# DB12 Benchmark + LHCb Benchmarking

**Andrew McNab**
University of Manchester
LHCb and GridPP

# DIRAC 2012 fast benchmark ("DB12")

- What is benchmarking about?
- DB12 origins and current status: within DIRAC; within DB12 repo
- DB12-in-job vs DB12-at-boot
- DB12-at-boot in MJF
- Consistency results from Manchester

# LHCb benchmarking

- Use cases
- HS06 vs "Job Power"
- DB12-in-job vs "Job Power"
- DB12-at-boot vs "Job Power"

# DIRAC 2012 fast benchmark (DB12)

# CPU Performance Benchmarking

- Fundamental aim of benchmarking is to attempt **to predict the rate at which a given computer can run applications of interest**
  - Prediction either relative ("it will be twice as fast on this CPU as that one") or absolute ("these events will take 43.5 hours to simulate")
  - So benchmarking is about constructing theories of CPU performance
  - Usual requirements apply: **theories should be as simple as possible, and make accurate, consistent, reproducible predictions**
- CPU performance depends on multiple fundamental metrics
  - Clock speed, instructions per clock cycle, complexity of instructions, branch prediction, cache sizes, cache speed, memory speed, …
- Simple model is that speed in executing a given task is a linear combination of the fundamental metrics for that CPU
  - In general, weights will be different for different applications
  - **A good benchmark for a given application has the same set of weights for the metrics as the application itself**

# CPU Performance Benchmarking (2)

- However, the individual metrics' weights are not usually observable
- What we see is the overall benchmark speed and the overall application speed, and we compare those
- Benchmark suites (like SPEC06) attempt to provide multiple benchmarks with varying dependencies (weights) on the fundamental metrics of CPUs
  - Hope that benchmarks form a basis (in linear algebra terms)
  - The weights appropriate to any application can then be achieved by forming a linear combination of the basis set of benchmarks
  - eg appSpeed = 1.0 x busSpeed + 0.4 x cpuSpeed (fundamental metrics)

    = 0.4 x BM1 + 1.0 x BM2 (suite benchmarks)

    where BM1=(0.5 x bS + 0.5 x cS) and BM2=(0.8 x bS + 0.2 x cS)
- So, what benchmarks are appropriate for our application domains?
- And what is convenient? What provides a basis? Can represent any app?

# DIRAC 2012 fast benchmark (DB12)

- Benchmark that has been built-in to DIRAC since 2012
  - Very simple: Python random.normalvariate() run lots of times
  - Single benchmark is an excellent predictor of LHCb Monte Carlo!
- Run at start of each DIRAC pilot
  - “DB12-in-job” scenario
  - So available to LHCb, Belle II, ILC, CTA, EGI+GridPP DIRAC VOs, …
- Results available to pilot+payload and recorded centrally by DIRAC
- Scale factor originally tuned to match HS06 in 2012
  - Change in required normalisation to HS06 found at start of 2016
  - References to “DB16” just mean DB12/0.65
  - This is a moving target, so focus on DB12 now rather than trying to keep HS06 alignment

# DB12 repository in GitHub

- DB12 “ultimate upstream” at https://github.com/DIRACGrid/DB12
- Single Python file, containing the function originally used by DIRAC
  - DIRACbenchmark.py
- Also wrapper functions that will run multiple benchmark processes in parallel
- Each iteration less than a minute, and does thousands of inner loops
- Typically one initial and one final iteration that are discarded, and one or more iterations that timings are based on
  - This scenario avoids start-up costs (cache etc); and keeps everything busy if some benchmark processes finish early
- Python file can also be run as a command line utility
  - ./DIRACbenchmark.py —wholenode

# DB12 and Machine/Job Features

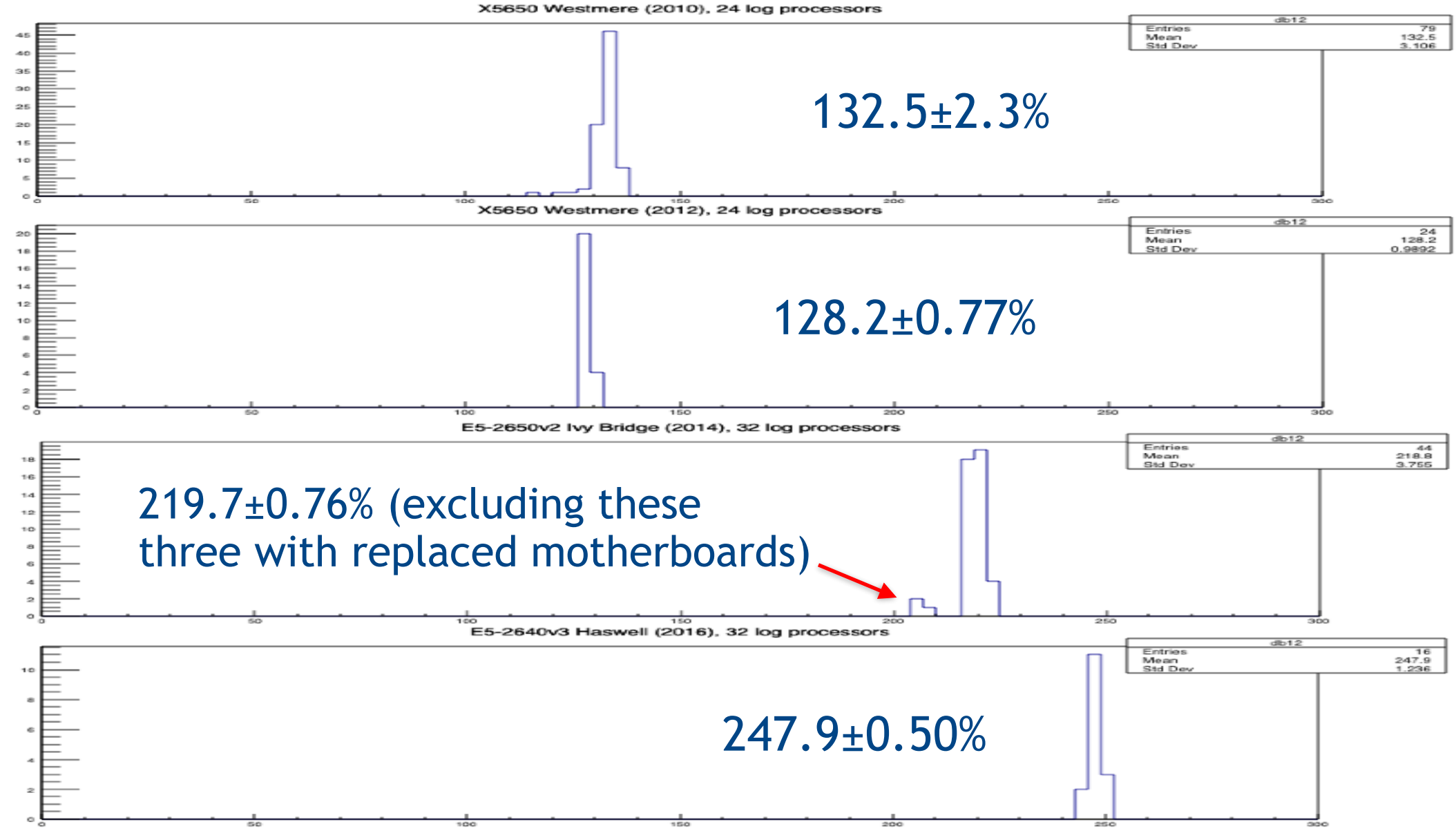
- Wholenode and jobslot modes of DIRACbenchmark.py try to use
  - $MACHINEFEATURES/total_cpu
  - and $JOBFEATURES/allocated_cpu

  to discover how many benchmark processes to run (can be overridden)
- RPMs/scripts in https://gitlab.cern.ch/machinejobfeatures/mjf-scripts exist for Torque, HTCondor, Grid Engine
  - Create $MACHINEFEATURES/db12 and $JOBFEATURES/db12_job if db12 info provided
  - By analogy with the hs06/hs06_job files created using measured performance of the machine carried out by the site
    - During commissioning?
  - But with DB12, benchmarking is straightforward to automate …

# db12-at-boot

- Either just install the mjf-db12 RPM or the scripts it contains
- Installs a SysV init script to run DB12 at the start of the boot process
  - So /etc/rc.d/rc3.d/S01db12
- Automated, so can avoid keeping a database of per-host values
- Uses ACPI to try to temporarily enable full processor speed (ie disable dynamic Turbo)
  - This avoids need for CPU to “warm up” for benchmarking
- Stores result in /etc/db12/ where MJF scripts will find it
- Next slides based on preliminary results from a mass-reboot at Manchester last week
- We see excellent consistency between machines of the same model
  - ±1-2% for a benchmark taking less than 2 minutes

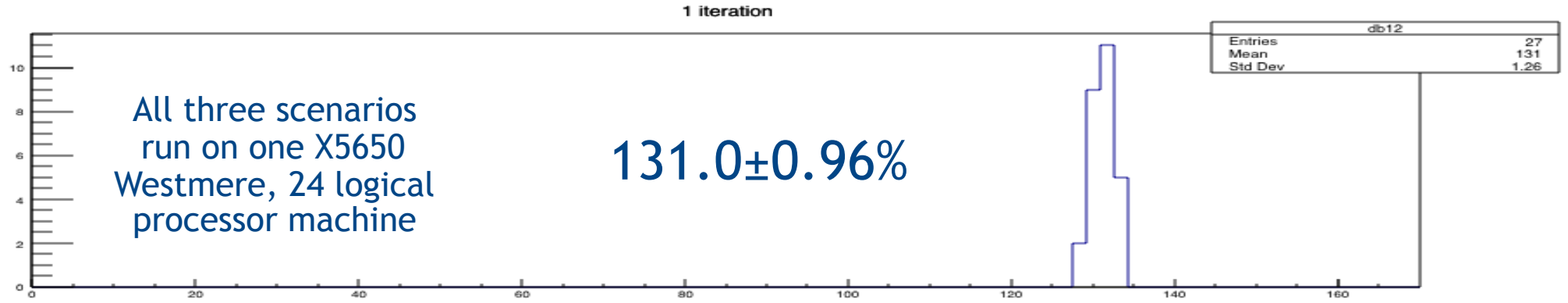
# db12-at-boot: multiple machines



# db12-at-boot: multiple iterations

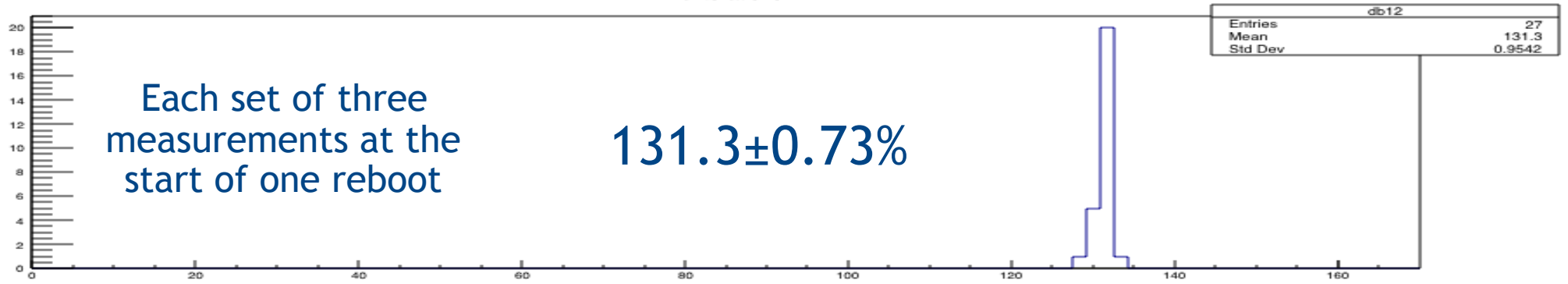
All three scenarios run on one X5650 Westmere, 24 logical processor machine

131.0±0.96%



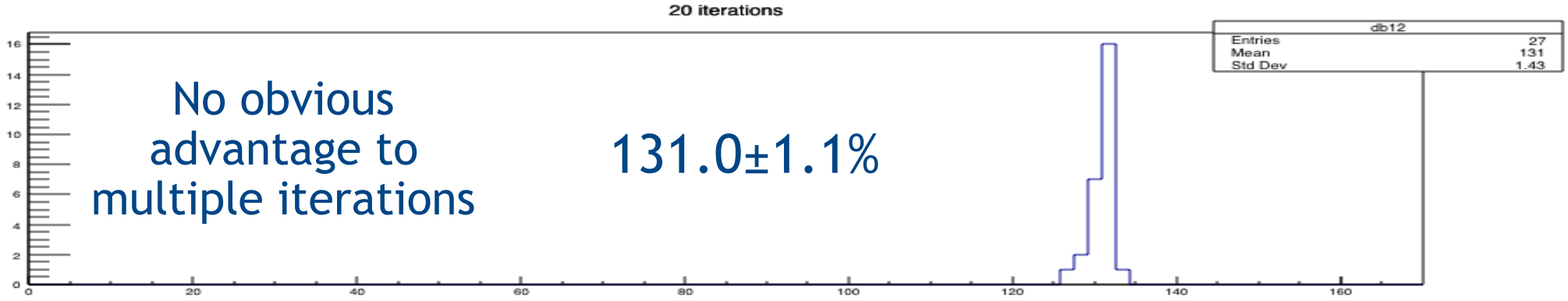

Each set of three measurements at the start of one reboot

131.3±0.73%




No obvious advantage to multiple iterations

131.0±1.1%

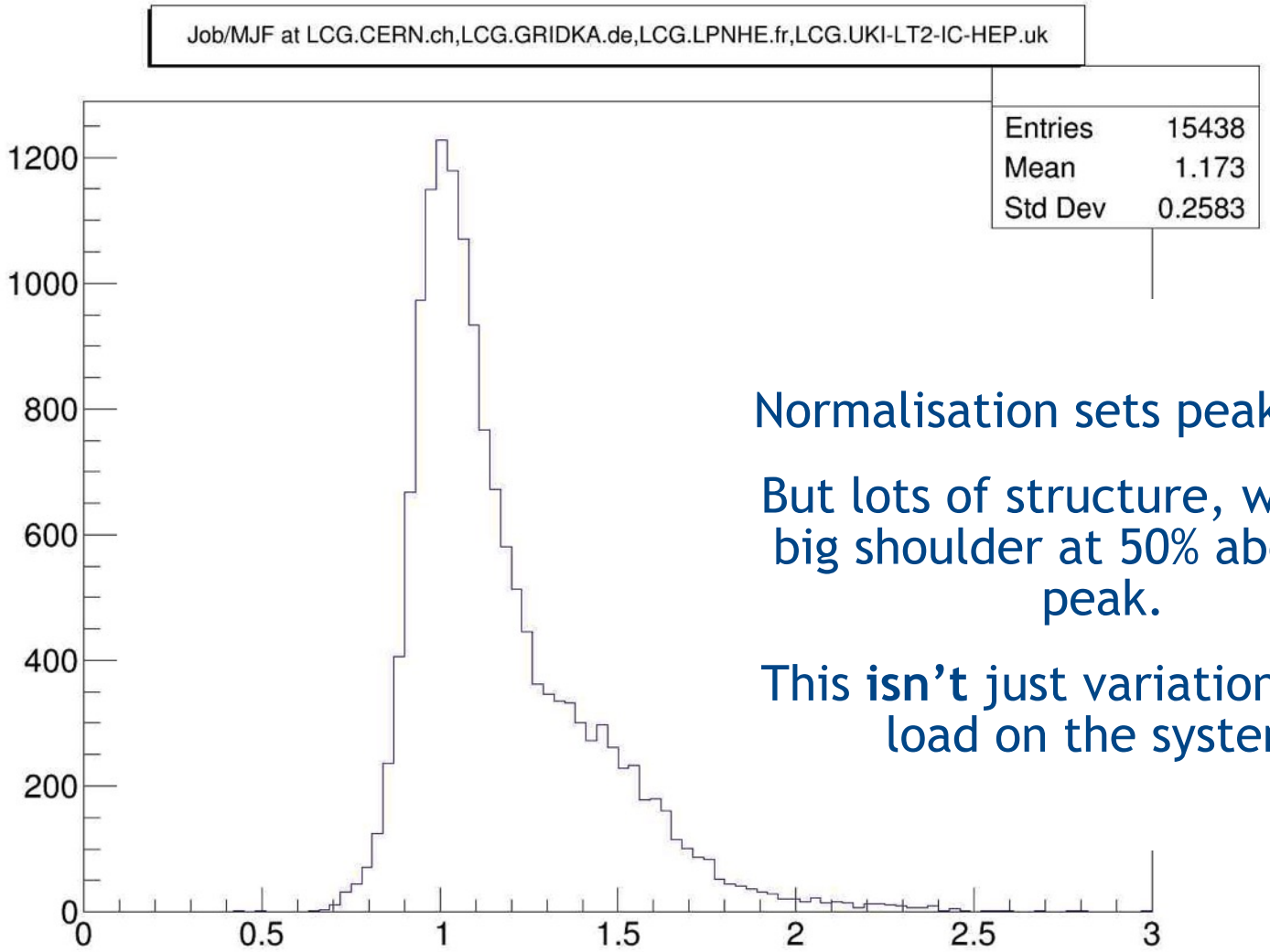


# Benchmarks vs LHCb jobs

# HS06, Job Power and DIRAC Power

- Graphs from Philippe Charpentier's comparisons of jobs and benchmarks
  - See Dec 2015 GDB talk, and CHEP 2016 talk/paper for more details of procedure
- Compare four estimates of power allocated to a job
  - **HS06** taken from Machine/Job Features
  - **Job Power** proportional to LHCb Monte Carlo events per second
  - **"DIRAC Power"** uses DB12-in-job or DB12-at-boot
- Job Power is normalised to be equal to DB12/0.65 (≈HS06) on a reference platform (E5-2650v2, 2.60GHz at RAL)
- Focus is on Monte Carlo since it's 75% of LHCb offline workload

# JobPower vs HS06 (from MJF)

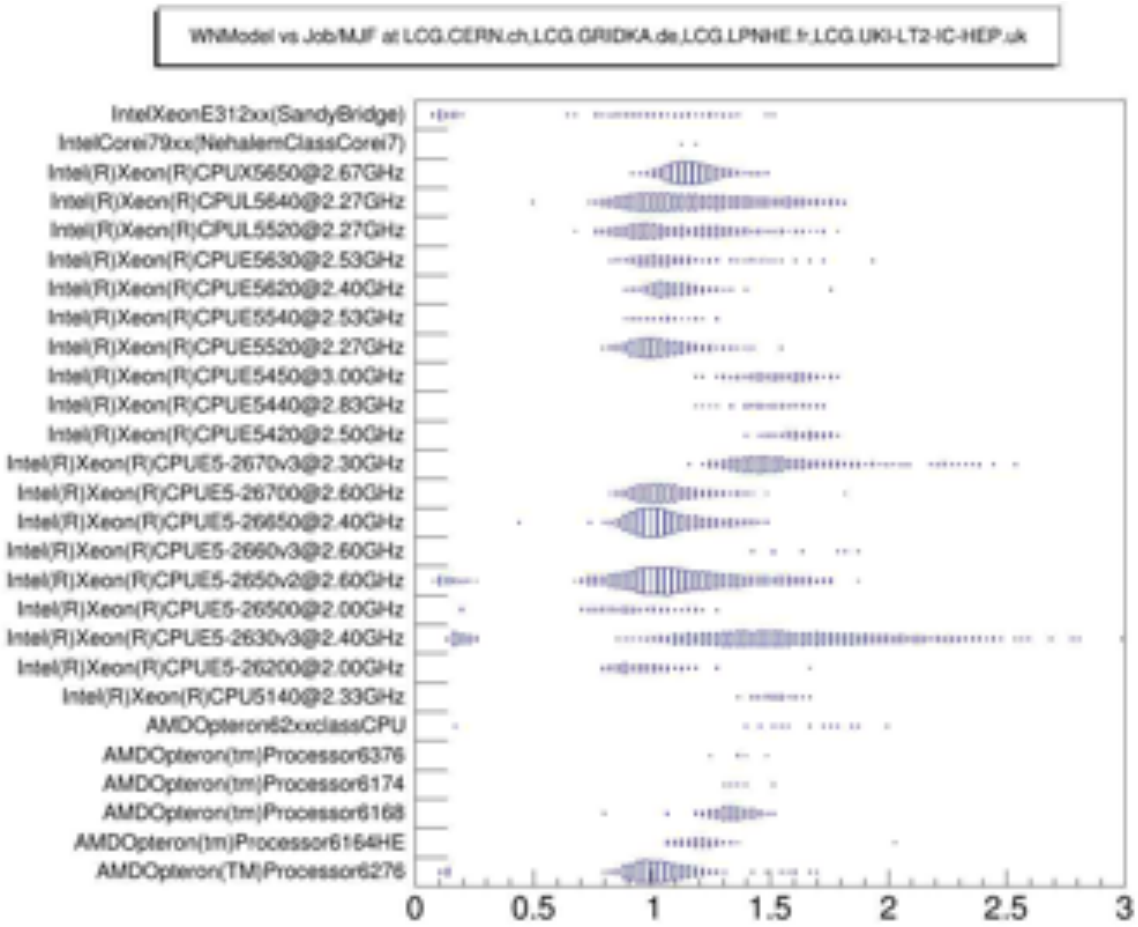



Normalisation sets peak to ~1.0

But lots of structure, with that big shoulder at 50% above the peak.

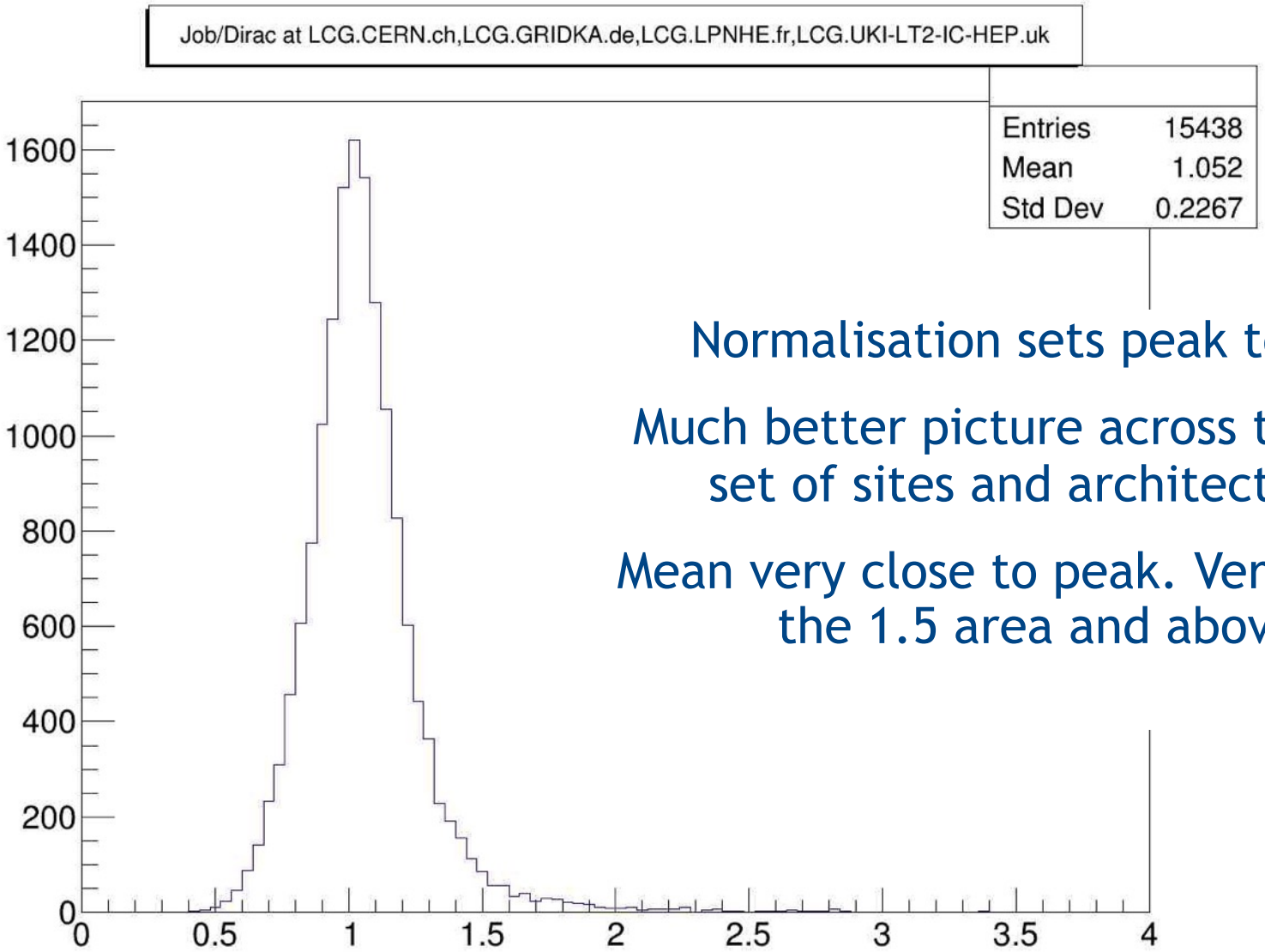
This **isn't** just variations in the load on the system.

# JobPower vs HS06 (from MJF)




Events per HS06 varying by ~50% depending on architecture.

# JobPower vs DB12-in-job benchmark

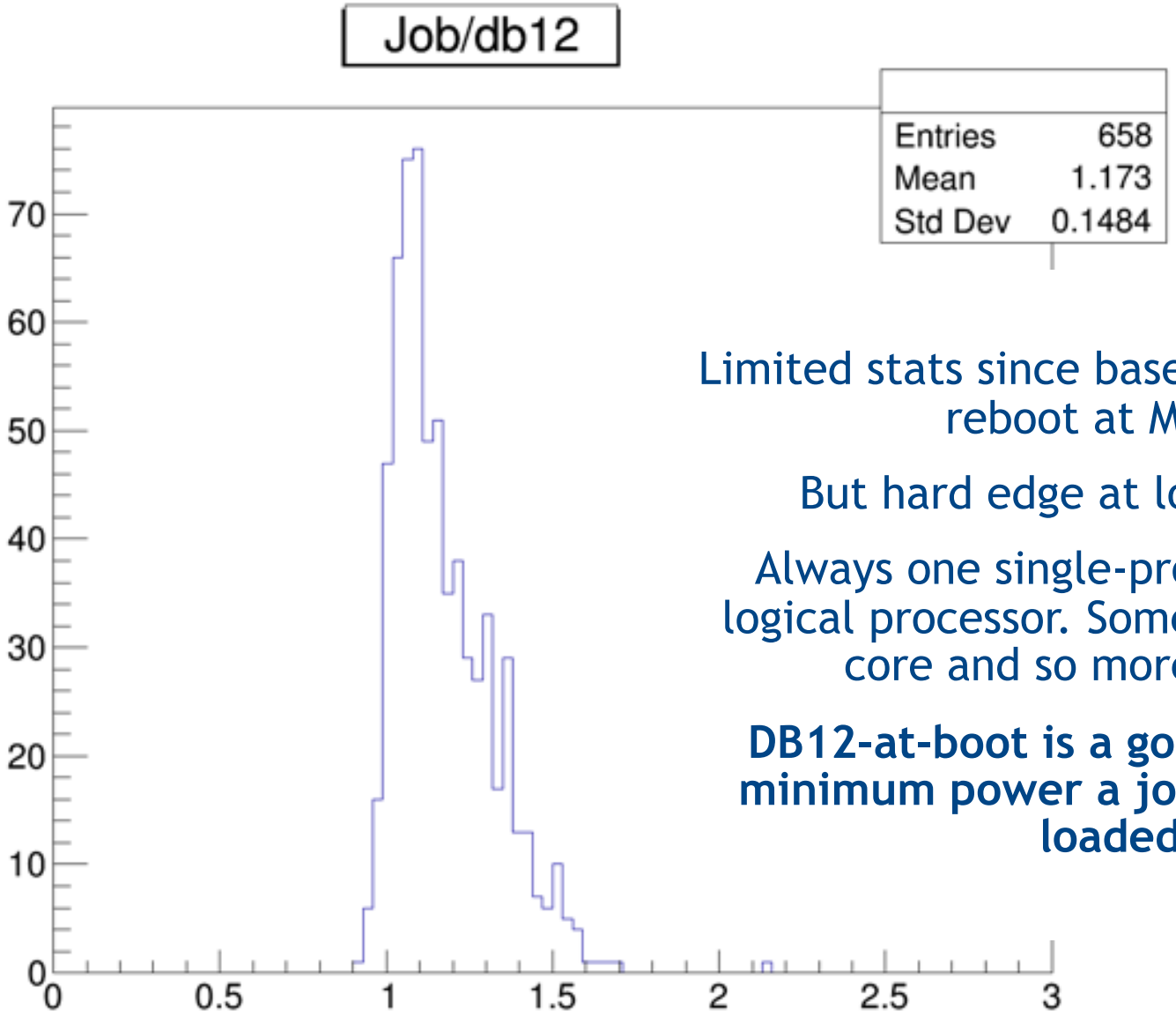



Normalisation sets peak to ~1.0

Much better picture across the same set of sites and architectures.

Mean very close to peak. Very little in the 1.5 area and above.

# JobPower vs DB12-at-boot benchmark




Limited stats since based on last week's mass reboot at Manchester

But hard edge at low power is clear

Always one single-processor job slot per logical processor. Sometimes get the whole core and so more "Job Power".

**DB12-at-boot is a good predictor of the minimum power a job will get on a fully loaded host**

# HS06 vs DB12-in-job vs DB12-at-boot

- HS06 is clearly subject to architectural complexities
  - Not a good predictor for LHCb Monte Carlo
  - Not as bad for LHCb reconstruction jobs though
- Both DB12 variants are much better than HS06 for LHCb MC
  - There are arguments for having both DB12 scenarios
- DB12-in-job at start of pilot/payload gives a true picture of what is really available to the job right now
  - However, other load on the machine could increase and you'll then get less power
- DB12-at-boot in MJF gives you the worst case that the site says it will be able to supply to the job
  - This is difficult for jobs to discover, even using results stored in a central database by previous jobs

# Conclusions

- DB12 is an Open Source fast benchmark that is simple for projects to use
  - Can be run within jobs for site validation, monitoring, “job planning”
- LHCb is happy with the DB12 benchmark
- Simple and portable, and better predictor for Monte Carlo events /second than HS06
- **We would like to see the DIRAC benchmark included in any WLCG fast benchmarking recommendations to experiments**
- DB12-at-boot makes it easy for sites to provide worst-case fully-loaded benchmark to jobs via MJF
  - Preliminary results indicate +/-2% variation between equivalent machines, for a benchmark taking ~2 minutes
  - Avoids need for per-site or central databases of fast benchmark results
    - LHCb will be happy to publish our per-machine DB12 measurements from DIRAC accounting if WLCG wants this though
  - Can also be used to identify misconfigured machines