Reconstruction of muon trajectories

Workshop on reconstruction

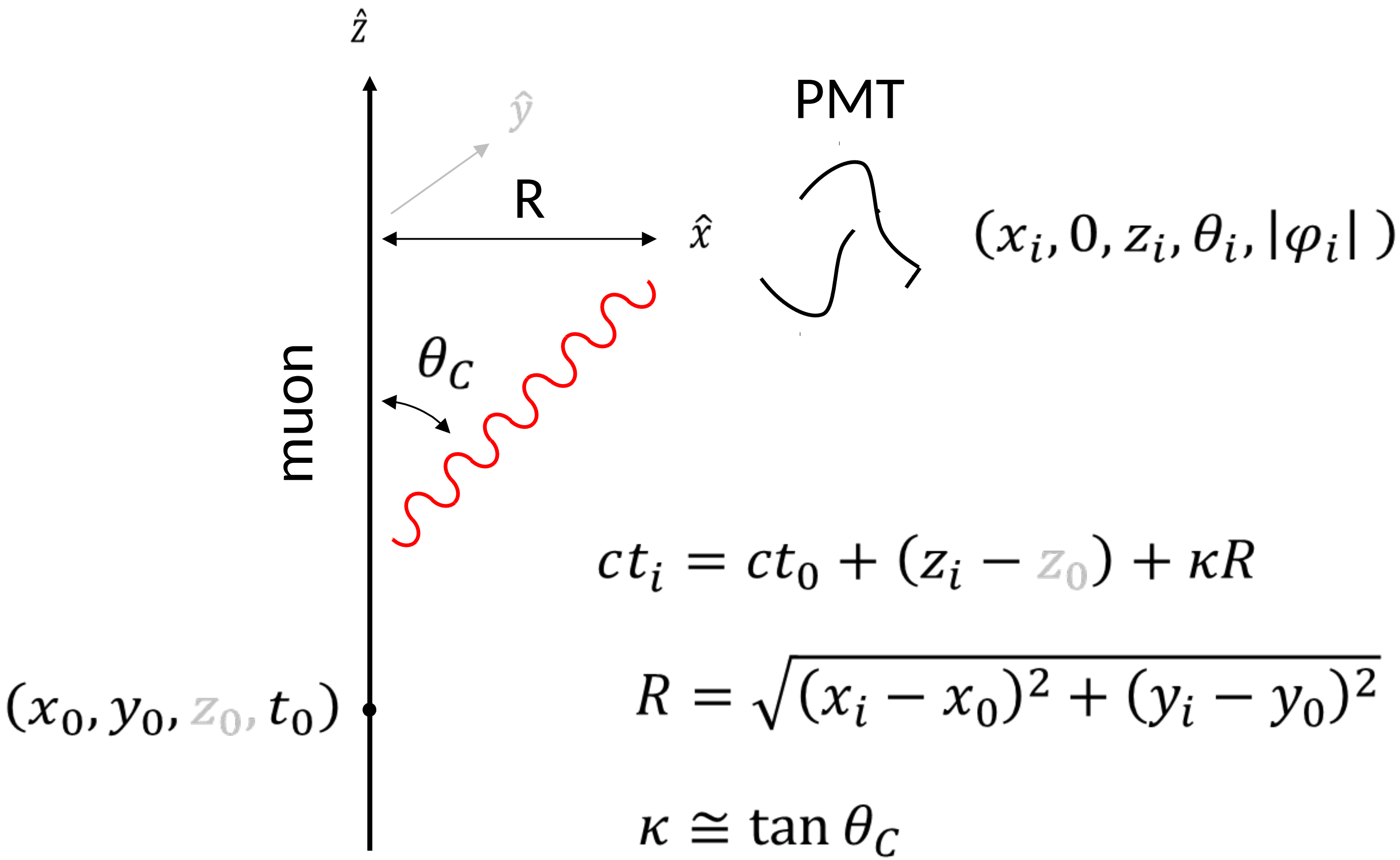
7-9 December 2016, Paris

M. de Jong

Introduction (1/5)

- The fit of a muon trajectory to the data is a non-linear problem
  - in the absence of light scattering and for a given direction of the muon, the problem has a linear solution [ANTARES-SOFT-2007-001]
- The PDF of the arrival time of Cherenkov light is a multi-dimensional function
  - calculation for direct and single-scattered light exists [ANTARES-SOFT-2010-002]
  - first derivatives of PDF are zero when fit parameters are too far off from true muon

Introduction (2/5)



$$ct_i = ct_0 + (z_i - z_0) + \kappa R$$

$$R = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2}$$

$$\kappa \cong \tan \theta_C$$

Introduction (3/5)

$$ct_i' \equiv \frac{ct_i - (z_i - z_0)}{\kappa}$$

$$ct_0' \equiv \frac{ct_0}{\kappa}$$

$$(ct_i' - ct_0')^2 = (x_i - x_0)^2 + (y_i - y_0)^2$$

$$ct_i'^2 - 2ct_i'ct_0' + ct_0'^2 = x_i^2 - 2x_ix_0 + x_0^2 + y_i^2 - 2y_iy_0 + y_0^2$$

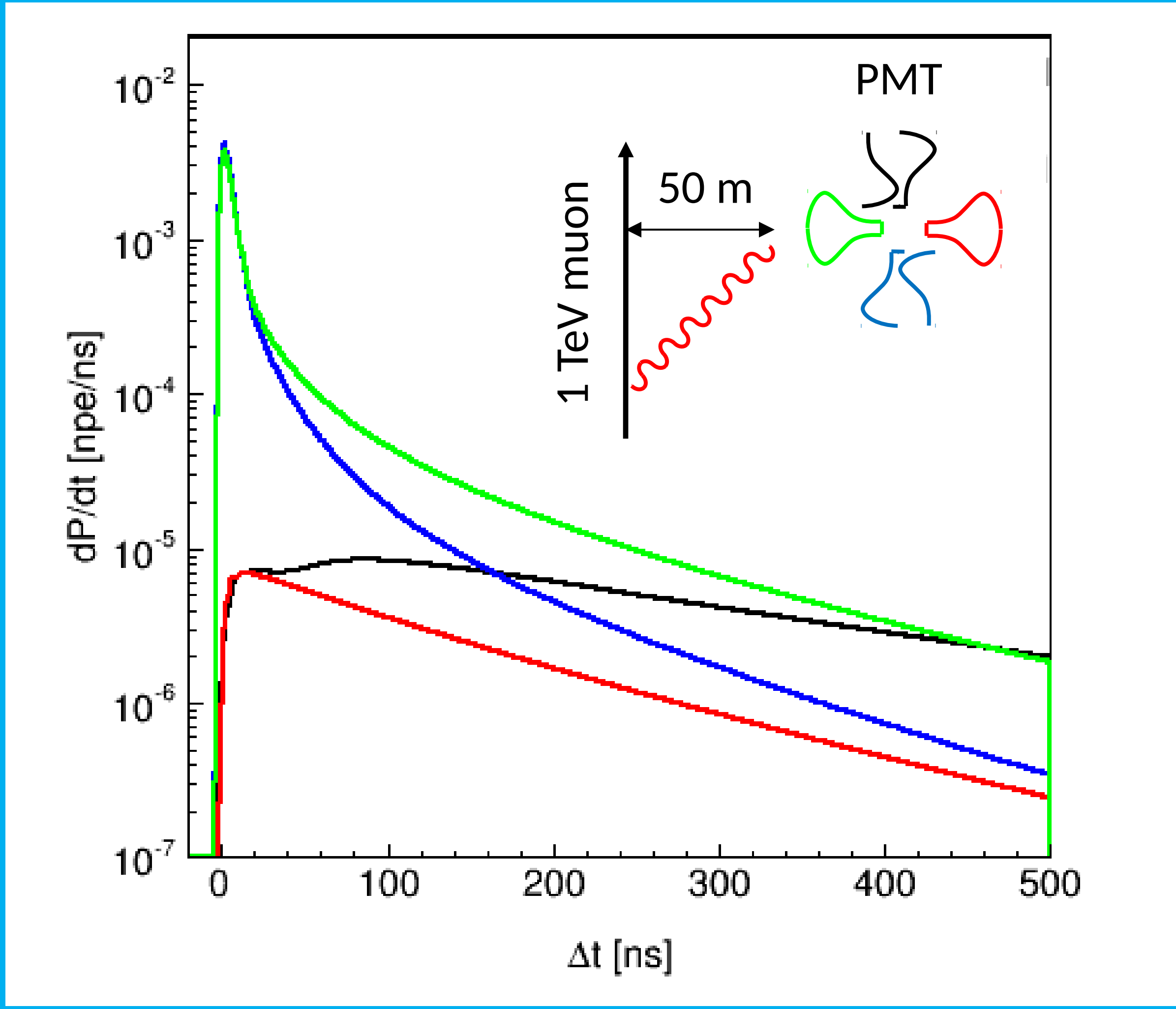
$$ct_j'^2 - 2ct_j'ct_0' + ct_0'^2 = x_j^2 - 2x_jx_0 + x_0^2 + y_j^2 - 2y_jy_0 + y_0^2$$

$-$

$$ct_i'^2 - ct_j'^2 + 2(ct_j' - ct_i')ct_0' = x_i^2 - x_j^2 + 2(x_j - x_i)x_0 + y_i^2 - y_j^2 + 2(y_j - y_i)y_0$$

parameters x_0, y_0 and t_0 appear in a linear way

Introduction (4/5)



Introduction (5/5)

1) JPrefit
    - input JDAQEvent ← *standard DAQ data format*
    - output vector<JFit>
2) JSimplex
    - I/O vector<JFit>
3) JGandalf
    - I/O vector<JFit>
4) JEnergy *intermediate data format*
    - I/O vector<JFit>
5) JStart
    - I/O vector<JFit>
6) JEvt
    - input vector<JFit>
    - output Evt → *standard offline data format*

JPrefit

- scan of solid angle using angular grid
  - cluster of L1 [and L0] hits in each direction
  - 3 parameter linear fit of $\{x_0, y_0, t_0\}$
- used algorithms
  - clusterizeWeight(.., JMatch3B/JMatch1D)
    - cluster algorithm (based on “clique”), see Analysis e-log 34
  - next_permutation()
    - all $\binom{n}{m}$ permutations of a subset of m out of n hits
  - JMatrixNZ
    - co-variance matrix to accommodate grid angle (next slides)
  - JEstimator<JLine1Z>

Covariance matrix (1/3)

$$\chi^2 = \sum_{i=1}^{n} \frac{(t_i - t_i')^2}{\sigma_i^2}$$

⇩

$$\chi^2 = \overline{y} \times V^{-1} \times y$$

$$y = \begin{pmatrix} t_1 - t_1' \\ \vdots \\ t_n - t_n' \end{pmatrix} \qquad V = \begin{pmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_n^2 \end{pmatrix}$$

Covariance matrix (2/3)

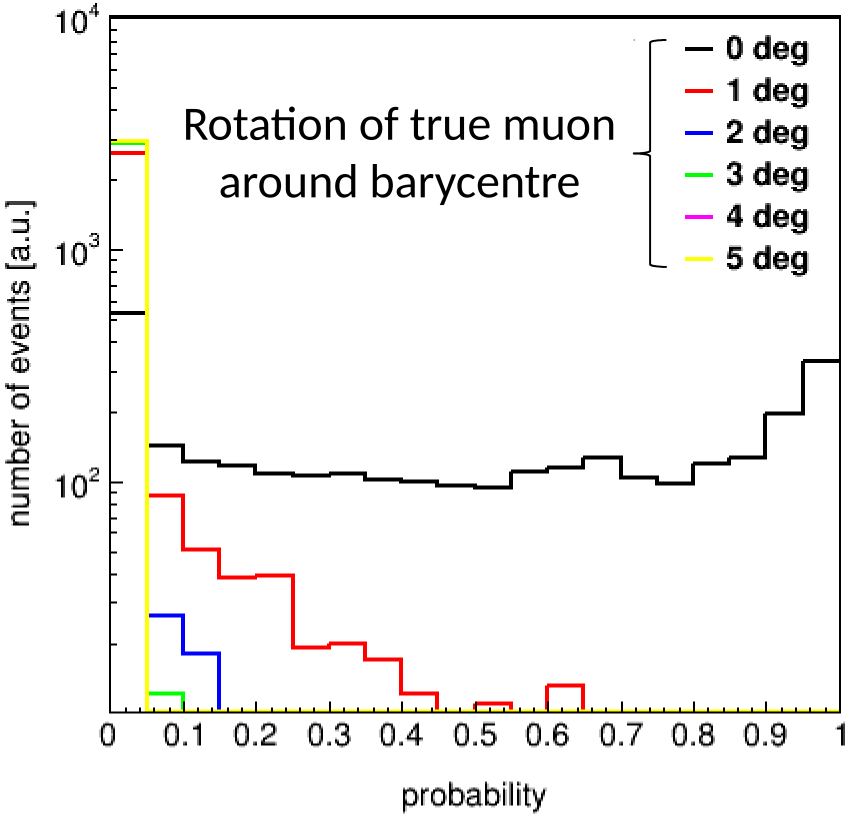
- Finite grid angle in JPrefit can be accommodated in covariance matrix using known uncertainties in direction cosines δT_x and δT_y

$$V'_{ij} \mathrel{+}=^{\P} \frac{\partial t'_i}{\partial T_x}\frac{\partial t'_j}{\partial T_x}\delta T_x^2 + \frac{\partial t'_i}{\partial T_y}\frac{\partial t'_j}{\partial T_y}\delta T_y^2$$

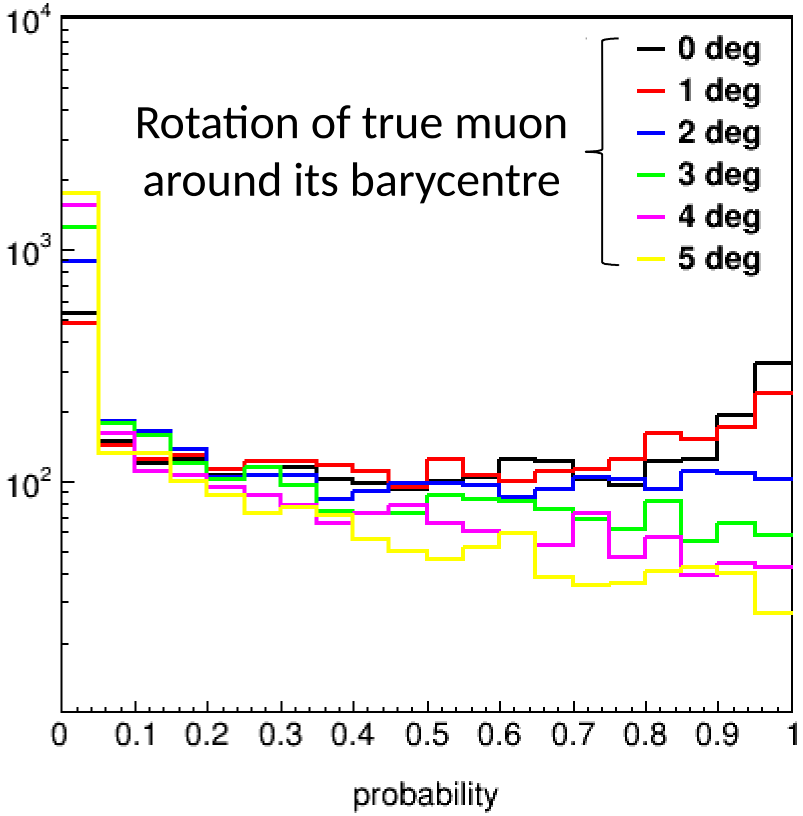
¶ Assumption of Gaussian errors – add errors in quadrature.

Covariance matrix (3/3)

normal χ^2



modified χ^2



up to few degrees, modified χ^2 is okay

JSimplex

- selection of N best start values
  - e.g. from JPrefit
  - first L1 and L0 hits within road width and time window
  - 5 parameter M-estimator fit of $\{x_0, y_0, t_0, T_x, T_y\}$
    - Powell's method
- used algorithms
  - JRegressor<JLine3Z, JSimplex>

$$\sum_{i}^{hit} \log\left(1 + 0.5\,\frac{(t_i - t_i')^2}{\sigma_i^2}\right)$$

JGandalf

- selection of N best start values
  - e.g. from JSimplex
  - first L0 hits within road width and time window
  - 5 parameter likelihood fit of $\{x_0, y_0, t_0, T_x, T_y\}$
    - Levenberg-Marquardt method
    - $\mathcal{F}(E, R_i, \theta_i, |\phi_i|, \Delta t_i)$
- used algorithms
  - JRegressor<JLine3Z, JGandalf>

$$\sum_i^{hit} -\log\left(\mathcal{F}\left(E, R_i, \theta_i, |\phi_i|, \Delta t_i \middle| x_0, y_0, t_0, T_x, T_y\right)\right)$$

JEnergy

- selection of N best start values
  - e.g. from JGandalf
  - all PMTs within road width
  - 1 parameter likelihood fit of $\{\log E\}$
    - Levenberg-Marquardt method
    - $npe_i = \int \mathcal{F}(E, R_i, \theta_i, |\phi_i|, t)\, dt$
- used algorithms
  - JRegressor<JEnergy, JGandalf>

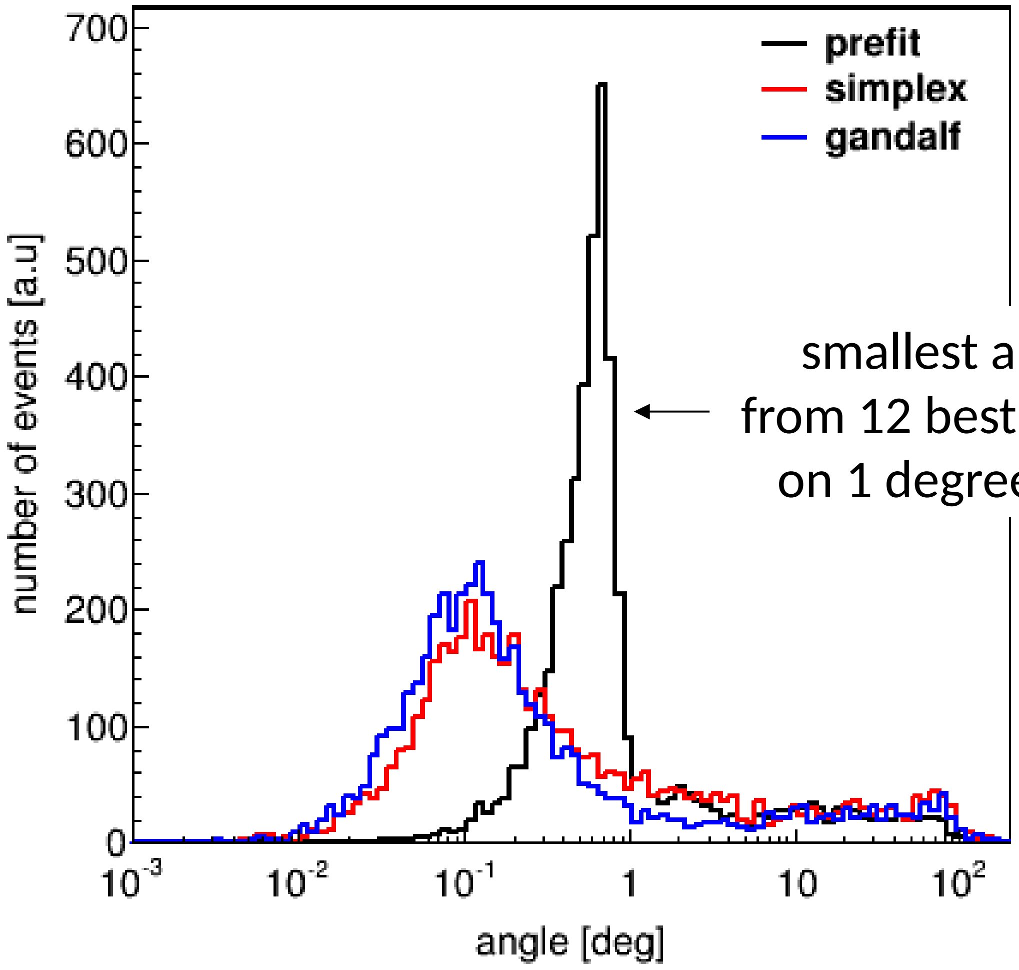
$$\sum_{i}^{\overline{PMT}} -\log\big(P(0|npe_i)\big) + \sum_{i}^{PMT} -\log\big(1 - P(0|npe_i)\big)$$

JStart

- selection of N best start values
  - e.g. from JEnergy
  - all PMTs within road width
  - 1 parameter scan of $\{z_i\}$ with i the PMT index
    - $npe_i = \int \mathcal{F}(E, R_i, \theta_i, |\phi_i|, t)\, dt$
    - moves track position $\{x_0, y_0, z_0\}$ to actual start z
- used algorithms
  - JRegressor<bool>
    - determines likelihood difference between signal + background (H1) and background only (H0) hypotheses for track segment at position z

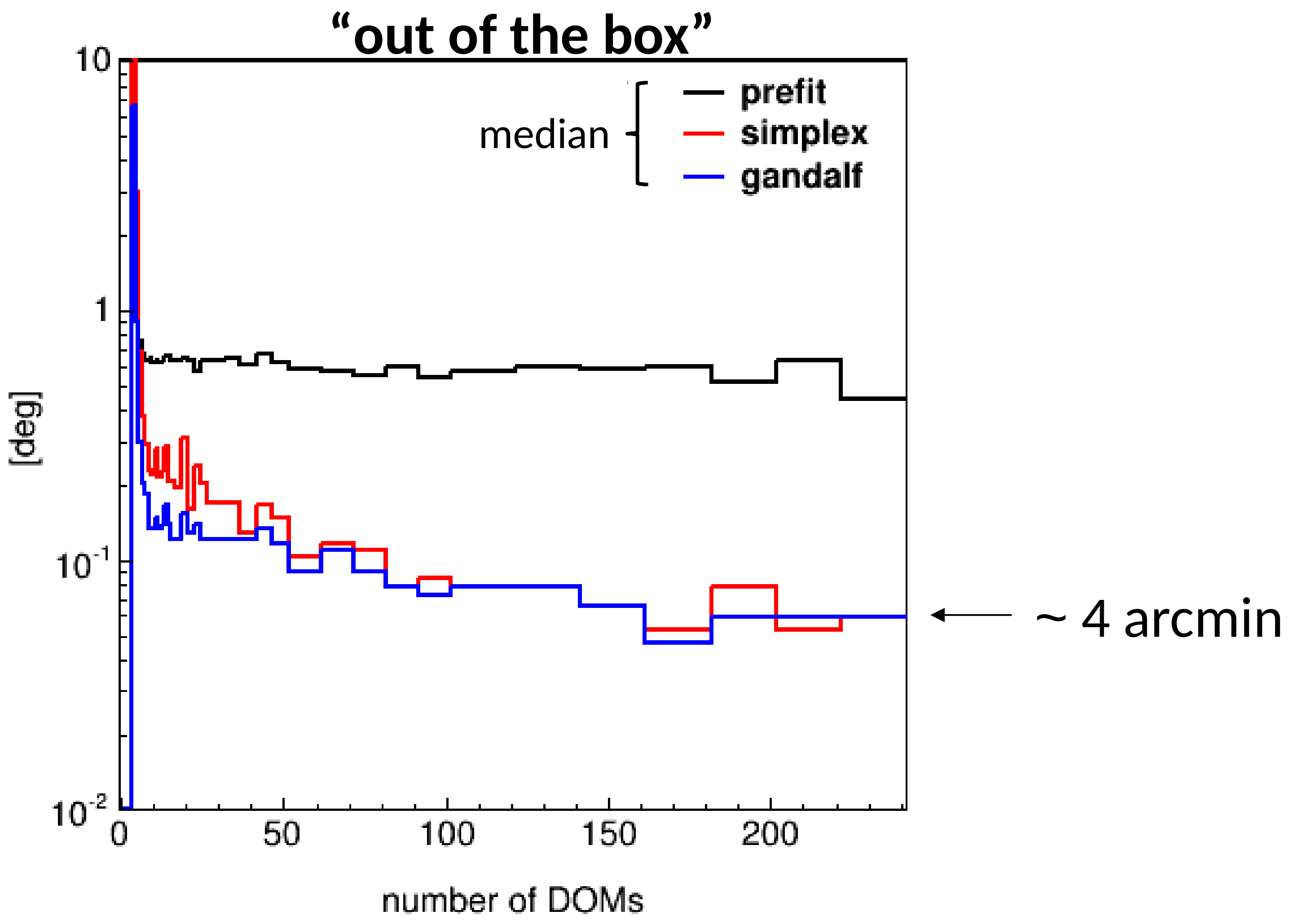
Angular resolution (1/4)

"out of the box"



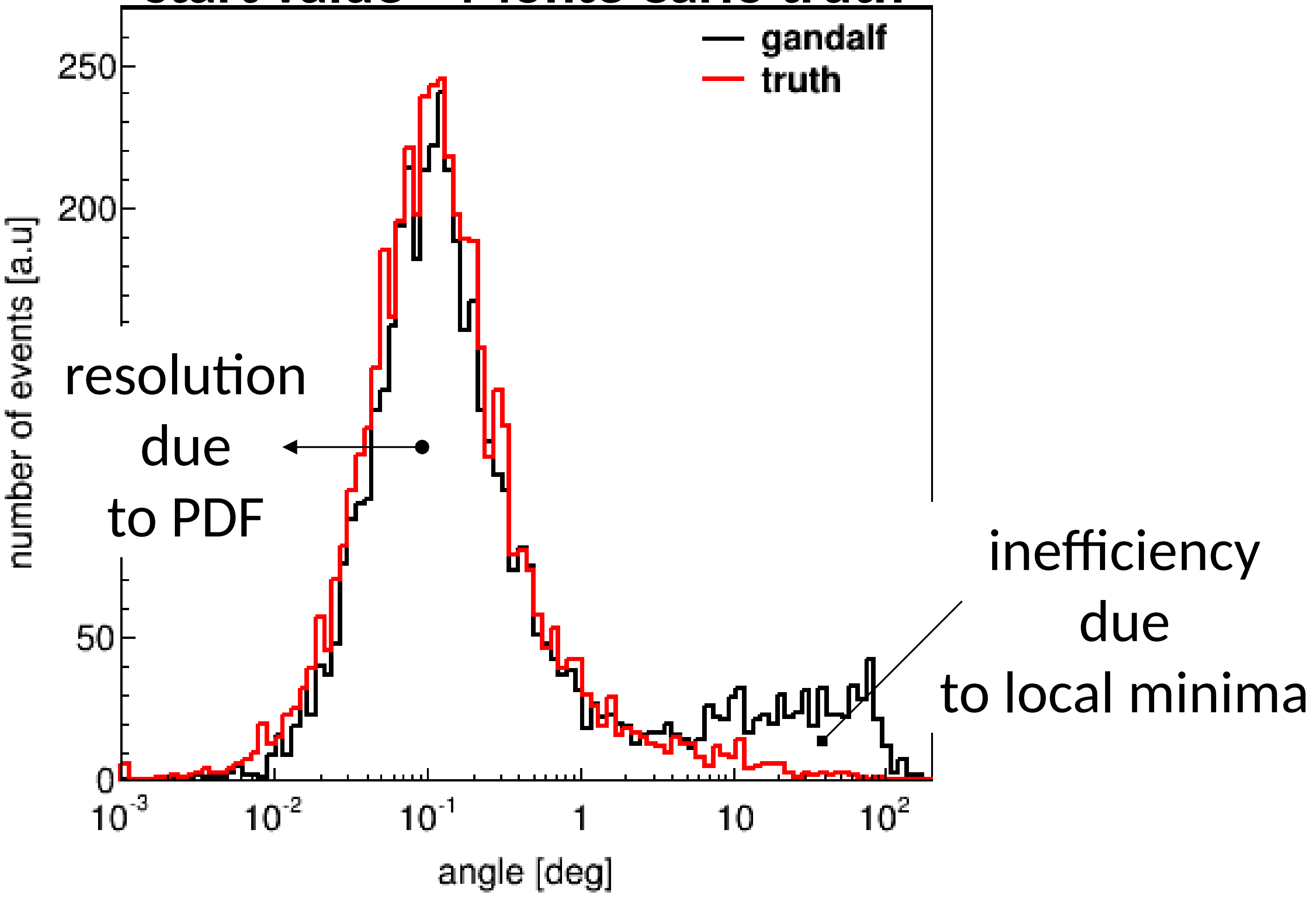
smallest angle
from 12 best prefits
on 1 degree grid

Angular resolution (2/4)



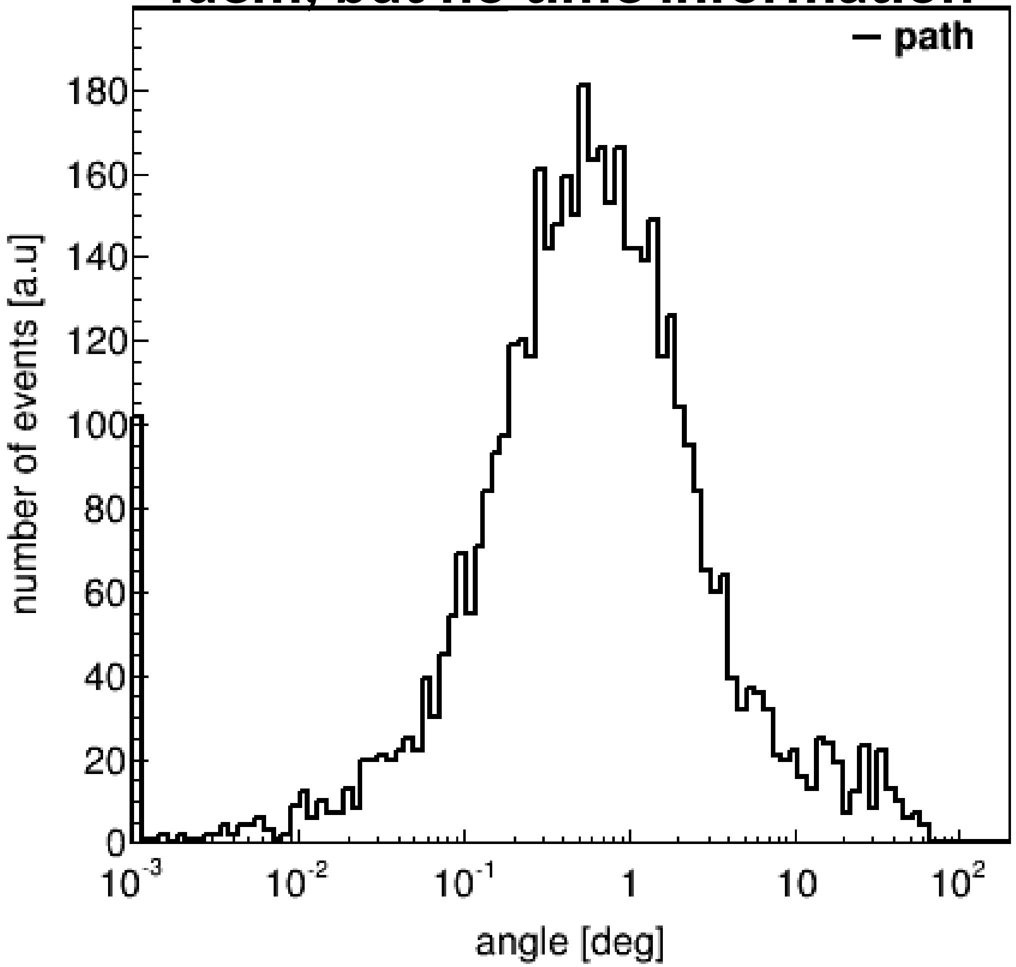
Angular resolution (3/4)

start value = Monte Carlo truth



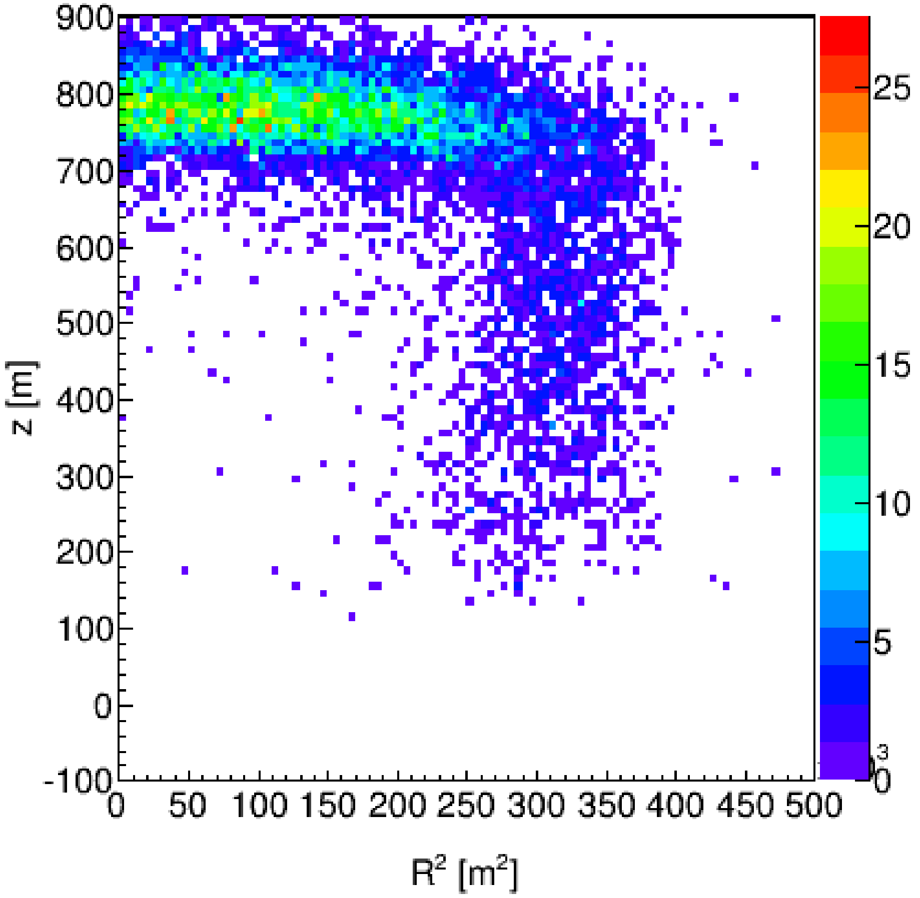
Angular resolution (4/4)

idem, but no time information



Start position

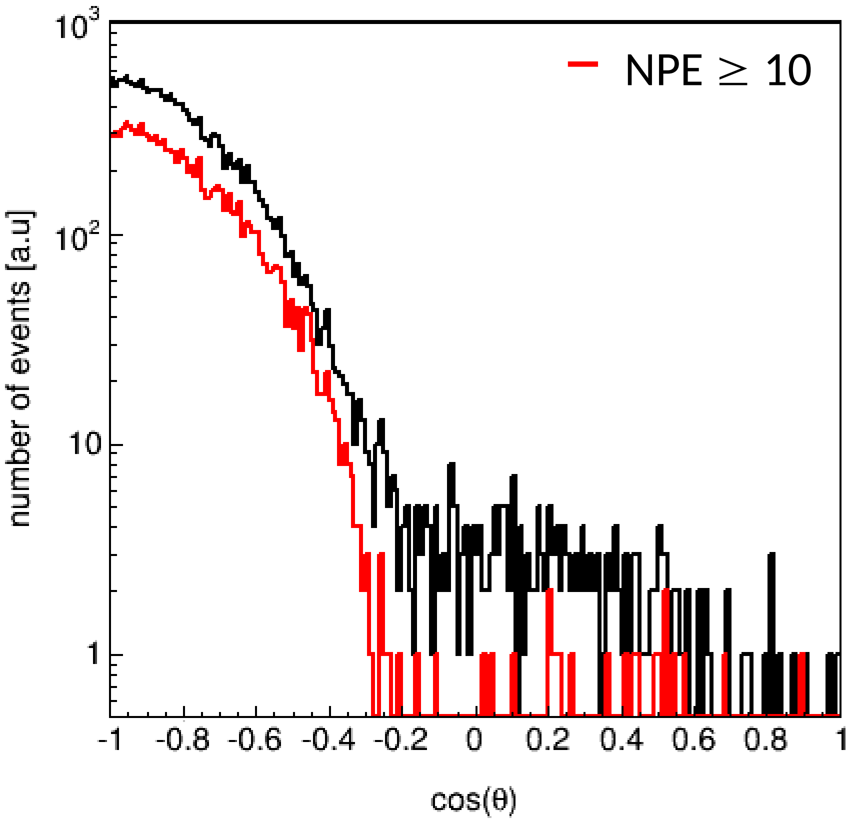
Atmospheric muons with energy E ≥ 10 TeV

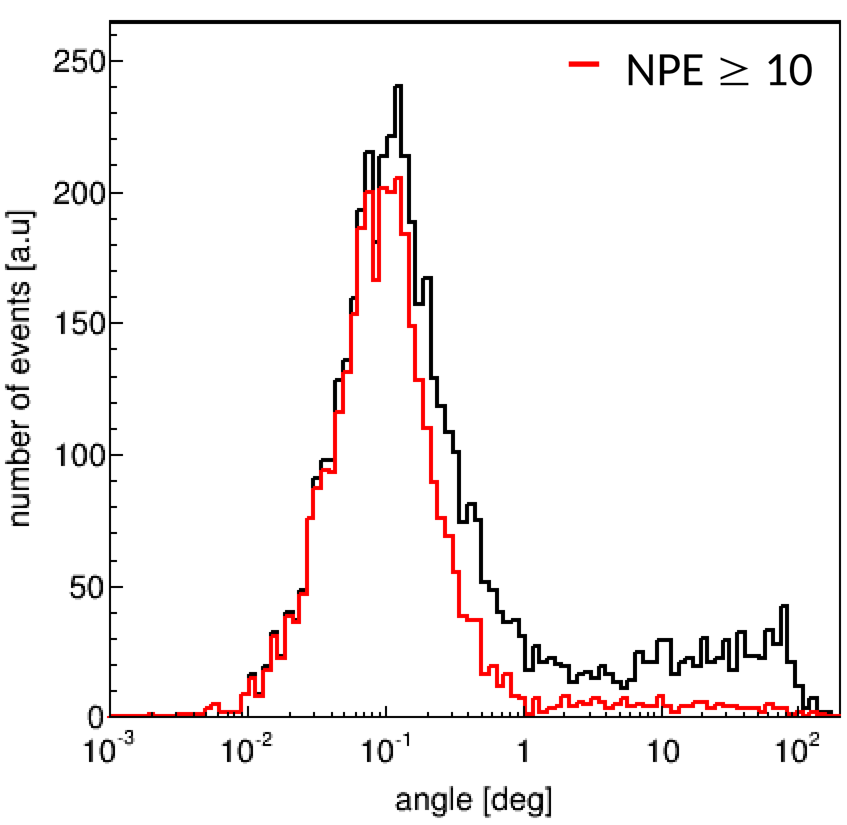


number of photo-electrons

muons

neutrinos





CPU usage (1/2)

CPU usage (2/2)

Application	Live time [s]	CPU time[§] [s]	Ratio CPU/Live
MUPAGE[¶]	2530	?	?
JSirene	2530	10328	4
JTriggerEfficiency	2530	9914	4
JPrefit	2530	14263	6
JSimplex	2530	1388	½
JGandalf	2530	7087	3
JEnergy	2530	2757	1
Total	2530	45737	18

¶ Threshold 100 GeV

§ 1 ARCA block; Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz

Summary

- Jpp constitutes a general purpose framework for a.o. fitting models to data
  - cluster algorithms
  - permutations
  - PDFs, PDFs and more PDFs
  - linear estimators
  - minimisers

Outlook

- JPrefit - JSimplex - JGandalf - JEnergy - JStart
  - excellent performance
  - low CPU usage
  - room for improvements
    - fine tuning parameters (< 10)
    - selection of start values
    - PDF optimisation
    - modelling of bremsstrahlung
    - shorten chain
- JSirene
  - run, test, be happy (or not)
  - prepare input from other applications…