CCπ^0 update

Robert Nelson
2009.8.6

- Briefly discuss unfolding. You've already heard enough about it.....
- Two unfolding examples
- Observed CCπ^0 events
- Efficiencies

A note on unfolding with a TH2D histogram

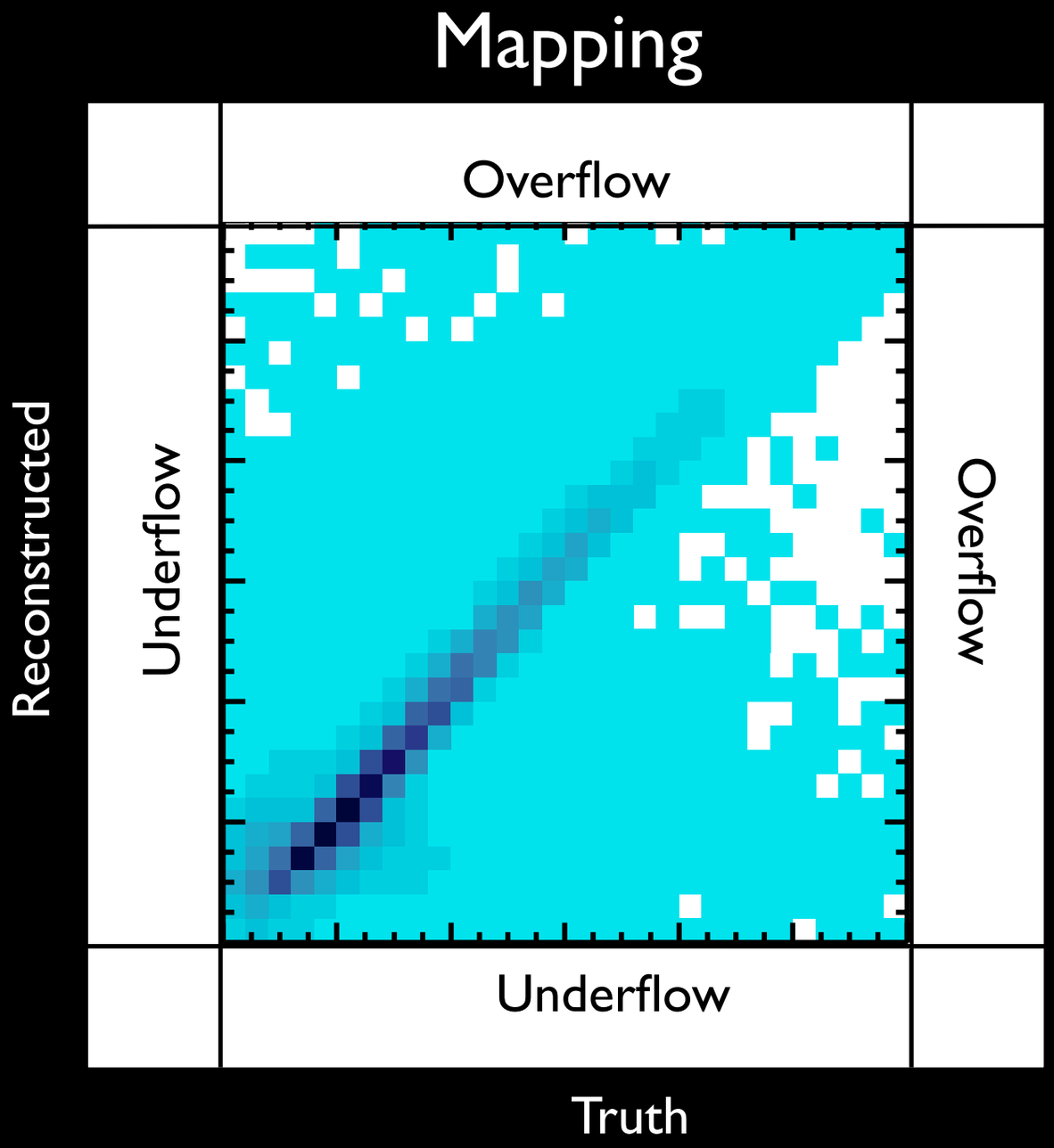
- The bin migration matrix, B, is the transpose of the column normalized matrix found by plotting a reconstructed quantity vs. its truth.

- The unfolding matrix, U, is the row normalized matrix found from the same mapping.

- To get an inferred true distribution, I, one can either do:
  $I = B^{-1} R$
  $I = U R$
  also, by construction, $T = U B T$.

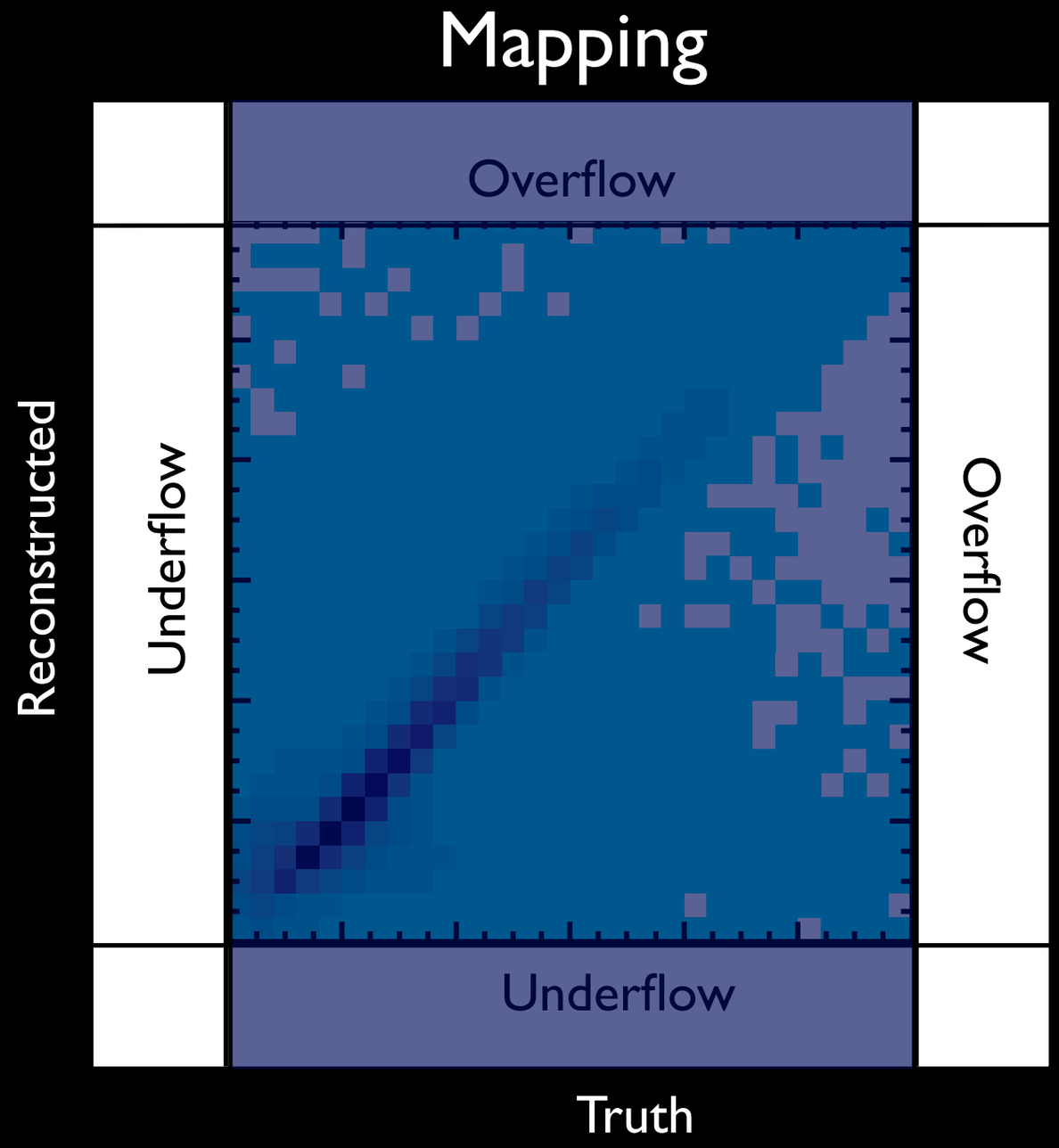
- The bin migration suffers from not always being invertible, but is unbiased.

- The unfolding matrix is slightly biased, but in the end is more robust do to not having to invert the matrix.



A note on unfolding with a TH2D histogram

- I 'lifted' the code to do the unfolding from Ryan's track-based analysis code from cvs.
- The track-based analysis used only the highlighted portion of the mapping to form U. Email discussion on next slide.
- I noticed some slight discrepancies in the unfolding that I attributed to not using the full histogram with all under/overflows included.
- I modified the code to include all over and underflows.
- The mapping now works perfectly.



Email exchange with Ryan circa 2007

Hi,

U/Oflows in the x direction are uninteresting since the f(E) function cuts off in both directions. (For E>Emax, I use f(E)=f(Emax), and similarly for E<Emin.) Or at least, that was the idea with this implementation. Other choices are possible.

You're correct that the 0th y row has no entries and doesn't contribute any information.

The (Ny+1)th row has a non-trivial distribution of x values, so that overflow bin is the only overflow row that contributes anything interesting, in proportion to the number of events that overflowed the EnuQE axis.

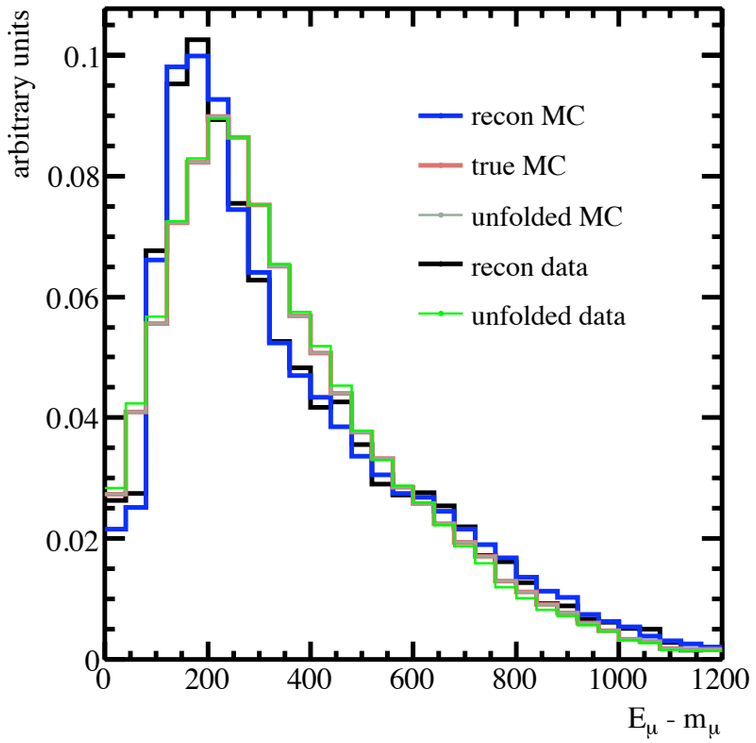
--Ryan

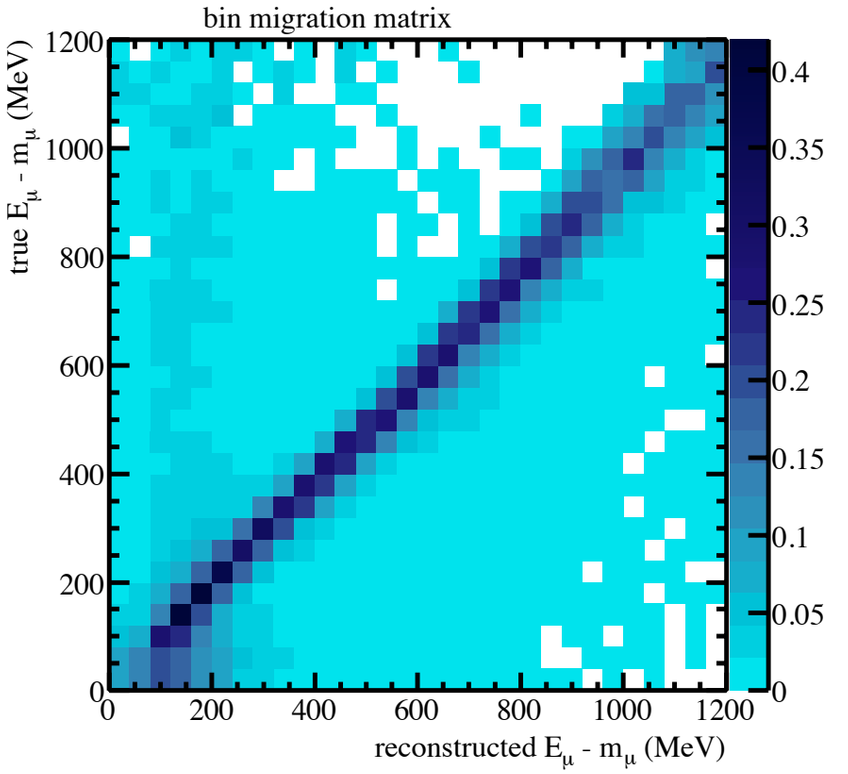
On Thu, 15 Feb 2007, Robert H. Nelson wrote:

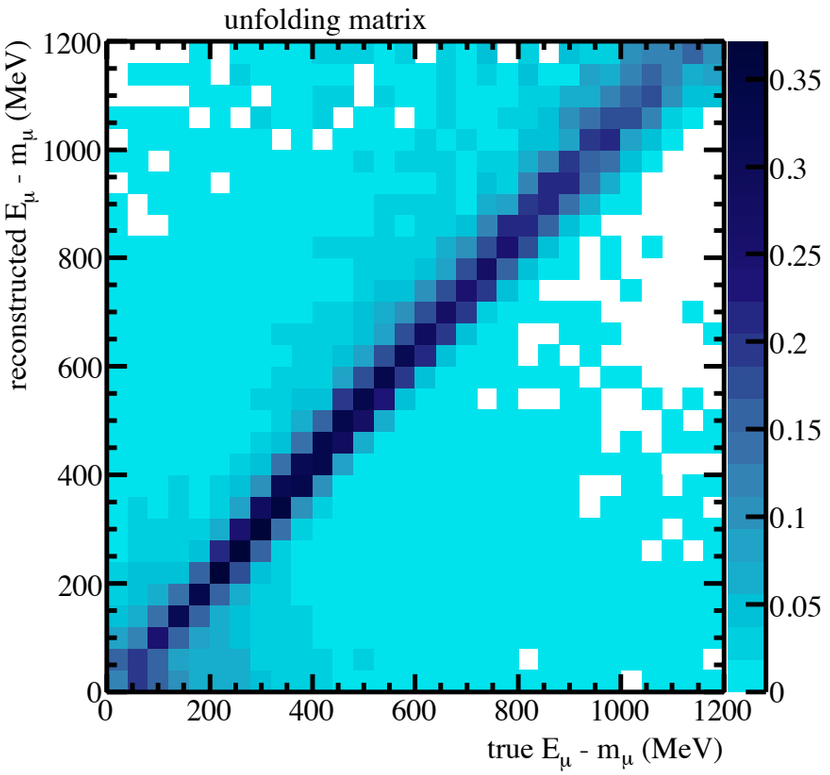
```
> Hi,
>
> I have question about the handling of over/underflows in the code.  In the
> function normalize_mapping() the code goes:
>
> Int_t normalize_mapping(TH2D &mapping ) {
>
>  Double_t norm;
>  Int_t xbins = mapping.GetNbinsX();
>  Int_t ybins = mapping.GetNbinsY();
>
>  // Oflows and Uflows, too
>  for (Int_t iY=0;iY<=ybins+1;iY++) {
>    norm = mapping.Integral(1,xbins,iY,iY,"");
>    if (norm>0) {
>      for (Int_t iX=1;iX<=xbins;iX++) {
>       mapping.SetBinContent(iX,iY,mapping.GetBinContent(iX,iY)/norm);
>      }
>    }
>  }
>
>  return 0;
> }
>
> Focus where you handle the under/overflows in the y direction and not the
> x.  If I'm not mistaken, the y-axis is the recontructed-E and the x-axis
> is truth-E.  In section 2.1 of the "Correcting the numu and
> pi->mu->nue..." technote it is stated that a cut on EnuCCQErecon > 150 MeV
> is applied. To me, this would seem that the underflow bin in y would then
> be identically 0. And that the overflow bin will be set to 1 (which I
> guess is fine).
>
> However, I was under the impression that the interesting under/overflow
> bins were in the other direction.
>
> Is my thinking correct?  If not, why?
>
> Thanks,
> Bob
>
```

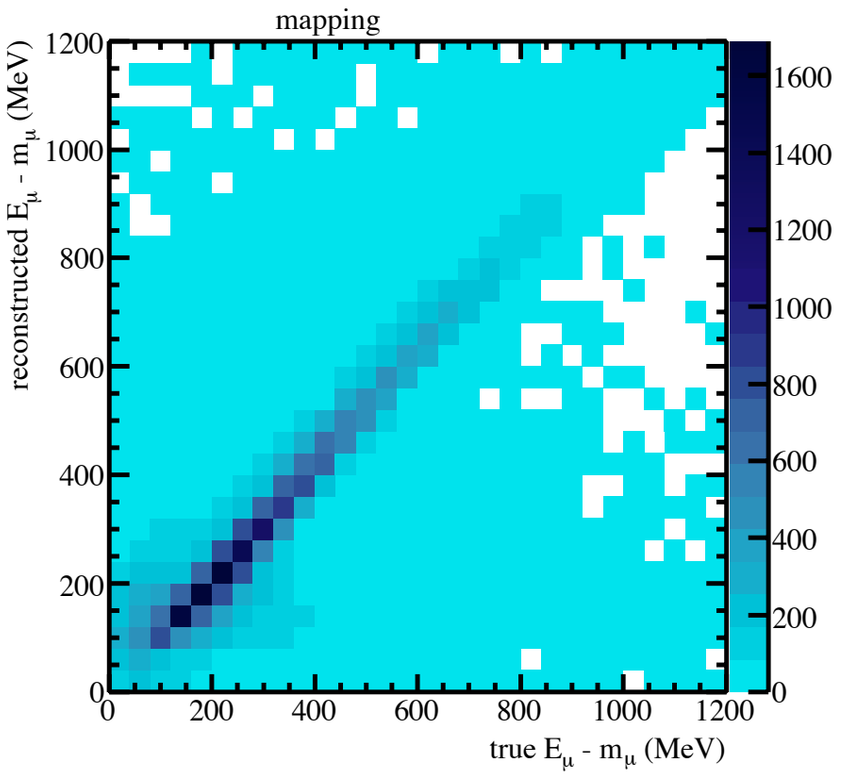
E_μ unfolding

- The mapping histogram is reconstructed vs true.
- The bin migration is the transpose of the column normalized mapping.
- The unfolding matrix is the row normalized mapping.
- The unfolded reconstructed histogram perfectly matches the true histogram.
- The unfolded data matches the true MC quite well.

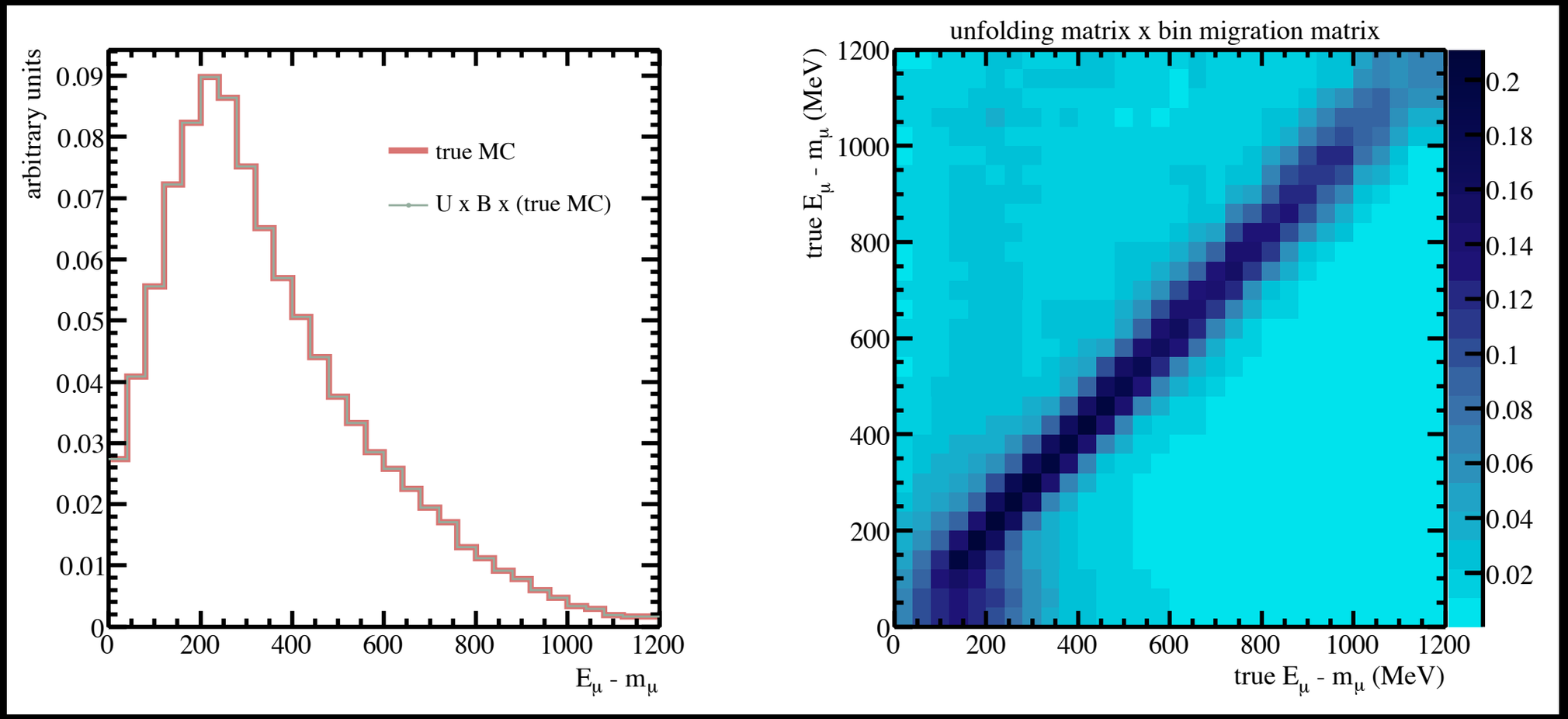








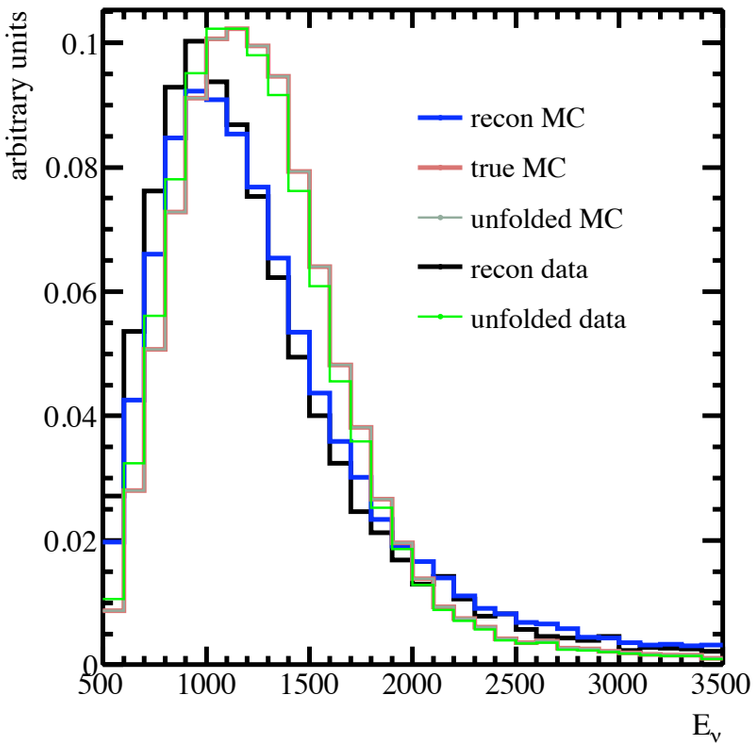
Eigenvector test

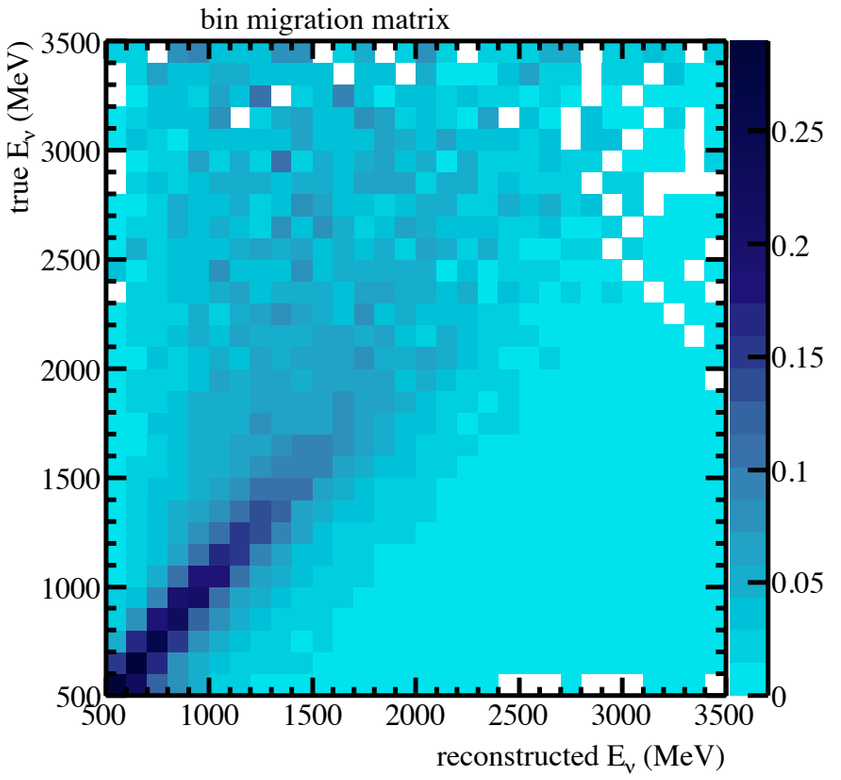


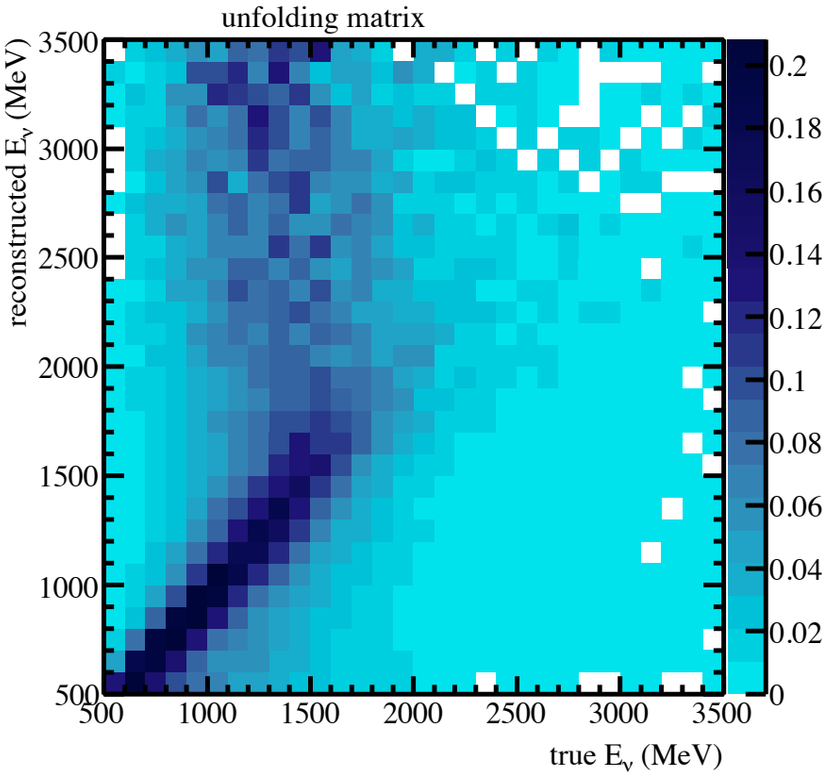
- The true histogram is an eigenvector of the product of the unfolding matrix times the bin migration matrix with eigenvalue of 1.

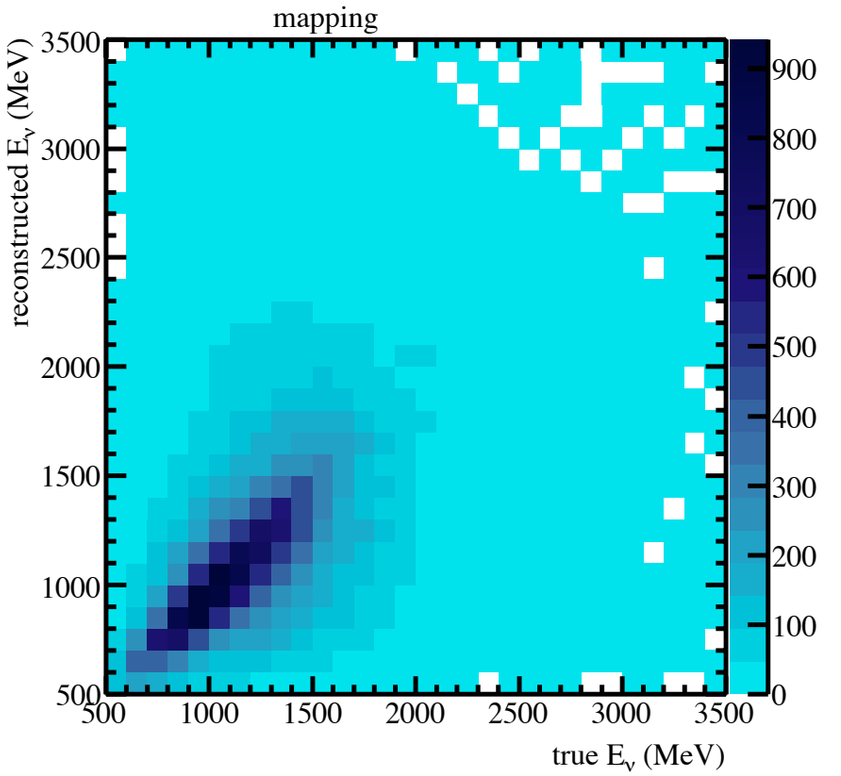
E_ν unfolding

- The mapping matrix is reconstructed vs true.
- The bin migration is the transpose of the column normalized mapping.
- The unfolding matrix is the row normalized mapping.
- High reconstructed neutrino energies don't look so good. The statistics are smaller there anyways.

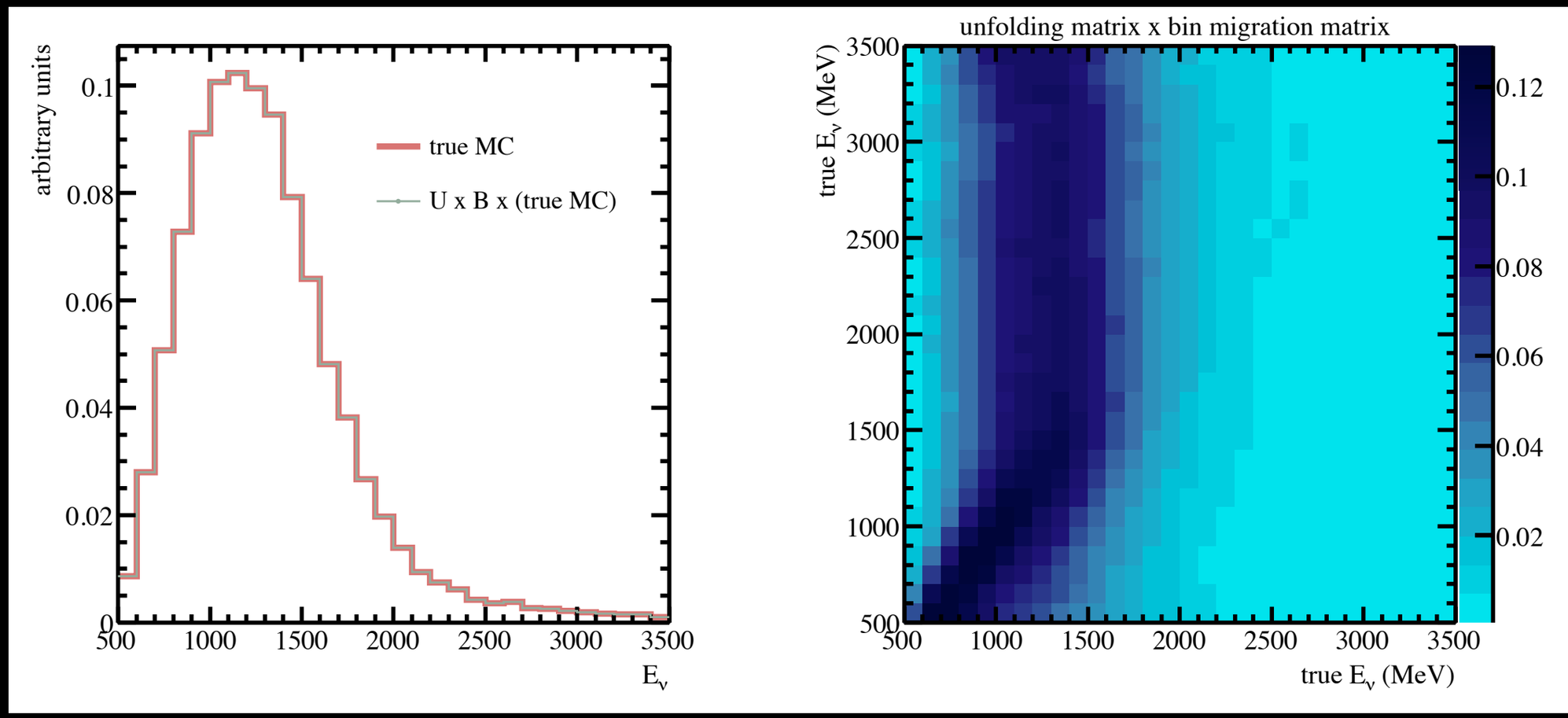








Eigenvector test

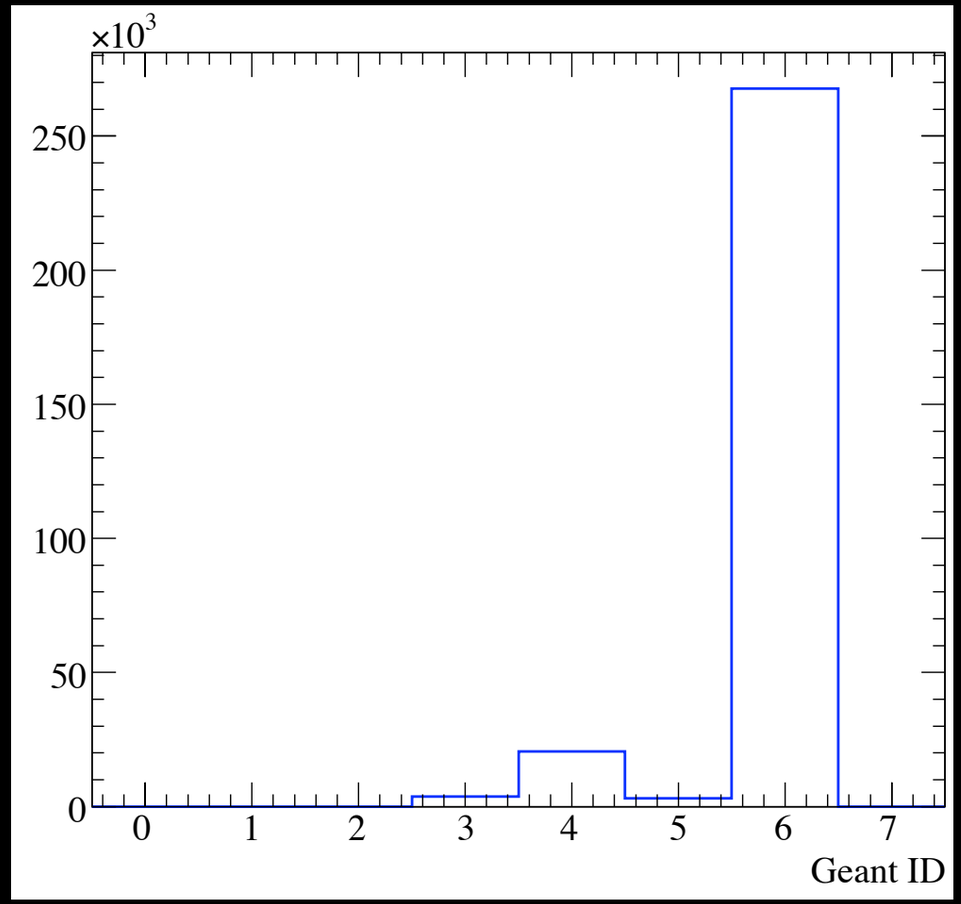


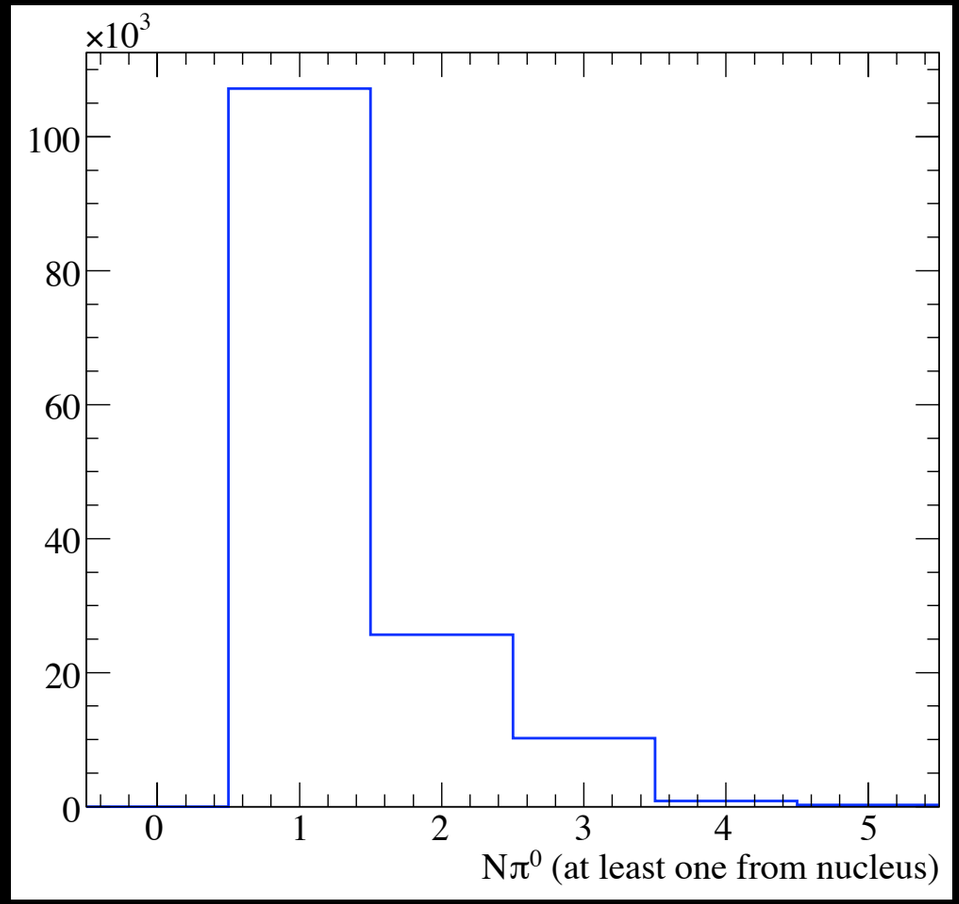
- Nasty looking matrix.
- Kind of neat that it still is an eigenvector.

- The unfolding is now coded up and working and it satisfies the closure test.
- The code can also handle non-square mappings, and variable bin widths quite easily.

Observed CCπ^0

- Currently, observed CCπ^0 events are any event that has at least one π^0 that came from the nucleus.
- These include CC electron events, and multi-π.
- Should we restrict the definition to muon events with only a single π that came from the nucleus?



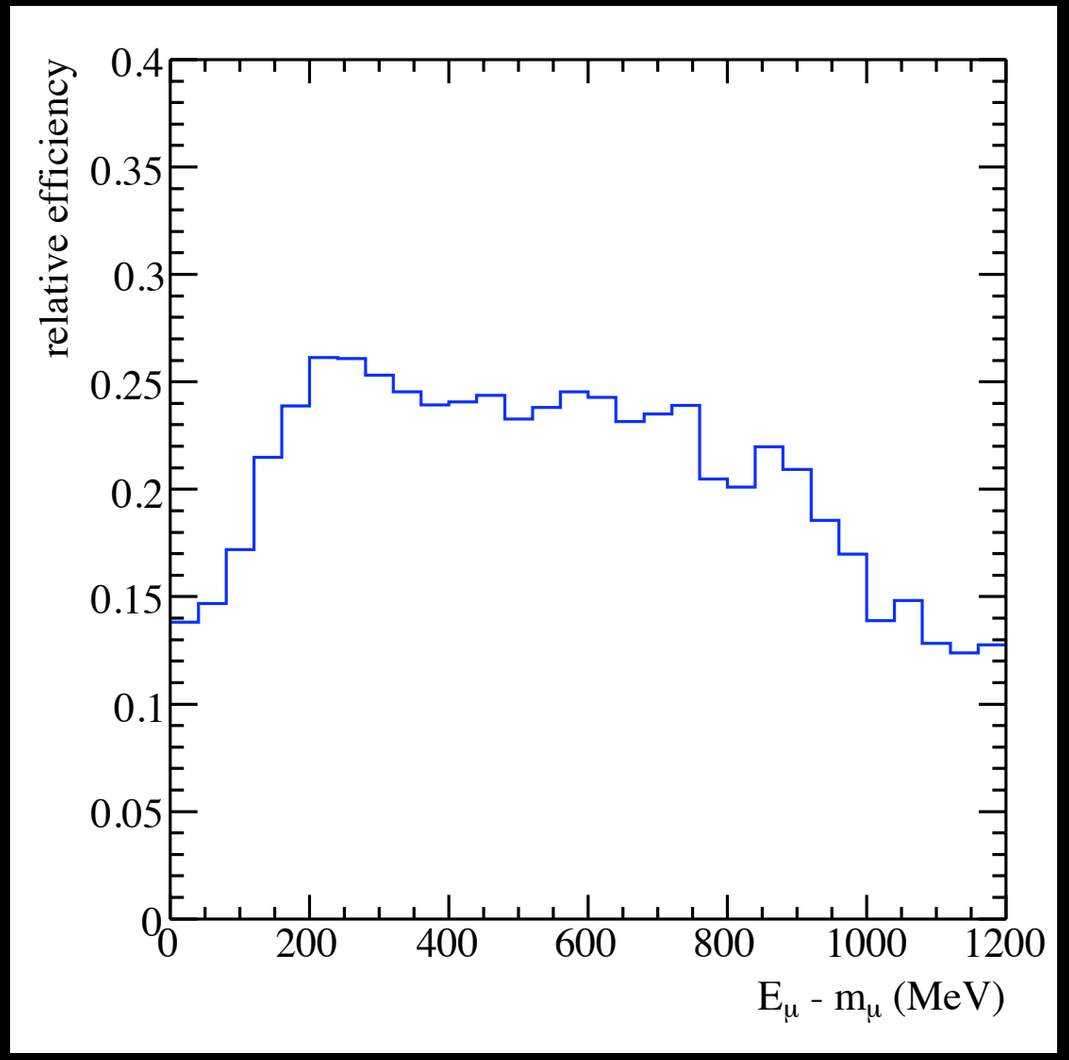


Efficiencies

- The relative efficiency is defined as the number of true observed CCπ^0 after cuts / same sample before analysis cuts after applying the 2SE filter and Martin's CCπ^0 filter.

- That filter keeps over 90% of observed CCπ^0 events. So the average efficiency shouldn't change too much from this.

- The full efficiency wasn't finished processing on the full CV in time for this meeting.



Remaining steps...

- Calculate the number of targets.
- Subtract backgrounds.
- Get the flux.
- Decide on binnings.
- Calculate cross-sections.
- Run all this code on all the multisims.