n-Ar Cross Section @ ProtoDUNE-ND

Nicholas Carrara on behalf of the analysis team

2x2 First Analysis Meeting
July 21, 2023

Two Major Changes for MiniRun5

Neutron Capture Gammas in edep-sim:

- Geant4 has an incorrect gamma cascade for neutron captures on Ar40.
- David Rivera is working on implementing the fix (from Jingo Wang in LArSoft) into edep-sim.

Microphysics Recombination:

- Sam Fogarty and Nick Carrara have been working on integrating LArNEST into larnd-sim/LArSoft.

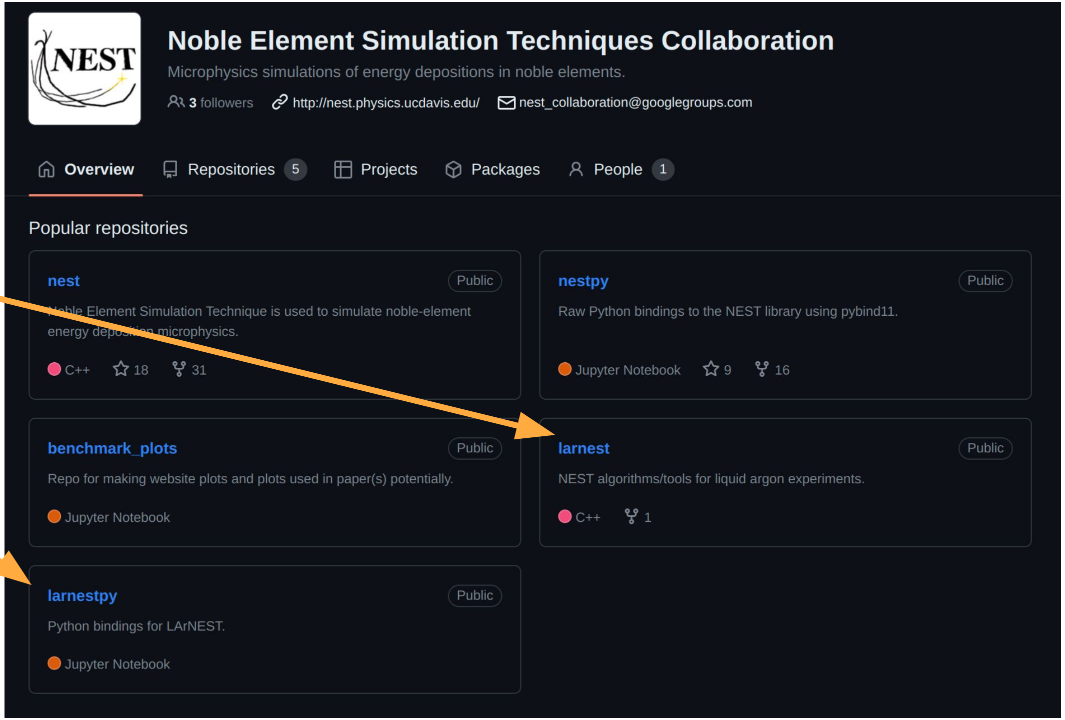
Data fit from various experiments for low energy NR, ER and Alphas

- ARIS
- SCENE
- Joshi
- WARP
- CREUS
- Regenfus
- MicroCLEAN
- Scalettar
- DarkSide
- Kimura
- Bondar
- Doke
- Lippincott
- Sangiorgio

LArNEST

Two repos for LArNEST,

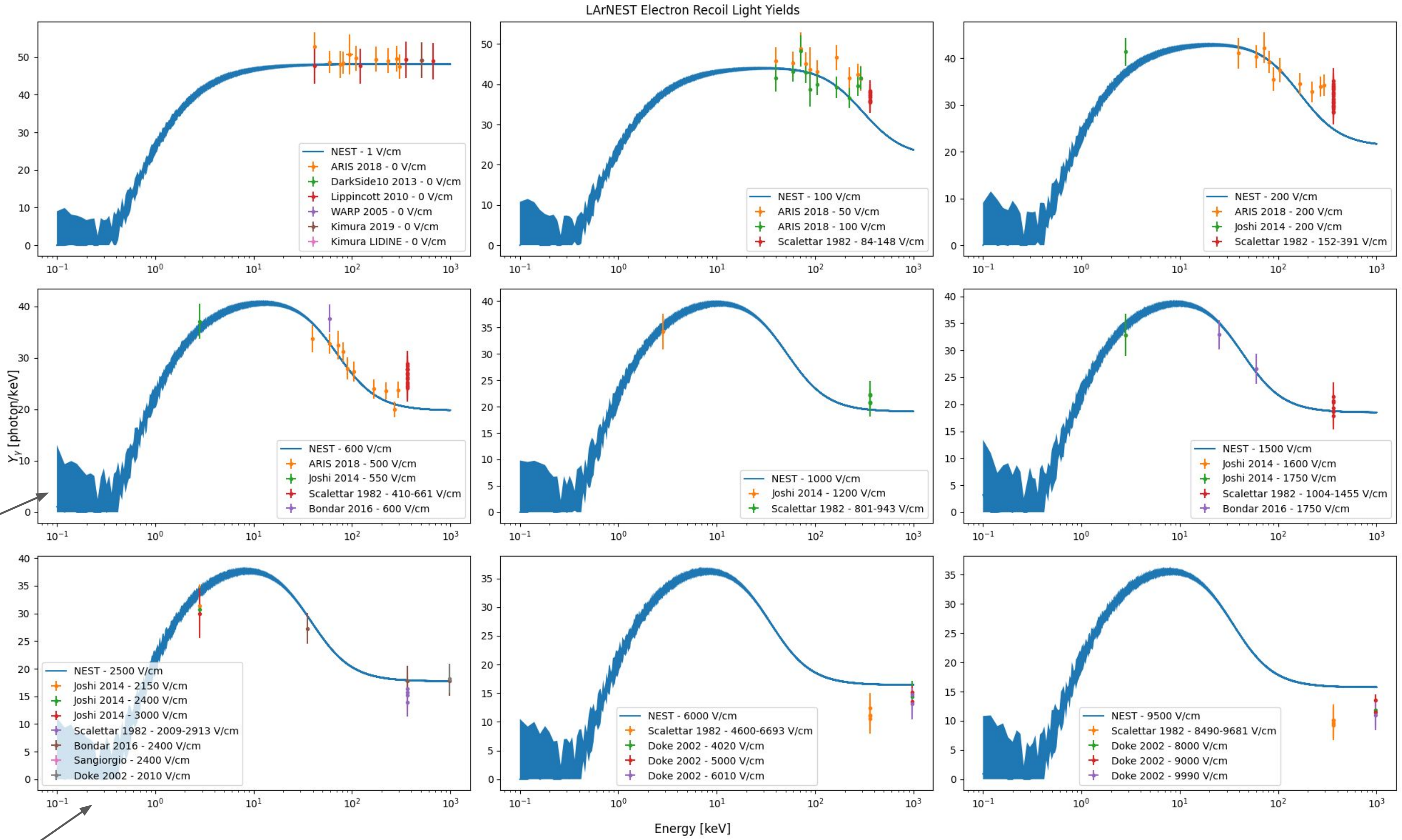
- c++ version for LArSoft (https://github.com/NESTCollaboration/larnest),
- python bindings for larnd-sim (https://github.com/NESTCollaboration/larnestpy)



LArNEST

Electron Recoil model is fit to the available data using functions from first principles (**work done by Justin Mueller and Ekaterina Kozlova**).

Fluctuations are Poisson/Fano.



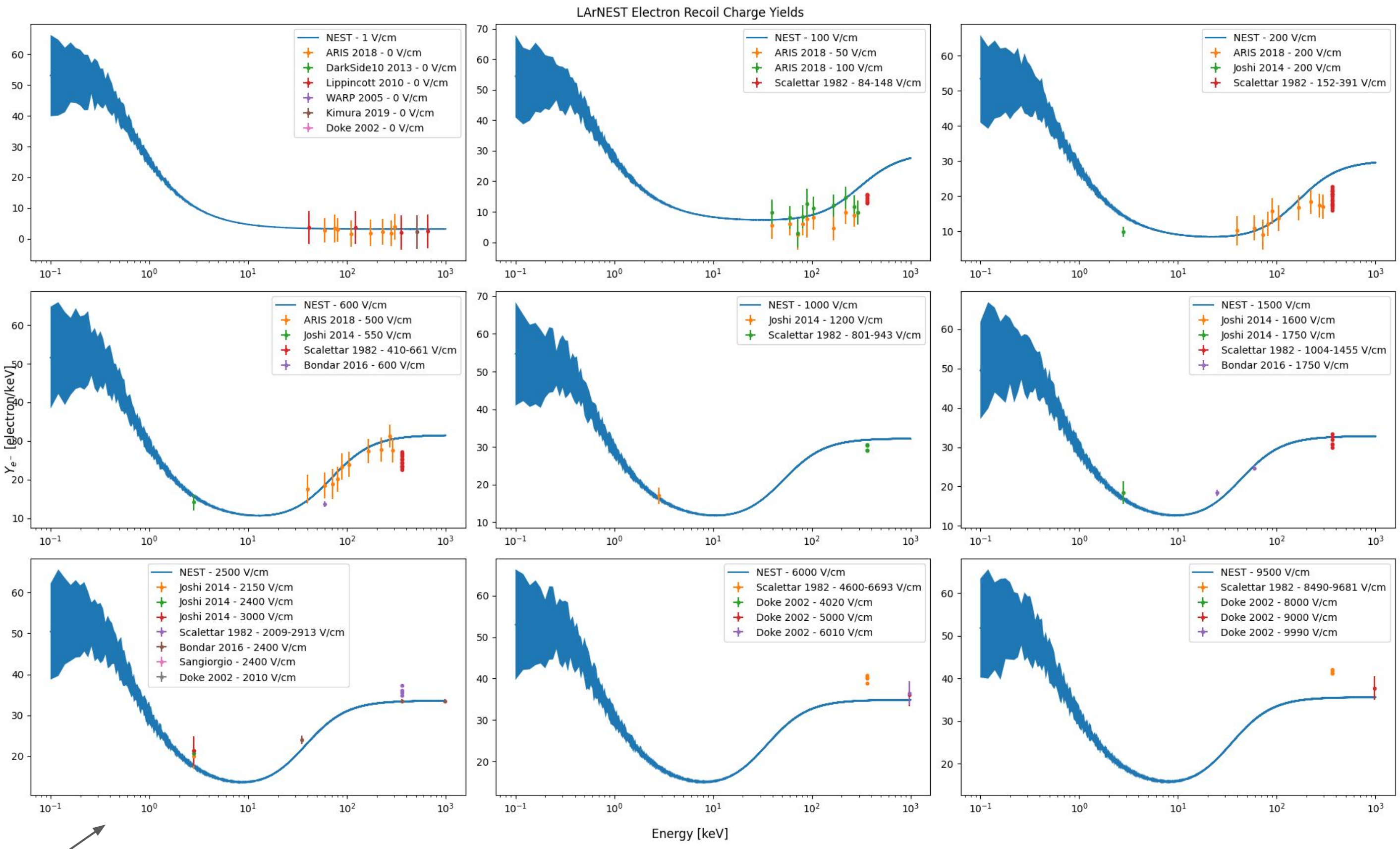
Electron Recoil Light Yields [photons/keV] for different electric field values [V/cm]

LArNEST

The function for electron yields takes (**energy, electric field, density**) as inputs, and is fit with **31** parameters.

It is adapted from the beta model for liquid Xenon in NEST.

Light yields are assumed anti-correlated for electronic recoils.



Electron Recoil Quanta Yields [electrons/keV] for different electric field values [V/cm]

LArNEST

Parameter	LXe	LAr
A_α	32.988	32.988
B_α	−32.988	−552.988
C_α	1	17.2346
D_α	0	−4.7
E_α	0.026715	0.025115
F_α	−0.33926	0.26536
G_α	0.6705	0.242671
A_β	1	0.778482
B_β	0.4607	25.9
C_β	1	1.105
D_β	621.74	0.4
E_β	−2.2717	4.55
F_β	53.502	−7.502
A_γ	1	0.659509
B_γ	1000	1000
C_γ	6.5	6.5
D_γ	1	5
E_γ	−11	−0.5
F_γ	47.408	1047.408
G_γ	1.9851	0.01851
A_ϵ	1652.264	1052.264
B_ϵ	1.415935e10 − 1652.264	1.415935e10 − 1652.264
C_ϵ	1	−5
D_ϵ	0.02673144	0.157933
E_ϵ	1.564691	1.83894
p_1	1	1
p_2	1.304	10.304
p_3	2.1393	13.0654
p_4	0.35535	0.10535
p_5	1	0.7
δ	28	15.7489
ℓ	−2	−2.07763

$$Y_e(E|\alpha,\beta,\gamma,\delta,\epsilon,p_k,\ell) = \alpha\beta + \frac{\gamma - \alpha\beta}{(p_1 + p_2(E+0.5)^{p_3})^{p_4}} + \frac{\delta}{p_5 + \epsilon E^{\ell}}, \quad (61)$$

where the p_k, δ and ℓ are constants[3], α is a function of the electric field and the density,

$$\alpha(\vec{E},\rho_Z) = A_\alpha + B_\alpha\left(C_\alpha + \left(\frac{|\vec{E}|}{D_\alpha + E_\alpha \exp\left[\frac{\rho_Z}{F_\alpha}\right]}\right)^{G_\alpha}\right)^{-1}, \quad (62)$$

while β is a function of only the electric field:

$$\beta(\vec{E}) = A_\beta + B_\beta\left(C_\beta + \left(\frac{|\vec{E}|}{D_\beta}\right)^{E_\beta}\right)^{F_\beta}. \quad (63)$$

The γ function depends on the electric field and the work function of LAr:

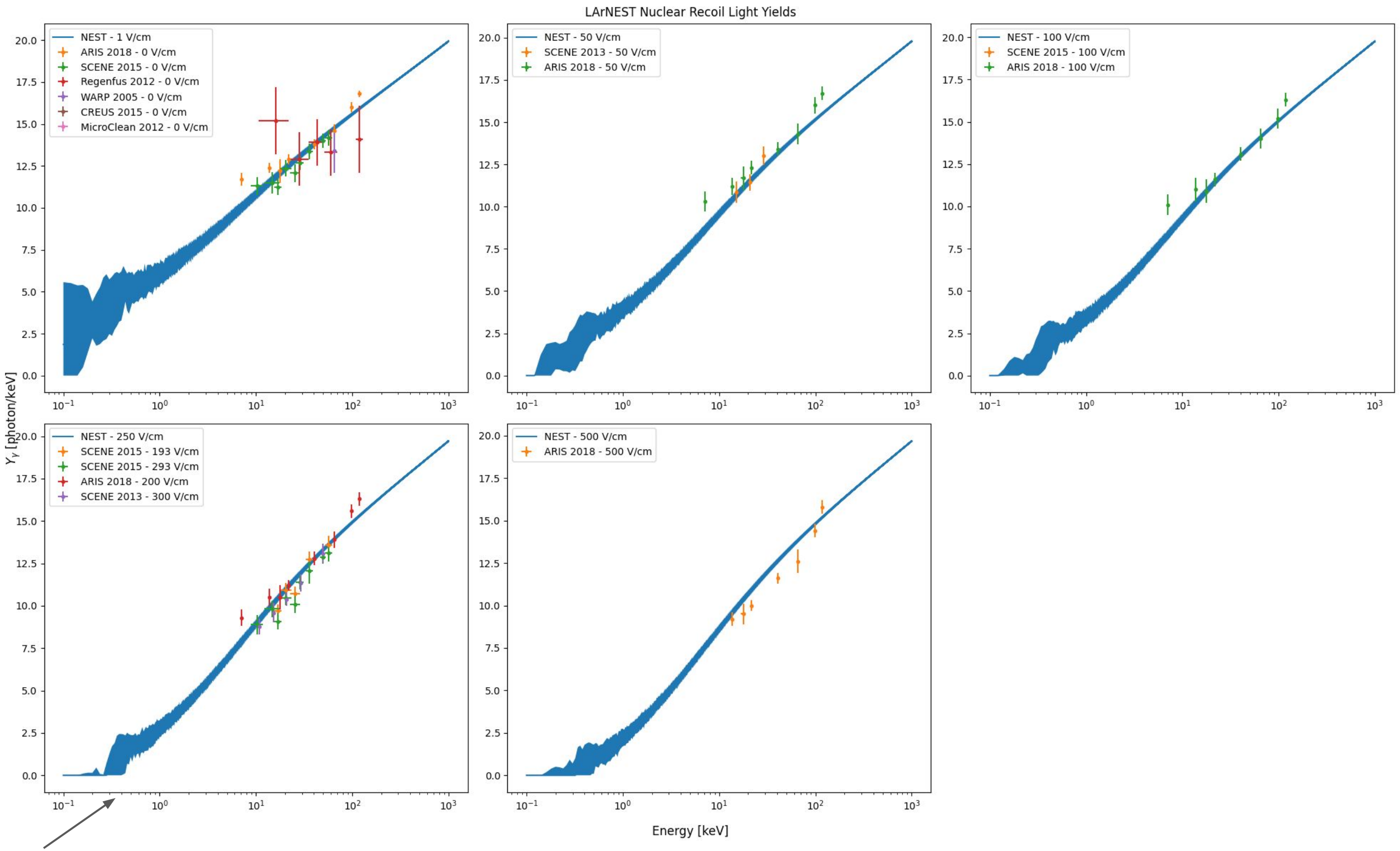
$$\gamma(\vec{E},W) = A_\gamma\left(\frac{B_\gamma}{W} + C_\gamma\left(D_\gamma + \frac{E_\gamma}{\left(\frac{|\vec{E}|}{F_\gamma}\right)^{G_\gamma}}\right)\right). \quad (64)$$

The ϵ function is the Doke-Birks function for LAr which is given by:

$$\epsilon(\vec{E}) = A_\epsilon + \frac{B_\epsilon}{\left(C_\epsilon + \left(\frac{|\vec{E}|}{D_\epsilon}\right)^{E_\epsilon}\right)}. \quad (65)$$

LArNEST

Similarly for nuclear recoils, a model for light yields and electron yields are fit according to available data, **but with a slight breaking of the anti-correlation**.



Nuclear Recoil Light Yields [photons/keV] for different electric field values [V/cm]

LArNEST

Nuclear Recoil Total Yields [quanta/keV] for different electric field values [V/cm]

The equation describing the total yield used in this study is given by a simple power law,

$$Y_q(E, \vec{E}|\alpha, \beta) = Y_q(E|\alpha, \beta) = \alpha E^{\beta}, \quad (57)$$

where α is a positive scalar with units $[\alpha] = (\mathrm{keV}^{\beta})^{-1}$ and β is a dimensionless real number.

Parameter	LXe	LAr
α	$11^{+2.0}_{-0.5}$	11.10 ± 1.4
β	1.1 ± 0.05	1.087 ± 0.01

$$\tilde{Y}_\gamma(E, \vec{E}|\alpha, \beta, \gamma, \delta, \epsilon) = \alpha E^{\beta-1} - \left(\frac{1}{\gamma|\vec{E}|^{\delta}}\right)\left(\frac{1}{\sqrt{E+\epsilon}}\right)$$
$$= Y_q - Y_e \left(1 - \frac{1}{1 + \left(\frac{E}{\zeta}\right)^{\eta}}\right)^{-1}. \quad (59)$$

Parameter	LXe	LAr
γ	0.0480 ± 0.0021	0.1 ± 0.005
δ	-0.0533 ± 0.0068	-0.0932 ± 0.0095
ϵ	$12.6^{+3.4}_{-2.9}$	2.998 ± 1.026
ζ	0.3 ± 0.1	0.3 (Fixed)
η	2 ± 1	2.94 ± 0.12

Slight breaking of a strict anti-correlation

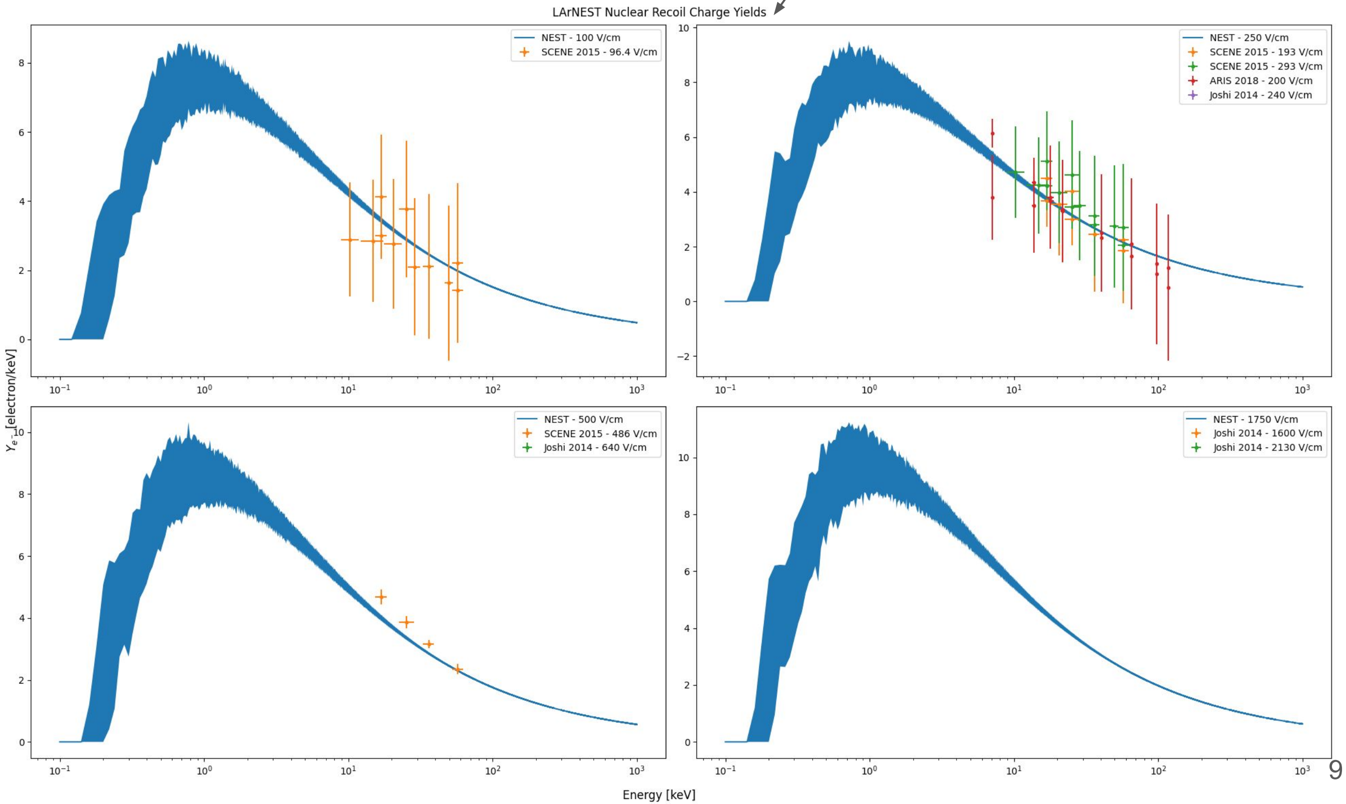
Nuclear Recoil Light Yields [photons/keV] for different electric field values [V/cm]

LArNEST

The equation describing the exciton (or charge) yield is given below:

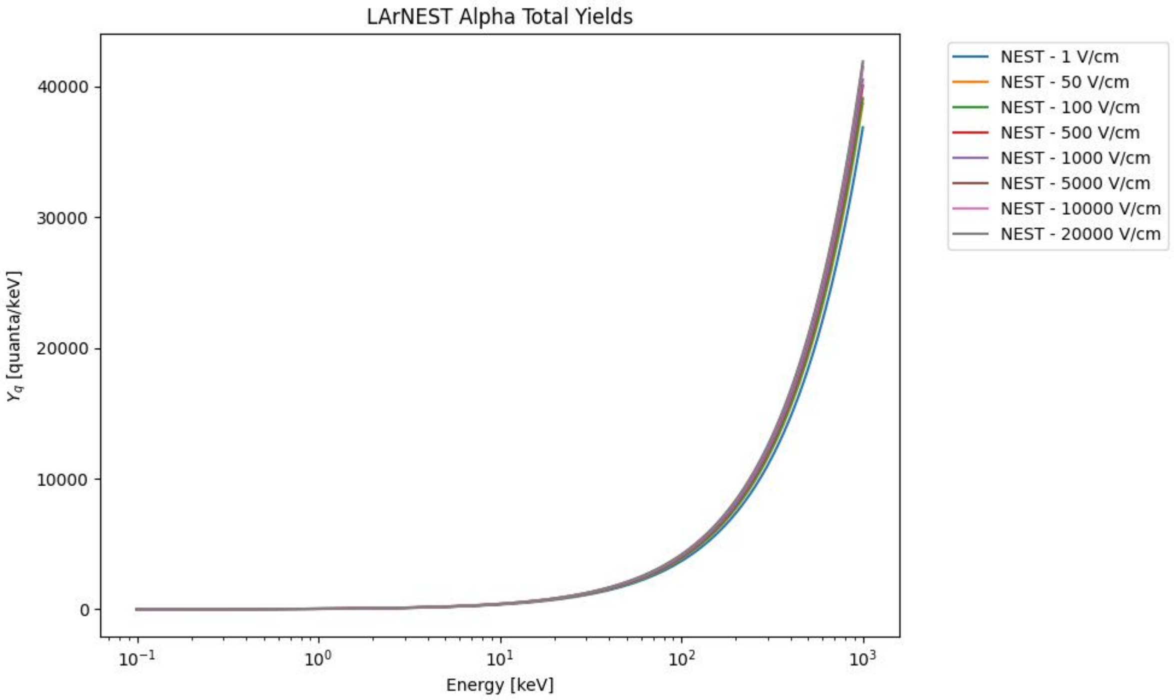
$$Y_e(E, \vec{E}|\gamma, \epsilon, \delta, \zeta, \eta) = \left(\frac{1}{\gamma|\vec{E}|^{\delta}}\right)\left(\frac{1}{\sqrt{E+\epsilon}}\right)\left(1 - \frac{1}{1+\left(\frac{E}{\zeta}\right)^{\eta}}\right), \qquad (58)$$

Nuclear Recoil Electron Yields [electrons/keV] for different electric field values [V/cm]



LArNEST

We use a slightly different model for alphas (**of which there is very little data!**).



LArNEST

Alpha Light/Electron Yields [(photons/electrons)/keV]

The equation describing the electron yield is,

$$Y_e(\vec{E}) = A\left(B - \left(BC + \frac{B}{D}\left[1 - \frac{E\log\left(1 + \frac{B}{D}F^{\frac{(G+|\vec{E}|^{H})^{I}}{J}}\right)}{\frac{B}{D}F(G+|\vec{E}|^{H})^{I}}\right]\right)\right), \quad (66)$$

where $A, B, C, D, E, F, G, H, I$ and J are scalar parameters. The fits for these values are given below:

Parameter	LAr
A	1.0/6200.0
B	64478398.7663
C	0.173553719
D	1.21
E	0.028752
F	0.01
G	4.71598
H	7.72848
I	−0.109802
J	3.0

$$Y_\gamma(\vec{E}) = \left(\frac{1}{A|\vec{E}|^{B}}\right)C\left(DE + \frac{D}{F}\left[1 - \frac{G\log\left(1 + \frac{D}{F}H^{\frac{(I+(\frac{|\vec{E}|}{J})^{K})^{L}}{M}}\right)}{\frac{D}{F}H(I + \left(\frac{|\vec{E}|}{J}\right)^{K})^{L}}\right]\right), \quad (67)$$

where $A, B, C, D, E, F, G, H, I, J, K, L$ and M are scalar parameters. The fits for these values are given below:

Parameter	LAr
A	1.5
B	−0.012
C	1.0/6500.0
D	278037.250283
E	0.173553719
F	1.21
G	2.0
H	0.653503
I	4.98483
J	10.0822
K	1.2076
L	−0.97977
M	3.0

LArNEST

LArNEST currently has eight options for calculating yields/fluctuations. The first three (**NR, ER, Alpha**) are described in the previous slides.

The **LeptonLET, LET and Legacy** versions were used in the early days of LBNE (M. Szydagis).

The **BOX and BIRKS** models are taken from the current larnd-sim quenching.

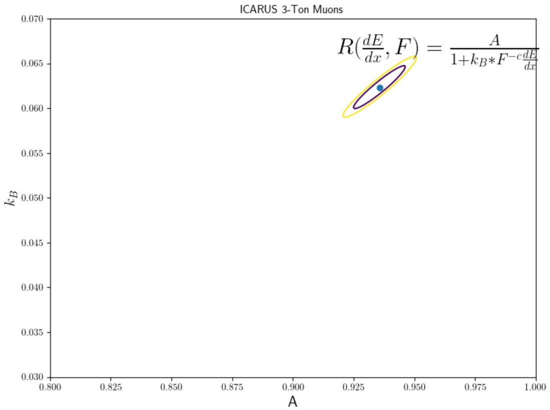
```
enum class LArInteraction
{
    NR = 0,
    ER = 1,
    Alpha = 2,
    dEdx = 3,
    LeptonLET = 4,
    LET = 5,
    BOX = 6,
    BIRKS = 7,
    Legacy = 8
};
```

```
You, 3 weeks ago | 1 author (You)
struct LArNRYieldsParameters
{
    double alpha = {11.10};
    double beta = {0.087};
    double gamma = {0.1};
    double delta = {-0.0932};
    double epsilon = {2.998};
    double zeta = {0.3};
    double eta = {2.94};
};
You, 3 weeks ago | 1 author (You)
struct LArERElectronYieldsAlphaParameters
{
    double A = {32.988};
    double B = {-552.988};
    double C = {17.2346};
    double D = {-4.7};
    double E = {0.025115};
    double F = {0.265360653};
    double G = {0.242671};
};
```

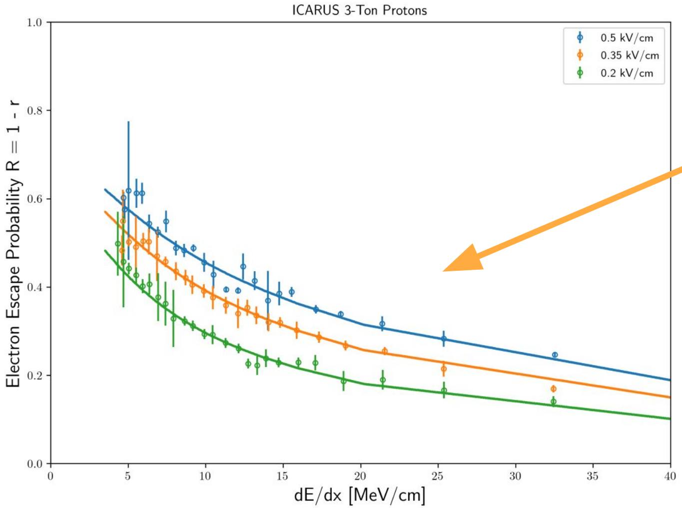
The several dozen parameters are all tunable at run time.

LArNEST

LArdEdxParameters were fit by **Justin Mueller** for a Birks type model on ICARUS/DM data.



Prefers c = 0.90



Best Result

Prefers c = 0.85

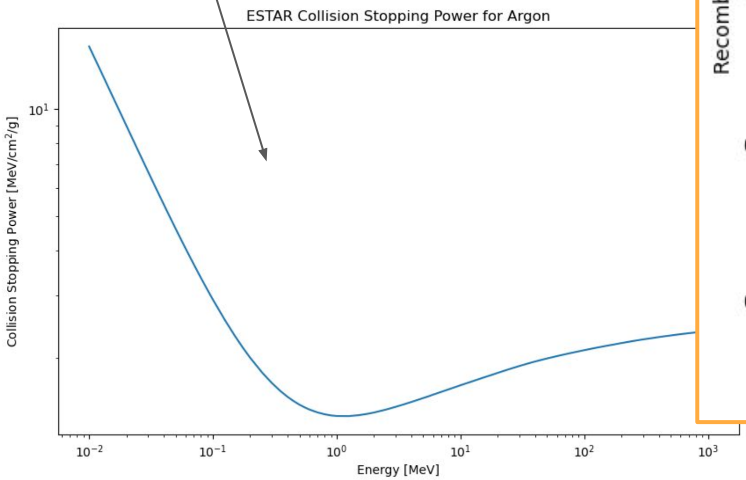
```
You, 33 minutes ago | 1 author (You)
struct LArdEdxParameters
{
    double A = {0.87};
    double kb = {0.045};
    double c = {0.9};
};

You, 3 weeks ago | 1 author (You)
struct ThomasImelParameters
{
    double A = {0.1};
    double B = {-0.0932};
};

You, 3 days ago | 1 author (You)
struct BOXParameters
{
    double alpha = {0.93};
    double beta = {0.207};
};

You, 3 days ago | 1 author (You)
struct BIRKSParameters
{
    double Ab = {0.800};
    double kb = {0.0486};
};
```

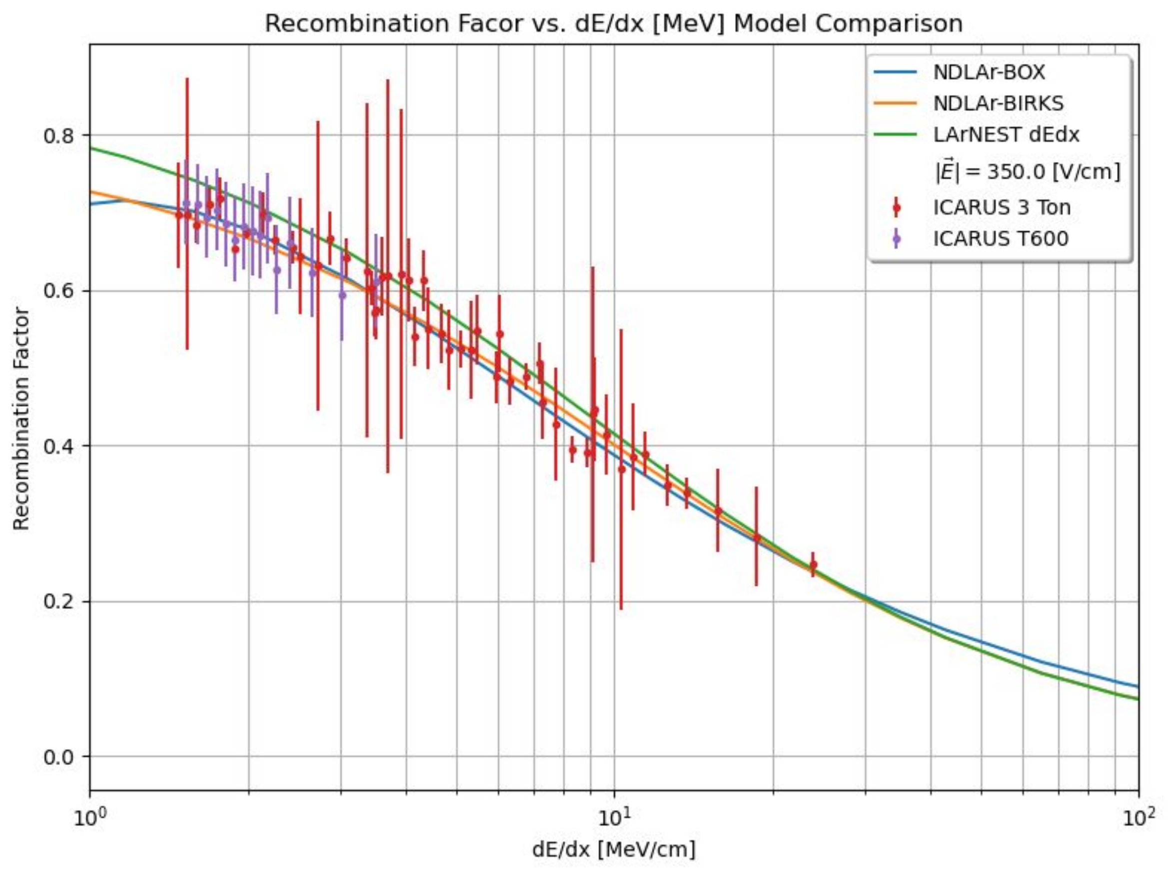
BOX and BIRKS correspond to larnd-sim models.

LArNEST Model Comparison

Comparison of larnd-sim BOX/BIRKS models with Justins dEdx model against ICARUS data.

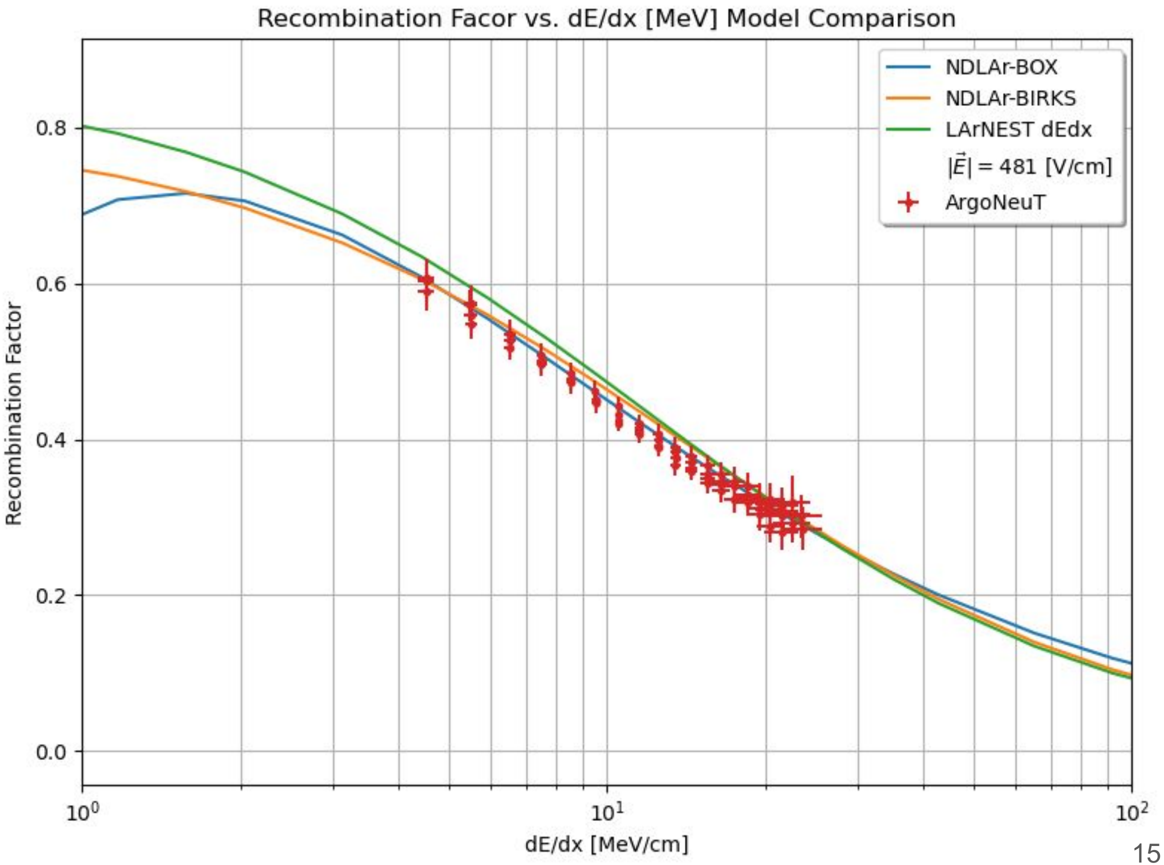
Using the associated scattering length from ESTAR stopping power of electrons in Argon.





LArNEST Model Comparison

ArgoNeuT parameters need to be tuned slightly for the dEdx model.



LArNEST **implementation** - progress & validations
Current Recombination Models in larnd-sim

- Currently larnd-sim only supports using a single recombination model at a time, and that is either the **Box model** (Baller, 2013 JINST 8 P08005) or the **Birks model** (Amoruso, et al NIM A 523 (2004) 275)
- It is of interest to allow larnd-sim to use models from LArNEST, namely the **ER model** for low-energy electrons, the **alpha model** for alphas, and the **NR model** for nuclear recoils

Currently in quenching.py:

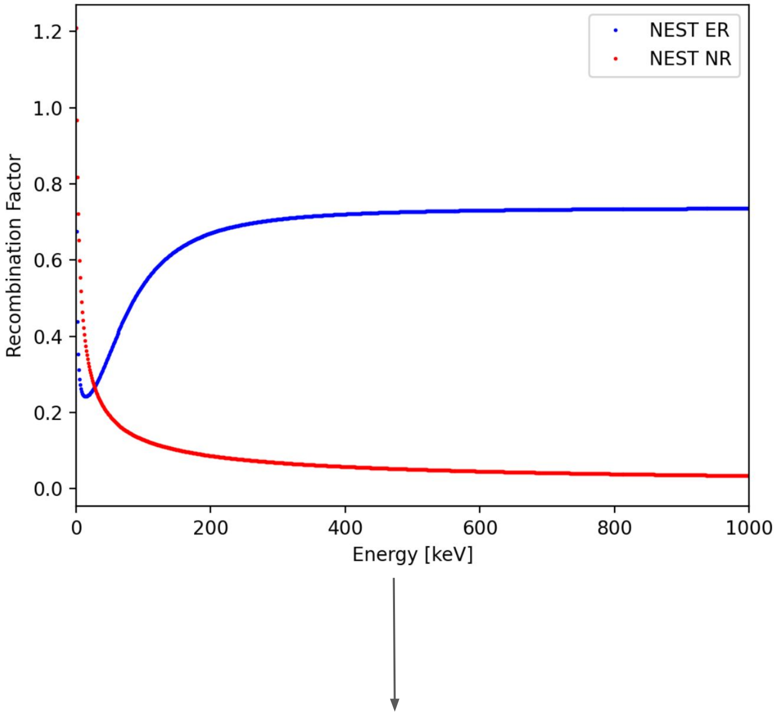
```
if mode == physics.BOX:
    # Baller, 2013 JINST 8 P08005
    csi = physics.BOX_BETA * dEdx / (detector.E_FIELD * detector.LAR_DENSITY)
    recomb = max(0, log(physics.BOX_ALPHA + csi)/csi)
elif mode == physics.BIRKS:
    # Amoruso, et al NIM A 523 (2004) 275
    recomb = physics.BIRKS_Ab / (1 + physics.BIRKS_kb * dEdx / (detector.E_FIELD * detector.LAR_DENSITY))
```

Implementing LArNEST into larnd-sim

- Implementing LArNEST into larnd-sim is complicated by the fact that larnd-sim requires many of its functions to work with the CUDA library (allowing the program to use GPUs to speed up computations).
  - The quench function, which is where recombination factors are calculated, uses CUDA. So LArNEST would need to play nice with CUDA.
  - Many python functions do not work with CUDA (such as ones in larnestpy)
  - LArNEST is written in C++, so it is not clear how to make it compatible with CUDA
- Potential solution: Rewrite LArNEST in pure python so that it works with CUDA
  - Would obviously get around the CUDA compatibility issues
  - Would be a challenge, as many python functions do not work with CUDA
  - This is a path that we can go down if we want to, but it will obviously take some time to get working

Implementing LArNEST into larnd-sim (alternative option)

- An alternative option to using LArNEST directly in larnd-sim is to `cache` the LArNEST data for various energies:
  - Run LArNEST for each model (ER, alpha, NR) for the range of potential energies
    just once prior to running larnd-sim
  - Save the results to a file (energies and recombination factors for each model), then load the data into larnd-sim
  - Interpolate the data when calculating recombination factors
- Benefits to this alternative option:
  - We do not need to add larnestpy as a dependency to larnd-sim, and we wouldn't need to run larnestpy for every simulation
  - We get around CUDA compatibility issues
- This implementation has been made and is being tested: https://github.com/sam-fogarty/larnd-sim/tree/feature_cached_LArNEST



Store in npz file

‘Cached-LArNEST’ Implementation

Added to simulation properties file:

```
pdg_to_recombination_model: # Box 1; Birks 2; NEST_ER 3; NEST_ALPHA 4; NEST_NR 5
    11: 3 # electron
    13: 2 # muon
    2212: 2 # proton
    321: 2 # kaon
    211: 2 # pion
    1000020040: 4 # alpha
    1000180400: 5 # 40Ar
    1000010020: 2 # deuteron
er_energy_threshold: 1.0 # MeV
default_recombination_model: 2 # 1 or 2
```

Dictionary to specify which recombination model to use for which particle

Energy threshold for NEST ER model

Default model to use if a simulated particle isn’t in this dictionary

‘Cached-LArNEST’ Implementation

- Added logic to pick recombination model for each segment
- Includes option to bypass all logic and just use default model
- If no particular model is picked, the default model is used (Box or Birks)

```python
@njit
def pick_model(model, E, dEdx, er_energy_threshold, default_model, use_default_model,\
               E_ER, E_NR, R_ER, R_NR):
    """
    Function to pick a recombination model for a particular segment.

    Args:
        model (int): recombination model number, as defined in consts.physics.
        E (float): energy in MeV, either corresponding to particle starting energy or segment dE
        dEdx (float): segment dE/dx in MeV/cm
        er_energy_threshold (float): threshold energy in MeV for using NEST ER model
            (NEST ER is used if particle energy is less than this threshold). Only relevant for electrons. but must still be specified.
        default_model (int): number corresponding to the model that should be used
            if the segment pdgID is not in the pdg->model dictionary in the simulation properties file.
        use_default_model (bool): if True, bypasses the if statements and always uses the default model.

    """
    recomb = 0
    if use_default_model:
        recomb = DEFAULT_MODEL(default_model, dEdx)
    elif model == physics.BOX:
        recomb = BOX(dEdx)
    elif model == physics.BIRKS:
        recomb = BIRKS(dEdx)
    elif model == physics.NEST_ER and E < er_energy_threshold:
        recomb = NEST_ER(E, E_ER, R_ER)
    elif model == physics.NEST_ALPHA:
        recomb = NEST_ALPHA(E)
    elif model == physics.NEST_NR:
        recomb = NEST_NR(E, E_NR, R_NR)
    else:
        recomb = DEFAULT_MODEL(default_model, dEdx)

    return recomb
```

'Cached-LArNEST' Implementation

Box and Birks models are unchanged, just put into their own functions

```
@njit
def BOX(dEdx):
    """
    Box recombination model. Baller, 2013 JINST 8 P08005

    Args:
        dEdx (float): Segment dE/dx in MeV/cm
    """
    csi = physics.BOX_BETA * dEdx / (detector.E_FIELD * detector.LAR_DENSITY)
    return max(0, log(physics.BOX_ALPHA + csi)/csi)

@njit
def BIRKS(dEdx):
    """
    Birks recombination model. Amoruso, et al NIM A 523 (2004) 275

    Args:
        dEdx (float): Segment dE/dx in MeV/cm
    """
    return physics.BIRKS_Ab / (1 + physics.BIRKS_kb * dEdx / (detector.E_FIELD * detector.LAR_DENSITY))
```

‘Cached-LArNEST’ Implementation

LArNEST ER and NR functions linearly interpolate the cached data for a particular energy

```
@njit
def NEST_ER(E, er_energies, er_recomb_factors):
    """
    LArNEST electron recoil (ER) recombination model used for low-energy electrons.
https://github.com/NESTCollaboration/larnestpy

    Args:
        E (float): Starting energy in MeV of the trajectory corresponding to the current segment.
        er_energies (:obj:`numpy.ndarray`): ER energies from LArNEST.
        er_recomb_factors (:obj:`numpy.ndarray`): ER recombination factors from LArNEST.
    """
    recomb = linear_interpolation(E, er_energies, er_recomb_factors, physics.ER_ASYMPTOTE_AVG)
    return recomb

@njit
def NEST_NR(E, nr_energies, nr_recomb_factors):
    """
    LArNEST nuclear recoil (NR) recombination model. https://github.com/NESTCollaboration/larnestpy

    Args:
        E (float): Segment dE in MeV.
    """
    recomb = linear_interpolation(E, nr_energies, nr_recomb_factors, physics.NR_ASYMPTOTE_AVG)
    return recomb
```

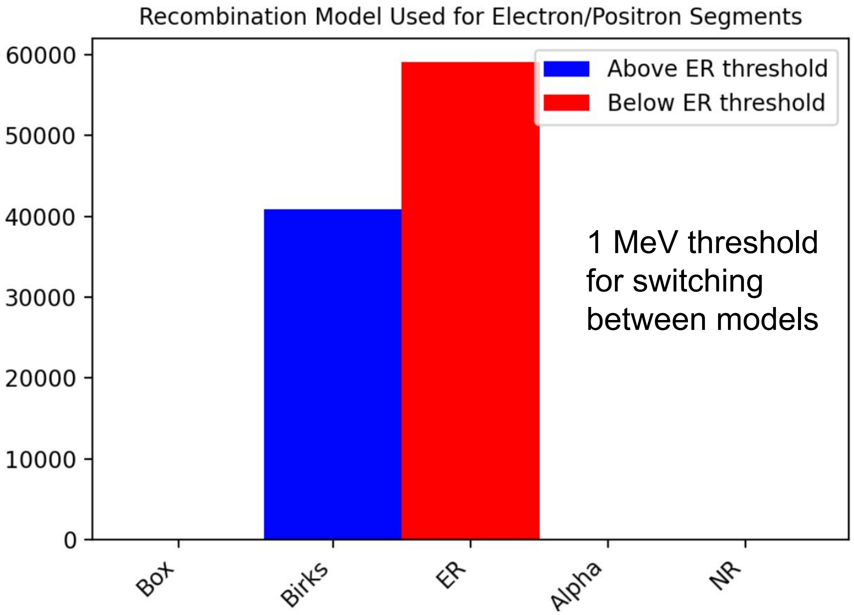
Alpha recombination factor is a constant w.r.t. energy, so it is implemented as just a constant for now

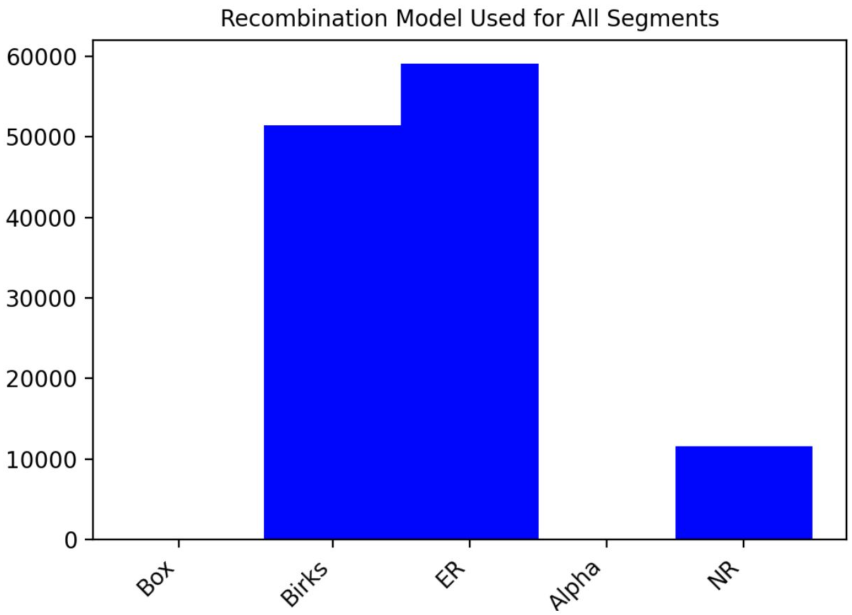
```
@njit
def NEST_ALPHA(E):
    """
    LArNEST alpha recombination model. https://github.com/NESTCollaboration/larnestpy

    Args:
        E (float): Segment dE in MeV.
    """
    return physics.ALPHA_R_FACTOR
```

‘Cached-LArNEST’ Implementation

- Benchmarking tests
    - On a MiniRun3 1e19 RHC file:
        - Cached-LArNEST: **0.536** seconds to run quenching
        - larnd-sim develop branch: **0.170** seconds to run quenching
        - 3.15x slower than original larnd-sim
    - On a 10k events 39Ar beta decay file (electrons < 565 keV)
        - Cached-LArNEST: **0.507** seconds to run quenching
        - larnd-sim develop branch: **0.165** seconds to run quenching
        - 3.07x slower than original larnd-sim
    - Note: The quenching speed depends on the number of data points in the cached LArNEST data due to the interpolation function.
    - Is this slow down acceptable?

‘Cached-LArNEST’ Implementation: MiniRun3 Tests

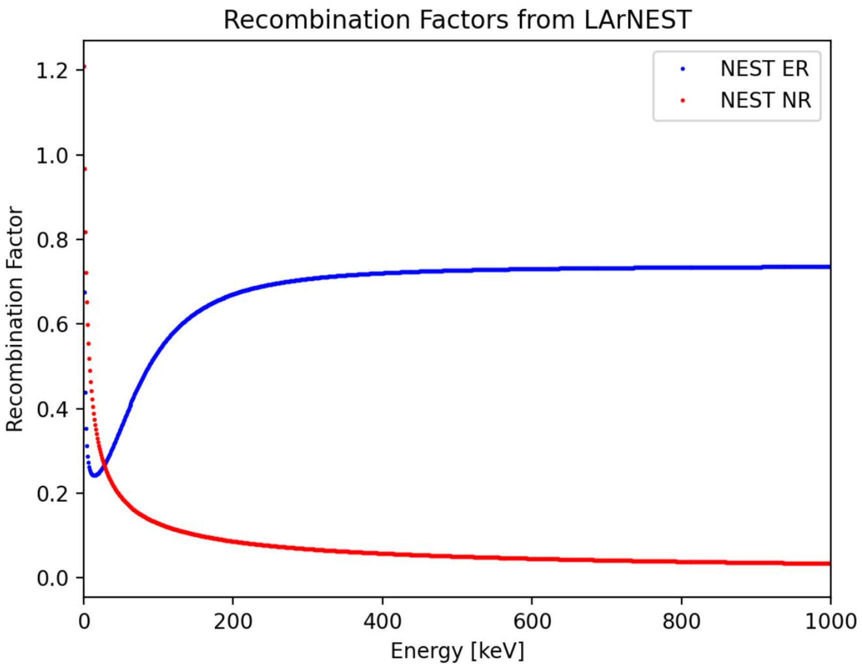




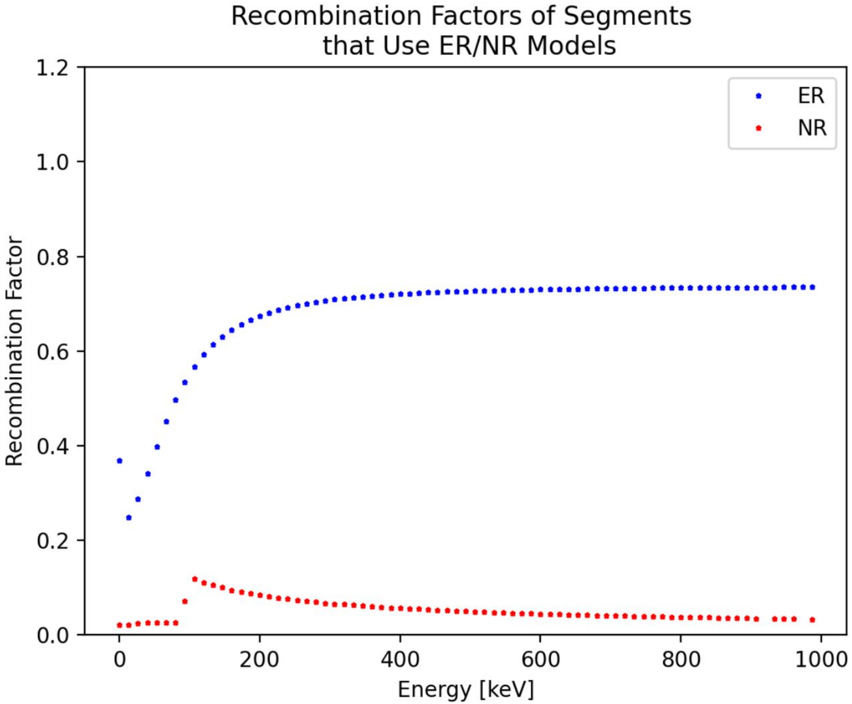
Questions:

- At what energy should we switch from ER to Box/Birks?
- Should we smooth the transition between models, rather than having a hard cut off?

‘Cached-LArNEST’ Implementation: MiniRun3 Tests



Input to larnd-sim



From larnd-sim output

Final Thoughts

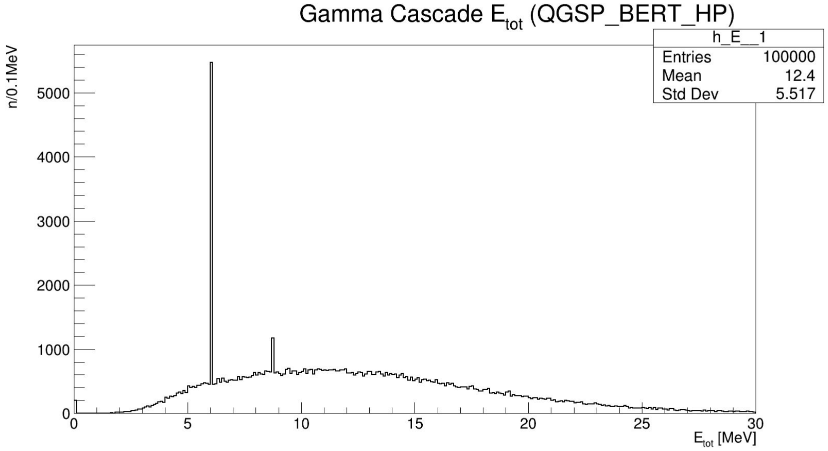
- We now have a version of larnd-sim that:
    - Can utilize the LArNEST electron recoil, alpha, and nuclear recoil recombination models
    - Allows for specifying different recombination models for different particles
- What kinds of tests would others like to see of this implementation?

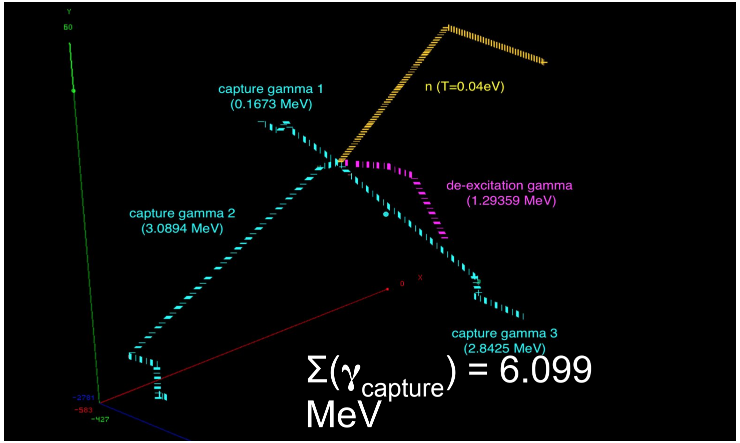
Modified *edepsim* physics list - profiling & validations

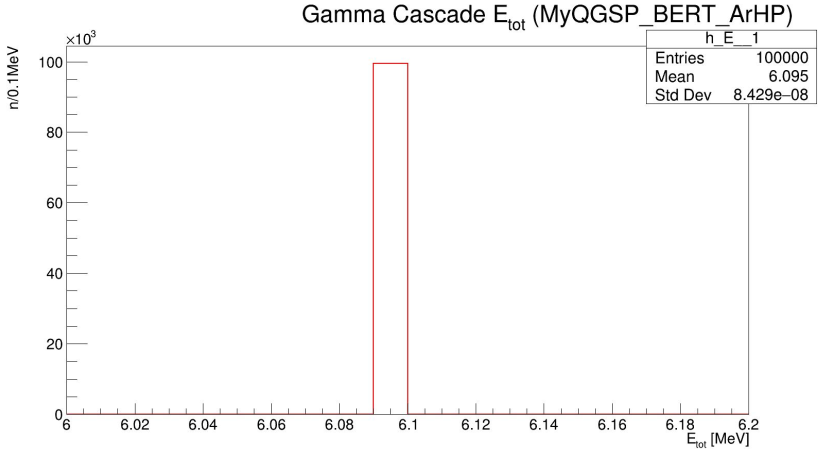
- Three physics lists considered:
  - **QGSP_BERT**: Default physics list used in edepsim
  - **QGSP_BERT_HP**: High precision physics for neutrons up to 20 MeV
  - **MyQGSP_BER_ArHP**: QGSP_BERT_HP with corrected version of gamma cascade production for neutron captures on Argon

Validation of gamma cascade energies

- Expect 6.1 MeV total energy for the gammas
- Custom physics list matches expectation







Output file (n, γ) thresholds

- Neutrons and gammas have momentum thresholds for trajectory info storage in output files
- Hit info is kept, but trajectory info for particles including the physics process that led to its creation may be lost

```
--- a/src/edepsim-defaults-1.0.mac
+++ b/src/edepsim-defaults-1.0.mac
@@ -24,8 +24,8 @@
 /edep/db/set/requireEventsWithHits false
 /edep/db/set/lengthThreshold 1 mm
 /edep/db/set/trajectoryAccuracy 1 mm
-/edep/db/set/neutronThreshold 50 MeV
-/edep/db/set/gammaThreshold 10 MeV
+/edep/db/set/neutronThreshold 0 MeV
+/edep/db/set/gammaThreshold 0 MeV

 ##################################################
```

Resource usage

- Head-to-head comparison of *same event* using example.gdml in edep-sim
- Single evt, with 10GeV proton burst (100 p) above the LArTracker
- Set the random seed for G4 simulation using: /edep/random/randomSeed

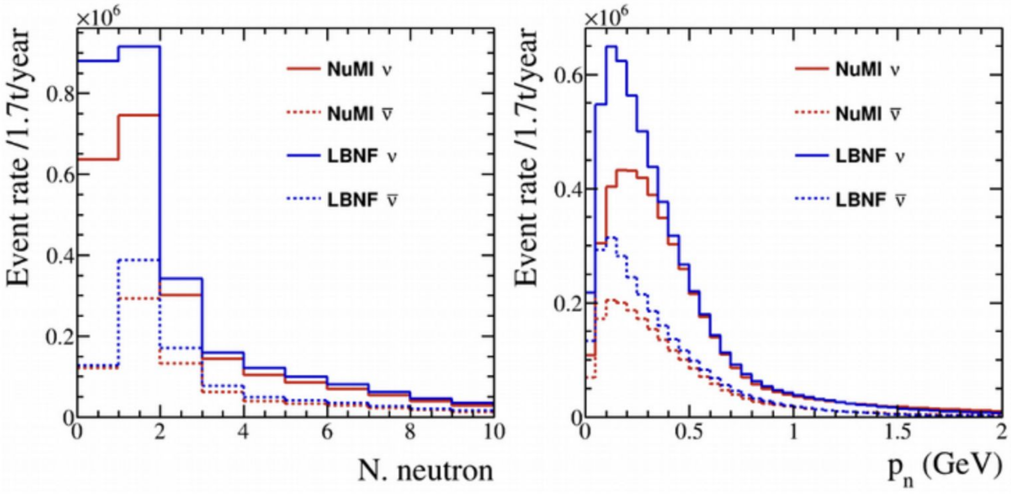
	QGSP_BERT (default cuts)	QGSP_BERT (zero cuts)	QGSP_BERT _HP (default cuts)	QGSP_BERT _HP (zero cuts)	MyQGSP_BE RT_ArHP (default cuts)	MyQGSP_BE RT_ArHP (zero cuts)
# Trajectories	45312	46122	53028	53901	91078	92450
Real Time (s)	260.98	257.25	1662.51	1748.50	1946.36	2007.01
Peak Memory (GB), Heap	1.40	1.402	2.634	2.668	3.532	3.528
File Size (MB)	17	17	21	22	24	28

n-Ar measurement

DUNE ND CDR
arXiv:2103.13910

Current idea is to use the PDS system to tag neutron TOF to reconstruct KE.

Slides from Mike Mooney (https://indico.fnal.gov/event/48610/contributions/212284/attachments/141944/179049/DUNE_NDLAr_AnalysisMeeting_21_04_08.pdf)



- Many neutrons from LBNF beam at DUNE ND-LAr (67 t)
  - Above plots scaled to ArgonCube 2x2 demonstrator (1.7 t)
- Sizable fraction of neutron scatters will produce proton of high enough energy to tag neutron in TPC

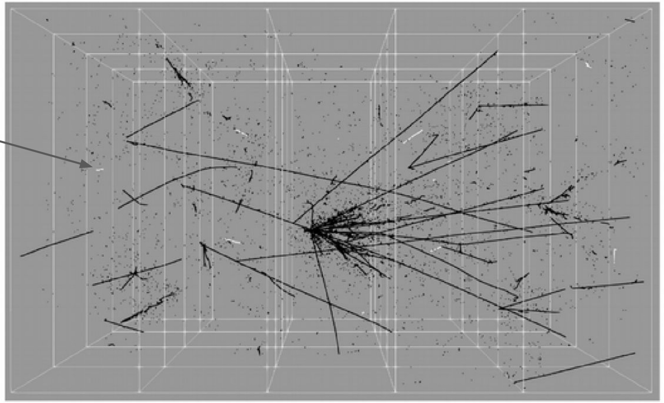
3

n-Ar measurement

Current idea is to use the PDS system to tag neutron TOF to reconstruct KE.

Slides from Mike Mooney (https://indico.fnal.gov/event/48610/contributions/212284/attachments/141944/179049/DUNE_NDLAr_AnalysisMeeting_21_04_08.pdf)

DUNE ND CDR
arXiv:2103.13910

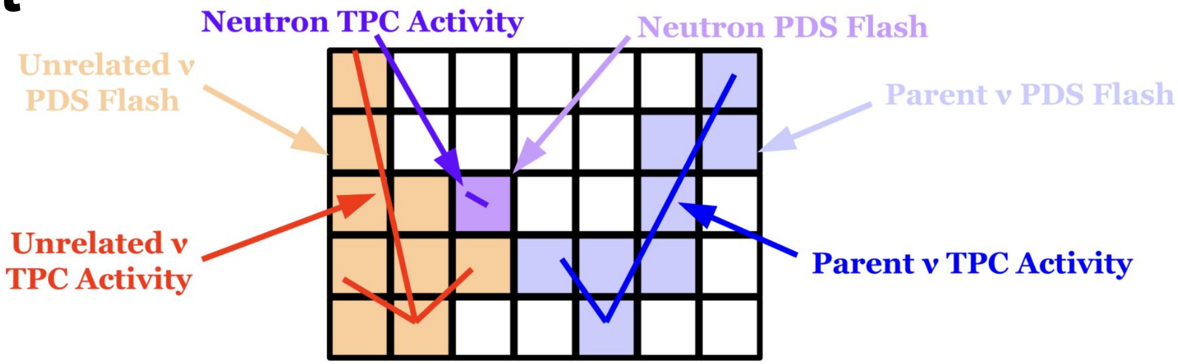


- Tag neutrons via scattering proton TPC tracks (white tracks in above), associate with parent neutrino interaction via light
  - Current PDS timing resolution specification (< 10 ns) allows unambiguous matching to associated neutrino via scintillation light
  - *Can we do more with even better PDS timing precision?*

n-Ar measurement

Current idea is to use the PDS system to tag neutron TOF to reconstruct KE.

Slides from Mike Mooney (https://indico.fnal.gov/event/48610/contributions/212284/attachments/141944/179049/DUNE_NDLAr_AnalysisMeeting_21_04_08.pdf)



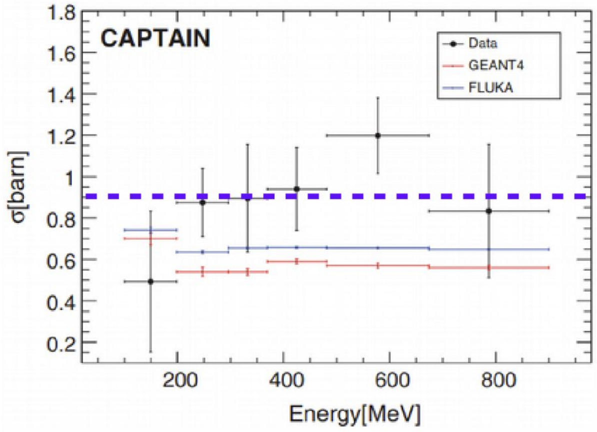
- ♦ Yes! With PDS timing resolution < 3 ns, can measure neutron time of flight (TOF) → neutron energy measurement
  - Energy from proton track in TPC less reliable – detector effects, nuclear effects such as Fermi motion wash things out
  - Reconstructed proton track coupled with TOF neutron energy measurement will provide data sample to tune above effects in MC
- ♦ Measure distance between ν vertex and neutron scatter (proton) track, divide by time of flight (difference of PDS times) → v → E
  - Modular ND-LAr design helps: keeps out late light from ν interaction

6

n-Ar measurement

Current idea is to use the PDS system to tag neutron TOF to reconstruct KE.

Slides from Mike Mooney (https://indico.fnal.gov/event/48610/contributions/212284/attachments/141944/179049/DUNE_NDLAr_AnalysisMeeting_21_04_08.pdf)



PhysRevLett.123.042502

$$dN_B/dx = -T\sigma_T N_B \rightarrow N_B(x) = N_0 e^{-T\sigma_T x}$$

$$T = \rho_{\text{L Ar}} \times N_{\text{Avogadro}}/m_{\text{Ar}}$$

$T \sim 2.11 \times 10^{22}$ cm^{-3}, $\sigma \sim 0.9$ barns

$\rightarrow \boldsymbol{P_{\text{survival}} \sim e^{-x/D}}$, $\boldsymbol{D \sim}$ **50 cm**

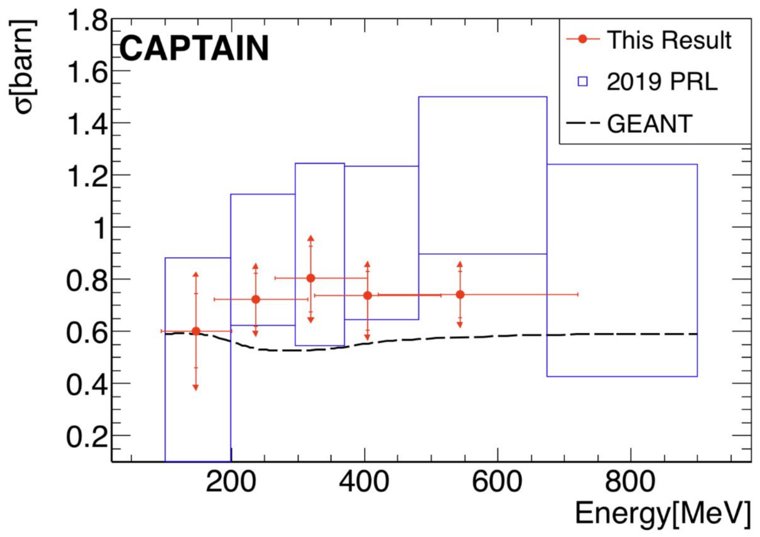
- Use Mini-CAPTAIN measurements to determine neutron survival probability as function of distance from ν vertex
  - Use $\sigma \sim 0.9$ barns for all energies (approximation/simplification)
  - Corresponds to attenuation length $\boldsymbol{D \sim}$ **50 cm**
  - Weight events in average $\Delta E/E$ plots according to $\boldsymbol{P_{\text{survival}} \sim e^{-x/D}}$
- Use minimum distance cut, $x > 50$ cm, to improve $\Delta E/E$

n-Ar measurement

Current idea is to use the PDS system to tag neutron TOF to reconstruct KE.

Slides from Sergey Martynenko (https://drive.google.com/drive/u/0/folders/1KzXxFFPBjOUBwub7Ta8pSR9jxfXXFc6M)

- The described measurement is consistent with the hypothesis of small cross section change across considered energy range;
- Current PRD result [https://doi.org/10.1103/PhysRevD.107.072009]
- Previous PRL result [https://doi.org/10.1103/PhysRevLett.123.042502]

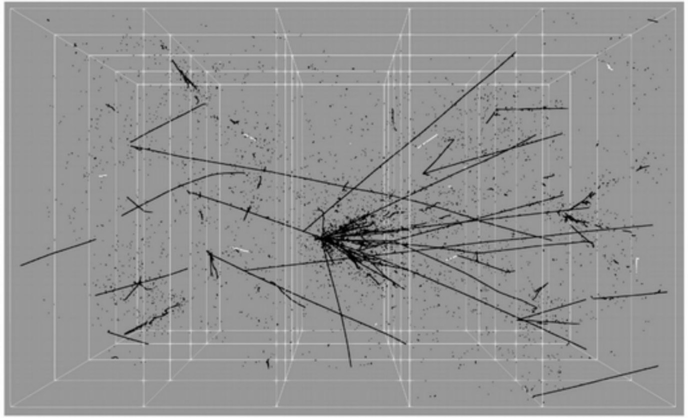


High purity proton sample - Bern cosmic ray data

Initial investigation into proton-related detector systematics using Bern module data

Preliminary proton selection using <4 hours of Module-1 data on hand

- Charge + light matching
- Track fitting with Hough transform
- Field response unfolding
- PIDA to discriminate HIPs from MIPs
- Void (inactive channel) analysis
- ML approach using Blip



Work in progress in coordination with ν_μCC̄0π analysis

- Recast analysis from "flow" files
- Analysis validation

BACKUP

Custom neutron physics list

- **Name:** MyQGSP_BERT_ArHP
- Used in ProtoDUNE-SP for simulations
- Integrated w/ edep-sim (feature/neutron_physics_DR)
- Based on QGSP_BERT_HP reference physics list in GEANT4
    - HP refers to high precision neutron physics
    - Applies to low energy neutrons (< 20 MeV)
    - Custom neutron capture physics (from J. Wang) corrects the gamma cascade production to match NNDC tables for Ar40
- Other changes:
    - Set proton range cut to 0
        - This keeps low energy nuclear recoils

National Nuclear Data Center tools

- https://www.nndc.bnl.gov/capgam/
- CapGam by Target provides data on thermal neutron captures on Ar-40 and Ar-41

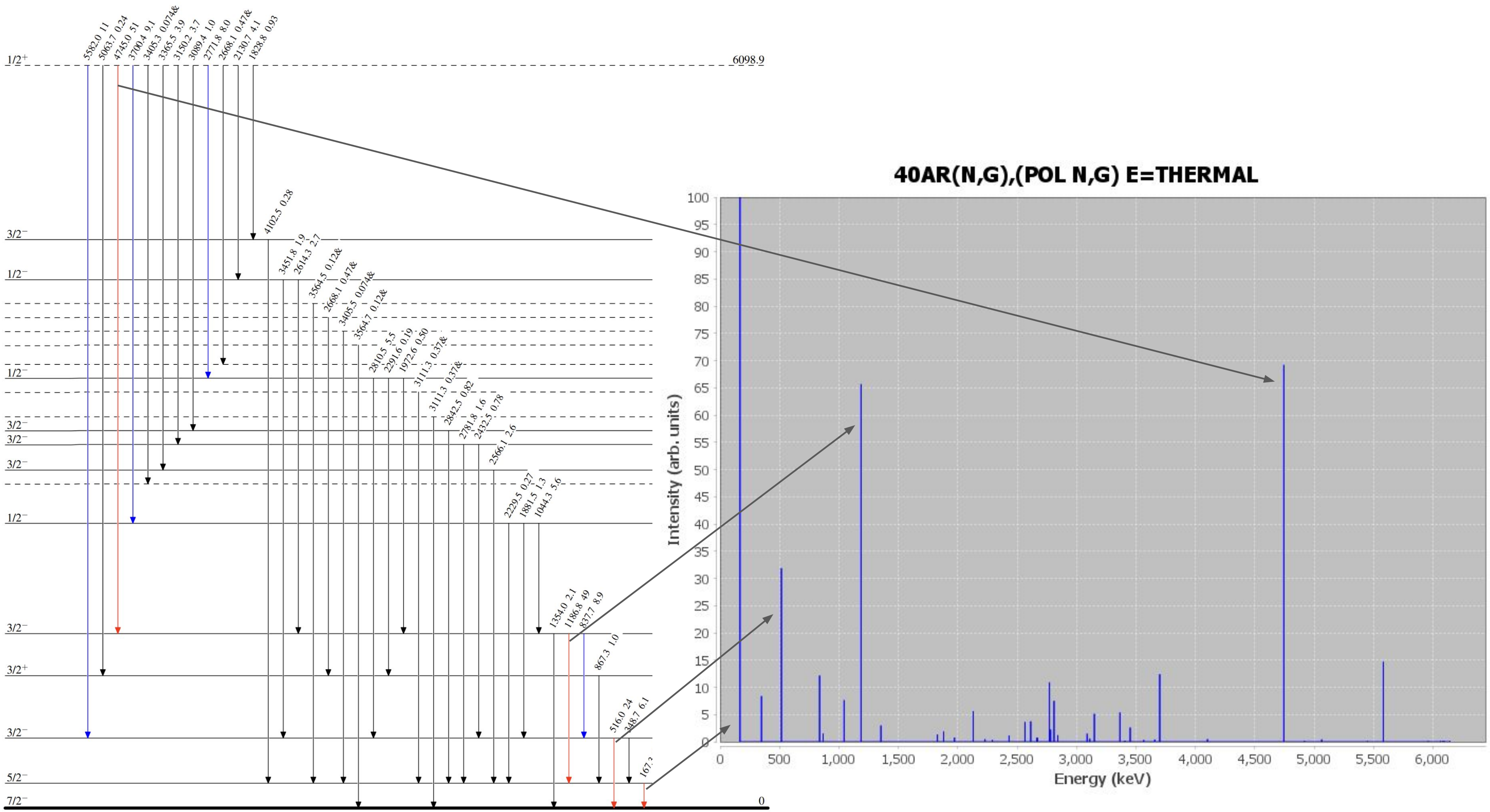
- NNDC Site Index
- **CapGam**
  - About CapGam
  - CapGam by Energy
  - CapGam by Target
- **Resources**
  - ENSDF
  - Nuclear Data Sheets
- **Networks**
  - USNDP
  - NSDD

Thermal Neutron Capture γ's (CapGam)

The energy and photon intensity with uncertainties of gamma rays as seen in thermal-neutron capture are presented in two tables, one in ascending order of gamma energy and a second organized by Z, A of the target. In the energy-ordered table the three strongest transitions are indicated in each case. The nuclide given is the target nucleus in the capture reaction. The gamma energies given are in keV. The gamma intensities given are relative to 100 for the strongest transition. %Iγ (per 100 n-captures) for the strongest transition is given, where known.

All data are taken from *Evaluated Nuclear Structure Data File (ENSDF)*, a computer file of evaluated nuclear structure data and eXperimental Unevaluated Nuclear Data List (XUNDL), both maintained by the National Nuclear Data Center, Brookhaven National Laboratory, on behalf of the U.S. Nuclear Data Program and Nuclear Structure and Decay Data network. The data for $A > 20$ are published in the *Nuclear Data Sheets*, Elsevier. The data for $A \leq 20$ are published in *Nuclear Physics **A***.

This research was supported by the Office of Nuclear Physics, Office of Science, US Department of Energy.



40AR(N,G),(POL N,G) E=THERMAL
Intensity (arb. units)
Energy (keV)
1/2+
6098.9
3/2−
1/2−
1/2−
3/2−
3/2−
3/2−
1/2−
3/2−
3/2+
3/2−
5/2−
7/2−
0

Simulated event for profiling