalg / DetectorInfo / **larproperties.fcl**

```
# Optical properties
# Fast and slow scintillation emission spectra, from [J Chem Phys vol 91 (1989) 1469]
FastScintEnergies:    [ 7.2,  7.9,  8.3,  8.6,  8.9,  9.1,  9.3,  9.6,  9.7,  9.8,  10
SlowScintEnergies:    [ 7.2,  7.9,  8.3,  8.6,  8.9,  9.1,  9.3,  9.6,  9.7,  9.8,  10
FastScintSpectrum:    [ 0.0,  0.04, 0.12, 0.27, 0.44, 0.62, 0.80, 0.91, 0.92, 0.85, 0.
SlowScintSpectrum:    [ 0.0,  0.04, 0.12, 0.27, 0.44, 0.62, 0.80, 0.91, 0.92, 0.85, 0.
ScintResolutionScale: 1.     # resolution factor used by G4 scintillation
ScintFastTimeConst:   6.     # fast scintillation time constant (ns)
ScintSlowTimeConst:   1590.  # slow scintillation time constant (ns)
```

Scintillation Yields: E-field

Strength of the electric field applied to the LAr impacts the amount of recombination
→ alters amount of charge (Q) and light (L)

Effect is anti-correlated, as electric field increases Q grows, L decreases. At 500 V/cm, energy deposit about equally divided between Q and L

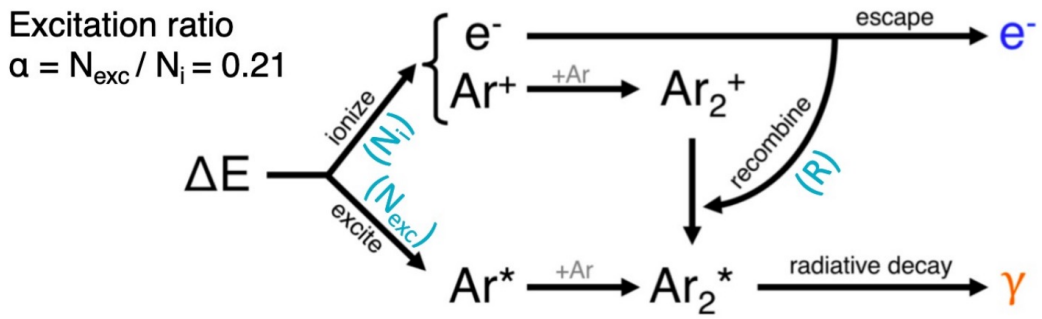
In LArSoft modelled in ISCalcCorrelated:

larsim / larsim / IonizationScintillation / ISCalcCorrelated.cxx

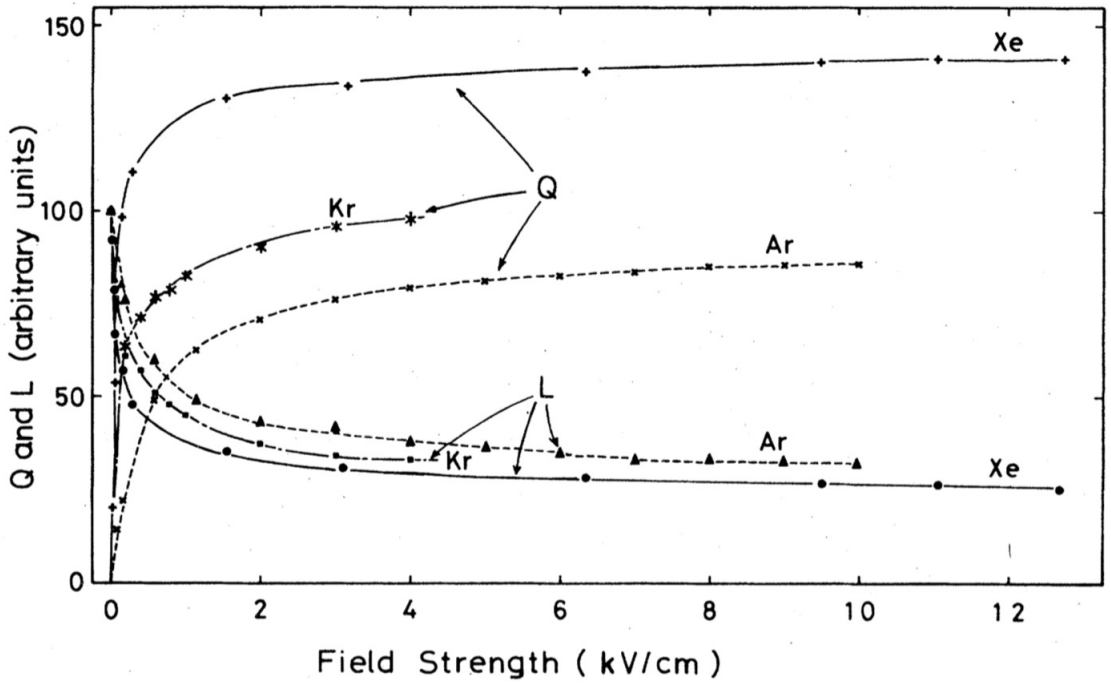
```
// using this recombination, calculate number of ionization electrons
double const num_electrons = (energy_deposit / fWion) * recomb;

// calculate scintillation photons
double const num_photons = (Nq - num_electrons) * fScintPreScale;
```

Services.LArG4Parameters.IonAndScintCalculator: "Correlated"



PHYSICAL REVIEW D **101,** 012010 (2020)



Phys. Rev. B 20, 3486 (1979)

Scintillation Yields: Particle Type

Light yield and fast/slow ratio depend on how ionising the particles are:

- muons: ~24000 photons/MeV, ~ 25% prompt light
- alphas: ~16800 photons/MeV, ~ 60% prompt light

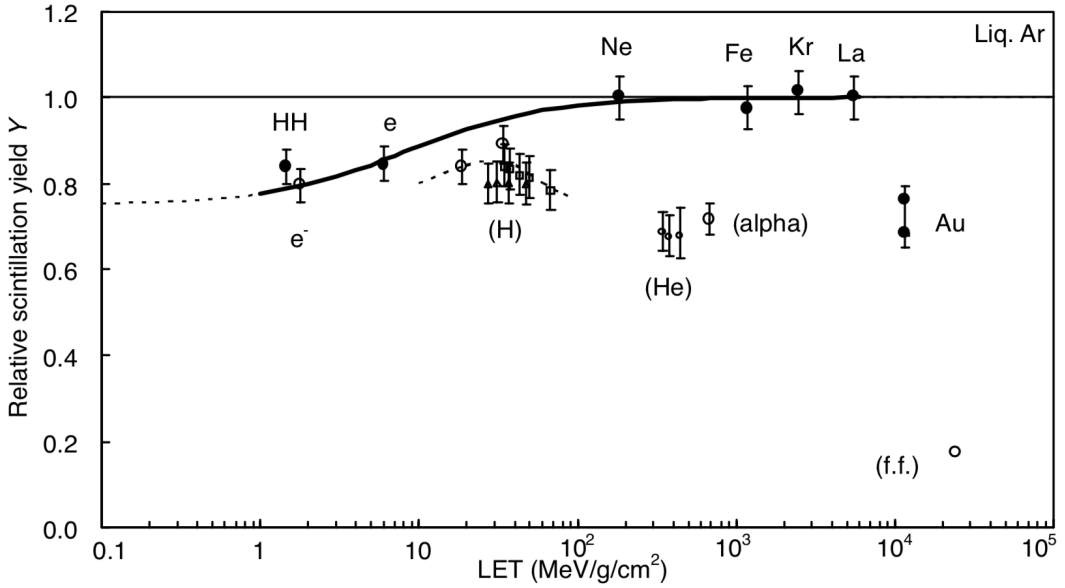
In LArSoft this is configured in larproperties.fcl, ScintByParticleType

In DUNE: LArQL model
[JINST 17 C07009 (2022)]

- combined charge–light model with corrections for dE/dx, E-field, etc.

```
ScintByParticleType: true

# Scintillation yields and fast/slow ratios
MuonScintYield:          24000
MuonScintYieldRatio:     0.23
PionScintYield:          24000
PionScintYieldRatio:     0.23
ElectronScintYield:      20000
ElectronScintYieldRatio: 0.27
KaonScintYield:          24000
KaonScintYieldRatio:     0.23
ProtonScintYield:        19200
ProtonScintYieldRatio:   0.29
AlphaScintYield:         16800
AlphaScintYieldRatio:    0.56
```



Particle	τ_S	τ_T	I_S/I_T	Reference
Electron	6.3 ±0.2	1020±60	0.083	Kubota *et al.*[a]
	(5.0 ±0.2)	(860±30)	(0.045)	(E = 6 kV/cm)[a]
	4.6	1540	0.26	Carvalho and Klein[b]
	4.18±0.2	1000±95		Keto *et al.*[c]
		1110±50		Suemoto and Kanzaki[d]
	6 ±2	1590±100	0.3	This work
α	~5	1200±100		Kubota *et al.*[e]
	4.4	1100	3.3	Carvalho and Klein[b]
	7.1±1.0	1660±100	1.3	This work
F.F.	6.8±1.0	1550±100	3	This work

Jpn. J. Appl. Phys. Vol. 41 (2002) pp. 1538–1545;
Ph. Rev. B 27 (1983), 5279

Propagation



Scintillation light propagation

Next step is to calculate how many photons reach our photon-detectors ⇒ **transport effects**

Emitted photons are lower energy than the first excited state of the Ar atom

⇒ pure LAr is transparent to its own scintillation radiation

However, during propagation photons may undergo elastic scattering on Ar atoms

⇒ Rayleigh scattering

Transport effects impact, in a non-negligible way, the light signals that our photon detectors see

It's important to understand and model this properly in liquid argon.

Rayleigh scattering in Argon

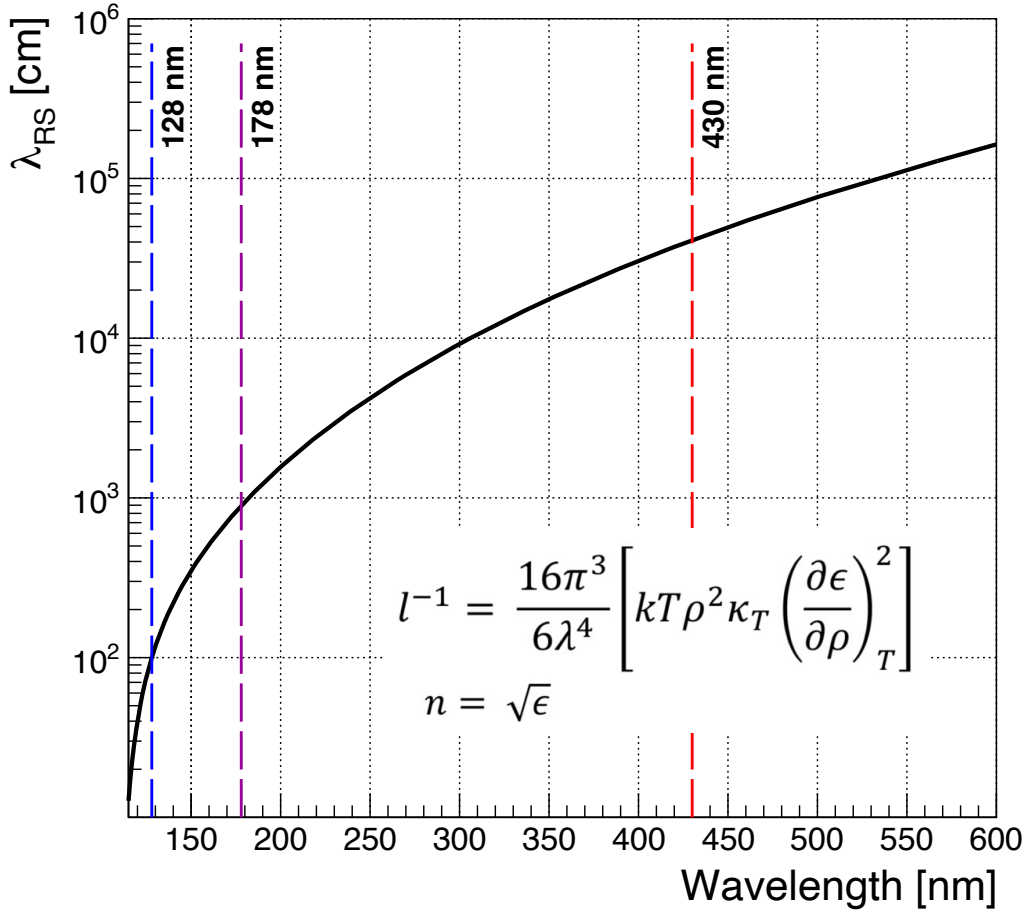
Rayleigh scattering length for VUV photons in Ar measured variously between ~50 – 120 cm

- hard to measure, small uncertainties in the index of refraction drastically change the scattering length λ_{RS}
- most recent measurement around 100 cm, adopted by most LArTPC experiments [JINST 15 P09009 2020]

λ_{RS} ~100cm << typical size of LArTPC detectors

→ has significant impact on light seen

In LArSoft: larproperties.fcl (legacy) or in GDML (new)



GDML (SBND example)

```
<matrix name="RAYL" coldim="2" values="1.18626*eV 1200800*cm
                                       1.68626*eV 390747*cm
                                       2.18626*eV 128633*cm
                                       2.68626*eV 54969.1*cm
                                       3.18626*eV 27191.8*cm
```

lardataalg / lardataalg / DetectorInfo / **larproperties.fcl**

```
RayleighEnergies: [ 1.18626, 1.68626, 2.18626, 2.68626, 3.18626, 3.686
RayleighSpectrum: [ 1200800, 390747, 128633, 54969.1, 27191.8, 14853.7
```

Detector materials – absorption / reflection

Detector materials (field cage, etc) may reflect or absorb scintillation photons

- can significantly impact light observed, especially important for emission close to the borders or in the corners of the detector – border effects

To model this need to define the properties of every material present

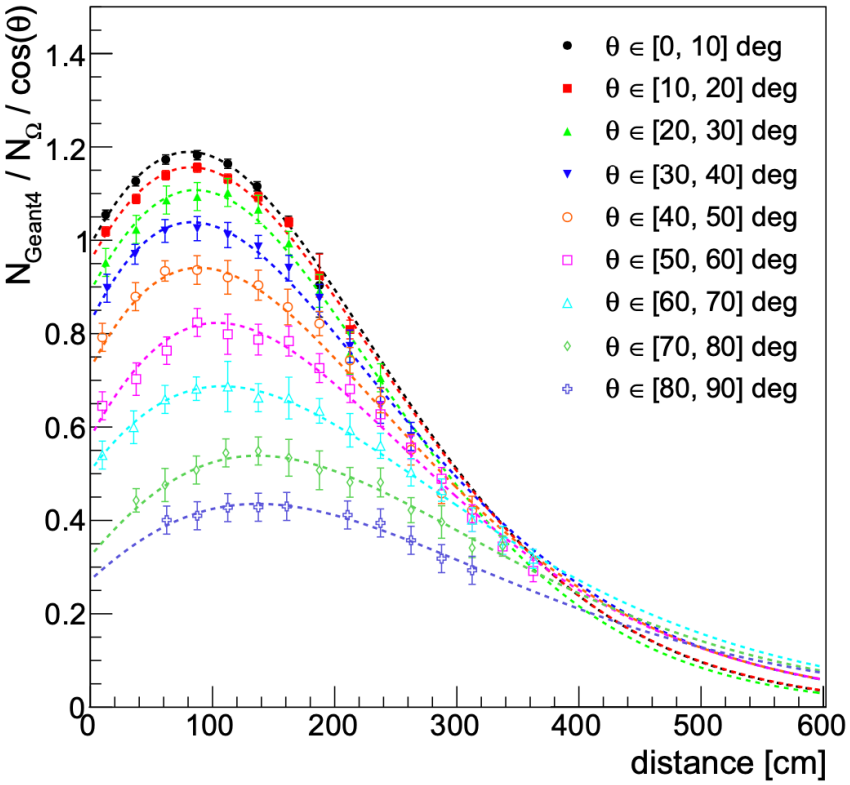
- many not massively well known
- defined either in larproperties.fcl (legacy) or in GDML (new)

GDML
(SBND example)

```
<matrix name="REFLECTIVITY_STEEL_STAINLESS_Fe7Cr2Ni"
                    e7Cr2Ni" coldim="2" values="1.77*eV 0.66
                       2.0675*eV 0.64
                       2.481*eV 0.62
                       2.819*eV 0.60
                       2.953*eV 0.59
                       3.1807*eV 0.57
                       3.54*eV 0.53
                       4.135*eV 0.47
                       4.962*eV 0.39
                       5.39*eV 0.36
                       7.*eV 0.27
                       15.*eV 0.25"/>
```

Modelling in fast optical simulation

Eur. Phys. J. C 81 (2021) 4, 349



Optical library / GENN network:

- Encoded in *visibilities* in each voxel directly (sort of) from full optical simulation

Semi-analytical model:

- Treated as a correction to the geometric prediction
- Parameterised based on difference between geometric prediction and full optical simulation
- Also correct for border effects in analogous way (reflections/absorption)

In LArSoft models implemented in larsim:

larsim/PhotonPropagation/PDFastSimPVS_module.cc (Library)
larsim/PhotonPropagation/PDFastSimPAR_module.cc (Semi-analytical)
larsimdnn/PhotonPropagation/PDFastSimANN_module.cc (GENN)

$$N_\gamma = N_\Omega \times GH'(d, \theta, d_T)/cos(\theta)$$

Time structure of detected signals

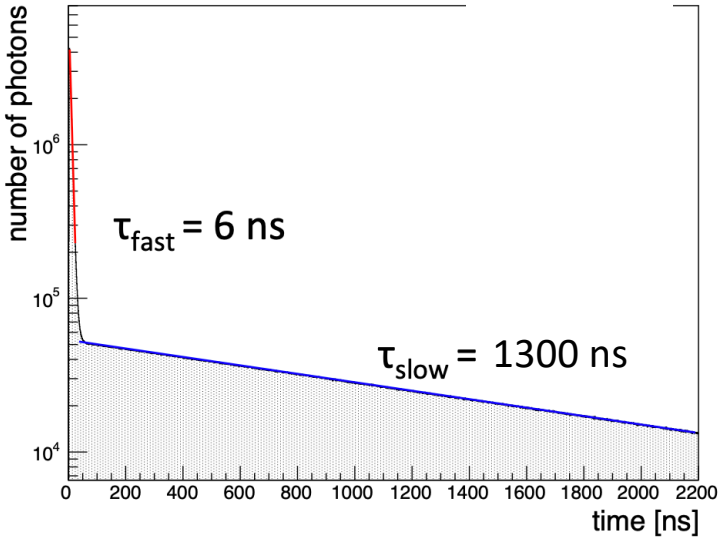
In large detectors transport effects impact the time structure of the observed scintillation light, especially important for the prompt light

$$t_\gamma = t_E + t_t(d,\theta) + t_{WLS} + t_{det},$$

t_E = *emission time*
t_t = *transport time*
t_{WLS} = *WLS delay time*
t_{det} = *detector time*

Scintillation (emission):

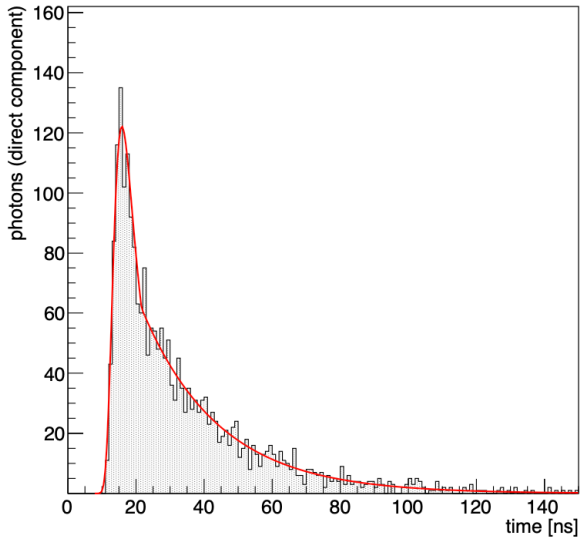
0.3 x τ_{fast}(6 ns) + 0.7 x τ_{slow} (1300 ns)



+

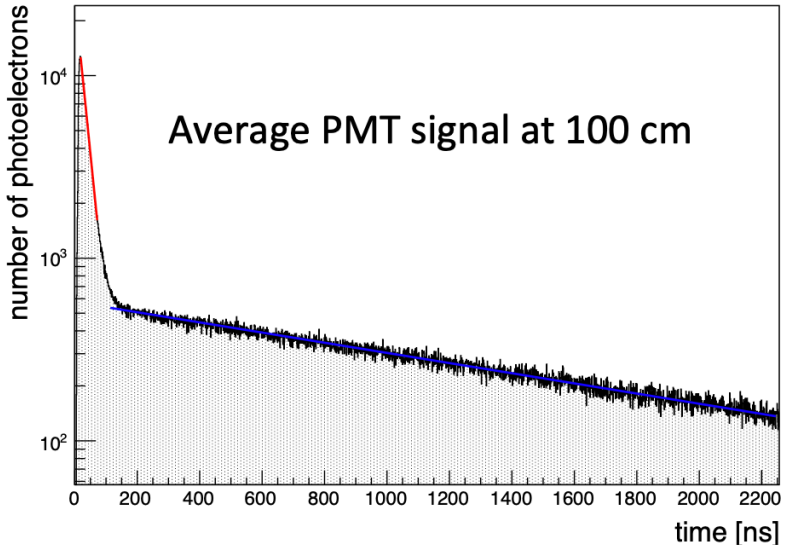
Propagation:

Direct transportation + Rayleigh Scattering



=

Observed:



Time structure of detected signals

In fast optical simulation, modelled using parameterisations of Landau + Exponential fits to full optical simulation

- developed in conjunction with semi-analytical model, but can be used in combination with any approach to get number of photons (library, GENN, etc.)

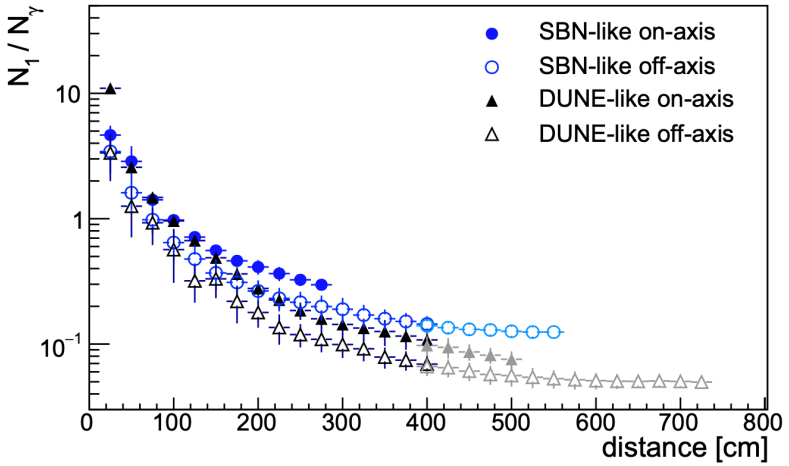
$$t_t(x) = \underbrace{N_1 \frac{1}{\xi} \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} e^{\lambda s + s \log s}\, ds}_{Landau} + \underbrace{N_2\, e^{\kappa x}}_{Exponential},$$

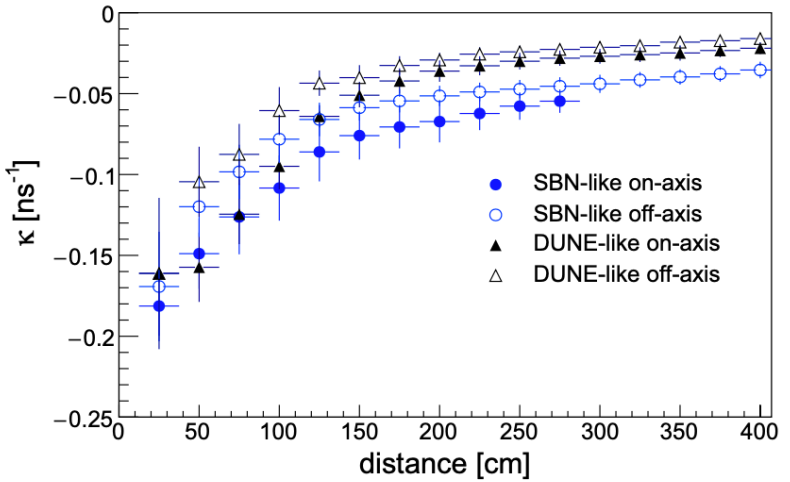
Implemented in LArSoft in larsim:

larsim / larsim / PhotonPropagation / **PropagationTimeModel.h**

larsim / larsim / PhotonPropagation / **PDFastSimPAR.fcl**

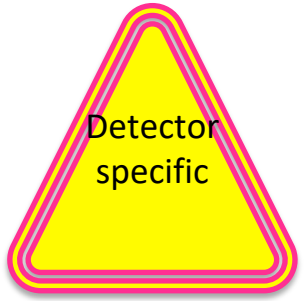
```
IncludePropTime:       true
```





Eur. Phys. J. C 81 (2021) 4, 349

Implementation in SBND



A hybrid model is the default approach used in SBND:

- semi-analytical model (hits + timing) inside TPC
- slimmed-down optical library outside TPC

Configuration in SBND:

sbndcode/sbndcode/LarSoftConfigurations/opticalsimparameterisations_sbnd.fcl

sbndcode/sbndcode/LarSoftConfigurations/PDFastSim_sbnd.fcl

```
# ******************************************************************************
#          PARAMETERS SETS FOR SEMI-ANALYTIC SIMULATION ARE DEFINED HERE
# ******************************************************************************

# VUV/DIRECT LIGHT: TIMING PARAMETERISATION

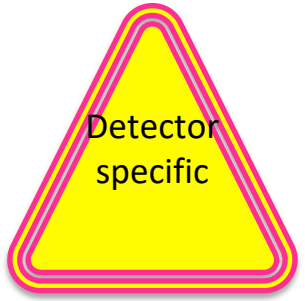
# VUV/DIRECT LIGHT: NUMBER OF HITS CORRECTIONS

# ******************************************************************************
#  VIS semi-analytic model, specific to SBND
# ******************************************************************************

# VISIBLE/REFLECTED LIGHT: TIMING PARAMETERISATION

# VISIBLE/REFLECTED LIGHT: NUMBER OF HITS CORRECTIONS
```

Implementation in DUNE far detectors



A hybrid model is the default approach used in DUNE:

- semi-analytical model (hits + timing) inside TPC
- slimmed-down optical library outside TPC (only available for some geometries)

Implementations also available in some DUNE geometries for GENN and are coming soon for GPU accelerated full optical simulation (Opticks, Celeritas)

Configuration in DUNE:

duneopdet/PhotonPropagation/opticalsimparameterisations_dune.fcl

duneopdet/PhotonPropagation/PDFastSim_dune.fcl

```
# ******************************************************************************
#          PARAMETERS SETS FOR SEMI-ANALYTIC SIMULATION ARE DEFINED HERE
# ******************************************************************************

# VUV/DIRECT LIGHT: TIMING PARAMETERISATION

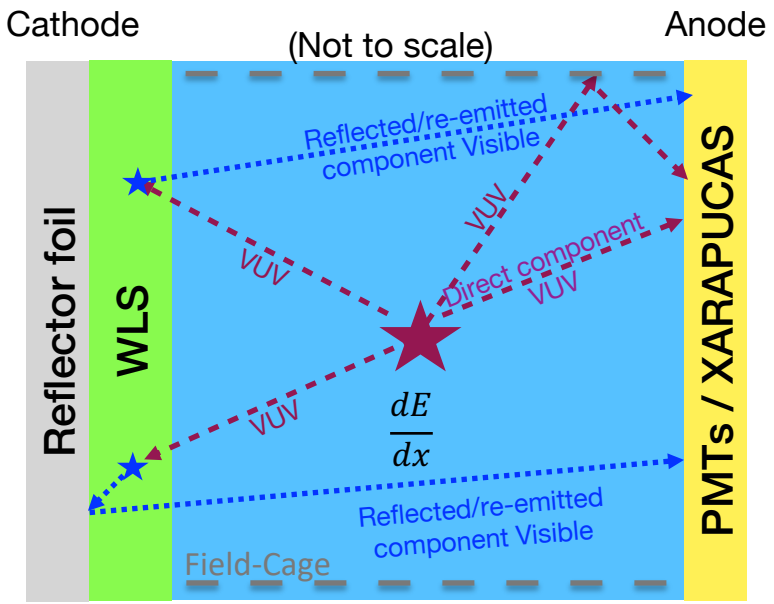
# VUV/DIRECT LIGHT: NUMBER OF HITS CORRECTIONS
```

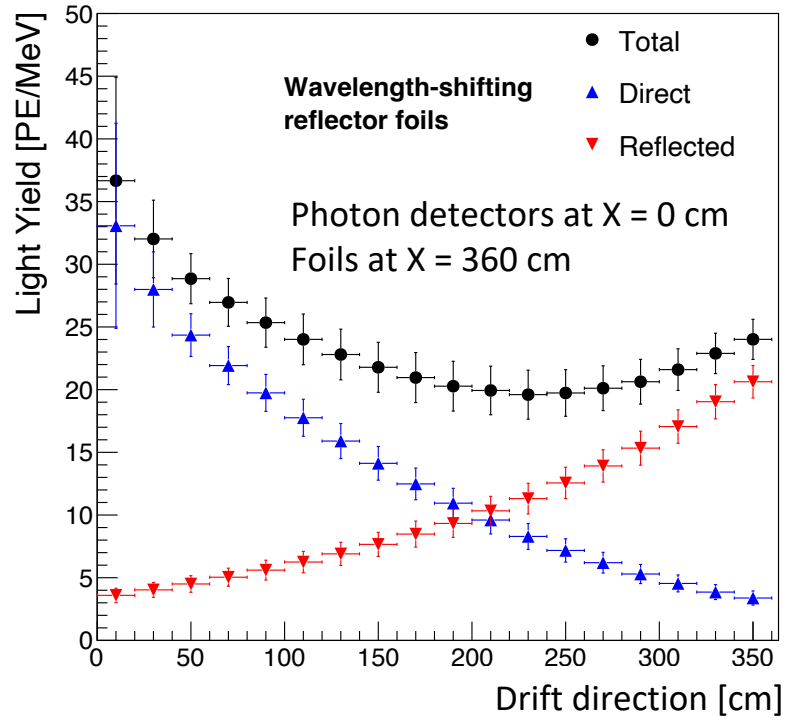
(Aside): Enhancing the Light Yield in LArTPCs

Photon-detectors typically located only on one side of LArTPCs – highly-non uniform light yield

- regions furthest from photon detectors may see insufficient light for low energy triggering, etc.
  –> methods to enhance light yield / improve light yield uniformity

1. WLS-Coated reflector foils (SBND, LArIAT)

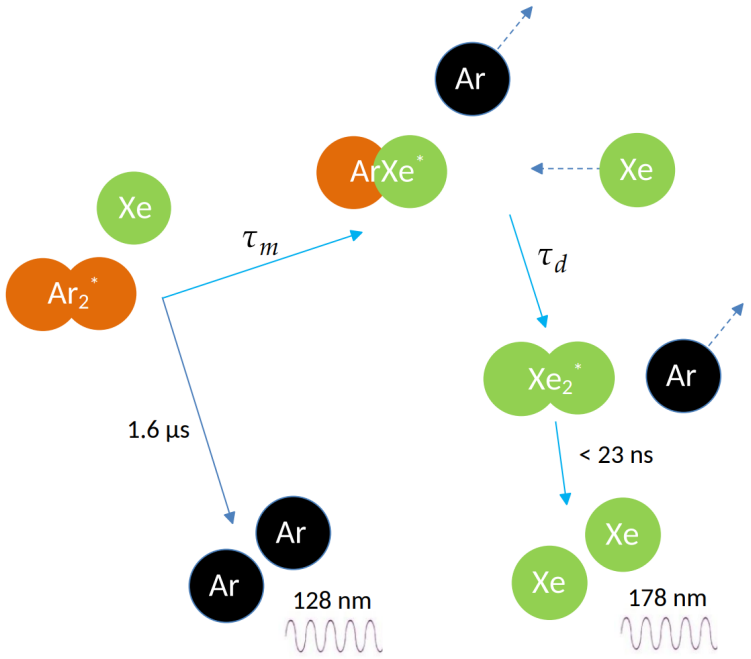


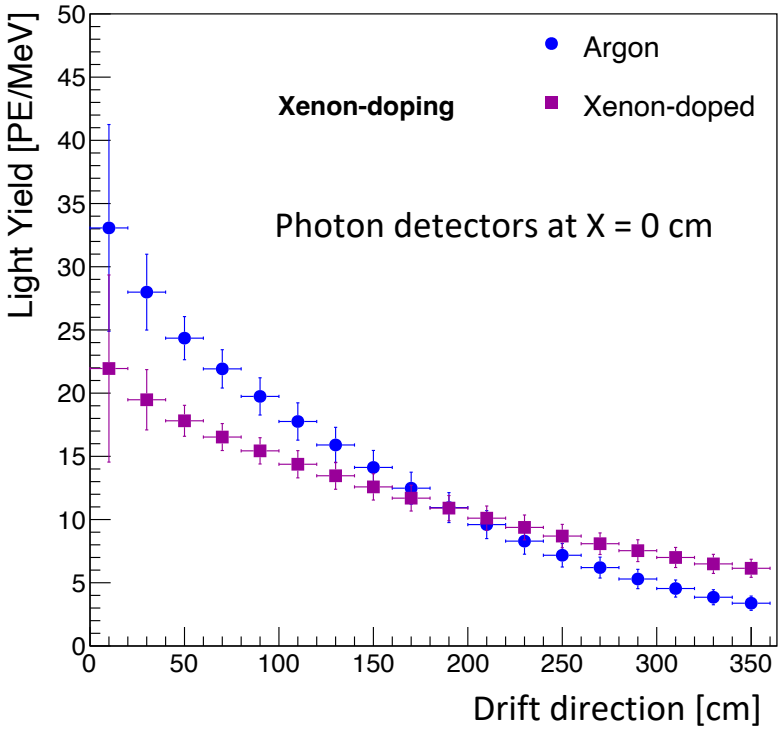


(Aside): Enhancing the Light Yield in LArTPCs

2. Xenon doping (DUNE-VD)

- shifts portion of scintillation photons 128 nm –> 178 nm (Xe excimer emission)
- Xe wavelength photons have much longer $\lambda_{RS} \sim 10$ m



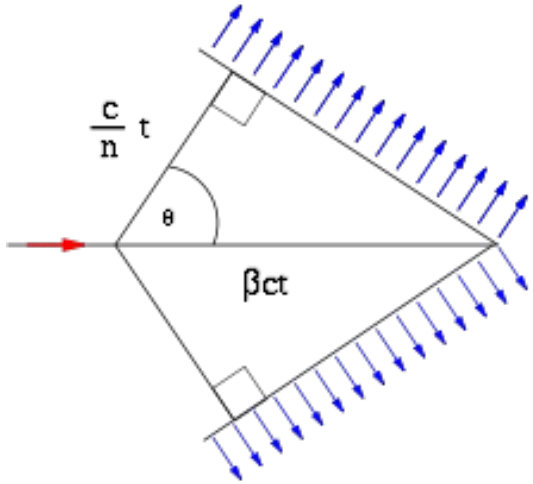


(Aside): Cherenkov radiation in LArTPCs

Particles propagating in LAr also produce Cherenkov light:

- emitted in a cone relative to particle direction
- Cherenkov photon yield approximately 1-2% of scintillation yield [*NIM A* 516 (2004) 348-363]

→ can be considered a second order effect with respect to scintillation emission, and is typically disregarded



Possible* to simulate in LArSoft:

dunesim/dunesim/Simulation/**larg4services_dune.fcl**

sbndcode / sbndcode / LArG4 / **larg4_services_sbnd.fcl**

```
EnableCerenkovLight: false # Cerenkov light OFF by default
```

*Beware: no fast optical simulation exists, will use very slow full simulation!

Detection



Detecting light in LArTPCs

VUV LAr scintillation light is hard to detect, absorbed by most materials

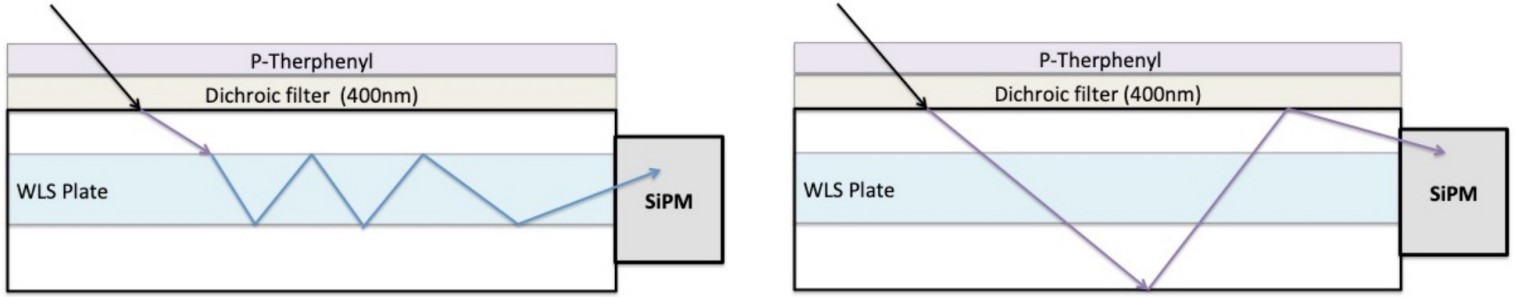
- need to make use of wavelength shifters, e.g. TPB, PEN

Photon detectors:

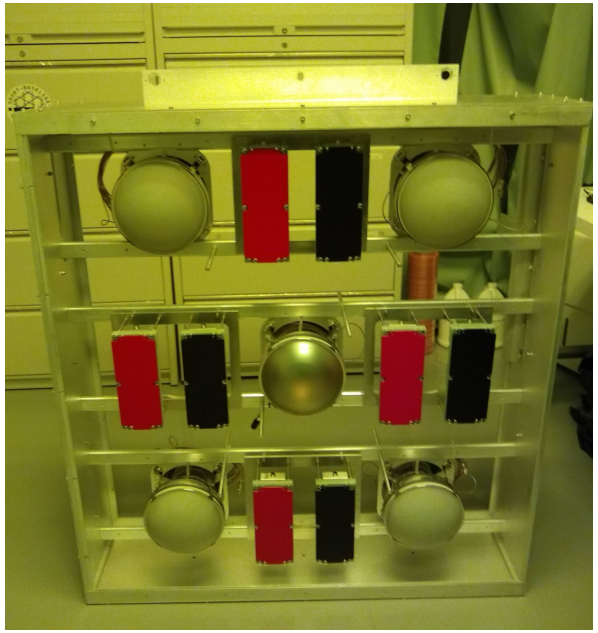
- PMTs coated with WLS (SBND) or with WLS-coated plates (MicroBooNE)
- X-Arapuca wavelength shifting light traps using SiPMs (DUNE, SBND)

Wavelength shifters emit ~isotropically,

–> lose 50% of light emitted away from photon-detectors



X-Arapuca operational principal



Photon detection system module in SBND, mixture of PMTs and X-Arapucas

Wavelength shifter in LArSoft

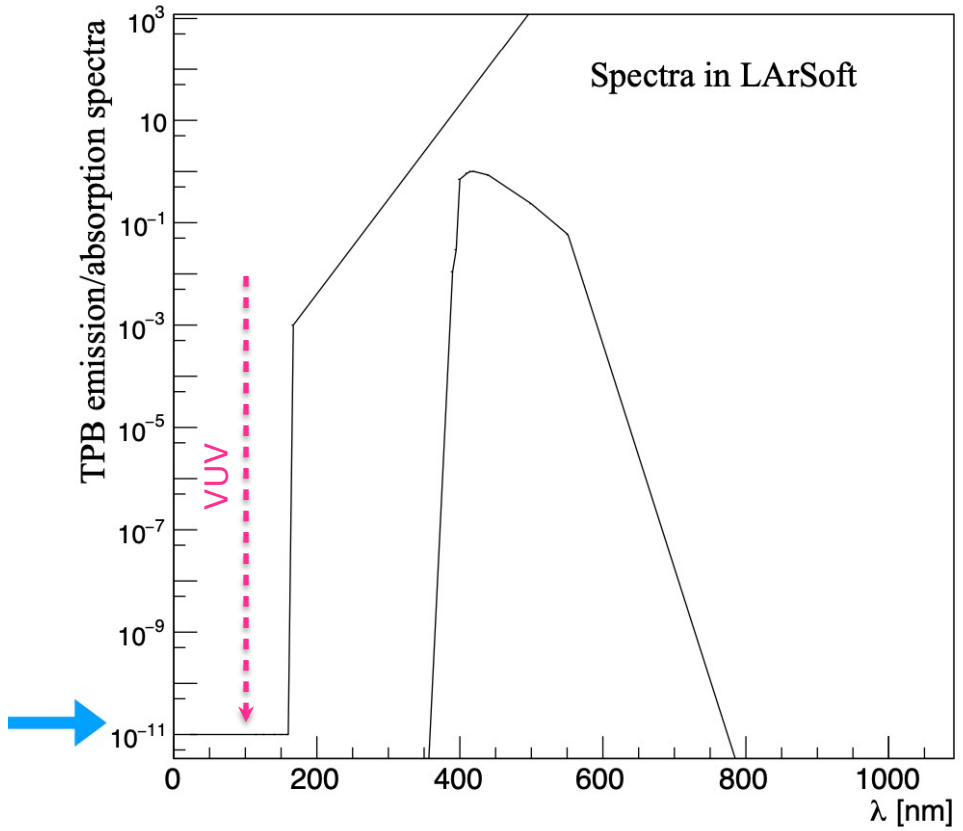
Process handled by G4OpWLS:

- initial photon killed and new one created with a different wavelength
- user defines absorption and emission spectra

In LArSoft parameterized either in larproperies.fcl (legacy) or GDML (new)

lardataalg / lardataalg / DetectorInfo / **larproperties.fcl**

```
# WLS - TPB properties original tpb [0.0, 0.0, 0.0, 0.0588,0.235, 0.853,
TpbEmmisionEnergies: [0.05,1.0,1.5, 2.25, 2.481, 2.819, 2.952,2.988,3.024
TpbEmmisionSpectrum: [0.0, 0.0, 0.0, 0.0588,0.235, 0.853, 1.0,1.0,0.9259,
TpbAbsorptionEnergies: [0.05,1.77,2.0675, 7.42, 7.75, 8.16, 8.73, 9.78,10
TpbAbsorptionSpectrum: [100000.0,100000.0, 100000.0,0.001,0.00000000001,0
```



Wavelength shifter time delay

TPB has complex time structure:

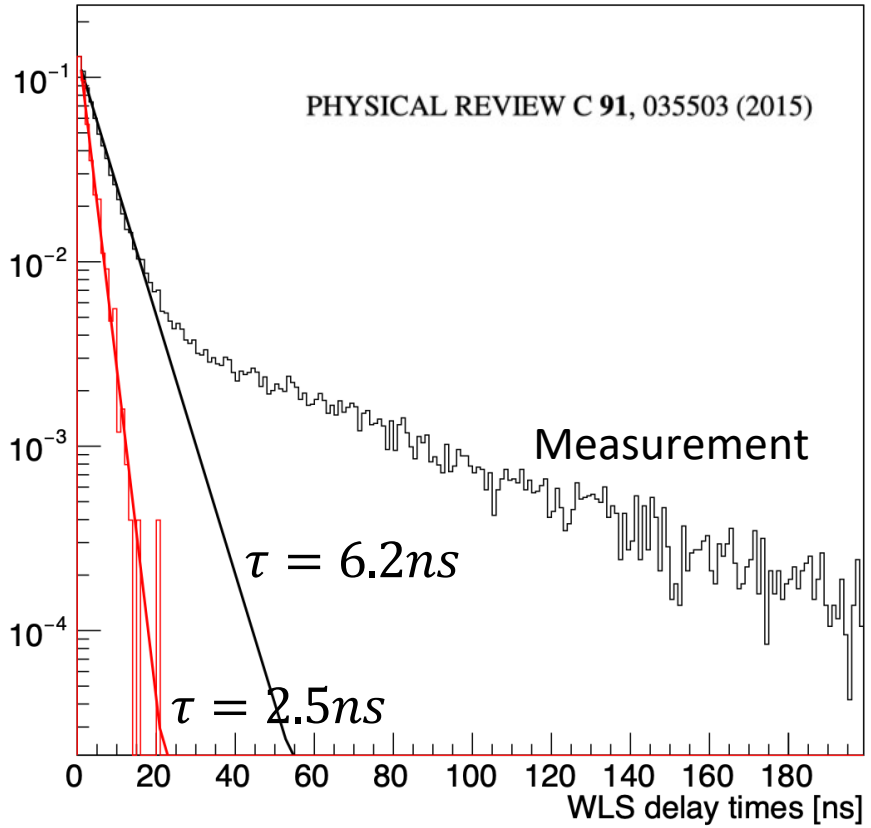
- bulk of light emitted promptly, but non-negligible longer components
- non-negligibly alters observed time distribution

Geant4 (G4OpWLS class) only simulates Delta or Exponential model (neither is the case)

Instead, can simulate this separately in LArSoft:

- tools created for SBND, but not currently done in DUNE
- instead, approximate with longer late light time constant

	Decay time (ns)	Abundance (%)
Instantaneous component	1–10	60 ± 1
Intermediate component	49 ± 1	30 ± 1
Long component	3550 ± 500	8 ± 1
Spurious component	309 ± 10	2 ± 1



Photon simulation output objects

Choice of output objects (data product)

- SimPhotons objects (collections of OnePhoton) save detailed information about each detected photon
- SimPhotonsLite objects reduce size at the price of keeping only the number of photons at a time-step

The kind of object you want to save in your simulation is specified in the configuration file by:

```
services.LArG4Parameters.UseLitePhotons: true  # false to save SimPhotons
```

lardataobj / lardataobj / Simulation / **SimPhotons.h**

```
class SimPhotons : public std::vector<OnePhoton>
```

```
// This structure contains all the information per photon
 // which entered the sensitive OpDet volume.

 class OnePhoton
 {
 public:
   OnePhoton();

   bool           SetInSD;
   TVector3       InitialPosition;
   TVector3       FinalLocalPosition; // in cm
   float          Time;
   float          Energy;
   int            MotherTrackID;
 };
```

```
class SimPhotonsLite
{
  public:
    SimPhotonsLite();
    SimPhotonsLite(int chan)
      : OpChannel(chan)
    {}

    int   OpChannel;
    std::map<int, int> DetectedPhotons;

    SimPhotonsLite& operator+=(const SimPhotonsLite &rhs);
    const SimPhotonsLite operator+(const SimPhotonsLite &rhs) const;

    bool operator==(const SimPhotonsLite &other) const;
};
```

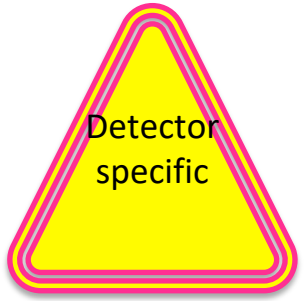
Full optical simulation vs fast optical simulation

Full optical simulation offers full flexibly, but impractical for large scale use

Fast optical simulation inflexible, parameters burned in during training:
–> needs re-training if detector design / understanding of light changes

	Full Optical Simulation	Fast Optical Simulation
Emission Timing Constants	Tunable	Tunable
Emission Energy Spectrum	Tunable	Burned in, impacts transport
Scintillation Yield	Tunable	Tunable
Rayleigh Scattering	Tunable	Burned in
Transport Time Modeling	Not needed	Burned in, but separate
Material Properties	Tunable	Burned in
OnePhoton vs LitePhotons	Chooseable	Chooseable

Signal digitisation



Finally, we simulate the resulting waveforms we'd see on the photo-detectors:

- each incident photon gets swapped for an electronics response, either constructed from parameters (idealised) or from a measured response + add expected noise

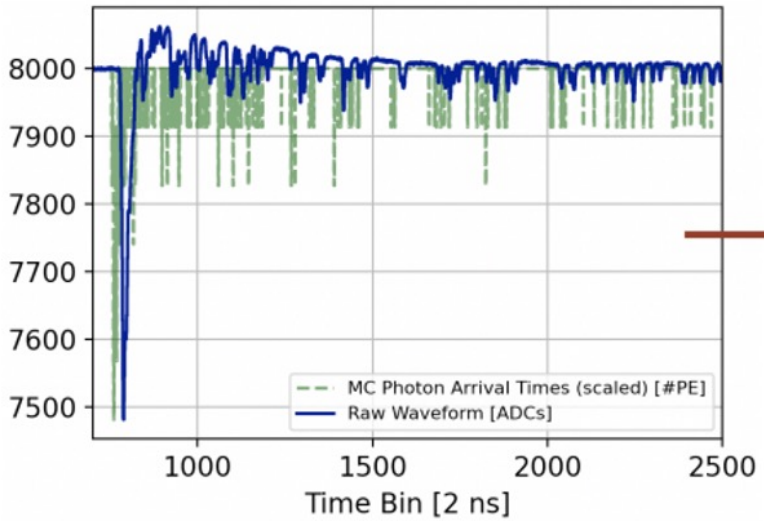
duneopdet / OpticalDetector / **OpDetDigitizerDUNE_module.cc**

sbndcode / OpDetSim / **opDetDigitizerSBND_module.cc**

```
sbnd_digipmt_alg:
{
  # Parameters for ideal SER simulation
  PMTRiseTime:           3.8        #ns
  PMTFallTime:           13.7       #ns
  PMTMeanAmplitude:      0.9        #in pC
  TransitTime:           55.1       #ns
  PMTChargeToADC:        -25.97     #charge to adc factor

  # Parameters for test bench SER simulation
  PMTSinglePEmodel: "data"
  PMTDataFile: "OpDetSim/digi_pmt_sbnd_v2int0.root"  # located in sbnd_data
```

Example PMT waveform in SBND compared with MC photon arrival times



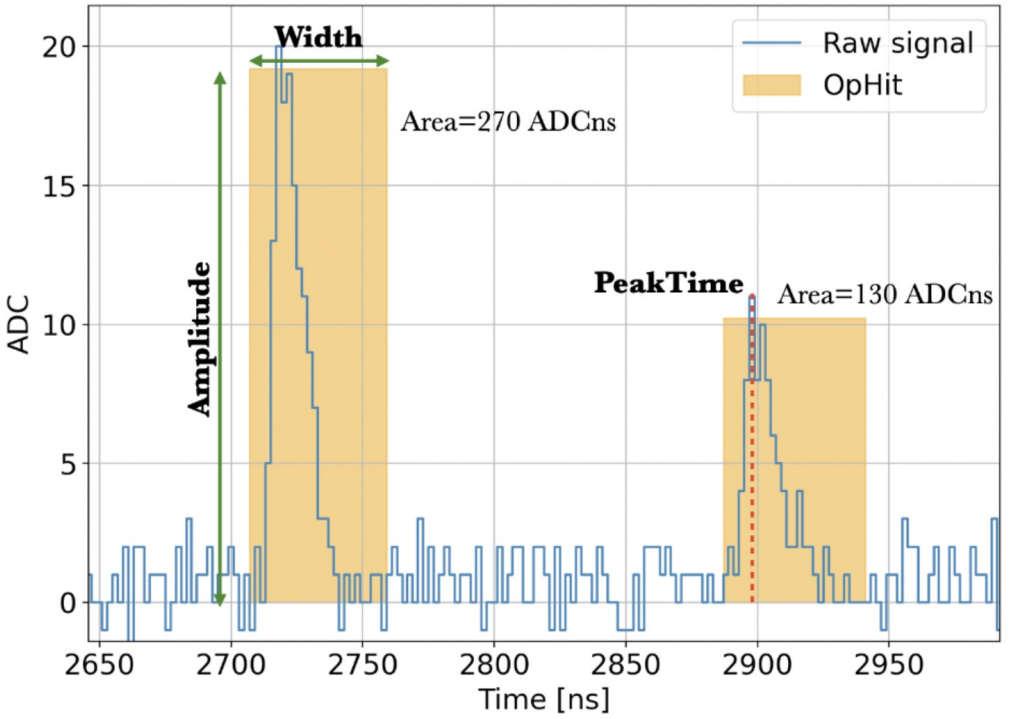
Undershoot due to AC coupling → bipolar single photo-electron response

Reconstruction

Step 1: optical hits

First, we look for pulses in the raw (or deconvolved) waveforms:

- in LArSoft, these pulses are stored in objects called OpHits
- OpHits are found when the waveform goes above a certain threshold and held until it drops back below
- this can lead to the merging of visibly separate optical signals, especially in the case of SiPMs (in the X-Arapucas)
- the OpHit time is decided by the first arriving photon



lardataobj / lardataobj / RecoBase / OpHit.h

```
int                 fOpChannel;
unsigned short      fFrame;
double              fPeakTime;
double              fPeakTimeAbs;
double              fWidth;
double              fArea;
double              fAmplitude;
double              fPE;
```

Step 2: optical flashes

Optical hits from different photon detectors that are time-coincident are combined into Optical Flashes

- these are analogous to clusters in the charge reconstruction, but matched in time rather than space

Having a flash allows us to reconstruct the position of the particles that generated the light (roughly)

This can then be used to match the light signals to the reconstructed TPC tracks: Flash Matching

lardataobj / lardataobj / RecoBase / OpFlash.h

```
private:

      double                fTime { 0.0 }; ///< Time on @ref DetectorClocksHardware
      double                fTimeWidth;    ///< Width of the flash in time [us]
      double                fAbsTime;      ///< Time by PMT readout clock
      unsigned int          fFrame;        ///< Frame number
      std::vector< double > fPEperOpDet;   ///< Number of PE on each PMT
      std::vector< double > fWireCenters;  ///< Geometric center in each view
      std::vector< double > fWireWidths;   ///< Geometric width in each view
      double                fXCenter { NoCenter }; ///< Estimated center in x [cm]
      double                fXWidth { NoCenter }; ///< Estimated width in x [cm]
      double                fYCenter;      ///< Geometric center in y [cm]
      double                fYWidth;       ///< Geometric width in y [cm]
      double                fZCenter;      ///< Geometric center in z [cm]
      double                fZWidth;       ///< Geometric width in z [cm]
      double                fFastToTotal;  ///< Fast to total light ratio
      bool                  fInBeamFrame;  ///< Is this in the beam frame?
      int                   fOnBeamTime;   ///< Is this in time with beam?
```

Step 3: flash matching

We want to match the light (PDS) and charge (TPC) information for the same interaction:

- provides t_0 for each TPC interactions – enables out-of-time backgrounds (e.g. cosmics) to be removed

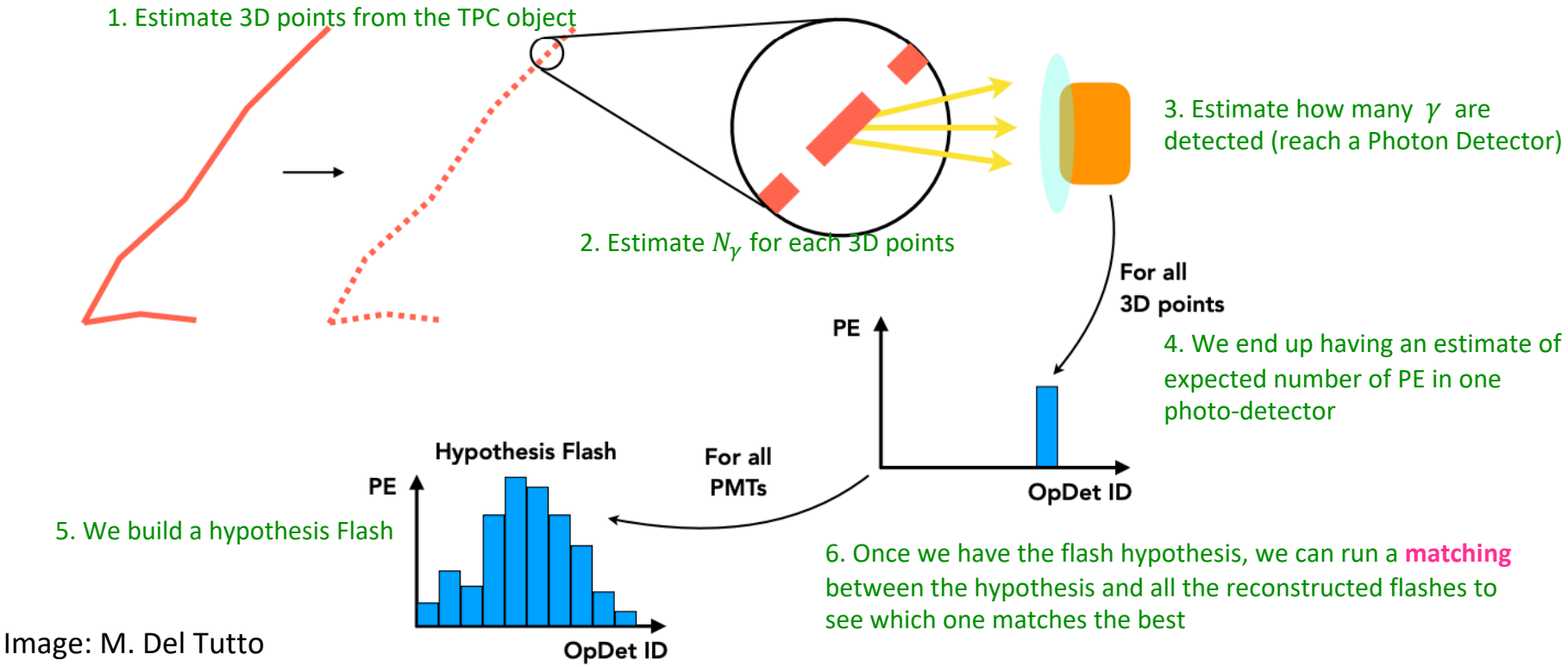


Image: M. Del Tutto

Summary

Simulating scintillation light in LArTPCs is tricky:

- cut some corners to get it working in LArSoft with reasonable resource usage (memory, CPU)
- corners are cut, so always room for improvement – lots of on-going development

Scintillation light historically underutilized in LArTPC neutrino detectors (mostly just for triggering and cosmic rejection); but a lot of very valuable applications, e.g.

- improved calorimetry [PRD 101 012010 (2020), PRD 111 032007 (2025), PRD 112 012019 (2025)]
  – 50% of energy deposited is going into the light
- particle identification and background rejection: pulse shape discrimination (similar to dark matter experiments), nanosecond-scale timing [PRD 108 (2023) 5, 052010]