GooFit on GPU versus *RooFit* on multiple CPU via *Proof-Lite* : a performance comparison

L.Cristella, A.Di Florio, A.Pompili

University of Bari & INFN-Bari

CMS Software/Computing R&D Meeting
May 18th 2015

Outline

- **PART I**
  - GooFit Framework
  - Hardware Configuration
  - Physics case chosen for GooFit application & performances comparison with RooFit
- **PART II**
  - GooFit/RooFit MC Toys setup : fitting & validation
- **PART III**
  - Performances (GooFit/GPU vs RooFit/ProofLite CPU) : results & discussion

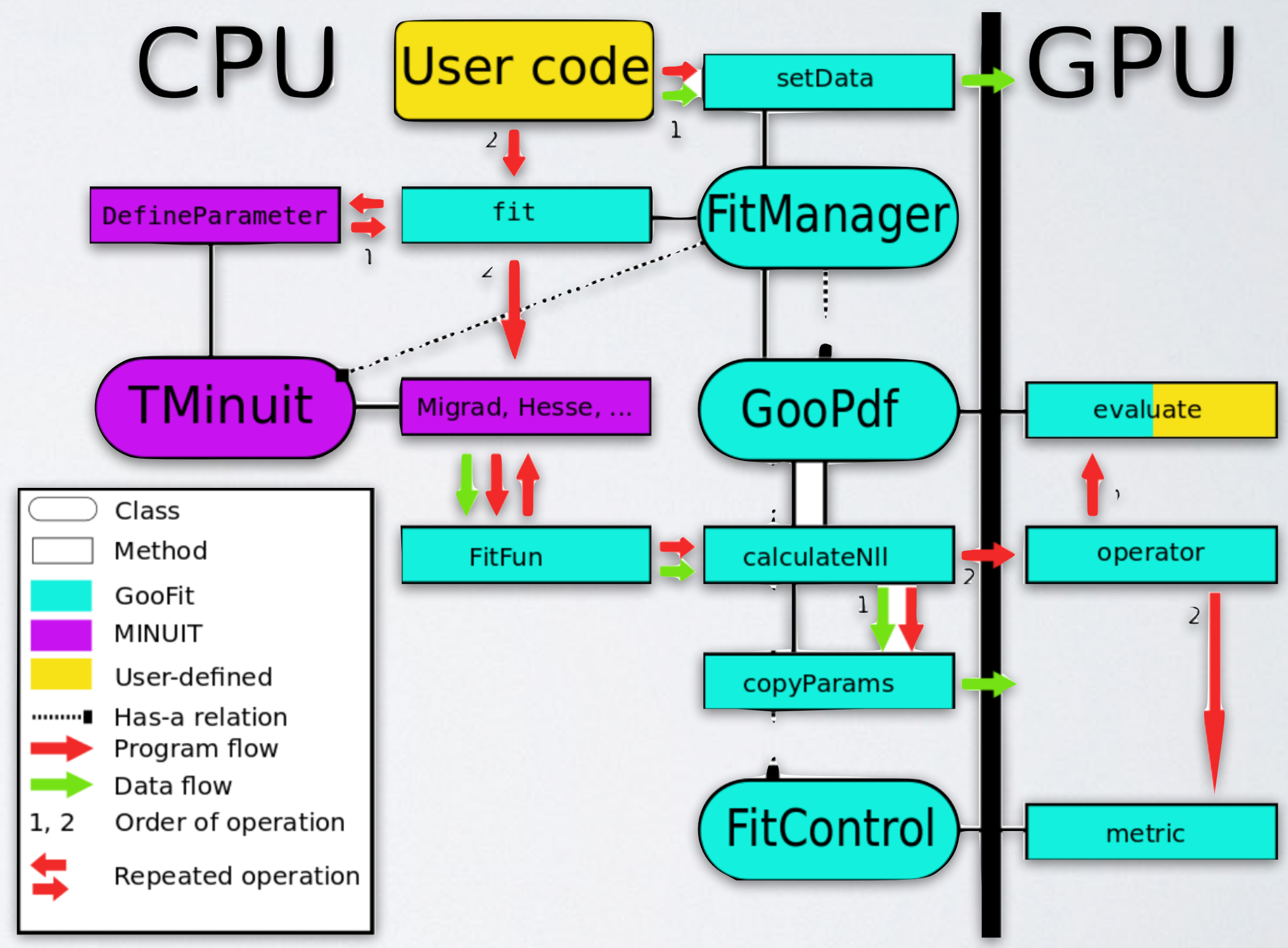
Part I - Introduction

GooFit Framework

- **GooFit is a large data processing application funded by NSF(*); it is an interface between MINUIT and a GPU which allows a *p.d.f.* to be evaluated in parallel.**

A GooFit macro has three main components:

- **The P.D.F. describing the physical process is defined in a *GooPdf* object containing the fit parameters as *Variable* objects**
- **The actual interface between MINUIT and the *GooPdf* is a *FitManager* object which:**
  - **assign the *DataSet* (containing one or more observables as *Variable* objects) to the *GooPdf***
  - **via *fit* method links <u>CPU-sided</u> MINUIT minimising algorithm (es. `MIGRAD`) to the <u>GPU-sided</u> likelihood (or chi-square) calculation (via *calculateNll* method)**
- **The object *FitControl* allows user to select which test statistic should be used (*es. unbinned/binned likelihood, chi-square,…)***



GooFit: A library for massively parallelising maximum-likelihood fits

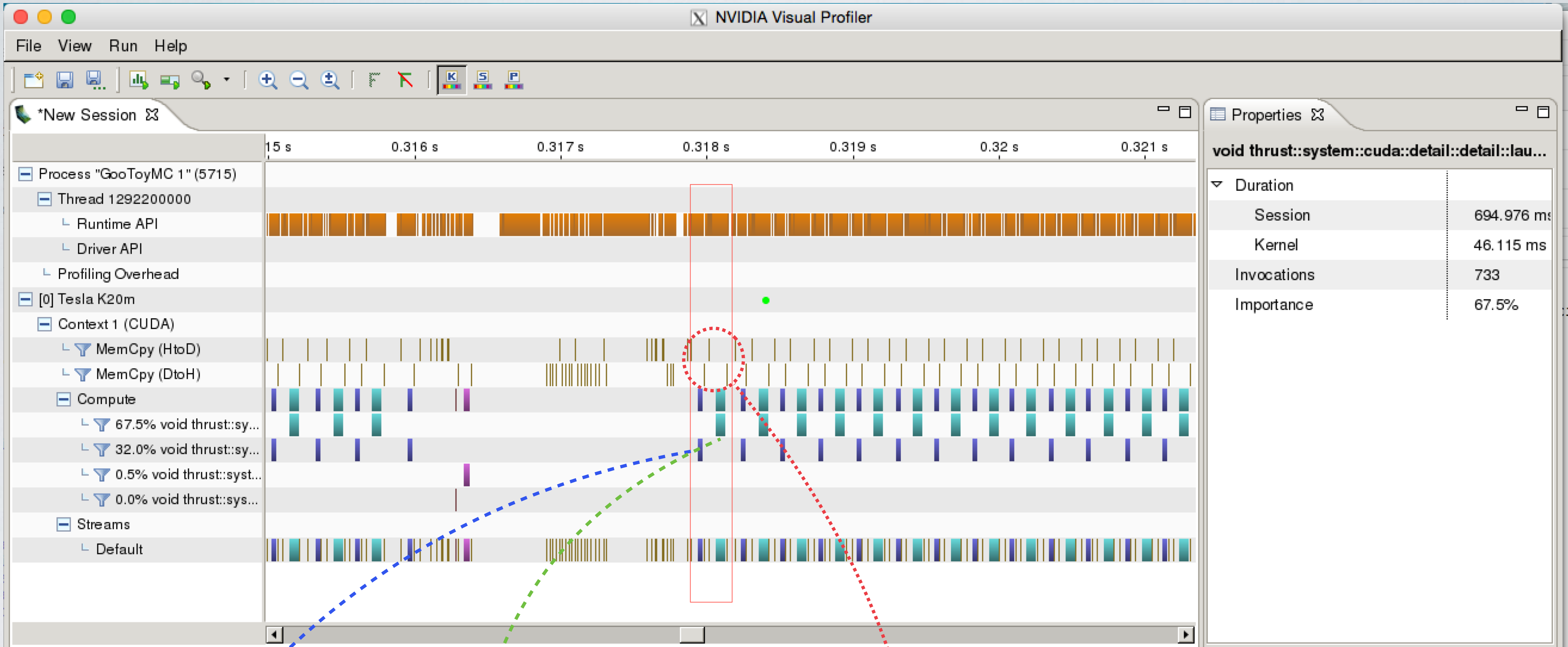
R. Andreassen et al., *J. Phys.: Conf. Ser.* **513** 052003, 2014

GPU : *p.d.f.* evaluation ⇄ **CPU :** Parameters tuning to minimise Neg-Log-Likelihood

(*) NSF-1414736 *Enabling High Energy Physics at the Information Frontier Using GPUs and Other Many/Multi-Core Architectures*

GooFit Profiling

- **Example of a snapshot of the profile of a GooFit process provided by Nvidia Visual Profiler :**



p.d.f. normalisation

Likelihood calculation

GPU : *p.d.f.* evaluation

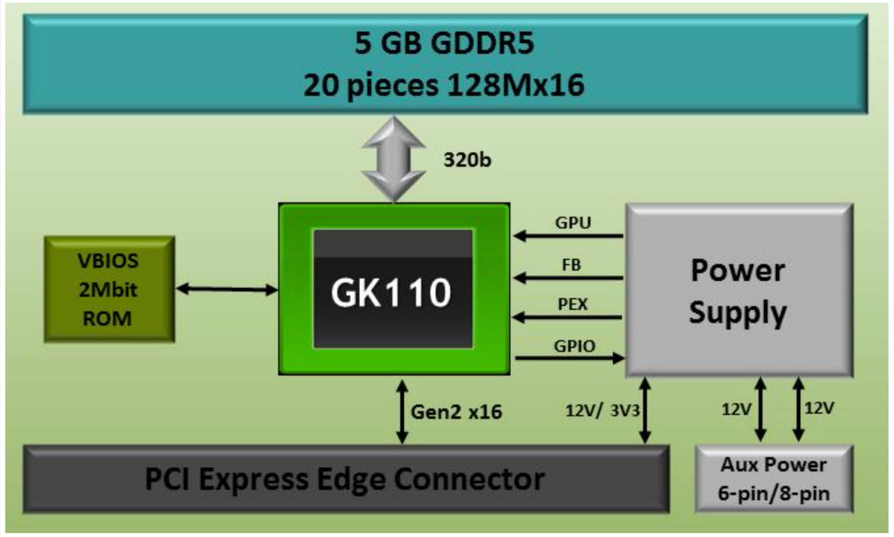
Fit parameters' flow between GPU & CPU

CPU : Parameters tuning to minimise Neg-Log-Likelihood

Hardware Configuration

- **In the Bari Tier2 we have a server equipped with 2GPUs Tesla K20 & 32 CPU cores**

  - **Tesla K20 block diagram :**



  - **Details :**

GPU	
Number of GPUs	2 x GK110
Number of CUDA cores	2 x 2,496
Memory size for GPU (GDDR5)	2 x 5 GB
Memory bandwidth for board	288 Gbytes/sec

CPU

- 32 cores : E5-2640 v2 @ 2.00GHz
- 64 GB RAM

Hyper-Q : Allows multiple CPU cores to simultaneously interact the CUDA cores on a single GPU.

- **Multi Process Server**

A single process may not exploit all the computing capability of the GPU.
MPS is a tool provided by nVidia that enables concurrent CUDA processes to be run at the same time on the same GPU by scheduling their access to the GPU resources (both RAM memory and CUDA cores)

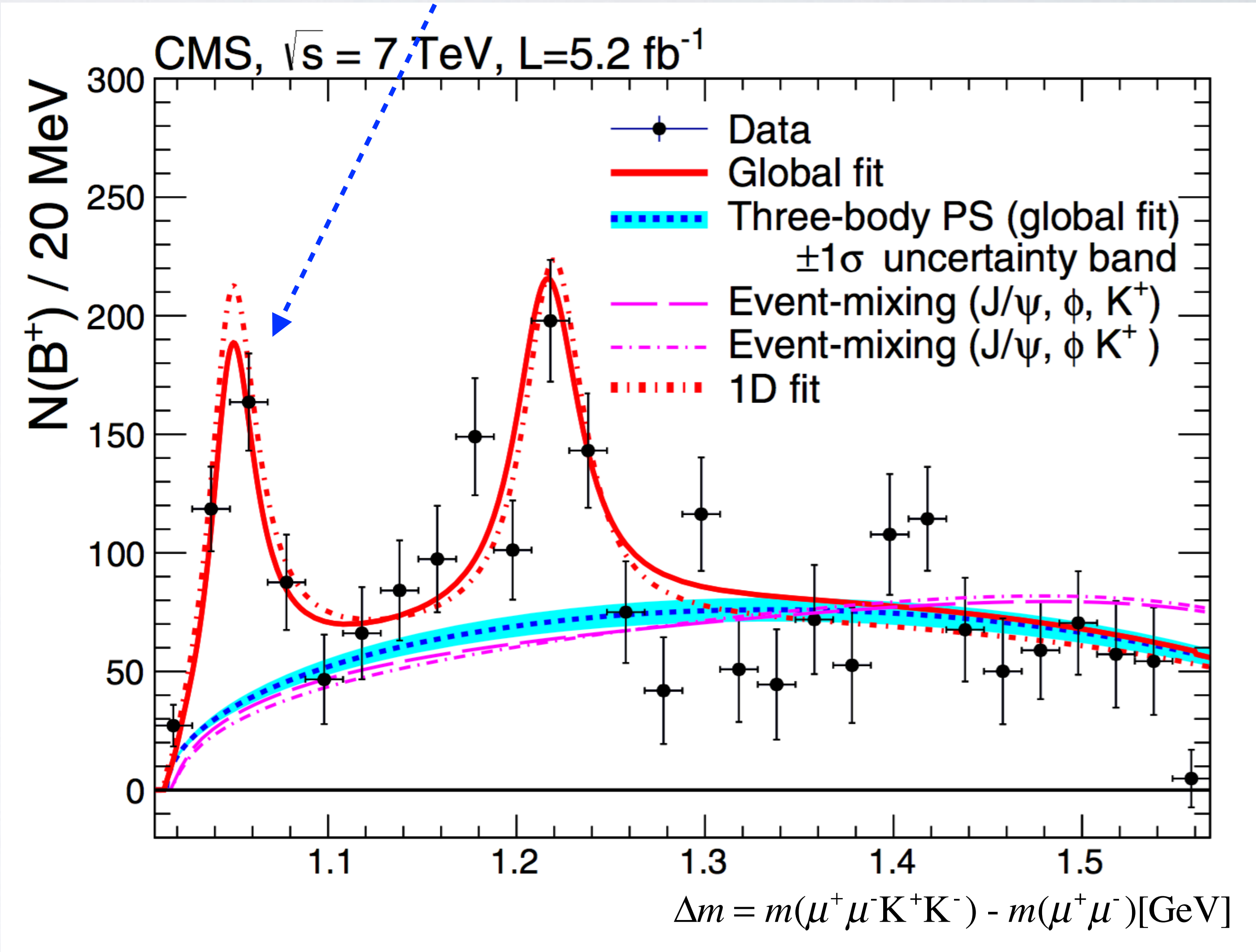
Physics case chosen for GooFit application - I

- **In CMS paper PLB 734(2014) 261, the significance for the structure in the $J/\psi\ \phi$ mass spectrum close to the kinematic threshold was evaluated also by means of toy MC simulations using RooFit:**

Peaking structure @ threshold with:

$$m = 4148.0 \pm 2.4(stat) \pm 6.3(syst)\ \text{MeV}$$

$$\Gamma = 28^{+15}_{-11}(stat) \pm 19(syst)\ \text{MeV}$$



To explore GooFit VS RooFit performances we exploit the same physics case

Physics case chosen for GooFit application - II

- To describe the adopted procedure let's refer to CMS AN-2011-415 (BPH-11-026)

1.6.3 Significance Evaluation: Toy MC

To evaluate the signal significance for the observed two structures a standart Toy MC simulation is also used. The following processes are followed for the significance measurement;

A MC simulation is used to estimate the probability that background fluctuations alone would give rise to signal as significant as that seen in the data for the structure at the $J/\psi\phi$ threshold. We generated 50.5 million ΔM spectra with 2340 events for each spectrum based on a three–body phase space shape, and then search for the most significant fluctuation in each spectrum in the mass range within 3σ of the CDF's parameter and with widths in the range of 10 MeV (half bin width) and 80 MeV (up to the middle of the two peaks). Then we obtain the $\Delta\chi^2$ distribution in the pure simulated background samples and compare it with the signal in the data. We found zero generated spectra with their largest fluctuation having a $\Delta\chi^2$ value to be equal or greater than the value obtained in the data. The resulting p-value, taken as the fraction of the generated spectra with a $\Delta\chi^2$ value greater than or equal to the value obtained in the data, is less than 2×10^{-8}, which corresponds to a significance of more than 5.0σ (Fig. 32).

- **Reminder**

In the asymptotic limit (i.e. as long as the normal sampling distribution is a valid approximation) the $\Delta\chi^2 = \chi^2_{(bkg)Fit} - \chi^2_{(bkg+sig)Fit}$ is the same as a likelihood ratio test statistic ($2\ln(\lambda) \simeq \Delta\chi^2$).
If some regularity conditions are satisfied, Wilks' theorem is valid and $\Delta\chi^2$ behaves as $\chi^2_{\Delta(\#d.o.f.)}$ namely a χ^2 function with a # of d.o.f. equal to the difference between the two fits/hypothesis.
Unfortunately, commonly encountered situations violate this requirements (more in the backup).

Part II - Toy MC process

Toy MC Setup

- **Toy MC Fitting Cycle (for each generated event):**

1) Generation of fluctuated background distribution (*phase-space* model) via *Hit or Miss* procedure (# of entries in this distribution = # of entries in the data distribution = 2342)

2) Null Hypothesis NLL(*) fit is performed with the *phase-space* model only

3) Alternative Hypothesis NLL(*) fit is performed with the *phase-space* + *Voigtian* p.d.f.(**) (non-relativistic BW convoluted with a Gaussian resolution function with width fixed @ 2MeV)

- **the fit is performed 8 times trying 4 width values and 2 mass peak values as starting points in order to "scan" the signal region of interest in the data. The signal yield is constrained to be >0.**
- **for each of these fits : the $\Delta\chi^2$ is calculated with respect to the Null Hypothesis fit**

4) Best $\Delta\chi^2$ among the 8 *alternative hypothesis* fit is chosen

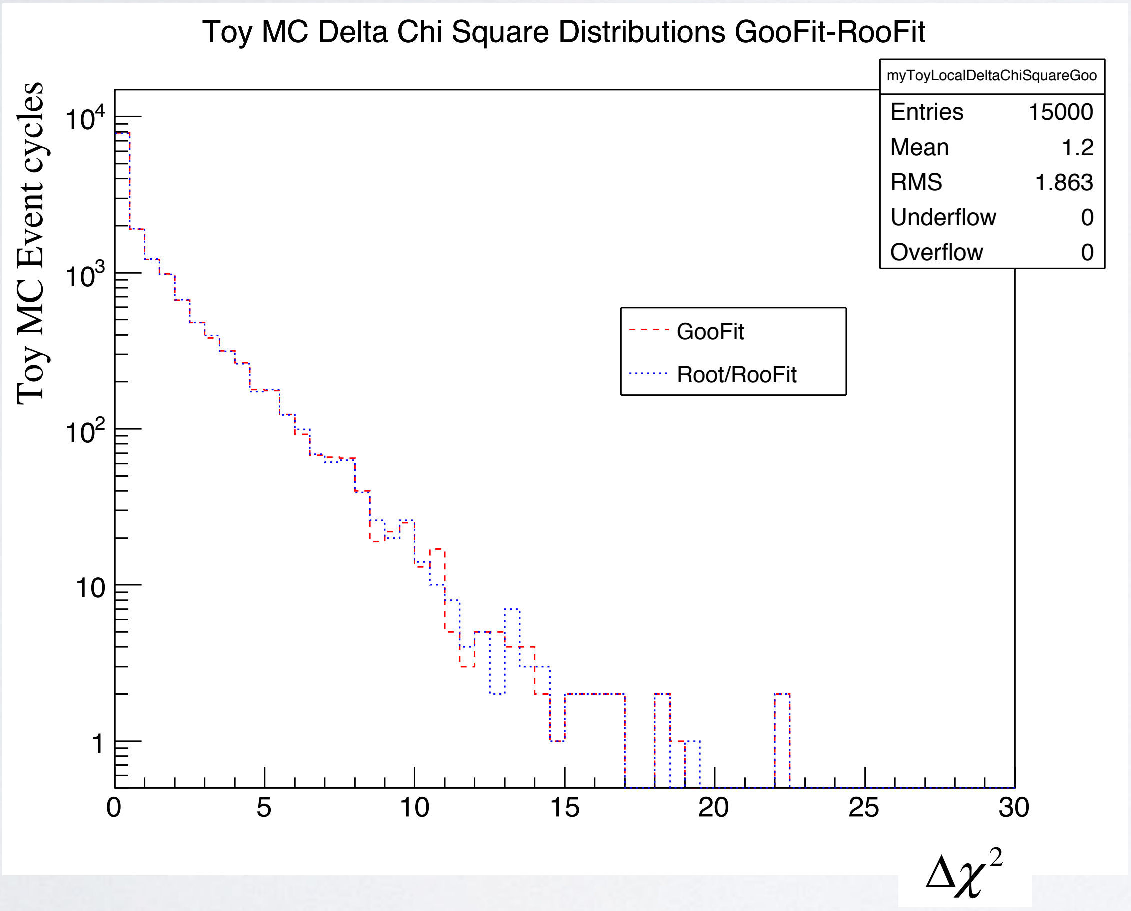
(*) Binned not-extended (NegLog)Likelihood fit [`MIGRAD` only] ;
for each bin the *p.d.f. value estimated by* ROOT *integration over the bin*
(time consuming but needed : steep signal w.r.t. bin size)

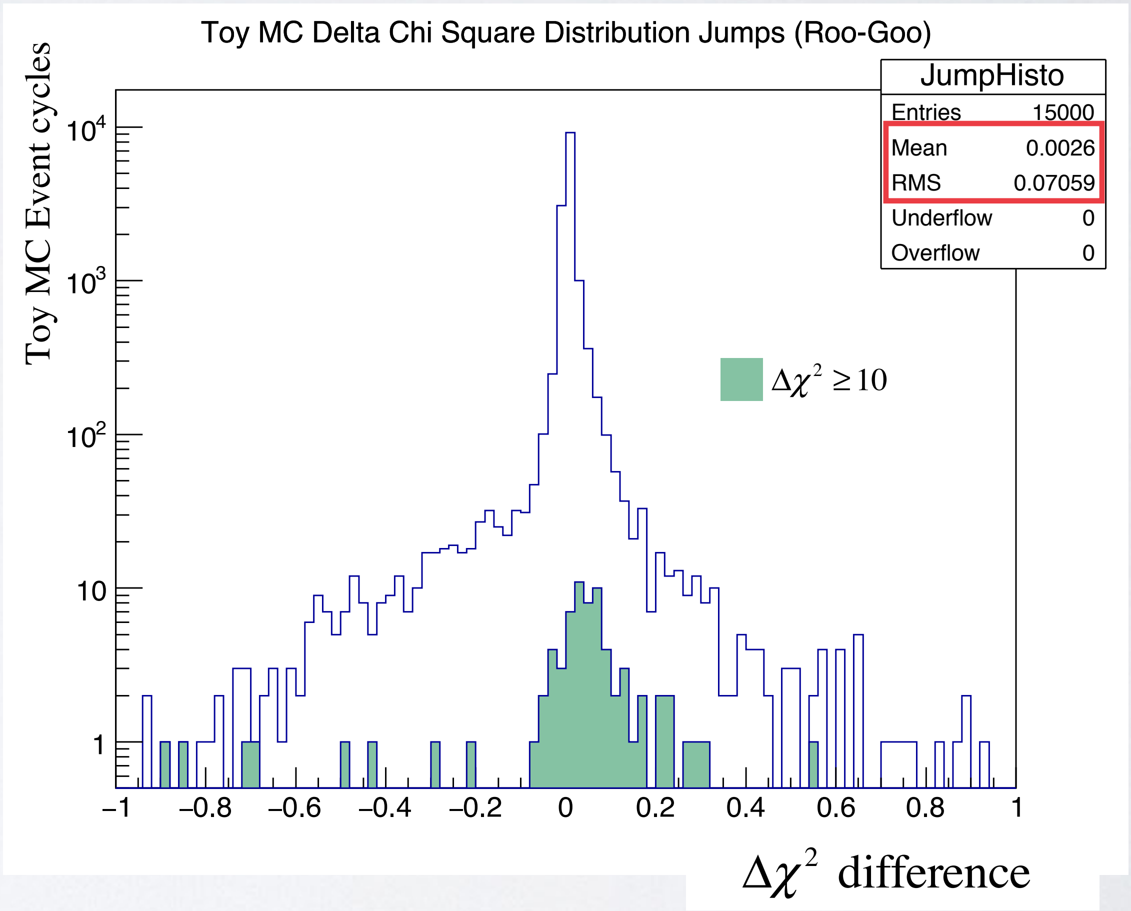
(**) the Voigtian p.d.f. is *truncated* to correctly account for the kinematic threshold

- **A $\Delta\chi^2$ distribution (over the events) is obtained (see next slides)**

GooFit Validation - I

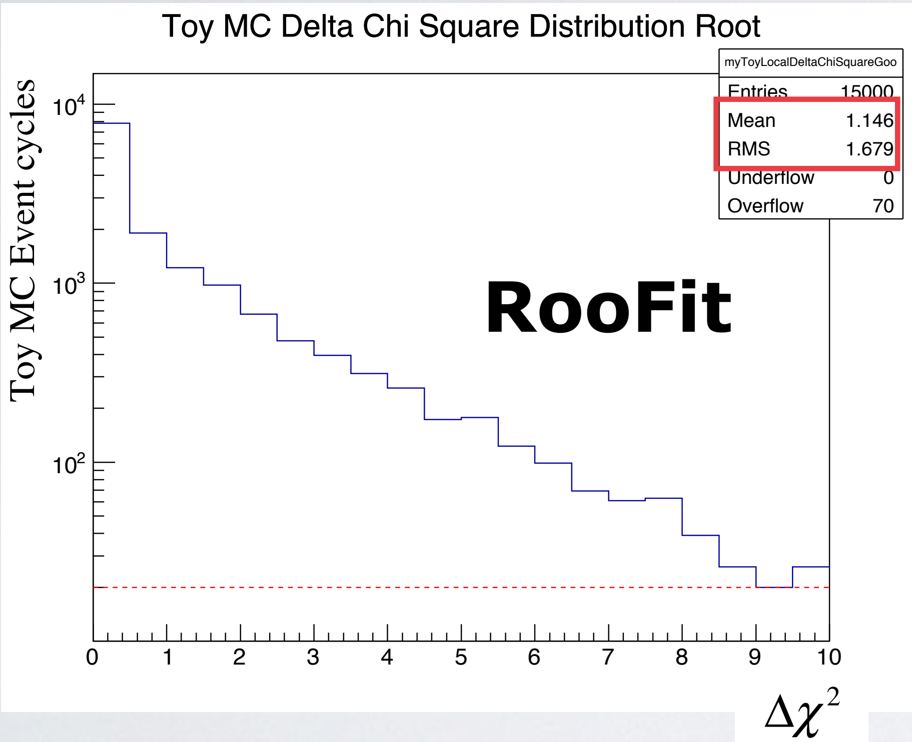
- **Validation of our CUDA/GooFit macro :**
  - **check that it provides on GPU a result equivalent to the one obtained by RooFit on CPU**
  - **first generate 15k of MC Toy distributions**
  - **feed with this common distribution both jobs**
  - $\Delta\chi^2$ **distribution comparison :**

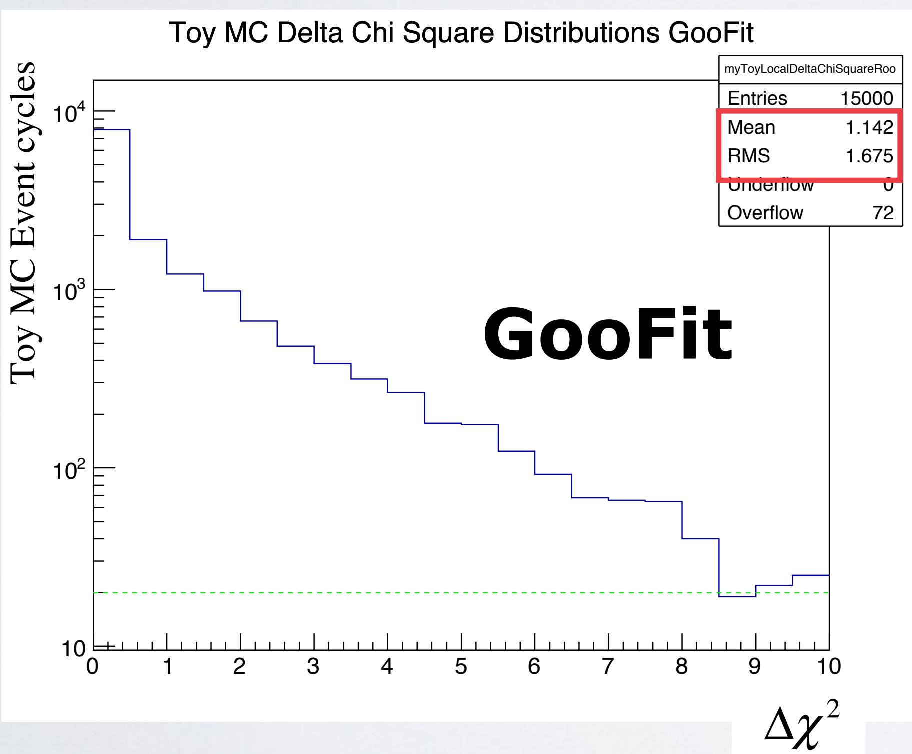


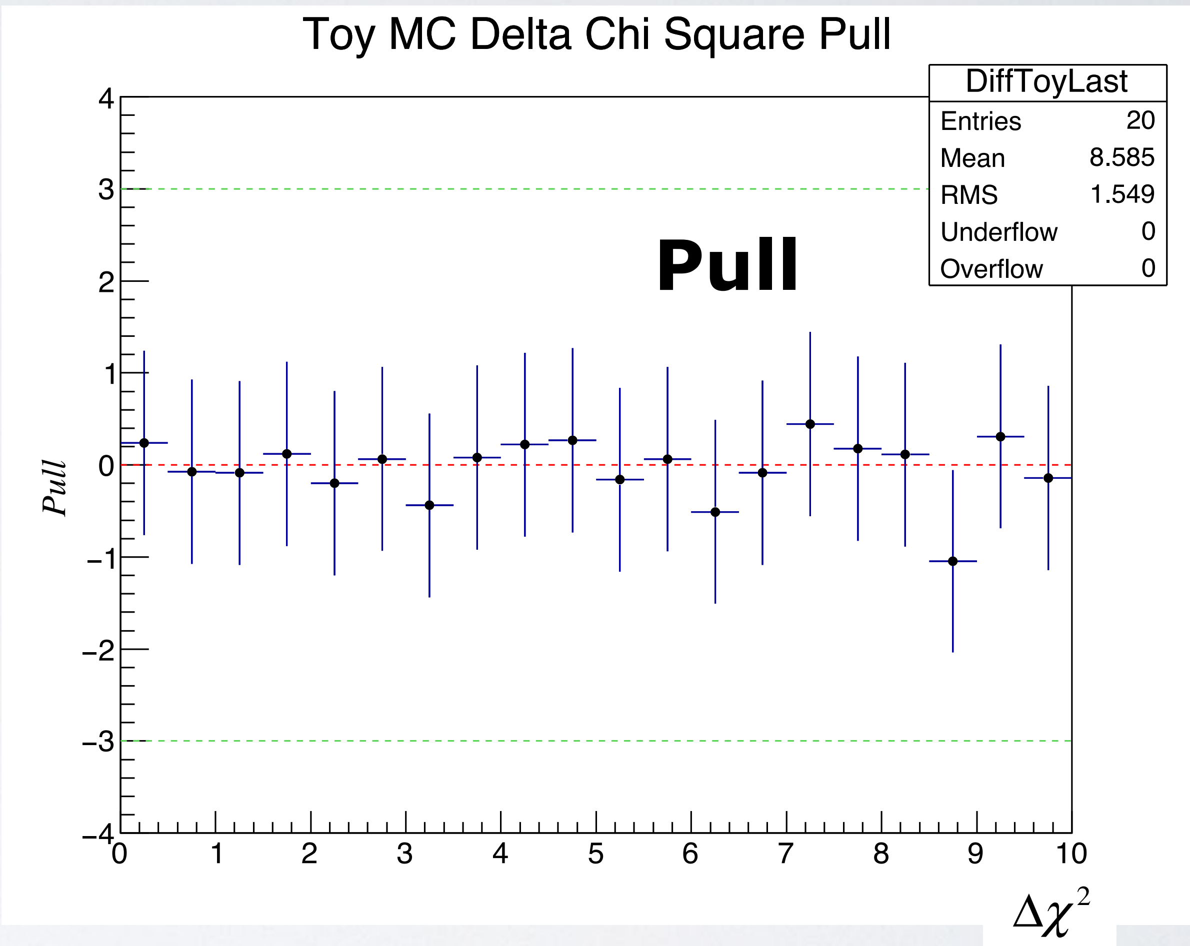


Very similar distributions; have a look at their compatibility as shown in next slide

GooFit Validation - II





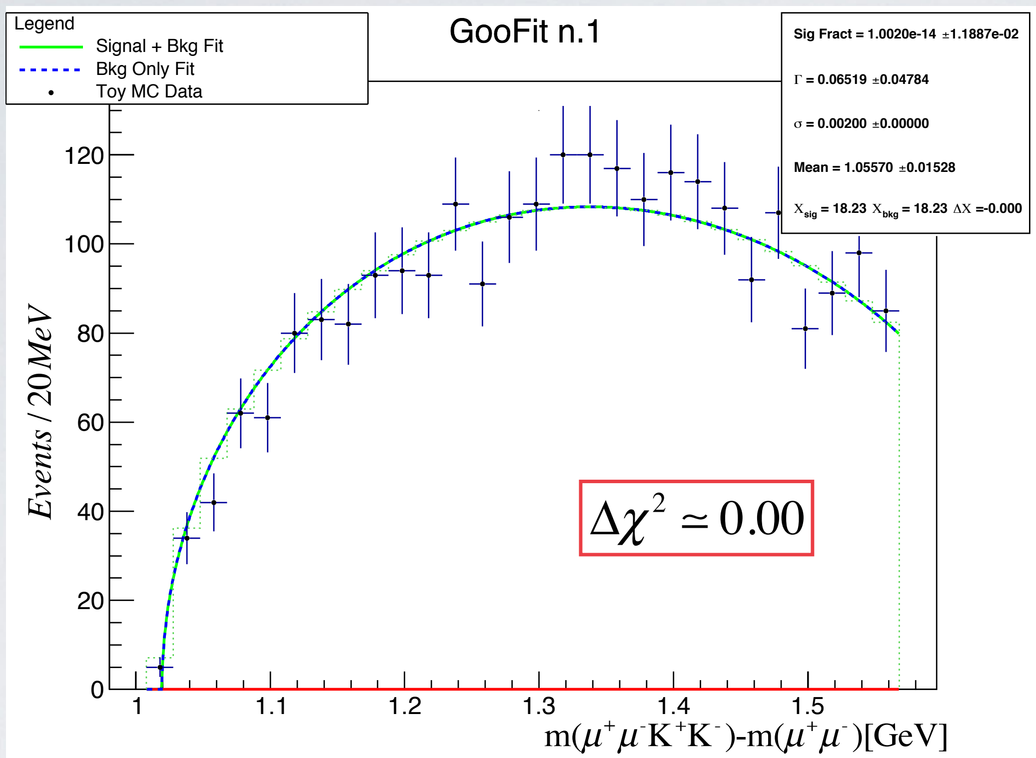


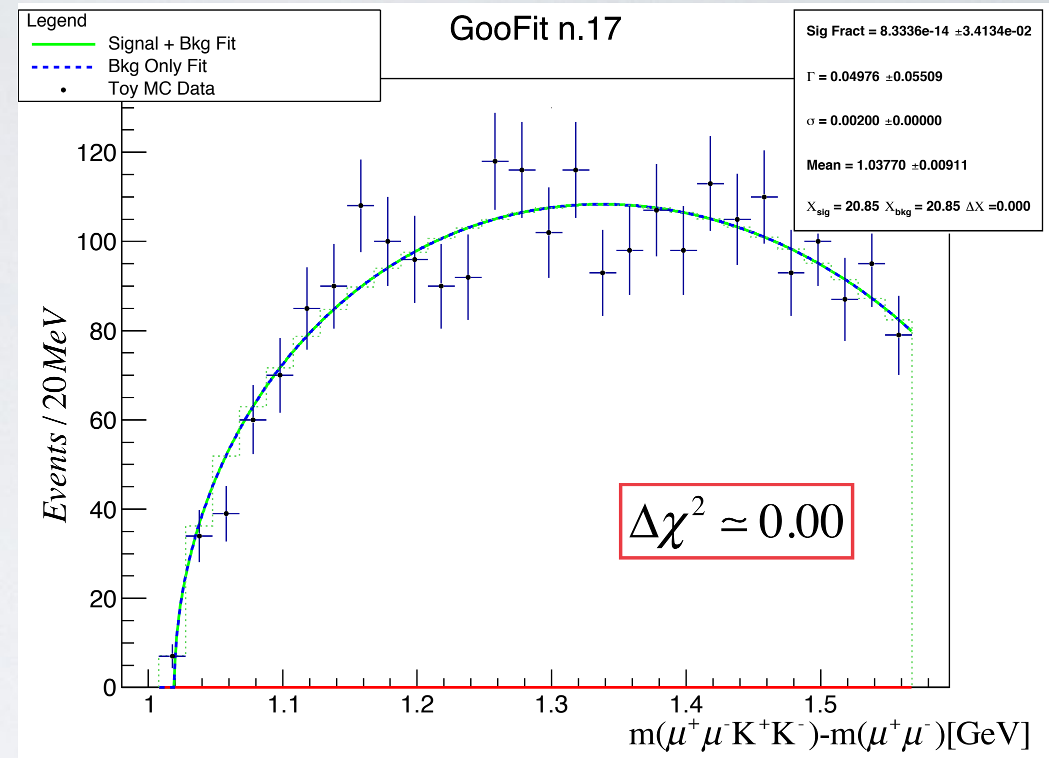
Have a look at some single fit comparisons as shown in next slides

GooFit Validation - III

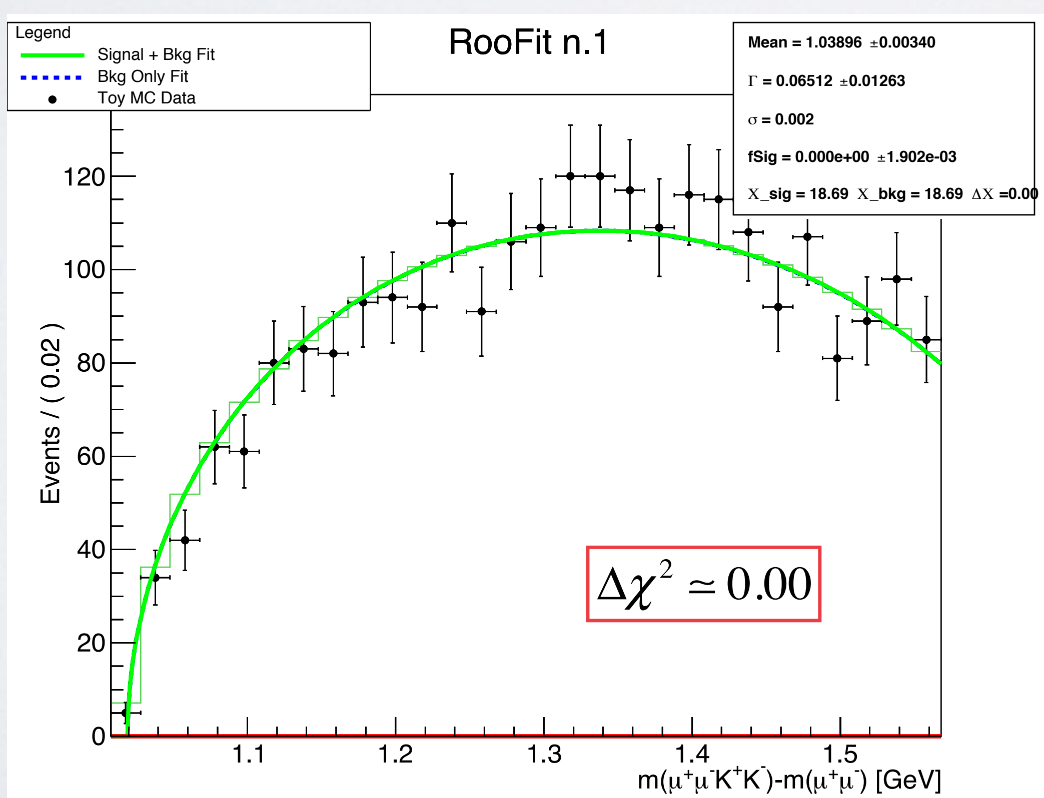
Examples of no fluctuation in signal region

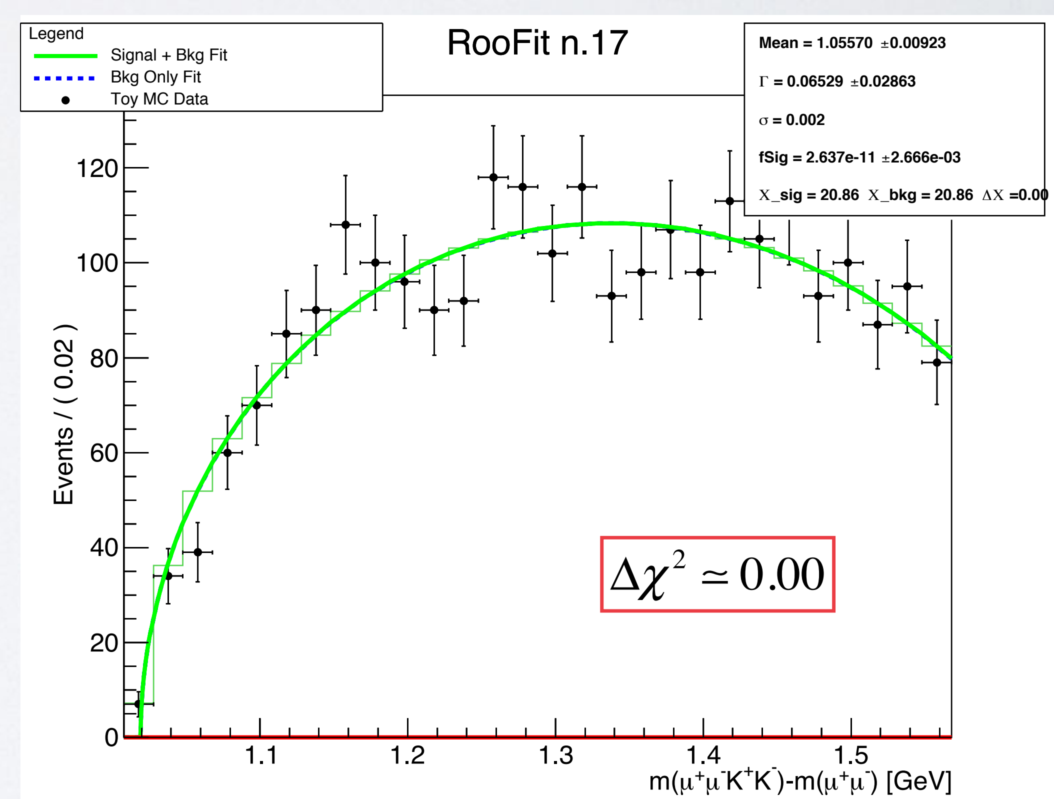
GooFit





RooFit

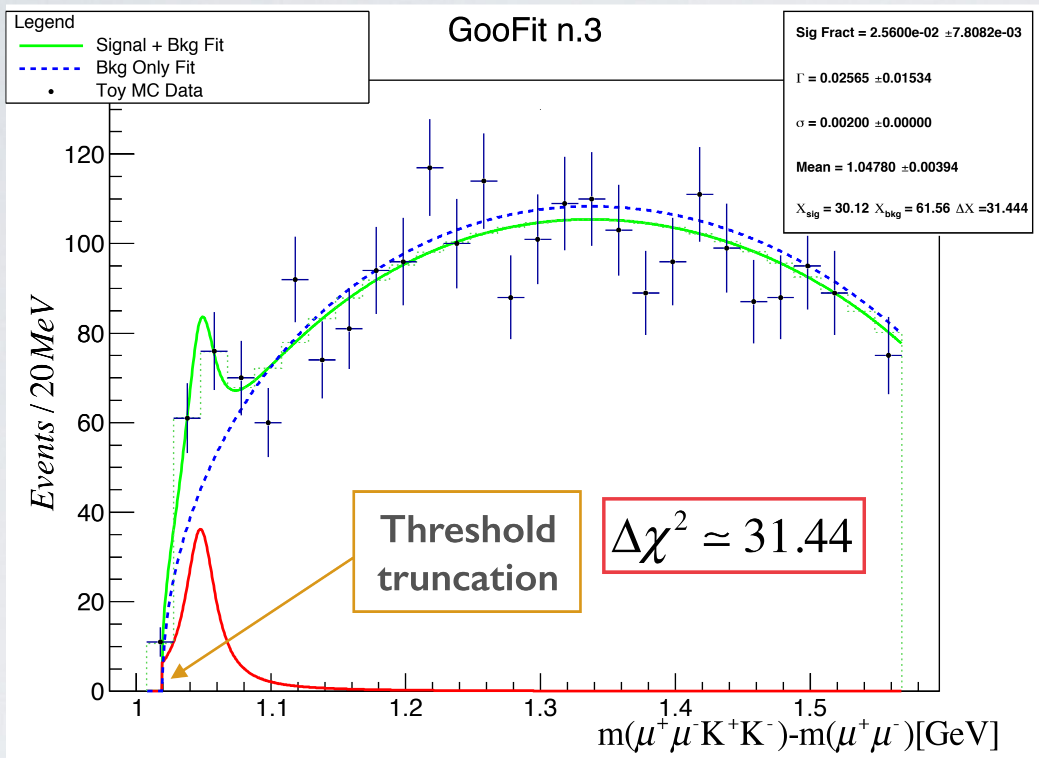


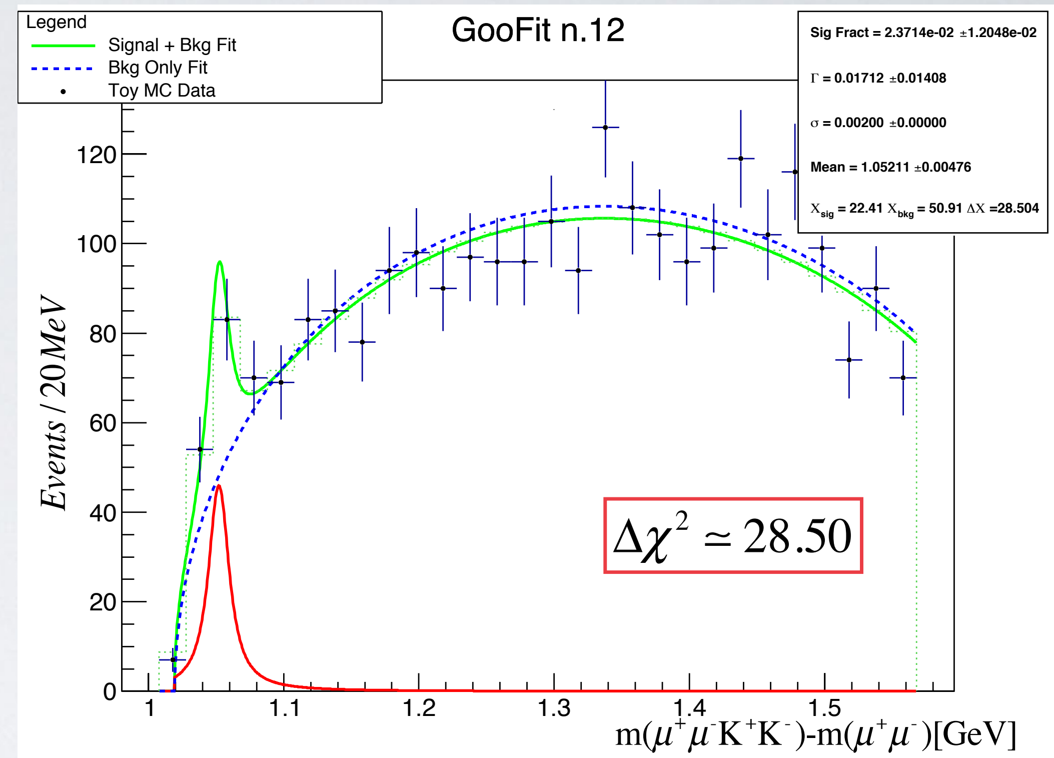


GooFit Validation - IV

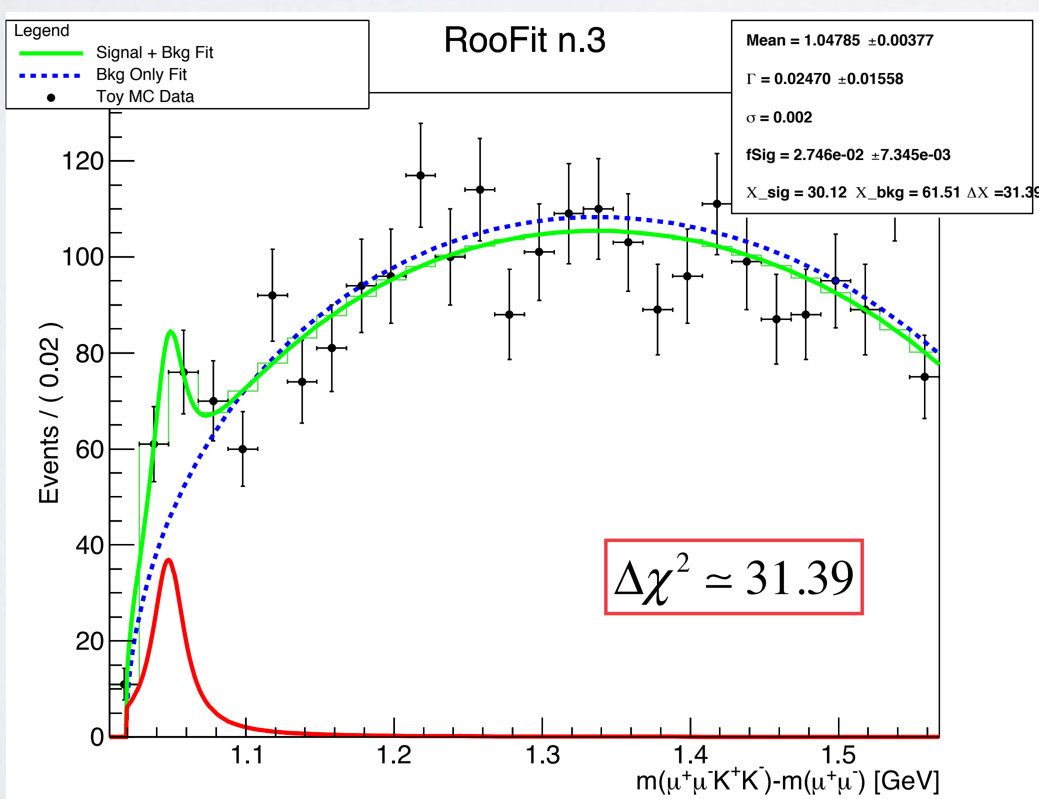
Examples of evidence of signal fluctuations

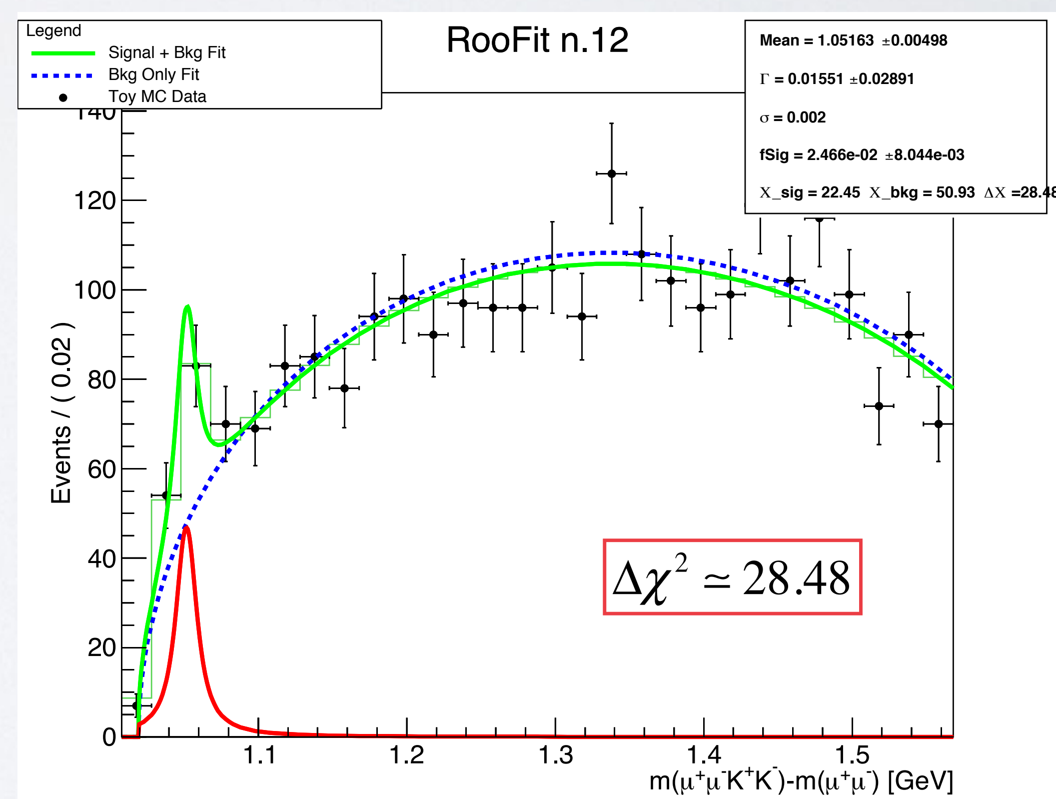
GooFit





RooFit

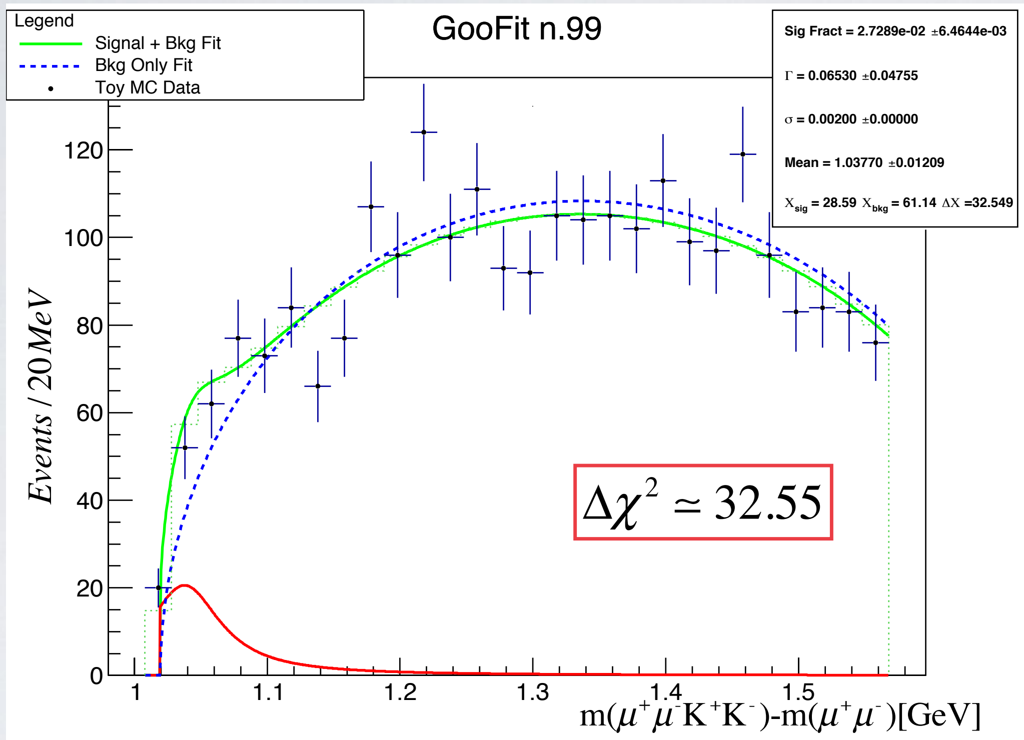


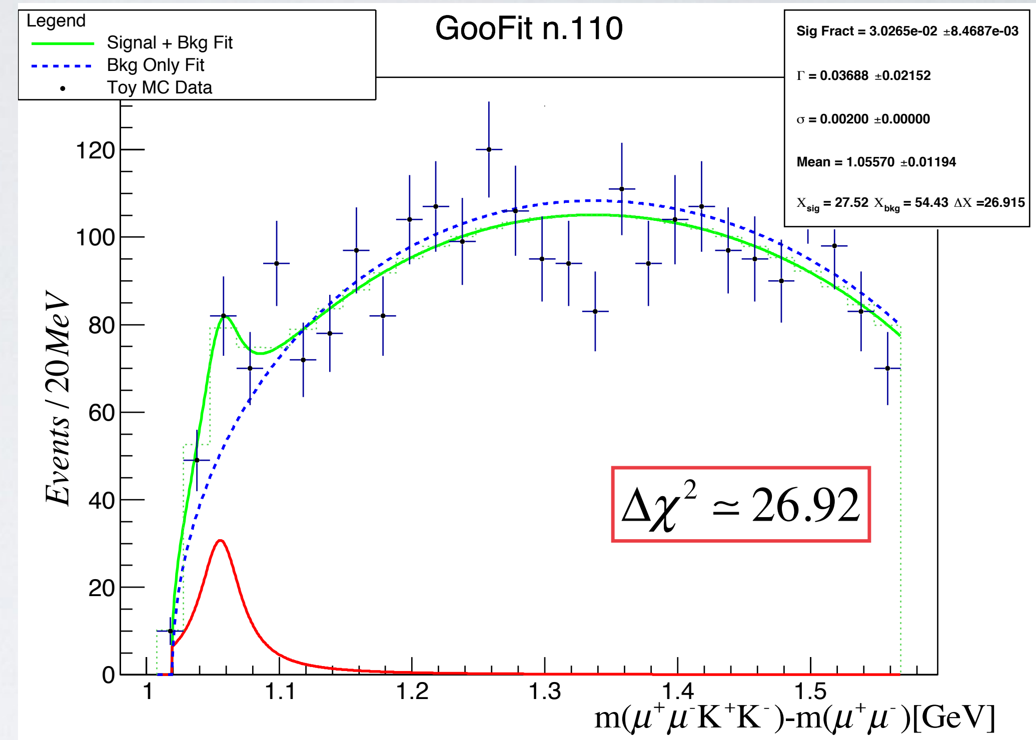


GooFit Validation - V

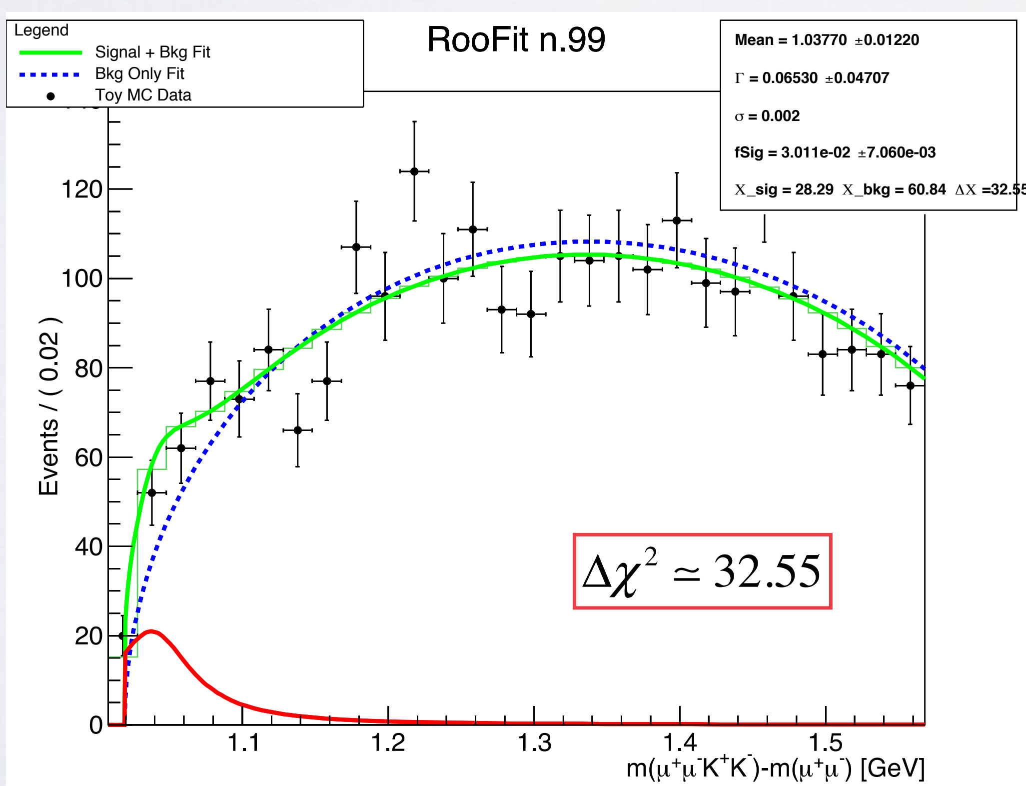
Examples of evidence of signal fluctuations

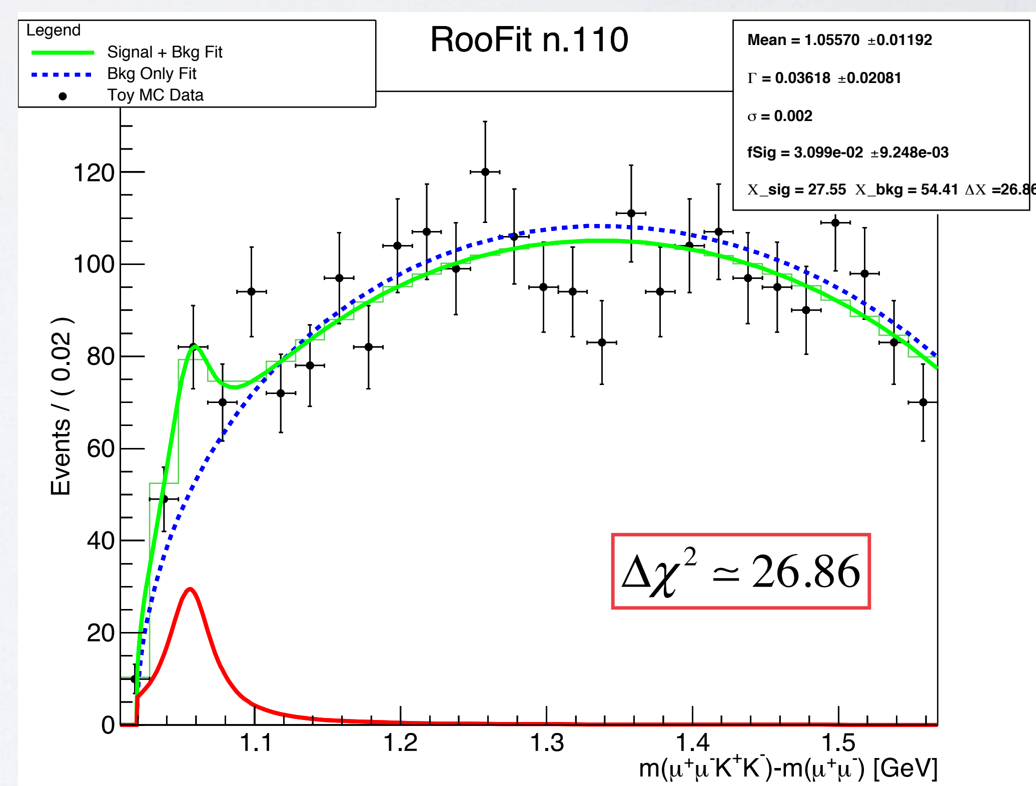
GooFit





RooFit





Part III - Perfomances

GooFit Performances - I

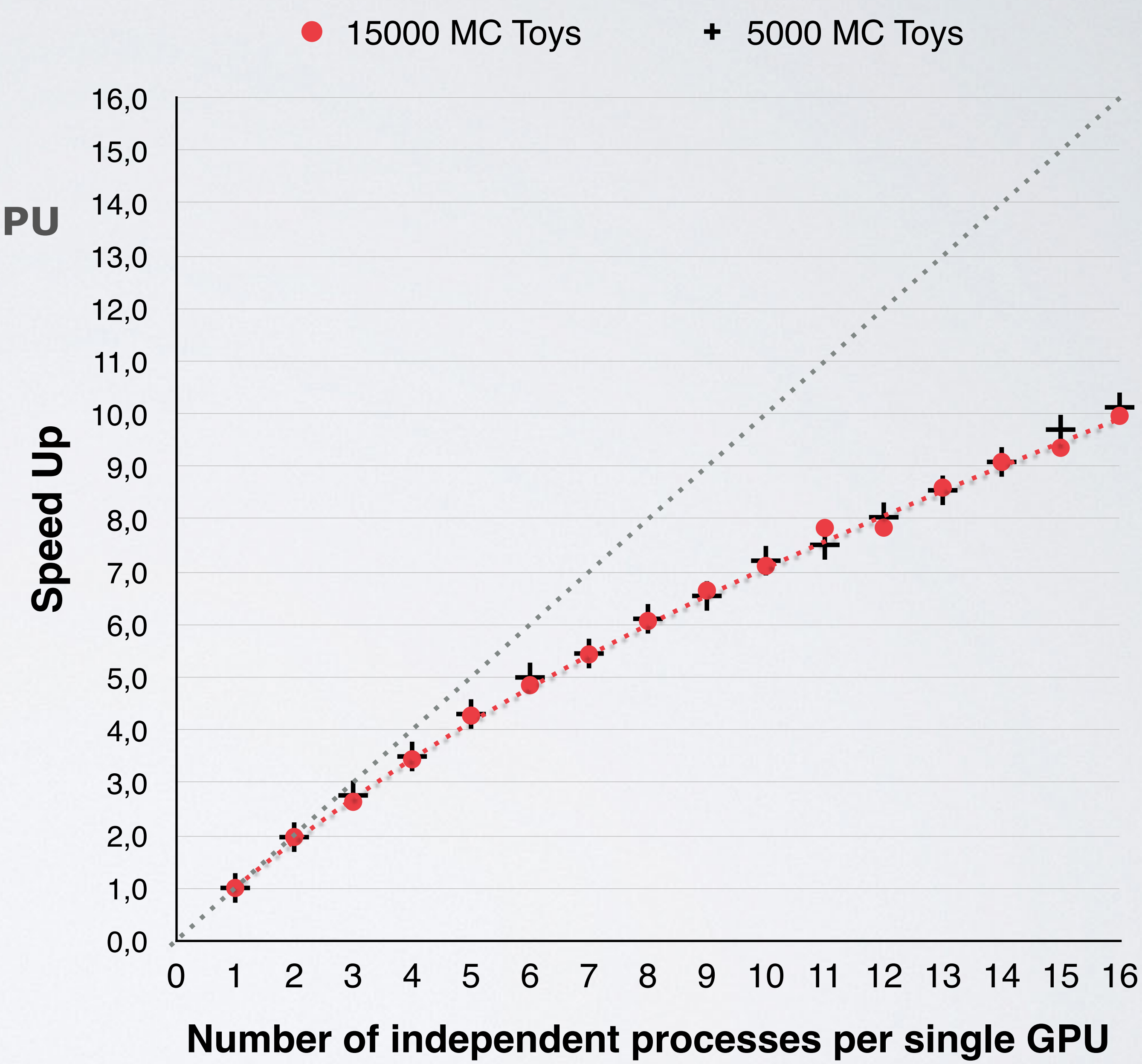
- **Speed Up as a function of number of independent processes per single GPU**

- **5K or 15K MC Toys per process**
- **Each process uses :**
  1 exclusively assigned CPU & 1 shared GPU
- **Up to n_{max}=16 concurrent processes (n_{max}=16 fixed by nVidia-MPS)**
- **Speed Up definition**

T_1 single process time

T_n n concurrent processes time

$$S_n = \frac{T_1}{(T_n / n)}, \qquad 1 \le n \le n_{max}$$

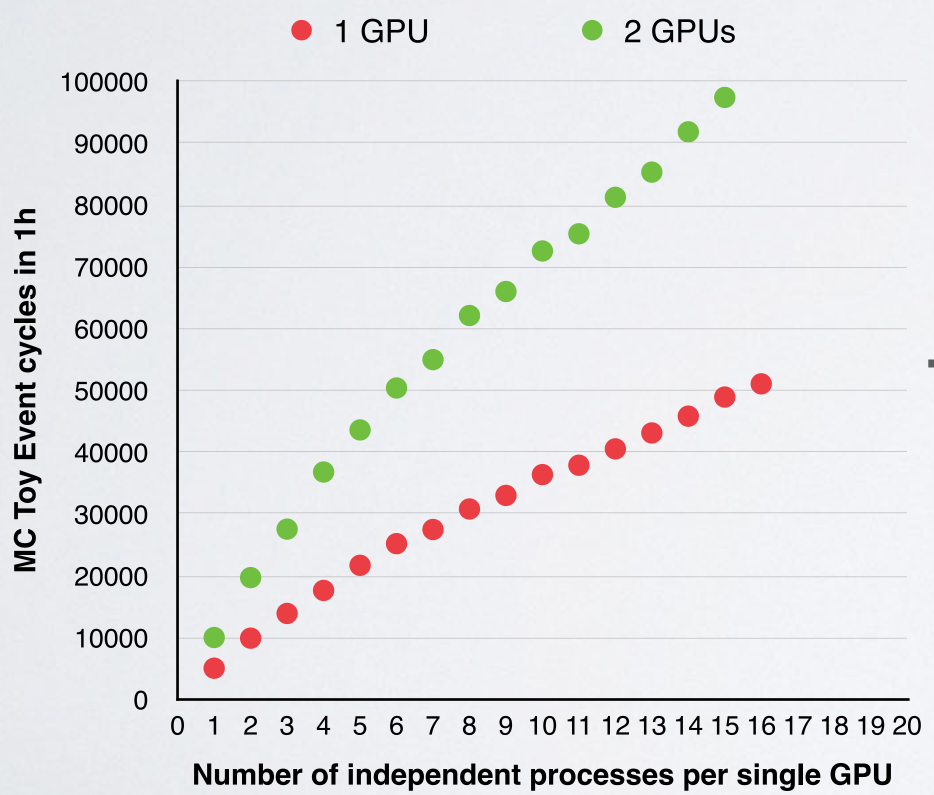


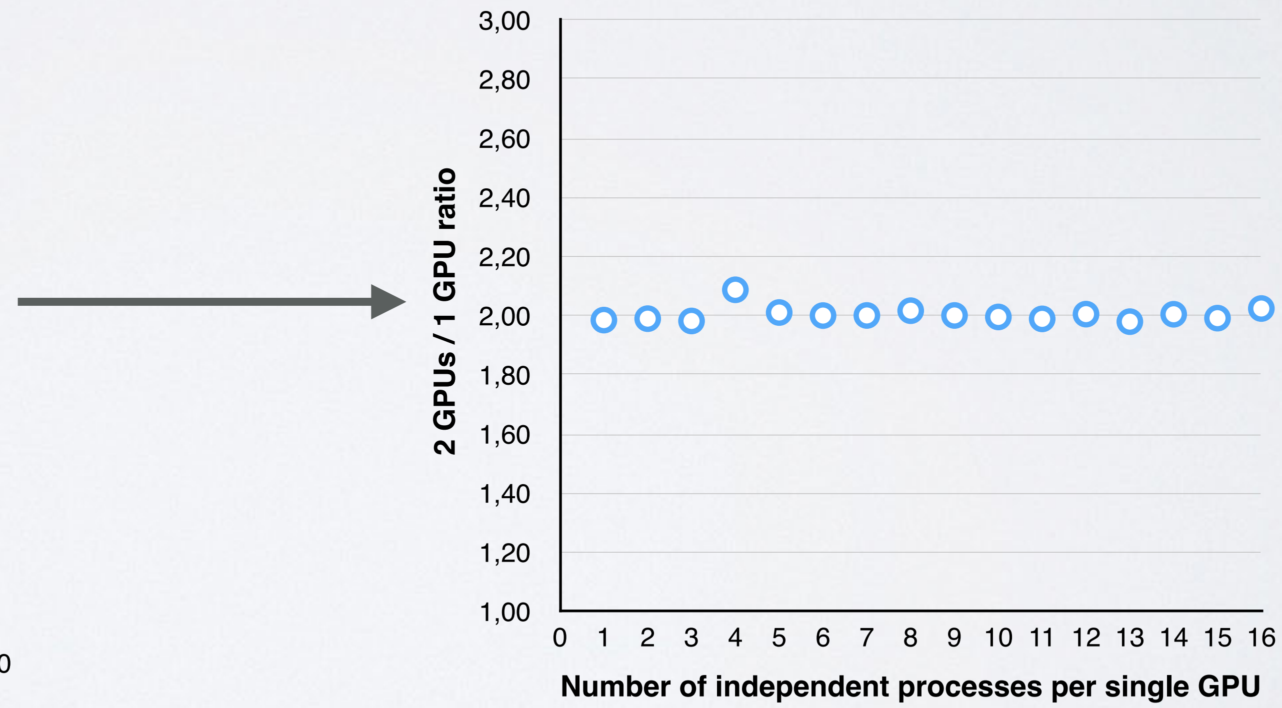
- **Comments :**
  1) **perfect scaling with # of process**
  2) **Speed Up shows a saturation behaviour.**
     This is expected : a single process works @ ~70% of the GPU computing capability and uses exclusively an assigned (dedicated) CPU

     Getting such Speed Up values means that MPS performs a sort of smart load balancing

GooFit Performances - II

- **Scaling behaviour while running on both GPUs available w.r.t. on single GPU**
  - **Two identical GPUs are available on our server**
  - **15K MC Toys per process**
  - **Each process uses : 1 exclusively assigned CPU & 1 shared GPU**
  - **Up to n_{max}=16 concurrent processes (n_{max}=16 fixed by nVidia-MPS)**
  - **First (second) group of n processes assigned to 1st GPU (2nd GPU)**





- **Comment : rather perfect scaling**

GooFit vs Proof-Lite performances - I

- **Since we have 32 CPUs on our server, we aim to compare the performances of GooFit on 2 GPUs against RooFit on 30 CPUs**

- **In order to run efficiently 32 RooFit process in parallel we use Proof-Lite[*](1 Master & up to 32 Slaves/Workers)**

- **We calculated Proof-Lite performance on this server**

Proof-Lite Speed Up

- **Speed Up definition**

T_1 single process time

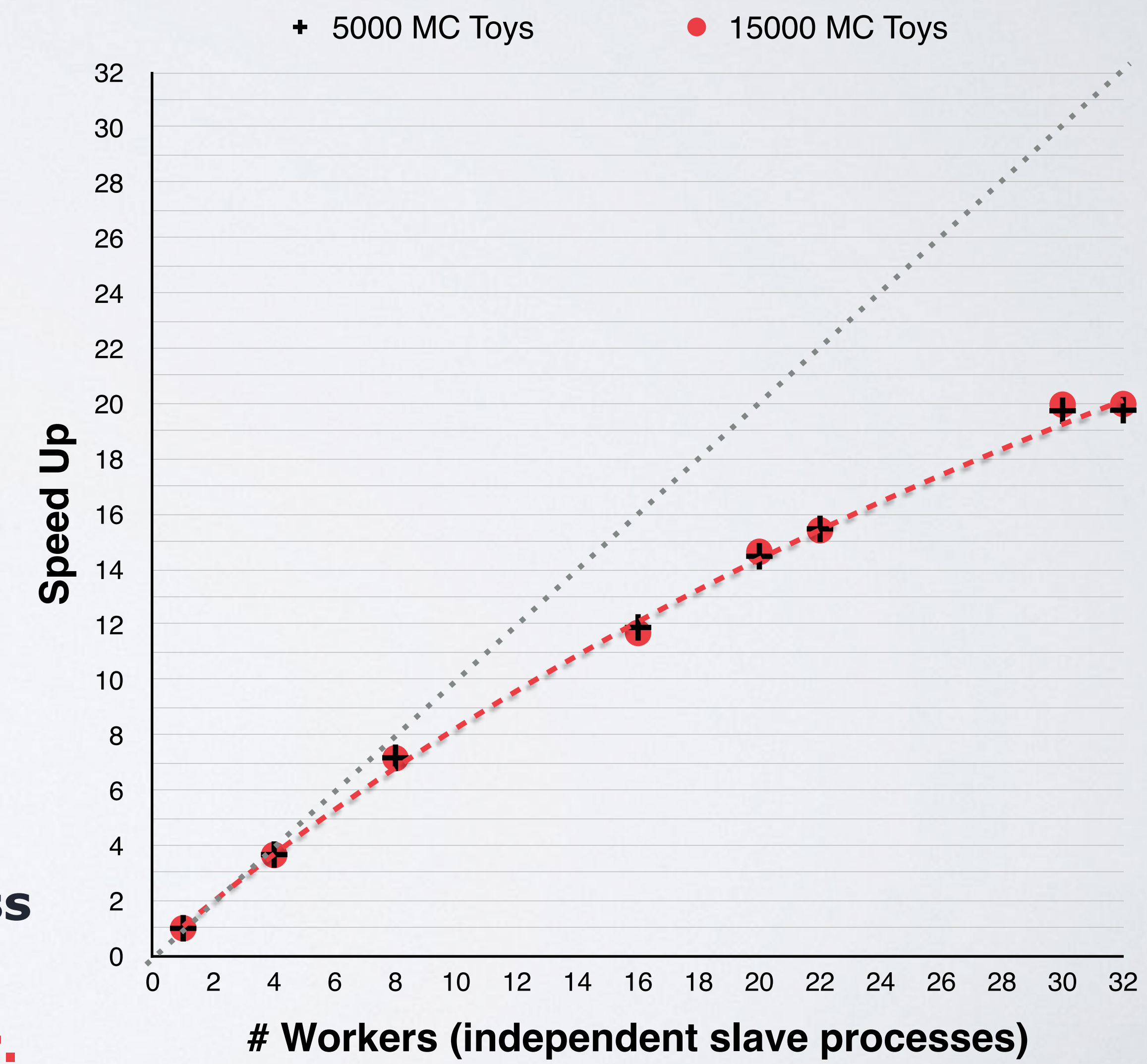
T_n time of n concurrent slave processes

$$S_n = \frac{T_1}{T_n}, \qquad 1 \le n \le n_{max} = 32$$

- **Comments :**

  1) perfect scaling with # of MC Toys per process

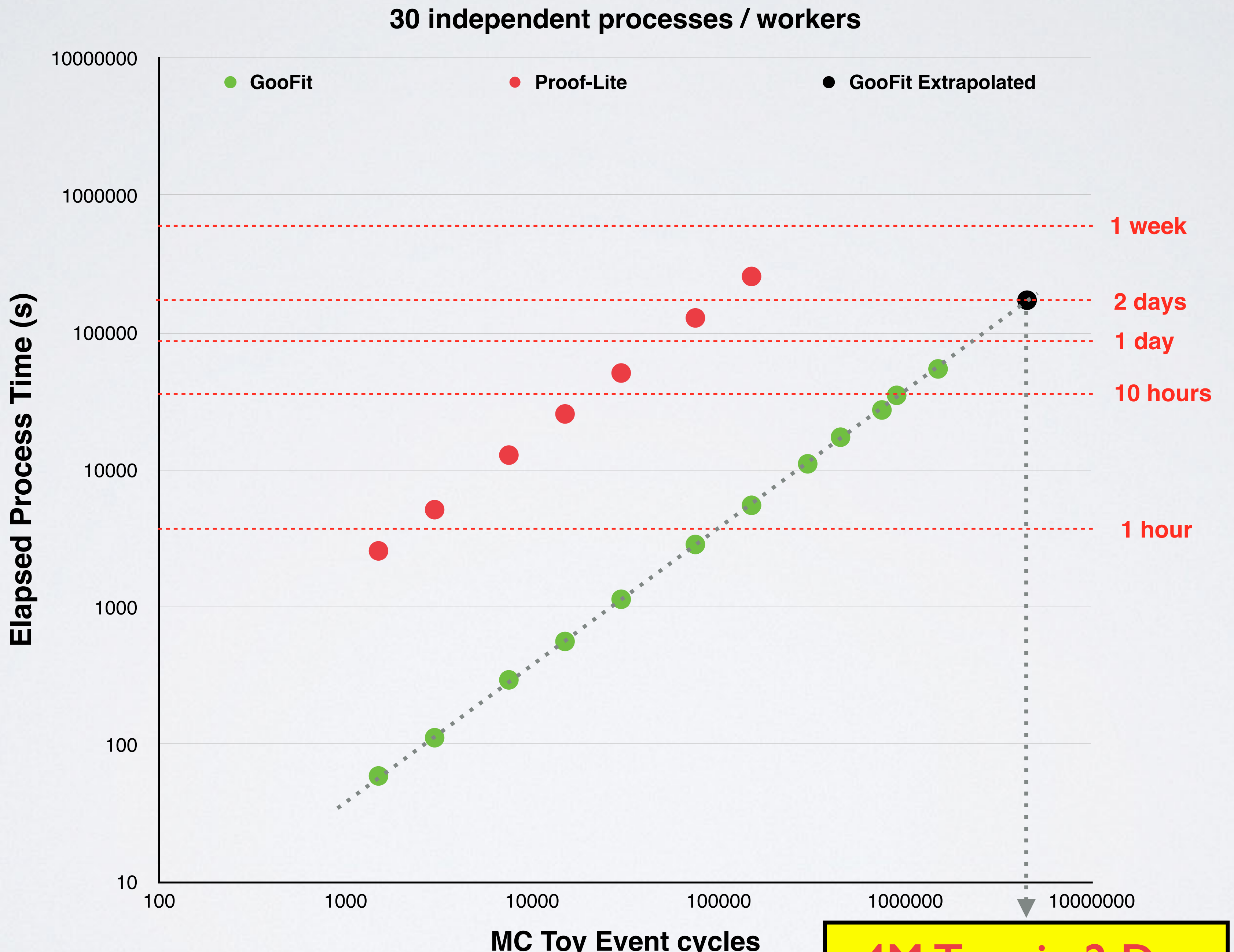
  2) Speed Up increases perfectly up to 8 workers, then shows a saturation behaviour.



[*] G.Ganis et al., *PoS ACAT08 (2008) 007*; A.Pompili et al., J. Phys.: Conf. Ser. **396** 032043, 2012

GooFit vs Proof-Lite performances - II

- We compare the time performances of 30 concurrent processes between **GooFit** (running on 2 GPUs + 30 CPUs) & **RooFit** (running via **Proof-Lite** on 30 CPUs)

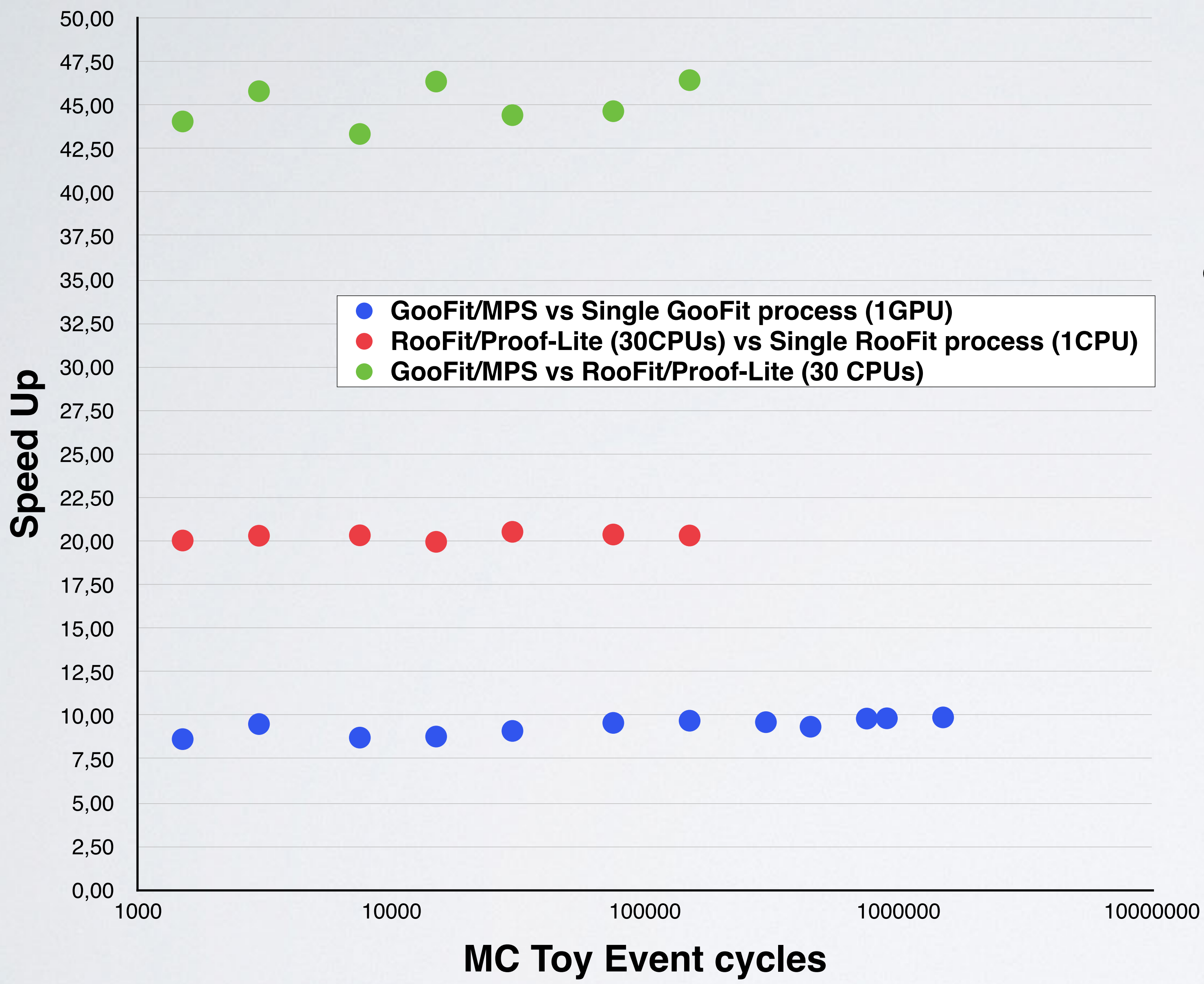


~4M Toys in 2 Days

- **In the next slide we show the relative Speed Ups**

GooFit vs Proof-Lite performances - III

- We compare the Speed Up of 30 concurrent processes between **GooFit** (running on 2 GPUs + 30 CPUs) and **RooFit** (running via **Proof-Lite** on 30 CPUs)



- **Comments :**

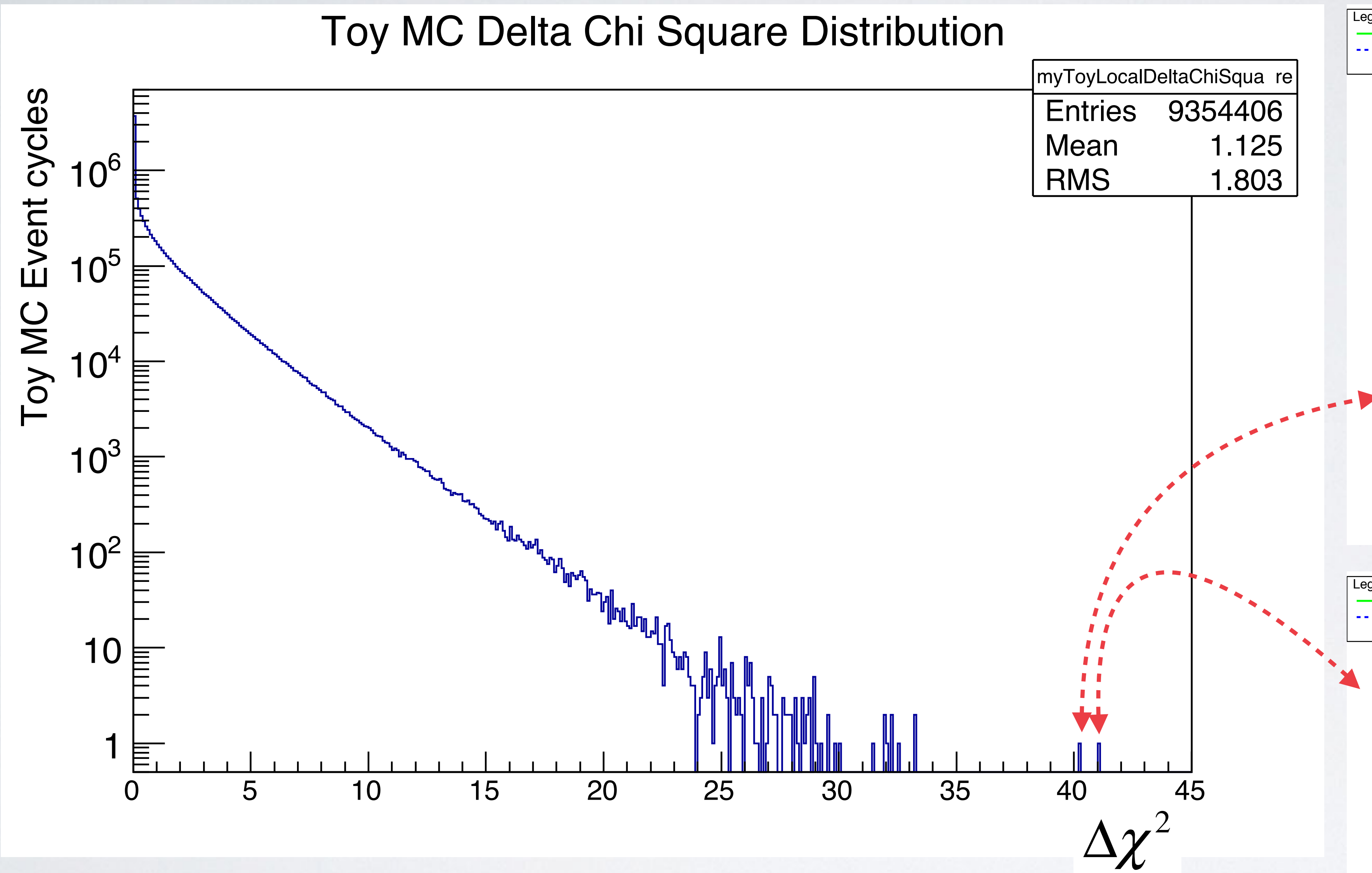
  1) **Rather stable Speed Up as a function of # of MC Toys (see ● , ●)**

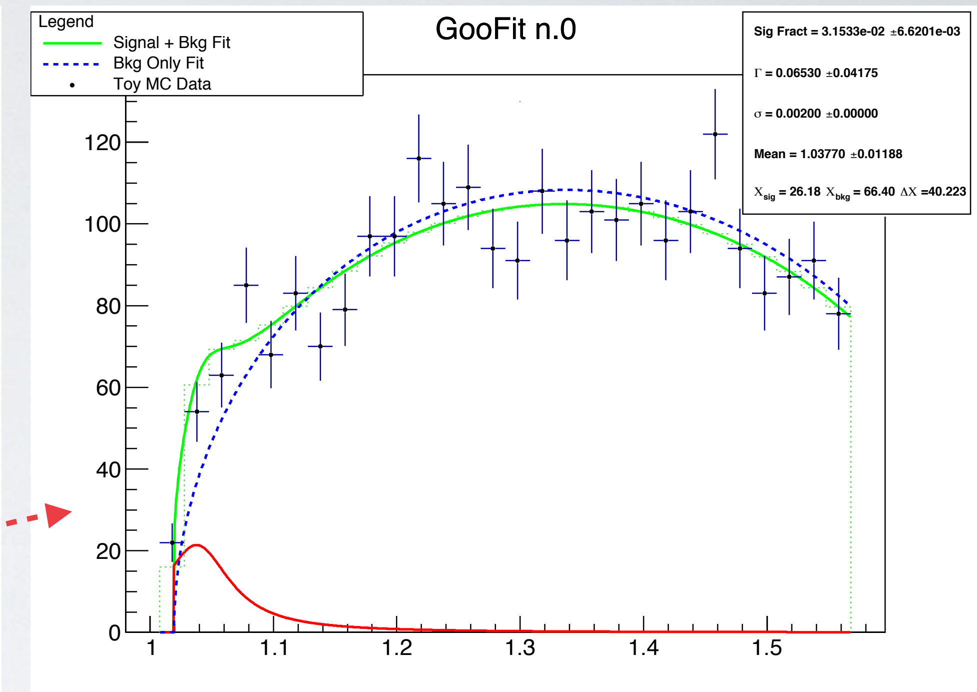
  2) **Speed Up between GooFit & RooFit is enough stable @ ~45(see ●)**

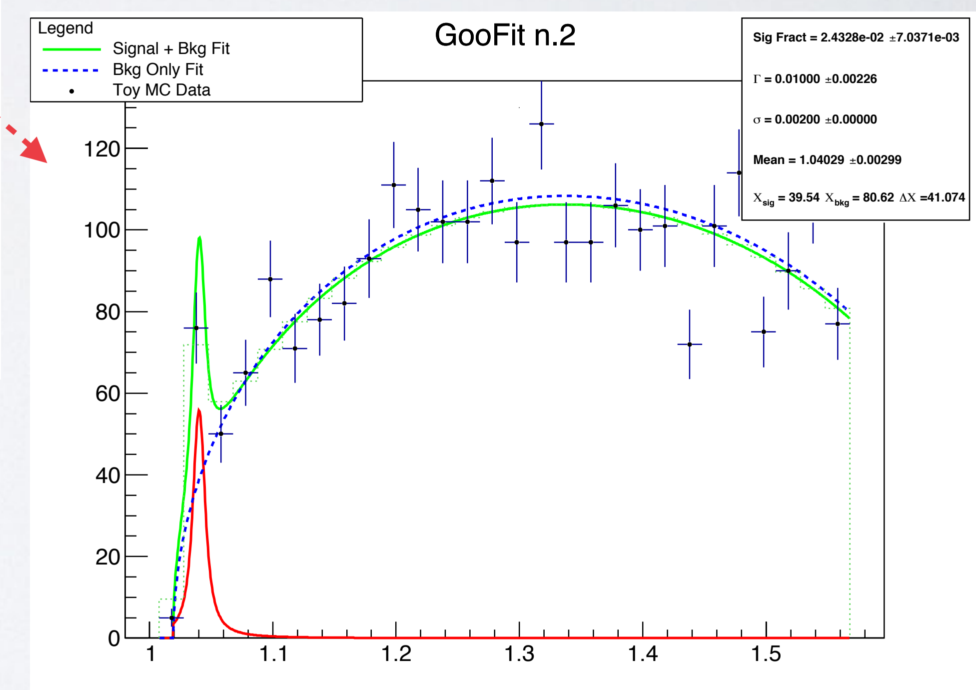
- **Extrapolation with this hardware configuration:**
  - **~40M of MC Toys with 20 days (GooFit)**
  - **~1M with 20 days (Proof-Lite)**

Example of large Toy MC production

- **As an example : we obtained a ~10M Toy MC $\Delta\chi^2$ distribution in ~5 days (using both GPUs)**







Conclusion & Outlook

Conclusions

- **Performance comparison :**
  - **GooFit single process (on GPU) *VS* RooFit (on CPU): Speed Up ~ 50**
  - **GooFit/MPS 30 processes on 2 GPUs *VS* RooFit/Proof-Lite on 30 CPUs : Speed Up ~ 45**

- **In similar applications with no need of integration over the bins, GooFit may gain a factor ~5 ÷10 (naive estimation).**
  Inclusion of LEE effect may affect the above estimated Speed Up.

Next steps

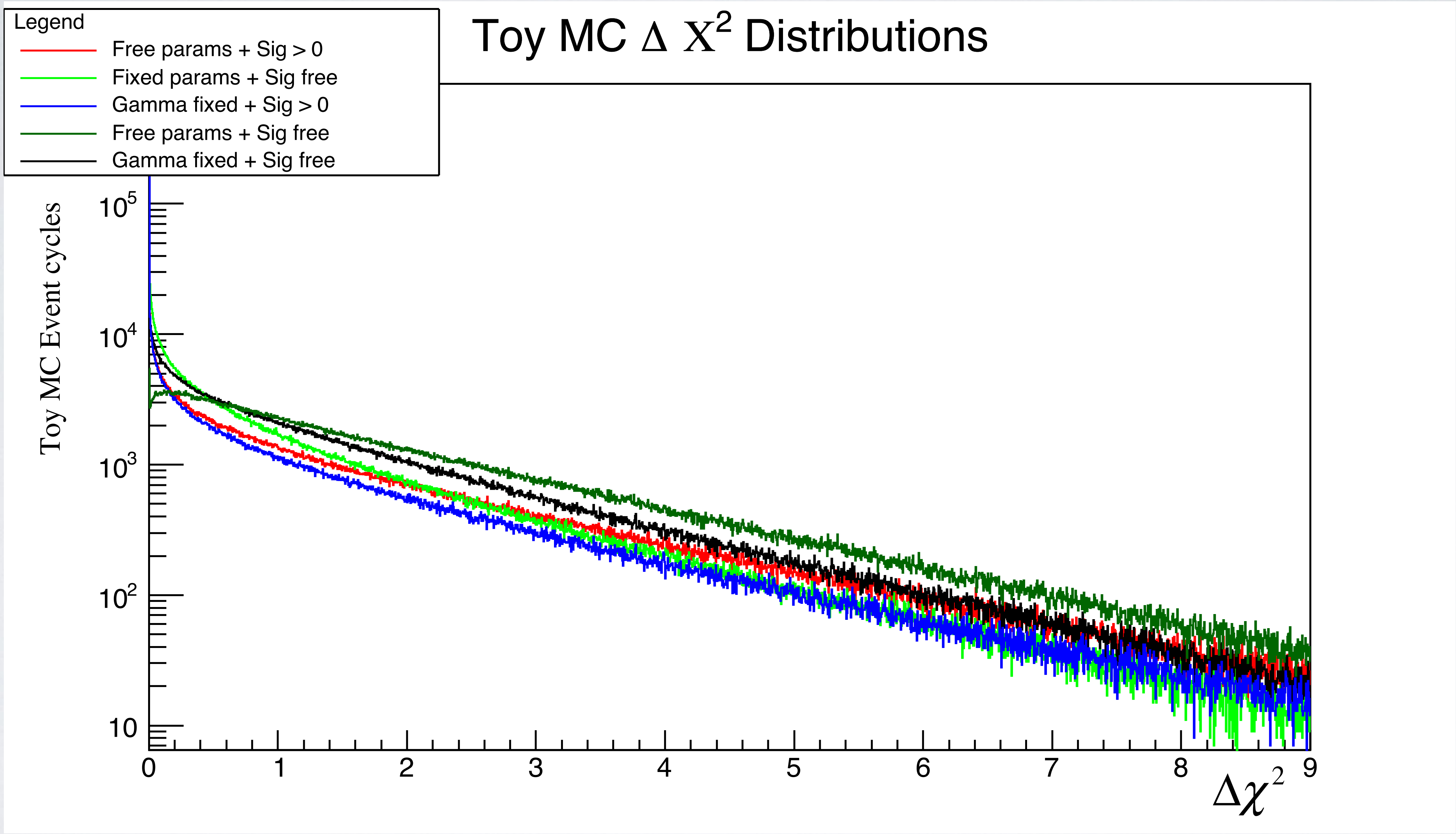
- **Performance comparison among different GPU technologies**
  [older : Tesla C2070 & newer : Tesla K40; both available @ Bari Tier2]

- **Completion of Adriano's Master Thesis**

Back Up

Comparison for different alternative hypothesis fit

- **The fit in red is our standard choice in the work of this presentation**



When constraining the signal yield to be non-negative the distributions (red & blue) are more peaked towards zero w.r.t. the case in which it is allowed to be positive or negative. This is consistent with findings of Narsky & Porter (2014) statistics book (ed. *WILEY-VCH).*