2KM mass production : vector generation

Maximilien Fechner

Special thanks to Y. Hayato, C. Walter, J. Bouchez

- MC mass production : requirements
- Vector generation : usual method
- Modifications

Requirements

volume	numus/T2K year
56t	~150,000
100t	~268,000
56t+1m = 185t	~496,000
100t+1m = 275t	~736,000

After all νe appearance cuts, ~ 300 νμ/yr/100t remain to study the BG

10yrs stats in 100t+1m --> 7.4 million ν_μ with R<325cm

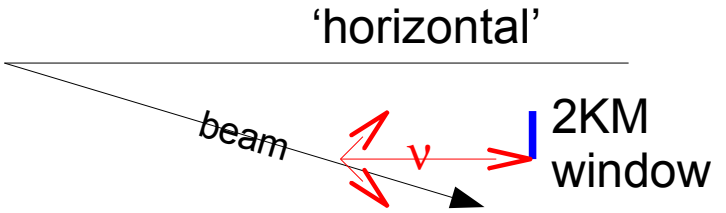
At kashiwa (icrcals* cluster), with 100 CPUs, it takes ~12h to generate 250,000 νμ events & 48h to reconstruct them ---> ~75 DAYS !

Of course the analysis will be done in parallel

We also need 500,000 νe (longer to process because of EM showers)

Beam simulation code

Beam Simulator : JNUBEAM



- Produces ν 4-vectors with weights
- GEANT3 simulation :

target -> horns -> decay vol. ; simulates parent hadrons, until they decay

- At SK : point like detector. Weigt = probability to hit SK ; all ν have the same direction
- At 2KM & nearer dets : hit & miss method

(if ν flies through detector opening save it else try again ; stop if more than 1000 attempts)

takes into account angle spread of ν beam + correlations between vertex and energy

Hit & miss somewhat inefficient @ 2KM : low stats in the ntuples from the web page

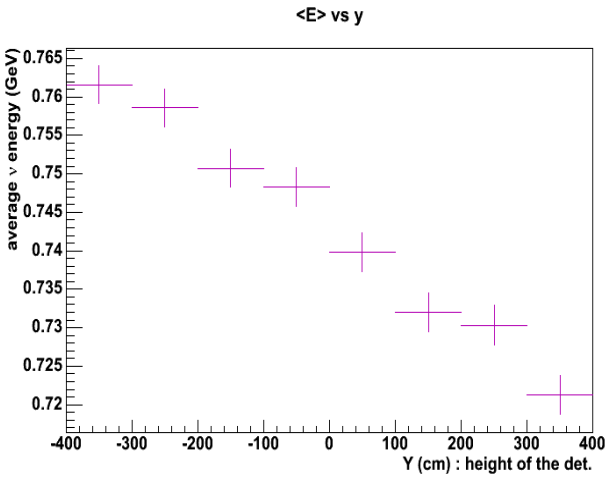
Vector generator

Hayato-san's code

- Different treatment for SK & 2KM to match JNUBEAM
- At SK : -make flux spectra for all ν flavours
  -sample ν energy from these distributions
  -shoot random vertex in SK, use single direction from JNUBEAM
  -call NEUT to generate the ν interaction
- At 2KM : - load all 2KM JNUBEAM ntuples into memory
  - loop over all events
  - if (random < (weight*xsec)/(max (weight*xsec))) keep the event
  - use the vertex & direction from JNUBEAM
  -call NEUT to generate the ν interaction

Vector generation (cont'd)

- SK method is faster but treats detector as a point (perfectly OK for SK)
- 2KM method :
  - preserves energy/vertex correlation



  - slow ?
  - oversamples the ntuples from JNUBEAM : need more ν 4-vectors than final number of ν interactions --> not possible for this MC production

-> Develop a method @ 2KM that is similar to SK & does not oversample

Modified method at 2KM

Want to keep vertex/energy correlation (~5% effect)

- --> bin the detector surface 800cmx800cm in 4x4 bins

ALGORITHM :

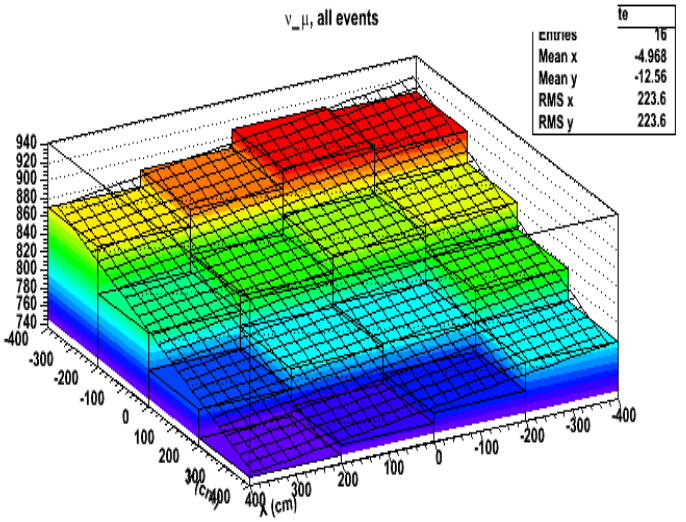
- shoot (x,y) vertex coordinate according to smoothed PDF(x,y)
- find the surface bin & shoot the neutrino energy according to the event rate spectrum in this bin
- use the average beam direction as ν direction (no angular spread, not a problem since angular spread smaller than angular resolution)

GET THE EVENT RATES :

- build one event rate (= flux * total xsec) histogram in each bin using the official JNUBEAM ntuples (flux) and NEUT 4.5.1 from Hayato-san [developed a stand alone program for this]

GET THE (X,Y) PDF :

- Fill each bin with integral of previously computed event rates distributions
- Fit with a plane : A+B*X+C*Y
- Use the plane as the PDF...



Linear approximation OK at this stage :

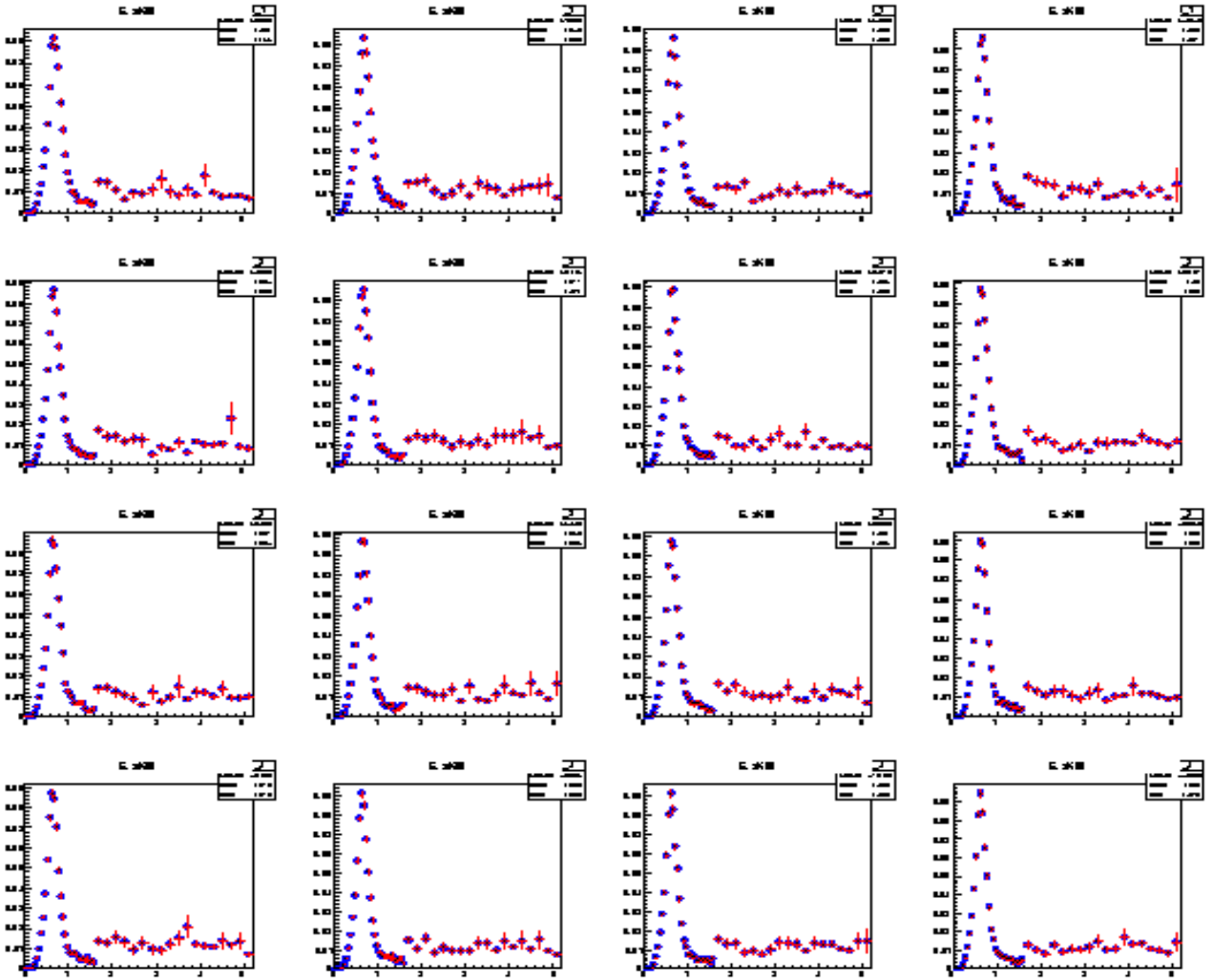
Equation = (-0.000616301)*x*0.01+(-0.00155077)*y*0.01+(0.0624416)

χ^2 min = 5.8 with ndof= 13

MODIFY the vector generation program to use this algorithm

REMARKS : all the event rates distributions are limited in precision by the number of events generated in JNUBEAM --> the statistical error on the output is larger than the simple 1/sqrt(N)... can't be better than the error on the input !

TESTS

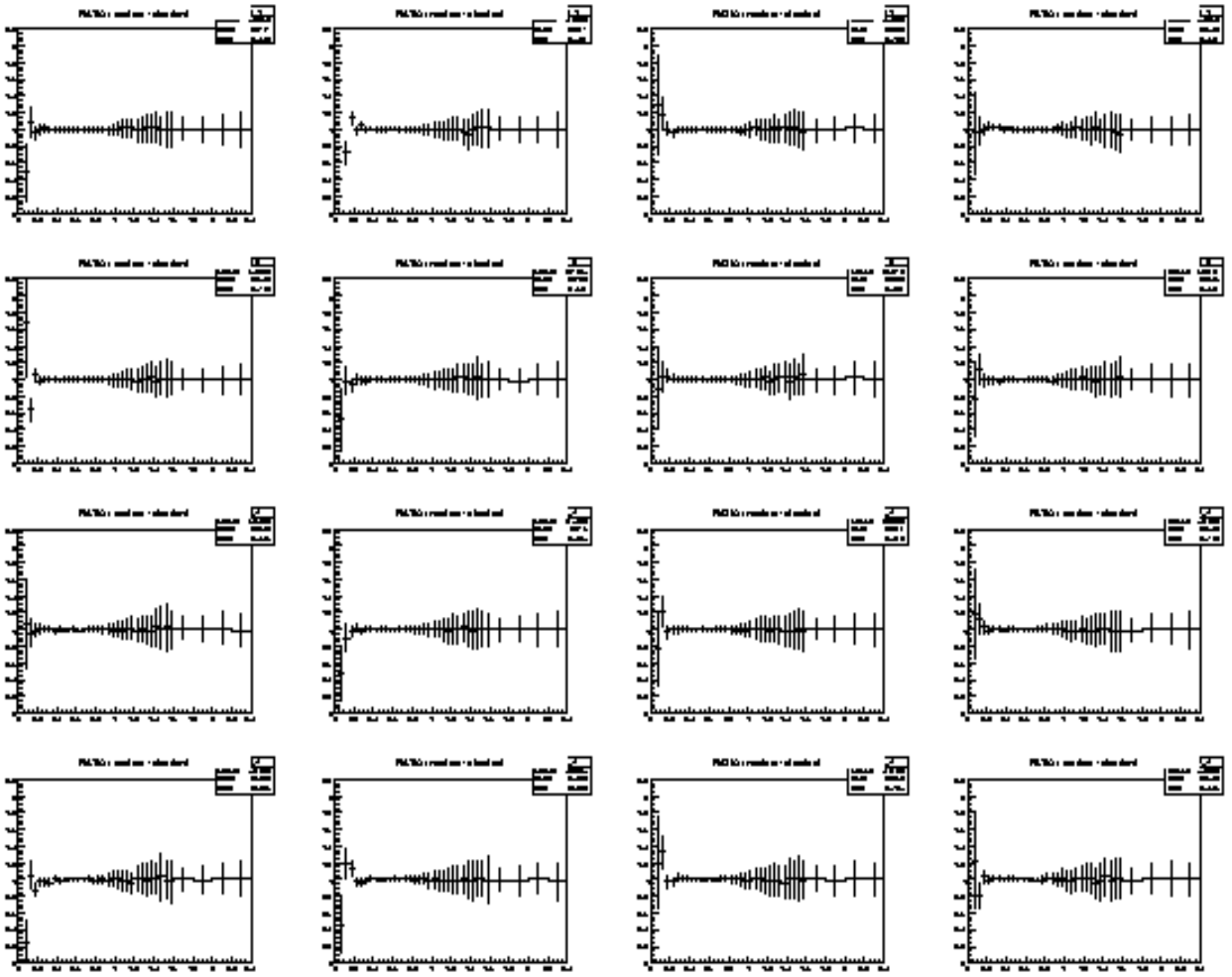


Energy distirbutions per (X,Y) bins

red : official vector ntuples

blue : generated with this method (10^7 ν)

Ratios of the previous histograms



Mostly flat except in first low energy bins
My interpretation : these bins have very few events --> statistical fluctuations

The new program is operational !

Conclusion

- Wrote a new algorithm for vector generation suitable for near detectors
- Approximate treatment of the energy/vertex correlation
- Runs very fast ; several million events not a problem (only disk space...)
- The output needs to be processed with 'nfsi' program to correctly simulate final state interactions