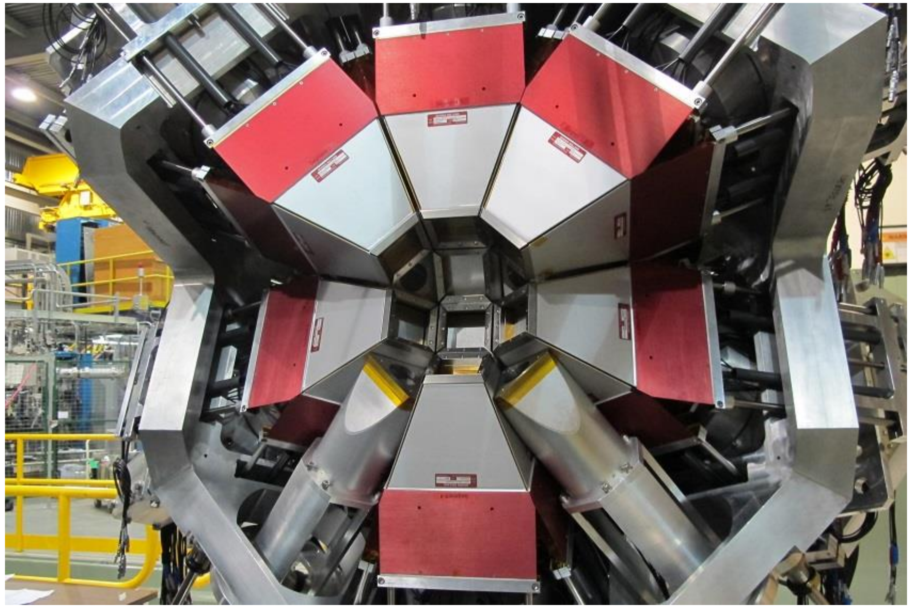
Experimental measurements of gamma-ray angular correlations with GRIFFIN



A practical guide

M. Bowry, 8th February 2019

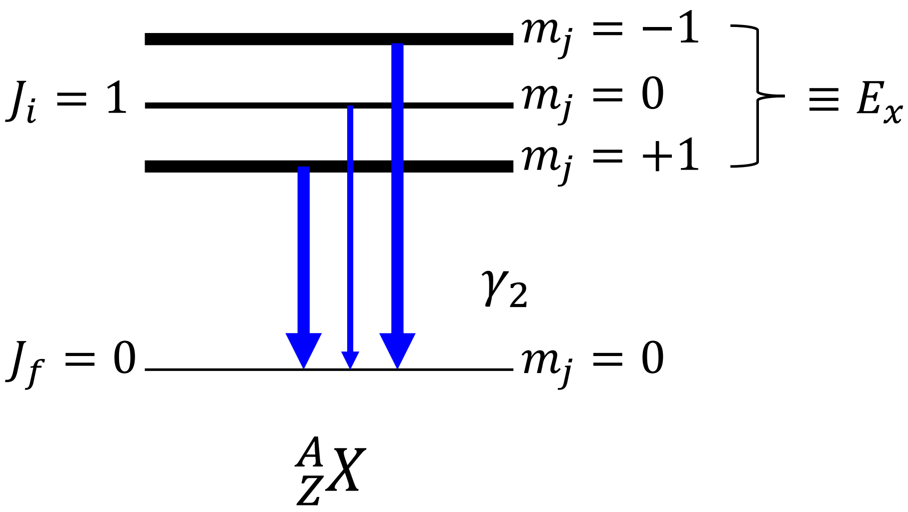
Recommended reading

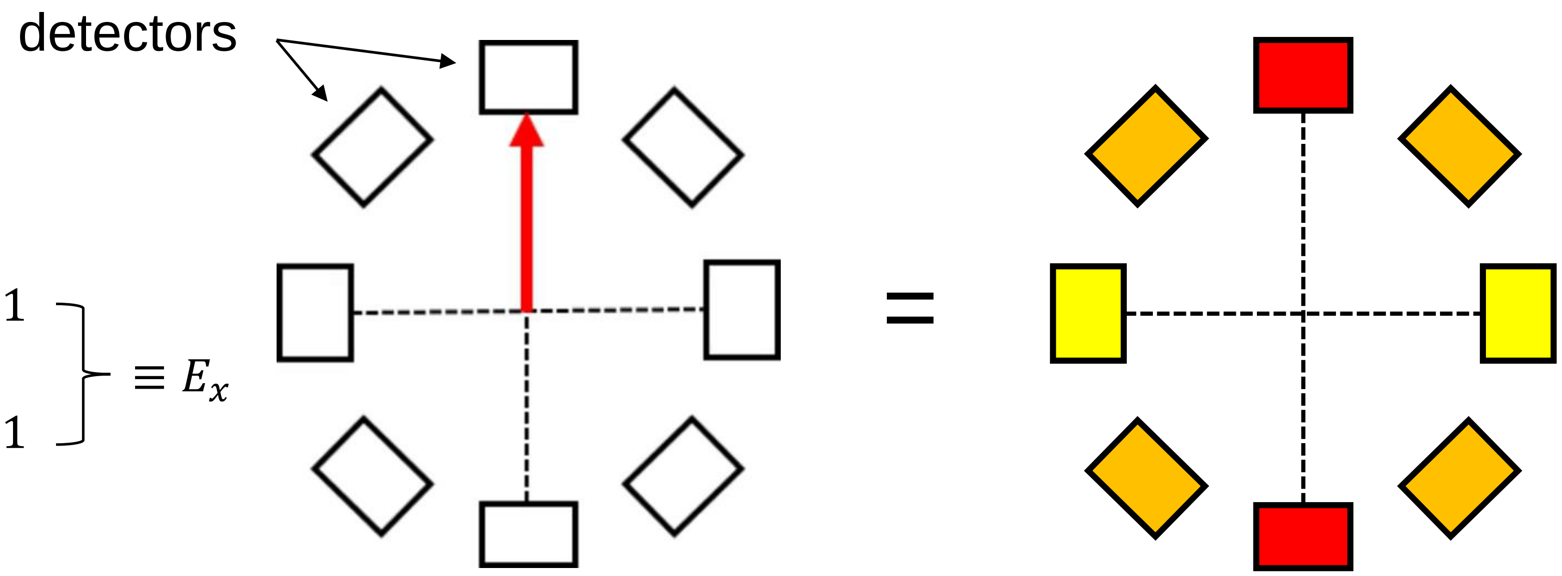
- *The GRIFFIN facility for Decay-Spectroscopy studies at TRIUMF-ISAC*, Nuclear Inst. and Methods in Physics Research A, 918, 9 (2019)
- *Gamma-gamma angular correlation techniques with the GRIFFIN spectrometer*, Nuclear Inst. and Methods in Physics Research A, 922, 47 (2019)
- *M1-E2 mixing ratios and conversion electron particle parameters for the electromagnetic transitions in ^{75}As*, Physical Review 180, 1043 (1969)
- *Nuclear Physics of Stars Appendix D*, C. Iliadis, pg. 599-617

- (Report) *Compton Polarimetry with GRIFFIN Documentation and Physics Review*,
  - Plone: Griffin->Reports->Co-op Report, Dan Southall

Introduction

- **Angular distribution** of radiation emitted in decay experiments
  - ❑Ensemble of “orientated” radioactive nuclei decay from state $J_i \rightarrow J_f$ (i.e. nuclei orientated with unequal population of magnetic sub-states), aligned with respect to experiment



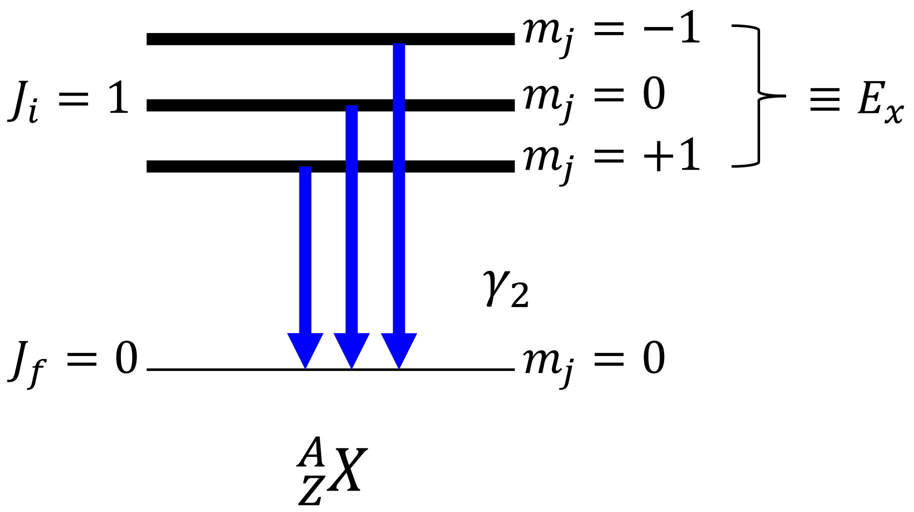


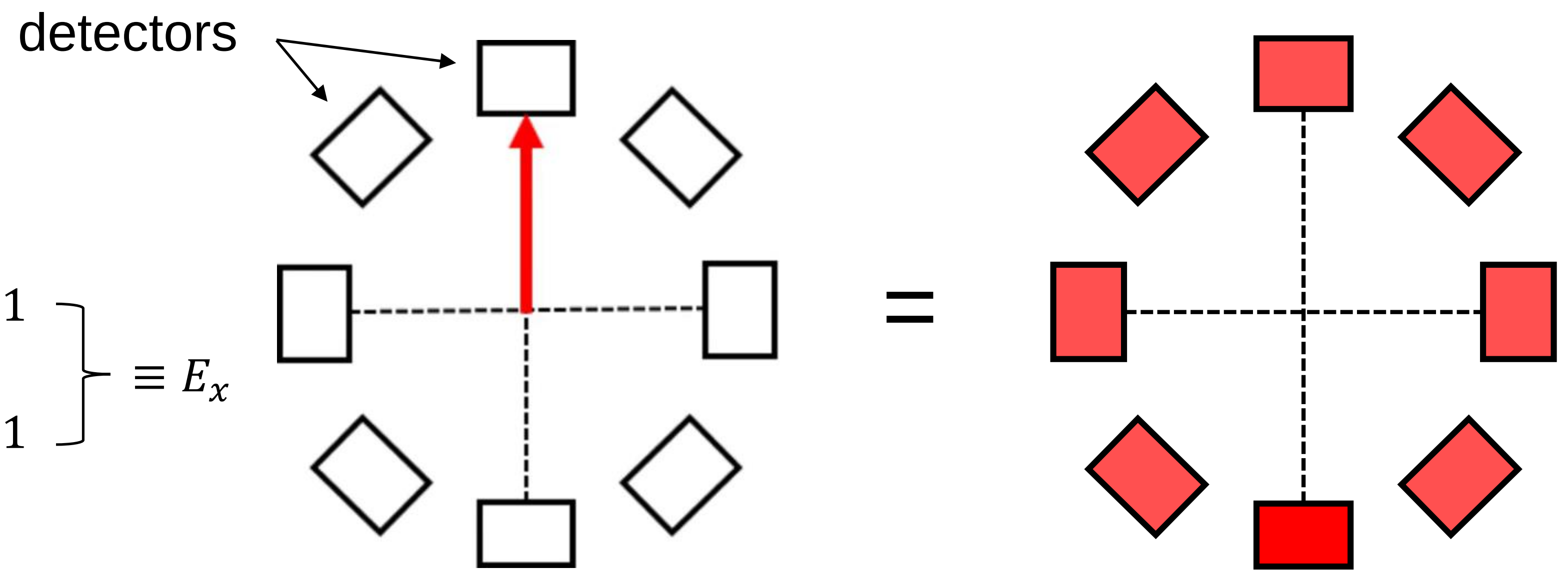
Rate N_{γ_2} per crystal reflects angular distribution

mbowry@triumf.ca

Introduction

- **Angular distribution** of radiation emitted in decay experiments
  - ❑Ensemble of radioactive nuclei with no preferred orientation (in β decay, no gate on outgoing β particle or not detected at all) and no alignment relative to the experiment.



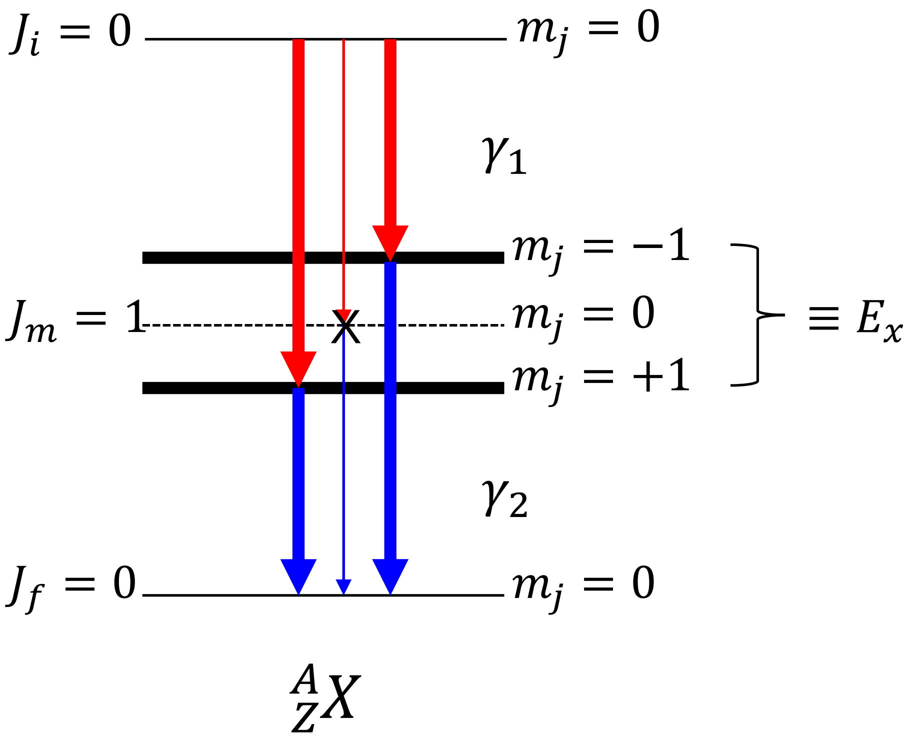


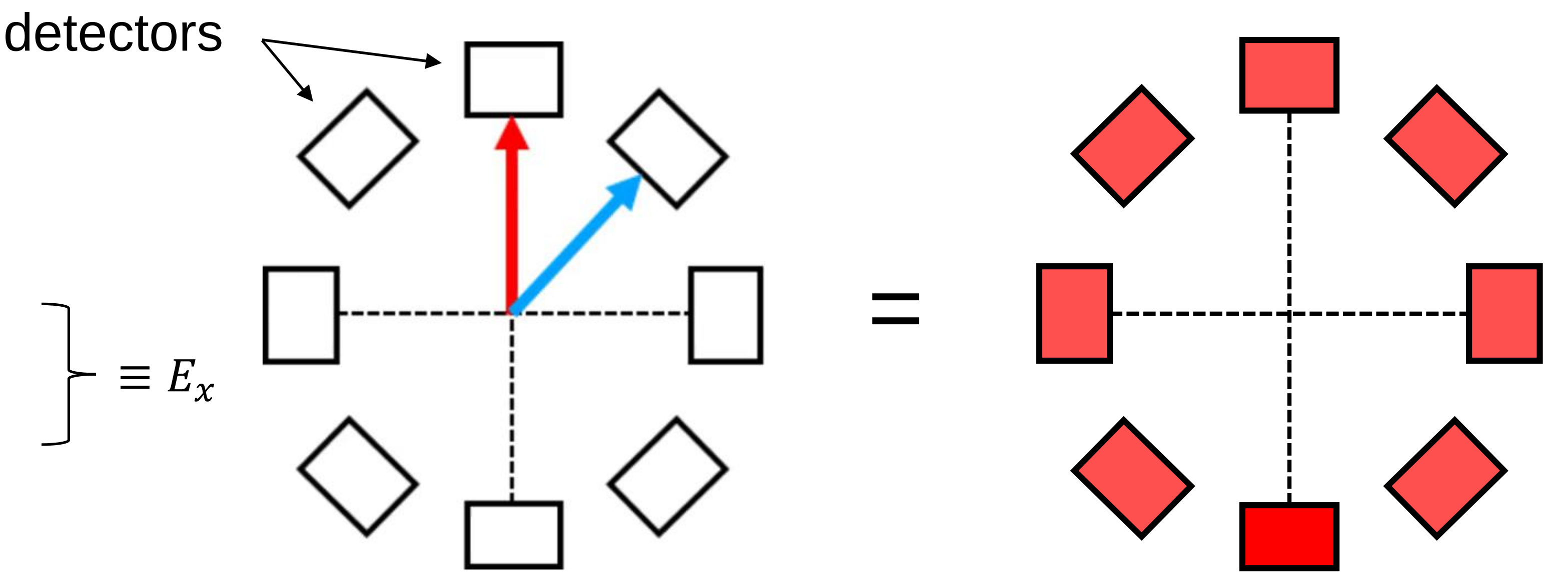
Rate N_{γ_2} per crystal is isotropic

mbowry@triumf.ca

Introduction

- **Angular correlation**: measure the **coincidence frequency** of two successive decays at different angles (γ_2 orientated with respect to γ_1)





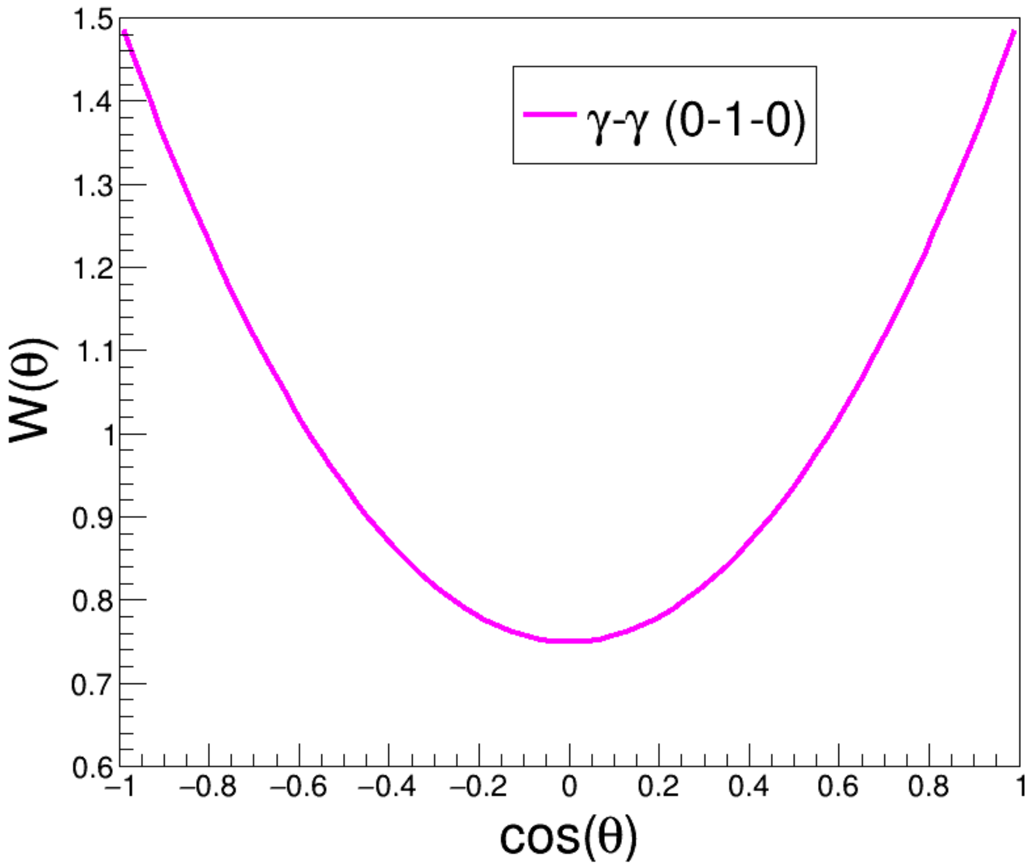
Rate $N_{\gamma_1\gamma_2}$ provides angular correlation

Introduction

- Angular correlation formula (generalized)

$$W(\theta) = \sum_{k=0,k=even}^{\infty} B_{ii} G_{ii}(t) A_{ii} P_i(\cos\theta)\,,$$

where:
$B_{ii} = initial\ nuclear\ orientation,$
$G_{ii}(t) = perturbation\ factor$
$(interaction\ with\ external\ electric, magnetic\ fields),$
$\boldsymbol{A_{ii} = nuclear\ structure\ [L_1, L_1', \delta_1], [L_2, L_2', \delta_2],}$
$P_i = Legendre\ polynomials$



- .. simplifies to a short expansion for many γ-γ cascades (L=1,2).

Introduction

- Quality factors
  - In general, the **measured** angular correlation does not represent nature exactly: $\boldsymbol{W_\theta}$ is highly sensitive to the experimental setup and other factors including:

    - ❑Non-negligible lifetimes of intermediate nuclear states ($G_{ii}(t)$)
    - ❑Detector size, geometry and efficiency.

  - For a complex setup like GRIFFIN, **simulations** provide the most direct route to extracting accurate measurements (of δ for example).

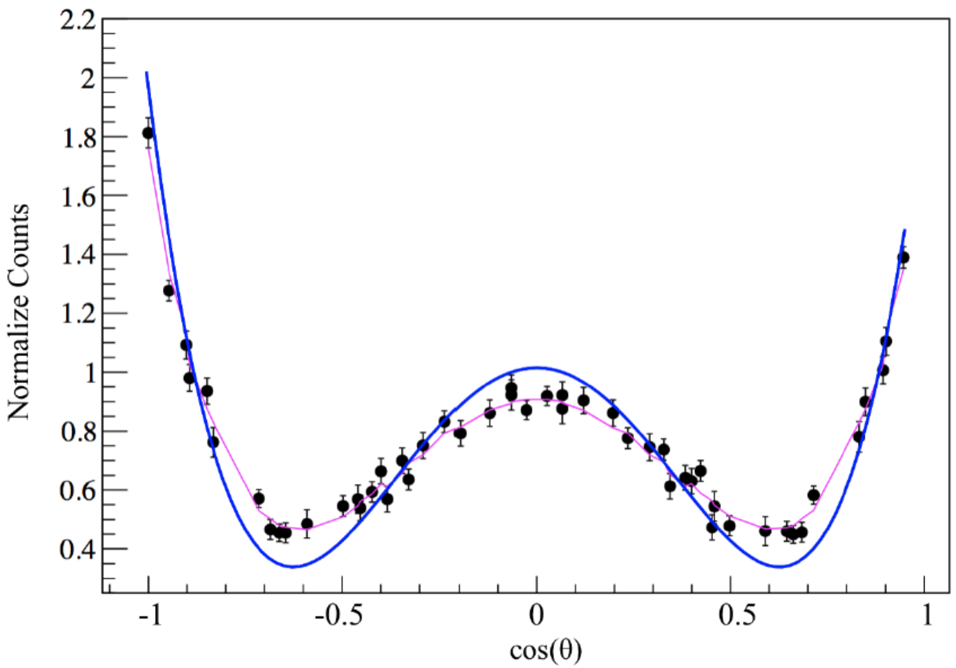


Figure 3: The χ^2/NDF between a full simulation of the ^{66}Zn $0^+ - 2^+ - 0^+$ cascade (magenta filled line) with $a_2 = 0.3571$ and $a_4 = 1.1429$ and data (black points) is 1.01. The blue solid line is the angular correlation expected from theoretically calculated a_2, a_4 coefficients without corrections for finite detector size effects.

J.K. Smith et al.

Analysis procedure

- **Unpack** data using GRSISort (MIDAS fragments → ROOT trees)
- **Sort** data using GRSI-Parallel ROOT Facility (i.e. parallelized analysis)
- Go to **.. /GRSISort/GRSIProof/**
- PROOF scripts have two parts, a header (.h) and main (.C)
  - **> cp AngularCorrelationSelector.C ggAngularCorrelation.C**
  - **> cp AngularCorrelationSelector.h ggAngularCorrelation.h**
  - Open the new files in a text editor (e.g. vim, gedit etc.).
  - Replace references to these new names within each file.

- See also slides by R. Dunlop available online (Plone):
  - https://grsi.wiki.triumf.ca/index.php/Access
  - Follow link to Plone page (requires registration)
  - General->Meetings->Joint Tigress and Griffin Collaboration Meeting 2017->GRSISort presentations

Setting up GRSIPROOF

- Currently, **ggAngularCorrelation** is setup to automatically generate γ-γ coincidence angles using spatial coordinates stored in the TGriffin library.
  - ../GRSISort/libraries/TGRSIAnalysis/TGriffin/TGriffin.cxx

```
TVector3 TGriffin::gCloverPosition[17] = {
   TVector3(TMath::Sin(TMath::DegToRad() * (0.0)) * TMath::Cos(TMath::DegToRad() * (0.0)),
            TMath::Sin(TMath::DegToRad() * (0.0)) * TMath::Sin(TMath::DegToRad() * (0.0)),
            TMath::Cos(TMath::DegToRad() * (0.0))),
   // Downstream lampshade
   TVector3(TMath::Sin(TMath::DegToRad() * (45.0)) * TMath::Cos(TMath::DegToRad() * (67.5)),
            TMath::Sin(TMath::DegToRad() * (45.0)) * TMath::Sin(TMath::DegToRad() * (67.5)),
            TMath::Cos(TMath::DegToRad() * (45.0))),
   TVector3(TMath::Sin(TMath::DegToRad() * (45.0)) * TMath::Cos(TMath::DegToRad() * (157.5)),
            TMath::Sin(TMath::DegToRad() * (45.0)) * TMath::Sin(TMath::DegToRad() * (157.5)),
            TMath::Cos(TMath::DegToRad() * (45.0))),
   TVector3(TMath::Sin(TMath::DegToRad() * (45.0)) * TMath::Cos(TMath::DegToRad() * (247.5)),
            TMath::Sin(TMath::DegToRad() * (45.0)) * TMath::Sin(TMath::DegToRad() * (247.5)),
            TMath::Cos(TMath::DegToRad() * (45.0))),
   TVector3(TMath::Sin(TMath::DegToRad() * (45.0)) * TMath::Cos(TMath::DegToRad() * (337.5)),
            TMath::Sin(TMath::DegToRad() * (45.0)) * TMath::Sin(TMath::DegToRad() * (337.5)),
            TMath::Cos(TMath::DegToRad() * (45.0))),
```

Theta, °

Phi, °

Setting up GRSIPROOF

- Both ROOT and GRSI-specific functions are employed in **ggAngularCorrelation.h** to return the available coincidence angles.

```
// function to calculate angles (from LeanCorrelations), implemented at the end of this file
std::vector<std::pair<double, int>> AngleCombinations(double distance = 110., bool folding = false,
                                                       bool addback = false);
```

Function definition

```
std::vector<std::pair<double, int>> AngleCombinations(double distance, bool folding, bool addback)
{
   std::vector<std::pair<double, int>> result;
   std::vector<std::pair<double, int>> grouped_result;
   std::vector<double> angle;
   for(int firstDet = 1; firstDet <= 16; ++firstDet) {
      for(int firstCry = 0; firstCry < 4; ++firstCry) {
         for(int secondDet = 1; secondDet <= 16; ++secondDet) {
            for(int secondCry = 0; secondCry < 4; ++secondCry) {
               if(firstDet == secondDet && firstCry == secondCry) {
                  continue;
               }
               if(!addback) {
                  angle.push_back(TGriffin::GetPosition(firstDet, firstCry, distance)
                                       .Angle(TGriffin::GetPosition(secondDet, secondCry, distance)) *
                                    180. / TMath::Pi());
               }
```

Detector modifications (etc.) go here!

Body

mbowry@triumf.ca

Setting up GRSIPROOF

- Histograms are defined in **ggAngularCorrelation.C** where the general format is:

```
#define ggAngularCorrelation_cxx
#include “ggAngularCorrelation.h”
//variable definitions (doubles, integers etc.)

void ggAngularCorrelation::CreateHistograms() { ; }

void ggAngularCorrelation::FillHistograms() { ; }
```

Histogram definitions

Body

Setting up GRSIPROOF

- Modify the .C file to suit the task.
  - **Remove histograms that will not be used** (including *Create..* and *Fill..* functions). Additional histograms mean more memory usage multiplied by the number of parallel threads!

“gammaGamma”, “gammaGammaBG”, “gammaGammaMixed” ☺
(+ diagnostic histograms)

Single-crystal analysis (no addback)

  - **Check histogram binning** (more bins = more memory usage).
  - **Add event-mixing depth functionality**. This is used to reduce the statistical uncertainty in the event-mixed coincidence spectra and is simple to implement.
  - **Add diagnostic spectra** to check angle calculations are functioning normally (*always assume* your code is broken from the beginning).

Setting up GRSIPROOF

- Gamma-gamma angular correlation matrices (ggAngularCorrelation.C)

```
for(auto g1 = 0; g1 < fGrif->GetMultiplicity(); ++g1) {
   auto grif1 = fGrif->GetGriffinHit(g1);
   for(auto g2 = 0; g2 < fGrif->GetMultiplicity(); ++g2) {
      if(g1 == g2) continue;
      auto   grif2 = fGrif->GetGriffinHit(g2);
      double angle = grif1->GetPosition().Angle(grif2->GetPosition()) * 180. / TMath::Pi();
      if(angle < 0.0001) continue;
      auto   angleIndex = fAngleMap.lower_bound(angle - 0.0005);
      double ggTime     = TMath::Abs(grif1->GetTime() - grif2->GetTime());

      if(ggTime < ggHigh) {
         fH2[Form("gammaGamma%d", angleIndex->second)]->Fill(grif1->GetEnergy(), grif2->GetEnergy());
      } else if(bgLow < ggTime && ggTime < bgHigh) {
         fH2[Form("gammaGammaBG%d", angleIndex->second)]->Fill(grif1->GetEnergy(), grif2->GetEnergy());
      }
   }
}
```

Setting up GRSIPROOF

- Gamma-gamma angular correlation matrices + event mixing

```
int check, lgsize, event_mixing_depth = 11;
std::vector<TGriffin> lastgrif;
..
for(auto g1 = 0; g1 < fGrif->GetMultiplicity(); ++g1) {
   auto grif1 = fGrif->GetGriffinHit(g1);
   for(auto g2 = 0; g2 < fGrif->GetMultiplicity(); ++g2) { ; }

   check = (int)lastgrif.size();
   if(check<event_mixing_depth) continue;
   for(auto lg = 0; lg < (check-1); ++lg){
   int multLG = lastgrif.at(lg).GetMultiplicity();

      for(auto g3 = 0; g3 < multLG; ++g3) {
         auto grif3 = lastgrif.at(lg).GetGriffinHit(g3);
         double angle = grif1->GetPosition().Angle(grif3->GetPosition()) * 180. / TMath::Pi();
         if(angle < 0.0001) continue;
         auto angleIndex = fAngleMap.lower_bound(angle - 0.0005);
         fH2[Form("gammaGammaMixed%d", angleIndex->second)]->Fill(grif1->GetEnergy(), grif3->GetEnergy());
      }
   }
}
lastgrif.push_back(*fGrif);
lgsize = (int)lastgrif.size();
if(lgsize>event_mixing_depth) {
   lastgrif.erase(lastgrif.begin());
}
```

Setting up GRSIPROOF

- **Add diagnostic histograms**. This includes the basics (hit energy summed over the whole array, time difference etc.) in addition to ensuring:

  ✓The calculated angles are accurate
  ✓The correct (grouped) angles are assigned to the appropriate bins.

- For example, the relative change in the integral (or the counts in a given photopeak) of the event-mixed matrices will be very similar to the relative **combinatorial weights** (# crystal pairs) and is easily checked.

Setting up GRSIPROOF

- In this example analysis, the number of angles is restricted to **7** (full analysis = 51 angle bins).

Index number: {0, 4, 10, 25, 40, 46, 50}
Angle (°): {18.8, 33.7, 60.2, 91.5, 126.2, 148.1, 180.0}

- Added filter to *Create* and *Fill* functions to exclude other angles. This is useful for any analysis to reduce the number of histograms.
- Filter can be modified to include the next set of angles etc.
- Issues encountered with ROOT v6.06.08 + GRSISort v3.1.3.4 regarding the ***angleIndex*** iterator (solution available). **Use GRSISort v3.1.3.5 or higher**.

Setting up GRSIPROOF

- **Run PROOF**. For example, in a terminal within a ***/data/*** directory:

```
> grsiproof /path2/analysis09636* /path2/ggAngularCorrelation.C --max-workers=4
```

- Common failure modes. All of the following **will cause a crash**:
  - **Histogram definition missing** in *CreateHistograms* function.
  - **Referencing hits in a detector branch that does not exist** in the AnalysisTree or is not defined in the header file.
  - **Iterating over a value or object (e.g. a histogram) that does not exist**. Most commonly occurs in a ***for(){ ; }*** loop.
- Calibration files cannot be entered as a PROOF argument (yet).
- PROOF compiled upon execution.

Setting up GRSIPROOF

Running PROOF with the default number of workers

- Most GRSI cluster machines have (effectively) 8 CPUs. The **default number of workers is 8**. To stop these processes:

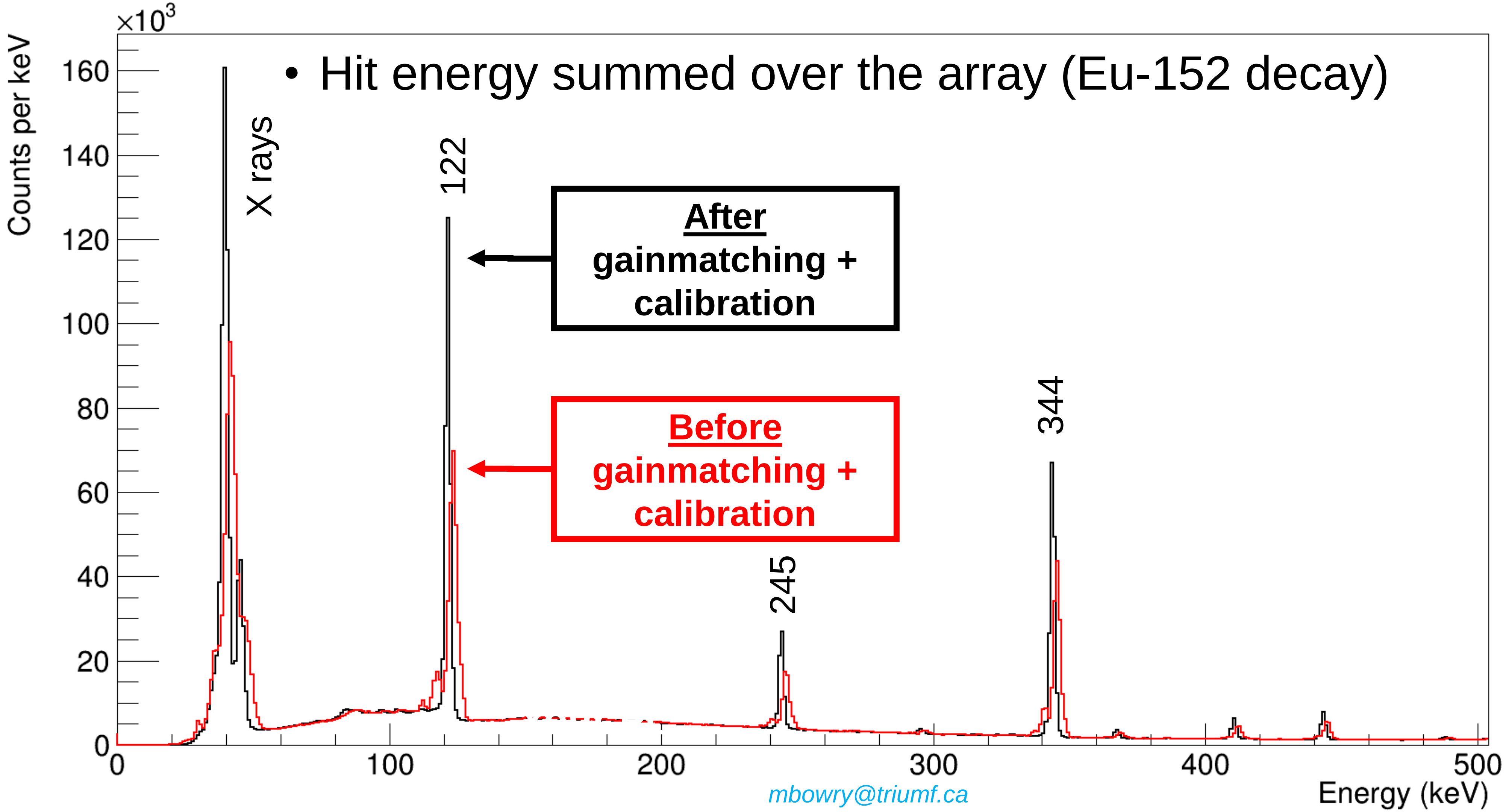
  > ps aux | grep proofserv

  > kill -9 [pid1] [pid2] .. [pidN]

- Reduce the number of workers, histograms or histogram binning and try again (e.g. add angle filter).

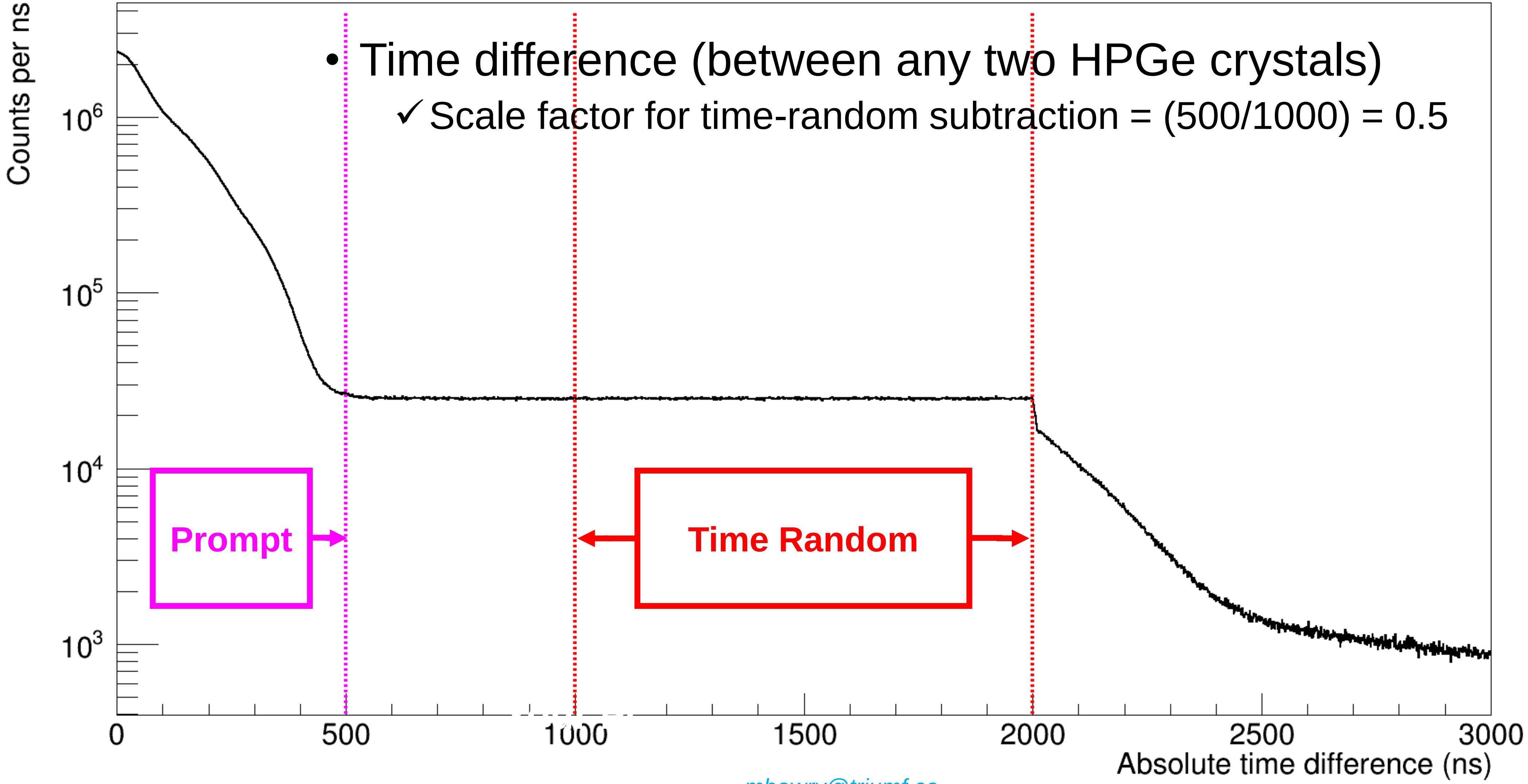
Results and analysis

- Hit energy summed over the array (Eu-152 decay)



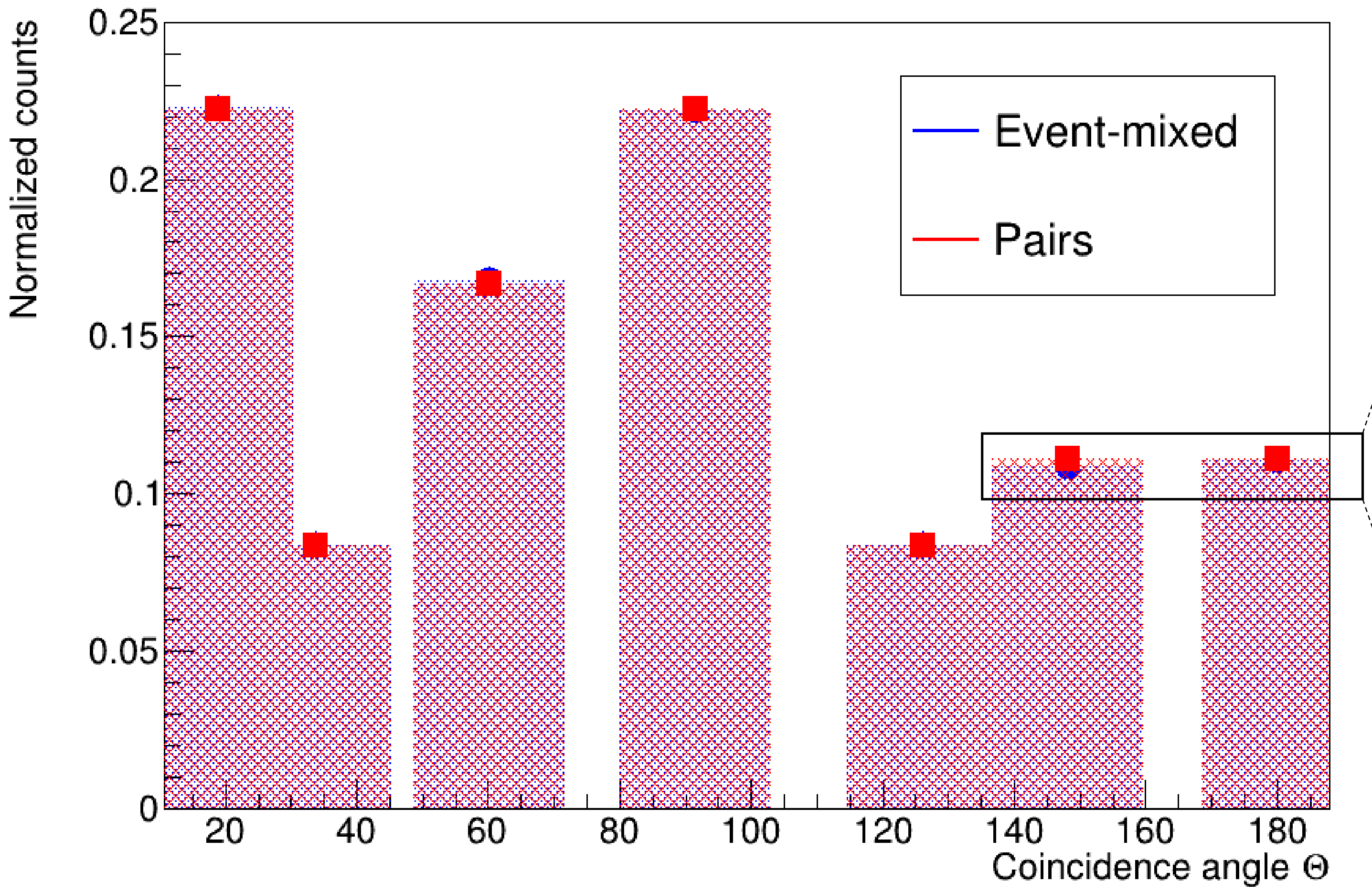
mbowry@triumf.ca

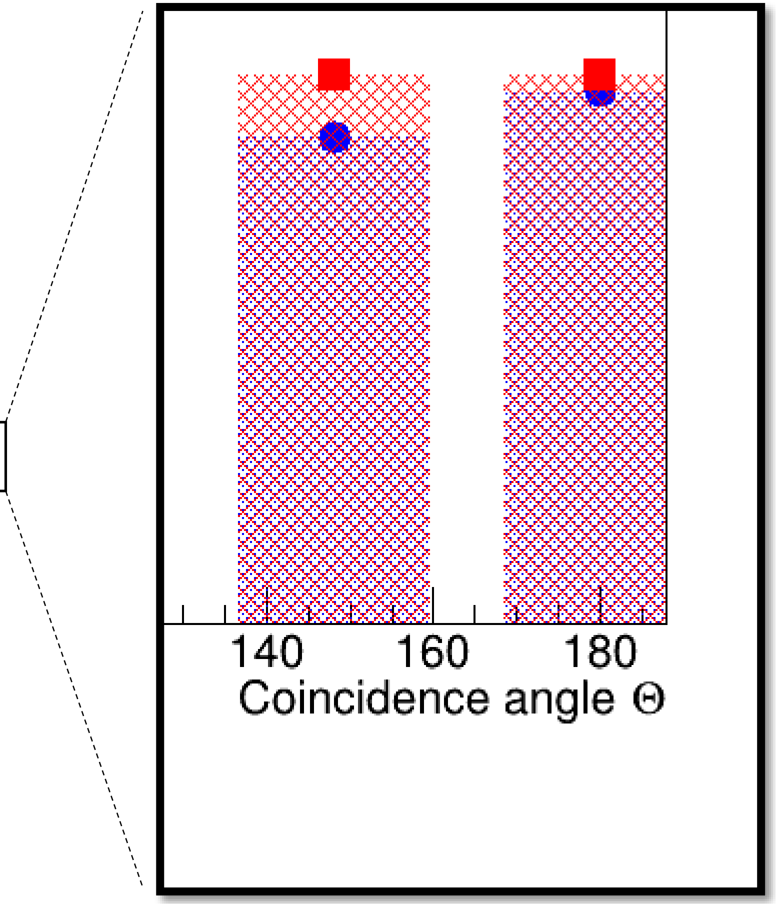
Results and analysis



Results and analysis

- Relative integrals: Event-mixed $\gamma\gamma$ matrices





Results and Analysis

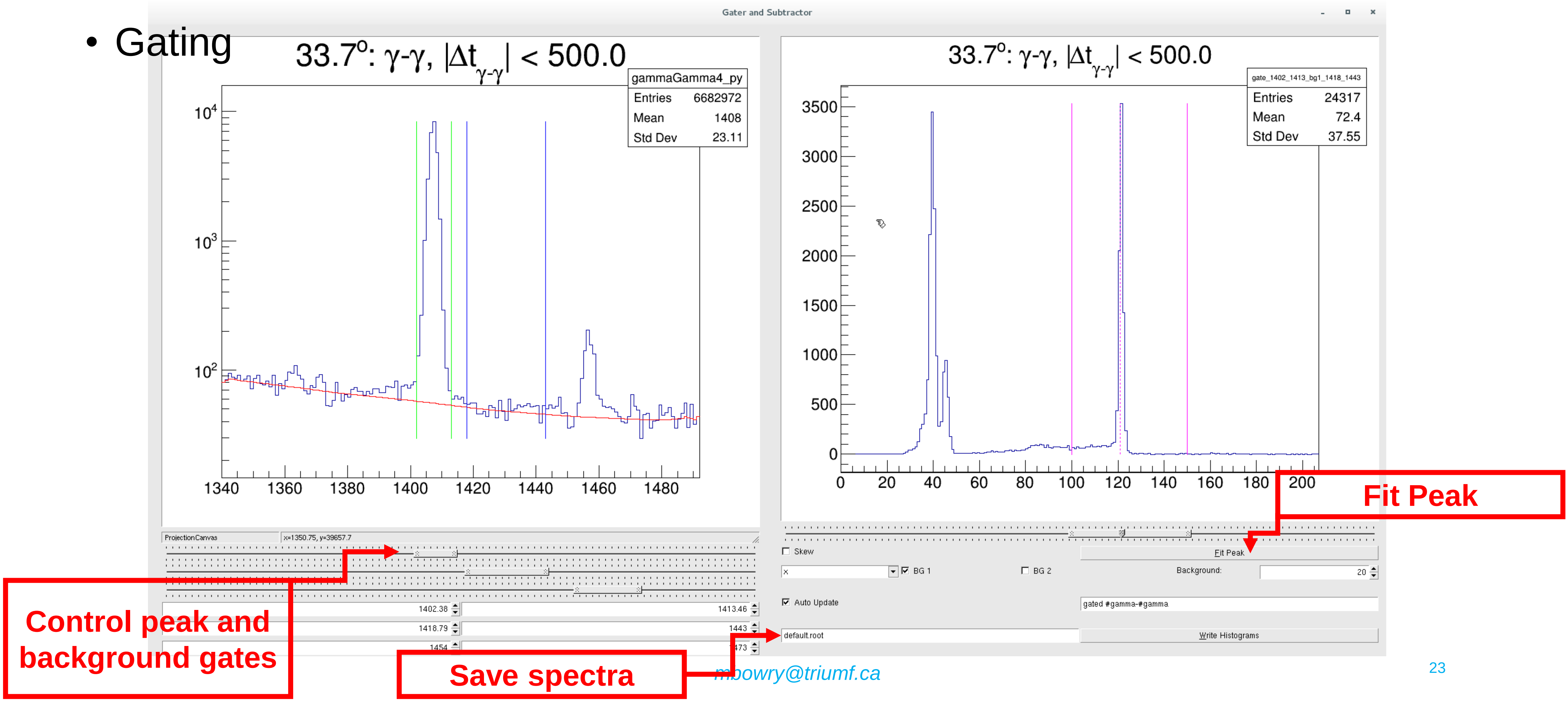
- Time-random (TR) subtractions

```
> grsisort
        > gammaGammaBG[i]->Scale(500./1000.);
        > gammaGamma0->Add(gammaGammaBG[i],-1.);
        > new TBGSubtraction(gammaGamma0);
```

- Note: TR subtractions **not** required for event-mixed $\gamma\gamma$ matrices
- Also available: James Smallcombe's **jRoot** environment for gating and peak fitting (added functionality w.r.t. TBGSubtraction): https://github.com/jsmallcombe/jRootAnalysisTools

Results and Analysis

- Gating



Results and Analysis

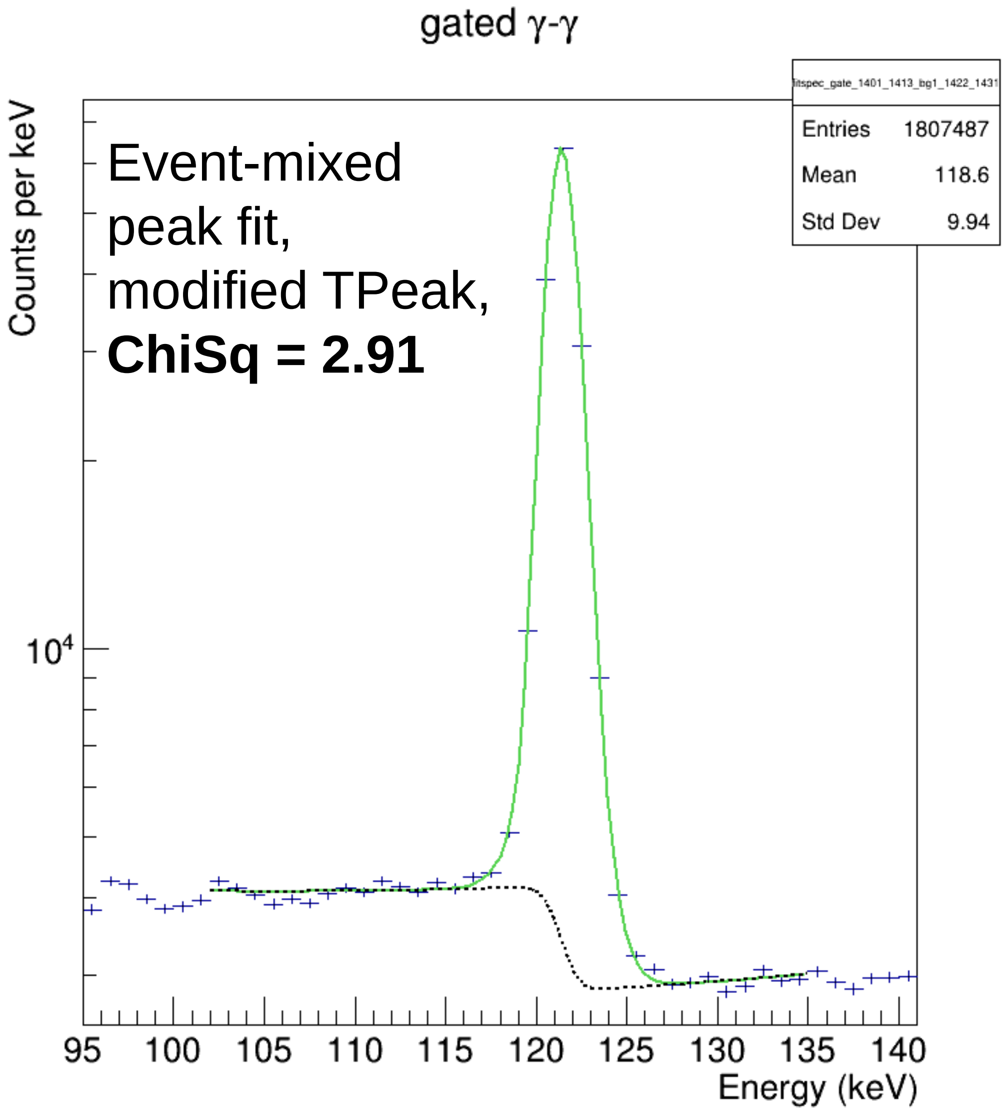
- Peak fitting

Reduced chi-squared values, 18.8 to 180.0°
Correlated : {3.00, 3.07, 1.87, 1.50, 5.30, 2.24, 2.12}
Event-mixed : {3.25, 1.77, 1.93, 2.91, 4.80, 2.90, 1.64}
<u>7,000 → 20,000 counts</u> in the correlated peaks.

.. always room for improvement!

- Control over peak parameters most useful
- Background and high-energy tail* have a significant effect on the success/failure of the fit.

*e.g. due to poor gainmatching, pile-up

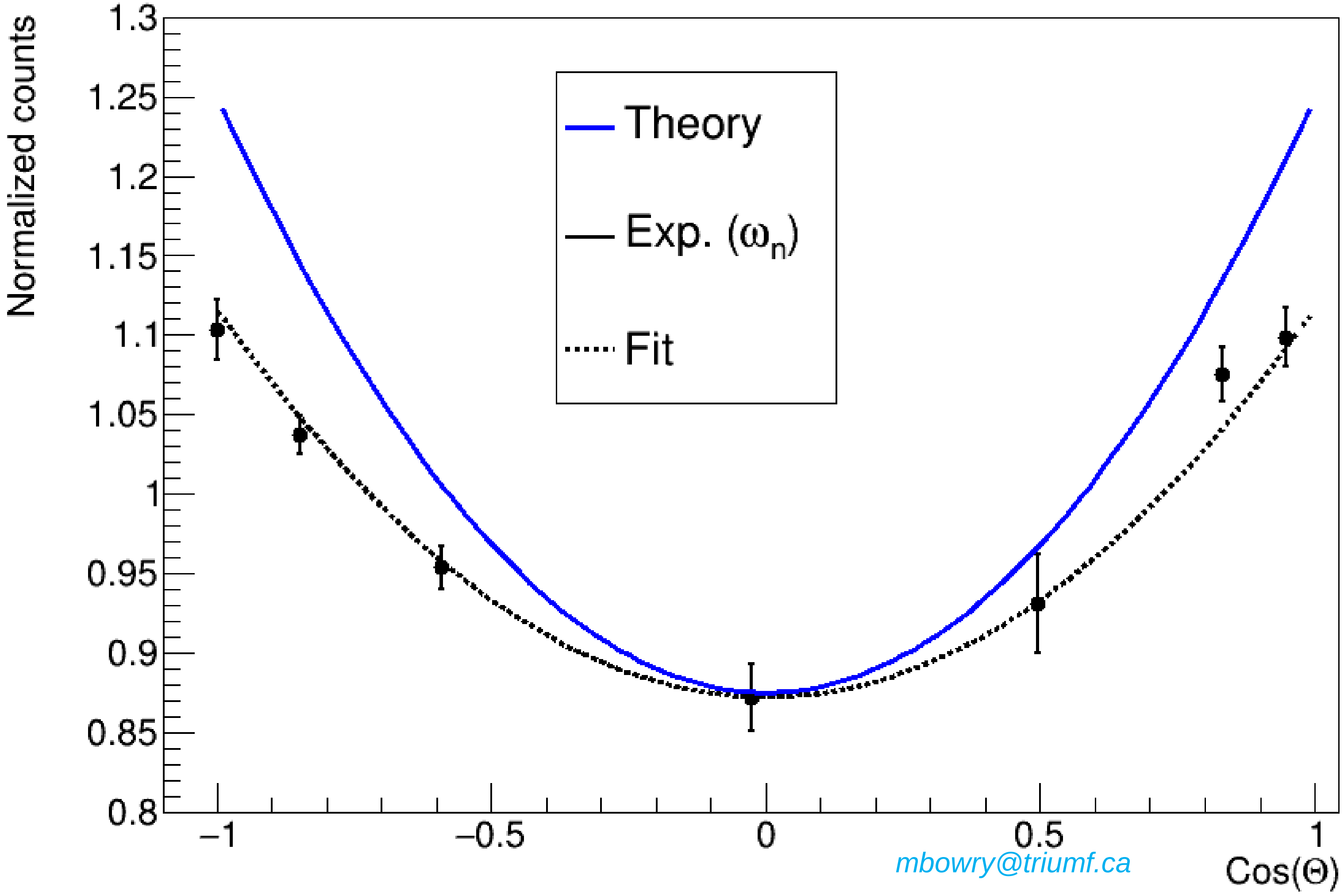


mbowry@triumf.ca

Results and Analysis

3 days later …

- Angular correlation



$$W_\theta = \frac{N_\theta^{\gamma\gamma}}{\omega_\theta F}$$

$$F = \sum_i^n N_{\theta_i}^{\gamma\gamma} \Big/ \sum_i^n \omega_{\theta_i}$$

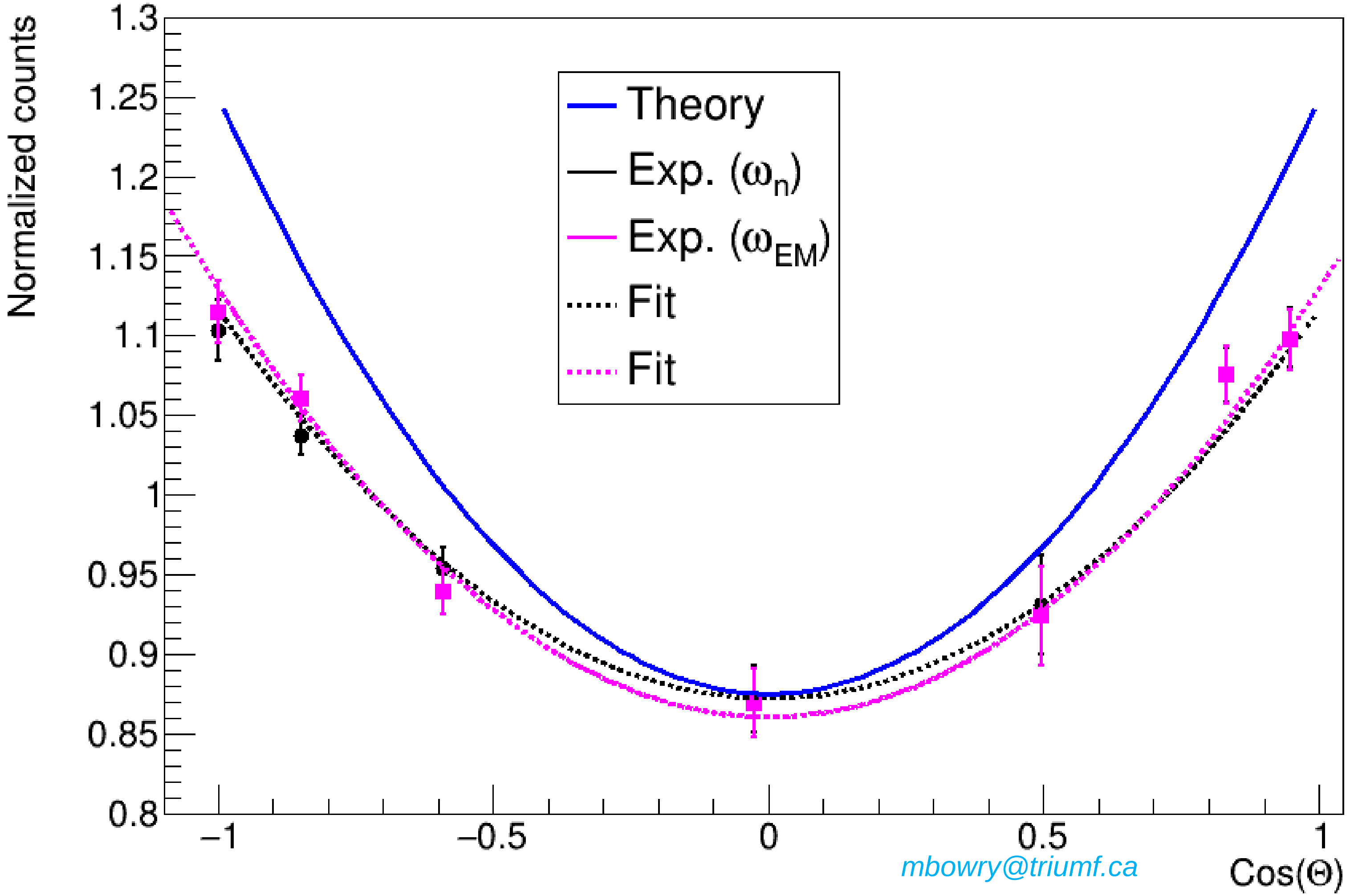
$$\omega_{\theta_n} = nominal\ weight\ (\#pairs)$$

mbowry@triumf.ca

Results and Analysis

3 days later …

- Angular correlation



$$W_\theta = \frac{N_\theta^{\gamma\gamma}}{\omega_\theta F}$$

$$F = \sum_i^n N_{\theta_i}^{\gamma\gamma} \Big/ \sum_i^n \omega_{\theta_i}$$

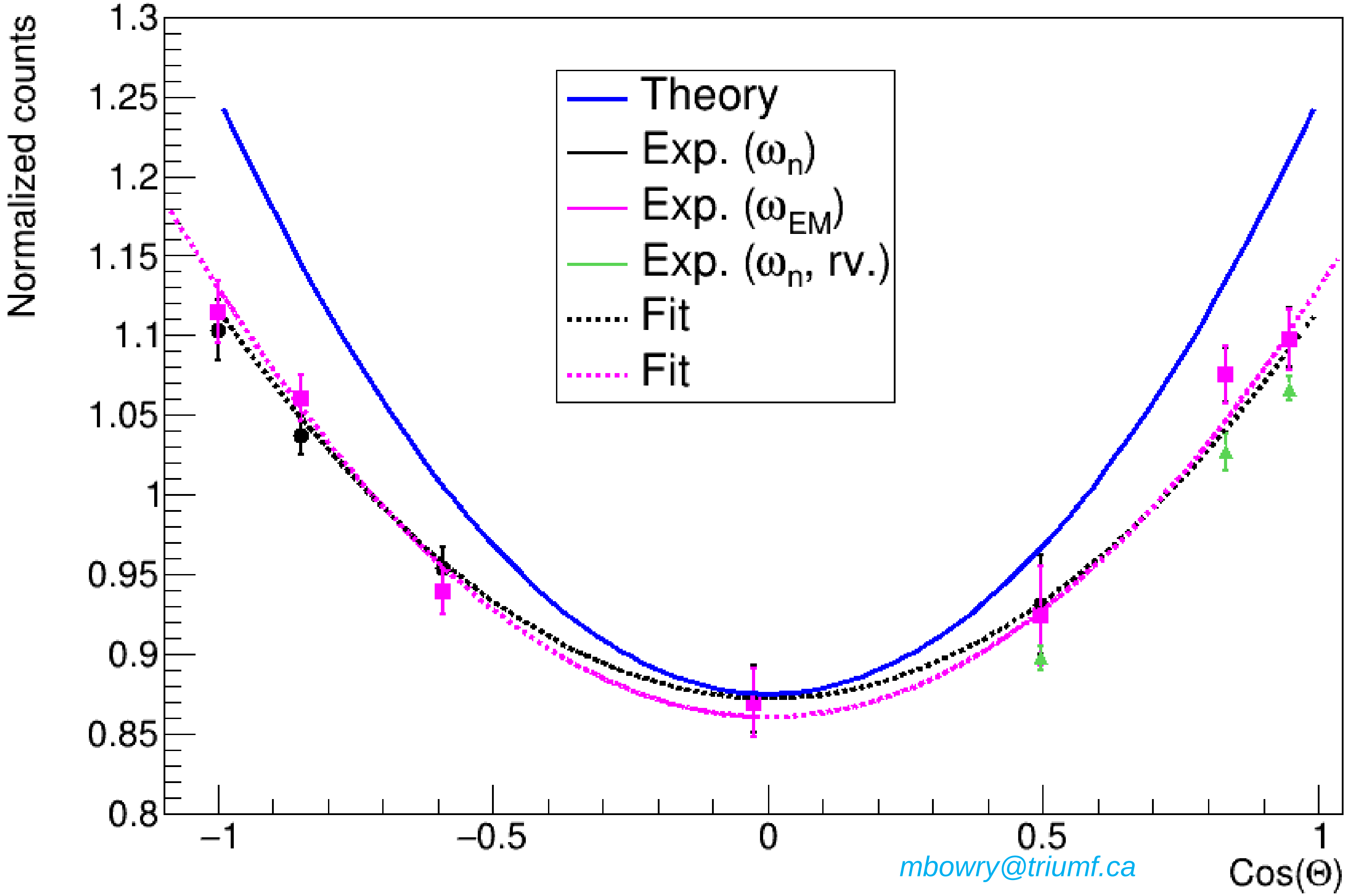
$$\omega_{\theta_n} = nominal\ weight\ (\#pairs)$$

$$\omega_{\theta_{EM}} = event - mixed\ weight$$

Results and Analysis

3 days later …

- Angular correlation, **reverse gate (fit the 1408)**



$$W_\theta = \frac{N_\theta^{\gamma\gamma}}{\omega_\theta F}$$

$$F = \sum_i^n N_{\theta_i}^{\gamma\gamma} \Big/ \sum_i^n \omega_{\theta_i}$$

$$\omega_{\theta_n} = nominal\ weight\ (\#pairs)$$

$$\omega_{\theta_{EM}} = event - mixed\ weight$$

mbowry@triumf.ca