Computing and software for the High Luminosity LHC

James Catmore[a,*]

[a] *Department of Physics, University of Oslo*
Fysikkbygningen, Sem Sælands vei 24, 0371 Oslo, Norway

E-mail: james.catmore@cern.ch

The success of the High Energy Physics community in addressing the challenge of LHC computing and software is borne out by the plethora of beautiful physics results shown at the ICHEP 2020 conference, and in dozens of similar meetings since 2010. Underpinning all of these results is the computing power and storage resources provided by the Worldwide LHC Computing Grid and the expertise embedded in millions of lines of software. The HL-LHC era poses two formidable challenges to those working on HL-LHC computing and software: the very significant increase in event rate and complexity from the LHC; and the increased heterogeneity of commercial computing hardware. The computing and data handling track of the ICHEP 2020 conference provided many interesting avenues of research for how these challenges can be addressed. They are summarised in these proceedings.

40th International Conference on High Energy physics - ICHEP2020
July 28 - August 6, 2020
Prague, Czech Republic (virtual meeting)

*Speaker

1. Introduction

The Large Hadron Collider (LHC) set a formidable problem for those working on the software and computing aspects of the project. A distributed computing model was devised, spread across potentially hundreds of data centres in universities and national laboratories, but with common fabric for data and job management. Such a model is known as a *computing grid* and so the resulting system became known colloquially in the community as *the Grid*, under the umbrella of the Worldwide LHC Computing Grid (WLCG) Collaboration. At the same time the community took a giant leap from procedural FORTRAN software to object oriented C++, developing new data processing frameworks from scratch.

In 2020 the status of the computing and software for the LHC can be summarised as follows. In terms of *computation*, the Grid provides more than 6 billion HEPSPEC06 (HS06) [1] hours per months for the LHC community, mostly via Intel or AMD multi-core CPUs with x86 instruction sets. More than an exabyte of *storage* capacity on tape (for long term archival) and disk (for regularly accessed data) are deployed. With respect to *network*, CERN is connected to each of the major Grid sites around the world on a dedicated, private, high-bandwidth network called the *LHC Optical Private Network* (LHCOPN) [2], such that links can sustain between 10-100 gigabits/second, leading to average data movement of 50 GB/second and around 50 million individual files per week. There is a complex patchwork of *software* in use in the HEP community. The experiments have dedicated frameworks written in C++ and configured in Python. They rely on many external packages from within and outside the HEP field. This adds up to many millions of lines of code, generally written for x86 architectures. Originally these frameworks were single threaded but are increasingly multi-threaded. Analysis software is as varied as the user community, but there is a strong movement towards the Python ecosystem and particularly interactive notebooks. The ROOT I/O layer and data analysis framework [3] underpins much of the software used in HEP. In recent years a very welcome organisational innovation was introduced in the form of the HEP Software Foundation (HSF) where ideas can be exchanged and commonalities and synergies identified and acted on.

The beautiful results presented by the HEP community at this ICHEP2020 conference, as well as in dozens of other conferences and workshops since 2010, are a testament to the success of LHC-era computing and software, since all of those results are underpinned by the computing power and storage capacity that the Grid provides, and by the expertise embedded in the software.

The High Luminosity LHC (HL-LHC) provides an additional set of demands on software and computing which are as daunting as the original challenge of the LHC, especially given that funding for computing and software will be at best sustained but not increased. They can be broken down into five general and related headings.

Computation: the unprecedented volume and complexity of the HL-LHC data, and more importantly the accompanying simulation, will have to be addressed in a computing environment where the speed of single-threaded execution no longer increases with newer models of processor, and technological improvements can only be harnessed through *concurrency*.

Portability: the challenge of computation may be partly addressed through the use of *accelerators* such as general purpose graphics processing units (GP-GPUs), but this is contingent on significant re-coding to use such technology efficiently, and given the range of hardware types on the market it is implausible that large parts of HEP software could be re-implemented for each one, pointing to the need for a common *portability library* such that the same software can be executed on a range of products.

Facilities: resources provided to the Grid are generally dedicated clusters or parts of clusters in universities and labs, but in recent years there has been a trend towards funding agencies providing their resources in the form of *High Performance Computers* (HPCs). Thanks to work by the experiments, many HPCs have been seamlessly integrated into the Grid, but some are difficult to use due to special access protocols or because they are largely composed of accelerators. Integrating such "difficult HPCs" and making efficient use of them is likely to become increasingly important in the coming years.

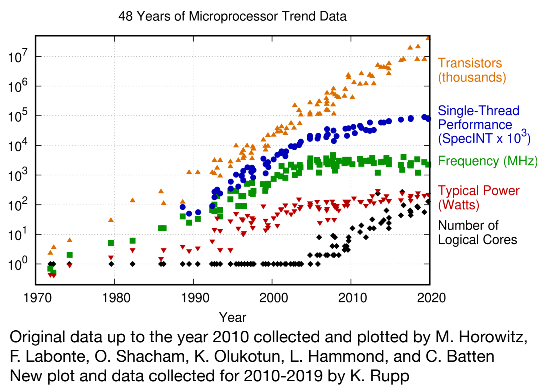
Storage: the cost per PB of disk storage has ceased to decline, and yet the demands of HL-LHC will be ten times greater than the LHC. Technology is unlikely to help here as there is no equivalent of GPUs for storage, so the problem must be solved with more efficient use of space (smaller and fewer data formats) and more use of tape for long-term storage, with expensive disk resources being used only as a cache for active data processing.

Analysis and data delivery: the data and simulated events must be made available to the user community at sufficient speed to allow analysis to proceed in a reasonable time frame, addressing the expectations of researchers that access will be possible via modern platforms such as interactive notebooks. This again points to smaller, flatter analysis formats, and also to using new data delivery technologies.

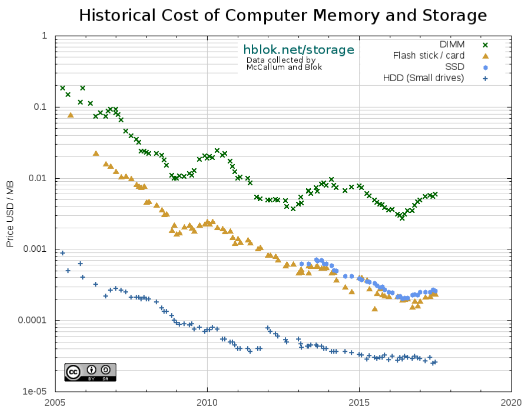
The success of the HL-LHC will require a great deal of innovation from those working on software and computing, involving all parts of the data processing chain from event generation and simulation to analysis by users. Computing models will have to evolve to push more data to cheaper storage media. Those writing software may need to learn new skills to adapt to greater levels of concurrency and hardware types. Fortunately the HEP community is highly innovative, and many individuals and bodies are pursuing the goal of sustainable computing in the HL-LHC era. The computing and data handling track at the ICHEP 2020 conference showcased many such examples, and these are summarised in this report. It is arranged in sections according to the headings above, with an additional section dedicated to machine learning and artificial intelligence. After a brief summary the relevant contributions from ICHEP2020 are highlighted.

2. The computation and portability challenge

Since the early 1970s, the number of transistors available for computation on an average processor has doubled approximately every two years. This trend, known as Moore's Law, continues to this day, as can be seen from the orange points on Figure 1(a). However, since the early 2010s the performance of a single core executing a single instruction thread no longer increases with new hardware, whereas previously it rose alongside the number of transistors (blue points). The reason for this can be seen clearly from the red points - the high density of transistors has limited the heat that can be dissipated and thus the power that can be consumed by the processor. Consequently the clock speed (green) and the single-threaded performance no longer increases as it did previously. In order to profit from the still increasing availability of transistors, it is necessary to use *concurrency*, such that multiple logical cores are put to work on the same problem, in multiple threads of instructions (black points).



(a) Processor performance since 1971 [4]



(b) Storage costs since 2005 [17]

Figure 1: Computational and storage technology trends

HL-LHC will produce events at up to ten times the current rates, with each event having up to 200 *pile-up* interactions per bunch crossing (as opposed to 30-50 now). In order to cope with the demands of this data and the accompanying simulation, the experiments must optimise existing workflows, and also make significant use of concurrency. Progress has been made in both regards. In terms of concurrency, multi-threading is now common in LHC experiments' software frameworks, allowing additional cores to be brought to bear on a given computational problem with minimal additional memory requirements. The 'next level' of concurrency would include use of GP-GPUs, which provide very large numbers of relatively weak (in terms of single threaded performance) cores, successful utilisation of which could yield performance improvements of several orders of magnitude. Such extreme concurrency is not applied straightforwardly to any problem, and some aspects of high energy physics computing may not be well suited to it. Despite the work to harness GP-GPUs being in its early days in the HEP community, several presentations at ICHEP2020 addressed the topic directly and demonstrated significant progress, alongside efforts to re-think workflows in general for the HL-LHC era.

2.1 Preparation of simulated events

Although the huge data rates expected from the HL-LHC are often highlighted, this is not expected to be the main consumer of CPU as Figure 2 from ATLAS indicates. Instead, the experiment expects that, assuming aggressive savings are made, by 2028 almost three quarters of its compute resources will be occupied in simulating the initial production of particles in the collisions (event generation), their subsequent interaction with the detector and conversion into digital signals (detector simulation and digitisation), and finally their reconstruction into analysis data. Other LHC experiments have made similar estimates. Reflecting the importance of minimising the compute impact of simulated event production, several talks at ICHEP2020 addressed this topic.

2.1.1 Event generation

Event generation is currently a relatively small part of the total CPU consumption of the LHC experiments. In the HL-LHC era the statistical uncertainty for many analyses will be significantly

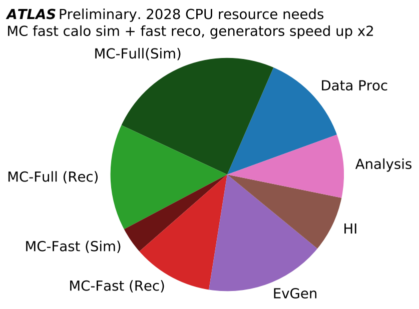


Figure 2: Breakdown of estimated compute workloads in 2028 for ATLAS [5]

reduced, exposing Standard Model background estimation as a major source of remaining uncertainty. This will require increasingly precise background modelling, including at higher orders of perturbative calculations, and will lead to greater demand on compute resources. Fortunately, event generation is likely to be amenable to parallelism on (e.g.) GP-GPUs. This is partly due to the nature of the calculations involved (including Monte Carlo integration), and also because event generators tend to be smaller standalone software packages that can be more easily profiled and migrated than the enormous software stacks maintained by the experiments.

At ICHEP2020, two contributions gave a flavour of the impact that GPUs might have on event generation. Rather than directly re-coding the software, both used the same interesting approach of re-implementing relevant parts of the event generation chain into Google's TensorFlow library [6]. This is a tool-kit for numerical computation using data flow graphs, and is mainly designed for building deep learning applications, but can also be deployed for other kinds of problem. It is engineered to run transparently on GPUs. The two tools presented - *VegasFlow* [7] and *PDFFlow* [8] - implement respectively the evaluation of high dimensional integrals based on MC algorithms, and PDF interpolation. Impressive speed ups (of an order of magnitude for LO calculations and 2-3 times for NLO) when run on a variety of GPUs were presented. The authors stressed that these results were without optimisation and migration of other parts of the workflow, so further improvements can be expected. As a proof of concept these are very encouraging results and it is hoped that further work along these lines will be done both by these authors and others.

2.1.2 Detector simulation

Detector simulation will be the largest consumer of compute resources for most of the experiments in the HL-LHC era. A general review of this topic was presented at ICHEP2020 [9]. Three tracks for improvement in the compute performance of detector simulation were identified: refactoring and internal improvements to the GEANT4 [10] toolkit, which is the cornerstone of detector simulation for the LHC experiments, particularly for high fidelity simulations; use of computational accelerators such as GPUs; and greater use of *fast simulation* which replaces detailed particle tracking methods with more approximate (and lower fidelity) techniques such as parameteric approaches or generative deep learning models.

On the first of these tracks, there is reason to be optimistic; for example, the GeantV project [11], which aimed to exploit vectorisation in particle transport in G4, identified significant savings by deploying better data caching, better instruction cache use (simpler code with fewer classes, indirections and code branchings), more compact code, better data layouts and fewer virtual function calls. The conclusion was that, although the gains from vectorisation were smaller than hoped, the associated refactoring, rewriting and modernisation of parts of G4 could yield a speed-up of several 10s of percent, which would have a very considerable impact on HL-LHC computing.

The use of computational accelerators, identified as the second track for sustainable simulation, is perhaps less encouraging. Unlike event generation, detector simulation is not a natural candidate for running on GPUs, due to the inherently ‘branchy’ nature of particle transport algorithms. Nevertheless research is going into several aspects, including the use of ray-tracing techniques.

The third track - using faster algorithms that replace the detailed particle transport algorithms of G4 with more approximate methods, are largely under the domain of the experiments. The LHCb Collaboration gave two contributions on this topic at ICHEP2020, on general fast simulation techniques [12] and those specifically related to calorimetry simulation [13]. LHCb identified a number of features that are common to most of the LHC experiments, including the use of mixed workflows, where certain parts of the detector which require very high fidelity are handled by G4, with other sub-detectors dealt with by faster more approximate methods. In principle one could go further, simulating particles from specific decays with full simulation, and handling less important states with fast techniques, but it must be borne in mind that simulated data of any kind requires scaling factors to be produced to ensure good matching with real data, and this becomes increasingly complex as each new flavour of simulation is added to the menu.

The second LHCb contributor discussed the use of generative models driven by deep learning techniques to simulate calorimeter response. This is again common amongst the LHC experiments, and is perhaps the most advanced and encouraging application of ‘deep’ machine learning in particle physics. Two neural network architectures have come to the fore in this regard - Generative-Adversarial Networks (GANs) and Variational Autoencoders (VAEs). They hold the promise of not only matching traditional parametrised methods in terms of computing performance, but also outperforming them in terms of physics fidelity. The training of the models can also be run on GPUs. Although work on these techniques is still at a relatively early stage, progress is such that they may see some use even in Run 3.

2.2 HL-LHC data processing

Several contributions at ICHEP2020 addressed the processing of real data. Unlike event generation and detector simulation, which are mainly impacted by the number of simulated events which must be created to match the real data, the compute resources occupied by data processing also increase with pile-up. Given that the pile-up may be as high as 200, steps to minimise the impact on the reconstruction time are of great importance for HL-LHC computing.

Approaches to addressing this problem include re-factoring both reconstruction workflows and software (see e.g. [14]) and deploying accelerators. Speakers from the ALICE Collaboration gave two presentations, on data processing in general [15] and on the use of accelerators in particular [16]. In a sense the challenges that the other experiments will face in Run 4 arrive for ALICE in Run 3, as significant upgrades will allow the experiment to collect data at 50kHz without a high level trigger, 100 times greater than in Runs 1 and 2. This will require online reconstruction (*synchronous* processing) during which larger components of the raw data, especially output from the time projection chambers (TPCs) will be compressed by a factor of up to 35. Use of GPUs for this step is mandatory, and ALICE has demonstrated very significant progress in this regard. The

ALICE O^2 data processing framework has been developed to allow offloading to GPUs, and the entire tracking software for the TPCs can run on them. Studies to ensure numerical consistency have been completed, and portability between hardware types (CPUs, NVidia and AMD GPUs) has been achieved via a single transparent interface, requiring only one code-base, with dynamic loading of the underlying libraries ensuring the framework can switch to the correct architecture. ALICE reports that for the TPC reconstruction a single graphics card can replace between 40 and 150 CPU cores, with an order of magnitude speed-up for TPC tracking. The eventual objective is to run all of the track reconstruction in the detector barrel on GPUs, and also use them for elements of the *asynchronous* reconstruction, which is executed later during shut-downs or pp runs.

3. The storage challenge

Storage for HL-LHC is a bigger challenge even than compute, because there is no equivalent of accelerators for storage, and opportunistic storage resources at commercial cloud centres are expensive. The cost of disk storage has ceased to decline in recent years as can be seen from Figure 1(b), and although tape is much cheaper and has more room for technical improvements than disk, there are few manufacturers and so the downward pressure on prices due to competition is less pronounced. Consequently, in order to handle the expected factor of ten increase in data and MC under a flat resources budget, the LHC experiments must “do more with less", that is, produce fewer and smaller data formats, and perhaps produce less Monte Carlo through measures such as minimising negative weights in generated events. Much of this work will rely on the resourcefulness of physics analysts to work with fewer variables whilst not sacrificing the quality or precision of the results.

Several contributions at ICHEP2020 addressed this challenge, notably the speaker from ATLAS [18] who introduced the *data carousel* concept, where expensive disk resources are used largely as an operational cache for ongoing production, with tape being the primary storage medium for all types of data. This mechanism requires very close collaboration between the central operations teams at the experiments and the data centres providing the tape resources, and also needs a flexible and sophisticated data management system such as Rucio [19], which was originally developed by ATLAS and is now used by several experiments, including by the Dune Collaboration who reported on its adoption at ICHEP2020 [20]. The ATLAS speaker also reported on the experiment’s new analysis model, which includes far fewer and much more compact analysis formats than is currently the case, thus treading a path already taken by CMS.

4. The analysis and data delivery challenge

For many particle physicists the analysis step is where their own software is used, tailored to their specific physics objectives. It is where final calibrations must be applied and resulting experimental systematic uncertainties evaluated, and where data and simulated events must be brought into final agreement through often complex re-weighting schemes. Consequently analysis software is a patchwork of software, tools and data formats, with wide varieties of approaches even within the same experiment. Nevertheless there remain some commonalities. Perhaps the most obvious of these is the ROOT framework, the authors of which gave a detailed presentation at

ICHEP2020 [25]. They focused on the upcoming ROOT 7 release, which represents a fundamental overhaul of many aspects of the framework, including the all-important I/O layer where the familiar TTree will be replaced with a new structure called *RNTuple*. The speaker discussed the impressive performance of this new data structure, and its connection to another innovation, the *RDataFrame* interface, which will see analysis code move from traditional explicit loops over events and user-implemented reading/writing of data to a more declarative style. In the same contribution there was also a report on updates to the other aspects of ROOT: graphics, histograms, parallelism and Python bindings. The speaker remarked that, although ROOT now faces "competition" in the form of the wider Python-based scientific ecosystem, there are good reasons to bet on ROOT, given the centrality of its I/O system in the experiments' frameworks and data formats, the highly competitive performance of its data structures, the fact that it is built with HEP workflows in mind, and that in any case, it can easily interface with tools from the wider ecosystem, including those delivering machine learning capabilities. A further contribution [26] addressed upgrades to the RooFit statistical analysis suite that is closely associated with ROOT, highlighting in particular adaptations for running on GPUs.

The question of high-speed delivery of analysis data to users was not addressed in detail at ICHEP2020, but is a topic of intense interest, since the movement to more declarative and interactive analysis (increasingly via Python notebooks) clearly requires a different approach to the batch-processing workflows familiar to most analysts today. Several major projects (e.g. [27]) are committing significant resources to this problem. The adoption of very compact analysis formats is an obvious pre-requisite for on-demand data delivery to become a reality.

5. The role of machine learning and artificial intelligence

The use of machine learning (ML) is well established in HEP, with several LEP- and Tevatron-era measurements and discoveries having been assisted by neural networks and boosted decision trees (BDTs). However, the capabilities of deep neural networks, made possible by modern computing power and fundamental developments in the field of artificial intelligence, are new and of great interest in HEP. For the moment, the main application remains physics analysis, where deep networks can be highly effective at discriminating minute signals from large backgrounds, given sufficient training data. Additionally, machine learning is finding increasing relevance in jet reconstruction and B-jet tagging, and as described earlier, as a means of fast simulation.

Several contributions at ICHEP2020 discussed the opportunities provided by advanced machine learning. Of particular interest was a talk [21] on ranking the importance of input features (variables) in influencing the decisions made by the algorithm. Such rankings enable analysts to cast light inside the so-called "black box" and provide an explanation as to why an algorithm behaved as it did, improving reproducibility and confidence in the results. Another contribution [22] gave a primer on techniques that can be used to improve the performance of deep neural networks. A third talk [23] showcased a proof-of-concept study on quantum machine learning in HEP applications.

One area where machine learning has yet to make much of an impact (other than in very limited applications such as pixel clustering) is track reconstruction. To address this, an open challenge (the TrackML Challenge) was set for the wider community on the Kaggle and Codalab platforms to try to beat established HEP track-finding software in terms of accuracy (first phase) and speed/accuracy (second phase). The conclusions of the challenge were presented at ICHEP2020 [24]. There was widespread interest in the challenge from competitors both within and outsider of HEP. None of the solutions were able to out-perform HEP code for both speed and accuracy, and the highest scoring individuals were often from within the field, indicating that domain-specific knowledge helps. Nevertheless, some of the new techniques will be worth following up in more detail. The dataset used in the challenge, and the hypothetical detector layout on which it was based, have proved to be valuable in other R&D applications.

It is clear that ML will have a growing impact on the field. Whilst there is unlikely to be a time in the foreseeable future when “all tracking" or “all simulation" is done entirely by ML techniques, ML elements will be added to an increasing range of applications, and this process is well under way.

6. Conclusions

In 33 excellent contributions, the computing and data handling track at ICHEP2020 highlighted a small part of the work under way across the HEP community to ensure that the needs of HL-LHC can be met. It is clear that the community has the skills, imagination and drive to successfully address HL-LHC computing - as long as the relevant personnel are able to remain in the field. It is essential that funding agencies and institutes understand that computing and software is as important for physics as detector development and construction. The days when software grew organically with the detectors are over - writing software and building computing systems for HEP now requires detailed project planning and management, and significant person power sustained over many years. Stable career paths need to be defined for those who wish to stay in HEP and work on computing. If this can be achieved, there is little doubt that the HEP community will pass the test of HL-LHC computing and lay the foundations for decades of beautiful new results and inspiring discoveries.

References

[1] M. Michelotto et al., J. Phys. Conf. Ser. **219**, 052009 (2010)

[2] E Martelli and S Stancu, J. Phys. Conf. Ser. **664** 052025 (2015)

[3] R. Brun, F .Rademakers, Nuclear Instruments and Methods in Physics Research A **389** 81-86 (1997)

[4] M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten, K. Rupp, *48 years of microprocessor trend data*, https://github.com/karlrupp/microprocessor-trend-data

[5] The ATLAS Collaboration, *ATLAS HL-LHC Computing Conceptual Design Report*, CERN-LHCC-2020-015 ; LHCC-G-178 (2020)

[6] M. Abadi et al., *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*, https://www.tensorflow.org/about/bib (2015)

[7] J. M. Cruz-Martinez, *VegasFlow: accelerating Monte Carlo simulation across platforms using TensorFlow*, talk at ICHEP2020,

[8] M. Rossi, *PDFflow: hardware accelerating parton density access*, talk at ICHEP2020

[9] G. Stewart, *Detector Simulation Upgrades for HL-LHC*, talk at ICHEP2020

[10] Nuclear Instruments and Methods in Physics Research A**835** 186-225 (2016)

[11] G. Amadio et al., *GeantV: Results from the prototype of concurrent vector particle transport simulation in HEP*, https://arxiv.org/abs/2005.00949 (2020)

[12] A. Morris, *Fast simulations in LHCb*, talk at ICHEP2020

[13] M. Rama, *Fast calorimeter simulation at LHCb*, talk at ICHEP2020

[14] The ATLAS Collaboration, *Fast Track Reconstruction for HL-LHC*, ATL-PHYS-PUB-2019-041 (2019)

[15] C. Zampolli, *ALICE data processing for Run 3 and Run 4 at the LHC*, talk at ICHEP2020

[16] M. Concas, *GPU-based online-offline reconstruction in ALICE for LHC Run 3*, talk at ICHEP2020

[17] J. C. McCallum and hblok.net, *Historical Cost of Computer Memory and Storage*, retrieved from https://hblok.net/blog/storage

[18] M. Svatoš, *Providing the computing and data to the physicists: Overview of the ATLAS distributed computing system*, talk at ICHEP2020

[19] Barisits, M., Beermann, T., Berghaus, F. et al. Comput Softw Big Sci **3**: 11 (2019)

[20] S. Timm, *DUNE Data Management Experience with Rucio*, talk at ICHEP2020

[21] A. Di Luca, *Automated selection of particle-jet features for data analysis inHigh Energy Physics experiments*, talk at ICHEP2020

[22] G. Strong, *On the impact of modern deep-learning techniques to the performance and time-requirements of classification models in experimental high-energy physics*, talk at ICHEP2020

[23] C. Zhou, *Application of Quantum Machine Learning to High Energy Physics Analysis at LHC using IBM Quantum Computer Simulators and IBM Quantum Computer Hardware*, talk at ICHEP2020

[24] A. Salzburger, *Conclusions from TrackML the HEP Tracking Machine Learning challenge*, talk at ICHEP2020

[25] A. Naumann, *Hello RNTuple and friends: what the new ROOT means for your analysis*, talk at ICHEP2020

[26] S. Hageboeck, *What the new RooFit can do for your analysis*, talk at ICHEP2020

[27] https://iris-hep.org