Looking into Radiative Effects Using RADGEN with Pythia

Richard Capobianco

Graduate Student of University of Connecticut – Argonne National Lab

SIDIS Group Meeting 3-25-2025

Motivation

- Semi-Inclusive Deep Inelastic Scattering (SIDIS) experiments allow us to address questions about the 3D structure of nucleons
- Azimuthal modulations in unpolarized SIDIS cross-section for charged pion electroproduction can give access to the Cahn and Boer-Mulders effects
  - **Boer-Mulders Effect:** Sensitive to the correlation between the quark's transverse momentum and intrinsic transverse spin in an unpolarized nucleon
  - **Cahn Effect:** Sensitive to the transverse motion of quarks inside the nucleon
- A non-zero Boer-Mulders requires quark orbital angular momentum contributions to the proton spin (aspect of the proton missing spin puzzle)

SIDIS Cross-Section and Boer-Mulders

The lepton-hadron Unpolarized SIDIS Cross-Section:

$$\frac{d^5\sigma}{dy\,dQ^2\,dz\,d\phi_h\,dP_{h\perp}^2} = \underbrace{\frac{x_B}{y}\frac{2\pi\alpha^2}{x_B y Q^2}\frac{y^2}{2(1-\epsilon)}\left(1+\frac{\gamma^2}{2x_B}\right)(F_{UU,T}+\epsilon F_{UU,L})}_{A_0}\left\{1+\underbrace{\frac{\sqrt{2\epsilon(1+\epsilon)}F_{UU}^{\cos\phi_h}}{(F_{UU,T}+\epsilon F_{UU,L})}}_{A_{UU}^{\cos\phi_h}}\cos\phi_h+\underbrace{\frac{\epsilon F_{UU}^{\cos 2\phi_h}}{(F_{UU,T}+\epsilon F_{UU,L})}}_{A_{UU}^{\cos 2\phi_h}}\cos 2\phi_h\right\}$$

The Boer-Mulders and Cahn effects are present in the Structure Functions:

leading twist

$$F_{UU}^{\cos 2\phi_h} \propto C\left[-\frac{2(\hat{P}_{h\perp}\cdot\vec{k}_T)(\hat{P}_{h\perp}\cdot\vec{p}_T)-\vec{k}_T\cdot\vec{p}_T}{MM_h}h_1^\perp H_1^\perp+\dots\right]$$

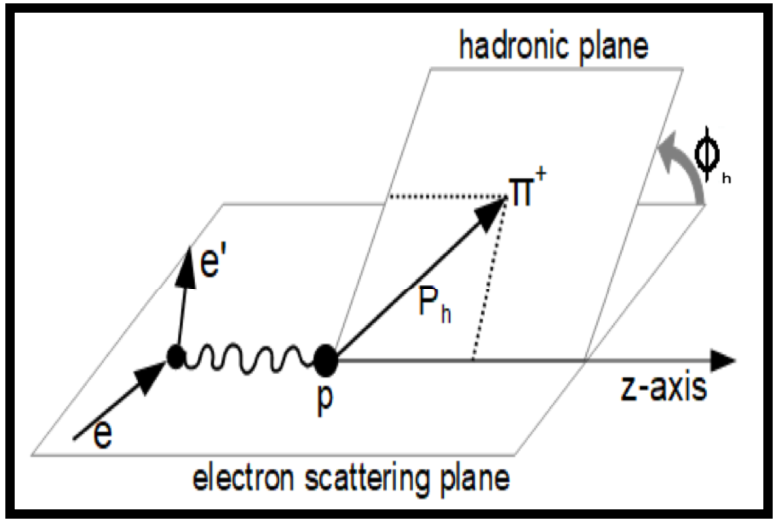
next to leading twist

$$F_{UU}^{\cos\phi_h} \propto \frac{2M}{Q}C\left[-\frac{\hat{P}_{h\perp}\cdot\vec{k}_T}{M_h}x\,h\,H_1^\perp-\frac{\hat{P}_{h\perp}\cdot\vec{p}_T}{M}f_1D_1+\dots\right]$$

BOER-MULDERS EFFECT

CAHN EFFECT

Interaction dependent terms neglected



Reaction Studied: $ep \rightarrow e\pi^+(X)$

Analysis Procedure

Experimental extraction of cross-section

$$\frac{d^5\sigma}{dQ^2 dy dP_T dz d\phi_h} = \frac{1}{\Gamma_\nu} \underbrace{\frac{1}{\Delta Q^2 \Delta y \Delta P_T \Delta z \Delta \phi_h}}_{\text{Bin Volume}} \frac{N}{R \cdot BC \cdot \eta \cdot N_0} \underbrace{\frac{1}{(N_A \cdot \rho \cdot t/A_w)}}_{\text{Target Number Density}}$$

Where:

- R = **Radiative Correction**
- η = **Acceptance Correction**
- N = Bin Yields
- N_0 = Life-time corrected incident electron flux
- BC = factor which evolves bin-averaged differential cross-section

Requires Monte Carlo (MC) Simulation

Radiative effects will be introduced using RADGEN

Required switching from 'clasdis' (PEPSI) to a modified version of Pythia 6 developed by HERMES/EIC (parameters tuned as in 'claspyth')

RADGEN Explanation – Introduction

- RADGEN is a MC generator for polarized and unpolarized DIS radiative events
  - Can generate Radiative Corrections for inclusive, semi-inclusive and exclusive DIS processes
- The radiative corrections come from the sum of these contributions simulated in RADGEN:

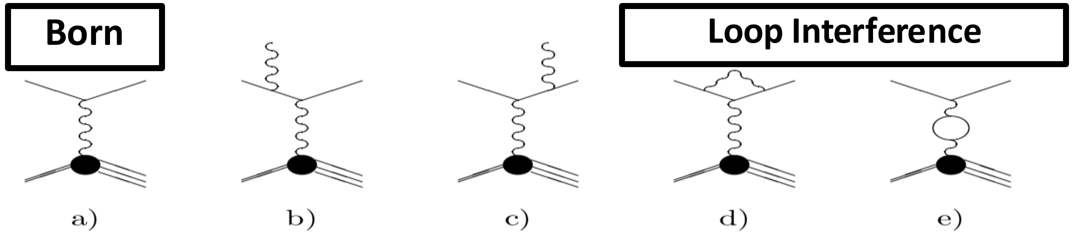


Figure 1: Feynman diagrams contributing to the Born and the radiative correction cross sections in lepton-nucleus scattering.

$$\sigma^{rad.corr.} = \sigma_{in} + \sigma_q + \sigma_{el} + \sigma_v$$

- Events generated based on their contributions to the observed total cross-section:

$$\sigma_{obs} = \sigma_{non-rad}(\Delta) + \sigma_{in}(\Delta) + \sigma_q + \sigma_{el}$$

  - Where $\sigma_{non-rad}(\Delta)$ includes the Born process (can be simulated by MC without RADGEN – i.e. GEN_{Born})
  - GEN_{Rad} and Data are the simulated and experimentally measured values of σ_{obs}
  - **Therefore, the Radiative Corrected (Born Level) Cross-section (σ_{cor}) can be given by:**

RC = $\dfrac{GEN_{Rad}}{GEN_{Born}}$ Radiative Correction Factor

$$\sigma_{cor} = \frac{Data}{RC} = \frac{GEN_{Born}\,Data}{GEN_{Rad}}$$

UCONN | UNIVERSITY OF CONNECTICUT Argonne NATIONAL LABORATORY U.S. DEPARTMENT OF ENERGY Office of Science Jefferson Lab clas

RADGEN Implementation in Pythia

1. **RADGEN begins by taking what Pythia started to generate for the scattered lepton**
2. **Then chooses scattering channel (non-radiative, elastic, quasielastic, or inelastic)**
   - Choice based on total observed cross section
3. **If a radiative channel is selected, the photon kinematics are then determined from a previously generated lookup table**
4. **The other kinematic variables (Q^2 and ν) are recomputed to obtain their true values:**

$$W^2_{true} = W^2 - 2E_\gamma(\nu + M - \sqrt{\nu^2 + Q^2}\cos\theta_\gamma), \quad \nu_{true} = \nu - E_\gamma,$$

$$Q^2_{true} = Q^2 + 2E_\gamma(\nu - \sqrt{\nu^2 + Q^2}\cos\theta_\gamma), \qquad x_{true} = \frac{Q^2_{true}}{2M\nu_{true}}$$

   - **These *true* variables are equivalent to what would be calculated by using $q = e - e' - \gamma$ where *e*, *e'*, and *γ* are the 4-vectors of the incoming beam, scattered electron, and radiated photon**
   - **Used to generate the rest of the Pythia event after returning these variables and the radiated photon kinematics**
5. **New event weight given by ratio of the radiatively corrected and Born cross sections**
   - Recalculation of weights uses the Mo-Tsai approach for the unpolarized Monte Carlo

(Initial) Issues with Pythia to be Addressed

The following issues were discovered in the implementation of RADGEN into Pythia when I started working with it in this analysis:

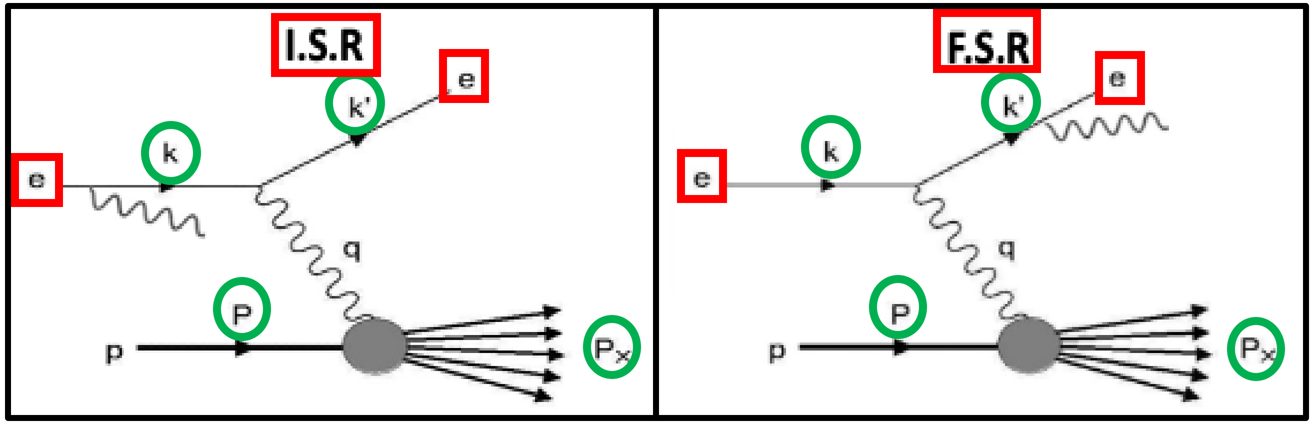
1. Did not flag whether the photon radiated from the incoming (ISR) or outgoing (FSR) electrons
2. Was not reporting the correct beam/scattered electron kinematics for events that radiated a photon (i.e., did not include the ISR/FSR effects on the electrons' kinematics)
3. The photon's kinematics were not reported in the same lab-frame as the other particles
4. Did not set the minimum photon energy parameter properly
5. **The FSR photon's azimuthal distribution around the scattered electron is not defined uniformly**

Point 5 is the only issue above that is still currently unresolved

Addressing Issues 1 and 2

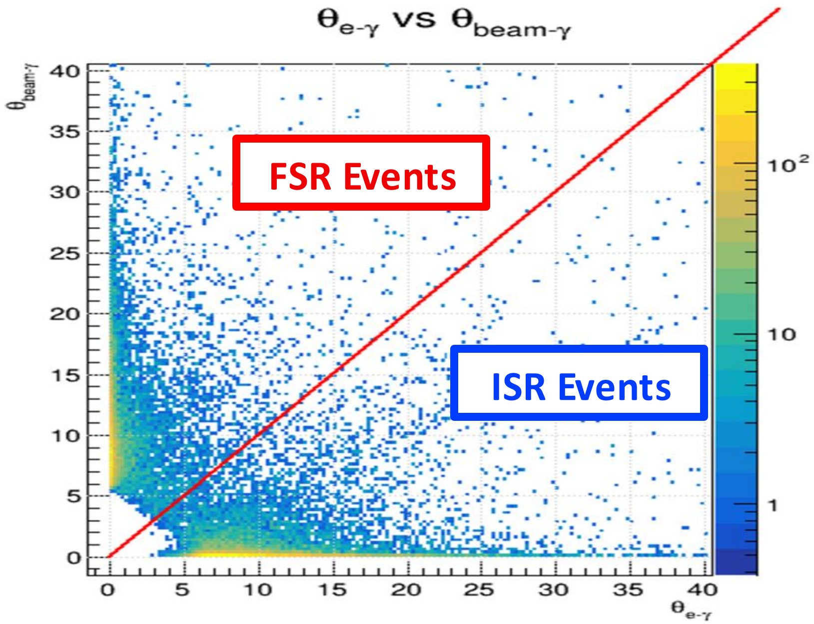
Issue 1: Did not flag whether the photon radiated from the incoming (ISR) or outgoing (FSR) electrons

- **Pythia only was giving me the information in the green circles – Missing info to be found in red squares**
  - Pythia used the 10.6 GeV beam energy as the beam at the vertex with/without RADGEN **(i.e., Issue 2)**



- **To distinguish ISR from FSR:**

Compare the angles between the radiated photon and the scattered electron ($\theta_{e\text{-}\gamma}$) and the beam ($\theta_{beam\text{-}\gamma}$)



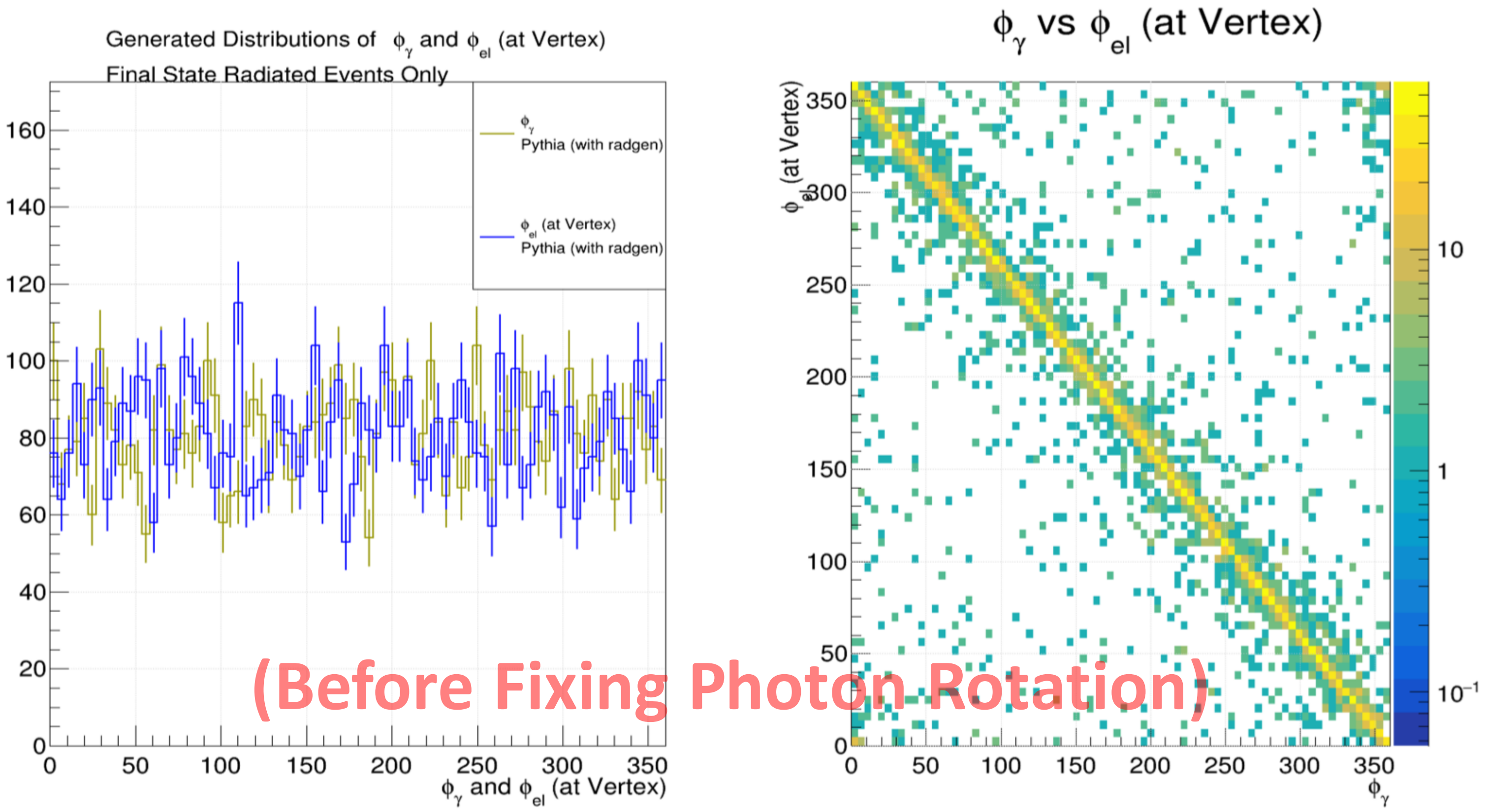
- If the **radiated photon** is more in-line with the **scattered electron** than the beam, **(i.e., $\theta_{e\text{-}\gamma} < \theta_{beam\text{-}\gamma}$)** then the event is **FSR**
- If the **radiated photon** is more in-line with the **beam** than the scattered electron, **(i.e., $\theta_{beam\text{-}\gamma} < \theta_{e\text{-}\gamma}$)** then the event is **ISR**

Once Flagged: Can resolve Issue 2 by applying momentum/Energy conservation to the 4-vectors of the electron that radiated the photon

Addressing Issue 3

Issue 3: The photon's kinematics were not reported in the same lab-frame as the other particles

- RADGEN assumed that the photon kinematics being returned were in the lab-frame, however, they were actually closer to being in the Center-of-Mass Frame (used by Pythia when generating the event)
- **Caused the FSR photon's ϕ-angle to be <u>anti-correlated</u> with the scattered electron in the reported frame**



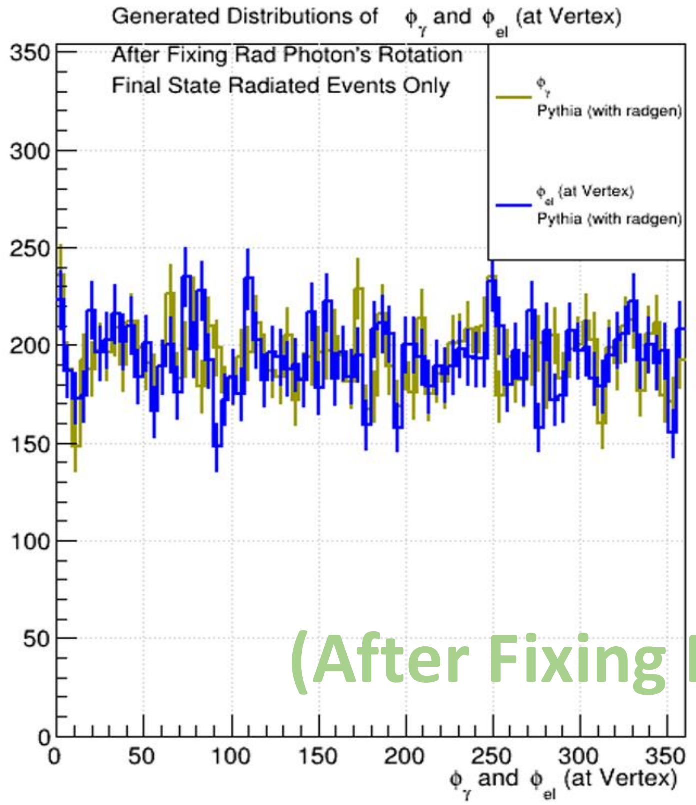
Both angles are measured from around the incoming beam

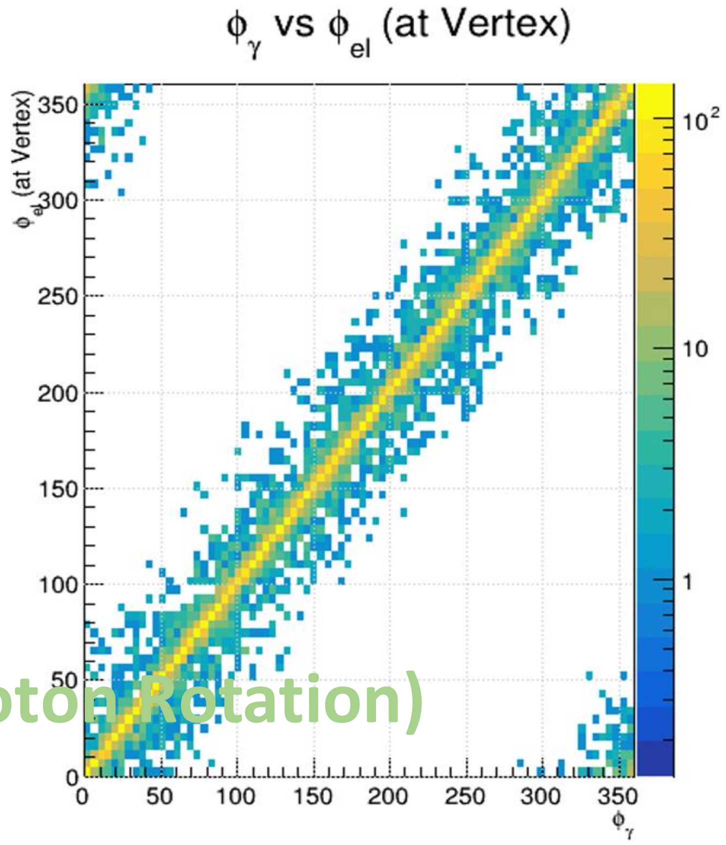
Includes <u>FSR</u> Events Only

Addressing Issue 3

Issue 3: The photon's kinematics were not reported in the same lab-frame as the other particles

- RADGEN assumed that the photon kinematics being returned were in the lab-frame, however, they were actually closer to being in the Center-of-Mass Frame (used by Pythia when generating the event)
- **Caused the FSR photon's ϕ-angle to be anti-correlated with the scattered electron in the reported frame**





(After Fixing Photon Rotation)

Both angles are measured from around the incoming beam

Includes FSR Events Only

Fixed by rotating the photon's kinematics around the x-axis (i.e., flipped the signs of the photon's y and z momentum)

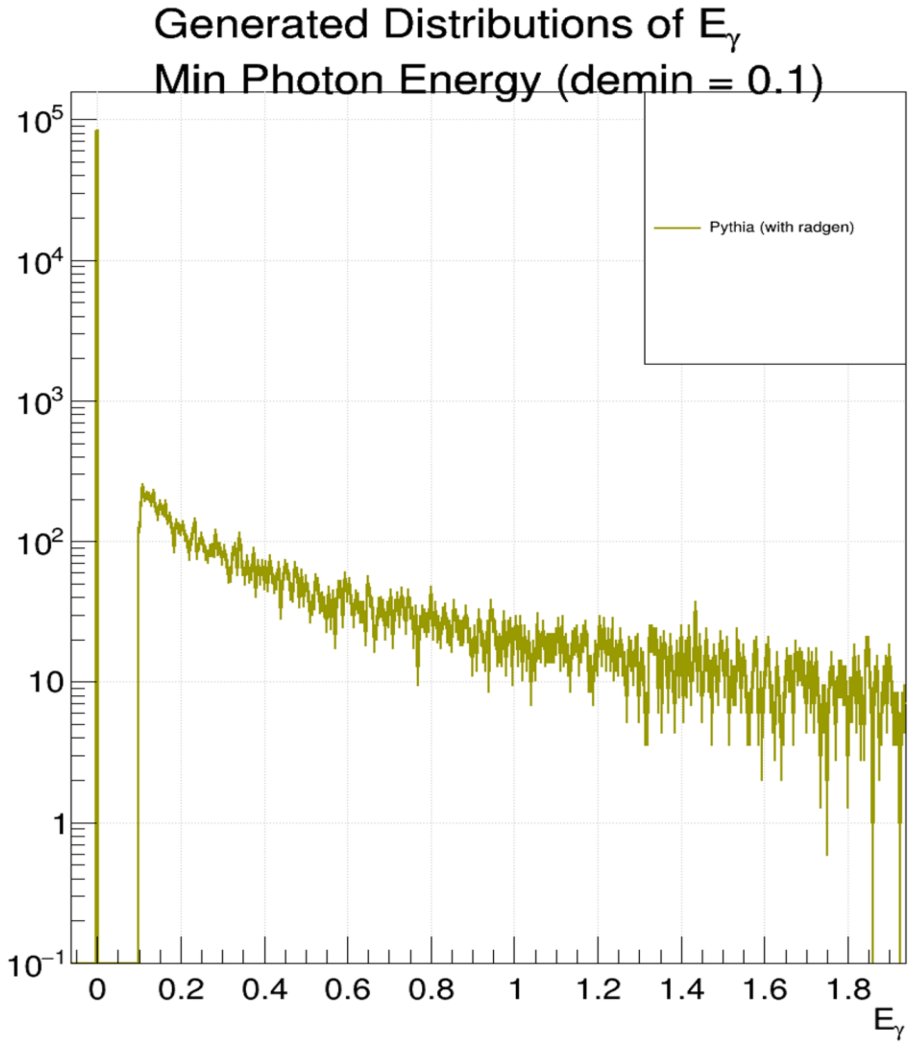
Addressing Issue 4

Issue 4: Did not set the minimum photon energy parameter (demin) properly

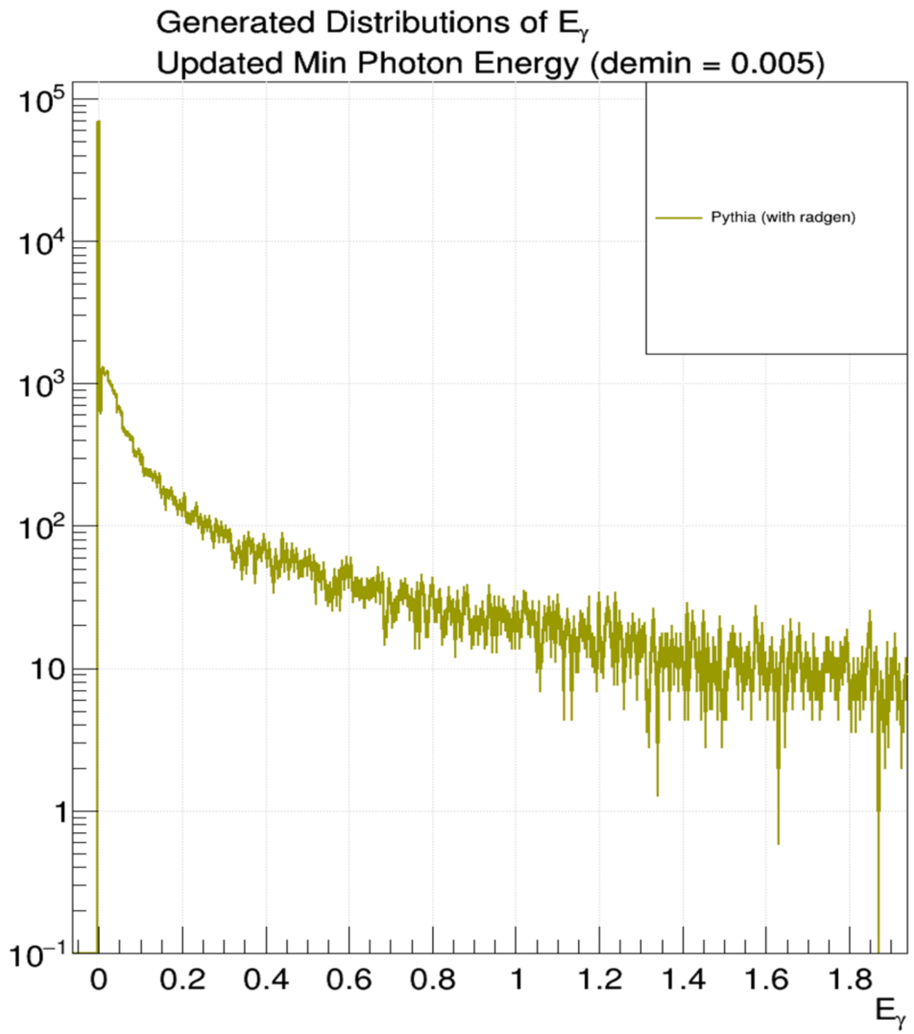
- **The RADGEN parameter demin controls the minimum radiated photon energy generated ($E_{\gamma min}$)**
  - Was set to 100 MeV for HERMES, but I later modified it to 5 MeV (to include photons near our MM_N peak)
- Another issue was found while looking into **Issue 5** (FSR ϕ_γ around the scattered electron):
  - For some reason, the distribution suddenly changed at around demin = 18-19 MeV **(behavior to be shown in the following slides)**
  - Origin may come from the θ_γ angle since it can be affected by a parameter called Δ which is set by this implementation of RADGEN using $E_{\gamma min}$
    - **i.e., Changing demin effects both Δ and $E_{\gamma min}$** (the RADGEN paper states that this is customary)
- Things impacted by changing Δ:
  - The integration limit (θ_{max}) of the cross section's integration over θ_γ is defined as $\theta_{max} = \min(\pi, \theta_\Delta)$ where θ_Δ can be found from $W^2_{true} = W^2 - 2\Delta(\nu + M - \sqrt{\nu^2 + Q^2}\cos\theta_\Delta)$
  - The radiative correction factor $\delta_R(\Delta) = \exp\left[-\frac{\alpha}{\pi}\left(\ln\frac{E}{\Delta} + \ln\frac{E'}{\Delta}\right)\left(\ln\frac{Q^2}{m^2} - 1\right)\right]$

Radiated Photon Energies

Before Modifications to RADGEN



After Modifications to RADGEN



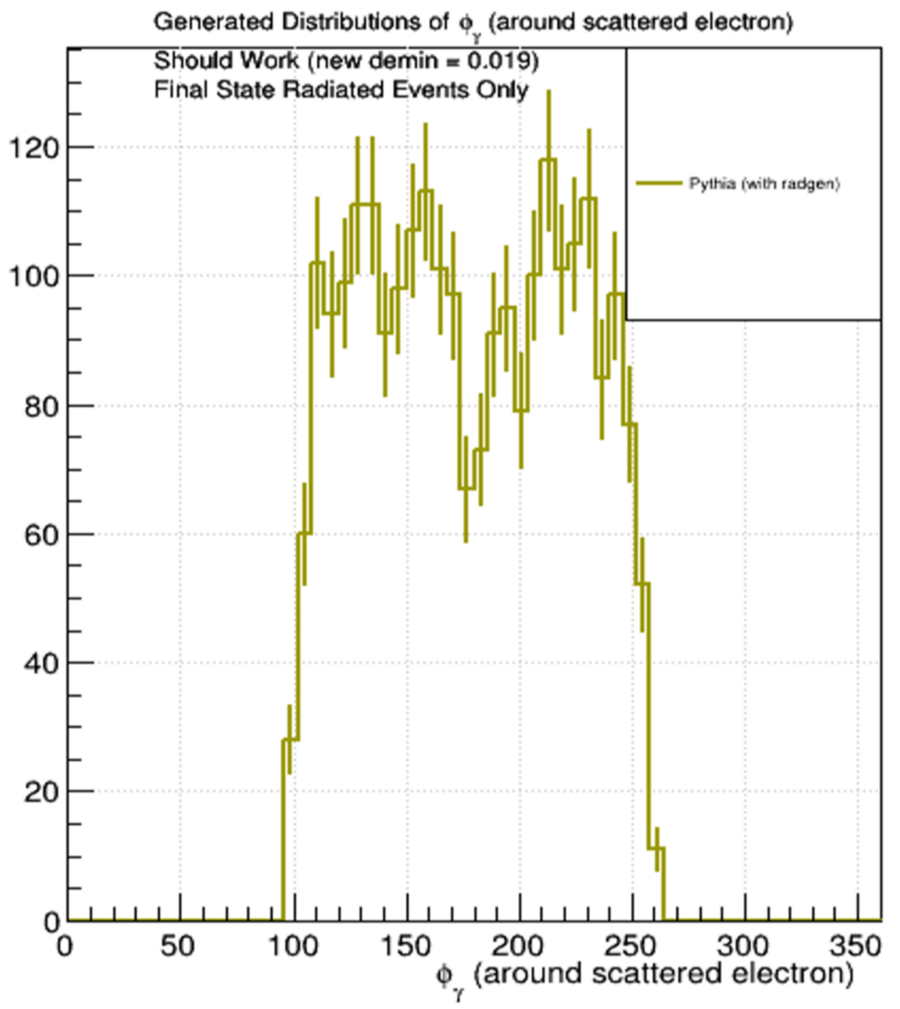
Changes from demin = 100 MeV to demin = 5 MeV

Includes Radiated and Non-Radiated Events

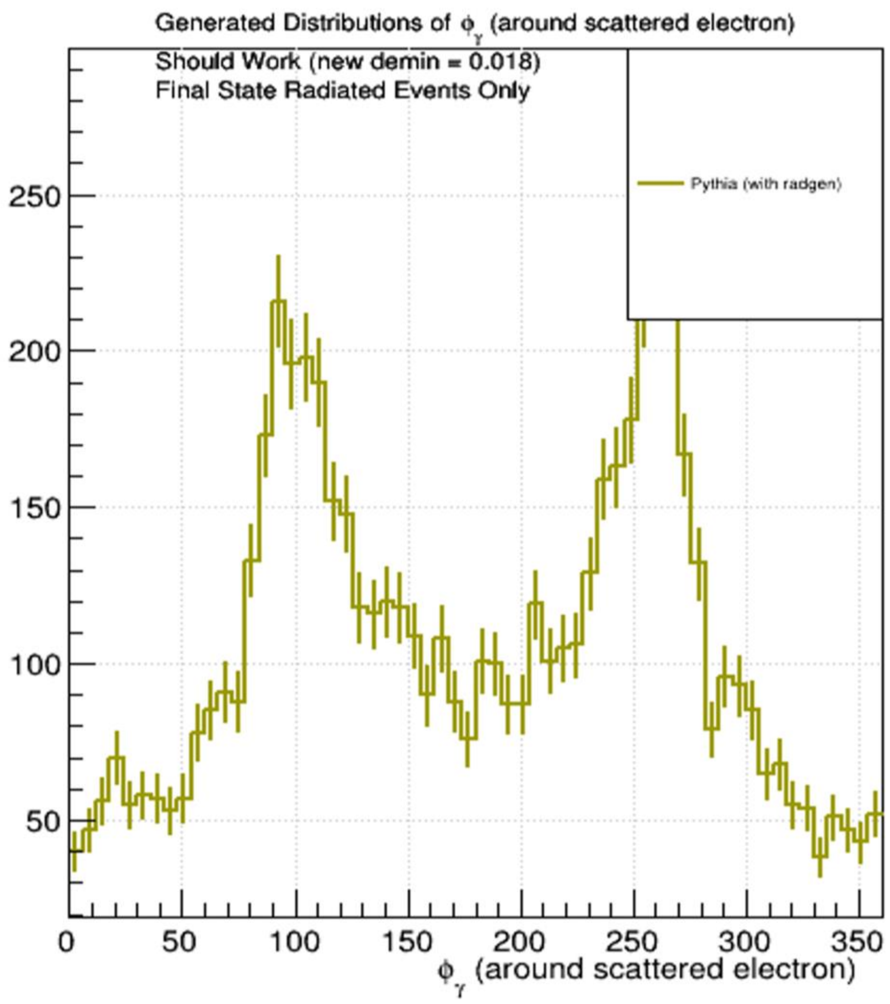
Bin Size = 5 MeV/bin

ϕ_γ of FSR Photons Around the Scattered Electron

Larger demin value (19 MeV)



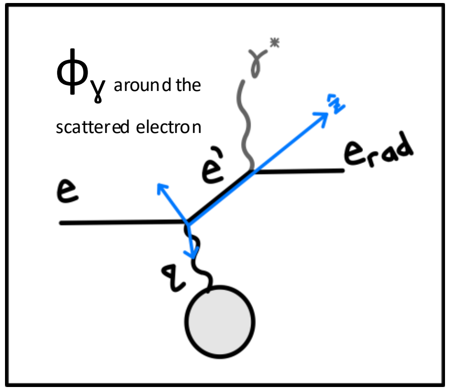
Smaller demin value (18 MeV)



The behavior of this angle changes suddenly between demin=19 MeV an demin=18 MeV

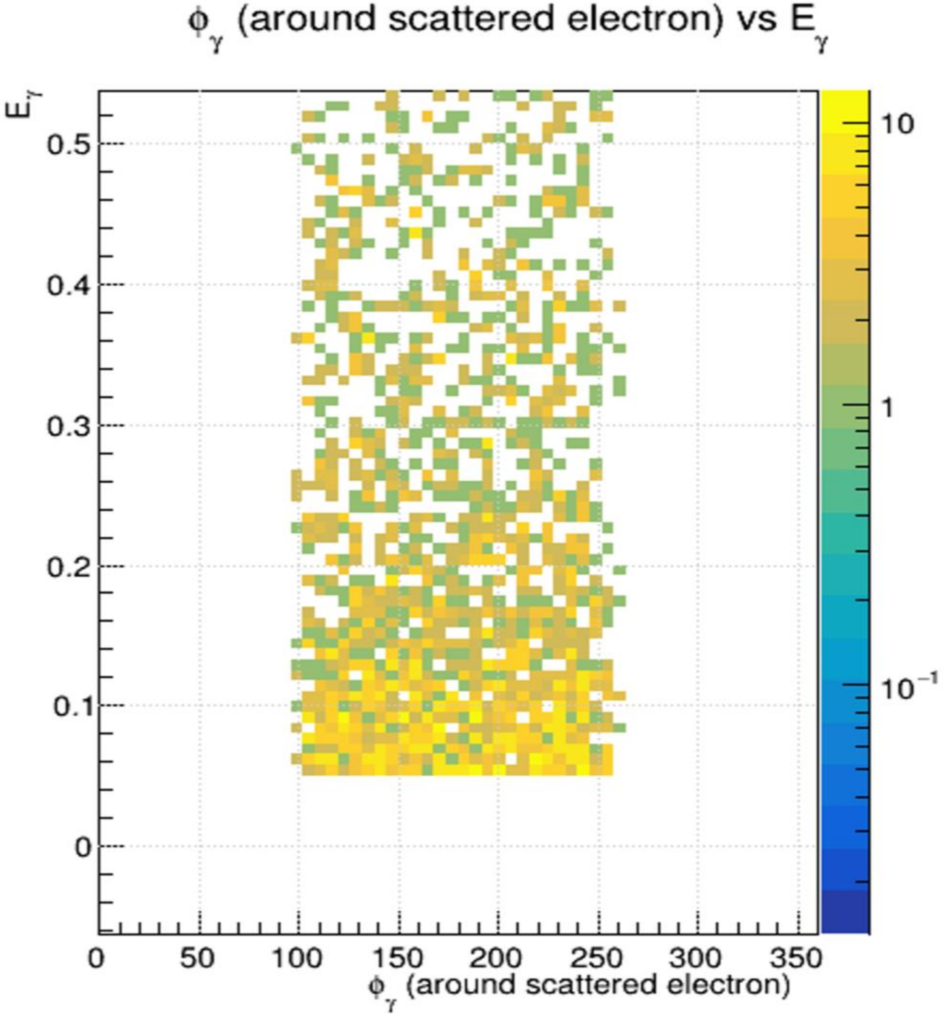
(Checked other values to confirm that this transition is not gradual)

Both behaviors are still wrong (See Issue 5)

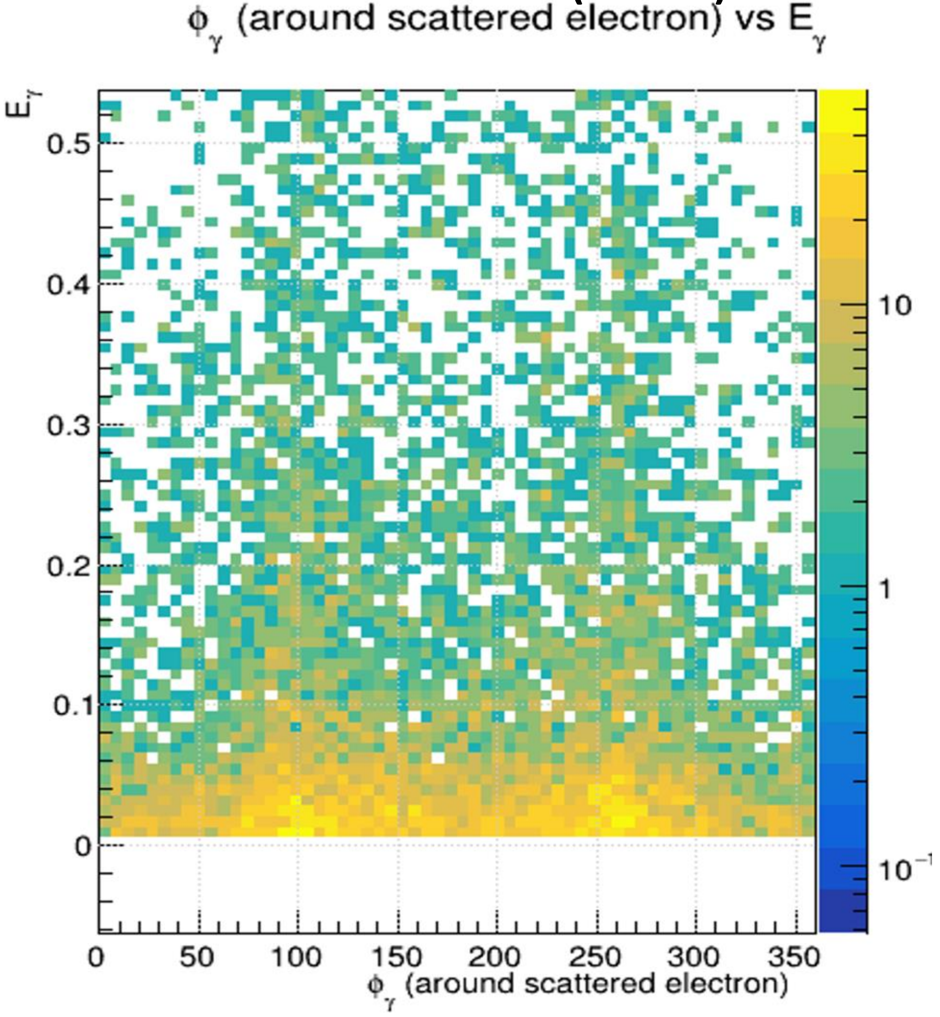


E_γ vs ϕ_γ of FSR Photons Around the Scattered Electron

Larger demin value (50 MeV)

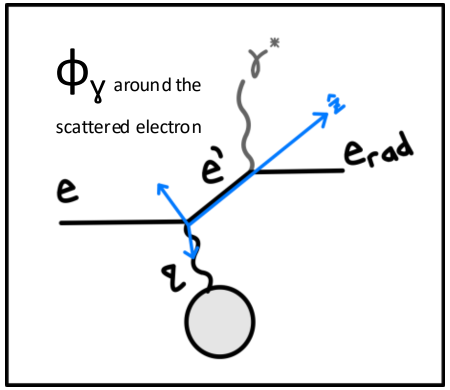


Smaller demin value (5 MeV)



Changing demin is <u>NOT</u> the same as cutting on the Photon's Energy

This parameter effects the all events generated with RADGEN, even those where E_γ > demin

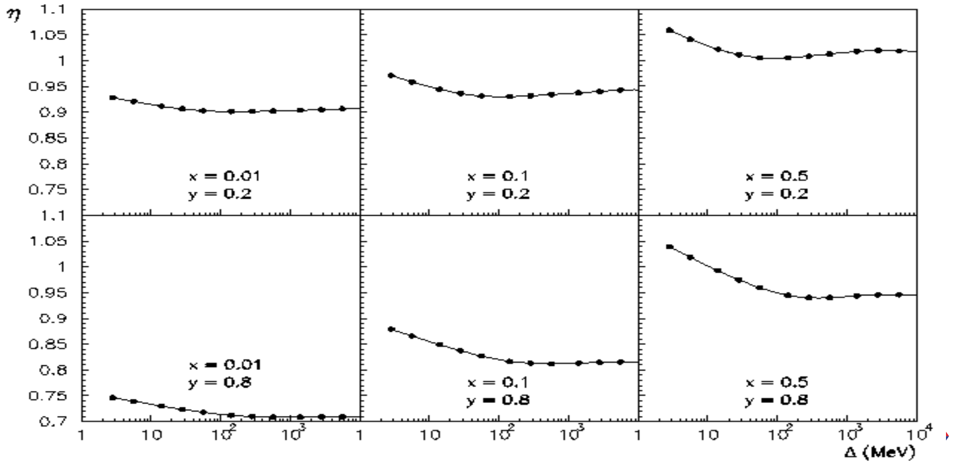


Addressing Issue 4: Fixing demin

- The RADGEN Paper states that Δ should be "sufficiently large to avoid numerical instabilities because for Δ = 0 the contribution $\sigma_{in}(\Delta)$ gets infinite" while also being "sufficiently small to reduce the soft photon region which is calculated only approximately"
  - Claims that the best choice for Δ is generally about 0.1% of the beam energy (Cites: arXiv:hep-ph/9403238)
  - That paper states the following:

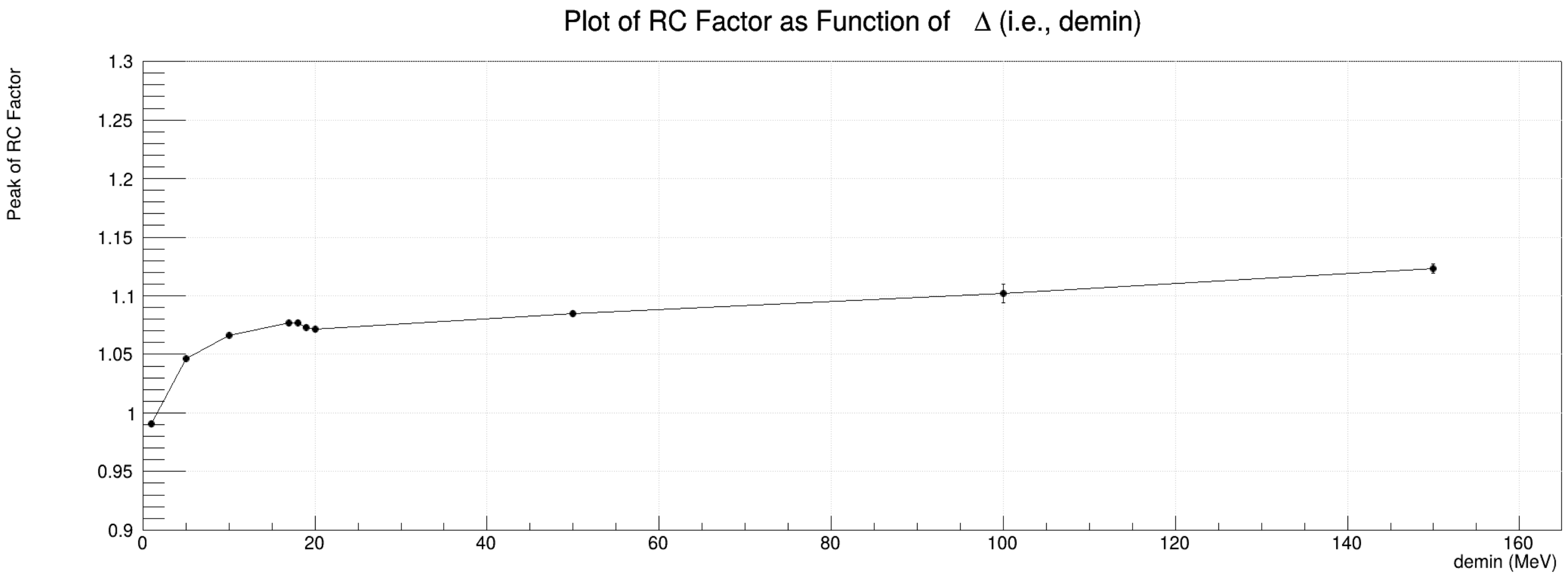
> Corrections depicted by the diagrams in fig. 1 can be divided into two groups: emission of real photons (fig. 1b) with energy larger than Δ and those with energy smaller than Δ (fig. 1b) together with virtual corrections (fig. 1c,d). Contributions from the soft photon emission and from the vertex correction *separately* are infrared divergent but the divergences cancel when the contributions are considered jointly [10]. The parameter Δ may have a meaning of the energy resolution or in another words a maximal energy of the emitted photon which is still not detectable in the experiment. It is often called the 'infrared cut–off' parameter. This interpretation implies that Δ may not be too large. Too small Δ must also be avoided since it may cause numerical instabilities in the computation of the soft photon contribution. The numerical results of the calculations should not depend on Δ.

  - Plotted the radiative correction factor $\delta_R(\Delta) = \exp\left[-\frac{\alpha}{\pi}\left(\ln\frac{E}{\Delta} + \ln\frac{E'}{\Delta}\right)\left(\ln\frac{Q^2}{m^2} - 1\right)\right]$ vs Δ to determine its ideal value*

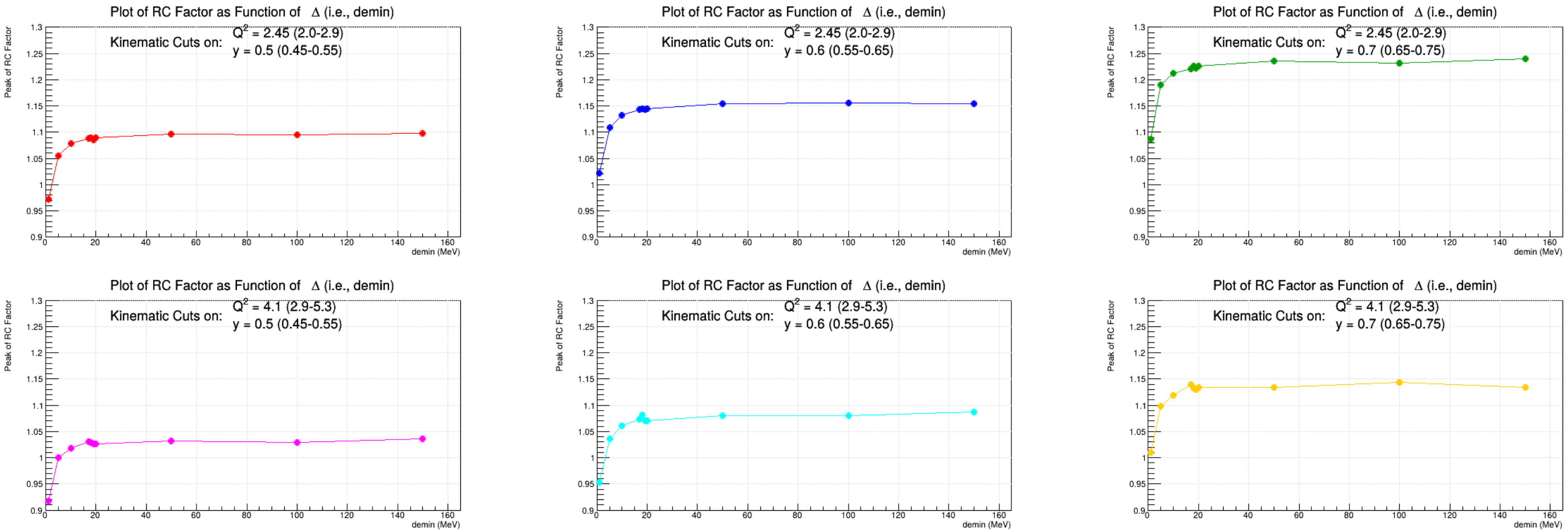


*The red text means that the paper that RADGEN cited defined this factor as the factor in the exponent while RADGEN defines it as the whole exponential term

Addressing Issue 4: Fixing demin



- demin = 5 MeV is too low by these standards
- The cutoff point that seems to be reasonable is around 20 MeV (i.e., where the sudden change in the ϕ_γ around the scattered electron begins)



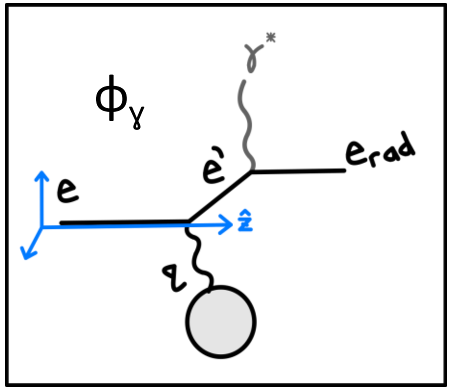
- **Conclusion:**

  Set demin > 18 MeV and try to address **Issue 5** based on those plots

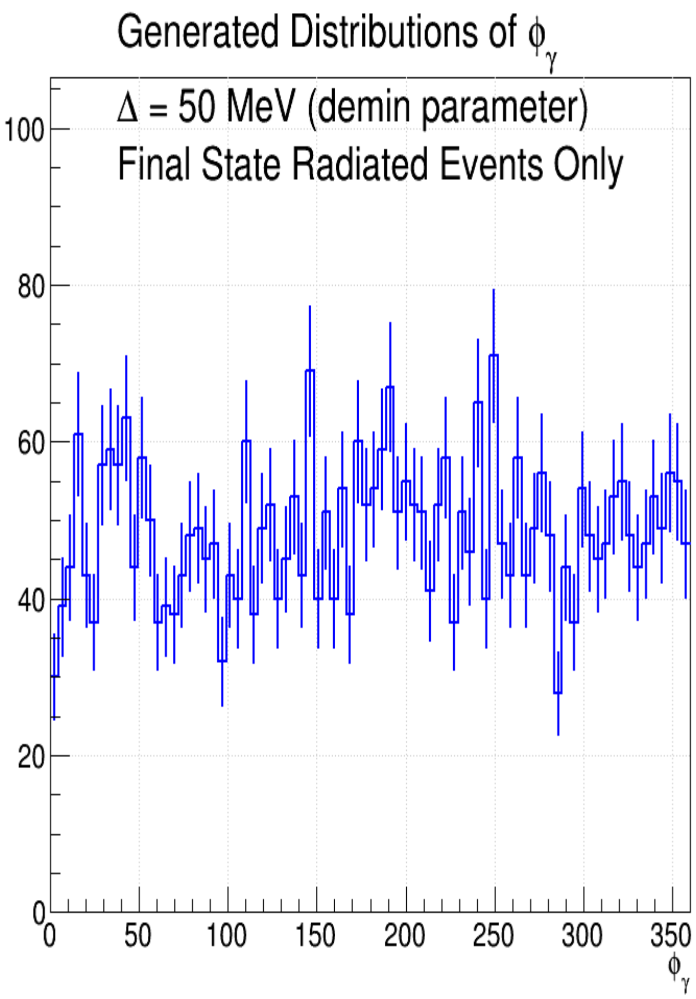
←Showing the Q^2 and y dependences

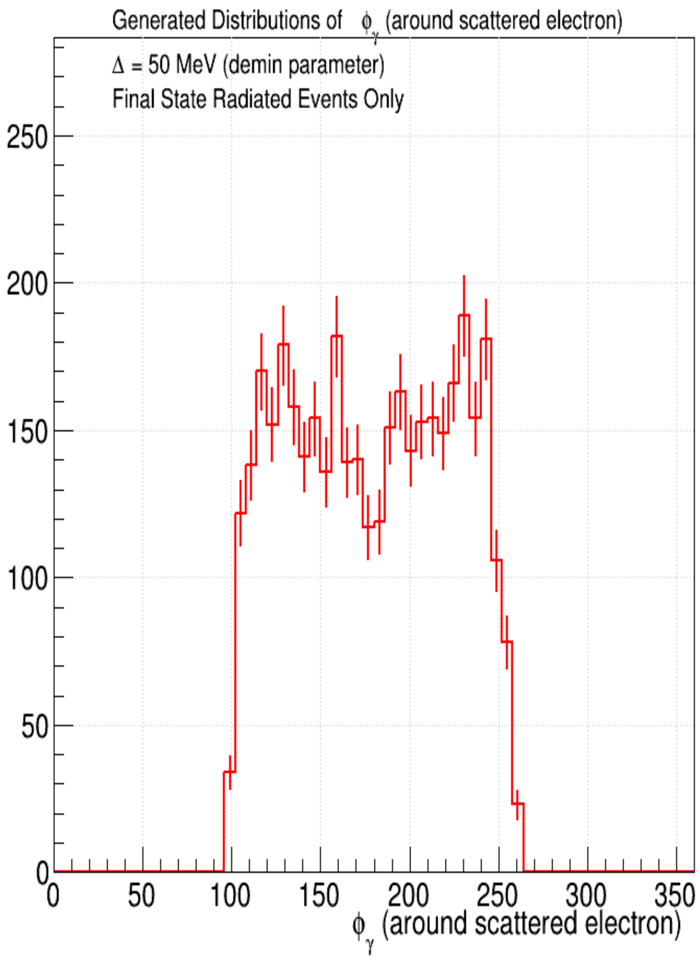
Jefferson Lab clas

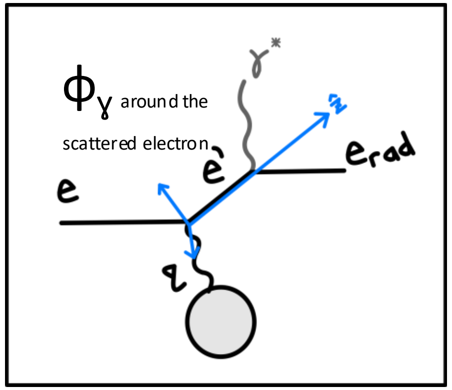
Issue 5: FSR ϕ_γ Relative to Beam and Scattered Electron



Looks Correct: i.e., ϕ_γ is generated evenly in the lab frame





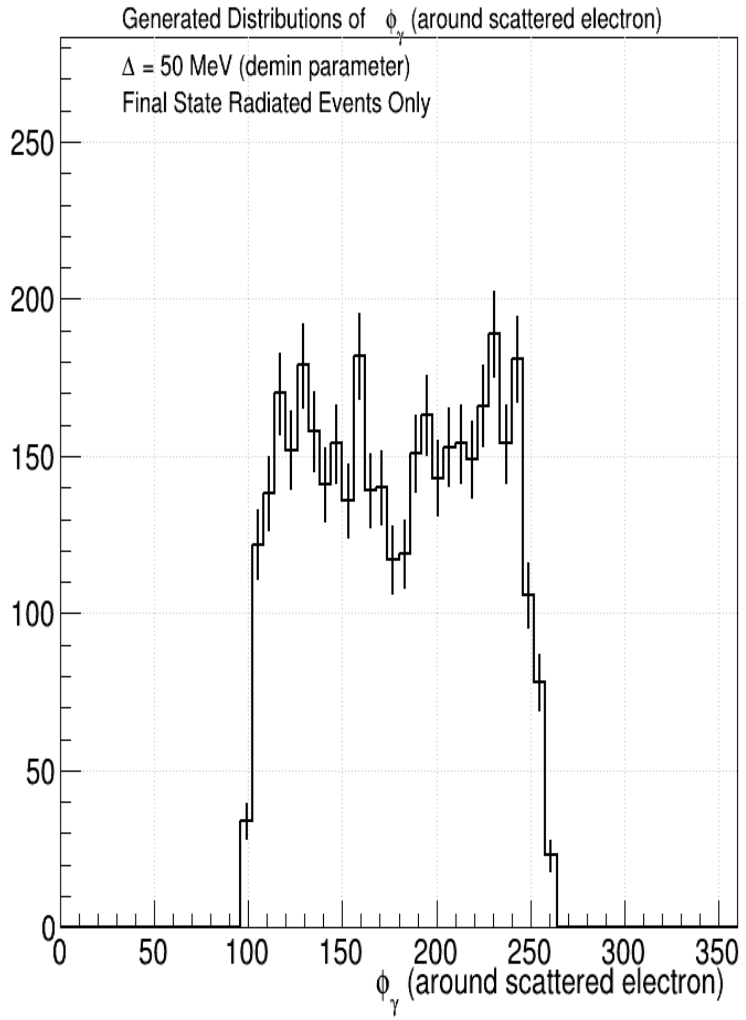


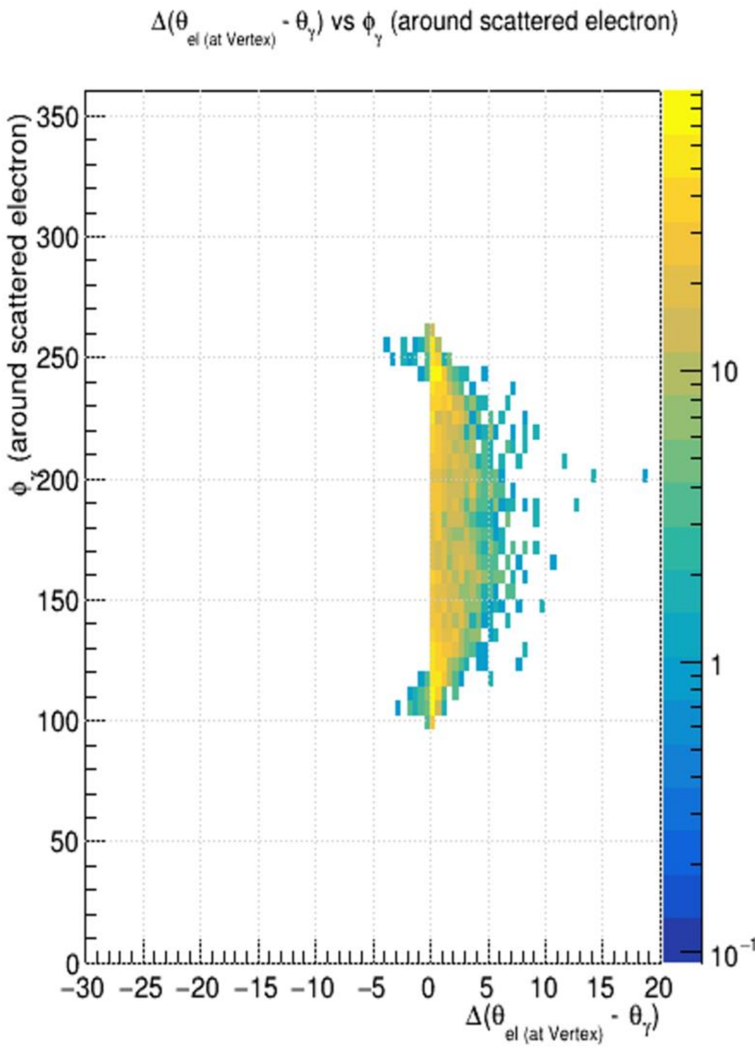
Looks Incorrect: i.e., ϕ_γ is expected to be flat around the scattered electron as well

Could not find a fix via different rotations of the ϕ angles, and values of Δ > 18 MeV do not change this behavior

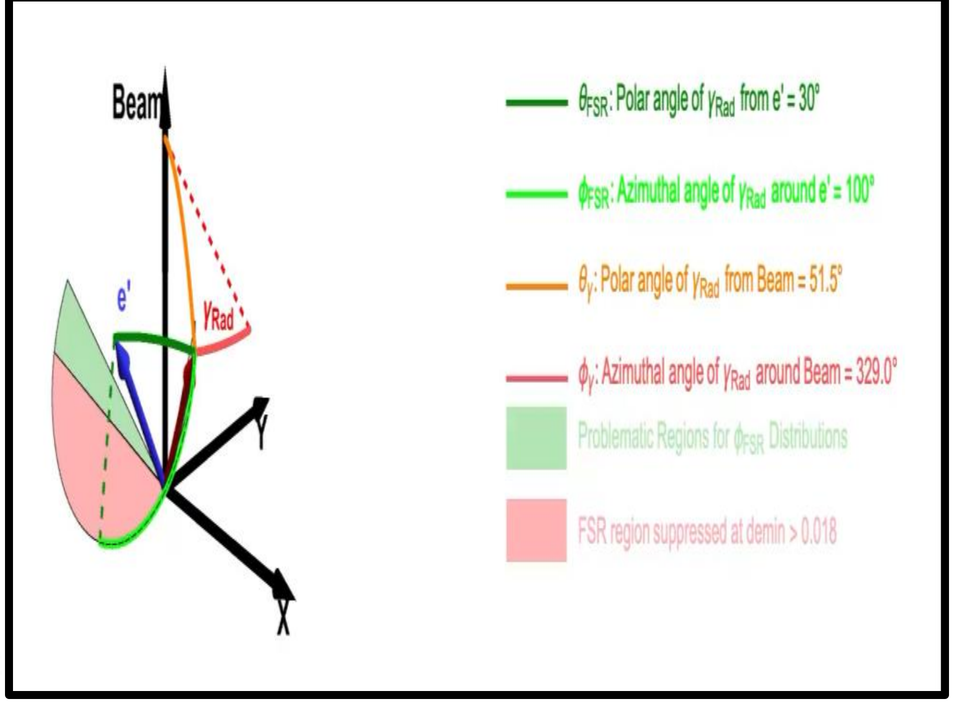
Includes FSR Events Only

$\Delta\theta_{e\text{-}\gamma}$ vs ϕ_γ of FSR Photons Around the Scattered Electron





Larger value of demin suppresses $\theta_{el} < \theta_\gamma$ in the Lab Frame
(i.e., negative values on the 2D x-axis)



(Attempted) Replication of the Plots from RADGEN

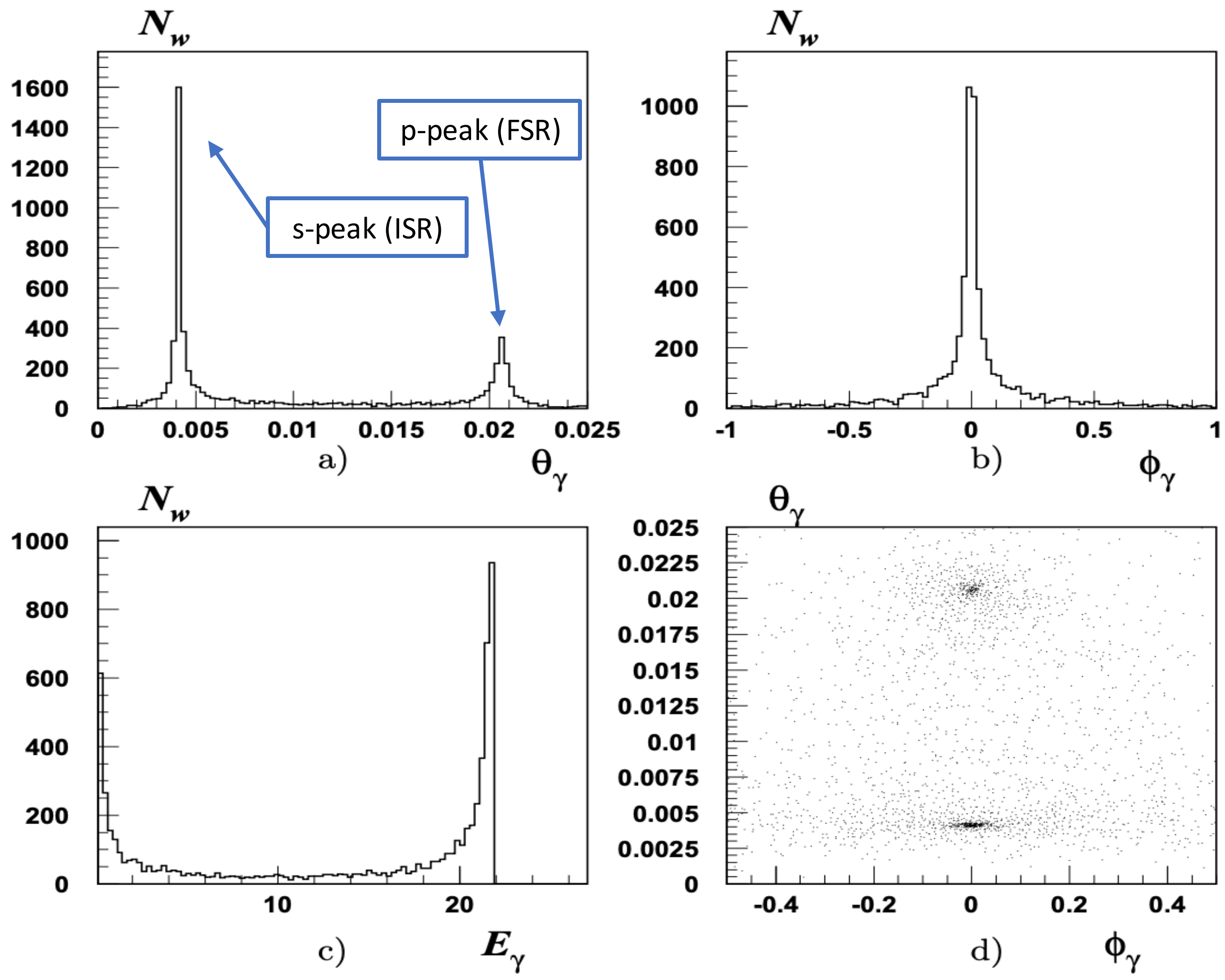


Figure 3: *The distribution of the radiation angles θ_γ a) and ϕ_γ b) and of the energy c) of the radiated photon for $x = 0.1$ and $y = 0.8$. The two-dimensional distribution d) shows θ_γ vs ϕ_γ.*

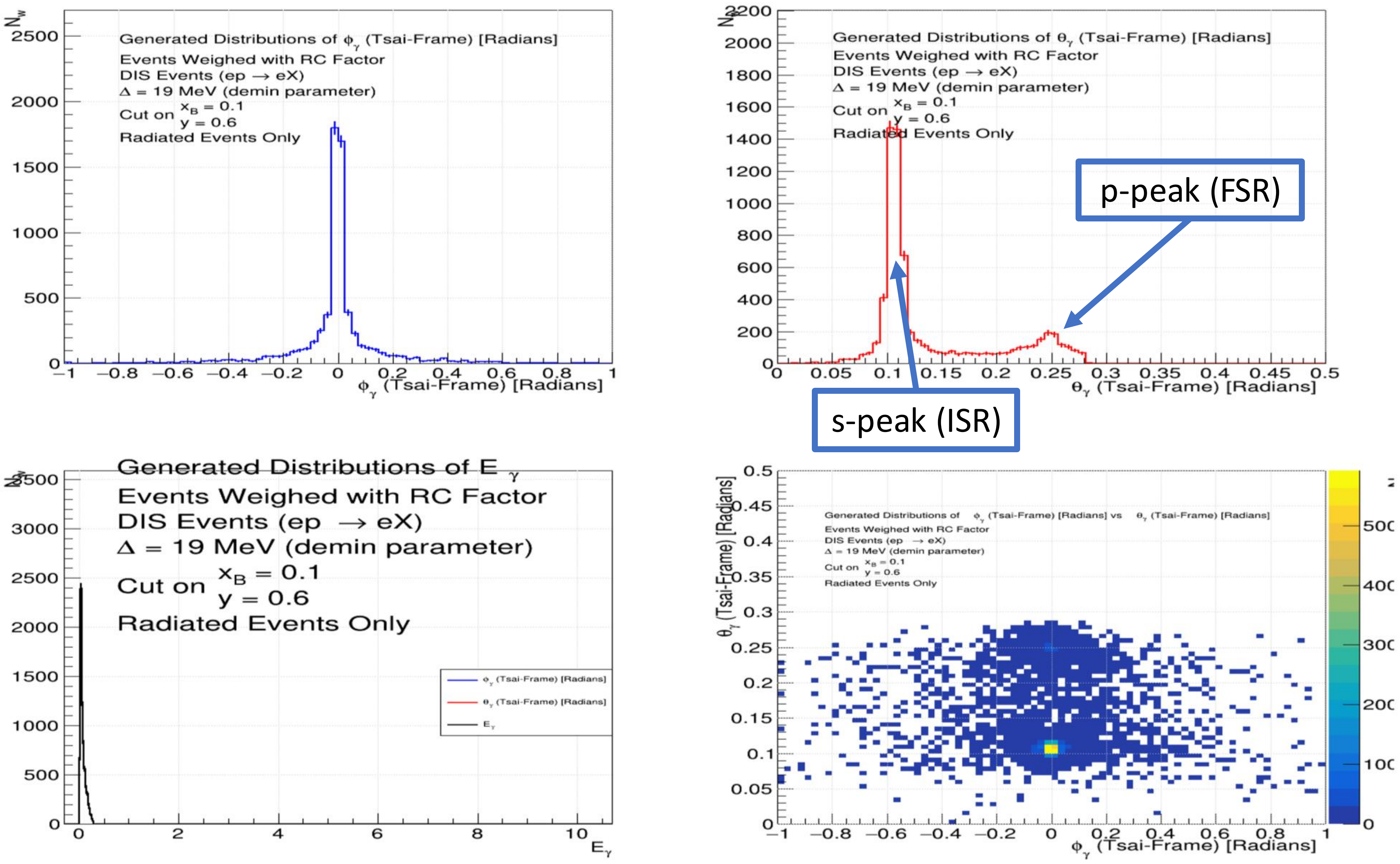
Plots from the RADGEN Paper

https://arxiv.org/pdf/hep-ph/9906408

- Shows the kinematics of the radiated photon (ɣ)
  - In Center-of-Mass (Tsai) frame
- Was based on a different experimental configuration and cuts
  - **My results on the next page will come from the DIS events but other kinematics may not be identical**
- Used only DIS events (ep→e'(X)) in these slides

*Angles in these plots shown in radians

(Attempted) Replication of the Plots from RADGEN



Generated with:

- Beam Energy = 10.6 GeV
- $0.085 < x_B < 0.115$ ($x_B = 0.1$)
- $0.585 < y < 0.615$ ($y = 0.6$)
- $0.98 < Q^2 < 1.42$
- Used fixed Hydrogen Target
  - RADGEN paper might have used ^3He

- **<u>demin = 0.019 (19 MeV)</u>**

Current Status of Monte Carlo

- **ISR Effects have been added to the event generator**
  - The beam energy can now vary event-by-event based on the photon given by RADGEN
  - Energy and Momentum are being preserved (i.e., implementation is working correctly)
- **FSR Effects have been added to the final kinematic calculations**
  - These effects do not impact the rest of the event generation process, so they can be accounted for at any time before the events get processed by the detector simulation
- **Modified RADGEN's photon energy threshold (demin) based on more detailed investigation**
  - Will use a value of demin > 20 MeV
- **Things that still need to be fixed:**
  - The radiated photon's angles as measured around the scattered electron **(problem with the FSR events)**
  - Could it be related to some type of suppression of FSR events or some type of bug in the code?

Thank you

Questions?

Acknowledgments and Thanks

- Contributions made by other members of the CLAS Collaboration and researchers at Argonne National Lab
- This work is supported by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics under contract number DE-AC02-06CH11357

Backup Slides

Configuration of Pythia with RADGEN

- Mainly used the configurations given by the ‘claspyth’ steering file: input.10.6gev.with-comments
  - Changes to the configurations detailed below

Ranges/Values used for generating Q^2, y, x_B, W^2, and the beam/target proton energies:

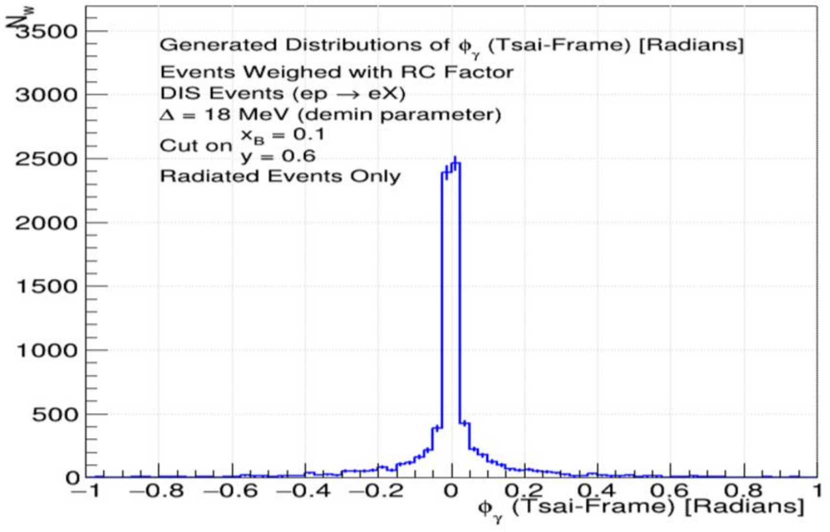
- **Q^2:** 0.85 to 20.0 GeV^2 (changed from the claspyth default of 0.00001 to 15.0 GeV^2)
- **y:** 0.15 to 0.95
- **x_B:** 0.05 to 0.95 (from clasdis)
- **W^2:** 4 GeV^2 (min)
- **Beam Energy:** 10.6 GeV
- **Proton Target Energy:** 0 GeV (at rest)

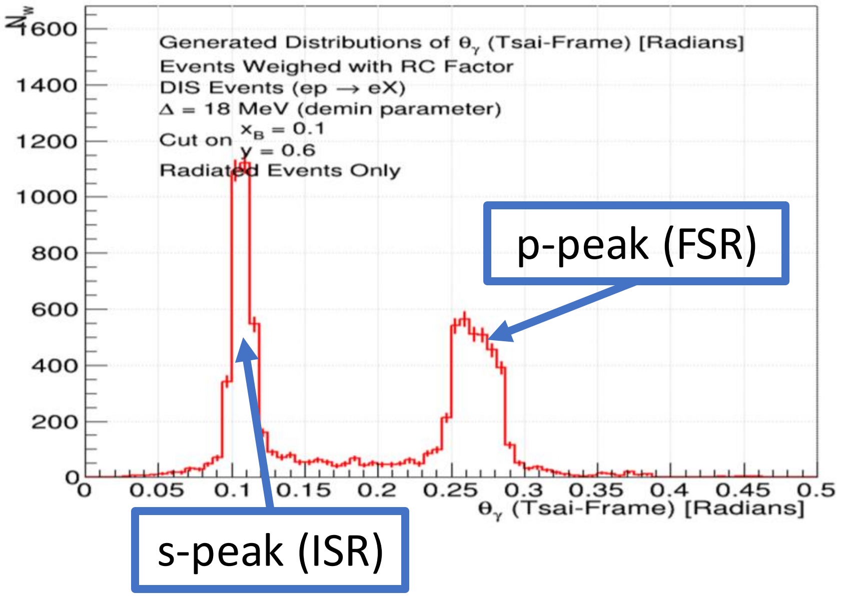
Additional changes added to better match clasdis:

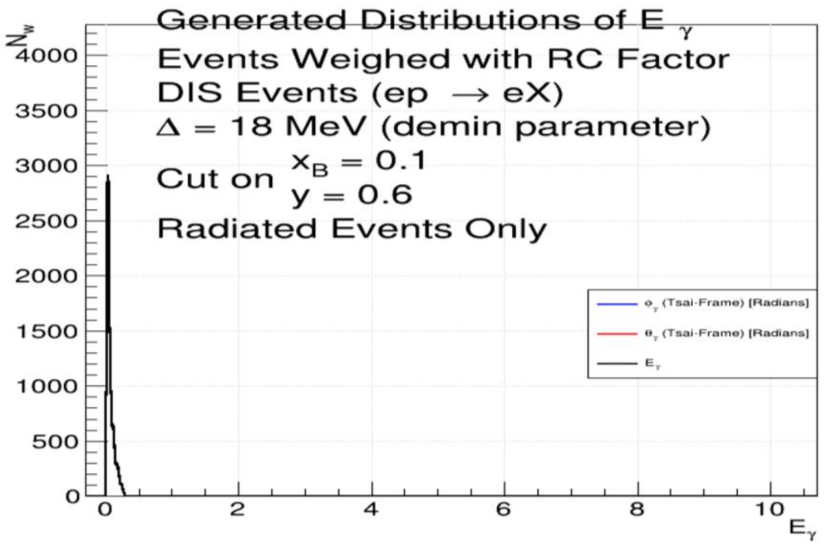
- **Changed the F2-Model/R-Parametrization from “ALLM,1990” to “F2PY,1998”**
- **PARJ(33) = 0.8 changed to 0.3**
- **PARJ(41) = 0.3 changed to 1.2**

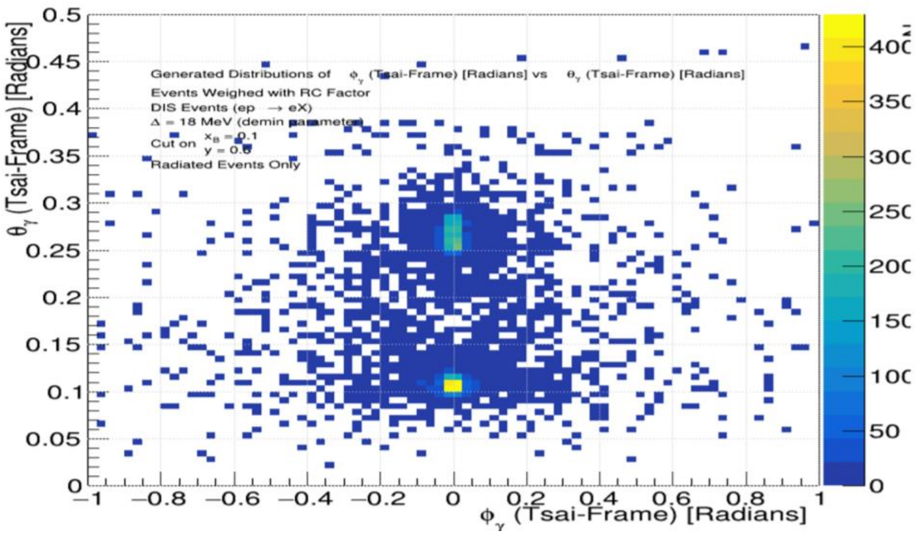
***PARJ(33)** defines the energy threshold stopping parton fragmentation and forming two hadrons.
***PARJ(41)** gives the ‘*a*’ parameter of the symmetric Lund fragmentation function

(Attempted) Replication of the Plots from RADGEN









Generated with:

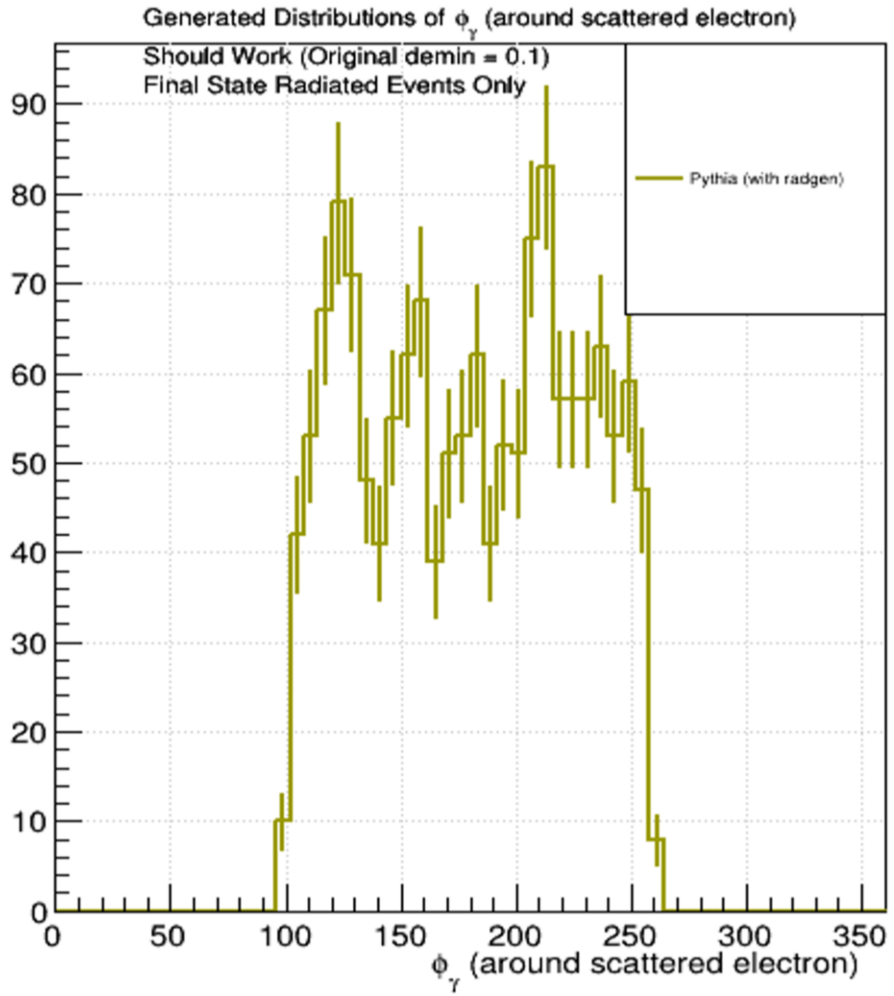
- Beam Energy = 10.6 GeV
- $0.085 < x_B < 0.115$ ($x_B = 0.1$)
- $0.585 < y < 0.615$ ($y = 0.6$)
- $0.98 < Q^2 < 1.42$
- Used fixed Hydrogen Target
  - RADGEN paper might have used ^3He
- **demin = 0.018 (18 MeV)**

- p-peak is heavily suppressed above this demin value
  - Fewer total events in this peak and the range is smaller (demin = 0.019 cuts off at about $\theta_{\gamma\text{-Tsai}} = 0.28$ rad)

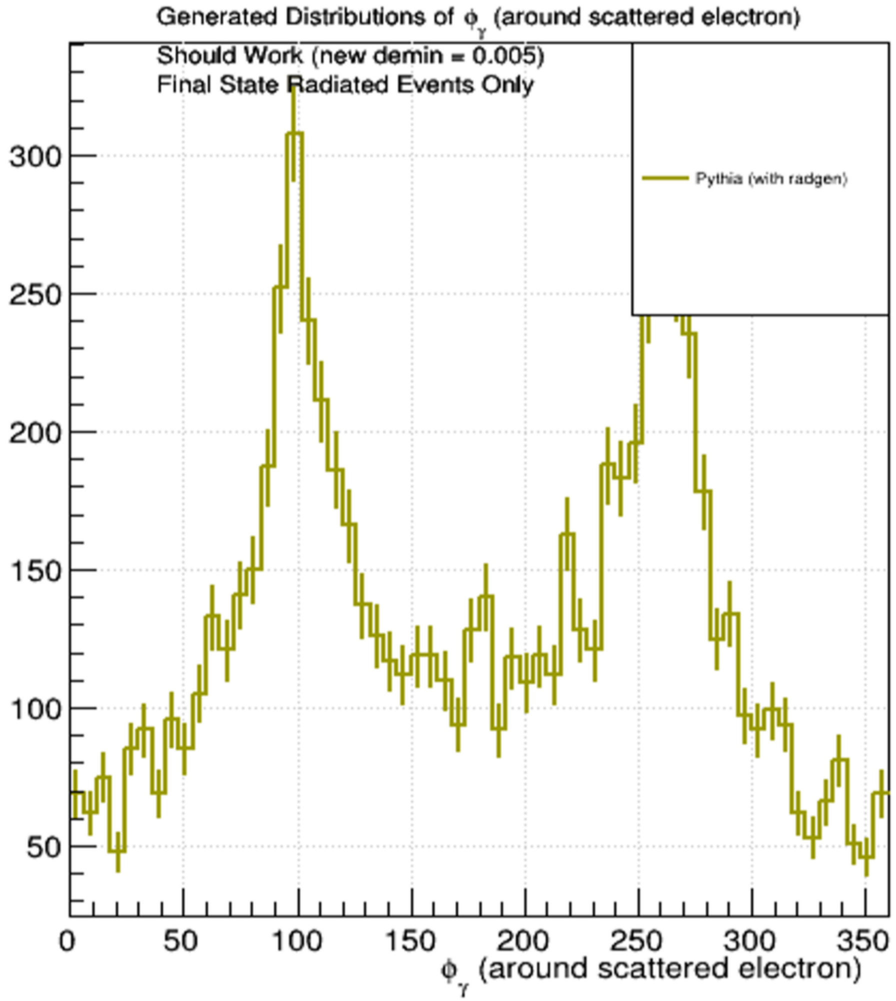
UCONN | UNIVERSITY OF CONNECTICUT Argonne NATIONAL LABORATORY U.S. DEPARTMENT OF ENERGY Office of Science Jefferson Lab clas

ϕ_γ of FSR Photons Around the Scattered Electron

Larger (Original) demin value (100 MeV)

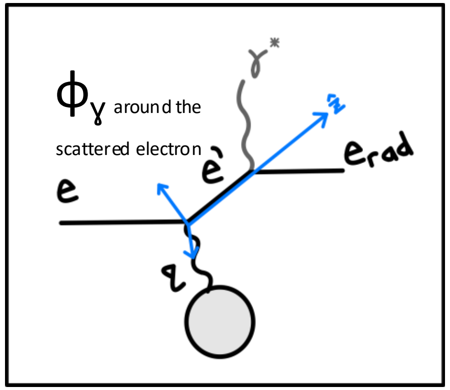


Smaller (New) demin value (5 MeV)

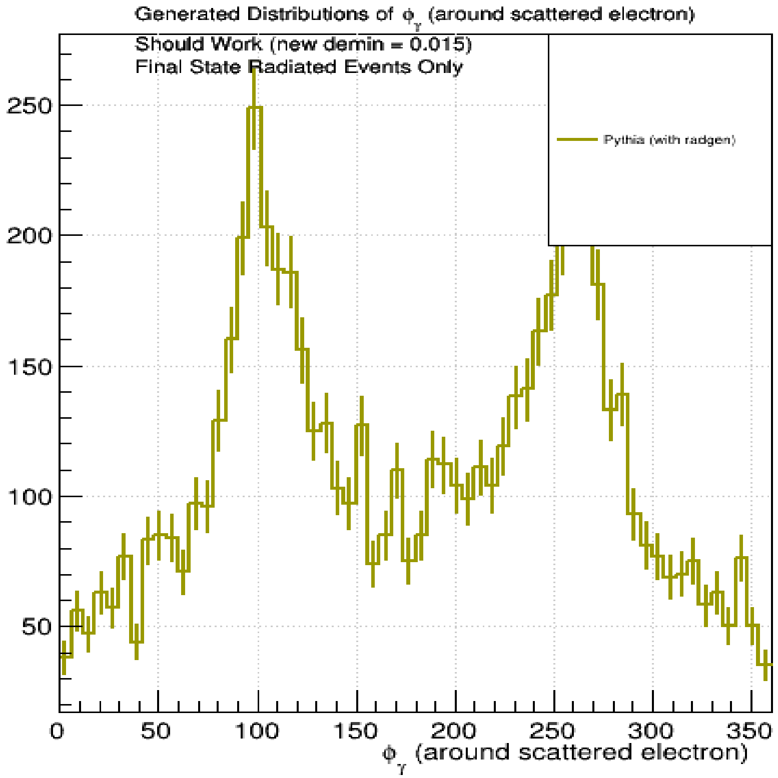


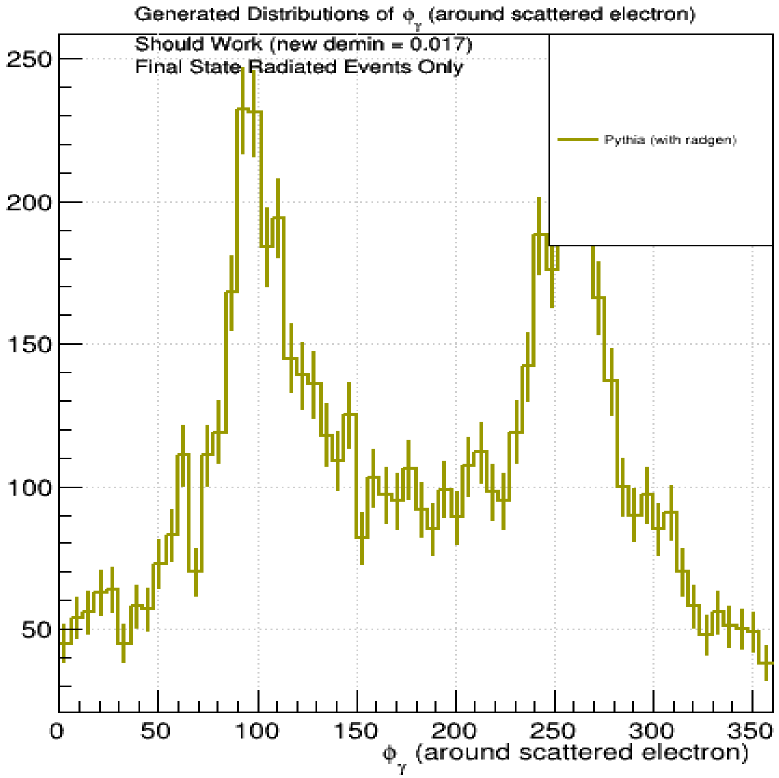
Changes from demin = 100 MeV to demin = 5 MeV

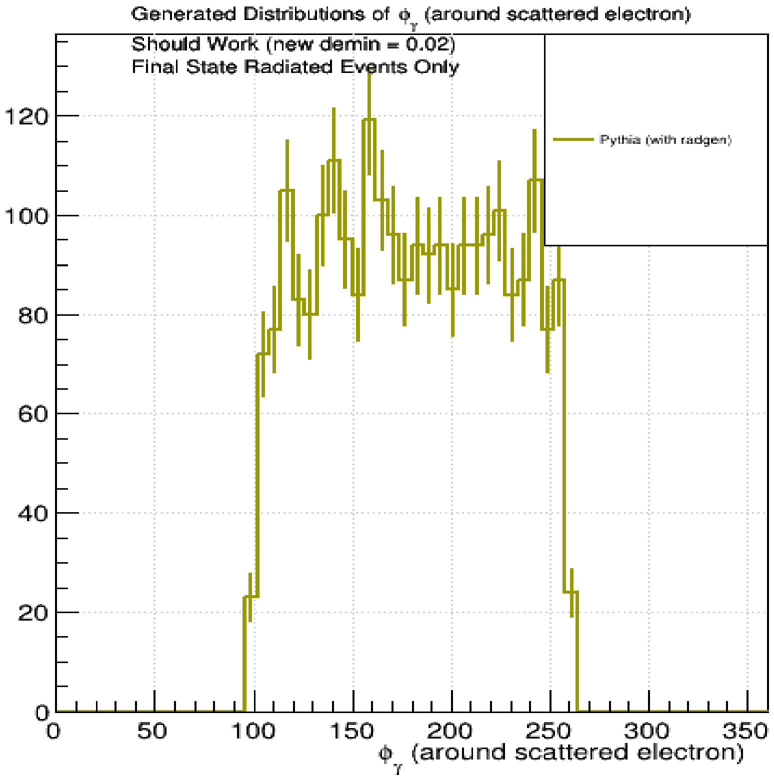
Both still look wrong, but in very different ways

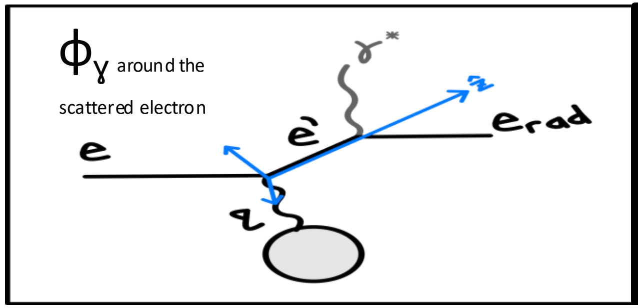


ϕ_γ of FSR Photons Around the Scattered Electron







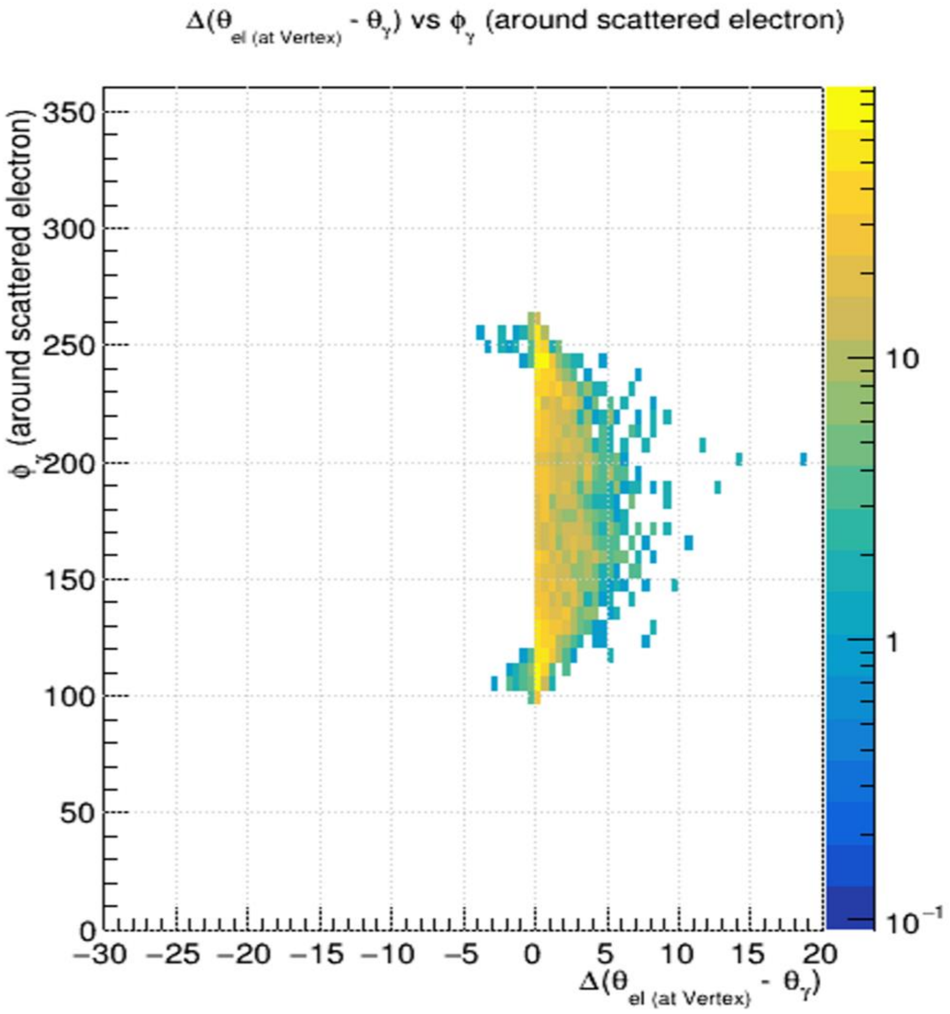


Confirmation that demin change is not gradual

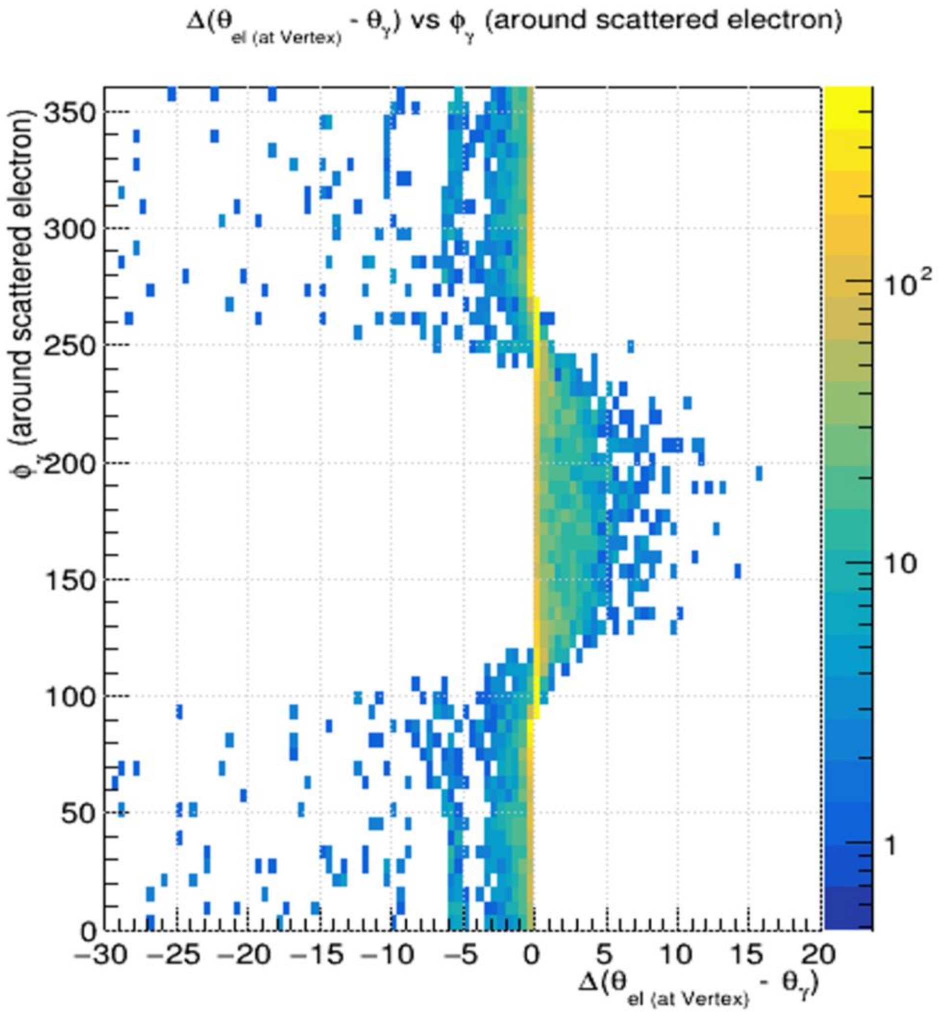
Plots shown from left to right have demin = 15 MeV, 17 MeV, and then 20 MeV

$\Delta\theta_{e\text{-}\gamma}$ vs ϕ_γ of FSR Photons Around the Scattered Electron

Larger demin value (50 MeV)



Smaller demin value (5 MeV)

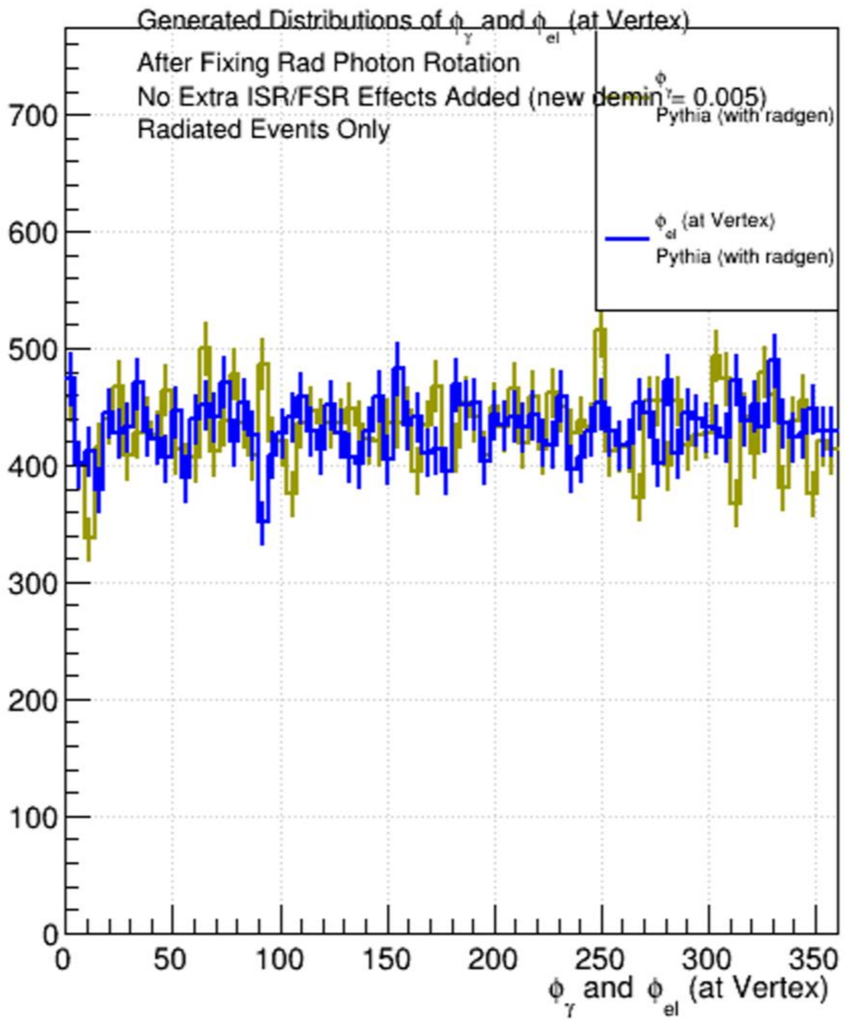


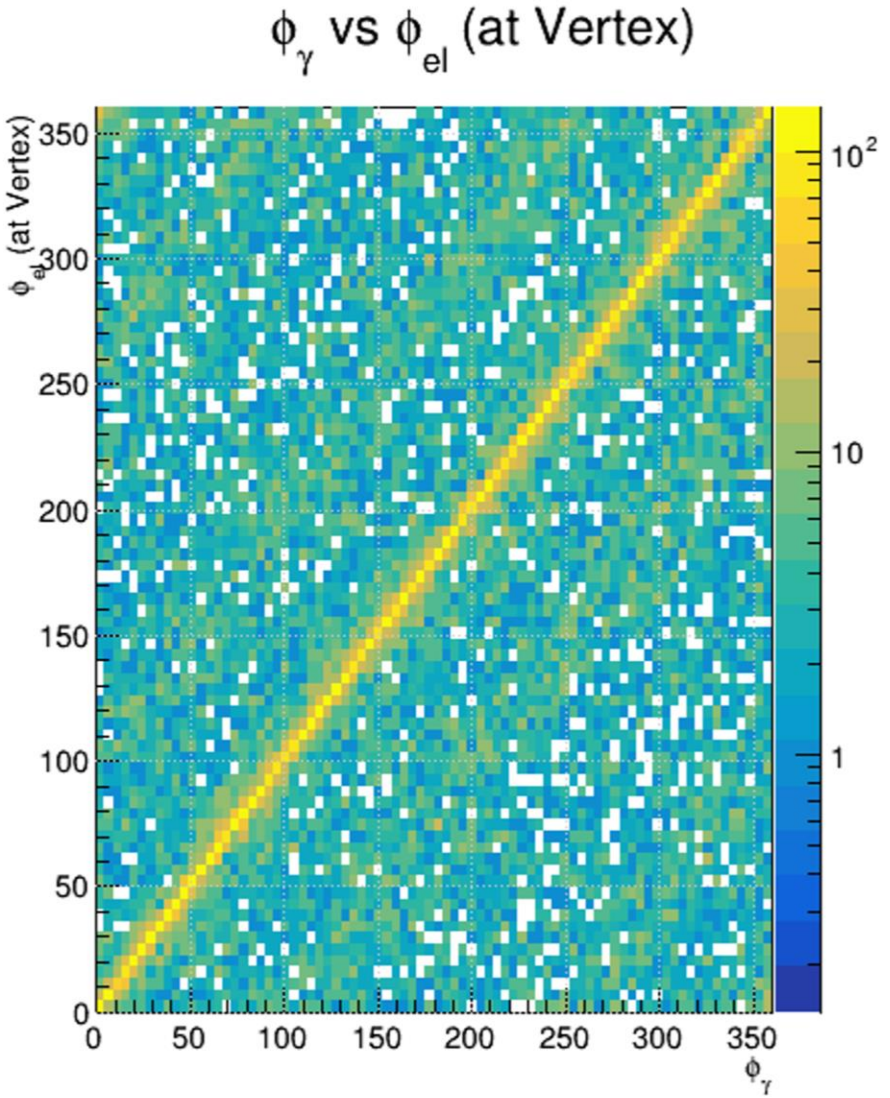
Larger value of demin suppresses $\theta_{el} < \theta_\gamma$ in the Lab Frame
(i.e., negative values on the x-axis shown here)

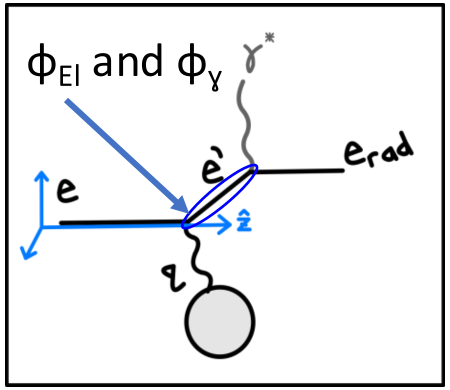
Result of soft photons?

Will be investigating how RADGEN generates θ_γ to address this
(looking for a reason to always generate the photon between the incoming/outgoing electrons, or a fix to remove this restriction)

ϕ_{El} vs ϕ_γ in Lab Frame for All Radiated Events (New-After Fixing Photon Rotation)



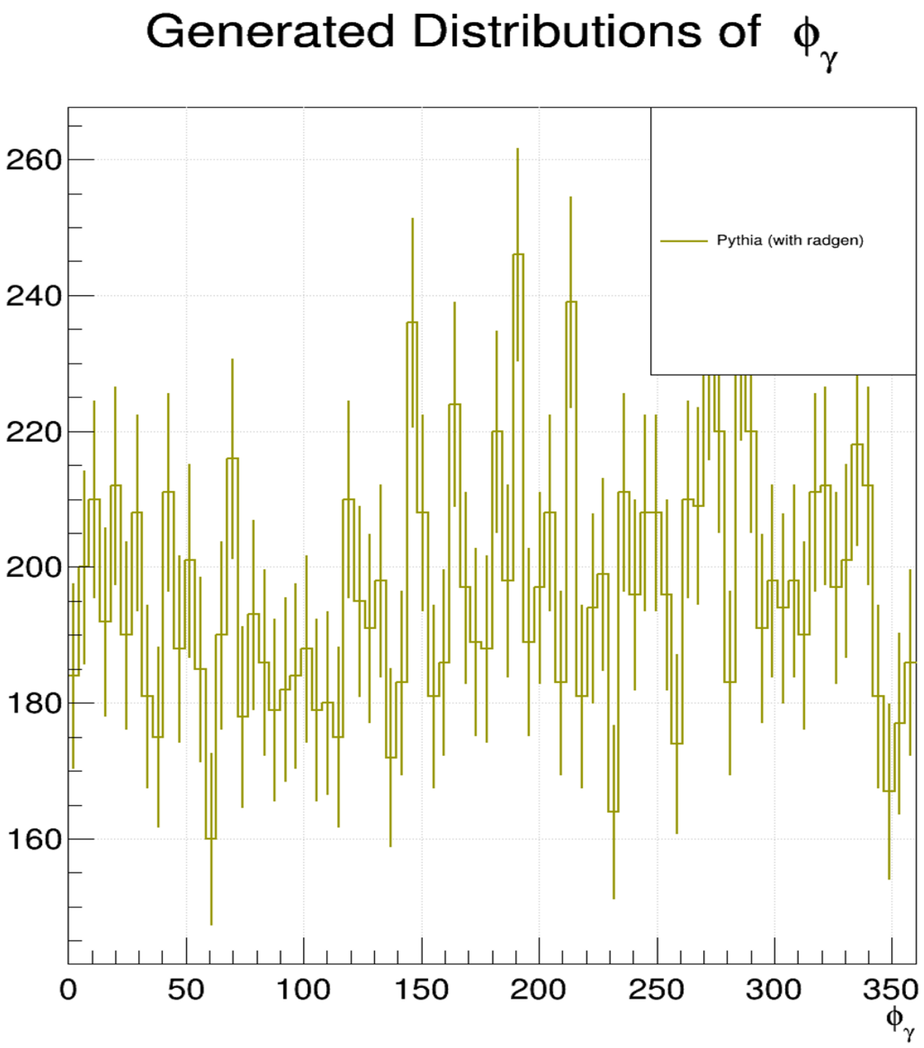


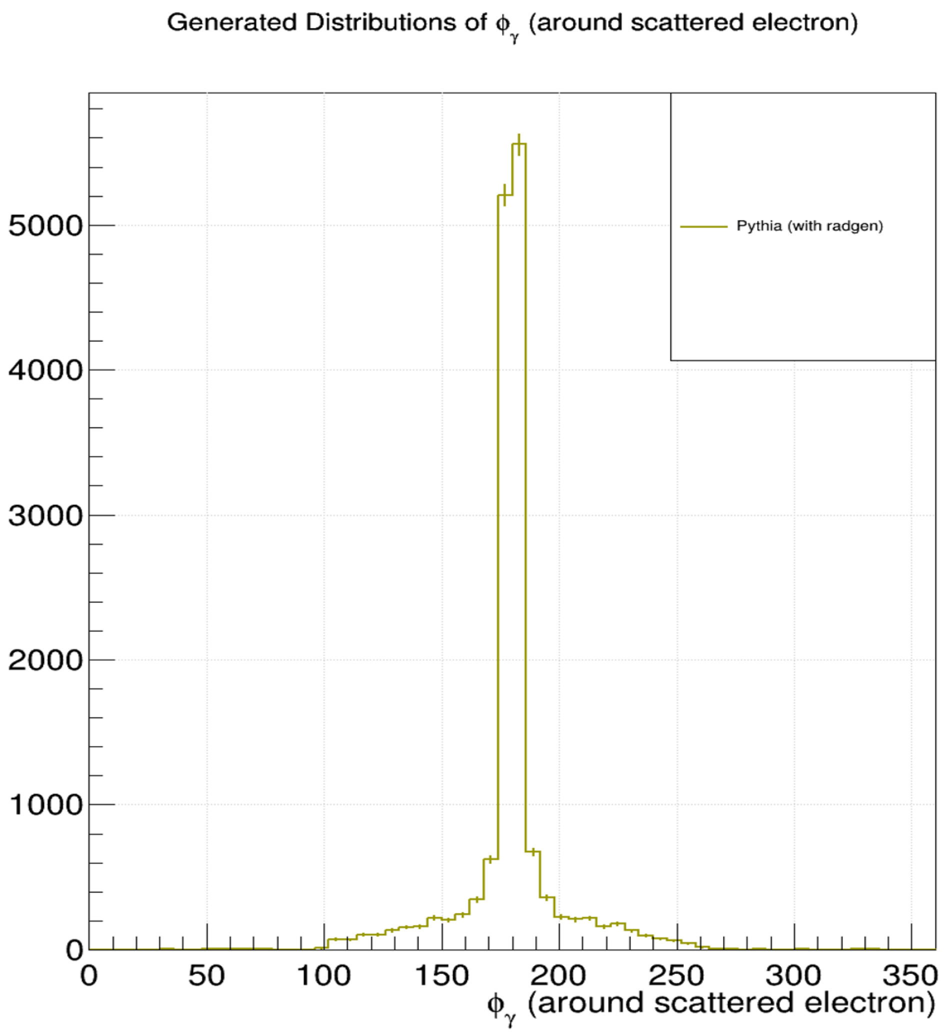


Both angles are measured from around the incoming beam

Includes <u>All</u> <u>Radiated</u> Events

ϕ_γ Relative to Beam and Scattered Electron

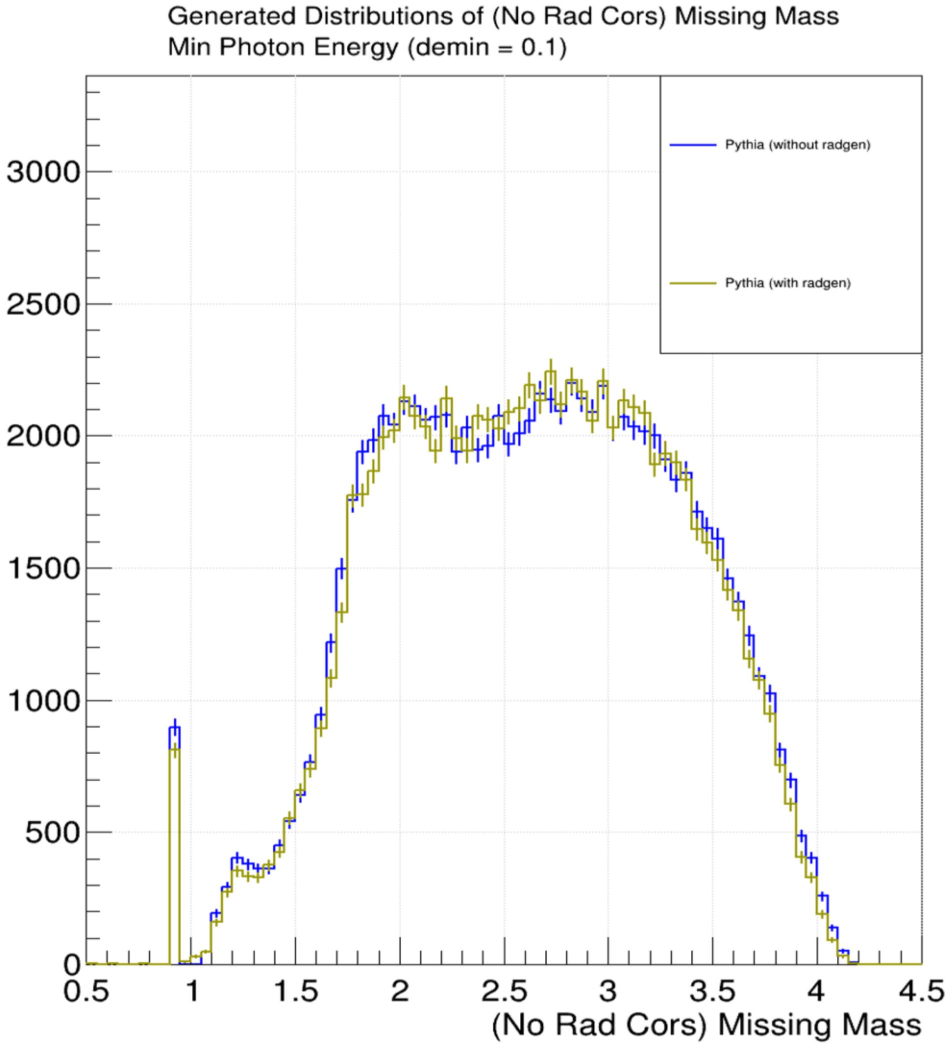




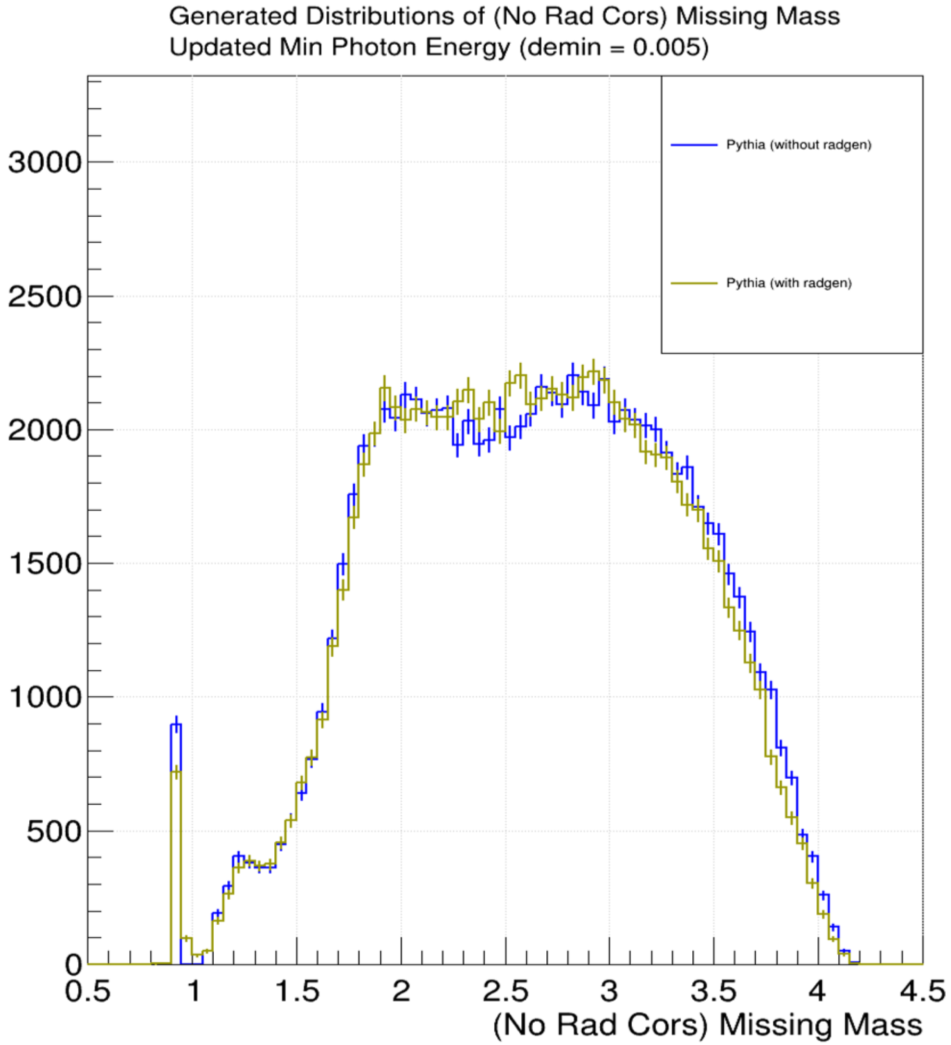
Includes All Radiated Events

Missing Mass without Radiative Corrections

Before Modifications to RADGEN



After Modifications to RADGEN



Missing Mass as Reported with <u>Uncorrected Radiated Kinematics</u>

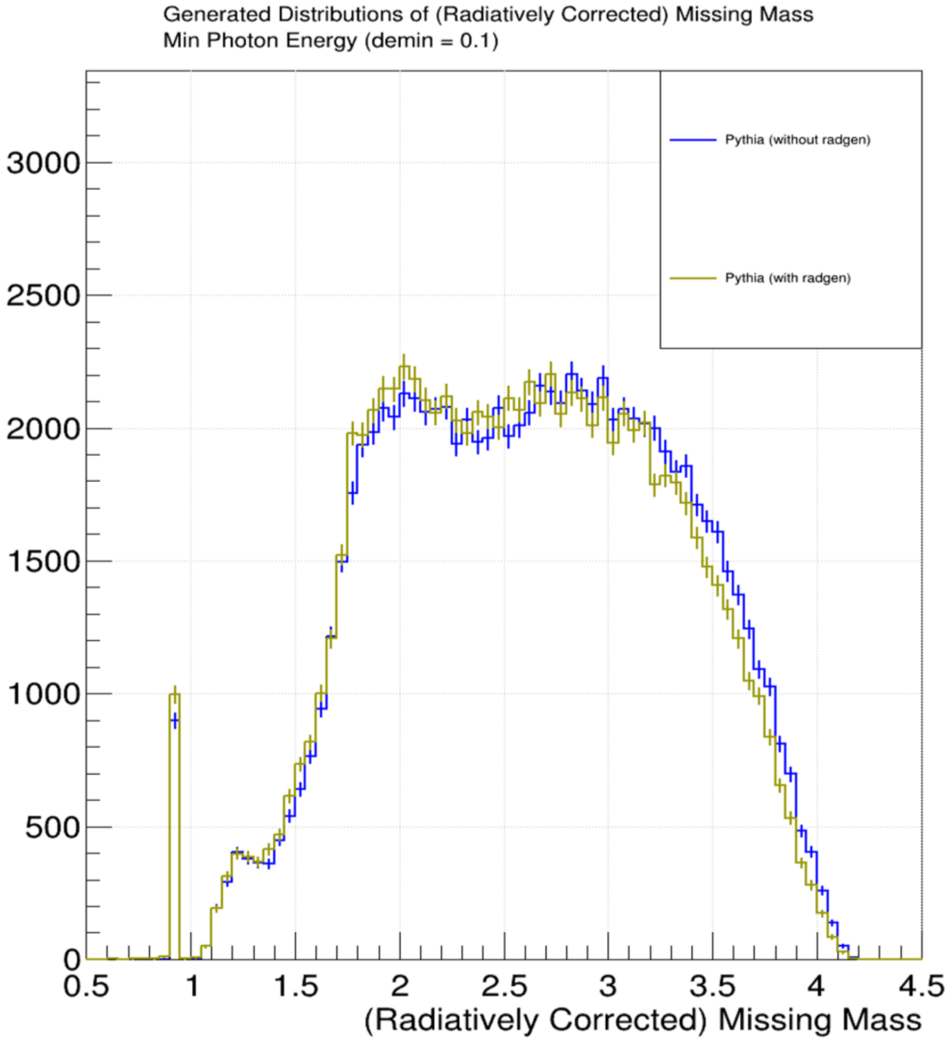
Includes Radiated and Non-Radiated Events

Blue line is without RADGEN

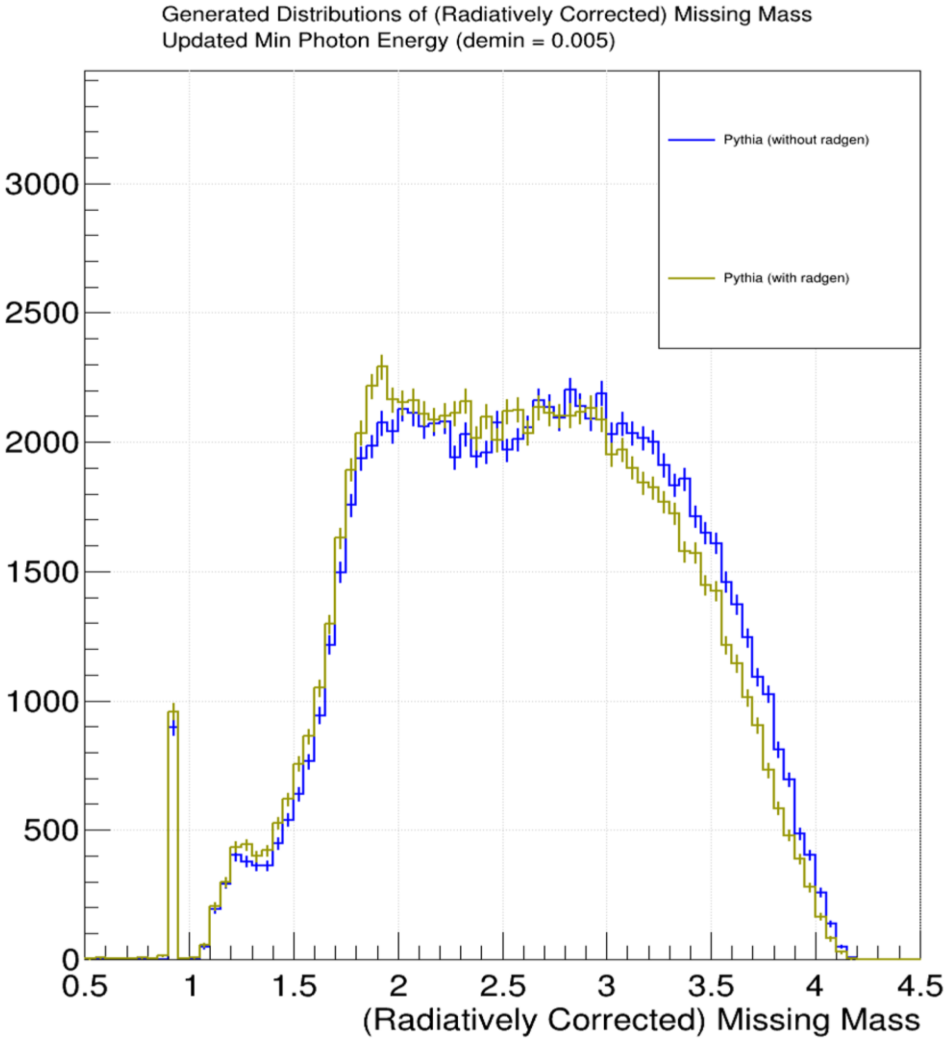
Bin Size = 50 MeV/bin

Missing Mass with Radiative Corrections

Before Modifications to RADGEN



After Modifications to RADGEN



Missing Mass as Reported with <u>Corrected Radiated Kinematics</u>

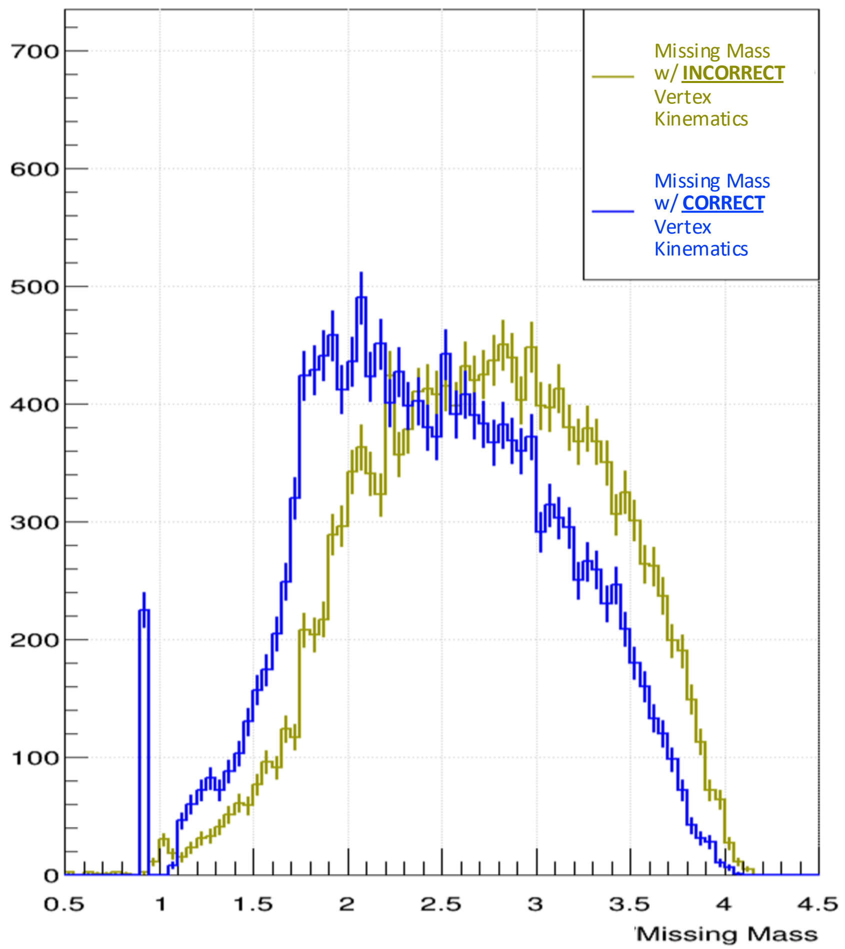
Includes Radiated and Non-Radiated Events

Blue line is without RADGEN

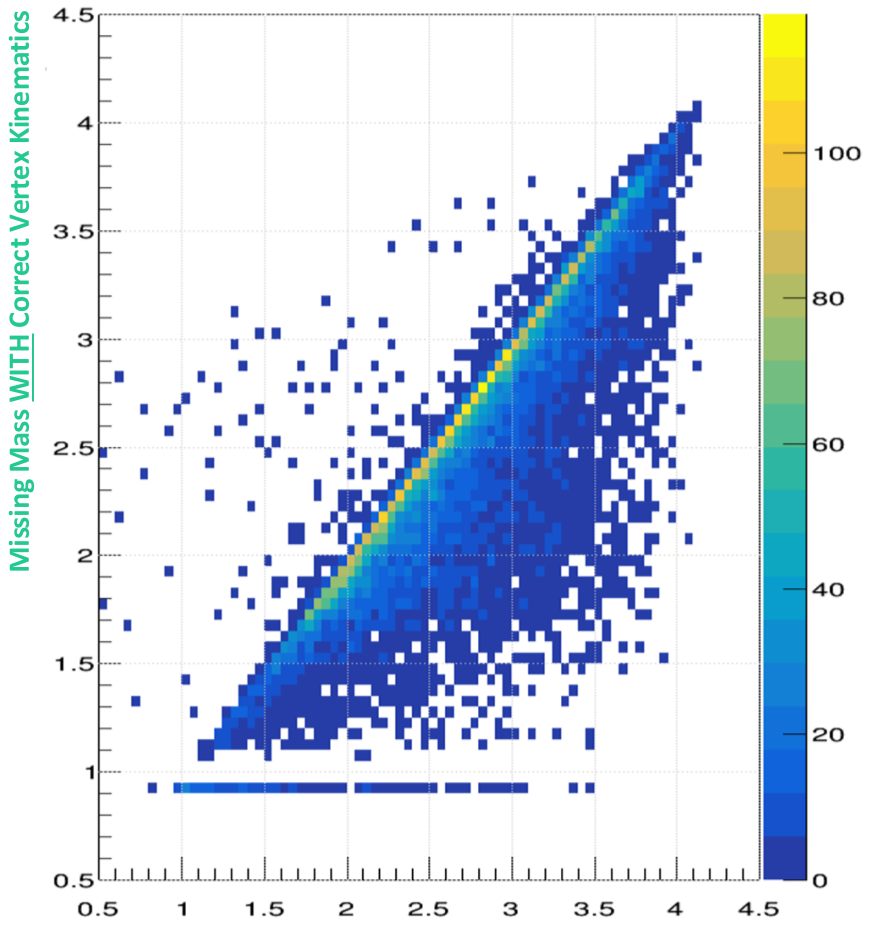
Bin Size = 50 MeV/bin

Missing Mass with/without Vertex Calculations

Generated Distributions of Missing Mass with/without Correct Vertex Kinematics



Missing Mass WITH Correct Vertex Kinematics vs Missing Mass WITHOUT Correct Vertex Kinematics



Includes Radiated Events Only

Energy/Momentum Conservation Details

What I did to verify that Pythia was only reporting the particles' Vertex Kinematics

Checking Energy and Momentum Conservation

Done with calculations of the total <u>initial</u> and <u>final</u> state particles' momenta and energies

- p_x, p_y, p_z, and E are all handled separately (full conservation of each component of the particles' 4-vectors)

Initial State Particles:
- The beam electron and target proton
- Make up the 1st 2 (non-header) lines of each event

Final State Particles:
- Status Code = 1 corresponds to the stable final-state particles (that will not undergo further decays)

Radiated Photons:
- Status Code = 55-57 corresponds to the different types of radiated photons (i.e., ISR, FSR, unidentified)
- **NOT INCLUDED IN CONSERVATION CALCULATIONS**

Total (Missing) Momentum/Energy:
- Absolute differences between the initial and final state totals
- **Energy and Momentum are conserved if these values are equal to 0**

Conservation Check:
- Threshold for conservation check is set to the total differences being less than $2.1*10^{-6}$ GeV (accounts for rounding errors at the maximum reported precision)

```python
def check_energy_momentum_conservation(event_in, print_lines=False):
    # Check energy and momentum conservation for a given event from TXT files
    initial_px, initial_py, initial_pz, initial_energy = 0.0, 0.0, 0.0, 0.0
    final_px,   final_py,   final_pz,   final_energy   = 0.0, 0.0, 0.0, 0.0
    for ii, entry in enumerate(event_in):
        if(ii > 0): # else is header
            Line_Index, status, PID, Parent, FirstDaughter, Last_Daughter, px, py, pz, Energy, Mass, Xvertex, Yvertex, Zvertex = entry.split()
            px, py, pz, Energy = map(float, [px, py, pz, Energy])
            if(str(Line_Index) in ['1', '2']):
                # First two particles as initial values
                initial_px     += px
                initial_py     += py
                initial_pz     += pz
                initial_energy += Energy
                if(print_lines):
                    print(f"{color.GREEN}{entry}{color.END}")
            elif(str(status) in ['1']):
                # Sum over final state particles
                final_px       += px
                final_py       += py
                final_pz       += pz
                final_energy   += Energy
                if(print_lines):
                    print(f"{color.CYAN}{entry}{color.END}")
            elif(str(status) in ['55', '56', '57']):
                if(print_lines):
                    print(f"{color.BLUE}{entry}{color.END}")
                else:
                    print(f"{color.BLUE}RADGEN EVENT (status = {status}){color.END}")
        # elif(print_lines): # Print Header
        #     print(f"{color.BOLD}{entry}{color.END}")
    total_px     = abs(initial_px     - final_px)
    total_py     = abs(initial_py     - final_py)
    total_pz     = abs(initial_pz     - final_pz)
    total_energy = abs(initial_energy - final_energy)
    X_Conserve = total_px     < 2.1e-6
    Y_Conserve = total_py     < 2.1e-6
    Z_Conserve = total_pz     < 2.1e-6
    E_Conserve = total_energy < 2.1e-6
```

Checking Energy and Momentum Conservation

Done with calculations of the total <u>initial</u> and <u>final</u> state particles' momenta and energies

- p_x, p_y, p_z, and E are all handled separately (full conservation of each component of the particles' 4-vectors)

```
================ Checking Event 498:
 1     21      11     0     3     4      0.000000    0.000000   -10.600000   10.600000
 2     21    2212     0     5     0     -0.000000    0.000000    -0.000000    0.938270
 Total Initial:                          0.000000    0.000000   -10.600000   11.538270
11      1      11     3     0     0     -0.649710    0.453333    -7.277024    7.320022
17      1    2212    14     0     0      0.273942   -0.065846    -1.348208    1.666552
18      1     211    15     0     0      0.397148   -0.339520    -0.738504    0.915351
20      1    -211    16     0     0     -0.305307   -0.002163    -0.365014    0.495916
22      1      22    19     0     0     -0.009405    0.051313    -0.100498    0.113232
23      1      22    19     0     0     -0.139667   -0.042117    -0.438938    0.462545
24      1      22    21     0     0      0.054423   -0.056035    -0.050844    0.093203
25      1      22    21     0     0      0.378577    0.001034    -0.280970    0.471451
26     55      22     1     0     0      0.002683    0.001624    -1.502146    1.502149
 Total Final:                            0.000001   -0.000001   -10.600000   11.538272
Passed Energy/Momentum Conservation Check
================ Event 498 Checked ================

================ Checking Event 499:
 1     21      11     0     3     4      0.000000    0.000000   -10.600000   10.600000
 2     21    2212     0     5     0     -0.000000    0.000000    -0.000000    0.938270
 Total Initial:                          0.000000    0.000000   -10.600000   11.538270
11      1      11     3     0     0     -0.580796   -0.115849    -2.320392    2.394778
19      1    -211    15     0     0      0.090067   -0.004205    -0.793537    0.810747
20      1     211    15     0     0      0.039992   -0.142813    -2.402285    2.410902
22      1     211    16     0     0      0.174444   -0.033037    -0.174092    0.285150
24      1    -211    17     0     0     -0.403250   -0.098386    -0.806265    0.917514
26      1    2212    18     0     0      0.188380   -0.116667    -1.162335    1.510123
28      1      22    21     0     0      0.008262    0.127079    -0.994173    1.006140
29      1      22    21     0     0      0.106482    0.226420    -0.822693    0.859900
30      1      22    23     0     0      0.040109   -0.008651    -0.028725    0.050087
31      1      22    23     0     0     -0.060567   -0.088668    -0.305704    0.324015
32      1      22    25     0     0      0.267620    0.135342    -0.520024    0.600302
33      1      22    25     0     0      0.030261   -0.031033    -0.078500    0.089672
34      1      22    27     0     0      0.012309    0.074360    -0.187911    0.202464
35      1      22    27     0     0      0.006686    0.076109    -0.003364    0.076476
 Total Final:                           -0.000001    0.000001   -10.600000   11.538270
Passed Energy/Momentum Conservation Check
================ Event 499 Checked ================

================ Checking Event 500:
 1     21      11     0     3     4      0.000000    0.000000   -10.600000   10.600000
 2     21    2212     0     5     0     -0.000000    0.000000    -0.000000    0.938270
 Total Initial:                          0.000000    0.000000   -10.600000   11.538270
 8      1    -211     6     0     0      0.201771    0.006471    -2.026003    2.040813
10      1    2212     7     0     0     -0.149186    0.413114    -0.349224    1.093265
11      1      11     3     0     0      0.449717   -0.557558    -3.798376    3.865330
12      1     211     9     0     0     -0.195680    0.323825    -0.837115    0.929190
14      1      22    13     0     0     -0.219134   -0.055405    -2.134567    2.146501
15      1      22    13     0     0     -0.087488   -0.130447    -1.454717    1.463172
 Total Final:                            0.000000   -0.000000   -10.600002   11.538271
Passed Energy/Momentum Conservation Check
================ Event 500 Checked ================

3 out of a total of 500 Events failed the Energy/Momentum Conservation Checks.
```

Only 3 out of 500 (test) events failed (>0.6%)

```python
def check_energy_momentum_conservation(event_in, print_lines=False):
    # Check energy and momentum conservation for a given event from TXT files
    initial_px, initial_py, initial_pz, initial_energy = 0.0, 0.0, 0.0, 0.0
    final_px,   final_py,   final_pz,   final_energy   = 0.0, 0.0, 0.0, 0.0
    for ii, entry in enumerate(event_in):
        if(ii > 0): # else is header
            Line_Index, status, PID, Parent, FirstDaughter, Last_Daughter, px, py, pz, Energy, Mass, Xvertex, Yvertex,
            px, py, pz, Energy = map(float, [px, py, pz, Energy])
            if(str(Line_Index) in ['1', '2']):
                # First two particles as initial values
                initial_px     += px
                initial_py     += py
                initial_pz     += pz
                initial_energy += Energy
                if(print_lines):
                    print(f"{color.GREEN}{entry}{color.END}")
            elif(str(status) in ['1']):
                # Sum over final state particles
                final_px       += px
                final_py       += py
                final_pz       += pz
                final_energy   += Energy
                if(print_lines):
                    print(f"{color.CYAN}{entry}{color.END}")
            elif(str(status) in ['55', '56', '57']):
                if(print_lines):
                    print(f"{color.BLUE}{entry}{color.END}")
                else:
                    print(f"{color.BLUE}RADGEN EVENT (status = {status}){color.END}")
        # elif(print_lines): # Print Header
        #     print(f"{color.BOLD}{entry}{color.END}")
    total_px     = abs(initial_px     - final_px)
    total_py     = abs(initial_py     - final_py)
    total_pz     = abs(initial_pz     - final_pz)
    total_energy = abs(initial_energy - final_energy)
    X_Conserve = total_px     < 2.1e-6
    Y_Conserve = total_py     < 2.1e-6
    Z_Conserve = total_pz     < 2.1e-6
    E_Conserve = total_energy < 2.1e-6
```

Checking Energy and Momentum Conservation

Done with calculations of the total <u>initial</u> and <u>final</u> state particles' momenta and energies

- p_x, p_y, p_z, and E are all handled separately (full conservation of each component of the particles' 4-vectors)

```
================ Checking Event 111:
 1   21    11    0    3    4    0.000000    0.000000  -10.600000   10.600000
 2   21  2212    0    5    0   -0.000000    0.000000   -0.000000    0.938270
 Total Initial:                 0.000000    0.000000  -10.600000   11.538270
12    1    11    3    0    0    0.527139    0.619579   -2.835272    2.949665
22    1   211   18    0    0   -0.111373   -0.053646   -0.282621    0.338580
25    1  -211   24    0    0   -0.131009   -0.379574   -1.789416    1.839220
26    1   321   24    0    0    0.235852    0.666548   -1.628670    1.842857
28    1    22   23    0    0    0.011931   -0.073097   -0.183096    0.197509
29    1    22   23    0    0    0.126642   -0.133207   -1.224507    1.238225
32    1  2112   30    0    0   -0.529434   -0.353321   -1.679466    2.026952
34    1    22   31    0    0    0.019776   -0.011880   -0.023368    0.032837
35    1    22   31    0    0   -0.160208   -0.308363   -0.748129    0.824895
36    1    22   33    0    0   -0.037286    0.066123   -0.120112    0.142090
37    1    22   33    0    0    0.047971   -0.039161   -0.085342    0.105443
 Total Final:                   0.000001    0.000001  -10.599999   11.538273
Energy is not conservered
|E_final  -  E_initial| = | 11.538273 -  11.538270| = 0.000003
================ Event 111 Checked ================

================ Checking Event 148:
 1   21    11    0    3    4    0.000000    0.000000  -10.600000   10.600000
 2   21  2212    0    5    0   -0.000000    0.000000   -0.000000    0.938270
 Total Initial:                 0.000000    0.000000  -10.600000   11.538270
11    1    11    3    0    0   -0.066866   -0.521425   -2.244595    2.305333
16    1   211   14    0    0    0.354415    0.579926   -0.575478    0.901431
17    1  -211   14    0    0    0.314798    0.619261   -2.455616    2.555800
18    1  2212   14    0    0   -0.255657   -0.232859   -1.579715    1.869608
19    1  -211   15    0    0    0.001966   -0.040225   -0.349705    0.378675
20    1   211   15    0    0   -0.161460   -0.302576   -1.297263    1.349071
25    1    22   22    0    0   -0.067479    0.069992   -0.342946    0.356461
26    1    22   22    0    0   -0.144183   -0.046680   -0.374971    0.404440
27    1    22   23    0    0    0.053620   -0.096964   -0.326280    0.344581
28    1    22   23    0    0    0.001713   -0.011055   -0.559275    0.559386
29    1    22   24    0    0    0.000440   -0.056000   -0.313671    0.318631
30    1    11   24    0    0    0.084005    0.035924   -0.112970    0.118613
31    1   -11   24    0    0   -0.035309    0.002681   -0.067515    0.076240
 Total Final:                   0.000003   -0.000000  -10.600000   11.538270
Momentum in 'x' is not conservered
|px_final - px_initial| = |  0.000003 -   0.000000| = 0.000003
================ Event 148 Checked ================

================ Checking Event 425:
 1   21    11    0    3    4    0.000000    0.000000  -10.600000   10.600000
 2   21  2212    0    5    0   -0.000000    0.000000   -0.000000    0.938270
 Total Initial:                 0.000000    0.000000  -10.600000   11.538270
12    1    11    3    0    0   -0.170014    0.487886   -3.130902    3.173245
19    1  -211   17    0    0   -0.082911    0.146584   -0.423816    0.476929
24    1   321   18    0    0    0.149582   -0.021651   -0.532327    0.741523
27    1  -211   21    0    0   -0.088291    0.009671   -0.330720    0.369789
28    1    22   22    0    0   -0.053408   -0.068678   -0.017533    0.088750
29    1    22   22    0    0    0.066456   -0.150864   -0.117942    0.202698
30    1  2212   23    0    0    0.137229   -0.005485   -2.015129    2.227096
31    1   211   23    0    0   -0.177493    0.217480   -0.658569    0.729379
32    1    22   25    0    0    0.020949   -0.025634   -0.195790    0.198569
33    1    22   25    0    0   -0.087559   -0.352804   -0.918265    0.987597
35    1   211   34    0    0    0.046108    0.017231   -1.364328    1.372331
36    1  -211   34    0    0    0.239352   -0.253736   -0.894682    0.970364
 Total Final:                  -0.000000   -0.000000  -10.600003   11.538270
Momentum in 'z' is not conservered
|pz_final - pz_initial| = |-10.600003 - -10.600000| = 0.000003
================ Event 425 Checked ================
```

Failed Events from test

- Only off by 0.000003 GeV in most cases (likely still just due to rounding errors)
- These errors are not related to radiated photons/RADGEN (can happen to any event checked in this way)
- <u>**Conclusion**</u>: Energy and Momentum <u>**ARE**</u> conserved from the reported kinematics <u>**WITHOUT**</u> the radiated photons
  - This means that the incoming/outgoing electrons naturally are the vertex kinematics used to generate the other final state particles (i.e., either the beam is supposed to be $k_{rad} = e - \gamma$ or the scattered electron is supposed to be $k' = e'_{rad} + \gamma$)

Pythia+RADGEN Event Generation Details

How I modified the event generator to account of ISR/FSR effects

Coding Procedure for Implementing ISR/FSR Effects

1) Pythia generates random values for Q^2, y and $\phi_{e'}$

New Q^2 and y

New $\phi_{e'}$

```
C...Generate phase space point and check against cuts.
          LOOP=0
  121     LOOP=LOOP+1
          RadState=0
          call Reset_Beam_Vars()
          DO 131 I=1,2
              IF(MINT(140+I).NE.0) THEN
C...Pick x and Q2
                  MINT(199)=0
C...Modified by liang to make it also running for low energy++
                  geny=mcSet_YMin*(mcSet_YMax/mcSet_YMin)**PYR(0)
                  genQ2=mcSet_Q2Min*(mcSet_Q2Max/mcSet_Q2Min)**PYR(0)
                  gennu=geny*ebeamEnucl
                  geneprim = ebeamEnucl - gennu
                  genx = genQ2/(2*massp*gennu)
                  genW2 = massp**2.-genQ2+2*massp*gennu
C...Modified by liang to make it also running for low energy--
                  genphi=0.
C...Check to have sensible ranges for variables
                  minq2 = PMS(1) * geny**2. / (1.- geny)
                  if (genQ2.lt.minq2) then
                      ! WRITE(*,*)"genQ2.lt.minq2"
                      GOTO 121
                  endif
C...Check x and Q2 go toghether
                  if (genQ2.gt.(2.*gennu*massp)) then
                      ! WRITE(*,*)"genQ2.gt.(2.*gennu*massp)"
                      GOTO 121
                  endif
                  temp=(genQ2/(2.*ebeamEnucl*geneprim))-1.
                  if ((temp.le.1.00).and.(temp.ge.-1.0)) then
                      etheta = acos(temp)
                      genthe = sngl(etheta)
                      genphi=PARU(2)*PYR(0)
                      PHI(I)=genphi
                      ppt=tan(etheta)
                      ppx=ppt*cos(PHI(I))
                      ppy=ppt*sin(PHI(I))
                  else
                      GOTO 121
                  endif
```

Coding Procedure for Implementing ISR/FSR Effects

1) Pythia generates random values for Q^2, y and $\phi_{e'}$

2) Call RADGEN (using ‘call radgen_event’)

```
! calculate radiative corrections
      nu=sngl(gennu)
      q2=sngl(genq2)
      phi=sngl(genphi)
      yys=sngl(geny)
      xxs=sngl(genx)
      call RADGEN(mcSet_EneBeam,q2,nu,yys,xxs,phi,PhRAD,q2true,nutrue,
     +     radweight)
```

Variables from Pythia

Returns new Q^2, ν, radweight, and the 4-vector of the radiated photon (PhRAD)

Coding Procedure for Implementing ISR/FSR Effects

1) Pythia generates random values for Q^2, y and $\phi_{e'}$

2) Call RADGEN (using 'call radgen_event')

- Gets the photon's momentum, θ, and ϕ by using a previously generated lookup table and by integrating over probability densities
- Generates these values in the Tsai-Frame
- Once generated, RADGEN then rotates the photon into the 'lab frame' **BUT** it makes a convention error:
  - RADGEN assumes that the beam is in the +z direction in the lab frame but Pythia sets the beam in the **-z direction**
  - **This rotation goes to the CM frame, not the lab frame**
  - This caused RADGEN to report the photon with a missing rotation → **I add it after the photon is returned by rotating these momentum components around the x-axis**
    - **(rotation around y-axis did not fix the issues caused by this mistake)**

```
*
*      radiative photon
*
*        write(*,*)"dom=",dom,"dthg=",dthg,"dphig=",dphig
         deg=dom     ! Photon Energy
         dthg=dtk    ! Photon θ Angle
         dphig=dphk  ! Photon φ Angle
         sigma_total=sngl(tbor+tine+tpro+tnuc)

         q2tr=y+rrout*taout ! New Q^2
         anutr=sx/ap-dom    ! New ν
C        write(6,*) 'ita,deg,dthg,dphig,q2tr,anutr,xtr',
C     &                ita,deg,dthg,dphig,q2tr,anutr,
C     &                q2tr/(2.*amp*anutr)

         dgpz=deg*dcos(dthg)
         dgpxy=deg*dsin(dthg)
         dgpx=dgpxy*dcos(dphig)
         dgpy=dgpxy*dsin(dphig)
C         write(*,*)dphig,dthg,deg
*
*---momentum components in the LAB-system:
*---two rotations needed - first within the scattering plane
*---around the y-axis and second around the new z-axis (beam
*---direction) by Phi of the scattering plane
*
         dgplx=-dgpz*dsts+dgpx*dcts
         dgply=dgpy
         dgplz=dgpz*dcts+dgpx*dsts
         dcphi=dcos(dble(genphi))
         dsphi=dsin(dble(genphi))
         dplabg(1)=dgplx*dcphi-dgply*dsphi
         dplabg(2)=dgplx*dsphi+dgply*dcphi
         dplabg(3)=dgplz
```

Coding Procedure for Implementing ISR/FSR Effects

1) Pythia generates random values for Q^2, y and $\phi_{e'}$

2) Call RADGEN (using 'call radgen_event')

3) Return to Pythia (and rotate photon)

- Reset_Beam_Vars() is always called before RADGEN to make sure the beam is set appropriately (avoids reapplying ISR effects from prior RADGEN calls)
- BeamVary is a variable used to control whether my modifications are applied (BeamVary = 0 uses the original format)
- RadState flags ISR vs FSR events
- Rotating around the x-axis flips the signs of p_y and p_z

```
122             if (qedrad.eq.1) then
                   call Reset_Beam_Vars()
                   ! write(*,*)"Calling radgen_event"
                   RadState=0
                   call radgen_event
                   IF (BeamVary.NE.0) THEN ! Add the rotation of the radiated photon only if the ISR/FSR effects are included
                                           ! (maintains consistency with older reporting)
                      dplabg(2)=-dplabg(2)
                      dplabg(3)=-dplabg(3)
                   ENDIF
                endif
```

Rotation of the Radiated Photon

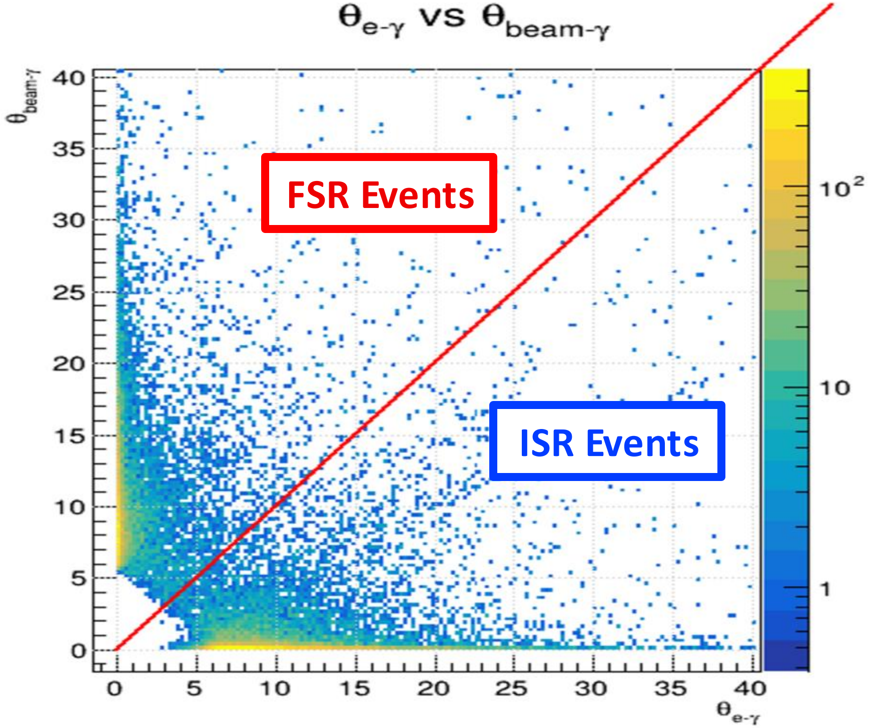
Coding Procedure for Implementing ISR/FSR Effects

1-3)…

4) Determine ISR, FSR, or No Radiation

- Check_I_or_F_SR_New() checks ISR vs FSR conditions and updates the flag (angles calculated using the dot product of the momentum vectors – replaces prior method of just using the polar angles)
- Apply_ISR_Effects() sets the new beam energy/momentum using conservation laws with the photon kinematics given by RADGEN

```
        call Check_I_or_F_SR_New(RadState)
        IF((RadState.EQ.1).AND.(BeamVary.EQ.1)) THEN
            ! WRITE(*,*)"RadState (ISR) = ", RadState
            ! ISR beam correction - Change the Beam Energy at the event vertex
            call Apply_ISR_Effects(dble(RadState))
        ENDIF
        Y(I)=dble(mcRadCor_NuTrue)/dble(mcSet_EneBeam)
        Q2(I)=dble(mcRadCor_Q2True)
```



Coding Procedure for Implementing ISR/FSR Effects

1-4)…

5) Calculate/update remaining kinematics and cuts (same as normal Pythia)

6) Update the Scattered electron's ϕ angle

- Calculates a new angle by summing the x and y momentum components of the pre-RADGEN scattered electron and the radiated photon to calculate ϕ (prior versions did not account for the type of radiation or the frames of the particles)

```
C...Try 'new' phi
                    IF ((qedrad.ne.0).and.(mcRadCor_EBrems.gt.0)) then
                        IF (BeamVary.EQ.0) THEN
                            ! Beam Variations are turned OFF (using default calculation of PHI(I))
                            emom=sqrt(geneprim**2-masse**2)
                            PHI(I)=atan2((emom*ppy+dplabg(2)),
     &                        (emom*ppx+dplabg(1)))
                            IF (PHI(I).lt.0) THEN
                                PHI(I)=PHI(I)+PARU(2)
                            ENDIF
                        ELSEIF((RadState.EQ.2).OR.(BeamVary.EQ.2)) THEN
                        ! Only recalculate the scattered electron's φ angle if the event is FSR (or if ISR/FSR effects are turned off)
                        ! ISR photons should not be correlated with the φ angle of the scattered electron (only the incoming electron)
                            ! The radiated photon should be rotated back into the CM frame before being added to the scattered electron
                            ! (Done by flipping the sign of y back to what RADGEN gave)
                            emom=sqrt(geneprim**2-masse**2)
                            PHI(I)=atan2((emom*ppy-dplabg(2)),
     &                        (emom*ppx+dplabg(1)))
                            IF (PHI(I).lt.0) THEN
                                PHI(I)=PHI(I)+PARU(2)
                            ENDIF
                        ENDIF
                    ENDIF
```

Old Version (no rotation or flag of ISR/FSR)

RadState = 2 means that this is applied to FSR events only

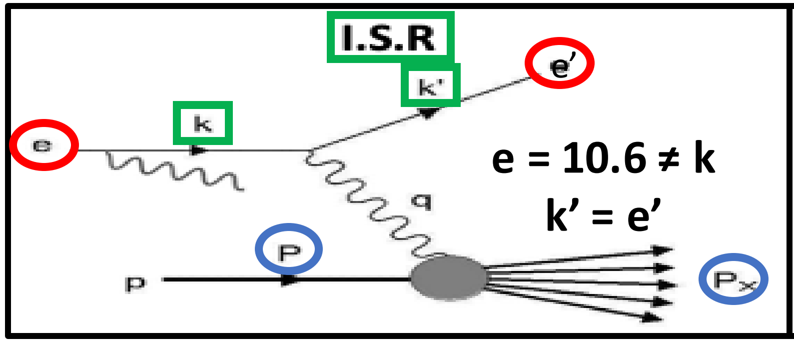
Sign of y momentum is flipped back to what RADGEN gave (used as CM frame)

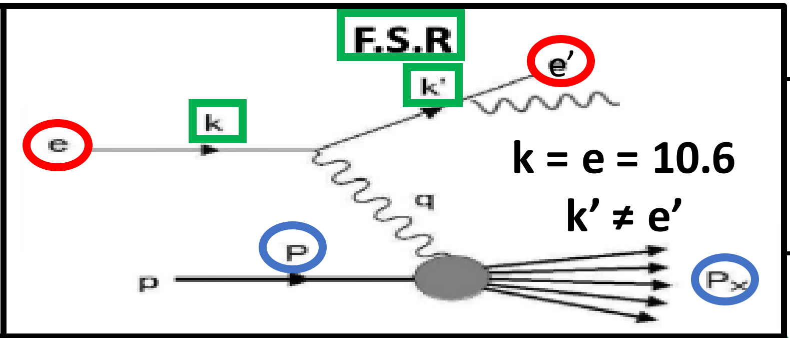
Coding Procedure for Implementing ISR/FSR Effects

1) Pythia generates random values for Q^2, y and $\phi_{e'}$
2) Call RADGEN (using 'call radgen_event')
3) Return to Pythia (and rotate photon)
4) Determine ISR, FSR, or No Radiation
5) Calculate/update remaining kinematics and cuts (same as normal Pythia)
6) Update the Scattered electron's ϕ angle
7) Save and continue
8) Report the FSR scattered electrons (with status code 1) as the scattered electron recorded in the event record minus the radiated photon's kinematics (as per $\boldsymbol{e'_{rad}} = \boldsymbol{k'} - \gamma$)

RADGEN Explanation – How to Include into My Analysis

1. **Apply ISR effects within Pythia to change the beam momentum/energy at the vertex**
   - Effects the kinematics of the electron with status code 21 (1^{st} particle in the event record)
   - Done before calculating the scattered electron (and all other particle kinematics) to ensure that momentum/energy is conserved (as per the earlier slides)
2. **Apply FSR effects before using GEMC to simulate event reconstruction**
   - Can be applied at the end of the Pythia event generation to the scattered electron as it is written to the event record
   - Only need to apply to the scattered electron with status code 1 (another copy of the vertex scattered electron was already saved with a status code of 21 as the 3^{rd} particle in every event record, but that one is generally ignored by GEMC)





- GREEN SQUARES → Used to Calculate MC GEN
- RED CIRCLES → Used to Calculate MC REC
- BLUE CIRCLES → Used in Both