

JMarkov – an update

ANTARES/KM3NeT group meeting
Martijn Jongen

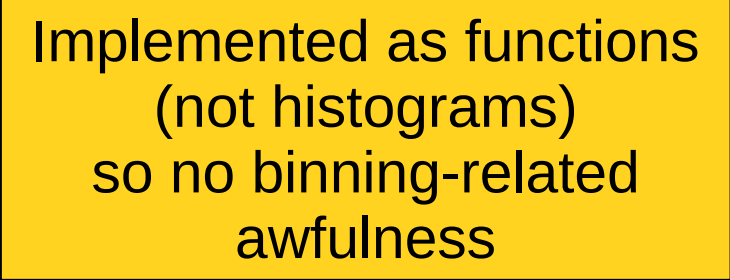
Refresher

- Photon propagation
- Markov Chain Monte Carlo (MCMC)
 - generate representative sample of photon paths
 - all paths hit the target by construction
 - easy to get high statistics
 - number of scatterings is fixed
 - **problem:** hard to get relative normalization
- **new:** photon tracking
 - create random photons, follow them until they hit the target or are absorbed
 - often-used approach
 - relatively simple
 - many photons simulated, only a few hits for far-away targets

Hybrid approach

- Do both **MCMC** and **tracking**
- Use same input code
 - source distribution
 - scattering model
 - target efficiency
- Results can be compared easily
- Utilize strengths of both methods

Implemented as functions
(not histograms)
so no binning-related
awfulness



The diagram consists of a yellow rectangular box with a black border. Inside the box, the text reads: 'Implemented as functions (not histograms) so no binning-related awfulness'. Three black arrows originate from the left side of the box and point to the three sub-items of the 'Use same input code' bullet point: 'source distribution', 'scattering model', and 'target efficiency'.

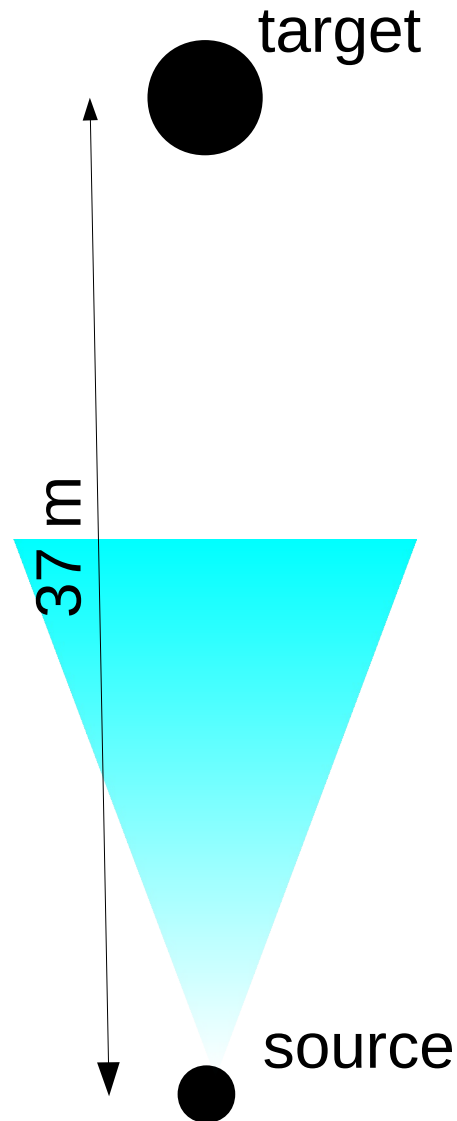
A sanity check

MCMC

- infinitesimally small target (by definition)

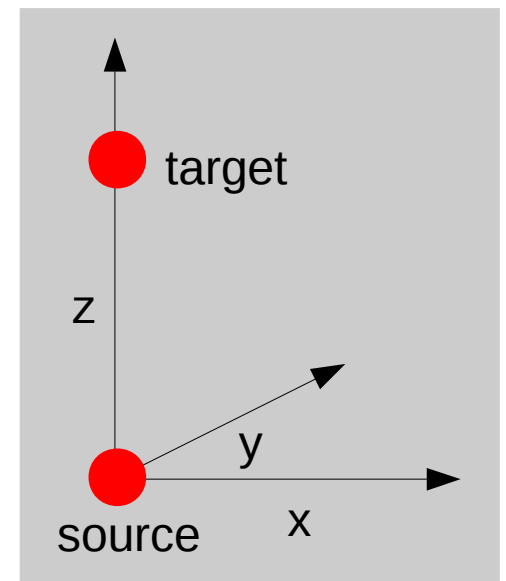
Tracking

- target $r=0.216$ m



A sanity check

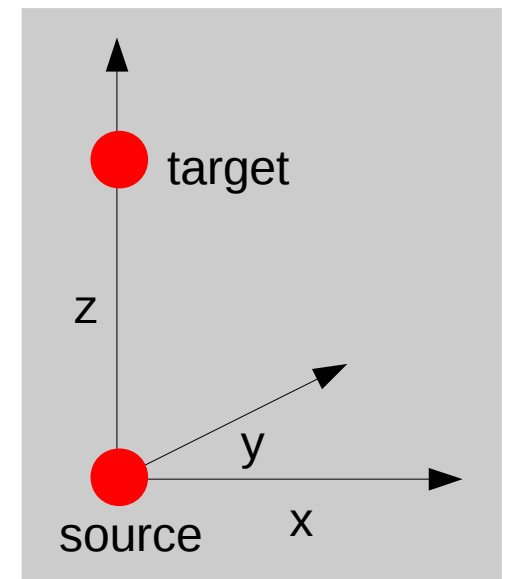
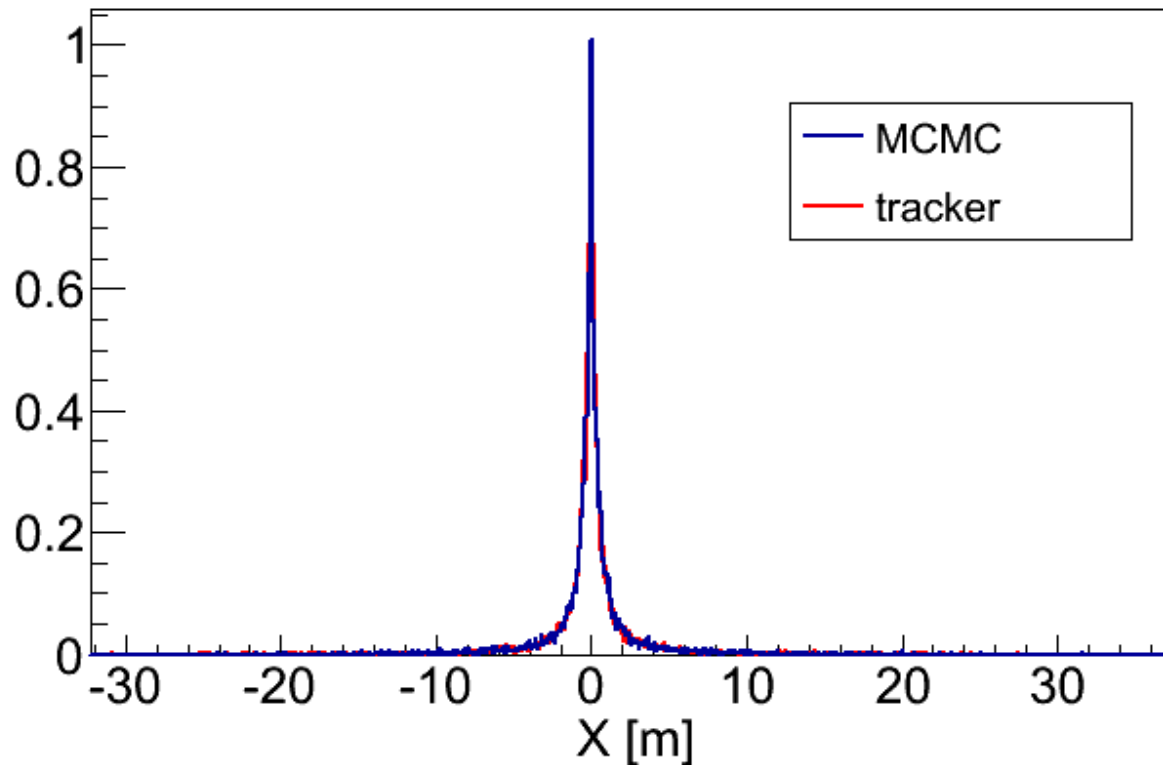
- Source profile
 - pointed upwards
 - isotropic with 90 degree opening angle
- Scattering model
 - details in backup slide)
 - effective scattering length = 50 m
 - absorption length = 50 m
- Target efficiency
 - isotropic target



A sanity check

x10
tracker
statistics

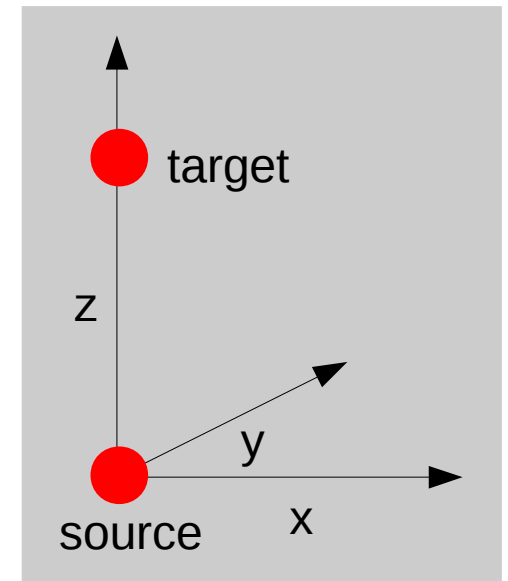
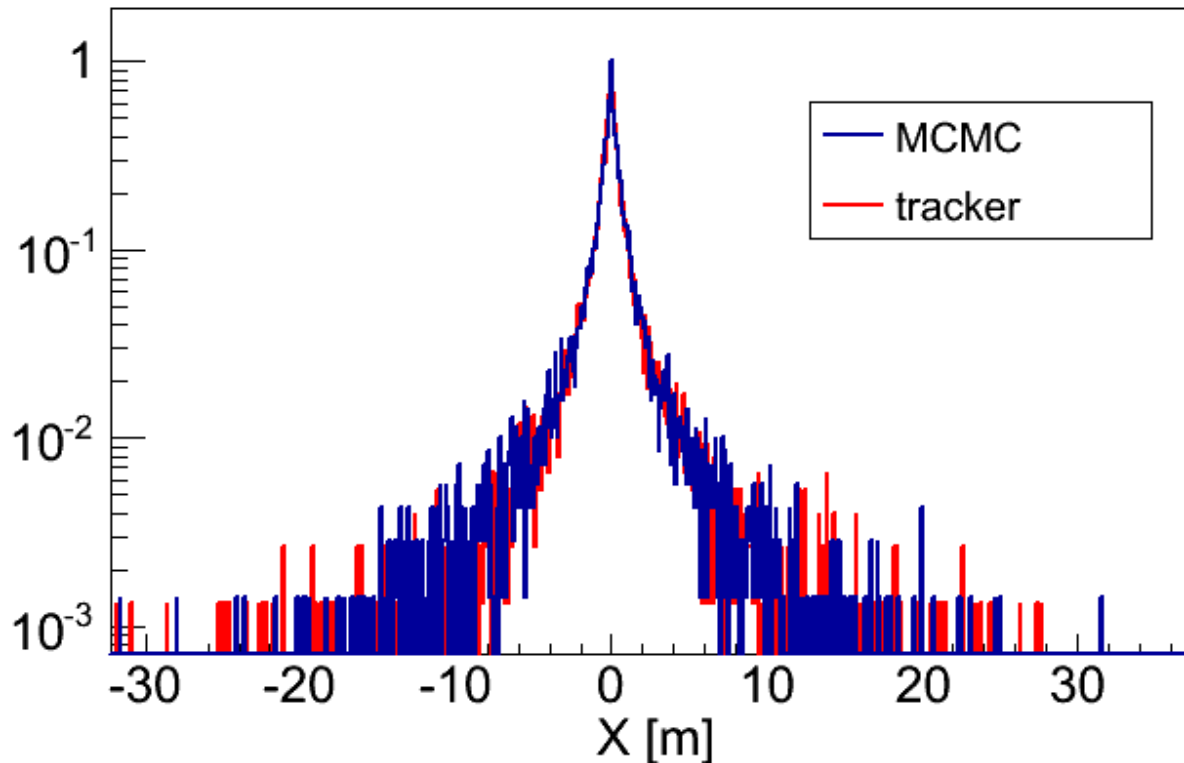
Vertex position (nscat = 1, vertex = 1)



A sanity check

x10
tracker
statistics

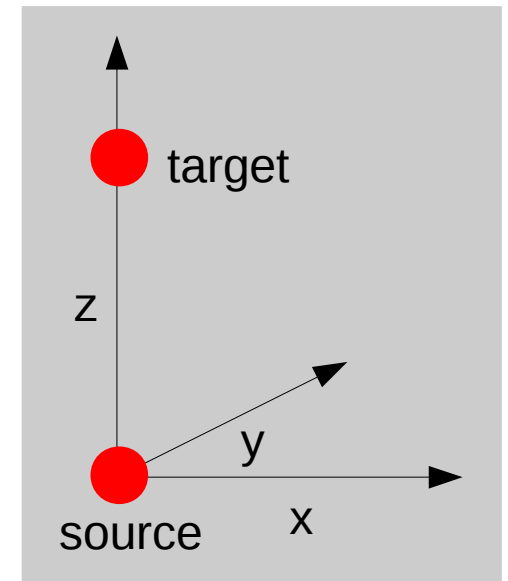
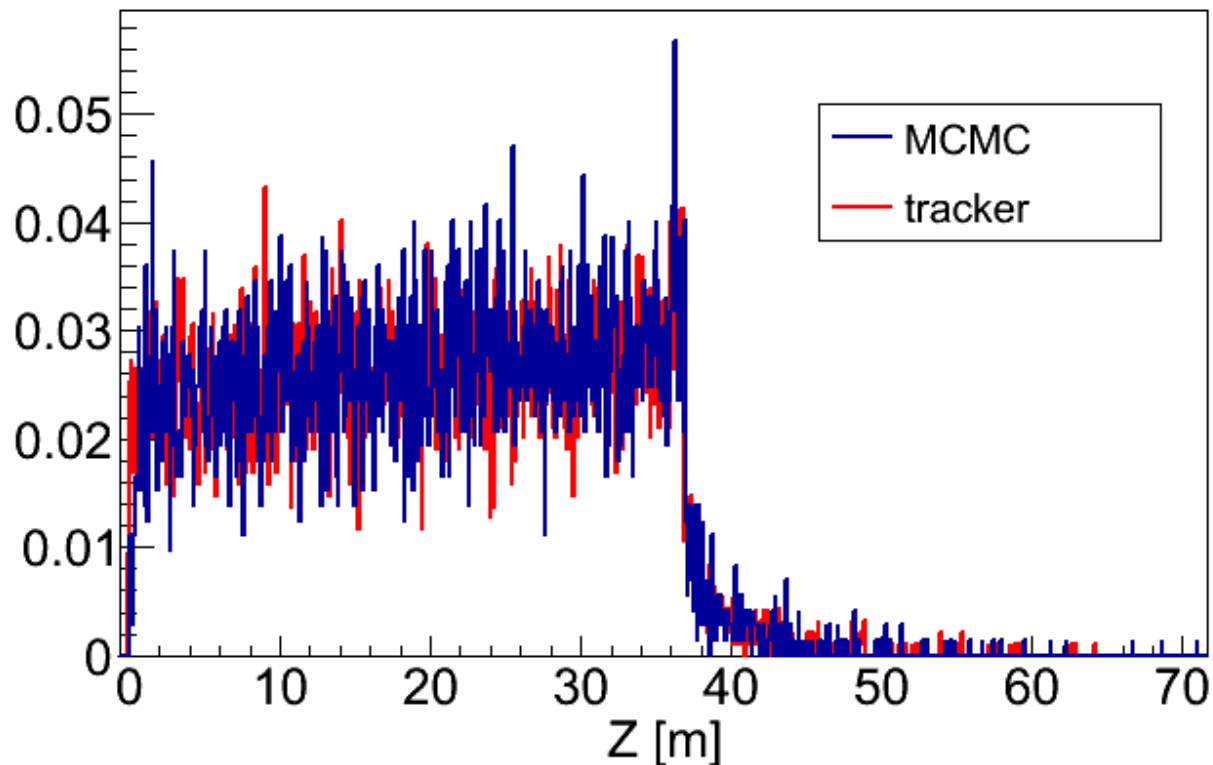
Vertex position (nscat = 1, vertex = 1)



A sanity check

x10
tracker
statistics

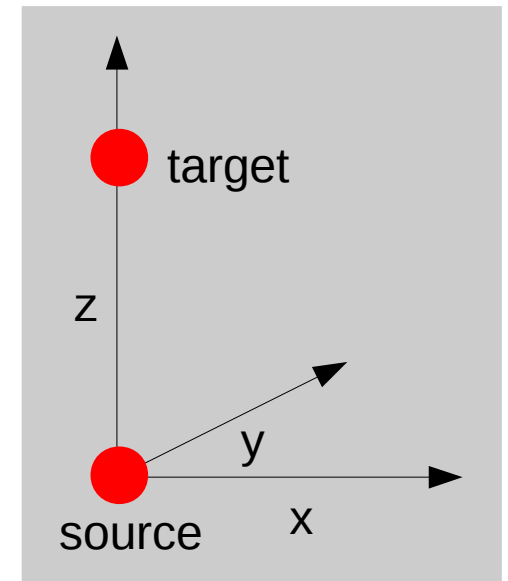
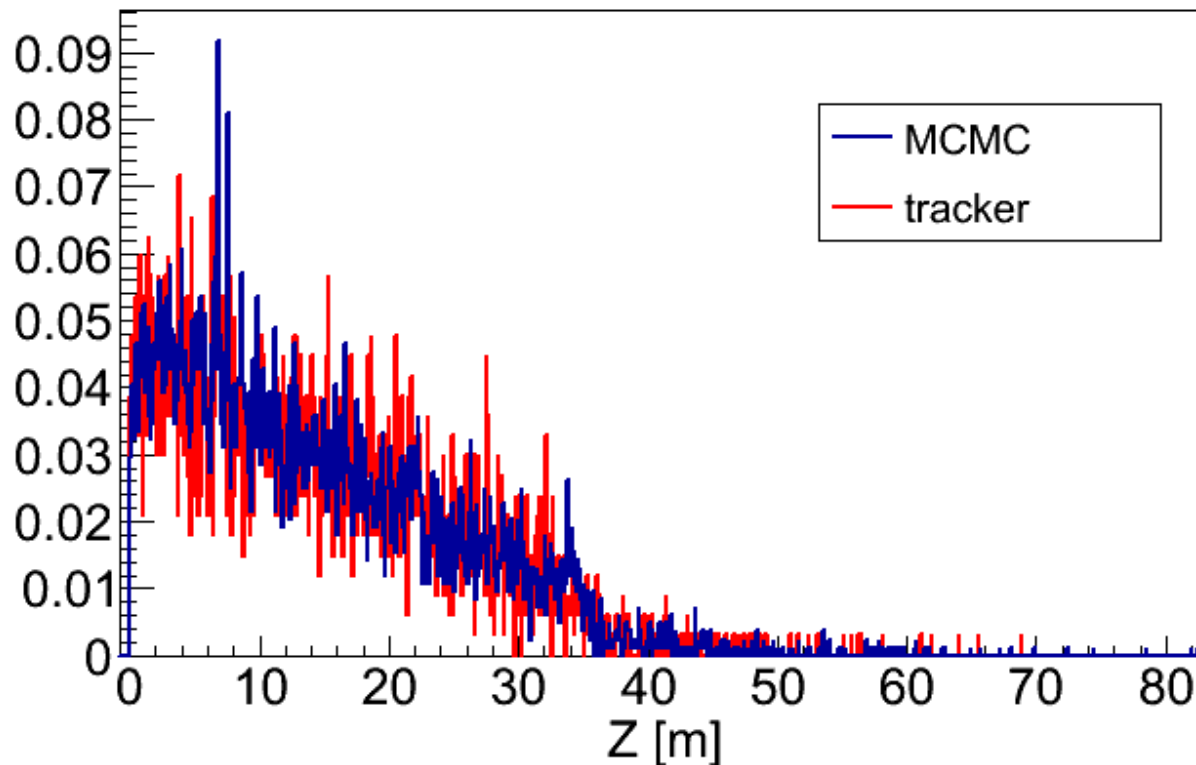
Vertex position (nscat = 1, vertex = 1)



A sanity check

x10
tracker
statistics

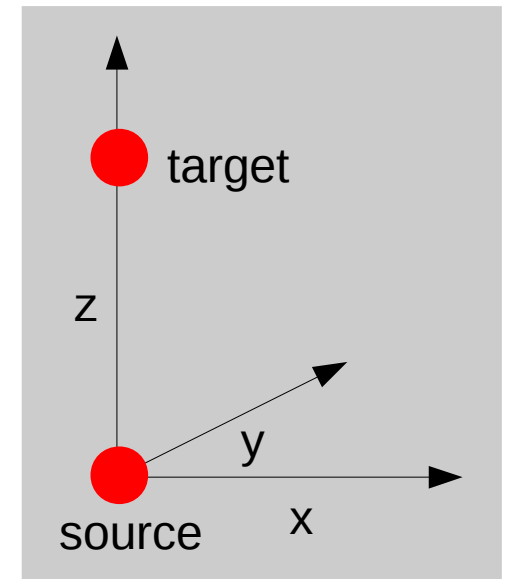
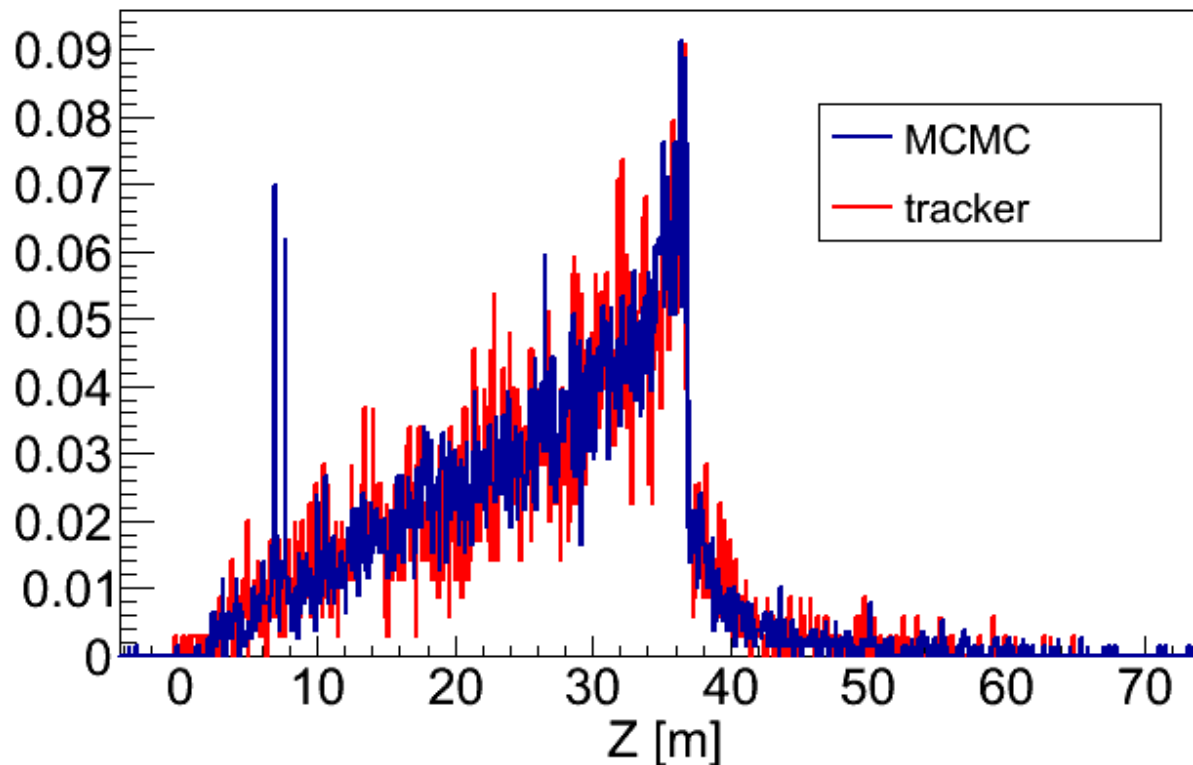
Vertex position (nscat = 2, vertex = 1)



A sanity check

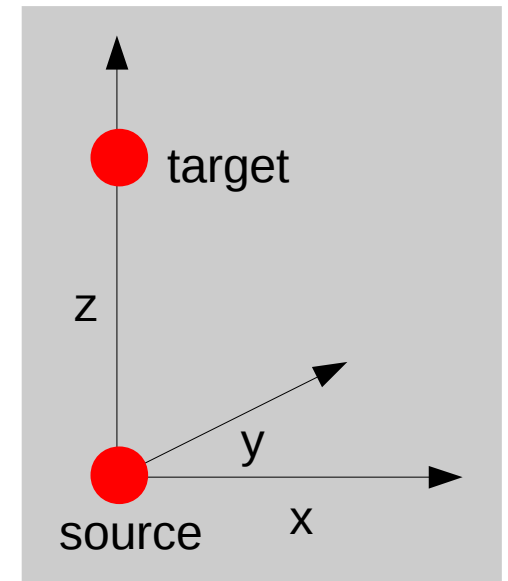
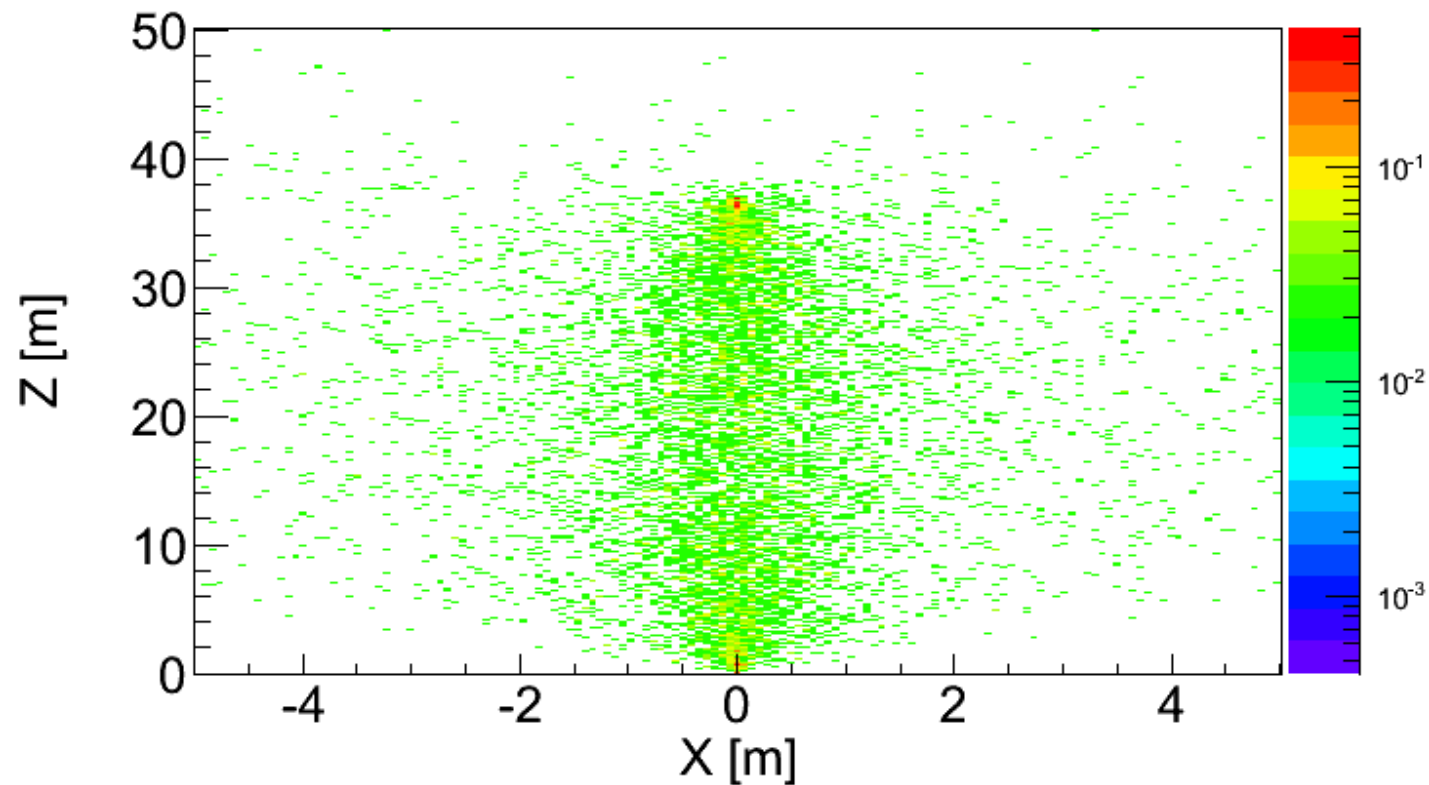
x10
tracker
statistics

Vertex position (nscat = 2, vertex = 2)



A sanity check

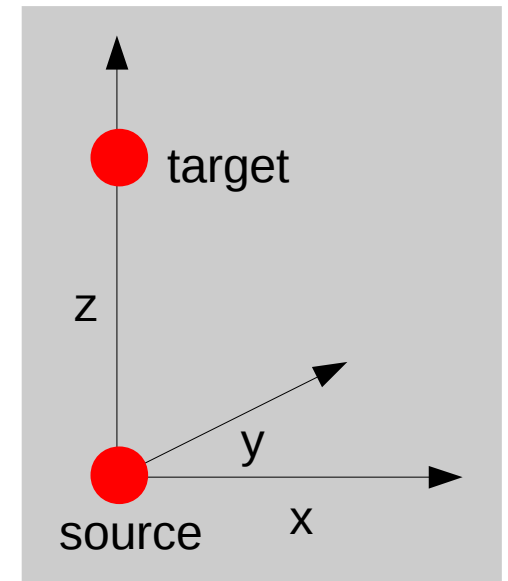
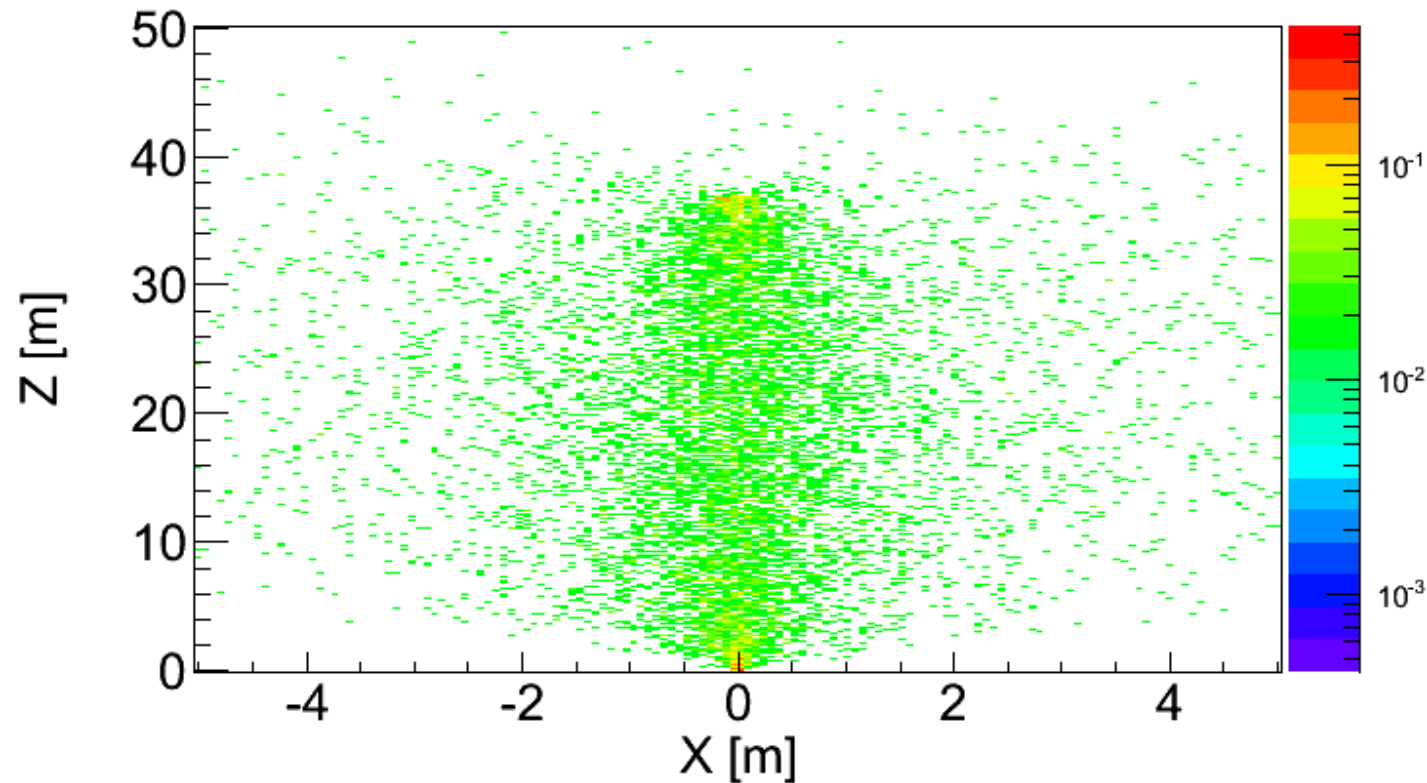
MCMC vertex position (nscat = 1, vertex = 1)



A sanity check

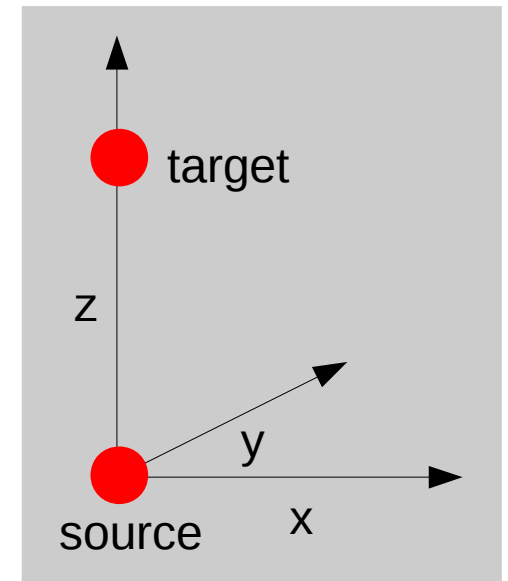
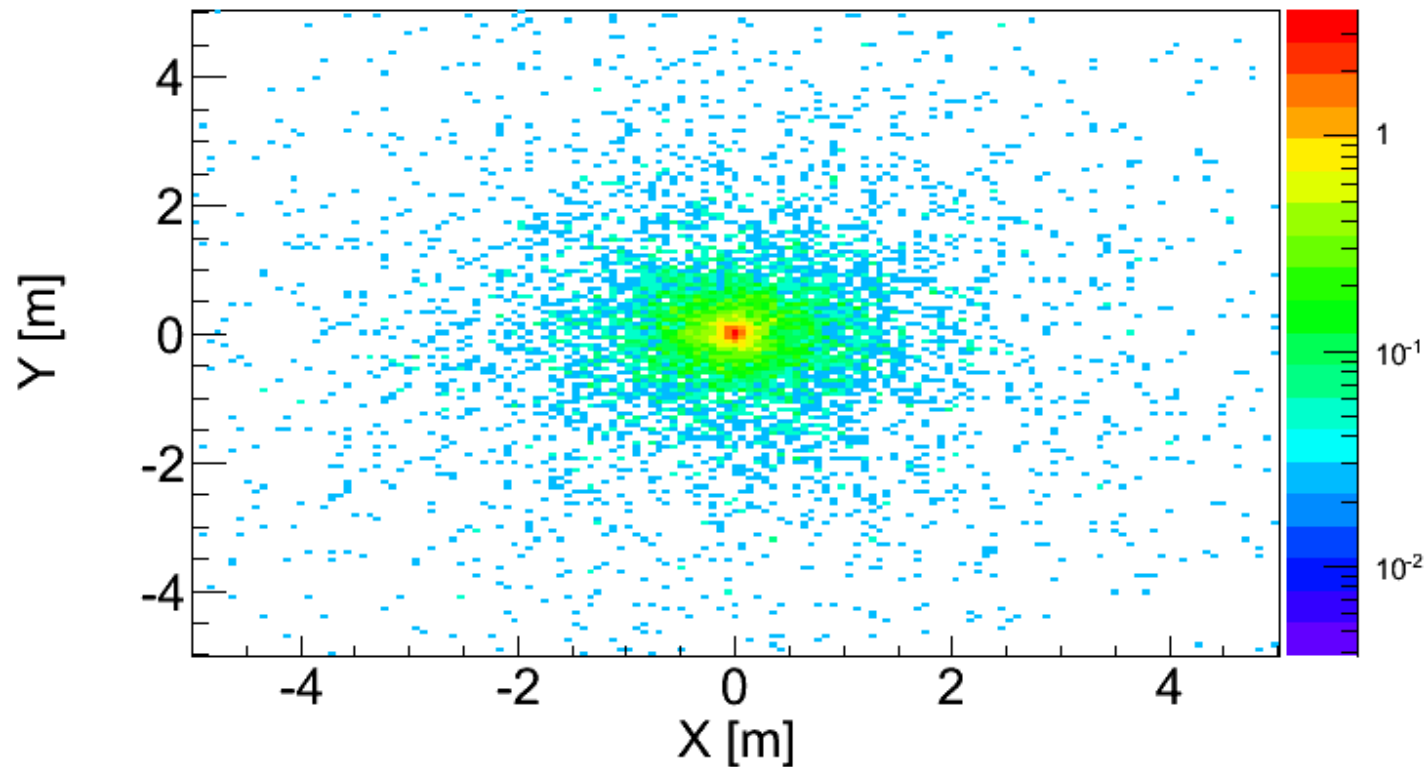
x10
tracker
statistics

Tracker vertex position (nscat = 1, vertex = 1)



A sanity check

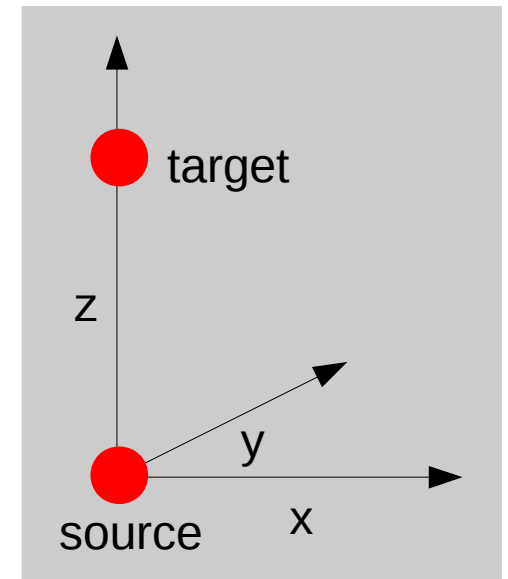
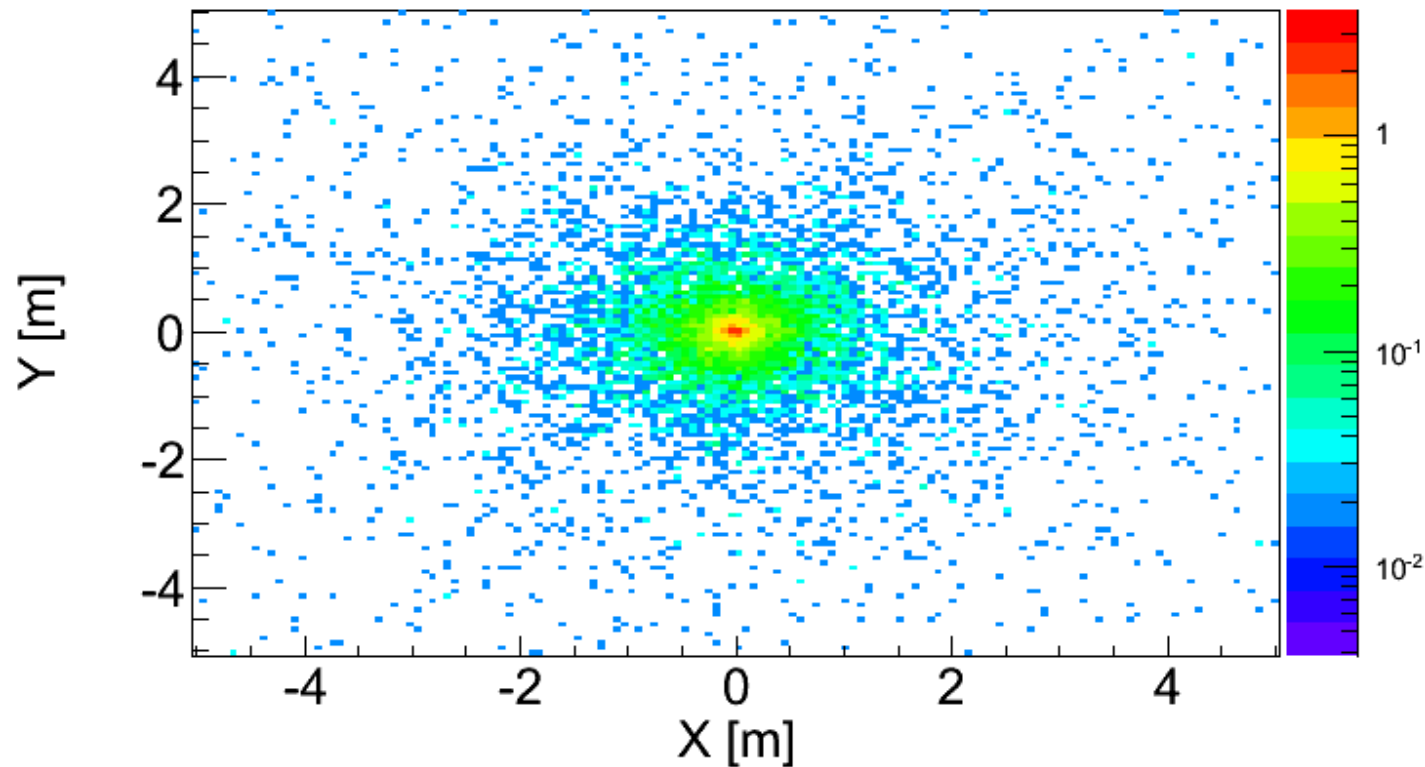
MCMC vertex position (nscat = 1, vertex = 1)



A sanity check

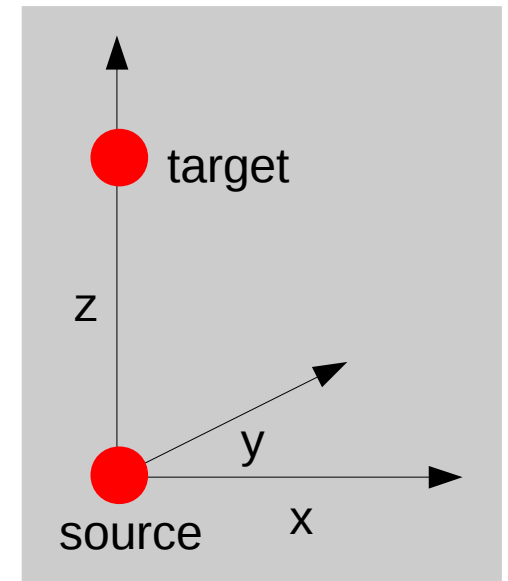
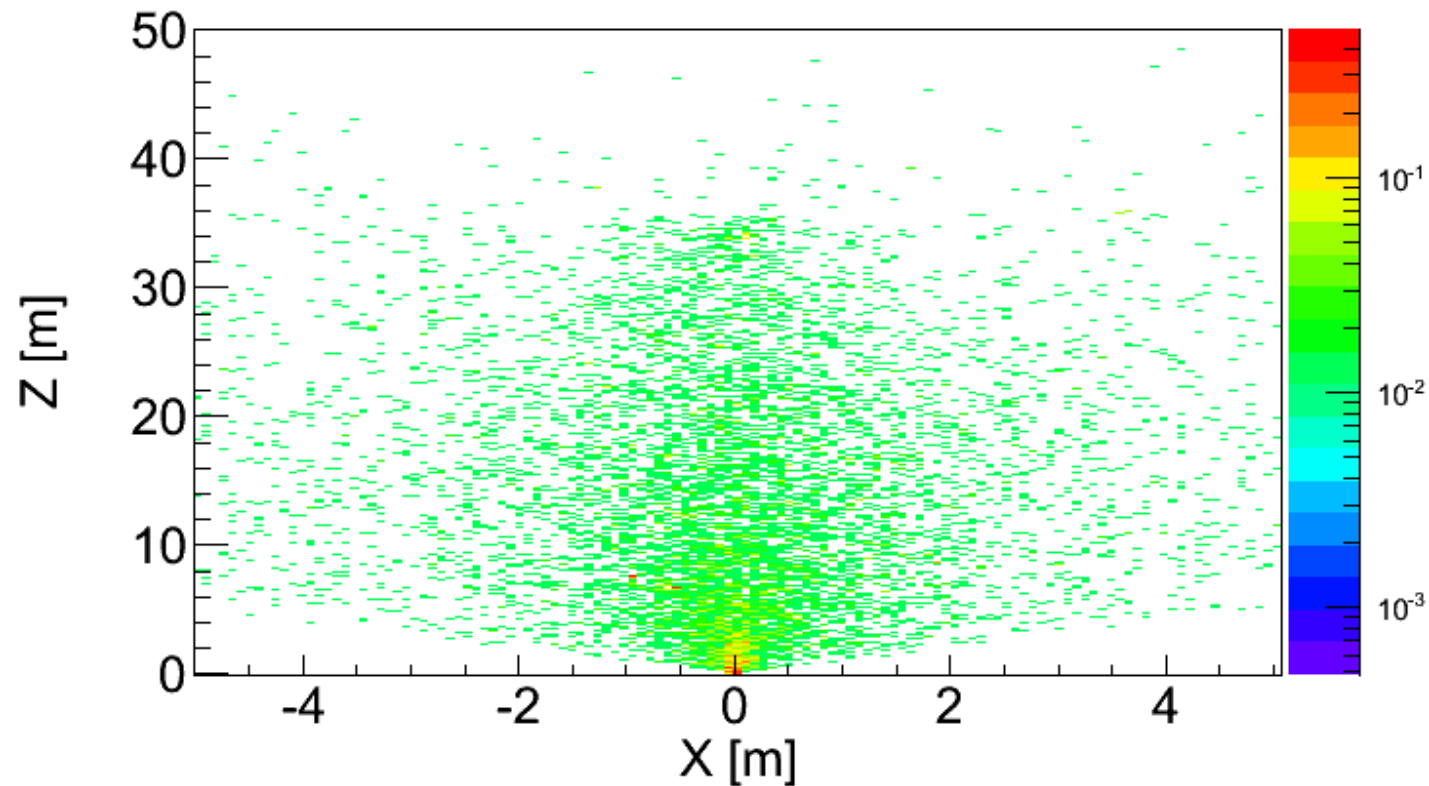
x10
tracker
statistics

Tracker vertex position (nscat = 1, vertex = 1)



A sanity check

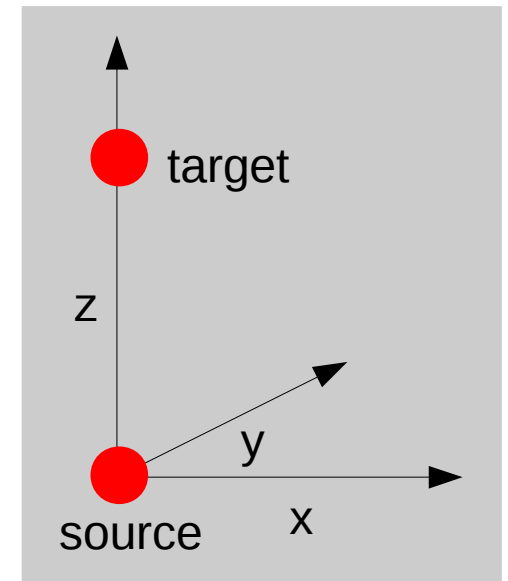
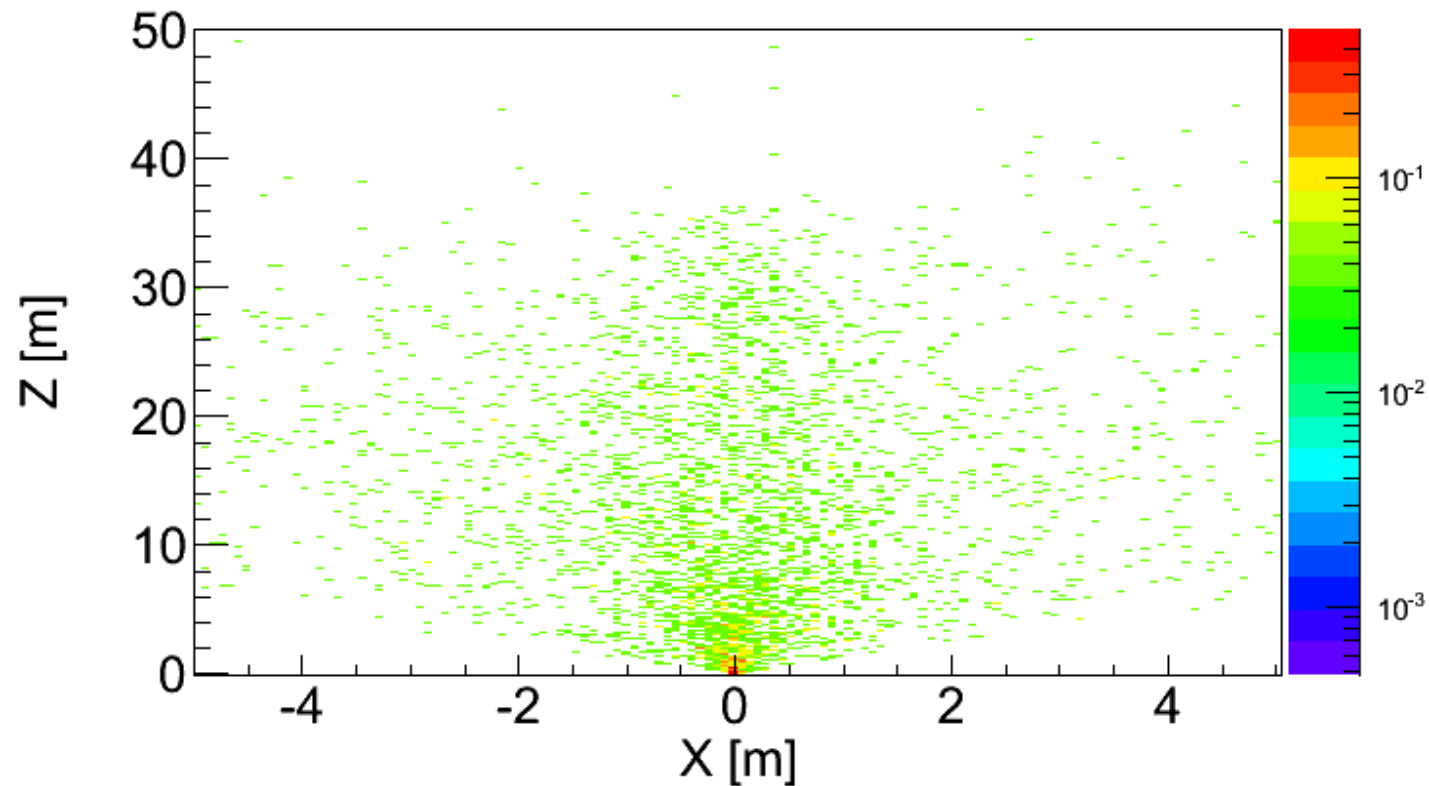
MCMC vertex position (nscat = 2, vertex = 1)



A sanity check

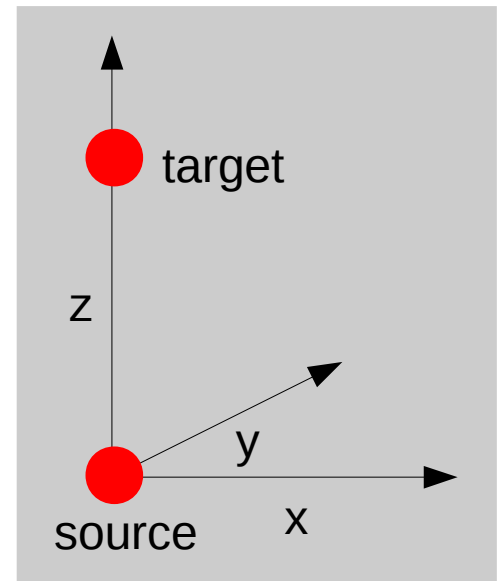
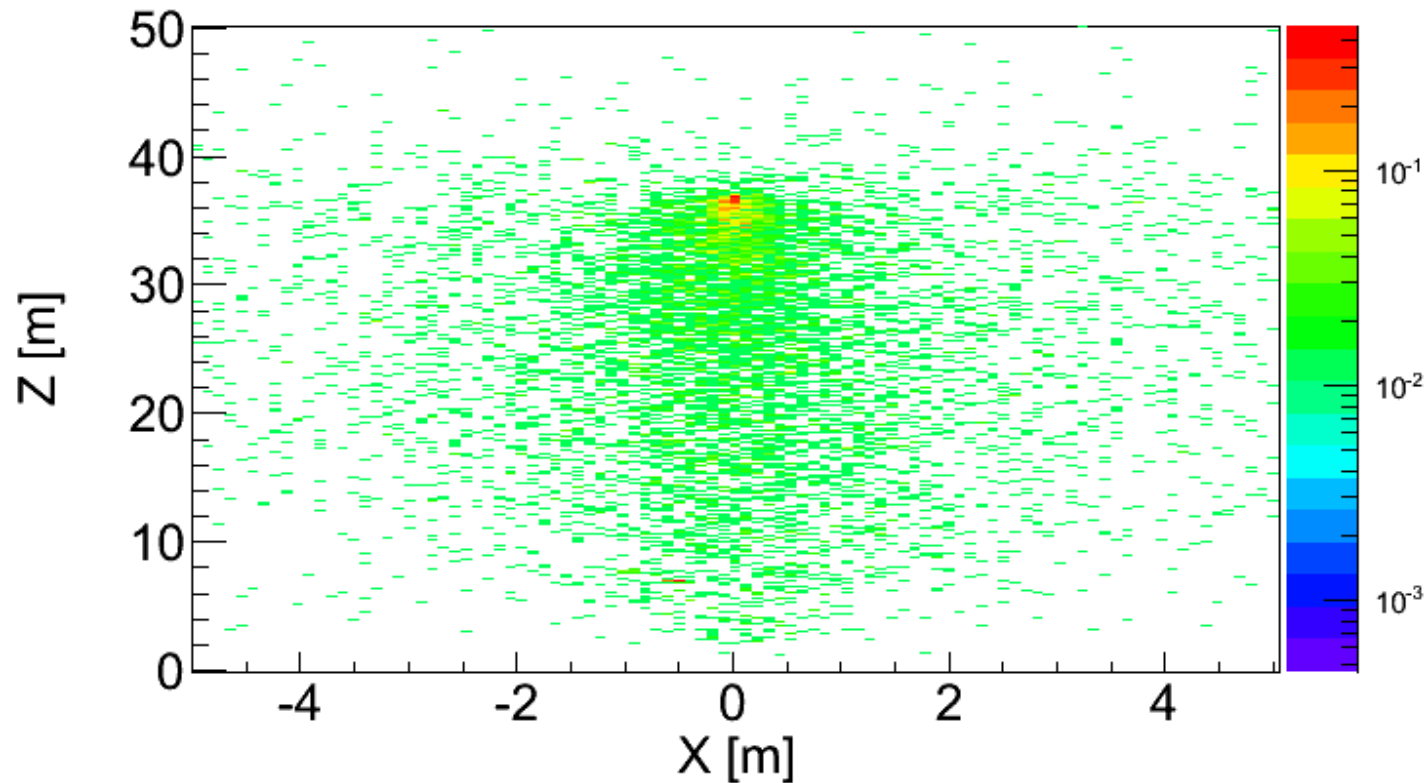
x10
tracker
statistics

Tracker vertex position (nscat = 2, vertex = 1)



A sanity check

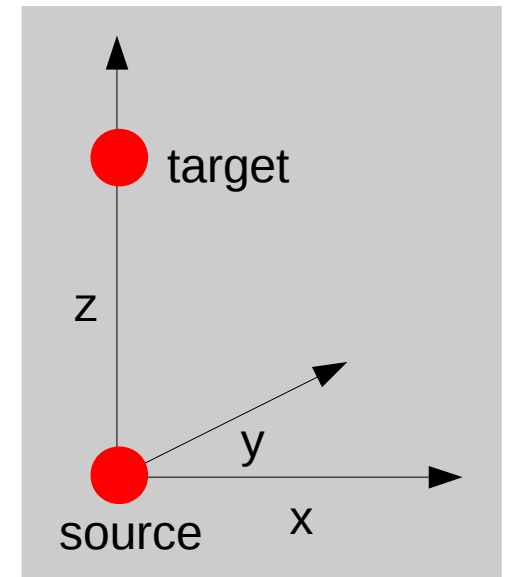
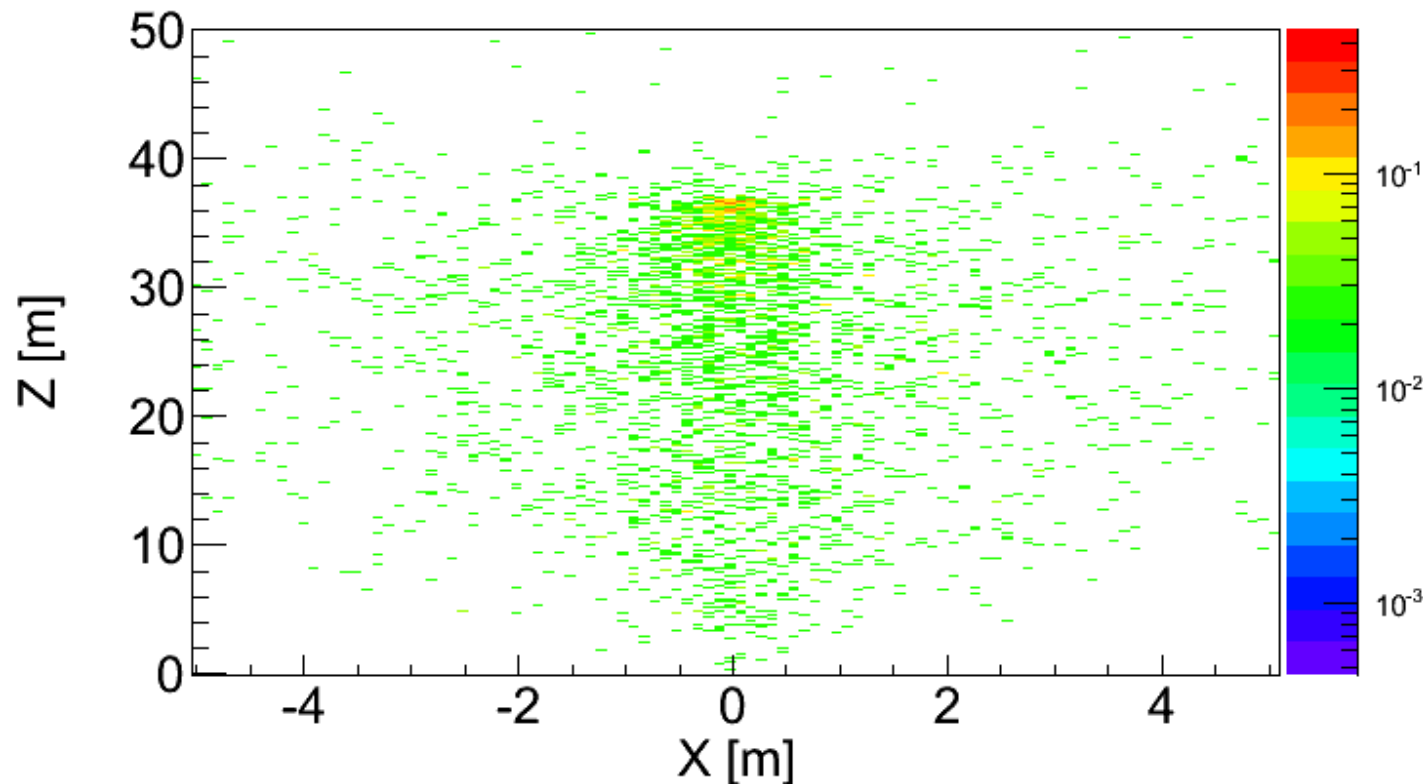
MCMC vertex position (nscat = 2, vertex = 2)



A sanity check

x10
tracker
statistics

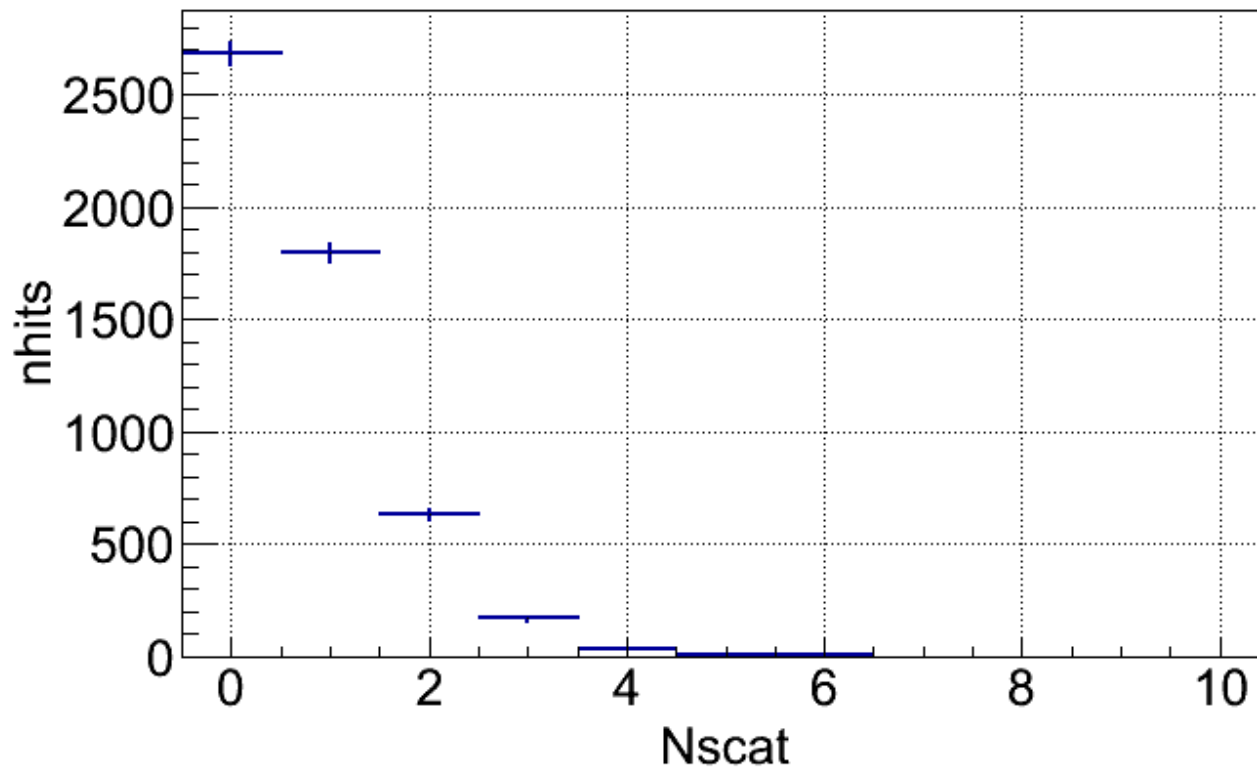
Tracker vertex position (nscat = 2, vertex = 2)



Note that this is a bit more spread out due to the finite target size.

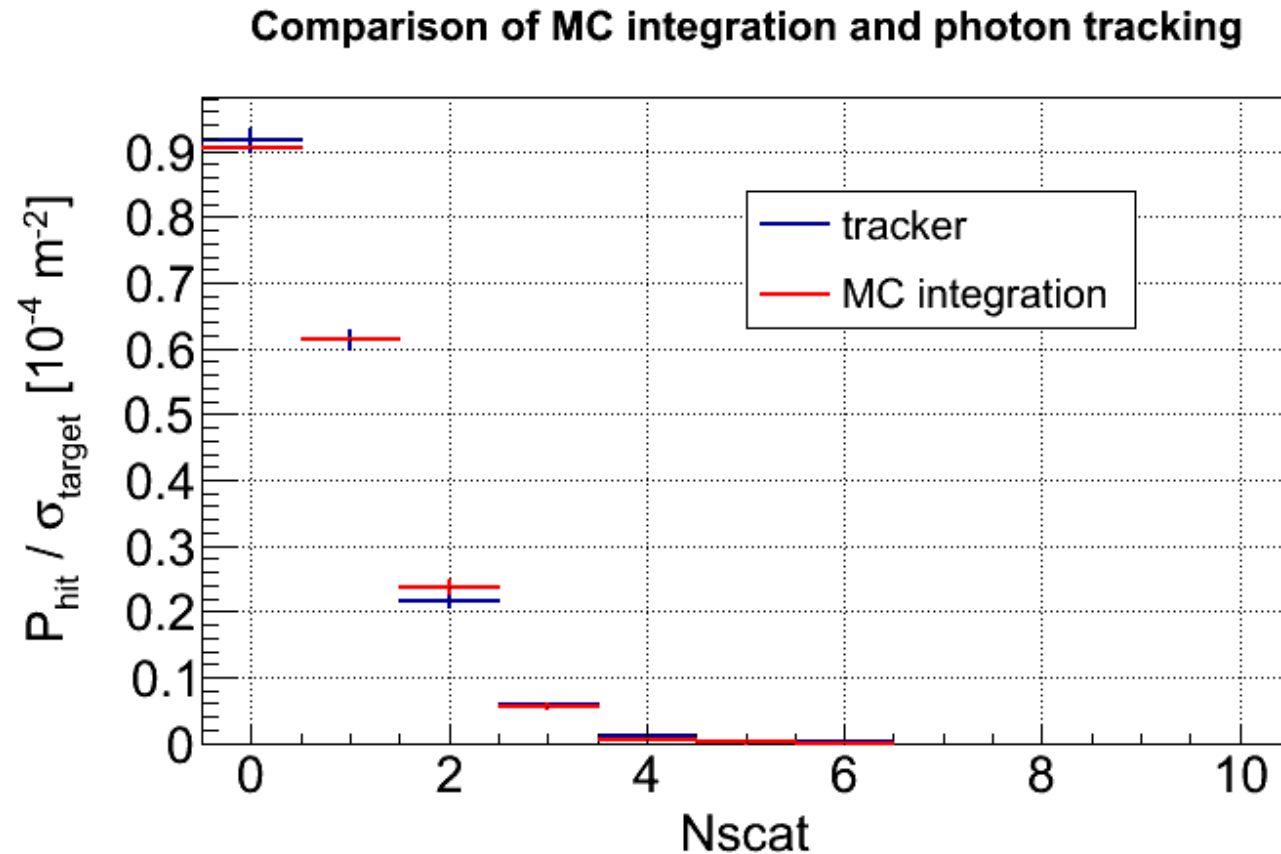
A sanity check

Tracking (200M generated photons)



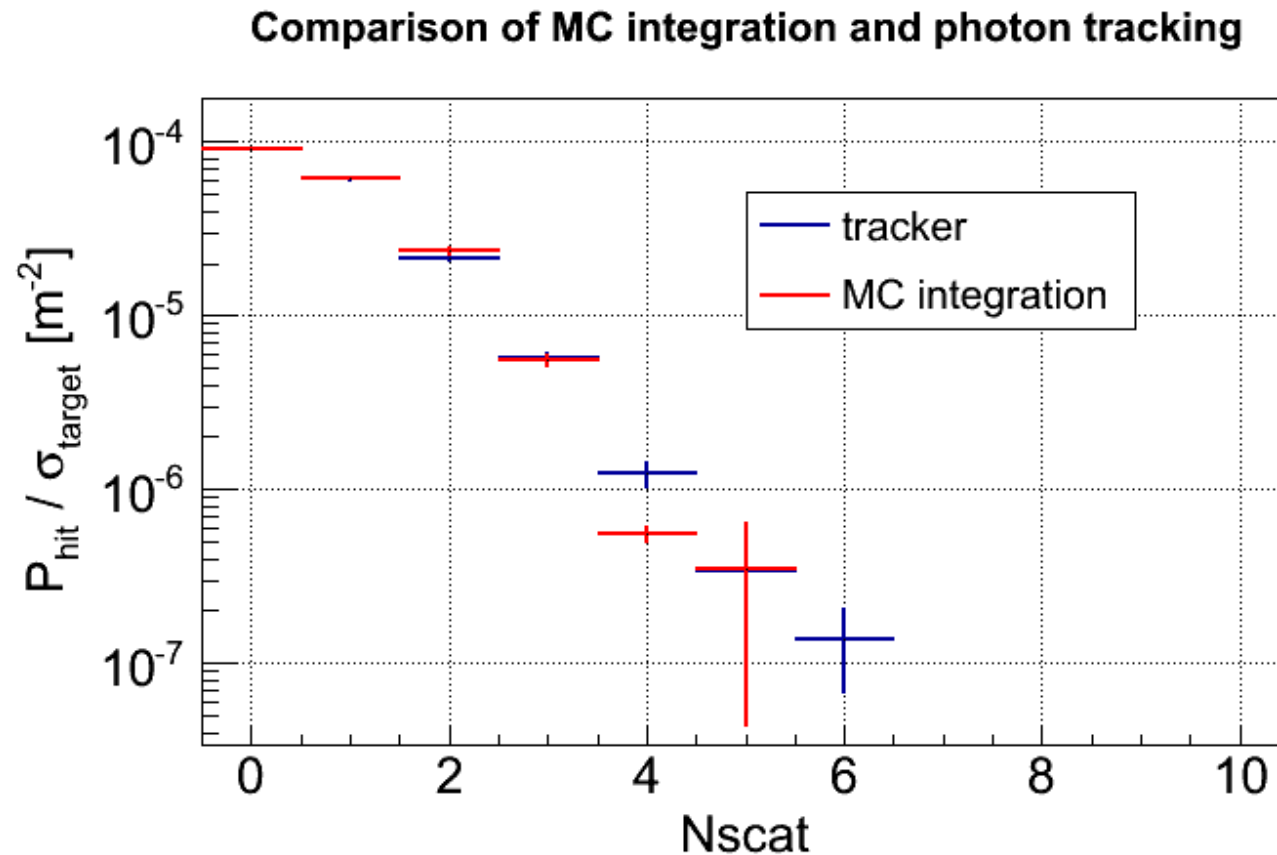
Total Phit is $\sim 2.7 \times 10^{-5}$

A sanity check



Note: using ensemble-based sampling, 10M samples

A sanity check



Note fluctuations and poor error estimate of MC integration

Performance

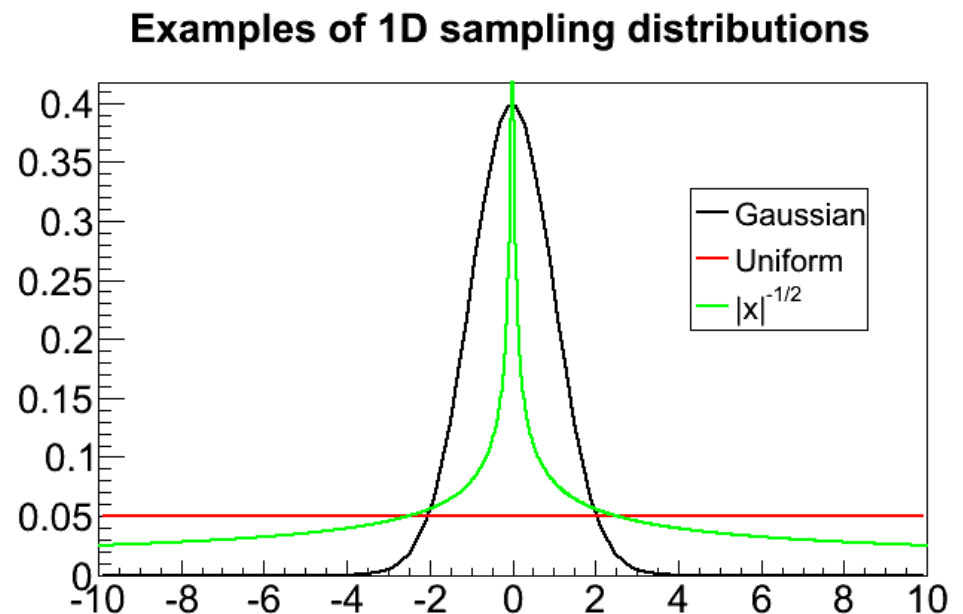
- This whole example: <8 mins on a single core
 - except where it says
- Photon tracking
 - 200M photons in approx. 5 mins
 - 5.3k hits
- MCMC path generation
 - very fast
 - about half a minute for 70k paths (nscat = 1 to 6)
- Integration by importance sampling
 - about 2 minutes for 10M samples and nscat = 1 to 6



x10
tracker
statistics

MC integration: 1D

- Integrate some function $f(x)$
- Importance sampling
 - generate points x_i from a **sample distribution $g(x)$**
 - $g(x)$ is **normalized** to 1
 - evaluate $f(x_i)$ for each point
 - each contributes $f(x_i)/g(x_i)$
 - integral is average of contributions
- **Efficient when $g(x)$ resembles $f(x)$**



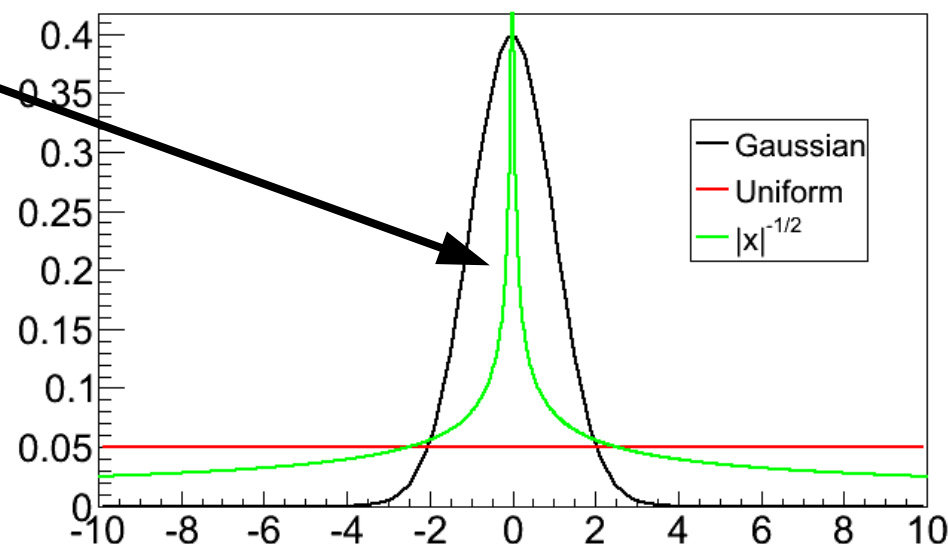
MC integration: paths

- We integrate the **path probability density** ρ
 - $\rho = d\text{Phit} / (dV_1 dV_2 \dots dV_n d\sigma_{\text{target}})$
 - integrate over the full spatial volume for each scattering vertex to get $d\text{Phit} / d\sigma_{\text{target}}$
 - dimension: **$D = 3 \times n_{\text{scat}}$**
- Importance sampling
 - generate paths from some sampling distribution **g**
 - g is **normalized** (i.e. D -dimensional integral over g is 1)
 - each path p contributes $\rho(p)/g(p)$
 - integration is more **efficient when g resembles ρ**

MC integration: singularities

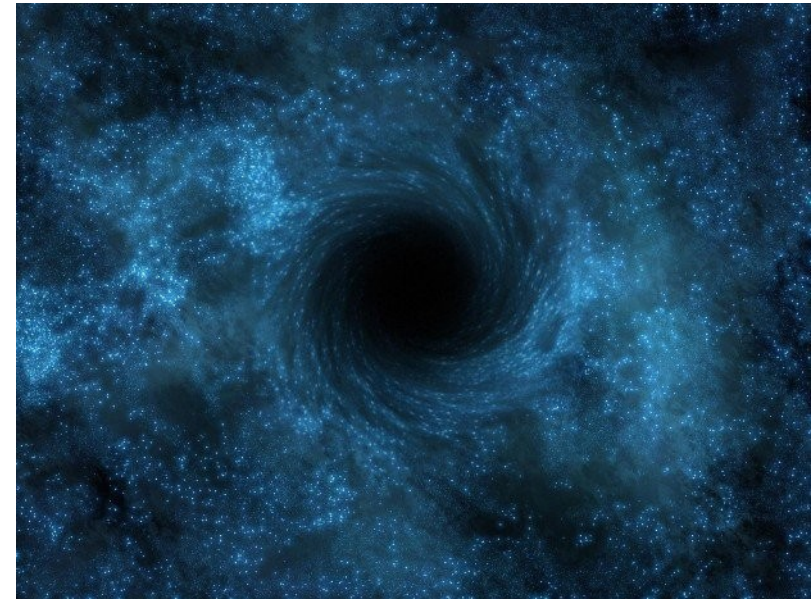
- Sometimes $f(x)$ goes to infinity at some point, but the integral over the region is still finite
- Computers dislike infinity
- E.g. $1/\sqrt{|x|}$ on the right
- Sampling is troublesome
 - integral contributions will have a large tail
 - huge statistical fluctuations
 - average **unstable**
 - **poor error estimation**
 - unless... the singularity is matched in the sampling distribution!

Examples of 1D sampling distributions



MC integration: singularities

- occur in path probability density
- $1/r^2$ terms
- when vertices are very close to each other
- e.g. in single scattering
 - probability density blows up when the scattering vertex is near to the source or the target

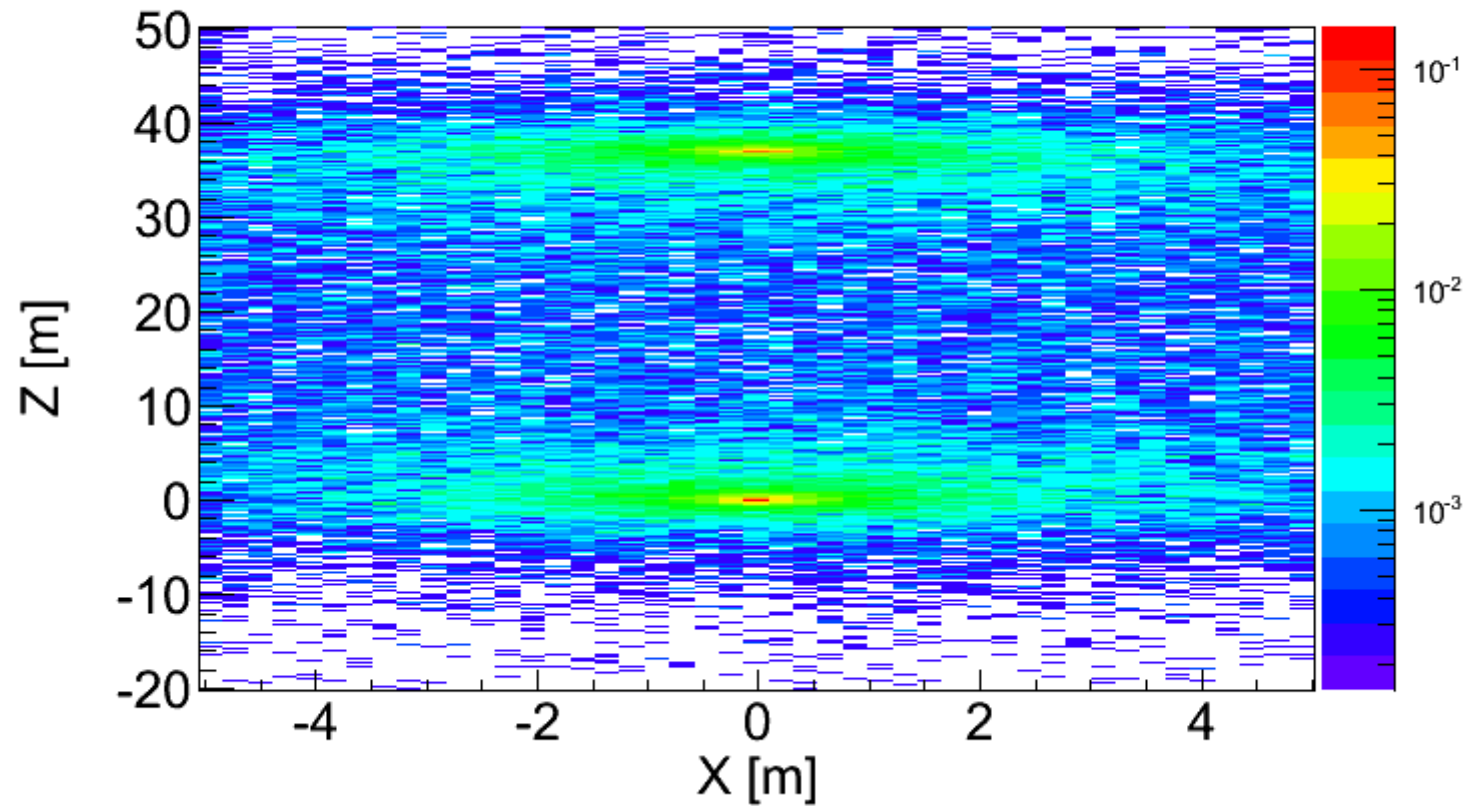


Example scenario 2

- Same as before but now
 - isotropic source
 - isotropic scattering
 - considering only single scattering
- Singularities become more prominent
 - but they were always there!
 - Will test MC integration methods in this scenario

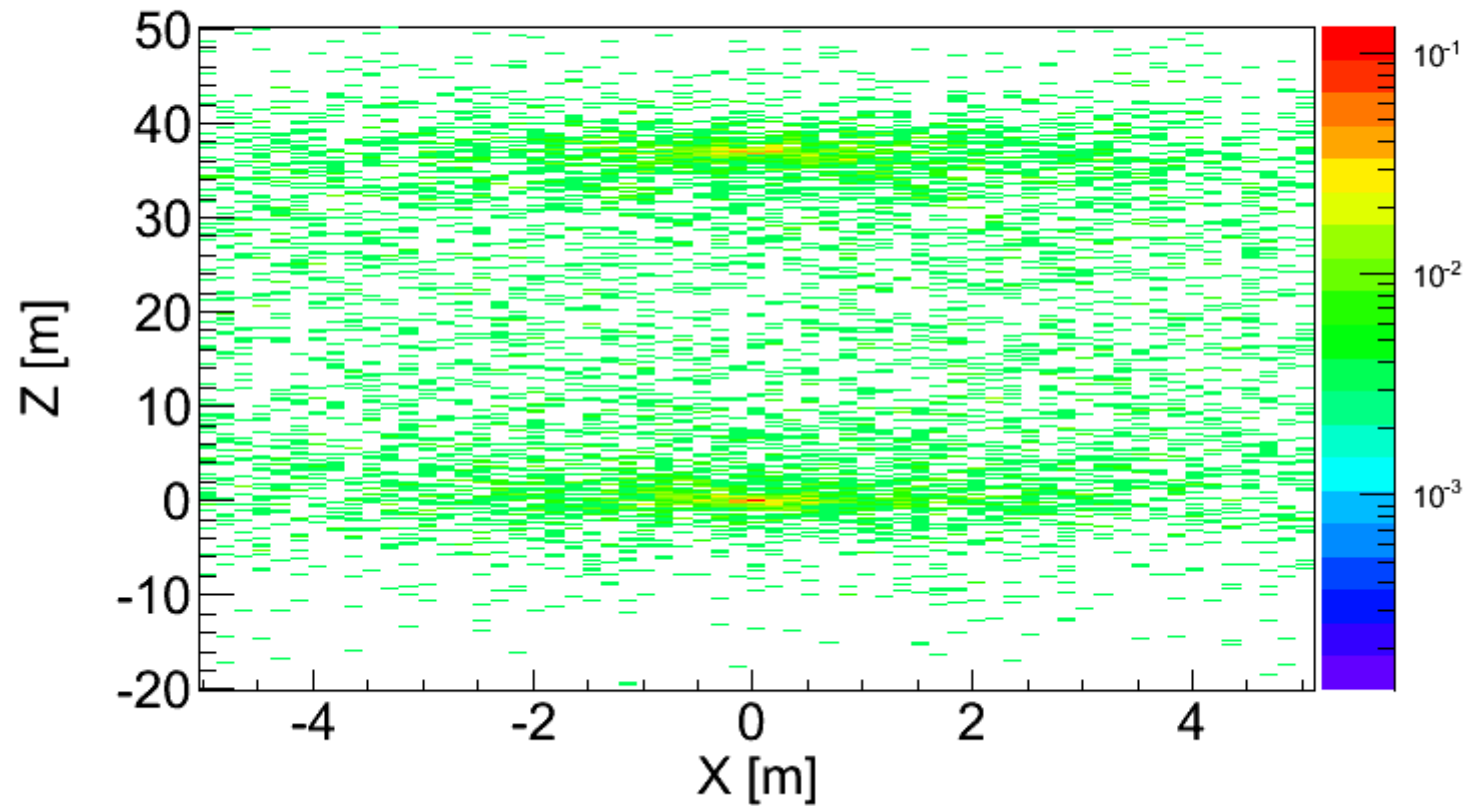
Example scenario 2

MCMC vertex position (nscat = 1, vertex = 1)

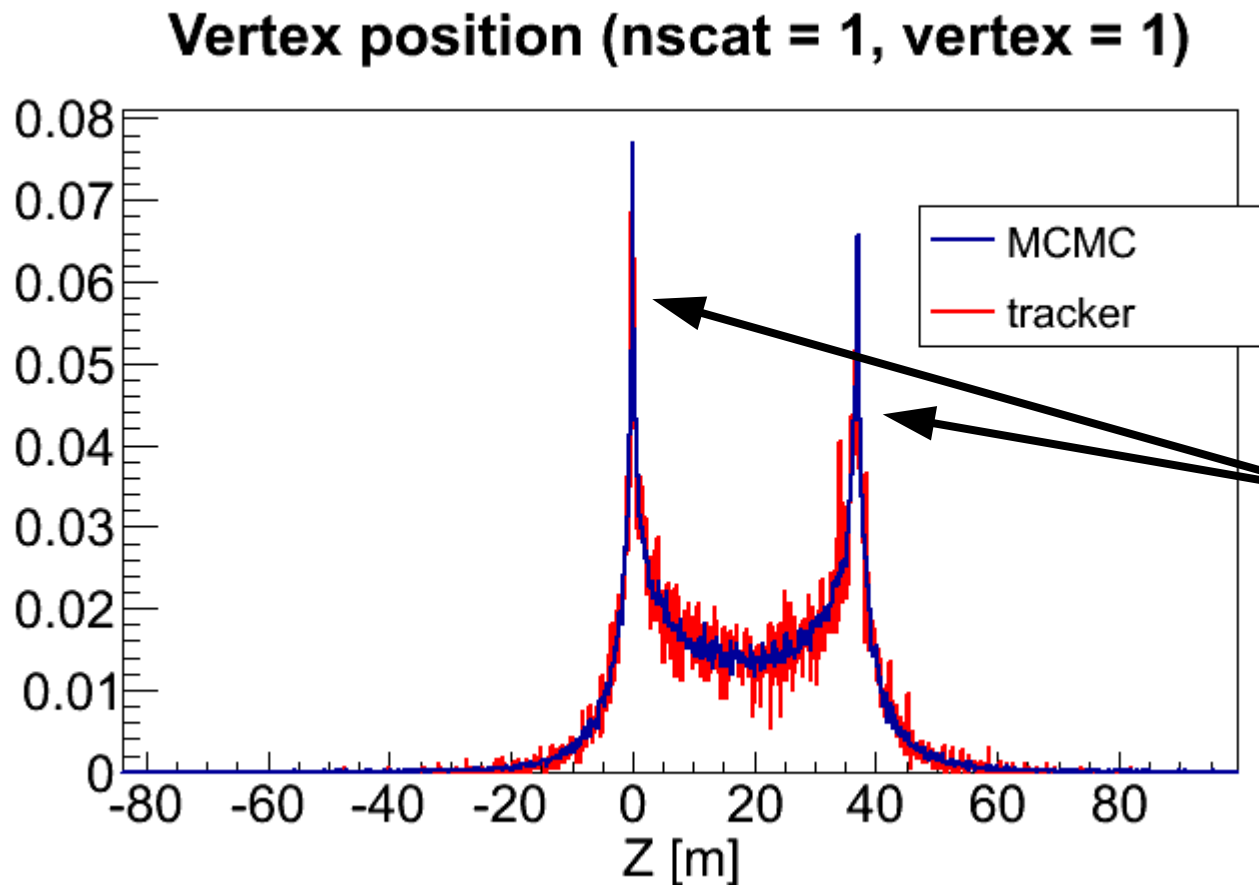


Example scenario 2

Tracker vertex position (nscat = 1, vertex = 1)



Example scenario 2



Singularities show up in both MCMC and tracker. They are physical, **not an artificial feature** of the probability density!

Sample distributions

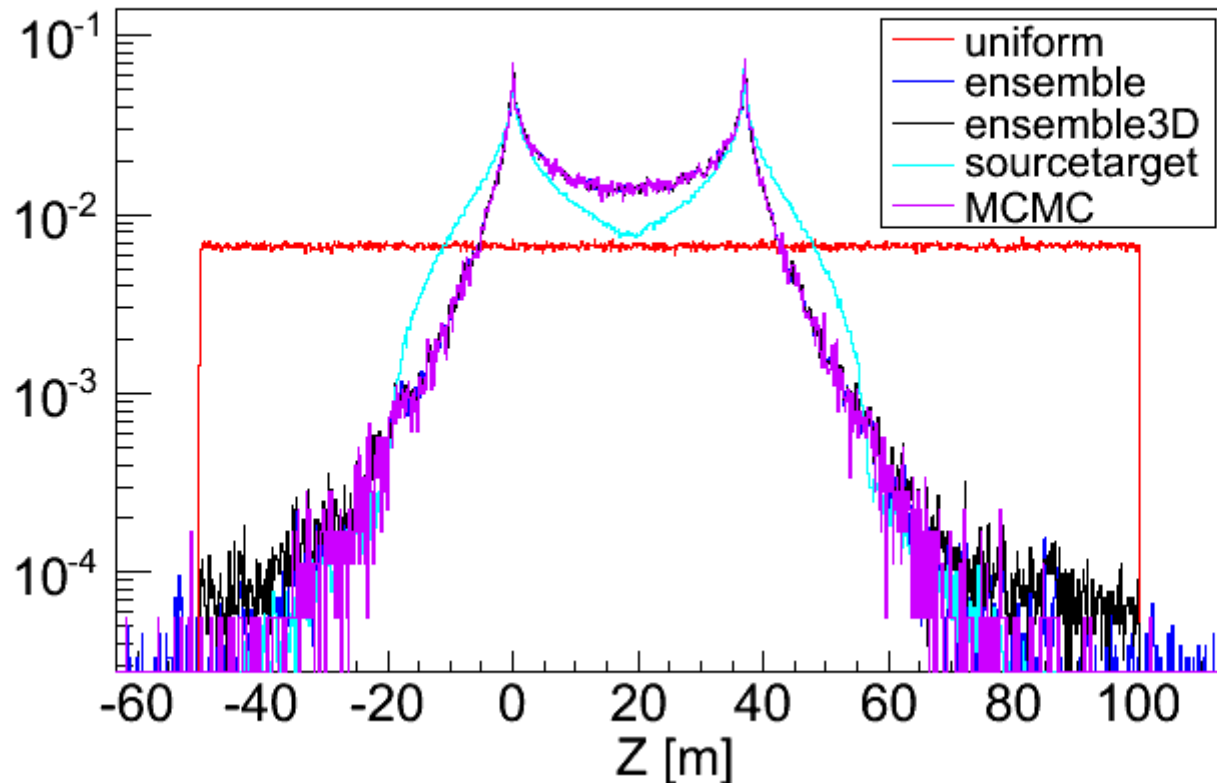
- **uniform**
- using MCMC ensemble as input
 - **ensemble** x, y and z from histograms
 - **ensemble3D** x, y and z from a single histogram
- **“sourcetarget”** (educated guess)
 - sum of three distributions
 - $1/r^2$ around source
 - $1/r^2$ around target
 - $\exp(-r/L)$ around center
 - manually tuned

Details

- 100k paths in MCMC ensemble (takes about 5s)
- 1M samples per integration method
- sample distributions tested on a dummy path probability density

Sample distributions

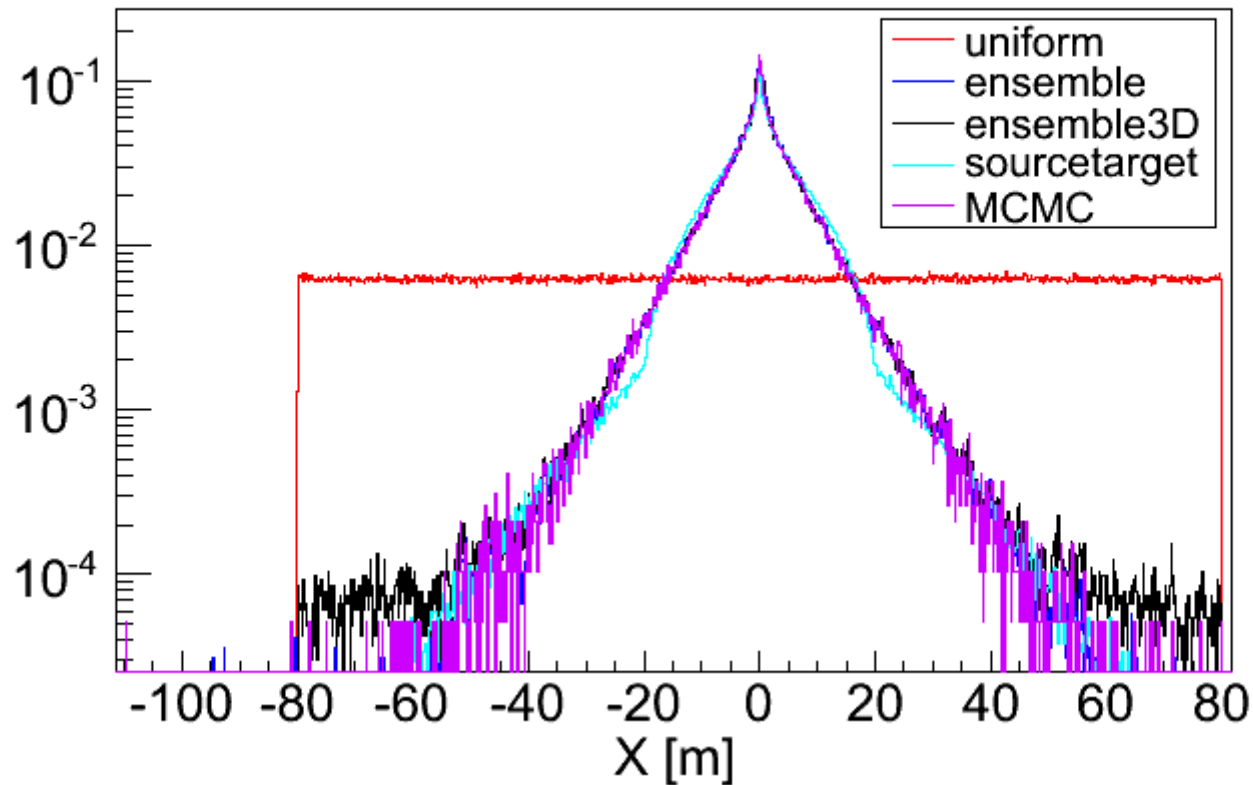
Vertex position (nscat = 1, vertex = 1)



Only “sourcetarget” has
true singularities

Sample distributions

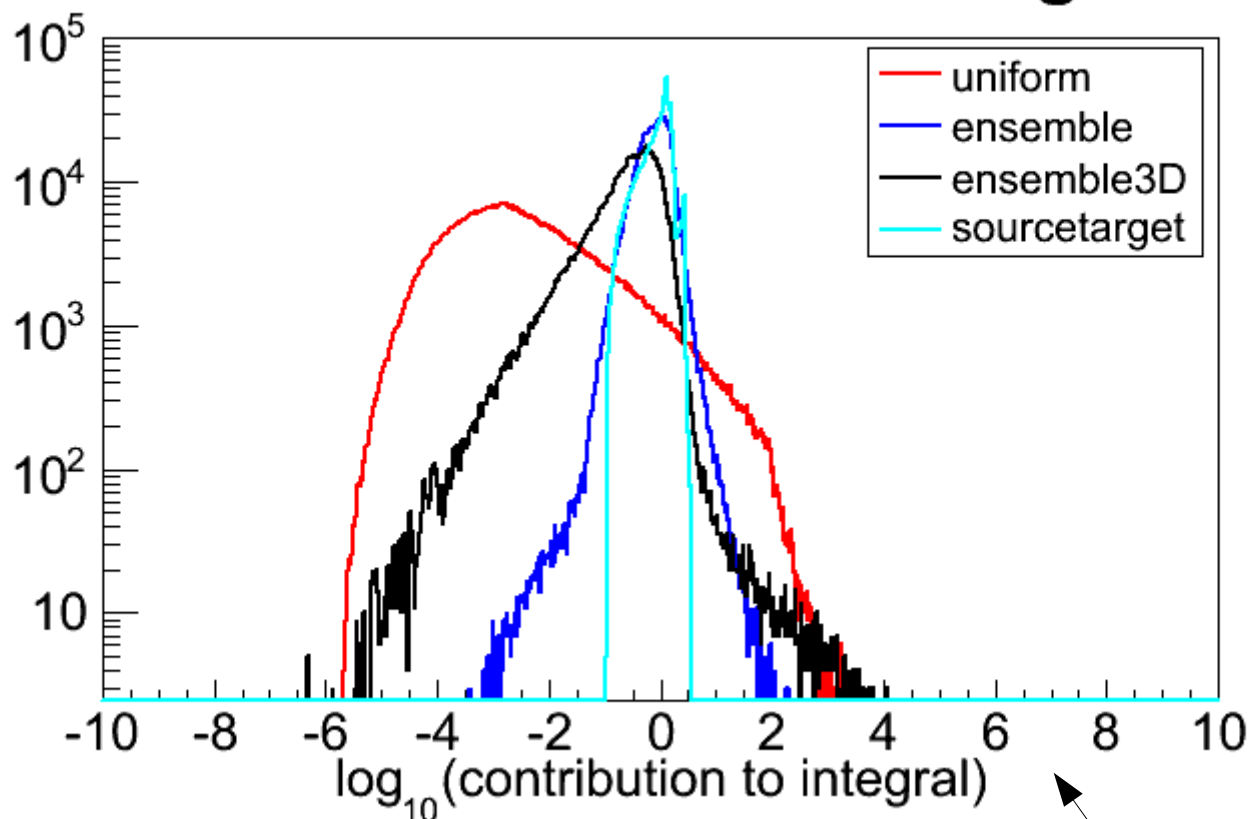
Vertex position (nscat = 1, vertex = 1)



Only “sourcetarget” has
true singularities

Sample distributions

Contributions to the integral



Should be as narrow as possible

- low values are wasted samples
- long tail on the right makes the answer unstable

Contributions multiplied by $1e5$ for readability

Results

Method	Result x 1e5	Error
tracking	0.968	0.0081
uniform sampling	0.959	0.0421
ensemble sampling	0.985	0.0027
ensemble3D sampling	0.921	0.0560
sourcetaget sampling	0.983	0.0005

- Based on running the sampling integration 10 times
- Sourcetaget sampling is the most stable
- Good agreement with tracking:
 $0.983 - 0.968 = 0.015$
(~2 x tracking error)

Conclusions

- Using MCMC and photon tracking within the same framework
 - vertex distributions are compatible for two test cases
- MC integration of the path probability density
 - works
 - singularities have to be handled with care
 - results are consistent with those of photon tracking
 - with some clever tricks it is much more efficient than photon tracking (at least for single scattering)

Backup slides

Example scenario 2

- Single scattering probability from photon tracking
 - tracked photons **$N = 20 \times 500 \text{ M}$**
 - single scatter hits **$n_{\text{hits}} = 14,174$**
 - $\text{Phit} = n_{\text{hit}}/N = 1.4174 \times 10^{-6}$
 - $\sigma = \pi 0.2159^2 = 0.146 \text{ m}^2$
 - **$\text{Phit} / \sigma = 0.968 \times 10^{-5}$**
 - error: $1/\sqrt{n_{\text{hits}}} \sim 0.8\%$

Scattering model

- Combination of two scattering models
- Henyey-Greenstein
 - 60.241 m
 - $g = 0.924$
- Rayleigh
 - 294.118 m
 - $a = 0.853$

