Progress on reconstruction.

Aart Heijboer, Nikhef.

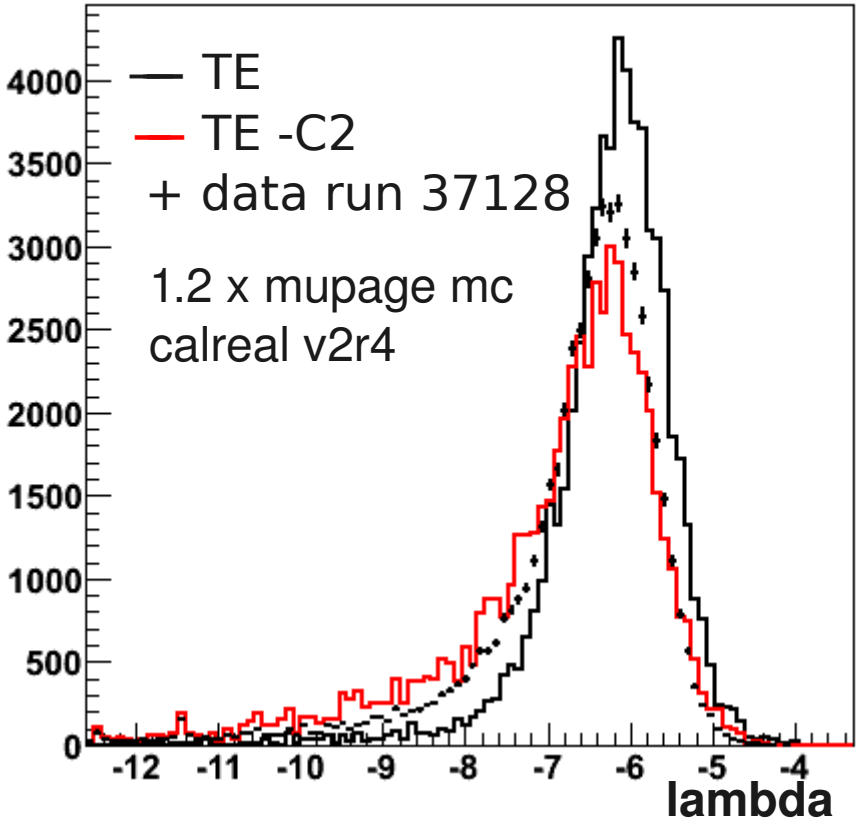
- new package available in CVS
- new version of 'AartStrategy'
- name: aafit

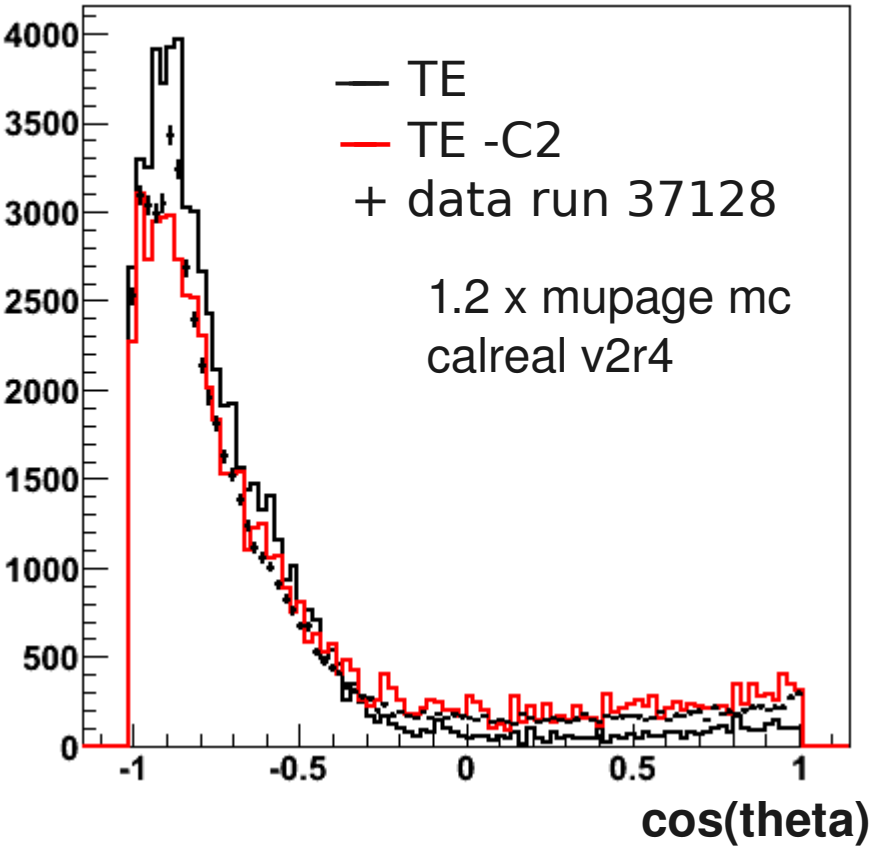
this talk:

- movitvation
- a bit of 'infomercial'
- some basic results

Context

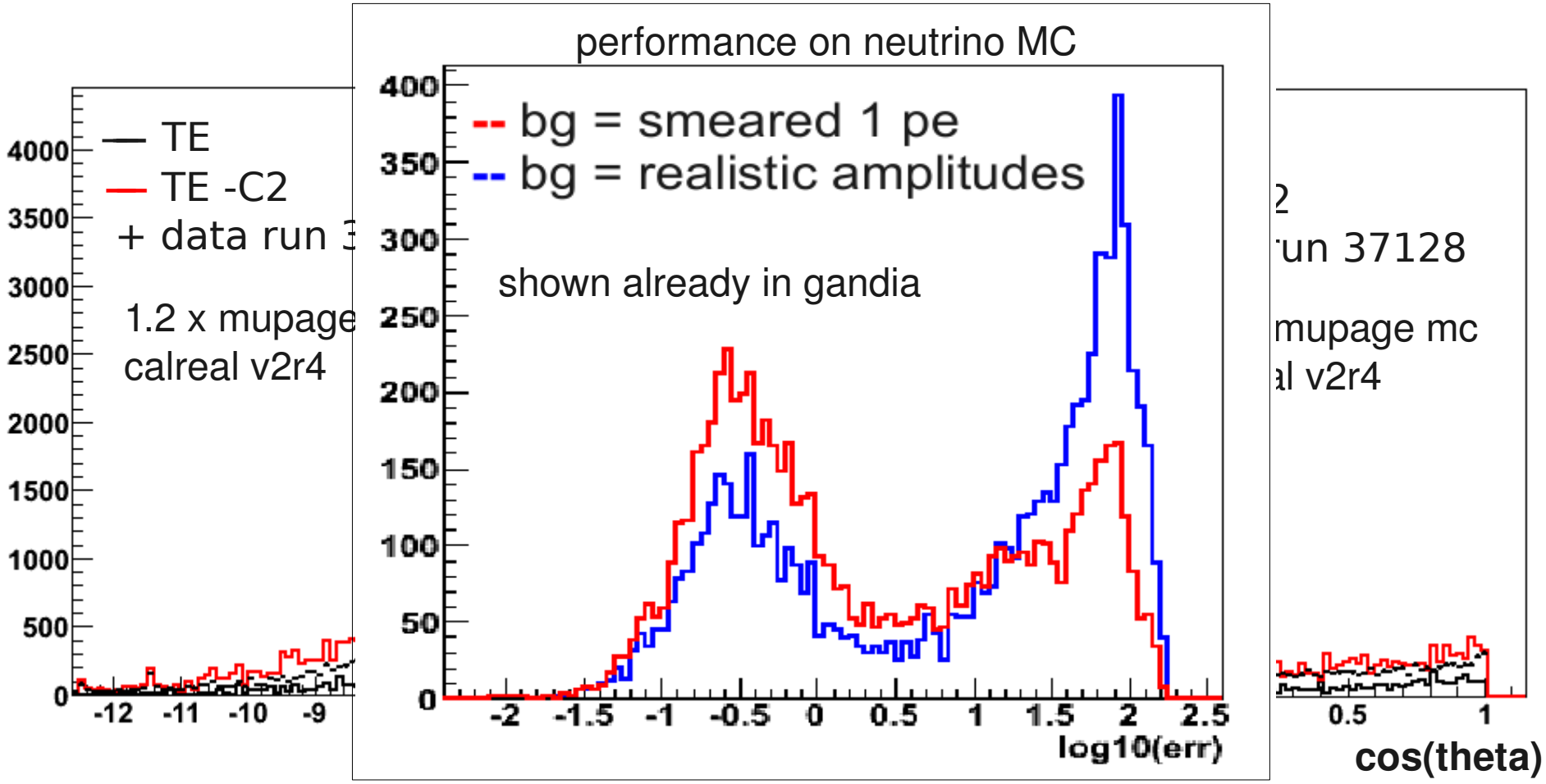
- correct background hit amplitudes need to be simulated
  - improves data/MC agreement (shown in Gandia using own simulation, now Trig Eff)
  - improvement! but...





Context

- correct background hit amplitudes need to be simulated
  - improves data/MC agreement
  - but serious deterioration of reconstruction performance → need to recover



Goals

- ***Rely less on hit-amplitude in signal-hit selection***
  - need to not assume they're all signal... *modifiy hit selections (=the easy part)*
  - missing some (high-amplitude) signal hits → 5% inefficiency → recoverd by:
    - new first selection: coincidences extended with bb-fit like hit selection
    - require them to obey causality relation (select largest causal cluster)

- ***NAG -> GSL***
  - ***Nag = Numerical Algorithms Group***
    - work *very* well, and fast,. but
    - non-free software
    - C-version only usable at lyon (precise licensing unclear)
    - older static fortran library works well (also off-site), but
      - may or may not be legal to use outside Lyon
      - we have no 64-bit version (impossible to use in 64bit exe's)
  - ***GSL = Gnu Scientific Library***
    - free
    - widely used
    - ~well documented
    - less black-boxy

Minimizers

E04 – Minimizing or Maximizing a Function **E04DGF/E04DGA**

NAG Fortran Library Routine Document

E04DGF/E04DGA

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

Note: *this routine uses* **optional parameters** *to define choices in the problem specification and in the details of the algorithm. If you wish to use* default *settings for all of the optional parameters, you need only read Section 1 to Section 9 of this document. Refer to the additional Section 10 and Section 11 for a description of the algorithm and the specification of the optional parameters.*

1 Purpose

E04DGF/E04DGA minimizes an unconstrained nonlinear function of several variables using a pre-conditioned, limited memory quasi-Newton conjugate gradient method. First derivatives (or an 'acceptable' finite difference approximation to them) are required. It is intended for use on large scale problems.

E04DGA is a version of E04DGF that has multithreaded applications (see Section 5 b called prior to calling E04DGA.

2 Specifications

2.1 Specification for E04DGF

```
SUBROUTINE E04DGF(N, OBJFUN,
1                 USER, IFAIL
INTEGER           N, ITER, IW
real              OBJF, OBJGR
EXTERNAL          OBJFUN
```

2.2 Specification for E04DGA

```
SUBROUTINE E04DGA(N, OBJFUN,
1                 USER, LWSAV
INTEGER           N, ITER, IW
real              OBJF, OBJGR
LOGICAL           LWSAV(120)
```

Chapter 35: Multidimensional Minimization 387

`gsl_multimin_fdfminimizer_vector_bfgs2` [Minimizer]
`gsl_multimin_fdfminimizer_vector_bfgs` [Minimizer]

These methods use the vector Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm. This is a quasi-Newton method which builds up an approximation to the second derivatives of the function f using the difference between successive gradient vectors. By combining the first and second derivatives the algorithm is able to take Newton-type steps towards the function minimum, assuming quadratic behavior in that region.

The `bfgs2` version of this minimizer is the most efficient version available, and is a faithful implementation of the line minimization scheme described in Fletcher's *Practical Methods of Optimization*, Algorithms 2.6.2 and 2.6.4. It supercedes the original `bfgs` routine and requires substantially fewer function and gradient evaluations. The user-supplied tolerance *tol* corresponds to the parameter σ used by Fletcher. A value of 0.1 is recommended for typical use (larger values correspond to less accurate line

[Minimizer]
…adient of the function at each
… increased by a factor of two.
…hen the algorithm backtracks
…l. A suitable value of *tol* for
… is inefficient and is included

…he function at each evaluation

[Minimizer]
[Minimizer]
…nd Mead. Starting from the

exit E04DGF

great progress

enter gsl_multimin_fdfminimizer_vector_bfgs2

- both are quasi-newton, conjugate gradient methods requiring (analytical) 1st derivatives.
- but GSL implementation seems less forgiving → debug

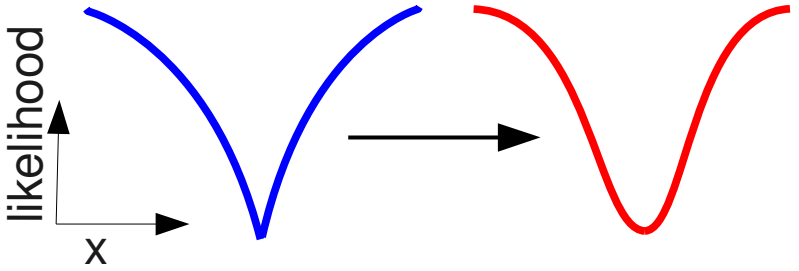
Bug fixes and other improvements

Several issues found while test driving the GSL minimizer...

- Scale the problem, so that angles have numerical values of same order as the positions. (i.e. scale up angles by factor 100).
  - observe better convergence, especially with the gsl minimizer.

- remove lookup tables used in PdfFitter
  - they cause inaccuracies in the derivatives that sometimes confuse the minimizer
  - all PDF functions now compute their analytical derivatives.

- fix wrong sing in a term of derivative of dist_function (for fullpdffitter)
  -
- insurance against track going through the om-pos
  - causes non-differentiable peak in the likelihood function → now smoothed
    k = sqrt(v*v – l*l + epsilon) , epsilon = 1 cm.



- remove capping of variables: causes derivative errors. (qq in pdffitter:nag_conj_grad_fun)

- remove angular-acceptance term from the M-estimator score function
  - results are better without it.

On the inside...

- Fitters mostly derived from reco code
- severely cleaned up and parts completely rewritten for GSL
- Internally uses new hit and track classes (with extensive io capabilities to existing formats)

- Compared to CalReal/Physics reuse the good/difficult bits, but also severely reorganize some parts which were too convoluted.
  - some parts remain complicated, though (e.g. finding the right alignment).

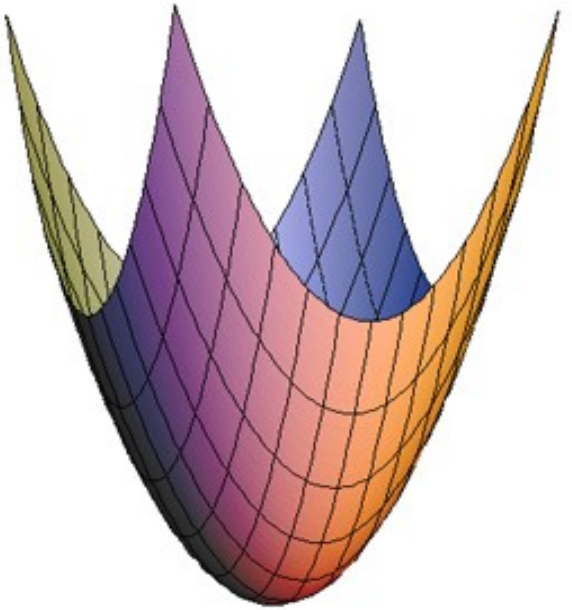
- Fitters and 'strategy' are incorporated in ROOT with rootcint
- Allows calling the algorithm from:
  - compiled c++
  - root macro (cint)
  - python script (through pyRoot)

- Dependencies: many :-(
  - Physics, antares-daq/antcc, antares-daq/Trigger,
  - GSL (currently from my sps dir:
    - the default installed version on SL5 is too old

- Interfacing with other data formats (and usage in seatray) should not be difficult

As a user...

- Deal with only one script: reconstruct.py
  - 200 line python script
  - can be called directly in batch job (no need for wrapper scripts)
  - can be configured with env. variables (for batch running)



- Input options
  - PhysicsEvents (data or TE output)
  - ascii text files
  - MCEW very easy to add

- Output options
  - Calreal-style FullEvent's (with copying of MC data)
  - My private ntuple-format (welcome to try → 2007&8 available)
  - easy to support others
  - option to through away the hits: 2007&8 data take 54GB

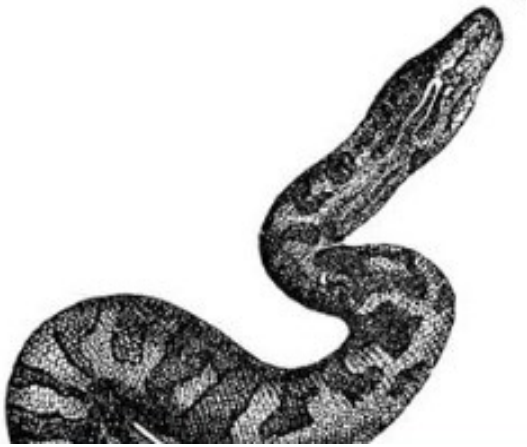
- Error estimates are back!
  - computed by inverting matrix of 2nd derivatives of likelihood function
  - still needs
    - testing (pull distributions)
    - somehow writing to the FullEvent

As a user...

excerpt from user-script reconstruct.py

```
# open the input file, prepare the strategy

f = EventFile( rec_input_file );

f2 = TFile( rec_output_file ,"RECREATE")
T2 = TTree("FULL","FullEvt Tree");
T2.Branch("FULL", fullevt ,32000, 1)

strat = AStrategy()

# event loop !
nevents = min ( f.nevents , rec_nevents )

while True:

    if not f.next() : # loads the next event
        print f.whatsup; break
    if f.index >= nevents :
        print 'done processing all the',nevents,'requested events' ; break

    if not rec_mc : cfg = get_valid_cfg ( cfgs , f.evt )

    cfg.apply_to( f.evt , True, False )

    ok = strat.fit( f.evt.spes )   # use mc_spes for mc hits

    if ok:
        e.trks = strat.fit
    else :
        failures[ strat.fail_code ] +=1

    if ok or rec_write_all_events :
        if rec_calreal_output: e.write( fullevt )
        T2.Fill()

    # print some signs of life every now and then
    if f.index % 100 == 0 :
```

up there: load ROOT and 'reco' shared library.



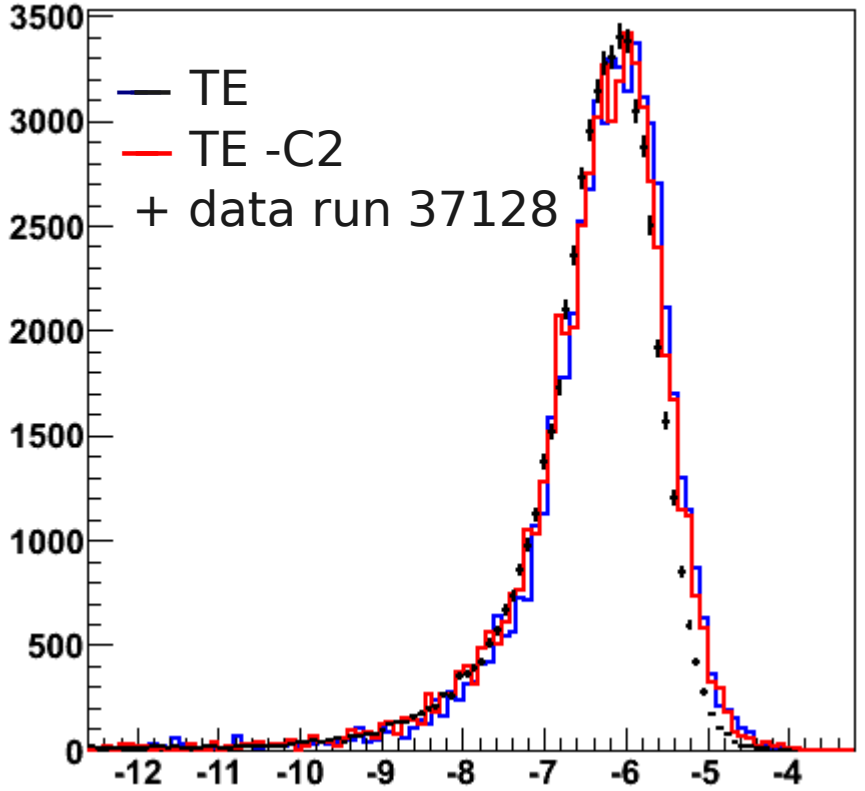
Speed

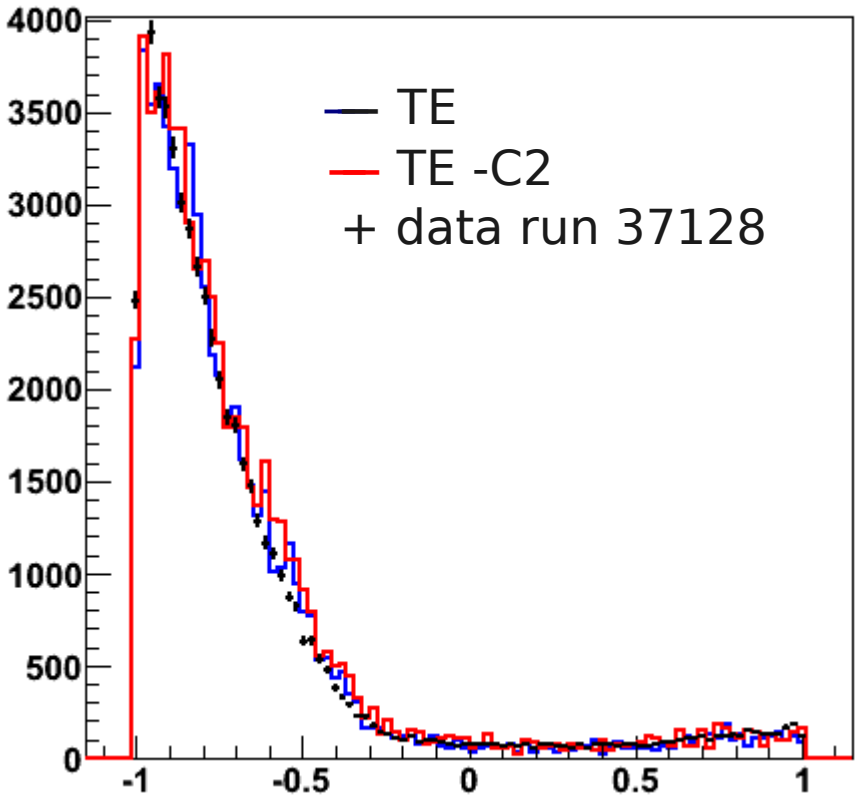
for a 12-line run

action	#calls	time	time-per-call	time-per-event	%wall-time
wall time	1	5059.4	0.0829396	**0.0829396**	100
strategy	61001	4711.9	0.077243	0.077243	**93.1316**
hit selection	61001	290.221	0.00475765	0.00475765	5.73628
M-estimate fitter	422980	1504.23	0.00355626	0.0246591	29.7314
Org Pdf fitter	386009	1551.02	0.00401808	0.0254261	30.6561
Final Pdf fitter	54768	1270.32	0.0231945	0.0208245	25.1081

- 90% of events undergoes full algorithm upto Final Pdf fit.
- 12 events per second (20 for 5-line data).
- Most time spent in minimizers.
- overhead from IO, applying calibration & alignment, etc: only 7%

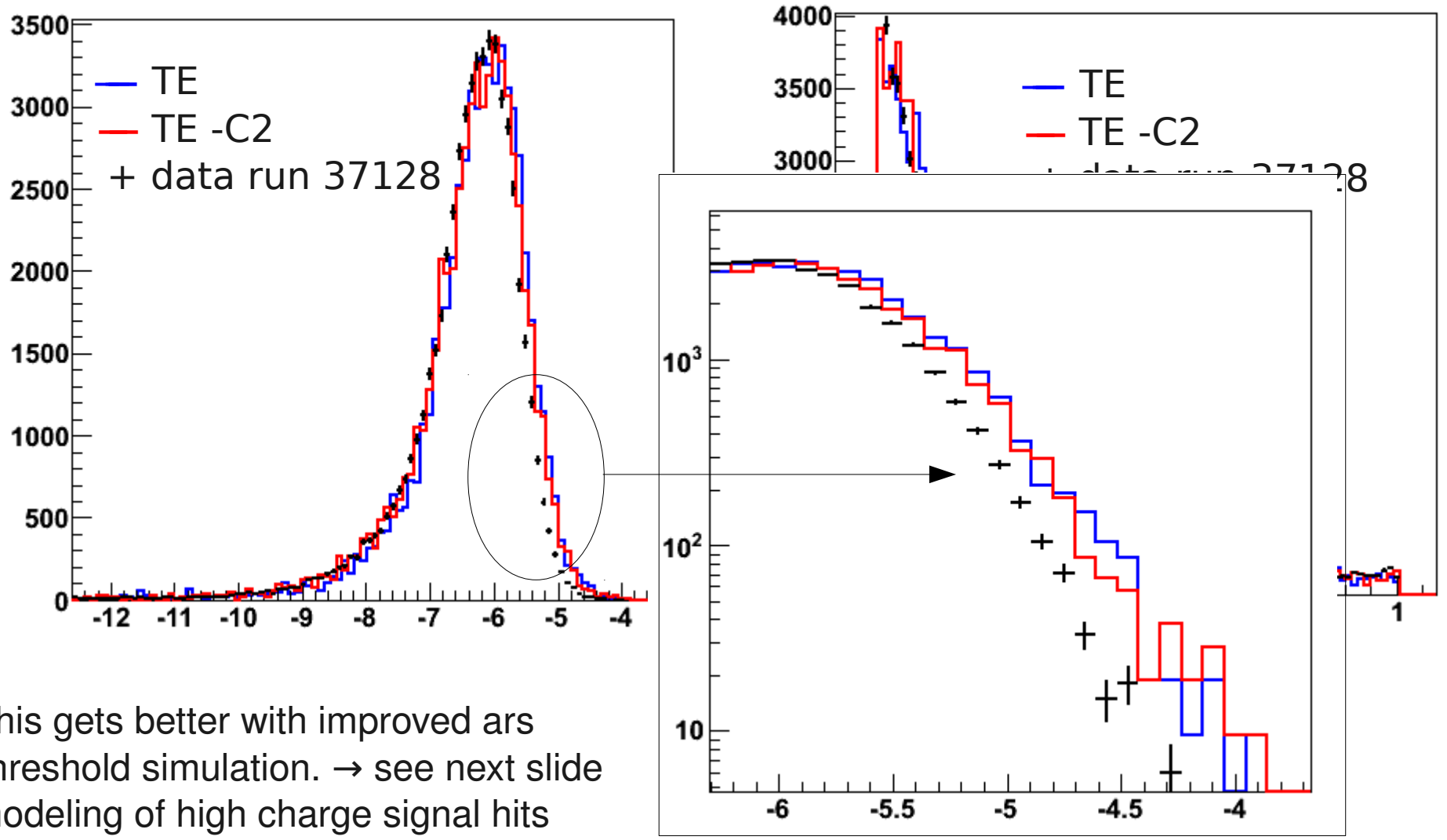
results: downgoing muons





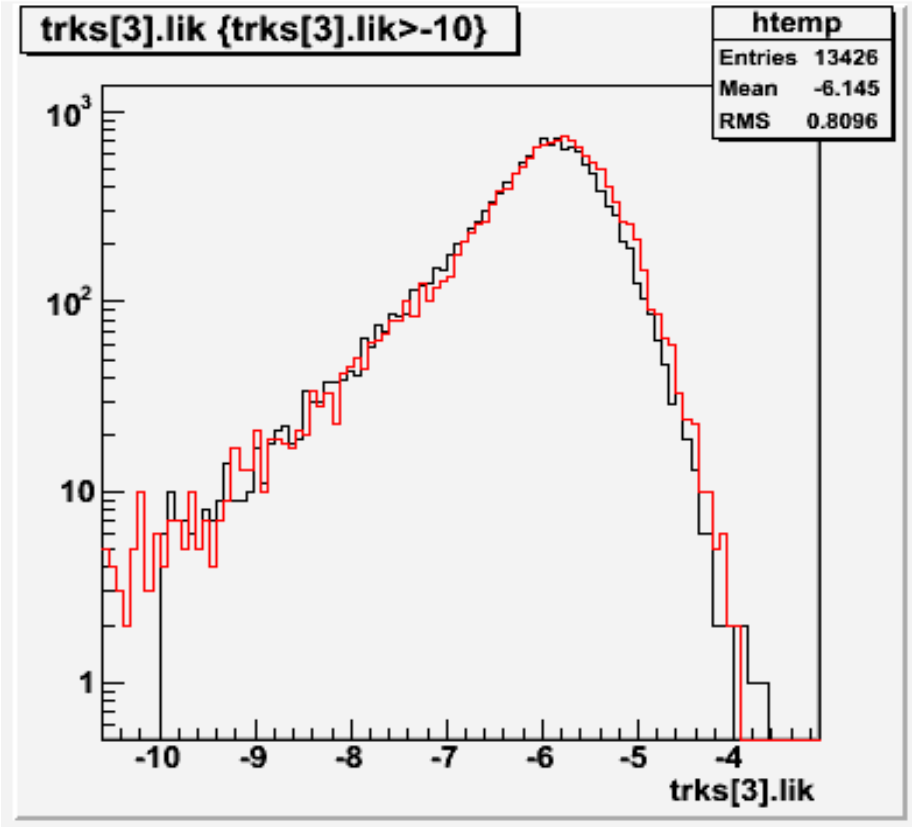
- greatly increased robustness agains wrong bg amplitude modeling
- much better data/mc agreement in general
  - not 100% percent understood why agreement with -C2 is so much better than for calreal → must still be something impefect in ampltude modeling
- but.... nothing is perfect..

results: downgoing muons

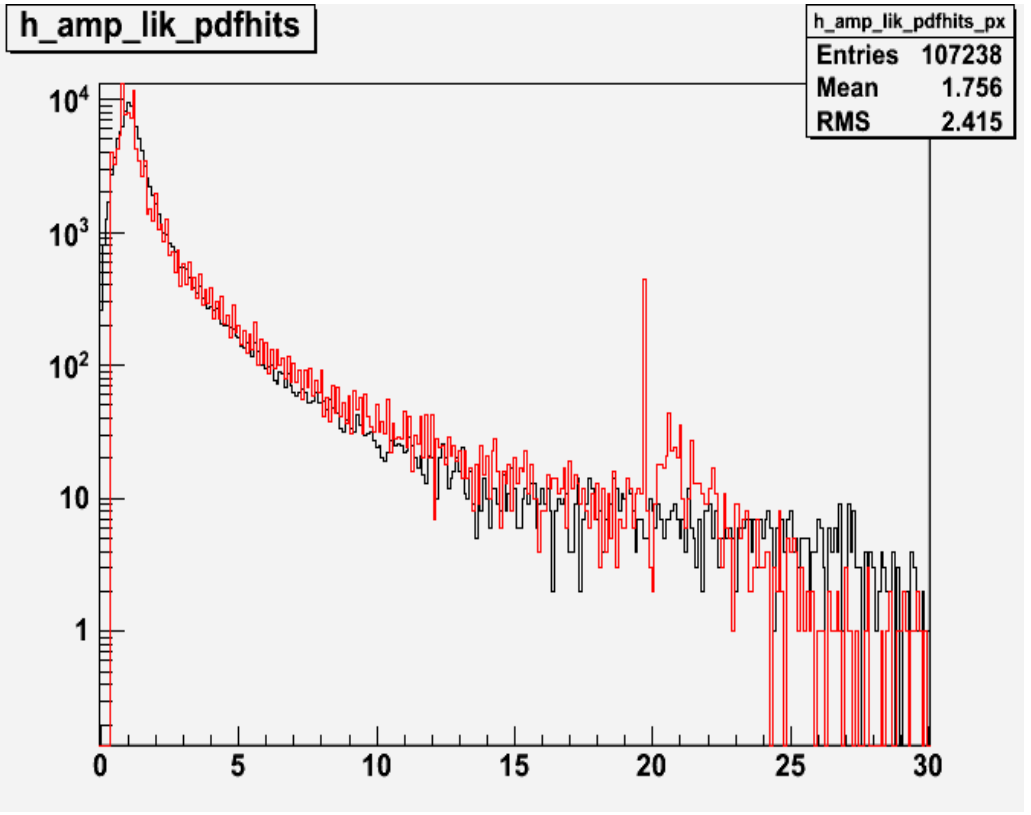


- This gets better with improved ars threshold simulation. → see next slide
- modeling of high charge signal hits can be improved too. → see next slide

Intermezzo: 2 recent observations



- comparing my production (red) with official one (only 5 line 3pe only so far)
- observe: less events in high lambda region (like we see in data).
- Difference = more realistic ARS thresholds
- → **this will be another step in right direction!**

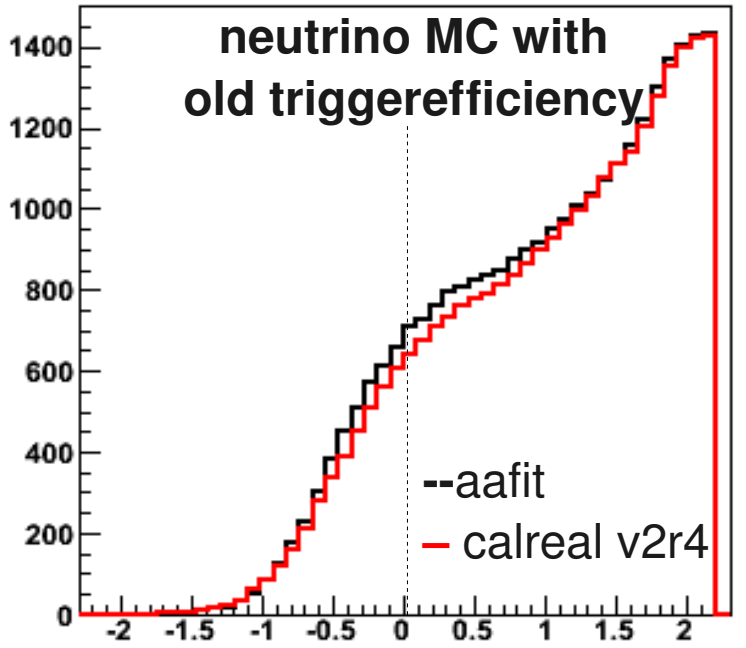


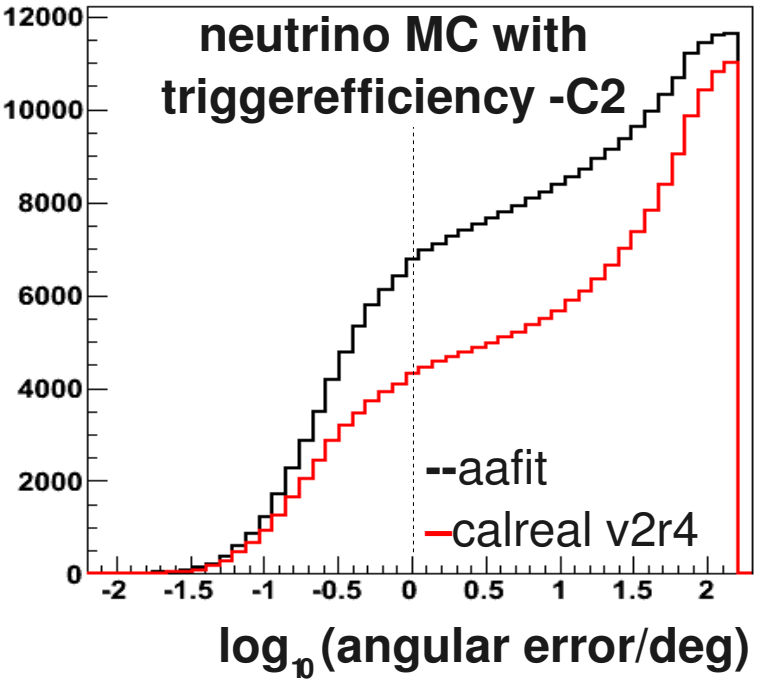
- amplitude distribution (now including signal)
- in mc 50% more hits with A>5pe than in data
- agreement clearly still not perfect
- may be reason for improved agreement in aafit vs calreal
- does it matter?? → will investigate

results: neutrino mc

comparison on ν mc with realistic amplitudes

- *aafit recovers many badly reconstructed events*
- *On the old mc : 10% more events reconstructed*
- *On MC with realistic bg, aafit reconstructs 50% more events better that 1 deg.*
- *nb: when using TriggerEfficiency without the -C2, the MC severely overestimates the performance, when using calreal.*





Summary

- New reconstruction version available
  - package name: aafit
  - severely cleaned up version of same algorithm that is in Calreal
  - Flexible file input and output options (calreal-compatible output available)
  - running happily on SL5 with 64-bit executable
- In CVS.... version tag coming

- in general
  - TriggerEfficiency -C2 big step in right direction for data/mc
  - realistic ars threstholds also matter (perhaps need to robustify algorithm some more)

- aafit
  - recovers performance on neutrino-mc
  - and improves data/mc for downgoing mu mc

- However:
  - not solution to all data/mc problems : upgoing neutrino candidates still show quite bad agreement (not shown in this talk)
  - will keep working (still many ideas) → stay tuned.