Monte Carlo Toy Based Confidence Limits with iDDS

Christian Weber

Confidence Limits in Analyses

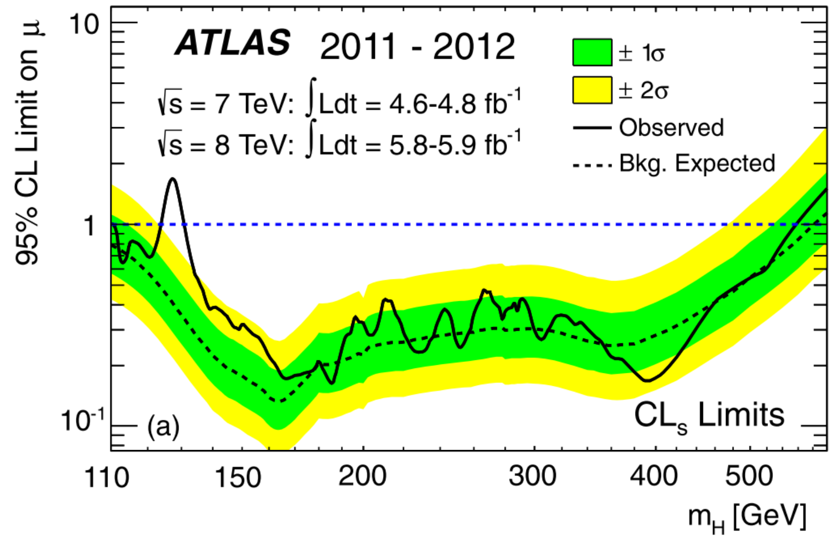
Analysis goals

- exclude some range of relevant phase space for the considered theory / process

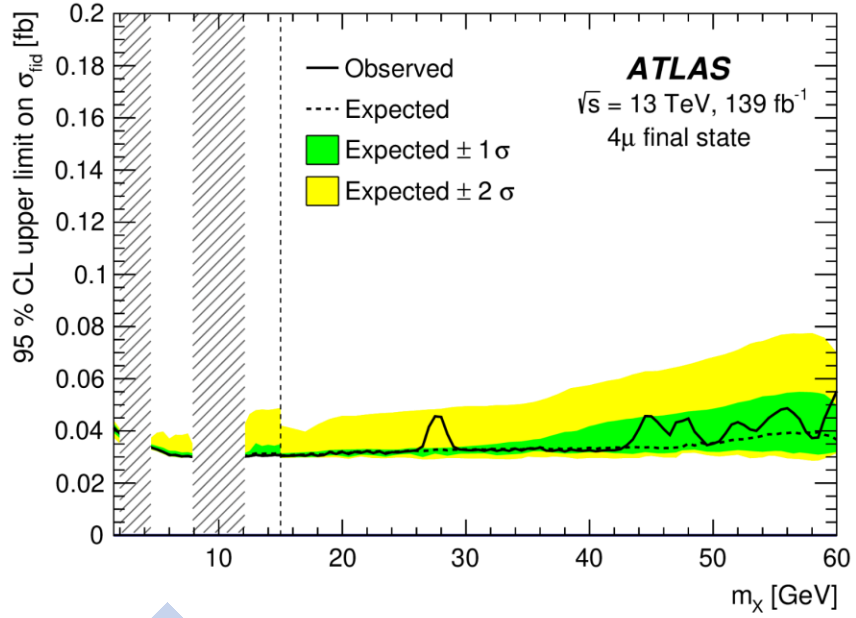
or

- show that obtained results is meaningfully different from what could have obtained by chance

Done estimating what results for a given model are expected at a chosen confidence level and comparing them to the observed result

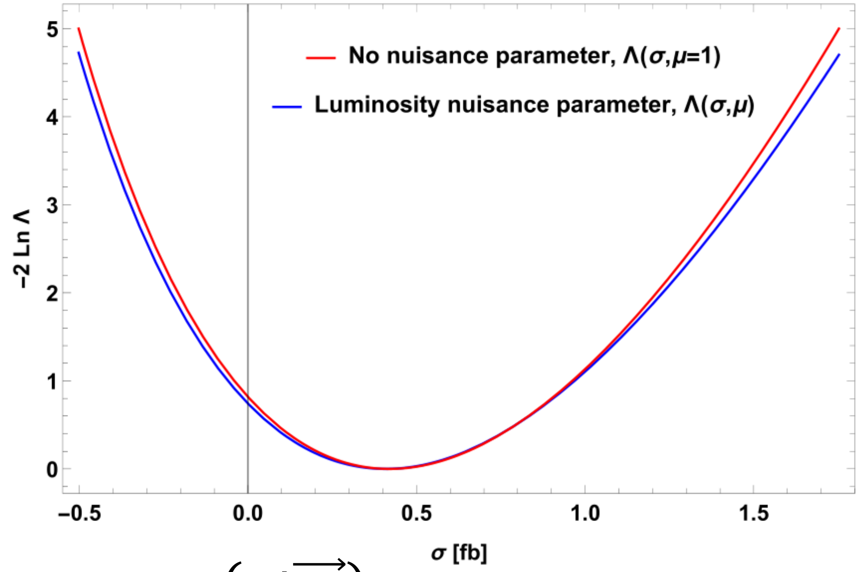


Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC



Search for Higgs bosons decaying into new spin-0 or spin-1 particles in four-lepton final states with the ATLAS detector with 139 fb^{-1} of pp collision data at $\sqrt{s}$=13 TeV

Hypothesis Test, Background



Define likelihood describing model

$L(\sigma|\vec{n}\,)$, e.g. $L(\sigma|\vec{n}\,) = \prod_i \mathrm{Pois}(b_i + \sigma\, s_i|n_i)$

Generate Monte Carlo toys for background only hypothesis $\sigma = 0$

$\overrightarrow{n^*} = [n_1^*, n_2^*, \dots]$

Estimate limit on σ using Monte Carlo toy $\overrightarrow{n^*}$

e.g. using profile likelihood $\Lambda(\sigma) = \frac{L(\sigma|\overrightarrow{n^*})}{L(\hat{\sigma}|\overrightarrow{n^*})}$, $\hat{\sigma}$ is global maximum of $L\left(\sigma|\overrightarrow{n^*}\right)$

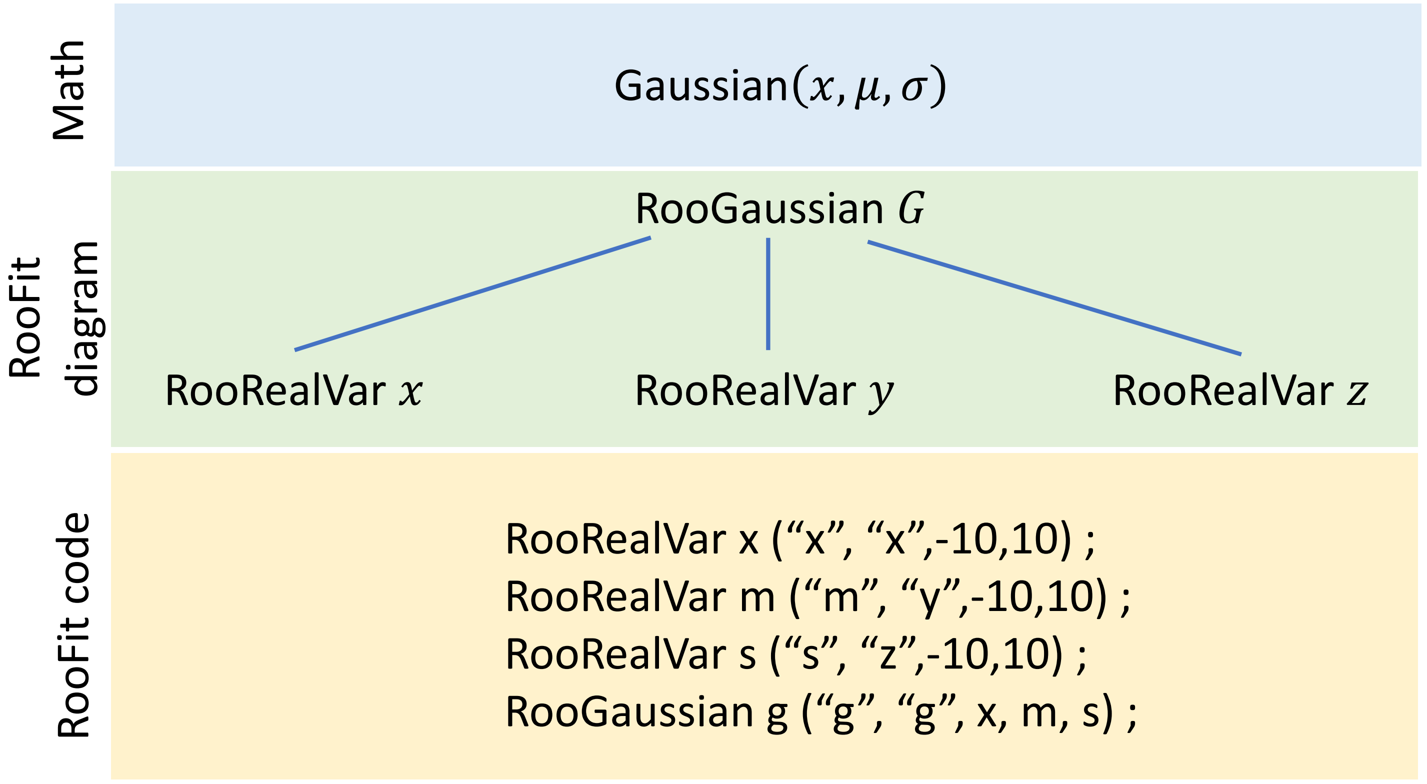
find σ^* such that $-2\ln\Lambda(\sigma^*) = S^2$ for CL limit on σ at significance S given $\overrightarrow{n^*}$

Do repeatedly with different Monte Carlo toys to get distribution of σ^*

Yields expected CL limit on σ and uncertainty bands around CL limit on σ

Hypothesis Test – Implementation I

Likelihood model, and associated data, configurations are stored in a RooFit.RooWorkspace



```
RooRealVar x (“x”, “x”,-10,10) ;
RooRealVar m (“m”, “y”,-10,10) ;
RooRealVar s (“s”, “z”,-10,10) ;
RooGaussian g (“g”, “g”, x, m, s) ;
```

- RooFit provides toolkit for modeling the expected distributions
- mathematical objects comprising likelihoods and PDFs are represented via C++ objects
- RooWorkspace serves a container class for all objects created
- Build more complex likelihoods from fundamental building blocks

```
w.factory(“Gaussian::gauss1(x[0,10],mean1[2],sigma[1])”)
w.factory(“Gaussian::gauss2(x,mean2[3],sigma)”)

w.factory(“SUM::sum(g1frac[0.5]*gauss1, gauss2)”)
```

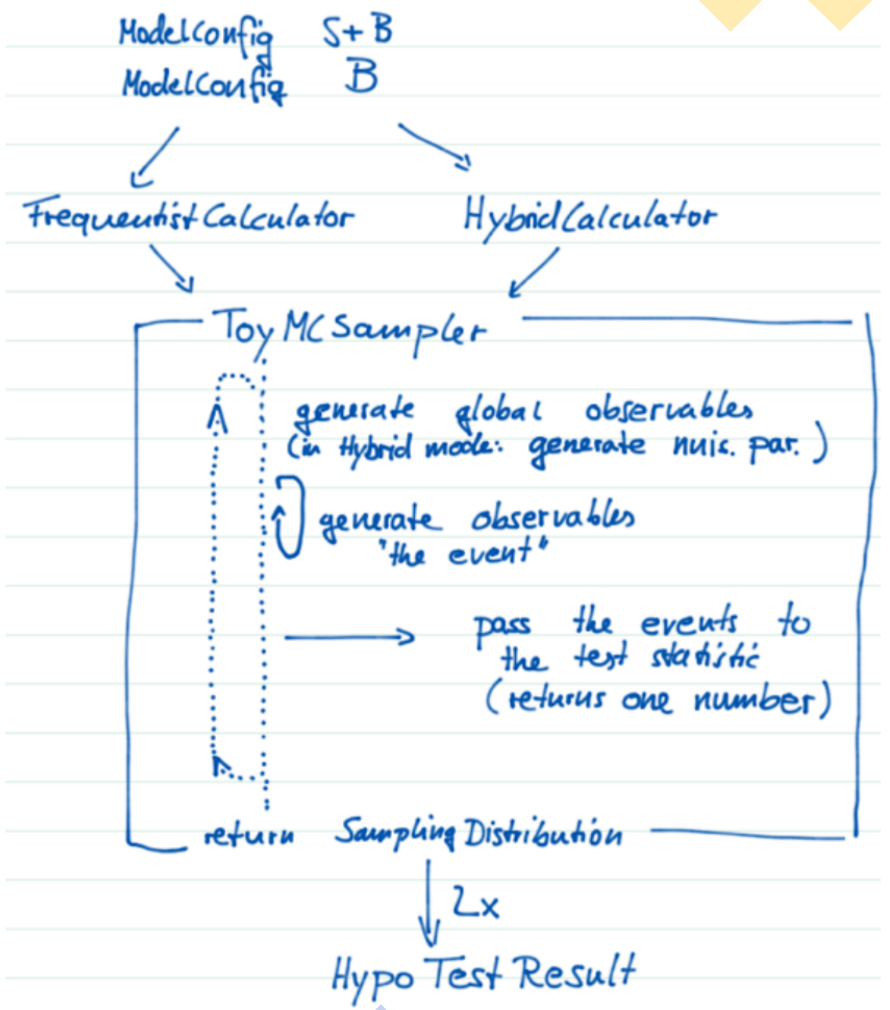
Hypothesis Test – Implementation II

Derive two model configurations from RooWorkspace
One background only
One signal + background

FrequentistCalculator
implements a frequentist procedure for sampling the test statistic distribution, uses ToyMCSampler

ToyMCSampler generates Toy Monte Carlo events for a given parameter point and evaluates a test statistic

HypoTestInverter yields upper limits by processing the test statistic distribution


ModelConfig S+B
ModelConfig B
FrequentistCalculator
HybridCalculator
ToyMCSampler
generate global observables
(in Hybrid mode: generate nuis. par.)
generate observables 'the event'
pass the events to the test statistic
(returns one number)
return Sampling Distribution
2x
HypoTestResult

Taken from RooFit/RooStats Tutorial

MC Toy limits on grid

Current HypoTestInverter usage to produce limits

- Load appropriate root version
- Download & build repository with the appropriate implementation of the HypoTestInverter
- Select parameter space and submit jobs to grid
- Monitor jobs, download & trim intermediate job results when done
- Postprocess outputs locally once all subjobs are done

Hypothesis Test with iDDS

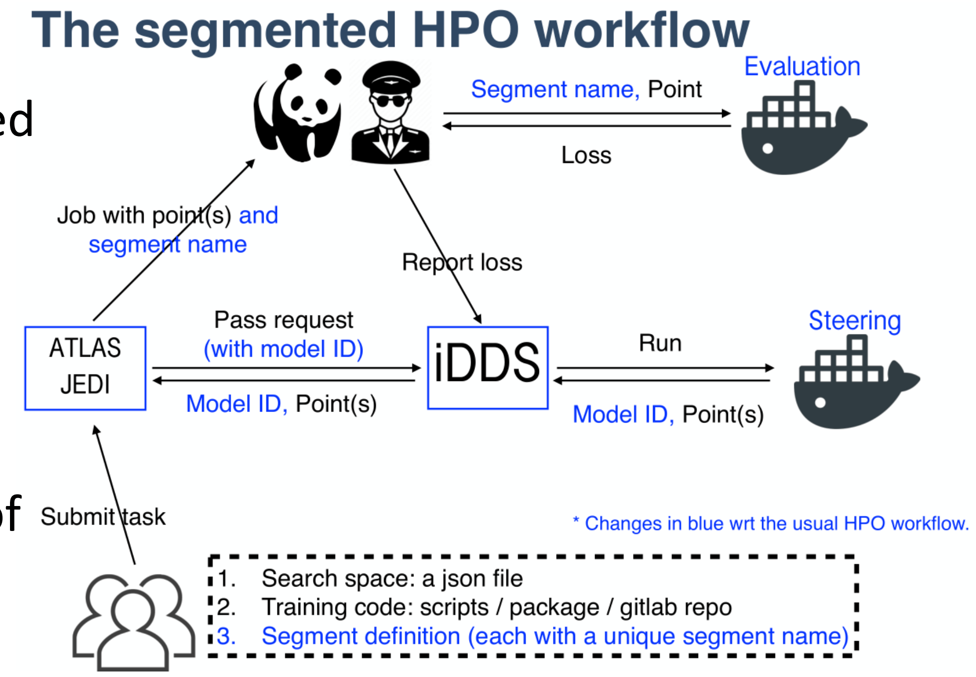
Monte Carlo toy calculations are independent of each other

Post processing required to extract limit from sampled test statistic distribution, requires all Monte Carlo toy calculations to be completed

Piggyback on Hyper Parameter Optimization infrastructure (HPO)

- Special Steering Container that runs a fixed number of Evaluation steps and coordinates what task the Evaluation container does
- Evaluation part performs Monte Carlo Toy calculations, trimming of intermediate results, or combination thereof

iDDS – intelligent Data Delivery Service


The segmented HPO workflow
Evaluation
Segment name, Point
Loss
Job with point(s) and segment name
Report loss
Pass request (with model ID)
ATLAS JEDI
iDDS
Run
Steering
Model ID, Point(s)
Model ID, Point(s)
Submit task
* Changes in blue wrt the usual HPO workflow.
1. Search space: a json file
2. Training code: scripts / package / gitlab repo
3. Segment definition (each with a unique segment name)

Many thank to Wen Guan, who provided the special Steering Container!

MC Toy limits on grid – iDDS - implementation

- Based around Container executing Hypotestinverter https://hub.docker.com/repository/docker/christiantmweber/standardhypotestinverter_for_idds
- Implements StandarHypoTestInv from https://gitlab.cern.ch/chweber/StandardHypoTestInv

  forked from https://gitlab.cern.ch/atlas_higgs_combination/software/StandardHypoTestInv

On the user side everything is contained in one python script https://gitlab.cern.ch/chweber/StandardHypoTestInv/-/blob/master/iDDS_script/iDDsHypoTester.py

One command to submit limit setting*

```
python iDDsHypoTester.py
--inputFile <file with RooWospace>
--pointsToScan <rage of Parameter of Interest to scan>
--nToys  <#signal & background toys per scan point>
--rucioScope <target rucio scope>
--nParallelComputations <number of times to repeat the calculations>
```

+ additional options to specify details of the RooWorkspace

* After setting up the environment with four lines

MC Toy limits on grid – iDDS - implementation II

python iDDsHypoTester.py

Creates than steering script for iDDS

Steers to container to make < nParallelComputations> computations of Monte Carlo Toy scans

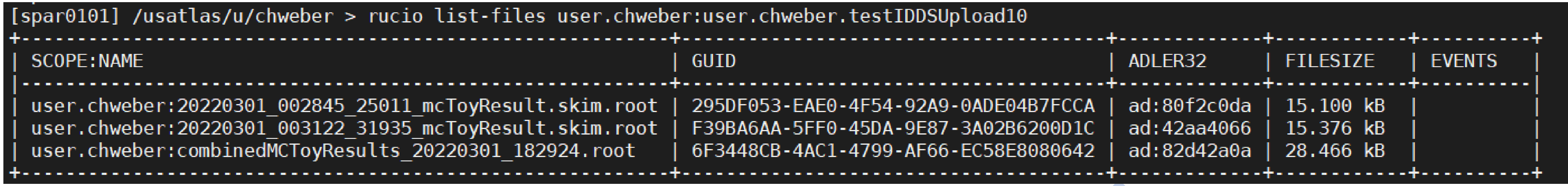
Steering script and <`inputFile`> are automatically uploaded to iDDS

Each Monte Carlo Toy scans is performed explicitly with random seed

Individual scan results are shared between instances via upload to < rucioScope >

After < nParallelComputations > scans combination step is automatically triggered

Results are uploaded to < rucioScope >, also displayed in logs



```
[spar0101] /usatlas/u/chweber > rucio list-files user.chweber:user.chweber.testIDDSUpload10
+-----------------------------------------------------------+--------------------------------------+-------------+------------+----------+
| SCOPE:NAME                                                | GUID                                 | ADLER32     | FILESIZE   | EVENTS   |
|-----------------------------------------------------------+--------------------------------------+-------------+------------+----------|
| user.chweber:20220301_002845_25011_mcToyResult.skim.root | 295DF053-EAE0-4F54-92A9-0ADE04B7FCCA | ad:80f2c0da | 15.100 kB  |          |
| user.chweber:20220301_003122_31935_mcToyResult.skim.root | F39BA6AA-5FF0-45DA-9E87-3A02B6200D1C | ad:42aa4066 | 15.376 kB  |          |
| user.chweber:combinedMCToyResults_20220301_182924.root   | 6F3448CB-4AC1-4799-AF66-EC58E8080642 | ad:82d42a0a | 28.466 kB  |          |
+-----------------------------------------------------------+--------------------------------------+-------------+------------+----------+
```

Conclusion

iDDS interface for HypoTestInverter now available

- easy to use
- streamlines tasks for users
- new application for iDDS

Future Options

- Extend the interface to significance calculations
- Allow the scan range, number of toys and number of scans to be optimized algorithmically while running on iDDS
- Enable usage of GPU resources for toy limit calculations

Again, many thank to Wen Guan, for advice and support with respect to iDDS!

¡Thank you!

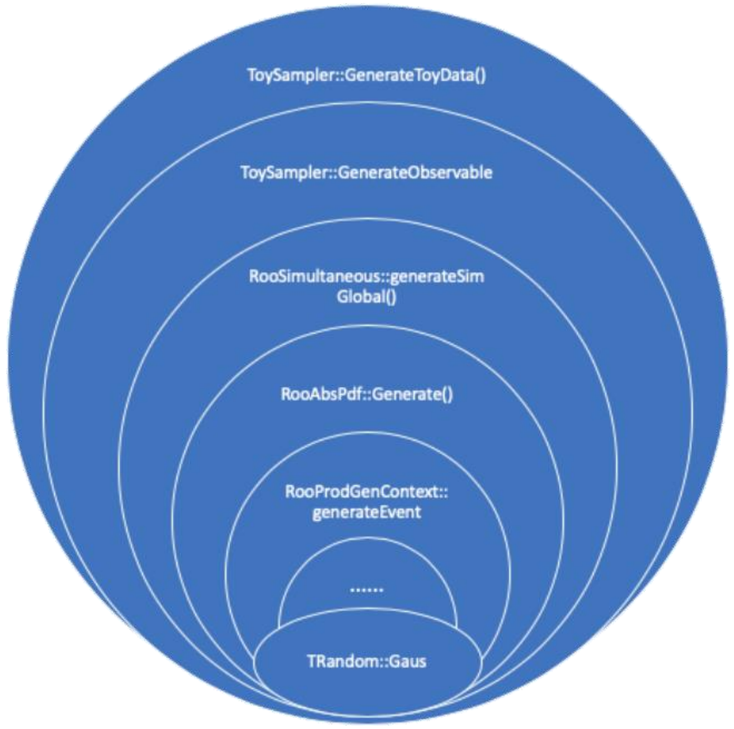
¡End of Presentation!

Enabling of GPU resources for hypothesis testing

More distant goal is to make GPU resources available for hypothesis testing using ROOT

Requires a ROOT version where the relevant methods are CUDA enabled

All studies here on ROOT and GPUs are taken from **Xola Mapekula**


ToySampler::GenerateToyData()
ToySampler::GenerateObservable
RooSimultaneous::generateSim Global()
RooAbsPdf::Generate()
RooProdGenContext:: generateEvent
......
TRandom::Gaus

dependencies of sampling distribution function for generation of MC toys

ROOT.StandardHypotestInverter with GPUs

Flow of MC Toy limit calculation

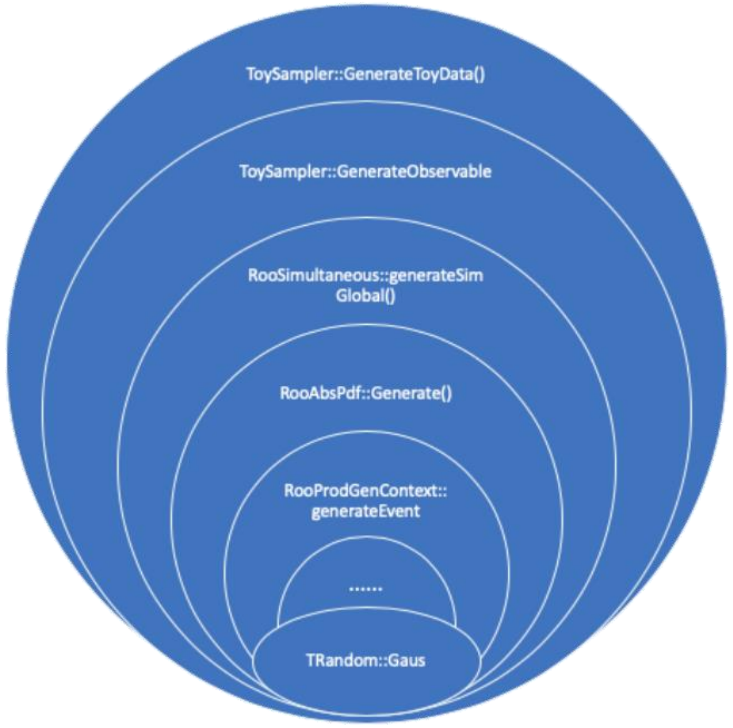
1. Generate MC toys for construction of test statistic
2. Evaluate test statistic by finding values interest and nuisance parameters that minimize profile likelihood
3. Use optimize parameter values to calculate p_0-value

Calculation of toys central for hypothesis test, derived from likelihood

$$L_i(\mu_k, \theta) = \prod_{j=1}^{N} \frac{(\mu s_j + b_j)^{n_j} e^{(\mu s_j + b_j)}}{n_j!} \prod_{j=1}^{N} (\theta_j^0 - \theta_j; \sigma_j)$$

Relevant Likelihoods are products of Gaussian and Poisson probability density functions

Estimate possible speed up on toy calculation with GPU


ToySampler::GenerateToyData()
ToySampler::GenerateObservable
RooSimultaneous::generateSim Global()
RooAbsPdf::Generate()
RooProdGenContext:: generateEvent
......
TRandom::Gaus

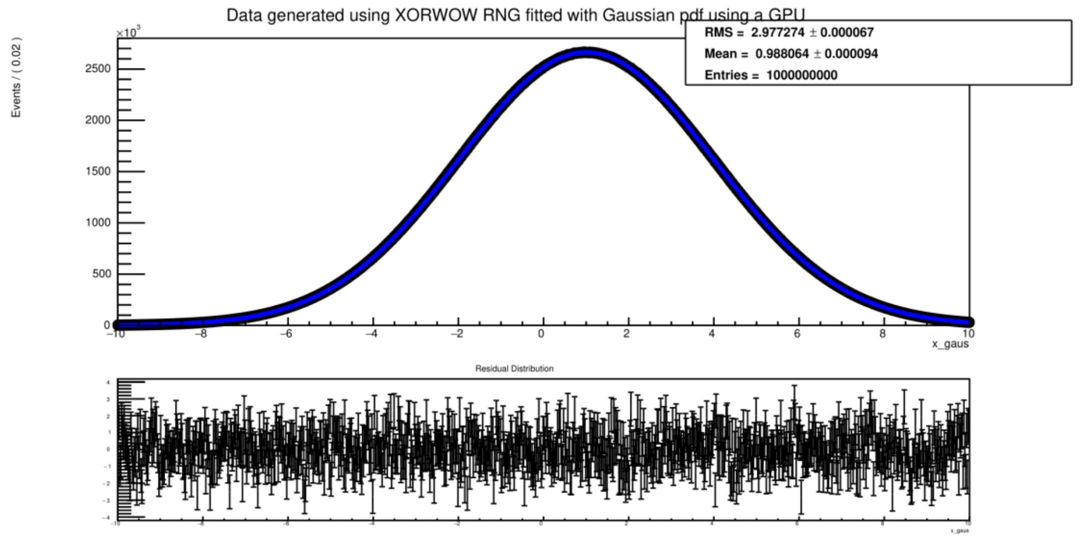
dependencies of sampling distribution function for generation of MC toys

Toy GPU calculation speedup estimate

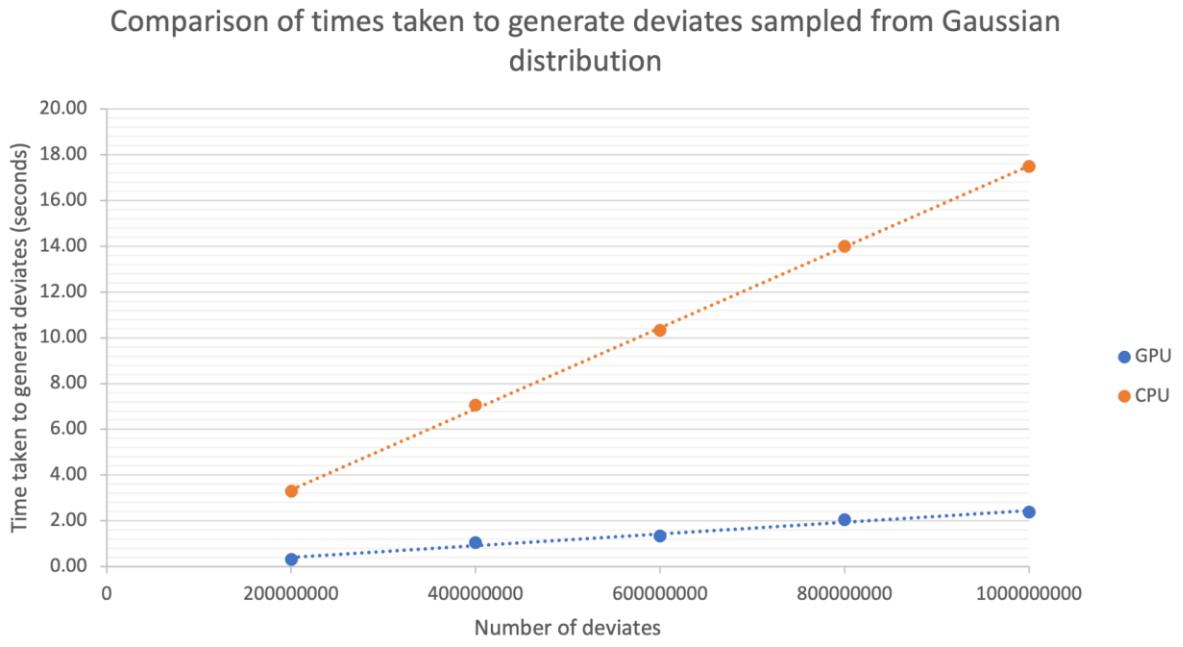
Generation of toy ends in call for calculation of Gaussian deviate

Use calculation of Gaussian deviate as proxy for Toy calculation

Replace Mersenne Twister (MTGP32) with XORWOW RNG to enable running on GPU



Gaussian fit to 10^9 Gaussian deviates generated with XORWOW on GPU



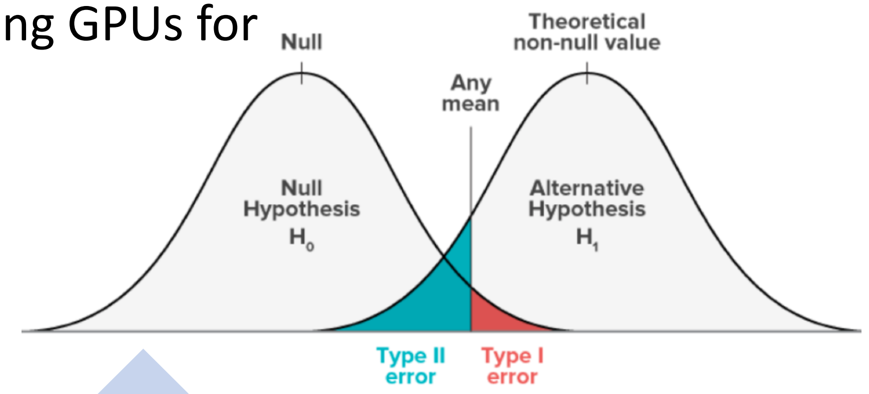
7.2x speedup @ 10^9 toys

Work and figures related to GPUs from Xola Mapekula

Conclusion

- Ongoing work to enable Monte Carlo toy based hypothesis testing with iDDS
- Piggybacks on HPO service
- Setup StandardHypotestInverter evaluation container for iDDS
- Tested with the HPO framework on the grid with existing HPO steering container
  - Generate one-dimensional uniform parameter space to map onto range of parameter of interest
- Next step: add postprocessing of toy results to Steering Container

- Future: enable utilization of GPU resources
  - See here and here for presentations of Xola's work on enabling GPUs for ROOT.StandardHypotestInverter


Null
Theoretical non-null value
Any mean
Null Hypothesis H_0
Alternative Hypothesis H_1
Type II error
Type I error