Pulses & Weights

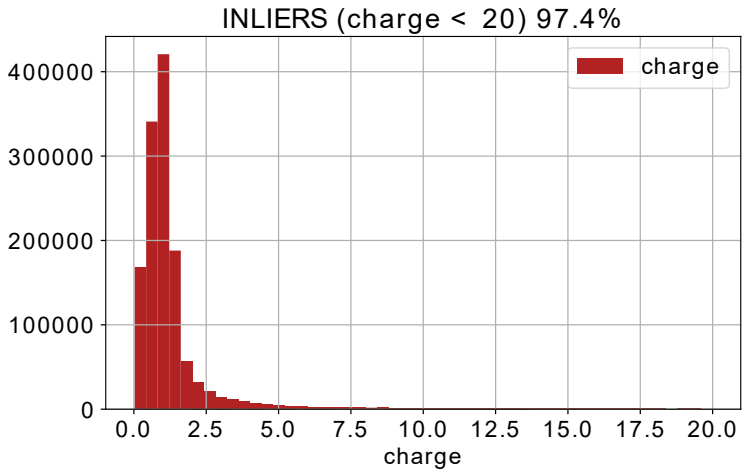
Kasper Pedersen, Msc. Student
Icecube, NBI

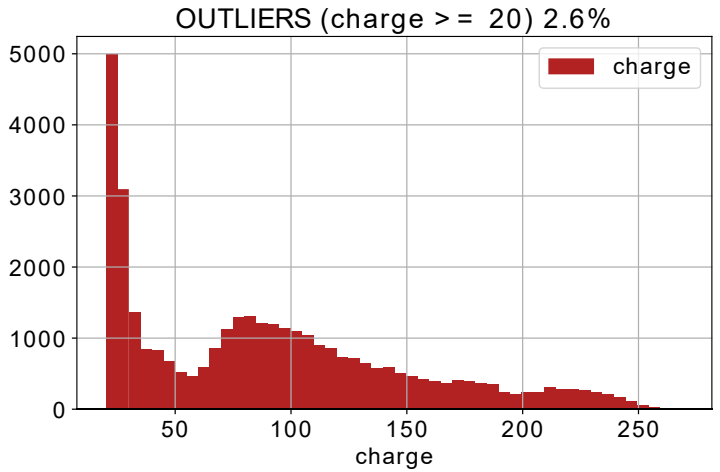
UNIVERSITY OF COPENHAGEN

Recap

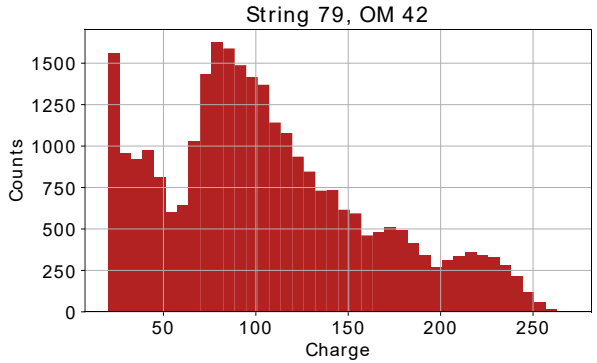
- Found high charge pulses (charge >= 20) in MC test data in SRTTWOfflinePulsesDC

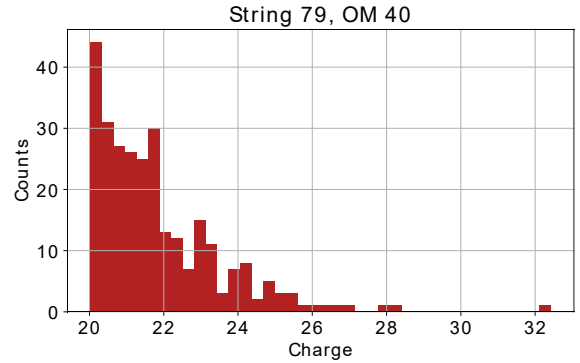
- Started digging

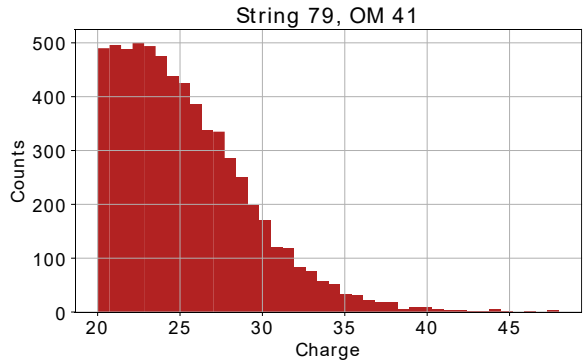




High Charge Pulse distibutions in OMs







Generating Test MC

- Step 1: Make a particle
  - e^-, 50Gev in DeepCore
- Step 2: Propagate Photons
  - Some weird stuff happens, explanation in next slides
- Step 3: Create hits from photons
  - More weird stuff
- Level 1: Filtering
- Level 2: More Filtering and Processing

Generating Test MC

- Step 1: Make a particle
  - e^-, 50Gev in DeepCore

This is what I have been digging into

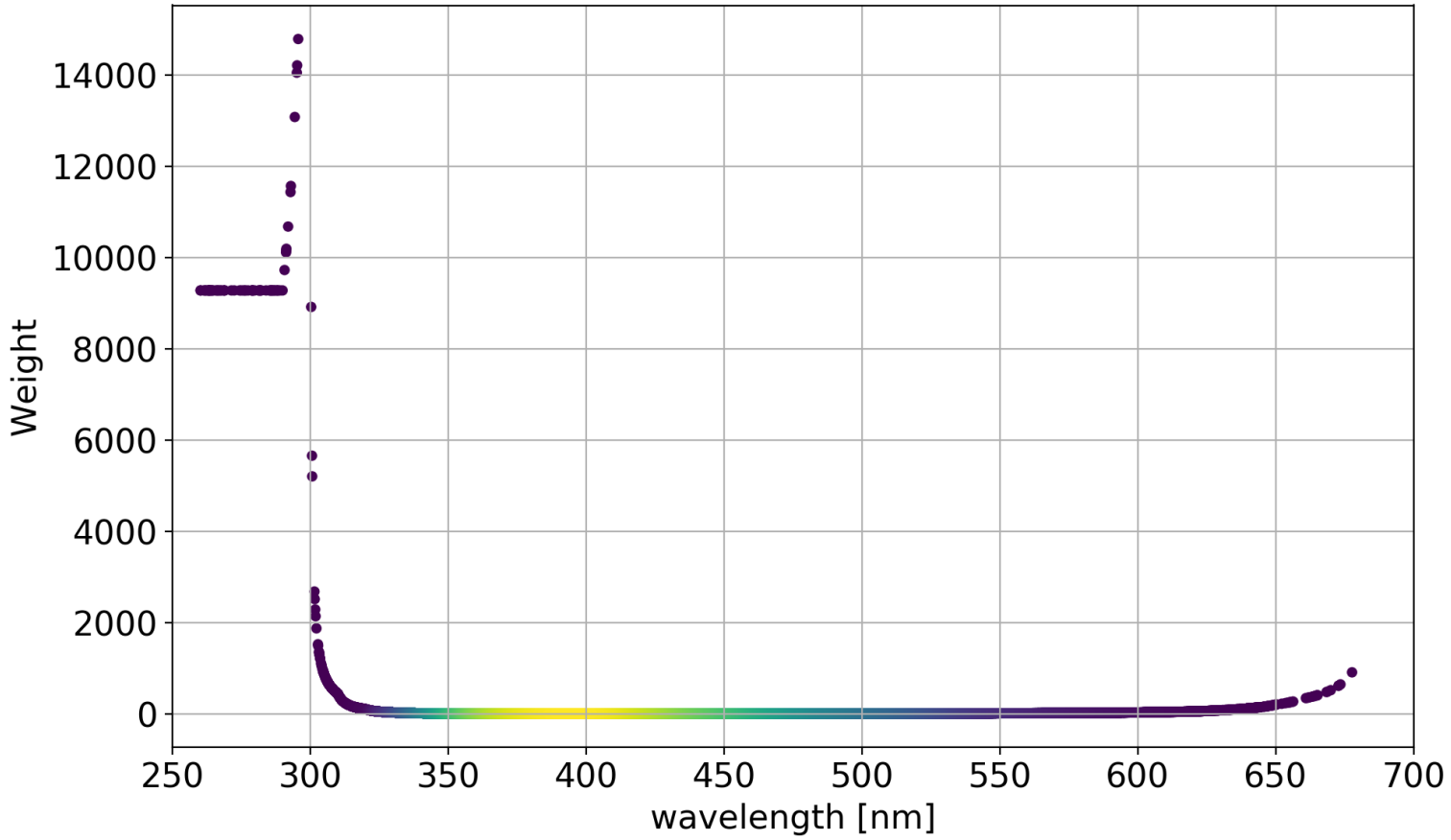
- Step 2: Propagate Photons
  - Some weird stuff happens, explanation in next slides
- Step 3: Create hits from photons
  - More weird stuff
- Level 1: Filtering
- Level 2: More Filtering and Processing

Step 2 – Photon Propagation

- Each photon is given a weight
- If photon hits a DOM it is saved and has attributes:
  - Weight
  - Wavelength
  - Time (From creation to hit ?)
  - Direction
  - etc.

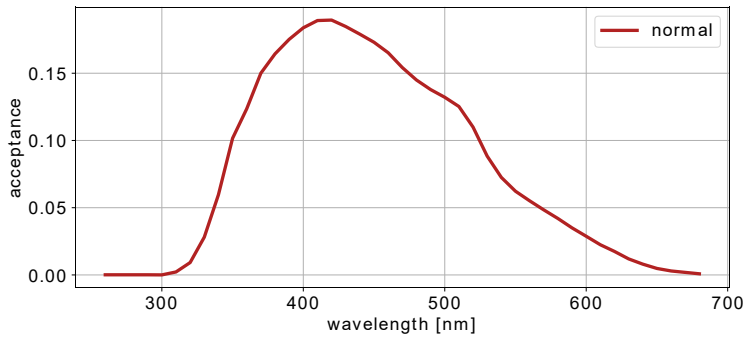
- Weights are used to calculate hitProbability so let's look at the weights of the photons

- Strange rise and plateauing below 300 nm

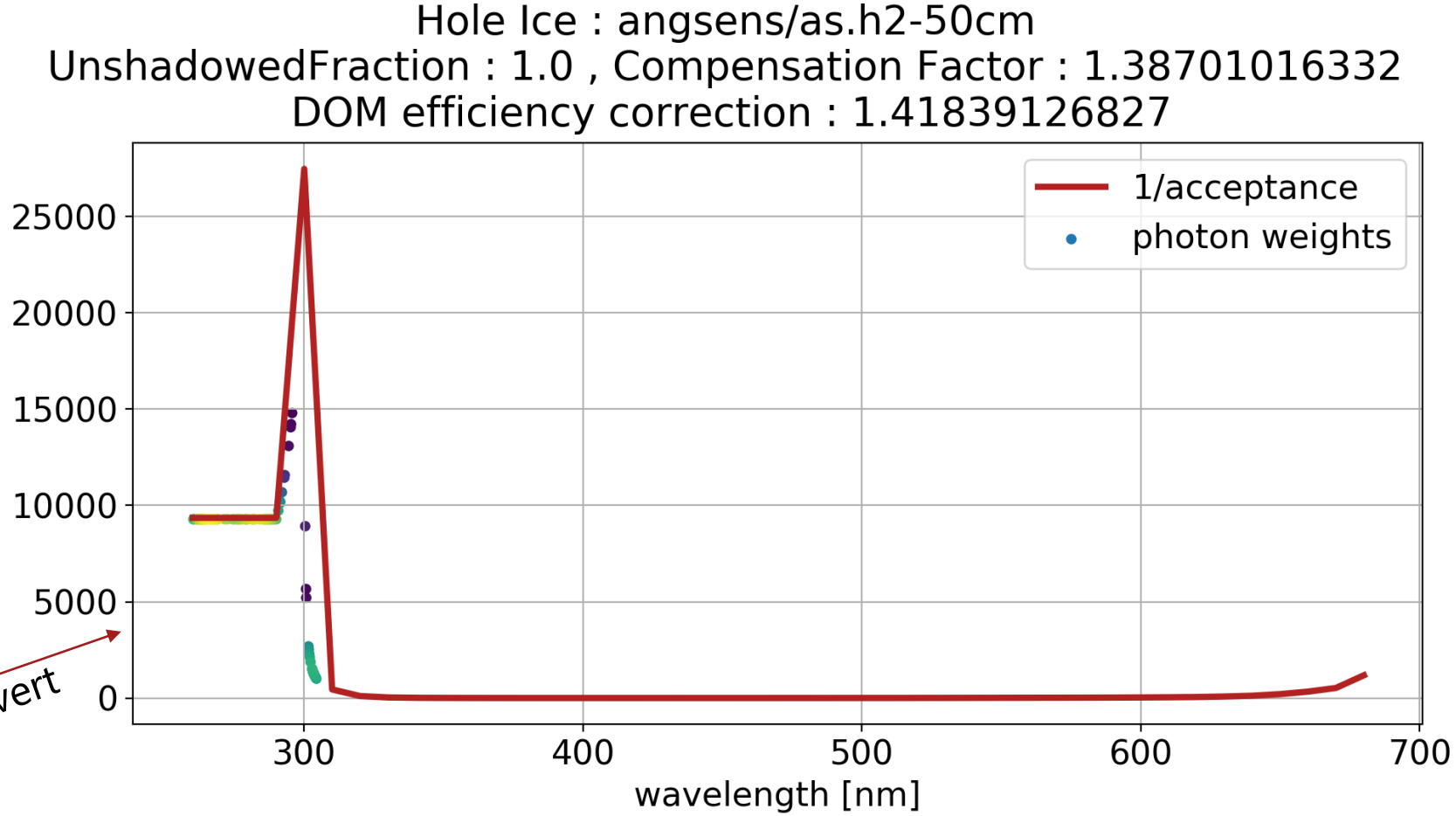


- Weights are used to calculate hitProbability so let's look at the weights of the photons

- Strange rise and plateauing below 300 nm
- Coincides with the inverse of the DOM wavelength acceptance
- Not same HoleIce as input ?!?

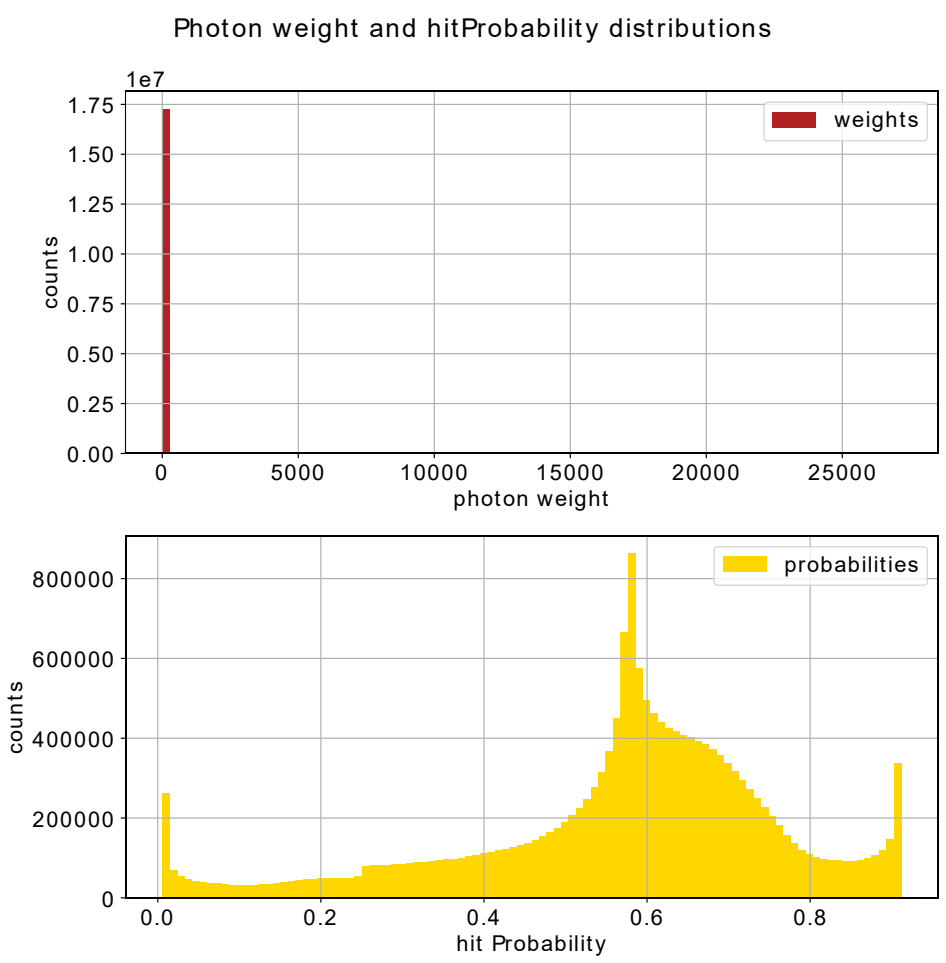


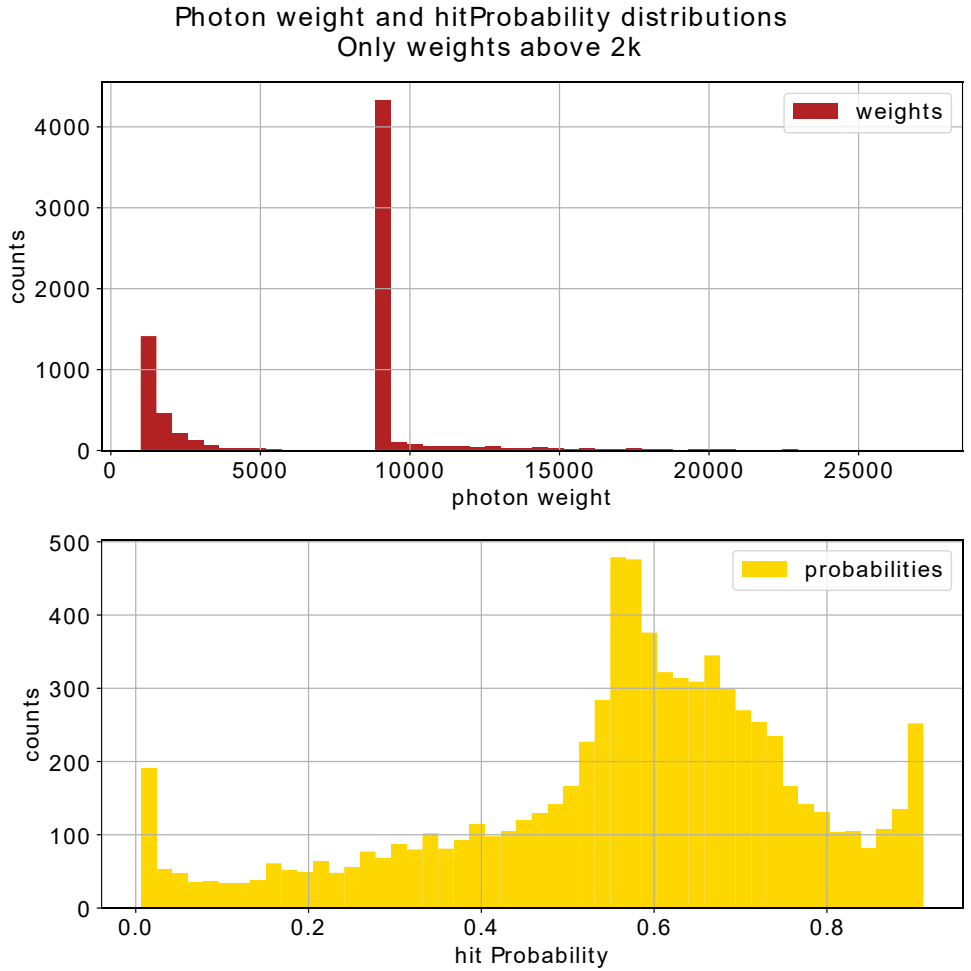
invert



But high weights has low probability of being a hit, so we're somewhat fine?

hitProbability and weight distributions



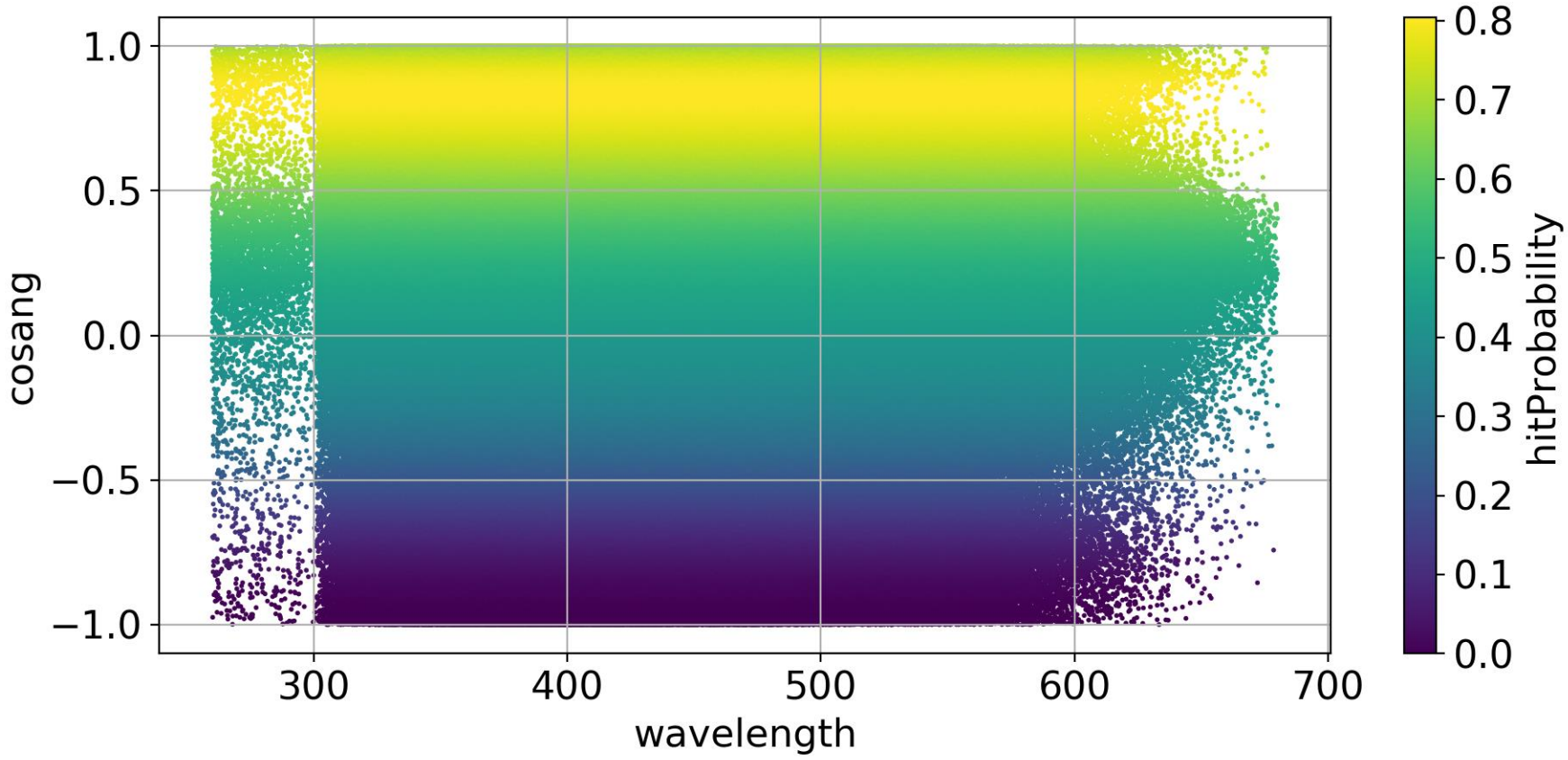


Step 3 – Make Hits from Photons

Let's look at the hit probability

$$hitProb = w_{photon} \cdot domAcceptance(\lambda_{photon}) \cdot domAngularSensitivity(\theta_{photon}) \cdot scalar$$

- But weight is inverse of domAcceptance so this is just 1?
  - No, a different holeIce model scales the domAcceptance



Ice: spice_3.2.1
Hole: flasher_p1_0.30_p2_-1

This needs to be investigated further

Back to the Pulses

- Step 3 process
  1. Make hits from photons
  2. Add noise (vuvuzuela)
  3. “Rosencrantz” PMTSimulation
  4. “Guildenstern” DOMLaucher

Back to the Pulses

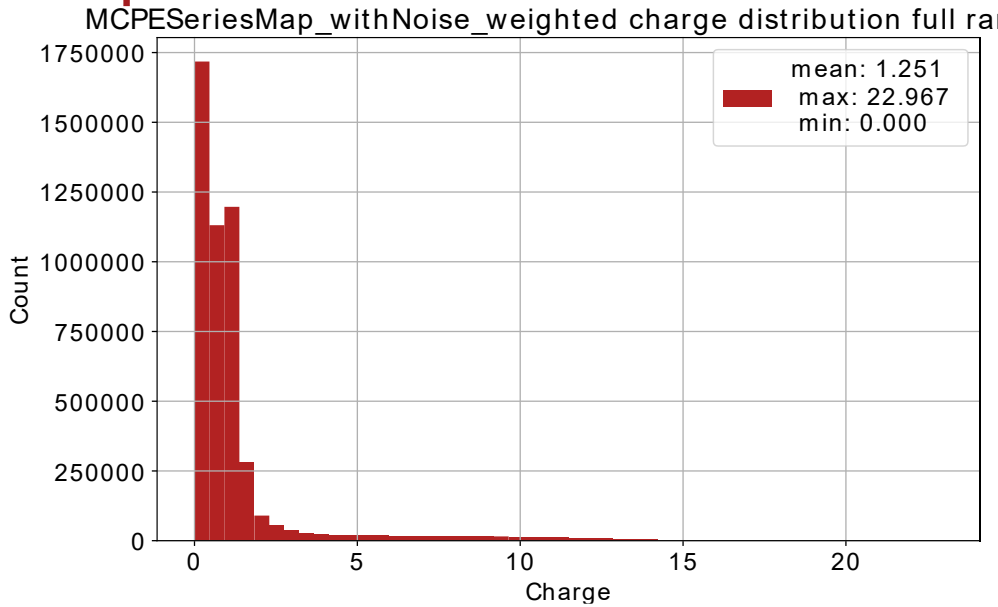
- Step 3 process
  1. Make hits from photons
  2. Add noise (vuvuzuela)
  3. "Rosencrantz" PMTSimulation ← Adds charge to the MCPE pulses
  4. "Guildenstern" DOMLaucher

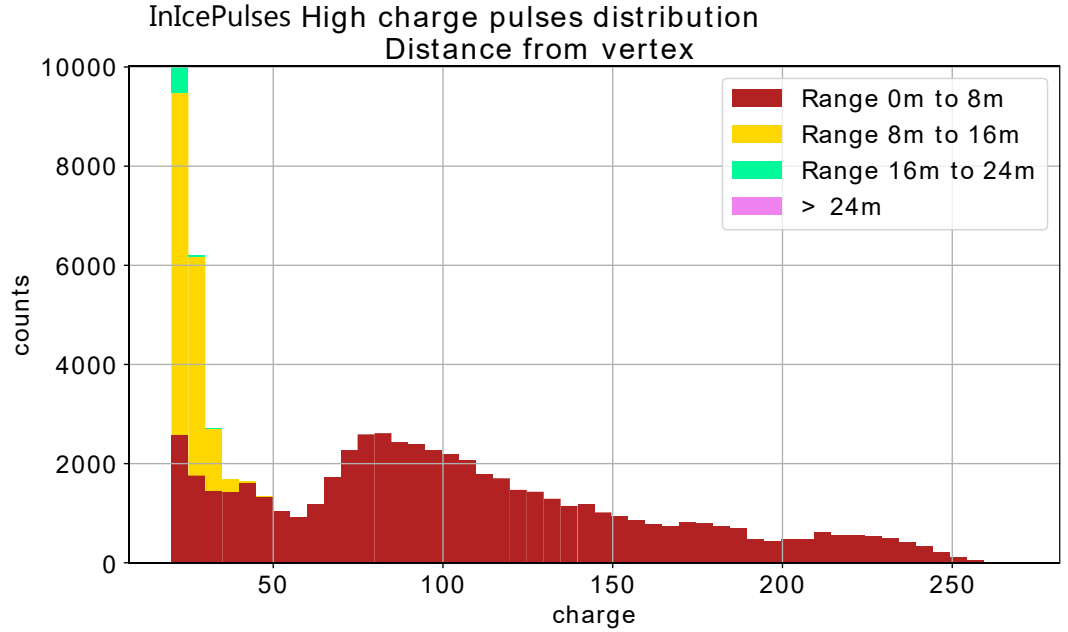
Do we see high charge pulses here?

Charge distribution in MCPE and Reco pulses

- No high charge pulses in MCPE pulses
- They first occur in Reco Pulses

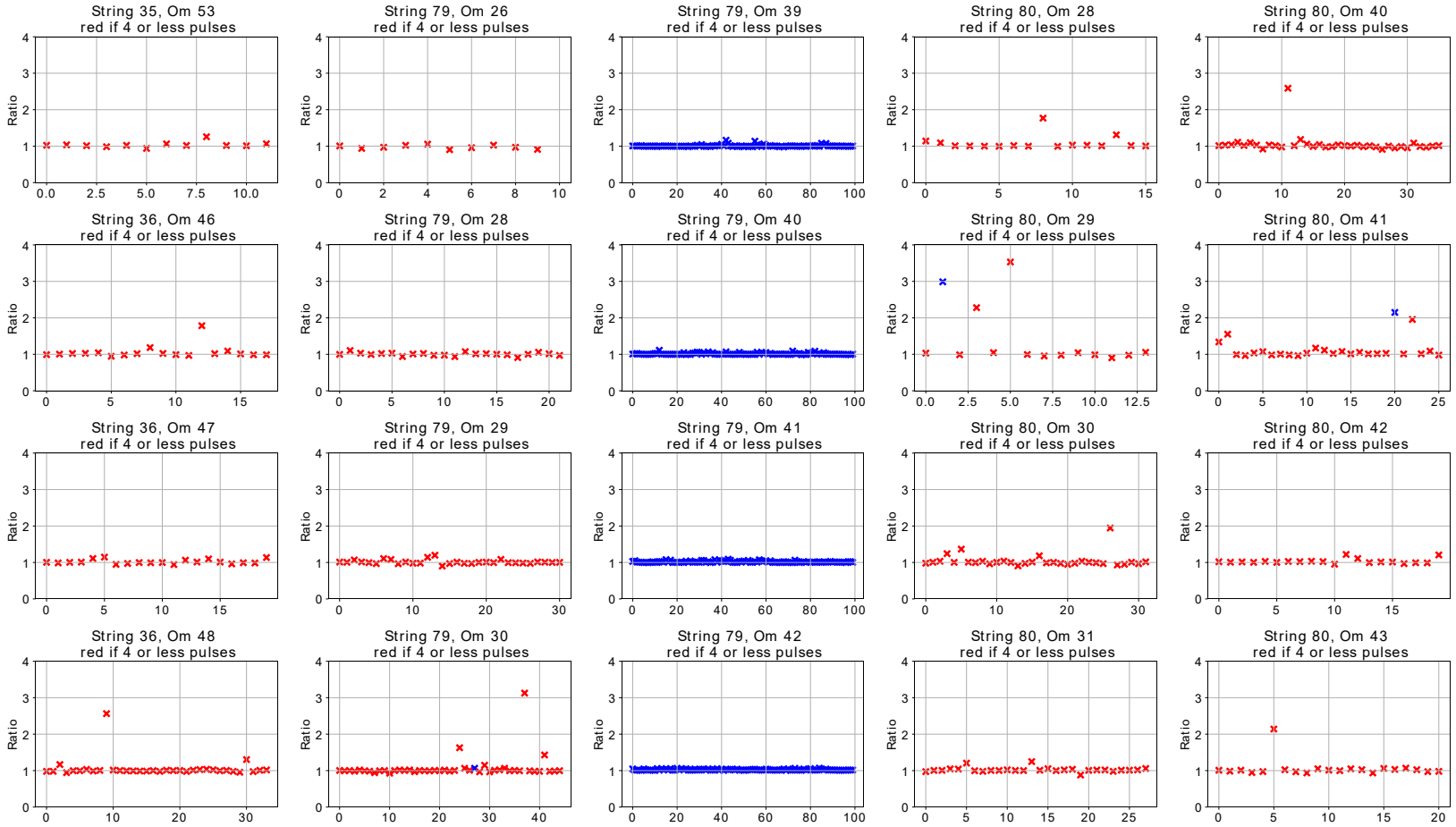
So are the RecoPulses wrong?





Charge conservation

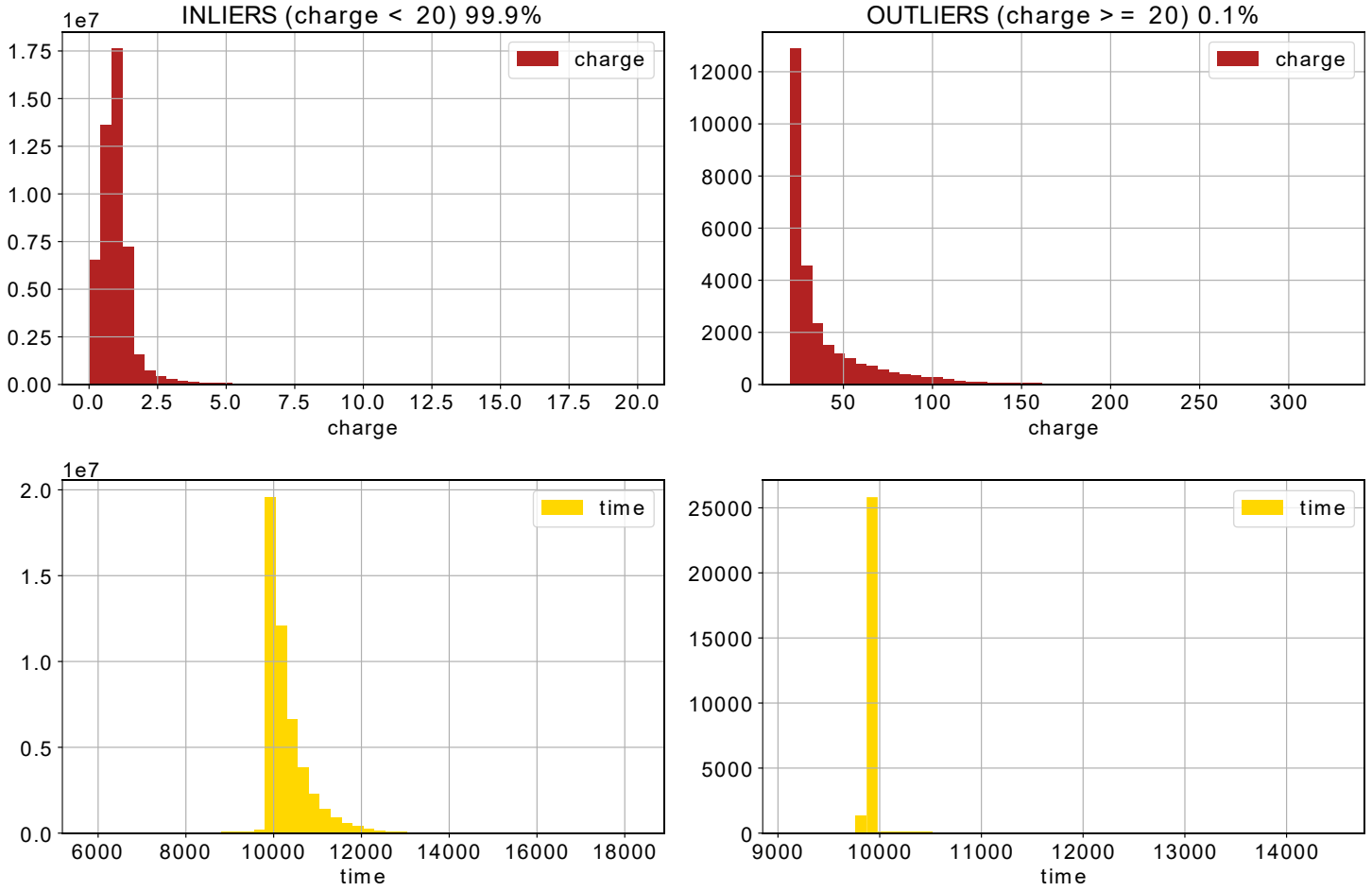
- To check how well the RecoPulses reconstructs, check the total charge pr event pr DOM
- $$\frac{\sum_{i,k} Q_{MCPE_i}}{\sum_{j,k} Q_{RecoPulse_j}} = C_k$$
- Looks good
- Few outliers



Compare to oscNext data

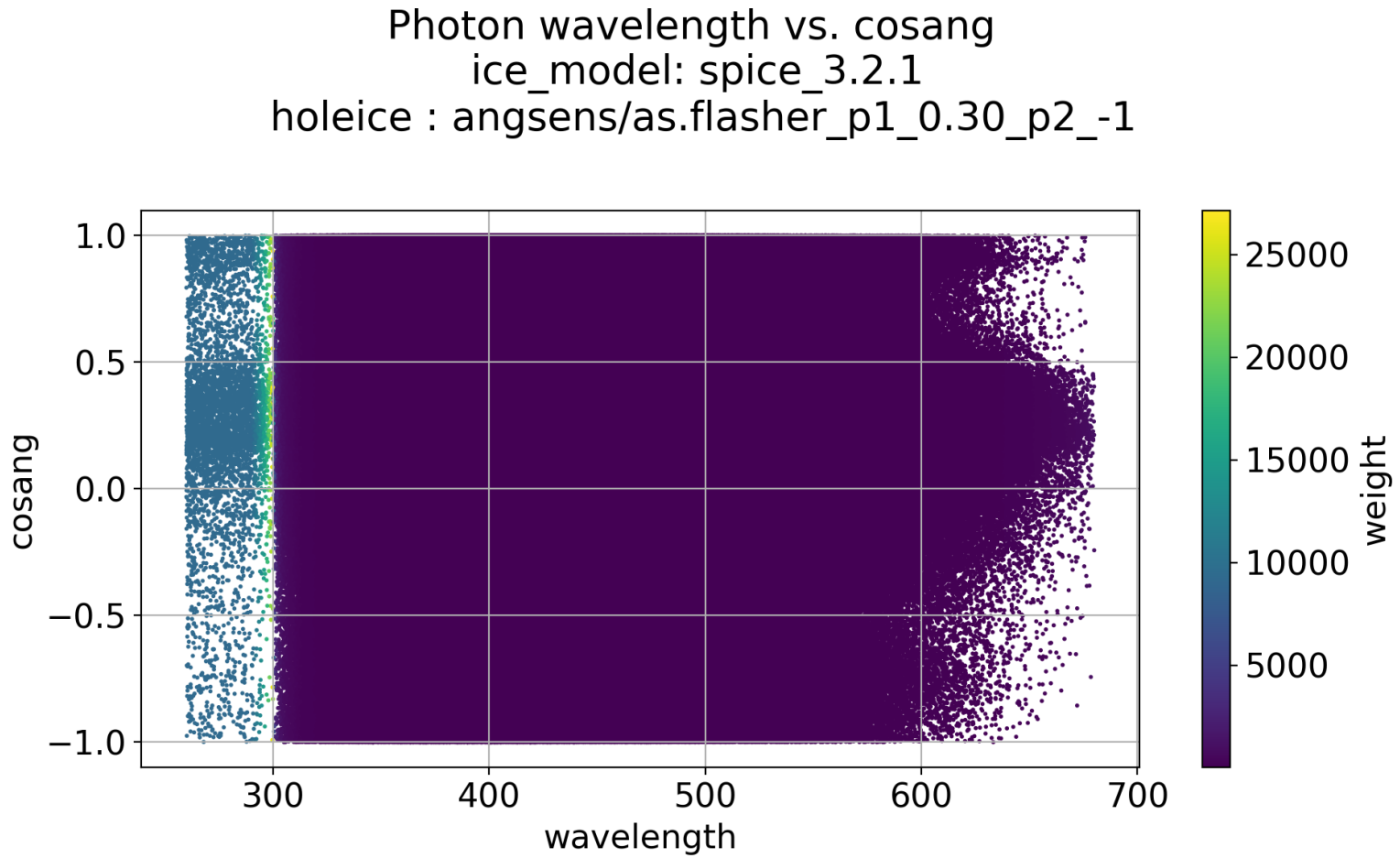
- See high charge pulses here as well but 0.1% (2.6% in our RecoPulses)
- Seems to be effect of having less RecoPulses than MCPEPulses but need to conserve overall charge

SRTTWOfflinePulsesDC
Data: /oscNext/pass2/genie/level7_v02.00/120000 (ν_e)

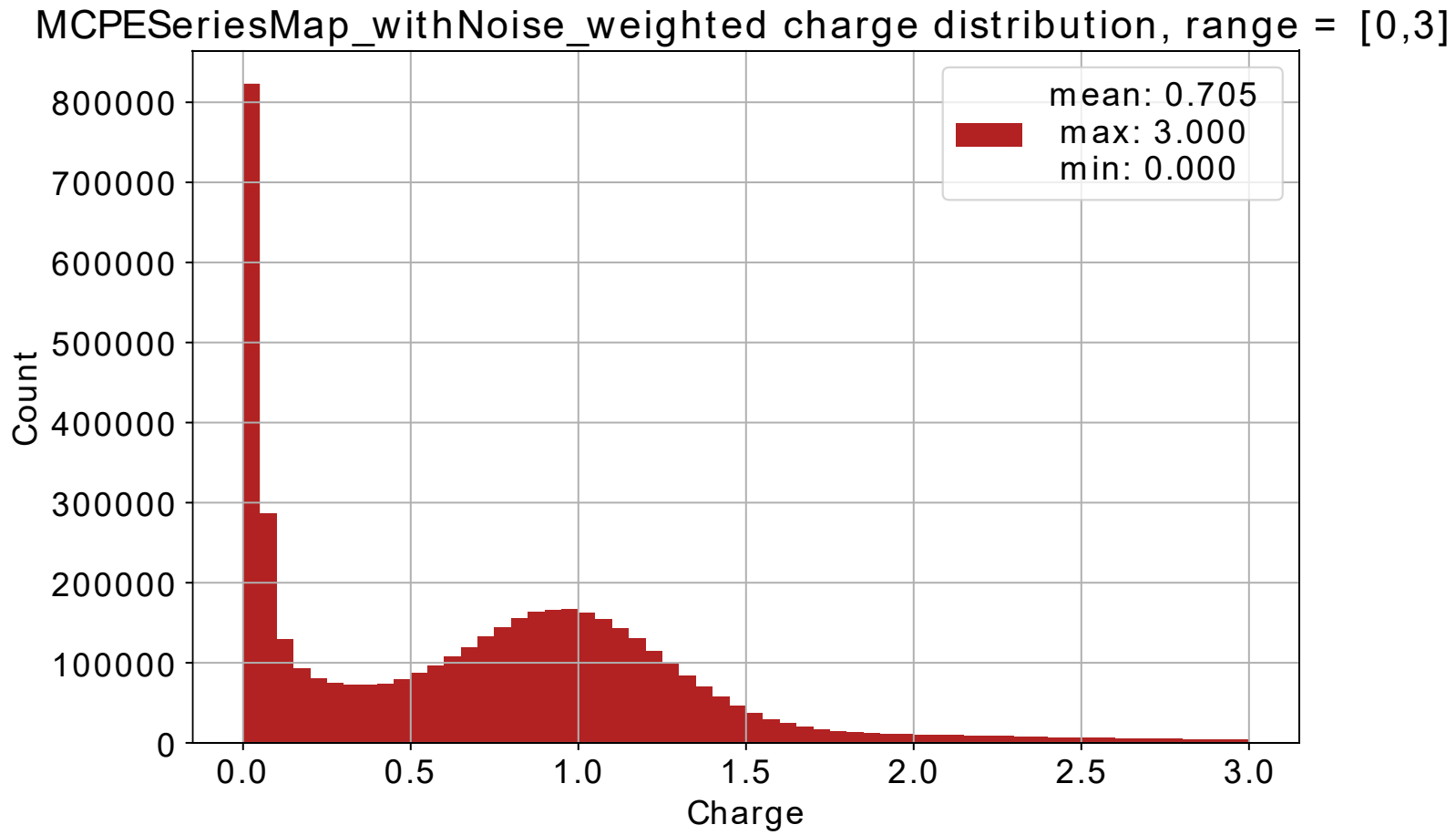


Backup

Wavelength vs cos(ang) with photon weight instead of hitProb

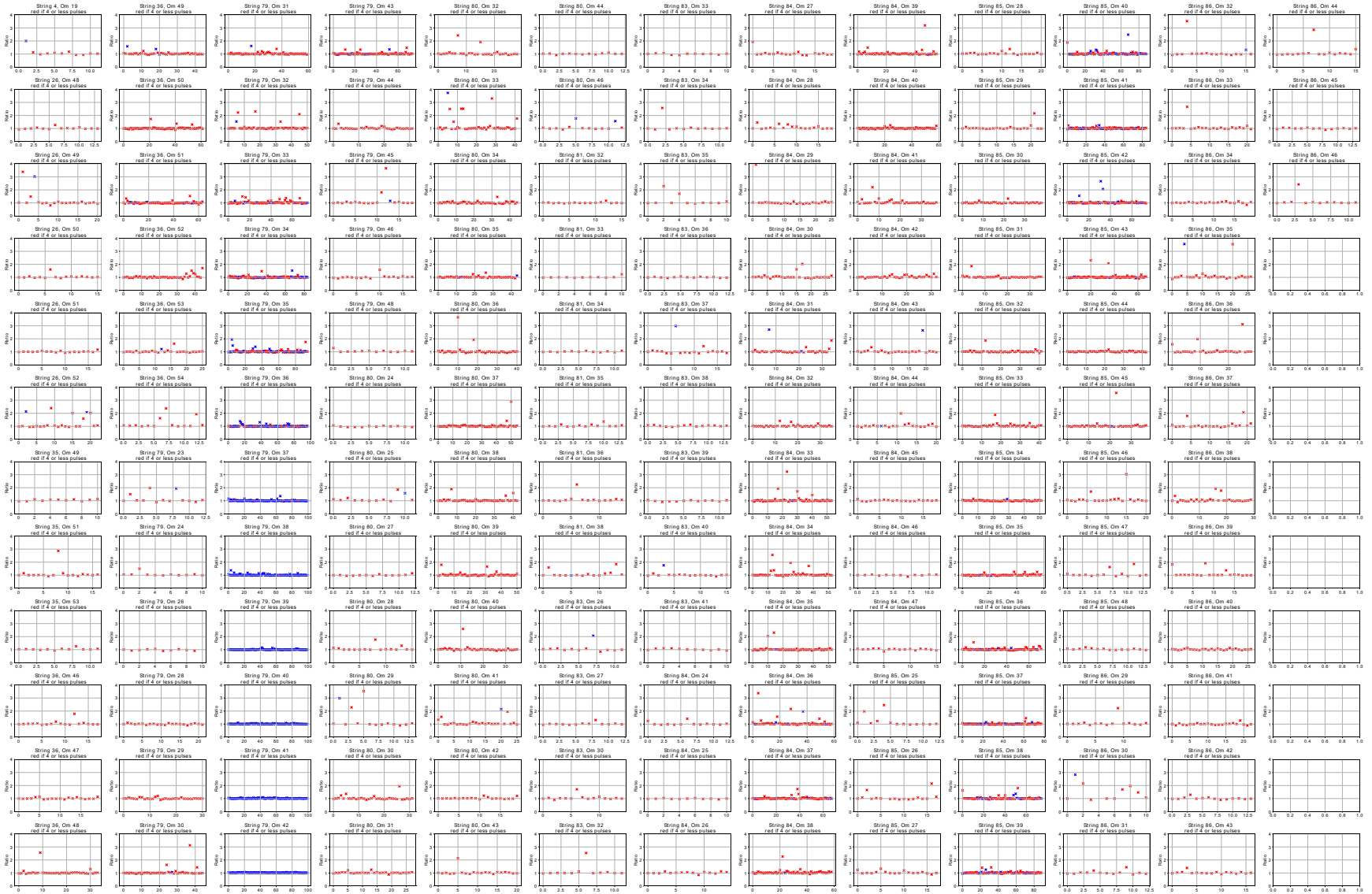


Zoom in on MCPEPulses charge distribution

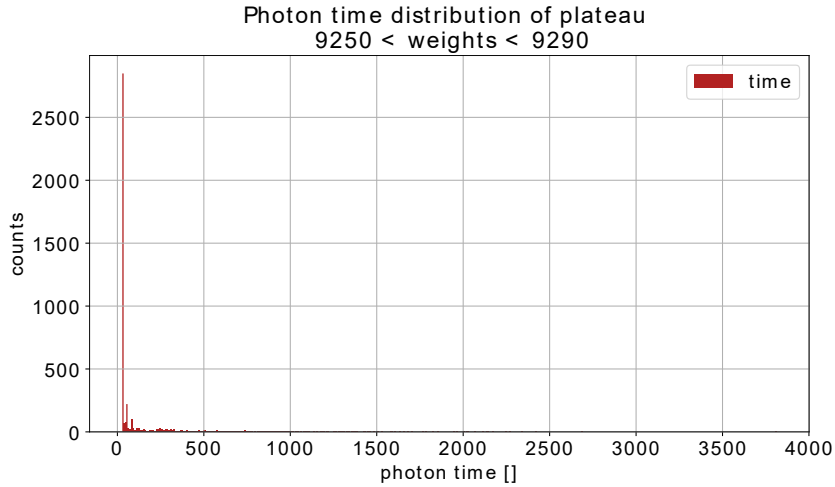


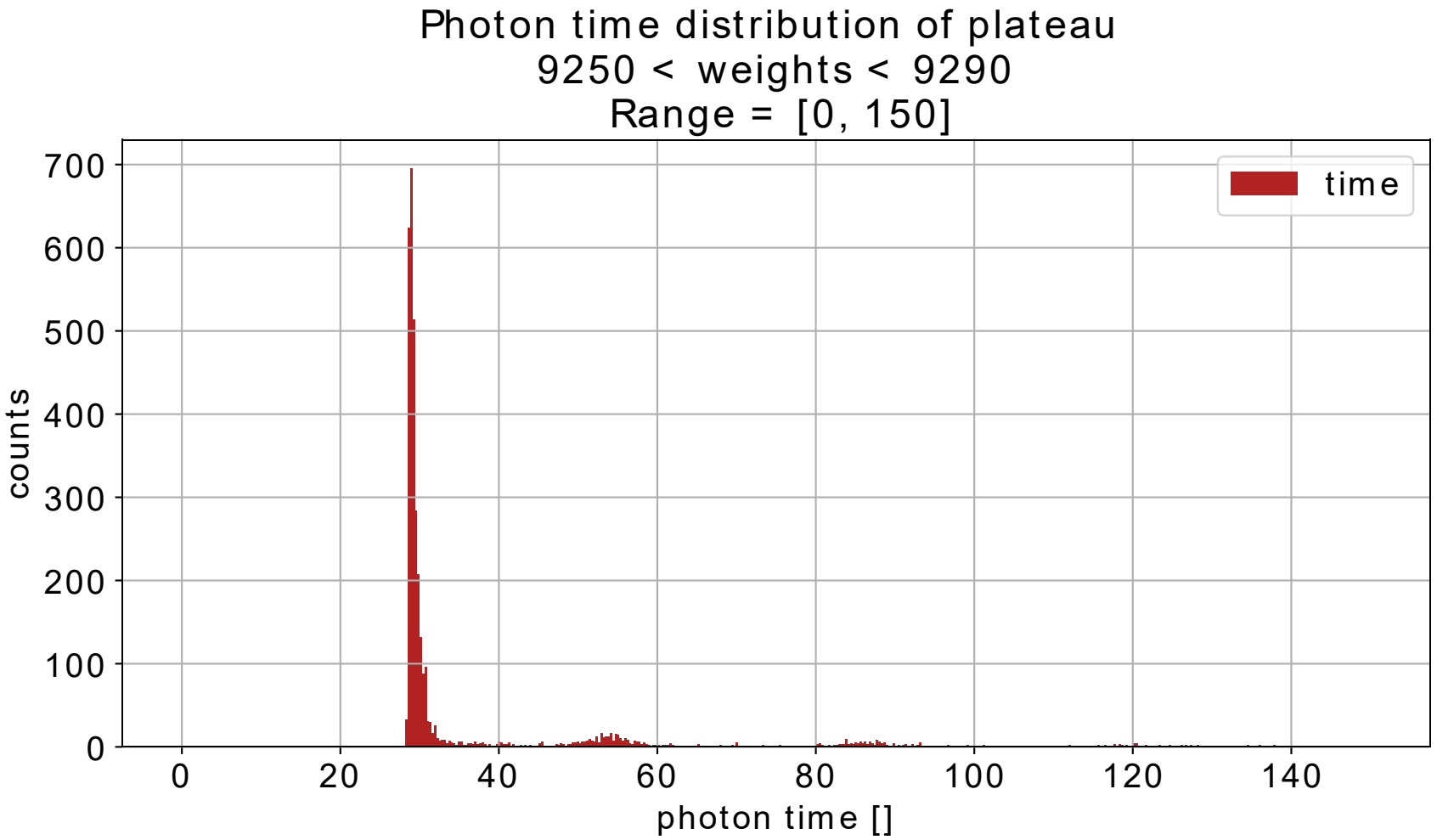
All Oms charge conservation test

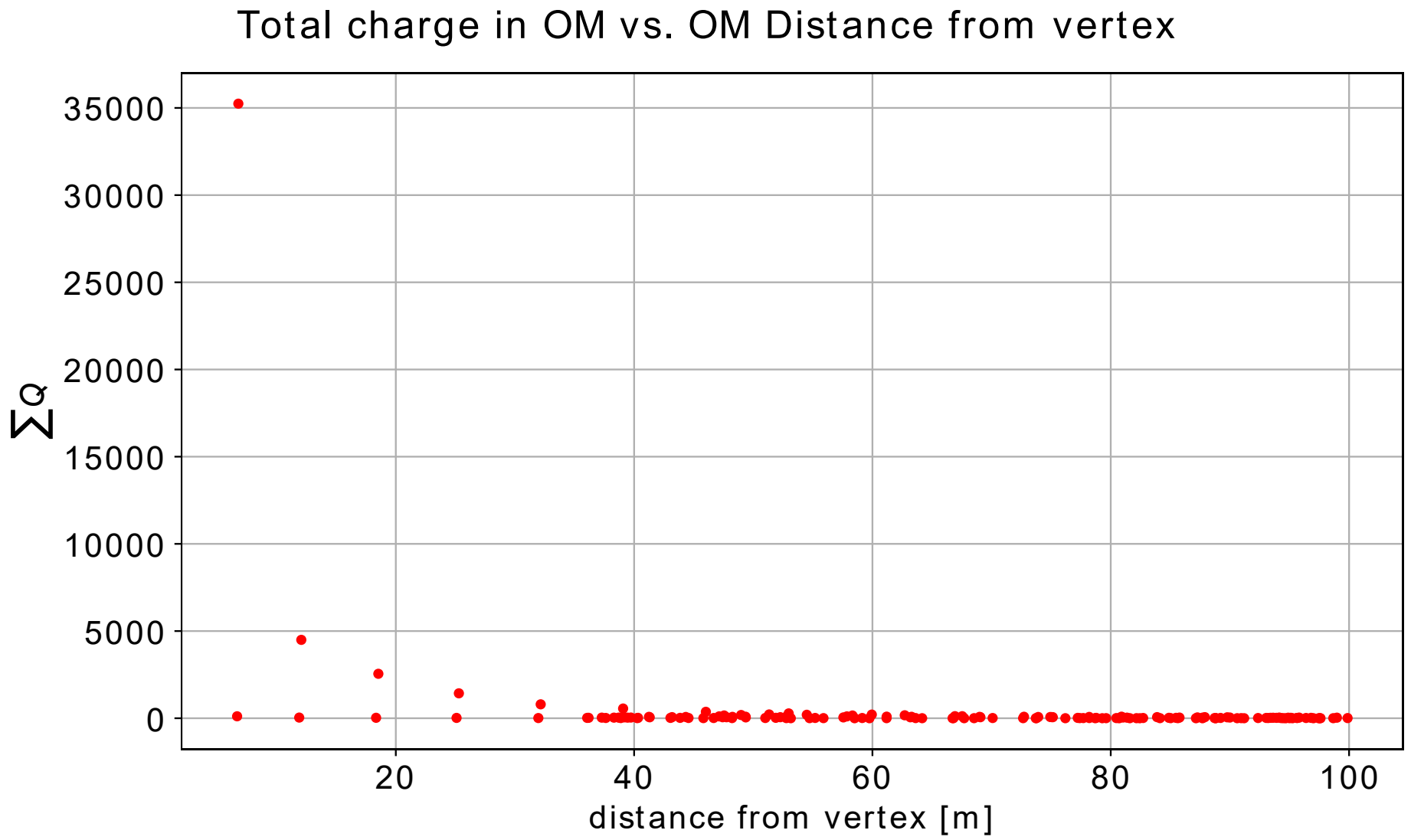
$$\frac{\sum_{i,k} Q_{MCPE_i}}{\sum_{i,k} Q_{RecoPulse_i}}$$



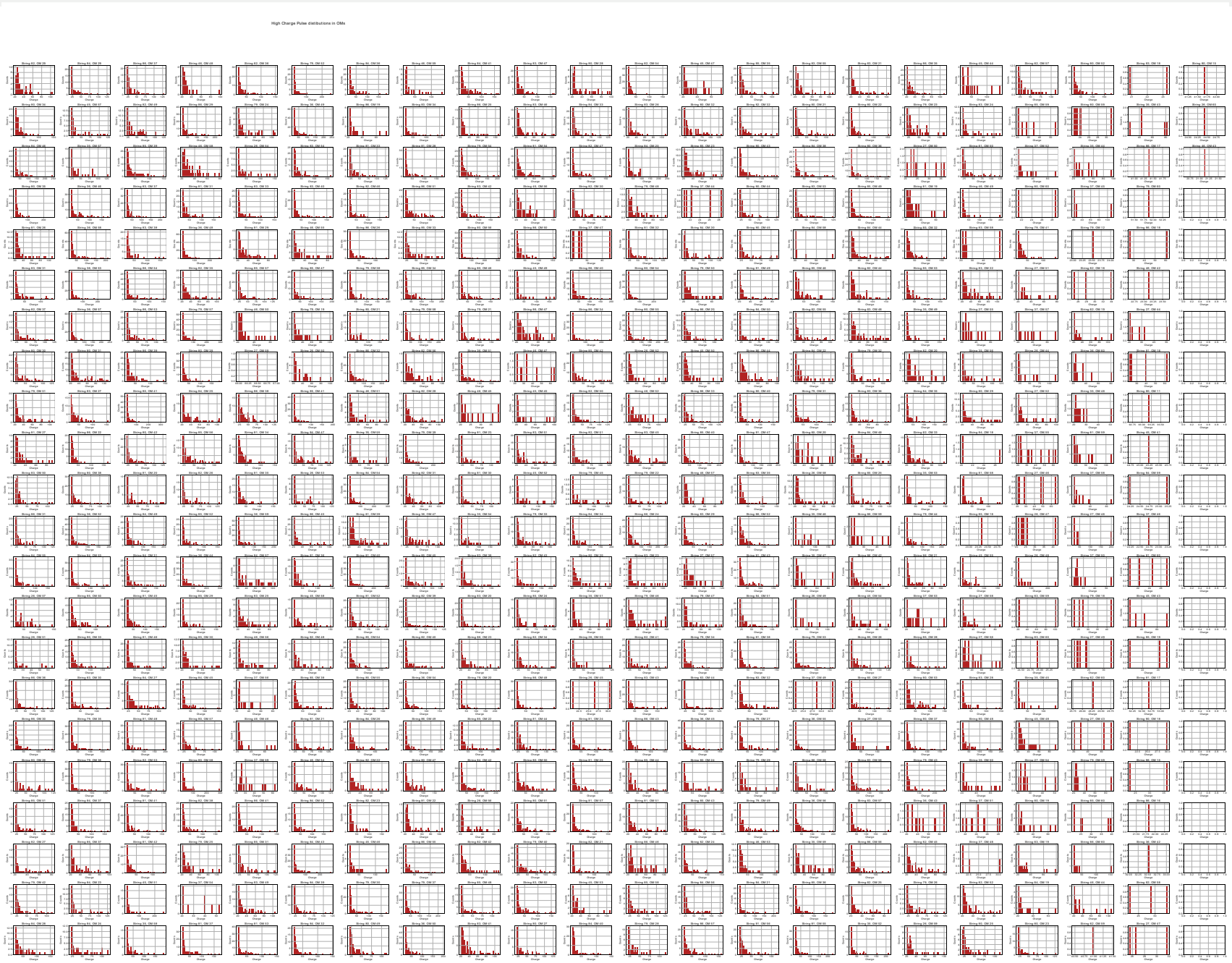
Time distribution of the photon weight plateau






Total charge in OM vs. OM Distance from vertex
35000
30000
25000
20000
15000
10000
5000
0
ΣQ
20
40
60
80
100
distance from vertex [m]

High charge pulses in oscnext data (ν_e)



Simulation Steps Extended

- Step 1: Create Particle

- Step 2: Simulate Photons
  - Inputs (in c++ script):
    - DomEfficiency = 1.0 * 1.2 for margin/systematic sets (?)
    - UnshadowedFraction = 1.2 same as DOMEfficiency (0.9)
    - IceModel = spice_3.2.1 (spice_mie)
    - HoleIce = angsens/as.flasher_p1_0.30_p2_-1 (h2-50cm)

- Step 3 : Add noise and convert to p.e
  - https://code.icecube.wisc.edu/projects/icecube/browser/IceCube/projects/clsim/trunk/python/traysegments/I3CLSimMakeHitsFromPhotons.py
  - Inputs (default)
    - DomEfficiency 1.0
    - UnshadowedFraction = 1.2 (1.0)
    - HoleIce = angsens/as.flasher_p1_0.30_p2_-1 (h2-50cm)
    - DOMOversizeFactor = 1.0 (5.0)
  - https://code.icecube.wisc.edu/projects/icecube/browser/IceCube/projects/clsim/trunk/private/clsim/dom/I3PhotonToMCPEConverter.cxx

- Level 1

- Level 2