Workshop Notebook

HSF Virtual Workshop, November 2020

The notebook here is for comments, discussions as well as any issues that should be followed up after the workshop. There are assigned note-takers for each session, but anyone is welcome to add their comments and observations.

As slides will be posted in advance, feel free to also write questions and discuss matters here before the presentation begins (questions will then be read by the chair).

- Keep comments in the right section for each talk
- Please don't delete anything anyone else wrote
- Please identify yourself so we know who made the comment
    - We suggest your name in []s, e.g. [Graeme S]

Day 1 - Thursday 19 November, HSF General Sessions

Introduction

- This is a sample comment [Graeme S]

Future Trends in Nuclear Physics SW and Computing

- One area where we could probably cooperate a lot is in training for software skills. How does that kind of training happen now in the NP community? [Graeme S]
    - A. Not everyone feels confident they have the right skills (can be improved); Most people come with Python training only; Documentation is poor!; No

coordinated plan at the moment - small groups; Would definitely benefit from more structure.

- In connection with the next talk - what's the uptake of Python? Interest in our topical meetings next year, for example …? Happy to connect better. [Eduardo R]
    - A. Have had presentations on Python in the roundtable. ROOT and Python are the top common tools - theorists particularly use Python.
- NP in HSF: the question seems still open, is there anything we can do on our side to make it easier? As it is clearly part of our identity to be open to others... (Michel J.)
    - A. Software and Computing Roundtable has been very helpful as a way to get ideas into the community. HSF invited to join the organisation now. HSF talk in the next one. This should help make connections. Please think about convenors from the nuclear physics community - we also have to get more involved from our side.
- How disruptive will EIC be given its size? (Doug Benjamin)
    - A. Yes, it is going to change things a lot as things scale up. Forum should help with this culture change.

PyHEP

- It's obviously a difficult thing to quantify, but do you have any estimate of how many published analyses have been produced primarily in python (i.e. more than just a few pyROOT scripts processing the output of C++ code)? Ditto Jupyter. [TJ Khoo]
    - A. We know of a few examples - at least a handful. Want to survey this to get a better idea in the future. Would like to examine citations to get an idea, but unfortunately citing is quite poor for software.
      A notable example from LHCb (active PyHEP collaborator): https://infoscience.epfl.ch/record/279779.
    - Follow-up Q: should we push for better citing in HEP papers?
    - A. Yes, have presented this point to LHCb at least (see https://hepsoftwarefoundation.org/assets/EduardoRodrigues-LHCb-2019-01-15.pdf taken from https://hepsoftwarefoundation.org/organization/presentations.html). We should continue to agitate on this point and encourage better practice.
- Astronomers have a famous plot of "number of programming language mentions in published papers," which shows a dramatic rise of Python at the expense of IDL and Fortran. However, do we *mention* the programming language used in our papers? [J Pivarski]
  Counting programming language mentions in astronomy papers
    - Tendency in my experience is no -- at most we cite generators, cross-section calculators, and the odd code for stats etc (typically "pheno" papers written by collaborators), but the language is not stated.
- Given how successful it was, are you going to run PyHEP virtual next year? [Paolo C]
    - A. Yes - there will be another virtual PyHEP next year. Want to keep the relationship/co-location with SciPy as this was really successful.
- Is there a picture of how many Python packages across HEP are written in pure Python vs via bindings to C/C++ etc? [Ben M]

    - A. Roughly ⅔ of the HEP packages we know are pure python for sure; either are bindings, e.g. boost-histogram binding the C++ Boost.Histogram.
- On a similar note, what's the balance between uses cases: e.g. use in Analysis vs other areas (DAQ/Simulation/Other). How to improve uptake in the latter, if appropriate? [Ben M]
    - A. Dirac is pure Python (as is Ganga, and Rucio). Of course it's not appropriate for everything. Xenon1T does have a full Python stack, including DAQ (but this is rare!).
- 1000 registered, but 200 showed up - do we have the right people showing up?
    - A. Did get 400 in the first session. Hard to say why people are not coming to the whole (or any?) of the workshop. No registration fees meant no barrier to entry, but also then low commitment… Some people did apologise for registering but not attending. Noted that only a few people complained about not being able to register, but no idea who could not and said nothing. Did think about a small registration fee, but not clear this would actually help. Next year will also try to improve technical solutions to avoid any hard limits at all (e.g. webcast - was used by CERN School of Computing; one can also livestream to YouTube)
    - I was one of those who wasn't able to register. [scott s]
    - If you are concerned about people flaking out, a proposal is in the Pay to Learn portion of this - https://docs.google.com/document/d/1BcI0jS_SWsQdeYZAt02ciIDtW-ATT4WapOhuIPx2B7M/edit#heading=h.9hdoursdncay you can ask for a registration fee, and if they attend you reimburse them their fee *plus* some additional amount to reward them for their attention/time [Sam Meehan]
        - Noted that no one is paying for travel at the moment

CERN R&D on Spack and the Turnkey Stack

- It's planned to possibly include glibc in build. How does spack currently pull in the right compiler? Does spack set this up? [Attila K]
    - Skipped this step because it happens automatically, but spack has a command/option to point spack to the compiler.
        - Thought spack would be able to pre-build packages, and have spack link the binaries.
    - Spack buildcache allows it to identify that a binary is the same one as what would be compiled locally, so a tarball of a package in one place could be picked up elsewhere. However, relocation is not quite all there.
        - Spack doesn't have a post-compilation step?
    - No, will be installed in prefix, and sets paths via rpaths. Philosophy of spack is not to relocate.
        - It can relocate as far as RPATHs go, it's an issue if a package hardcodes install-time paths (e.g. "/usr/local"). It is possible to rewrite these binaries (Conda can do so), but needs care and there are still limitations. I can't give a concrete example in HEP, but imagine a package installs a data file in /etc/package.data and encodes that path in the binary for runtime lookup. You cannot then move the package to a new prefix and have it work. [Ben M]

            - I'd say this is a buggy package too! [Graeme S]
                - I agree, but there is the very real requirement to install these buggy packages… I think Whizard is an example which can not be relocated, despite efforts from spi. [ValentinVolkl]
            - For full info, we discussed this in a Software and Packaging Tools meeting here: https://indico.cern.ch/event/848215/#12-best-practices-for-package [Ben M]
        - Majority of neutrino experiments at Fermilab absolutely require relocatability -- many HEP packages are already relocatable [Kyle Knoepfel]
            - I agree that it is nice to have relocatability for other applications, but if you are using spack, do you really need to relocate? I think we are fine without. [ValentinVolkl]
        - I thought that the relocation was a combination of: change RPATHs to new location; scan configuration files for old path, update to new; change any environment variables that are needed, old to new. I'd be definitely interested to know when and why that fails [Graeme S]
            - Me too! [Ben M]
            - As Ben mentions, a lot of errors are connected to paths hardcoded in the binaries. For the RPATHs, spack uses patchelf, which can overwrite and also grow the paths. Spack can also change installation paths in directories, but it can only shrink them. - this leads to very frustrating failure modes, when the success of the relocation depends on whether your original install path is longer than the new one. [ValentinVolkl]
            - Adding to ValentinVolkl: We have mostly dropped "buildcache" from our agenda for a while, because it's a headache to debug all this. Having fortran code that hardcodes paths in static variables that are not sized appropriately is really a challenge to debug [Christian Tacke]
            - At least for building binary packages, is it possible to build/install/package up using a sufficiently long path to cover the majority of use cases? [Ben M]
- Along similar lines, is there work to make spack more usable for HEP use case where resources are limited, and users don't have full control over the machine and want to maximise reuse of what is already installed [KevinPedro].
    - Developers are aware, and understand that concretiser should try to reuse packages, but spack is somewhat picky about dependencies. On the roadmap to deal with in next version, already fixed.
        - New concretiser in 0.16 loves already installed stuff (at least from packages.yaml) [Christian Tacke]
- Are we the only ones having a lot of issues about LD_LIBRARY_PATH? Like spack compiling ncurses, adding ncurses/lib to LD_LIBRARY_PATH and then suddenly bash or aptitude not working? Are people working on this? [Christian Tacke]
    - No, I see a lot of warnings when using spack installed ncurses, along the lines of /path/to/libtinfo.so.6: no version information available. Not sure if that

is related though. (Since recently also for libcurl) But I haven't really experienced any crashes, only the (annoying) warnings. [Thomas Madlener]
    - Exactly that. [Christian Tacke]
    - Actually I haven't seen those in a while (I used to in my ubuntu setup), I'm just trying to figure out how I got rid of those! [ValentinVolkl]
    - The reasoning is quite straight forward: The system ncurses and spack ncurses are not compatible. And using LD_LIBRARY_PATH makes systems binaries (like bash or aptitude) load the wrong ncurses. Which finally let's this break [Christian Tacke]
    - They can still be compatible (even with the warnings), but the spack installed libraries do not set a value in the elf file. [Thomas Madlener]
        - The problem is that spack does not seem to set the VERDEF field in the binary, but e.g. system bash tries to read that via VERNEED, see https://stackoverflow.com/a/38851073
    - I think you could try to install the same version of ncurses as the system (But Thomas is right, you might still see the warning about the versions). Or what definitely works, would be to install bash or whatever fails with spack :) [Valentin Volkl]
    - We're partly going two ways currently: 1) Let spack use ncurses&co from the system (using packages.yaml) 2) Try to remove LD_LIBRARY_PATH setting as much as possible. [Christian Tacke]
    - We should really try to move away from requiring LD_LIBRARY_PATH and configure packages using relative RPATHs instead. I recommend everyone to read http://xahlee.info/UnixResource_dir/_/ldpath.html for further info on why LD_LIBRARY_PATH is not a good solution. [Guilherme A.]
    - For packages installed into CVMFS, we know ahead of time the destination, so relocation can be avoided if they are configured initially with the final installation path. Then a staged installation can be used to create a tarball that can be extracted by the publisher. (See link at https://www.gnu.org/prep/standards/html_node/DESTDIR.html). [Guilherme A.]
- There are several problems that I have not seen being addressed by spack. One problem is the need to have full stack builds for each Linux distribution and each compiler. That wastes a lot of resources. It would be much better to have a base that works on all distributions in CVMFS and build HEP stacks on top of that. Same thing for debug vs release builds. It would probably be good to invest some time in getting split debug information to work for the release build, so that performance in production is not affected, while making debugging still possible when needed without requiring a separate stack (would make it easier to reproduce problems that need debugging as well). Finally, the stacks should be able to have both Python 2.x and 3.x available at the same time. Replicating full stack builds but only having different Python packages also wastes resources. Gentoo Prefix addresses most if not all of the problems mentioned above, but unfortunately it was kicked to the side and never looked into more seriously. However, it is being used by some people to create stacks with CVMFS: https://eessi.github.io/docs/compatibility_layer/ , https://www.eessi-hpc.org/ and also

https://hpckp.org/agenda/european-environment-for-scientific-software-eessi/ [Guilherme A.]
    - I didn't really know that Gentoo Prefix can run on so many platforms, that's very interesting! Apart from that I think a lot of the issues you list are outside of the responsibility of spack. The separate build for each linux platform can be avoided by building on top of Gentoo Prefix (I think I'll give that a try). And with the next versions spack is foreseen to be able to do something similar on its own, (by building its own glibc). I guess nothing saves you from building for different compilers. With Debug and release builds, I think that's the domain of the low level build tool to be able to provide debug information in separate files, then I see no problem for spack to install them. For packages with multiple versions you want to build against, like python 2 / 3, spack is very good - only the packages that depend on python are installed two times, all other dependencies are shared. While reducing the number of different builds needed for a release is of course a good concern, I think spack is also good at reducing the effort of making these different builds. [Valentin Volkl]
- Coming late to the party, sorry about that ;-) One naive question/proposal I would have, standing from an experiment-software developer point of view. I would love to see some kind of a central repository (not experiment specific), widely available, for _binaries_. Is there something already done in that respect ? Plans to do it ? [L. Aphecetche]
    - Relocation issues set aside, that would be a big time saver for (at least) trying to setup a (experiment-specific) software stack based on Spack. My use case here being to investigate whether Spack should/should not be used in a given experiment. Having to compile everything from scratch seems like a great deal of wasted time for such an exploratory phase, as many of the packages (thinking here about compilers, cmake, Root, etc..) are indeed shared by multiple stacks.
    - Yes, at key4hep we use the fairly general path /cvmfs/sw.hsf.org/spackages to deploy our packages (It includes ROOT and Geant4 and a good fraction of other packages in the lcg release, besides the experiment specific packages). It is perfectly possible to use this as an "upstream" spack installation (see https://spack.readthedocs.io/en/latest/chain.html) However, you probably need to use a spack version close to what was used to build it (the "key4hep" branch of https://github.com/key4hep/spack, which is 15.4 + patches at the moment). In the long run I think we'll try to have a spack build of the lcg releases which can be used in the same way. [ValentinVolkl]

Training - Activities

- Glad to see that you also had your recordings captioned. It's an important action towards being more inclusive. Out of curiosity, what tool did you use and who paid for it? [Eduardo R]
    - Used rev.com, paid for by IRIS-HEP
        - Same as used by PyHEP, paid for by python SW foundation
- Highlighted some lessons being reused from SW carpentry. What if some lessons are wrong/objectionable in the HEP context -- what do we do about this? [Kevin P]

    - Github is too basic, need a more substantial one. However, the SW carpentry ones are open-source, so we can improve them.
    - Have improved one and made open-source (based on CMS experience), to follow up.
    - CMS git tutorial public link: https://twiki.cern.ch/twiki/bin/view/CMSPublic/CMSGitTutorialPublic
- BTW, I can't agree with your conclusion that "Are we filling a niche that wasn't filled before? No". You definitely ARE filling a niche - far more training material than in the past. Great job! [Eduardo R]
- How many people actually follow the courses outside of the virtual training days, as opposed to postponing doing them because they are there? [Sebastian Lopienski]
    - Hard to say… we have statistics from YouTube, but haven't checked exactly.
        - From CERN computing school perspective, it's useful to ensure that people have to commit to attendance, as opposed to having the freedom, which they may then ignore.
            - From Binder, you can find out when people are running notebooks, so this lets you find out if people are following some of the lessons. [Jim P]
    - Want to avoid StackOverflow approach where they get solution to just one problem, but also don't want to hold a gun to somebody's head.
        - Can't skip the interactive sessions. [Henry S]
            - But doesn't work for the persistent lessons.
- Where do you get the <=5 mentors per student ratio? [Sebastian L]
    - Comes from Software Carpentry experience.
    - In my experience, Software Carpentry + C++ course, 5:1 wasn't really needed - in fact it was a bit hard to engage the students in a Zoom breakout room (which I thought might be improved with a little flash poll or something like that) [Graeme S]

Training - Community Building

- Virtual hackathon -- how do you see dealing with timezones? [David L]
    - Kick-off with discussion of module goals, but really just want to schedule time for collective work.
- Looking at the HSF training page, the YT videos are not visible. Should they be? [Michel J]
    - Not highlighted on the main page, but under the curriculum, one of the icons should point here.
- Thanks for highlighting motivation & obstacles. Have specific modules in mind, but can people bring their own ideas? [Sebastian L]
    - Have a curriculum that we'd like to flesh out, but anybody is free to use the hackathon as a way to carve out time to work.
    - If you come with your own idea, there may be a complementary idea which could come together to enhance yours. Example: set of modules for best practices to ensure good code quality before grid submission.
- Wondering whether interaction with universities has been considered? Presented as though university curriculum is lacking, and we are patching them, rather than improving what the universities offer. [Heather Gray]

    - Universities may benefit from having material that they can use in their courses, and correspondingly, we could benefit from what they already have. Could explore better collaboration.
    - There may be a Snowmass project to reach out to unis. Not clear how physics departments treat CS requirements - seems very non-uniform.
    - Since we have an HSF training programme, perhaps some can trickle down to physics programmes and be offered as credit. Some students have actually asked for this. At this time, we don't have that sort of clout, but it's something we aspire to. [Sudhir M]
    - Uni physics depts tend to see CS as non-significant, based on traditional attitudes, which are inconsistent with our viewpoint. Perhaps the seniors in HSF could push this more effectively in their own depts, where HSF itself may not be able to. [Kevin P]
        - UK: Midlands group has a collective school, and it would have been useful to bring in HSF material, had it existed at the time, which students would have received credit for -- so there really is an audience. It may be mostly a matter of marketing+sales. [Ben M]
            - May have to combat the fact that individuals may be paid to develop these courses, as opposed to just presenting them.
    - Be aware that different sub-disciplines of physics have varied computing needs (although they ~all have such needs). We have been discussing "upper level" computing for HEP, as opposed to basic computing skills. However, many depts are actively trying to fill in this gap. [Gordon W]
        - If we concentrate on simple things like basic python, there is plenty of material, so we should potentially focus on our community's specific needs.
- How to quantify the impact/success of your initiatives? [Eduardo R]
    - Roughly know how many people attend, might be able to estimate how much time they have been saved, but can't quantify what they have lost by not attending.
        - Also correlates with recognition + citation.
    - The WG is about training and careers, so some metric of success taking into account career progression rather than just attendance numbers would be useful.
        - Before/after surveys are short-term, while we should perhaps have longer-term follow-ups (e.g. ask what they applied from the training in their analyses 6 months later).
    - Many people being paid to understand how students are learning, and from this how to improve teaching. Would be useful to read the literature on this, before launching into our own efforts to quantify/improve. [Gordon W]
        - For now, relying on SW Carpentry, who have an established pedagogical model, if not necessarily a full set of metrics.

`Day 2 - Friday 20 November, Event Generators

Attendance: Max ~80

Introduction (Josh McFayden)

- James Catmore: possible to have a common event generation framework where common operations are factored out
    - Josh McFayden: Challenging but e.g. GPU optimized version of VEGAS could be made common but different generators have specifics/optimization etc. Generator authors can better answer.
    - Stefano Frixione: The short answer is no.

MC@NLO-Delta (Stefano Frixione)

- Josh McFayden: Giving up on higher precision should certainly be avoided, but do you have significant concerns about e.g. using resampling algorithms or reweighting (e.g.) LO+multileg to higher order calculations?
    - Stefano: Not giving up on precision but trying to keep up the same precision reducing the computing time. If reweighting works well for all distributions and processes then accuracy is not lost but proving this is not trivial. In any case, to prove this you need to simulate NLO.
    - Ben: even if you use resampling, you need a theoretical calculation that is as precise as possible
- Andrea Valassi: you say Delta does not change the formal properties of MC@NLO. Do you have any flexibility in choosing a different arbitrary Delta that suppresses negative weights even more, or are you forced to use the formula you showed?
    - Stefano: The 4 conditions on page .. should obey the properties on page 18. Within that class of functions for delta there are better and worse choices. The physical properties will change by some amount.
- Soumyadip Barman: is this already released? Will both the old MC@NLO and MC@NLO-Delta be present?
    - Stefano: it is in beta, it is difficult to say when it will be released. It also depends on new features in external dependencies, e.g. on Pythia.
    - Stefano: eventually both will be present, but this is tricky and is actually one of the reasons why the release is being delayed.

-- For the questions below, there was no time to discuss them --

- Josh McFayden: Ignoring for a moment the -ve weight events, could the difference between MC@NLO and MC@NLO-Delta be used as a matching uncertainty, rather than e.g. MC@NLO vs Powheg?
- Efe Yazgan: magnitude-wise the matching uncertainty seems large for ttbar. Do you know the approximate value for that?
- Josh McFayden: Are there CPU costs to the different folding schemes?
- Andrea Valassi: can you quantify the “disadvantage of longer running times”?

Neural Resamples (Ben Nachman)

- Andrea Valassi: slide 22, I am not convinced this should be treated as a binary classification problem: why did you choose cross entropy as a loss function? Have you considered using other optimization metrics that can be directly used for the

evaluation of physics precision (even if of course it is difficult to describe that in a single metric)? Could MSE or maximum mean discrepancy be better?
    - Ben: does not look like a classifier, but you can see this as a classifier. You could use MSE. Or also other metrics but would get complicated (or get the wrong answer).
- Stefano Frixione: one comment, cannot see how the positive resampling can be made general enough so that this will work for any observable
    - Ben: agree with that, also discussed by authors of positive resampling. (slightly more detail: the authors of positive resampling also talk about how you can see some improvement in variable X if you reweight on variable Y, although you have no universal guarantee that it will get better. This is a nice feature of the "full phase space" reweighting discussed at the end, where you reweight everything all at once and then you compute the observable later)
- Stefan: question on neural resampling, there must be some granularity. How can you understand this?
    - Ben: NN is a smooth function, with some dynamical interpolation/smoothing.
- Teng Jian Khoo: Can you expand on the uncertainty or sensitivity to fluctuations in the generated events? The "optimal K" setup seems to trade off efficiency in generating events versus potential fluctuations, where we've tended to want to avoid weights that are >1 for this reason.
    - Ben: the weights bigger than one are essential to get the uncertainties right, because you get the "right" fluctuations
    - Josh: By generating less events you get the same statistical power and when you generate same number of events you get better power.
    - Frixione: If you reject events either by cuts or accidents, when you have large weights would cause problems.
        - Ben: This is the usual problem with generation in general. If it's random it is not a problem.
- Zach Marshall: your NN accuracy is as good as your training. You need to use a huge training sample.
    - Ben: Training of the NN is very small and the CPU generation of the events could be important but we haven't quantified it.
    - (slightly more detail: CPU/GPU for the training of the NN is usually not a bottleneck. You are absolutely correct that the NN is an approximation and this will introduce some uncertainty. This is the million dollar question for all uses of neural networks where one wants to use the neural network output as a function approximator directly. Learning ratios is easier than density estimation directly, which is a big advantage (and we mostly care about interpolation not extrapolation), but clearly some validation for particular observables would be required and we may need to invent new validation methods as well.)

-- For the questions below, there was no time to discuss them --

- Josh McFayden: In practice what would be the inputs? All final state 4-vectors? If so, could you use this kind of technique to understand better exactly where in phase-space the negative weights come from? I don't know if this would be useful for e.g. further optising MC@NLO-Delta?

    - Answer: you can use anything, but for the top case, we used the 4-vectors of jets and leptons so any observable computed from them is properly reweighted. Yes, you can use this to identify regions of negative weights! This is something I studied but did not put in the paper.

Progress on porting MG5_aMC to GPUs (Stefan Roiser)

- Josh McFayden: I'm not sure I 100% follow what is included in the "Matrix Element" calculation here. I assume that if a "gridpack" run is performed this drops to zero on the grid? Has anyone profiled a run generating events from a gridpack?
    - Andrea Valassi: here we do not even have gridpacks, because there is no Vegas/MadEvent (we use a much simpler Rambo), nor is there any max weight estimation. We just do a run once. The "matrix element" is the calculation of the matrix element, so really the Feynman diagrams: you give initial/final 4-momenta, you get the scattering amplitude for each Feynman diagram, but here we have only two diagrams so it is fast. In a pp process, the ME part if 80% of CPU, both during gridpack creation, and during event generation afterwards.
- Enrico Bothmann: how is this effort related to earlier efforts starting in 2010?
    - Stefan: Argonne team are looking at their gVegas/gBases.
    - Andrea/Olivier: they had also done some work on HELAS (we are using ALOHA but this is not very different), but this is based on an old CUDA4.
- Stefano Fixione: the speedups may bit a bit misleading, because the phase space integrators are also essential (especially adaptive integration), it must be seen if their features may be maintained.
    - Olivier: integration is also important, but we are mainly focusing on ME calculation, which is an orthogonal problem. Also, the KEK team had shown that their gVegas adaptive integration was a valid proof of concept.
- Stefano Frixone: Factors of 1000 are a bit misleading as the PS is flat. In real example will it scale to adaptive integration?
    - Stefan/Olivier: what is mostly being sped up is the time of the ME independent of the PS integrator. For NLO the complication is computing loops on GPU
- Charles Leggat: What do you think the limit will be when you get to more complicated calculations (compute bound or memory bound?)
    - Stefan: Expect the ME calculation to grow dramatically
    - Olivier: e+e- compute bound, gg>tt~gg memory bound, maxing out registers so memory bound.
    - Stefan (post discussion): When being memory bound we should be able to e.g. cut our ME calculations into smaller kernels
- Charles: can you use bfloat or reduced precision?
    - Stefan: yes we can easily switch to single precision and gain a large speedup but it is the physicist that should decide if this is acceptable.

-- For the questions below, there was no time to discuss them --

- Josh McFayden: Is the C++ code generation used for this replacing fortran in standard MG5_aMC? If so, is it foreseen to be useful/worthwhile make this the default in standard MadGraph separate to this GPU-dedicated work?

    - Stefan: From a software engineering perspective we have started this in C++ as the expertise was better for several of us in the language. On a side note, we have not yet done a comparison of fortran vs C++/Cuda performance which could be also interesting.
    - Olivier: I have made a comparison between Fortran/C++ and the conclusion was that C++ implementation of complex number was 9 times slower compare to the fortran implementation. (C++ is technnically more secure concerning overflow computation which explains the speed difference). Compilation flag allowed to avoid that c++ issue.
- John Apostolakis: the current ratio of time for communication device=>host vs calculation seems very large. Is this because only parts of the code are being run on the GPU and others on CPU - or will survive when it is all running on GPU?
    - Andrea: the first, main, reason, is that the ME calculation is too simple (only two Feynman diagrams) and fast, so the copy comparatively is much larger, this will change with LHC processes like gg to ttgg (many more Feynman diagrams). In addition, currently we do not event unweighting at all, so we copy the 4-momenta of all events, but when we start accepting/rejecting events on the GPU, we will only copy a fraction of events from GPU to CPU. We are actually running all on GPU, but you need to write to disk from the CPU and we assume we need to copy everything from the GPU. Also, one thing we are not executing is a cross section calculation, or the unweighting, and we are copying data to the CPU as if it will happen on the CPU, but actually we will implement that on the GPU.
    - Stefan: In addition the memory copy latency should be hideable in cuda streams. We have started looking into this and looks promising.
- Josh McFayden: For the single vs double precision has this change been validated?
    - Andrea: the speedup has been studied, we estimate a factor 2.4 on a V100, see https://github.com/madgraph5/madgraph4gpu/issues/5
    - Stefan: the correctness (physics quality) has not been validated, I imagine we can do this once we have cross-section calculations.
    - Olivier: We need to look at distributions actually, the issue here is that this depends of the process. I expect issue for VbF/VbS process where they are large gauge cancellation or for processes with large interference.
- Graeme S: Also on single/double, any prospect of understanding when doubles are needed vs. singles? Double precision throughput on GPUs is very, very much lower than single so confining their use to “hotspots” would be ideal, if possible.
    - Stefan: Agree. On a professional card (V100) you have the double amount of single precision FP cores than double precision. Other non-pro cards are not even capable of processing double precision. For defining which parts (or all) of the workflow can be moved to single precision I think is a discussion
    - Although you said this is a physics question that’s only partly true - the physics needs impose a necessary precision, but the computer engineering can show how to reorder the calculation to benefit as much as possible from lower precision pieces (not easy though!) [Graeme]
    - Good point, to be honest we hadn’t started this discussion yet, so when we know what is the minimum precision needed we can try to work towards it.
- Josh McFayden: What is the main issue with running loop calcs on GPUs?y
    - Loop computation is using different library that need to be move to GPU first.

        - How much work is this migration…?:-)
        - Olivier: That part is not under my control … so I do not know, I guess quite a lot
    - And such computation need sometimes to be done in quadruple precision
- Xiaocong Ai: What is the typical size of the matrix calculated on GPU? Is there any matrix operation e.g. getting the matrix inverse on GPU?
    - They are some matrix-computation but those are spare 4times4 matrix for the most current operation. For optimization, this is not handle as a sparse matrix computation to avoid to multiply a lot of value by zero. So the real complexity of the computation is related to the number of Feynman diagrams (which grows like a factorial with the number of particles)

PDF/Vegas-Flow (Juan M. Cruz Martinez)

- Andrea Valassi: do you have examples of simple workflows integrating/interfacing VegasFlow and PDFFlow in C++/CUDA? This may be interesting for our MadGraph work, but python is not what we are using.
    - Juan: there are some examples in both repositories. The integration may not be completely trivial in some cases.
- Stefano: when you compare the performance of various scenarios, you do it with a given target accuracy. Is there a significant difference in the number of points that you must throw to reach the same accuracy?
    - Juan: no, it is very similar in standard Vegas and in VegasFlow.
- Josh McFayden: On slide 13 I was a little confused about exactly what the comparison between MG5_aMC and VegasFlow, is this is a pure Vegas vs VegasFlow comparison?
    - Juan: Not quite because the VegasFlow code is adapted from old fortran and is not multi-processing enabled. But since there is no optimisation of this old code it's likely that this would be significantly slower than MG5_aMC without the GPU.

Day 3 - Monday 23 November, Simulation

Geant4

- Expect speed up of 5-10%: could you elaborate where it comes from? Across the border or specific physics lists? [Gloria C.] Answered during the presentation that it seems mostly due to the magnetic field. [JA] Other (technical) code optimisations are also implicated.
- Timing of the major release next year precludes its use for at least the start of Run3. How is the time for releases chosen? [Gloria C.]
    - Gabriele - Geant4 releases are deployed on a yearly basis and the choice to deploy a minor or major release is dictated by the kind of features introduced,

requiring or not code migrations to users, but most importantly on the level of readiness of the developments for deployment on a production release.

- Fast simulations are very much tied to specific detector technologies and already a lot is done in particular by the LHC experiment in this field. How do you see the role of Geant4 in this? [Gloria C.]
    - HSF can help to make these contacts with the experiments and explore adapting experiment codes to be more generic.
- GPU are the new buzz word. You mentioned this is indeed not an ideal fit for simulation, but it may be for 'ad-hoc' problems. How do you see the interaction on this with big experiments frameworks and the fact they they will have access to both GPUs and CPUs resources to exploit? [Gloria C.]
    - [Marc] There are efficient applications of GPU for problems which don't "diverge" much : optical photons, low energy neutrons. The LHCb experiment may benefit from that, with the opticks package (https://simoncblyth.bitbucket.io/opticks/#). We have interaction with the author and consider making this available in Geant4.
- Would the tasking framework allow for batching work in Geant4, e.g., processing all of the particles of a certain type at the same time, which would improve instruction cache usage? [Graeme S]
    - Marc - interesting possibility, but not sure how easy it would be.
    - Jonathan, not really a way to say run all electrons on a single thread
        - My question was more oriented in the other direction, get all CPU threads to process electron tracks at the same time, which would have better use of the cache for electron physics processes [G]
        - This may have major implications for use on-device, e.g. use of GPUs in efficient ways [Adam Davis]
    - It would probably require too much of a communication overhead between the master thread and worker threads [Makoto A].
    - John A> a different model could have two threads collaborating on one in-flight event; the threads could share a few (2, 3?) concurrent containers, for a group of particle types. One thread could concentrate on consuming electrons, another on gammas - and each could take other types when they have exhausted their 'focus' particle type. Not obvious (to me) yet whether this model would fit well with current task implementation.
        - This is an interesting idea, but we need to make sure these 2-3 threads reside on the same core to share the particle stacks [Makoto A]
- What are the opportunities for the experiments to give feedback on the tasking model? Previous threading model was not a perfect fit for the expts. [Chris J]
    - Yes, there's an opportunity to give feedback.\
    - Default for 10.7 will be "classic" MT, Tasking can be enabled optionally at runtime (or application compile time) to test it [Ben M].
- Pushed more and more to use HPCs, which largely means GPUs these days. What happens if we fail to port the software? [Marco C]
    - Can try to use fast simulation, which is more intrinsically portable.
    - Try alternative architectures, like A64FX, which are easier than GPUs.
    - But we do have to explore this in R&D.

- Does Geant4 run on ARM chips and specifically A64FX (the one used in the Fugaku HPC)? [Andrea V]
    - Colleagues in Japan have started to discuss access, but this is early days. We might even get access to the Fugaku HPC if we are lucky.

Requirements from HEP experiments

- A lot of R&D activities started or in the process of starting in Geant4 and the experiment. How do the LHC experiment see the collaboration between themselves and Geant4 on this? [Gloria C.]
    - There are already multiple official occasions of collaboration/exchange between Geant4 and the LHC experiments (see the Geant4 Technical Forum, the recent LPCC workshop dedicated to full simulation requirements and the forthcoming one dedicated to fast simulation), they could be improved, of course.
    - The availability of the different experiments to give early feedbacks/be ready to try R&Ds prototypes outcomes is also very much important
    - Ideally it would be good (also with the help of the HSF community) not only to have fora where to exchange results about ongoing projects and results, but actually also to have some more concrete collaboration (maybe on specific projects) -- this very much depends on the ability of Geant4 and experiments to attract new developers, fund new projects.
- Graeme to G4: how do you see the requirements fit in with other communities fit in with those of HEP? Marc V. the most stringent requirements on speed are from the HEP/LHC experiments although everyone would like to have it faster.
- Paolo: Can metrics for new architectures be integrated into Geant4?

Machine Learning for Detector Simulation

- Slide 7: this would be a very nice plot to show, but the problem is that it's difficult to describe "physics accuracy" (for different distributions) in a single evaluation metric. Is there a consensus on what to use as an evaluation metric to compare the accuracy of two methods? Could one of the commonly used loss functions be used for that? [AndreaV]
    - Very important consideration, sometimes we use loss functions that do not encapsulate the physics. We should definitely think more about it. It is difficult to define a single evaluation metric.
        - Could define a "suite" of quantities, and for each algorithm, pick the quantity where it does the worst (to be conservative)
- What are the benchmarking metrics to say the ML fast implementation is good enough to be used instead of the detailed G4 counterpart? [Gloria C.]
    - Depends on what part of G4 the ML algorithm is trying to replace. Useful to establish a common set of physical quantities for common problems (e.g. EM showers)
- Need to train (on full sim). How many events do you really need? [Liz]

    - Paper cited on slide 11 explores this and one can then infer this number (maybe 1:10 000?). Have to ensure we take cost of training into the overall accounting.
    - A comment about statistics of the training samples, FastCaloGAN use 10k events for each energy point and this may not be enough to capture all the features and correlation in the tails where the statistics is low. Studies are ongoing to better understand the importance of statistics. [Michele]
    - not 100% sure we should compute the training samples in the ML budget. At least in ATLAS we use the same samples for the "traditional" parametric fast simulation, so those would be produced anyway. [Michele]
- Are end-to-end approaches more promising as they go right to final quantities? [Marilena]
    - Paper on slide 13 was for one specific analysis only - would be hard to generalise beyond that. There is room for both approaches.
- Approaches seem to simulate average quantities pretty well, but what about individual showers and their variations? [Soon]
    - Higher moments are examined to ensure that the distributions are correctly encapsulating the fluctuations. Example given of ATLAS FastCaloGan jet mass here (slide 16).
- A comment about statistics of the training samples, FastCaloGAN use 10k events for each energy point and this may not be enough to capture all the features and correlation in the tails where the statistics is low. Studies are ongoing to better understand the importance of statistics. I am not 100% sure we should compute the training samples in the ML budget. At least in ATLAS we use the same samples for the "traditional" parametric fast simulation, so those would be produced anyway. [Michele Faucci Giannelli]

adePT

- Summary:
  Andrei optimistic about possibility of leveraging GPU for simulation speedup. Current work/goal: can we get enough speedup at scale? Goals: build up from simple prototype to more complex/realistic setup. CopCore: externalizable library for GPU code infrastructure. Move from: “list of actions to do for a track” to “list of tracks doing the same action”. Portability: looking at Alpaka and OneAPI and Allen-inspired techniques. Rather than having the CPU decided the number of threads/blocks for each kernel (which would require data to be brought back to the CPU), have a single thread kennel spaw right sized sub-kernel. Discussion on-going between 3 GPUs project (Excalibur and Celeritas).

- John. Andrei mentioned “Typical GPU applications are not stochastic”. There are examples which are stochastic or at least have a tree-like progression of work in particular ray tracing.

- Will your software component for kernel launchers (and other features useful for CPU/GPU single source) be standalone and usable by others? Is this CopCore? We are doing something similar also for event generators. Standalone library? [AndreaV]

    - Andrei: Yes we will distribute an header only library that can be used by other projects.
- Paolo: Is there any value of having this layer rather than biting the bullet and using some of the existing frameworks? [Quote: “The simplest API is my API”
    - Andrei: We are in a rapid prototype phase and we do not know yet what will be “standard” when we are moving to production.
    - As an addendum to Paolo’s comment on not building up a large codebase that might inhibit migration to portable APIs - a key thing that’s being kept an eye on is the core interfaces/algorithms so that it’s (hopefully!) easier to migrate/hide differences. Very much inspired by/taking work/input from Allen here [Ben M].

Celeritas

- Summary:
  Objective: deliver a GPU-accelerated particle transport application for HEP detector Simulation. Collaboration including both HEP and HPC effort/experience. Multiple possible path to usage (standalone or through Geant4). Focusing on LHC simulation as first examples/requirements. Trade-off flexibility/configurability for performance. Bottom-up approach, strong focus on modularity and testing. Taking cues from successfully GPU based nuclear physics simulation (SHIFT) and from Geant4. Isolation of kernel code from data memory layout (will allow to easily switch the backend). Mini-apps: ray-tracing, Klein-Nishina photon transport (Validated against Geant4 for correctness and between CPU and GPU version for performance - starting number (upper bound) x150). Working on implementation of full stepping loop with a few physics process and simplified magnetic field handling.

- Questions:
- An example of using the Geant4 Tasking to “offload” (whether to GPU or Fast/Other) would be a good thing to integrate between the projects, and potentially as a core Geant4 example (or at least parts) [Ben M].
- Attempted using CUDA UVM? [Jonathan M]
    - Seth: Not really, mostly because the only copying that takes part are the upload of the primary particles and the end results.
    - Tom: (to Jonathan) Did you get a chance to do any performance testing that shows that UVM would be beneficial?
    - Jonathan: No, was inquiring due to productivity due to large undertaking
- Slide 14: Those speedups are nice on paper, but the problem is to get those 1M tracks.. [Witek]
    - We expect the GPU will have plenty of work to do if we run multiple events simultaneously. [Seth]
- Marilena: about slide 15 (mini-app) are you using any geometry and transport there? Can you clarify the Geant4 comparison.
    - Seth: For the stepping kernel, we are getting track of the location of the track but it is a uniform material (so no boundaries). The error statistic is high due to large bins and too few events (output too large for the laptop used).

- Andrei: You see a factor 150 and you mention it looks like an upper bound. How far from that do you think the realistic/complete prototype would be.
    - Seth: I would expect at least a factor 2 slower (x75) and with good hope to still improve from that.
    - Tom: For Shift we see factor 160 (with a problem that has a simpler geometry and less physics process than an LHC simulation).

Opticks

- Summary:
  Simulation of transport in Liquid Argon essential to next generation of Intensity Frontier experiments. Yields few 10,000's of photon per MeV. Current G4: hours to simulate each event of a typical Argon TPC, so some experiment use Lookup Tables instead but grow with the detector size and realistic case are already reaching the memory limit of grid nodes. Opticks show good promise to speed-up significantly the processing timing without the memory cost. Only copy the seed information to the GPU: generate and propagate the photon on the GPU and return just the hits. Hybrid workflow is implemented and available for testing. Plan on upgrading to use Geant4 tasking model to improve concurrency. First benchmark result: ~huors (Geant4) vs ~ second (Geant4+Optics), additional x2 (GPU with RTX).

- Questions:
    - Ben M.: Some of this work (the offloading) will also apply to Celeritas/Adept.
    - John A.: Could you clarify the meaning of the curves. Top/Bottom RealTime/CpuTime. Green vs Red: decrease by x10 the number random number seed uploaded. Blue = Red + RTX enabled.
      Hi John: The two curves of each color belong together. To get the timing we just used the unix time command, the upper curves of each pair show the real time while the lower curve is the sum of USER and SYS CPU time. The different colors represent different opticks operating points. For the green curve 100M of random numbers seeds is copied to the device. The red curve shows the same but here onle 10M of random seeds are used. Finally the blue curves show the result for 10M random seeds and the GPU ray tracing hardware acceleration (RTX) switched on. While the factor of 2 is a nice result at this point I don't understand why the CPU time is affected I would have expected that the two curves (real and CPU) would come closer together. So still plenty to learn about the way GPU's behave. Also having to generate random sees and copying to the device might not be necessary if we don't require any backtracing.

Day 4 - Tuesday 24 November, Software R&D

Phoenix Event Display

- Can GDML files be used, or are there convertors from that to the supported formats? [Ben M]. ROOT can of course read GDML and save to that format [Ben M].
- Is it possible to link to MC data [Thomas K.]
    - A: not yet available but would not be a large work to do
- How long is it to start with Phoenix? [Thomas K.]
    - A: depends on the geometry format. One week should be enough if it is using a standard one.
- What about Phoenix internal event format? [Graeme]
    - It's a JSON based format: loader in charge of converting the experiment data format into this representation. Allows not to depend on the experiment EDM.
- Could something like LAr TPC be displayed as well?
    - May have some assumptions like cylindrical coordinates for clusters, but probably could get around this with a minimum of effort.

High-throughput data analysis with modern ROOT interfaces

- Was there interaction with the ATLAS AMG group about the ATLAS OpenData analysis? [Attila K.]
    - A [Stefan W.]: yes, code and data from ATLAS but the effort to convert the analysis to RDataFrame was made by ROOT
- Would you explain more about how the Object Stores are first class citizens with RNtuple? [Doug Benjamin]
    - RNTuple designed to be friendly with object stores. First testing implementation with Intel DAOS not yet available so too early to report.
- Issues encountered in recent tests with RDataFrame with the input file (format?, size?) [Thomas M.]
    - A: clearly a bug to be investigated, probably not related to RDataFrame itself but to the TTree handling with high compression format (eg, write time)
    - Will RNTuple make the things different? [Thomas M.]
    - A: yes and no… It will have more knobs to turn, this may help… but probably the same could be achieved with TTree
- How did you design RDataFrame to help working with many users attacking the same storage? [Graeme]
    - A. [Stefan W.]: RDataFrame is designed to optimize the bandwidth available by opening as many things as possible in //. May not necessarily help if at the end there is contention on the server. But should improve performance if the data is spread over enough servers.
        - It's one of the interesting scaling issues when users share access to a storage server, so worth looking at with facility colleagues (e.g. analysis facility discussions)

bamboo: easy and efficient analysis with python and RDataFrame

- How tied is bamboo to CMS nanoAOD data (or other CMS specific pieces, e.g. data discovery)? Would it be possible to use it in other experiments? [Graeme S]
    - A: only a few parts are specific to NanoAOD (or CMS services and formats), bamboo was designed not to be tied to a single tree format. NanoAOD is the best supported (and documented) case, but adding more is possible (I have implemented the “decoration” of a few other tree formats, e.g. the ATLAS open data for as much as was needed for the tutorial example here). Documentation needs to be improved for this particular need.
- Is there a way to output the graph or some other visualisation of the analysis workflow for checking or presentations etc? [TJ Khoo]
    - A: not implemented yet… but certainly possible, the internal information for doing it is there.
- Can you be more specific about the features to handle systematics, including adding MC information? [Heather G.]
    - A: several parameters can be used but changing/adding MC data is more difficult.
      What is currently there:
        - Alternative event weights (generator, parton shower, efficiency corrections), these are quite light for the RDF graph (an additional weight and histogram)
        - Alternative values for non-weight inputs (e.g. jet/MET kinematics). These may influence the control flow, so as soon as they are used in a cut, the RDF graph needs to be branched, and everything from there on duplicated (so these have a higher impact on memory and CPU usage).
        - Changing a sample isn’t really a change to the RDF graph, but rather an additional sample to fill histograms for, and substitute for the nominal when used later on, so this depends on the tool that’s used to make plots or do statistical analysis. A possible (tested) implementation for a specific case is described in https://gitlab.cern.ch/cp3-cms/bamboo/-/issues/67#note_3846019 but we don’t have a convenient helper method for this yet (related code, for selecting a data-driven instead of simulated background, exists, so it should be possible)

Analysis Description Language for LHC-type analyses

- What was the incentive to use a text format rather than a more structured format like JSON? [Andreas S.]
    - A: text file is really easy to read for a user who wants to check it
    - Andrea: in the past very often prone to come up with our own HEP solutions but paid the price in the medium/long term of being alone to maintain things when the authors left the field. Converters exist for example to make JSON easy to read.

- On the same line, how much effort was spent on writing the text parser? What was the real added value? [Attila K.]
    - A: parser based on Lex/Yacc. In the long run, it would be good to have somebody who writes a parser converting the analysis code to an intermediate representation like LLVM one.

podio - latest developments and new features of a flexible EDM toolkit

- Guess an obvious comment given the earlier talk on Phoenix (and maybe then for Ed), but how easy/difficult to support a Podio EDM in Phoenix? [Ben M.]
    - A: I had a similar thought, but for an actual answer I would have to look at how the loaders do their thing in Phoenix. From the PODIO side it should be fairly straight forward to generate any additional code, by just defining a new template using jinja2 and hooking that up in the code generator [Thomas M.]
- Related: how easy/hard to integrate with other components in the HEP world (e.g. ROOT/Frameworks like Gaudi/Art/etc [Ben M]
    - A: it should be pretty easy in principle. As for Gaudi, it is part of Key4HEP so it is already integrated with PODIO.
- Focused mainly on IO and memory but also very involved in computing. Is it possible to define the kind of Math library to use to handle vectors and the likes? [Andy S.]
    - A: not clear how hard it is to integrate this into PODIO but the the fact that memory representation is continuous should help.
- Is it possible to say more on ROOT/SIO performance difference? [Enrico G.]
    - A: not really, no attempt to investigate so far, the goal was to demonstrate the multi-backend capabilities. But would be good to follow up and do a more in-depth analysis.

Use of auto-differentiation within the ACTS toolkit

- Can you comment on the relationship between the library you use and the (unsupported) Eigen module [Josh B]
    - Not really, as I did not use the Eigen one
- Looking at the performance, are there other places that come to mind where it would be possible to replace current implementations (numerical) and therefore optimise? [TJ]
    - Project has only just started, so not yet looked at other applications
    - In general in tracking code as long as one can write down the equation of motion the semi-analytic derivatives can be evaluated (often they are huge). The fact that auto-diff gets the same result is very encouraging - current use is as a validation tool.
- If used for validation, does it give you an estimate of the precision? [Goetz G]
    - Precision is driven by the stepwise integration. Auto-diff is exact wrt the (adapted) step that has been taken.
    - Generally auto-diff produces pretty efficient numerical code, so losing precision is not usually a problem.

Reconstruction for Liquid Argon TPC Neutrino Detectors Using Parallel Architectures

- How much code had to be hand-vectorised? [Graeme]
    - Vectorisation happened just in the implementation of the fitting algorithm. Comparing to some libraries, hand version was actually better, but keeping this within the fitting algorithm also means that it can be replaced by a better one.
- Spack is quite effective for compiling for specific architectures depending on which nodes you compile on.
- AVX512 could get factor 8 speedup, where you get factor 2, why the difference? [Soon Yung Jun]
    - Loss of efficiency might come from not knowing in advance how many interactions are needed for convergence
- Do you use analytic gradients in your minimisation algorithm? [Josh Bendavid]
    - Yes

GPU-accelerated machine learning inference for offline reconstruction and analysis workflows in neutrino experiments

- How much data needs to be moved between Fermilab and the Google Cloud during initialisation and then event-by-event? [Attila K.]
    - Almost 4Gbits (transferred in 2s)
- Can the data be compressed usefully? Would it help reduce the time? [TJ]
    - A: has not been explored yet, to be done
- Economically this makes more sense than a GPU on every worker node, but would your final implementation be Google-based, or have a local GPU farm? [Graeme]
    - Yes, would want a local farm eventually.

GPU-based tracking with Acts

- Are there performance comparisons between CUDA and SYCL/oneAPI [James C]
    - The code is not exactly the same between the two versions. CUDA is a bit faster, 10-15% (CUDA version also has smaller overheads, so works better with small numbers of seeds).
- Geometry is different between sim and tacking, but their could be commonalities worth exploring [John A]
    - Yes, ACTS would like to explore this and discuss how to take advantage of VecGeom based code.
- Does the triangularisation affect the precision? [TJ]
    - Not really for tracking (which construct a surface geometry anyway). Main idea was to have all surfaces represented by the same types, so could use the same (efficient!) code for finding the intersections in a parallel search

A SYCL-based prototype for fast calorimeter simulations

- Slide 10, performance results with SYCL. Was the original CPU time for a single thread? [Attila K]
    - Yes, just one thread
- AMD hardware - buy a Playstation!
- Surprised that everything had to go into one compilation unit (per kernel) [John A]
    - Yes, but it's not terrible for FCS as it's quite simple code. But this is a limitation of Kokkos at the moment.
    - You can probably compile to *static* libraries, then link into the same shared library as needed for the executable [Attila]
        - Charles - yes, this works, but requires more drastic changes to the build setup in CMake. Didn't want to do that so that the code could also still go back to the ATLAS production version
    - Even with static libraries, all device symbols accessed from a single kernel do have to be in the same *translation unit*. This is because of limitations in device linking by lvcc. Kokkos disallows the use of relocatable device symbols.