Report on the 2020 ENCCS training survey

K. Thor Wikfeldt, ENCCS Training Coordinator.

January 25, 2021

1 Introduction

The EuroCC National Competence Center Sweden (ENCCS, https://enccs.se/) started on September 1 2020 as one of 33 national nodes of the EuroCC project (https://www.eurocc-project.eu/). The mission of ENCCS is to develop competence, knowledge and support in Sweden to enable researchers to take advantage of forthcoming (pre-)exascale EuroHPC resources. A key part of this mission will be to deliver training events for researchers aiming to scale up their HPC workloads, adapt their research software to new hardware or adopt artificial intelligence (AI) or high-performance data analytics (HPDA) methodologies in their research.

To guide the development of a novel ENCCS training portfolio, a survey of training needs within HPC, AI and HPDA was designed in the first months of the project and broadly disseminated to researchers and engineers in both academia and industry in Sweden. In this report we present results from the first survey conducted in 2020 and suggest how these could be used in the development of the ENCCS training portfolio. We also hope that the survey and its results can be useful to other HPC/AI/HPDA training providers both in Sweden and other EuroCC member countries. Similar training surveys will be conducted by ENCCS annually to monitor how training needs evolve, to probe how successful past training events have been and to identify areas that can be improved.

2 Method

2.1 Questions

The main aim of the ENCCS training survey is to build an understanding of what specific training topics are needed by Swedish researchers and in which format they should be provided. Additional questions were added to the survey to get a picture of the background, experience and HPC/AI/HPDA usage of survey respondents. After collecting ideas from ENCCS team members as well as collaborators from SNIC, the survey was composed using a tool called Survey&Report (https://www.artologik.com/en/SurveyAndReport.aspx) available for Uppsala University staff. It contained the following questions:

1. Please specify your current position
   - Undergraduate student
   - PhD student
   - Postdoc
   - Researcher
   - Assistant/associate/full professor
   - Research software engineer / scientific programmer
   - Researcher/developer in small/medium company
   - Researcher/developer in large company
   - Data scientist / machine learning engineer in small/medium company
   - Data scientist / machine learning engineer in large company
   - If other, please specify
2. Please specify your main scientific discipline or field of work. Please take the entry which is closest to your main field [drop-down menu with list of disciplines]
3. If you work in a public or academic institution, please write its name
4. Do you already have research problems or cases which could be scaled up if you had access to much larger computational resources? [yes/no/don't know]
5. If you tomorrow got access to 10 times larger computational resources than you currently have, what would be the main impediments for you to taking advantage of it?
6. Roughly how many years of HPC or AI/HPDA experience do you have?
   - High Performance Computing
   - Artificial Intelligence and High Performance Data Analytics
7. Please give us an idea of your regular HPC usage
   - I use existing software on HPC resources

- I develop my own HPC software
- Both of the above
- If other, please specify

8. How many CPU cores and/or GPUs do you usually use for your jobs?
   - Typical number of cores
   - Maximum number of cores
   - Typical number of GPUs
   - Maximum number of GPUs
9. Roughly how many core/GPU hours on average do you use per month?
   - CPU core hours
   - GPU hours
10. Please specify which existing HPC/AI/HPDA software packages you use (if any)
11. If you (or your group/company) develop or contribute to HPC software, can it run on heterogeneous computing architectures? (e.g. CPU+GPU machines)
12. If you (or your group/company) contribute to or use HPC software, do you face I/O bottlenecks? If yes, do you use high-performance data analytics (HPDA) techniques? [Yes but I don't use HPDA techniques / Yes and I use HPDA techniques / No / Don't know]
13. Are you using AI methods (neural networks, deep learning, etc.) in your work? [Yes / No]
14. What type of training are you interested in attending and at what level? [Introductory, Intermediate, Advanced, Not interested, Don't know what it is, Other - please specify]
    - Specific types of simulations/modeling
    - Specific HPC packages
    - Programming languages
    - MPI
    - OpenMP for CPUs
    - OpenMP for GPUs
    - OpenACC
    - CUDA
    - AI/Deep Learning
    - HPDA
    - Performance engineering
15. Please rate how valuable you find these different types of training events
    - Presentations with slides
    - Live demonstrations / code-along
    - Guided exercises / hands-on sessions
    - Group work and discussions
    - Supervised project work
    - Self-paced learning
16. How many days do you think is optimal for training events?
17. Do you prefer in-person or online training events? [In-person, Online, Don't know]
18. Would you like to be able to access HPC systems through a cloud environment? [Yes, No, Don't know]
19. Any final comments?

2.2 Dissemination

The 2020 ENCCS training survey was conducted from mid-October to mid-November. It was advertised via the following dissemination channels:

- Email to all PIs of SNIC Large projects on Oct 20
- Another email to SNIC Large PIs where also SNIC Medium PIs were included on Nov 11.
- Email to around 40 industry contacts on Oct 21.
- Announcement in the SNIC training newsletter on Oct 23 and Nov 16
- Announcement in SeRC mailing lists on Oct 28 and Nov 10

- Announcement via RISE mailing lists on DATE

In total 134 survey responses were obtained. The number of responses per day is shown in figure 1. The largest number of responses was obtained when the first announcement was sent out to principal investigators (PIs) on SNIC Large projects on Oct 20. The announcement via SeRC mailing lists on October 28 also appears to have been effective, and on Nov 10-11 when emails went out to both SNIC Large and Medium PIs and the SeRC mailing list a large peak is seen. Even though SNIC training newsletters are highly effective in attracting registrations to training events, the training survey announcements on October 23 and November 16 did not lead to many immediate survey responses.

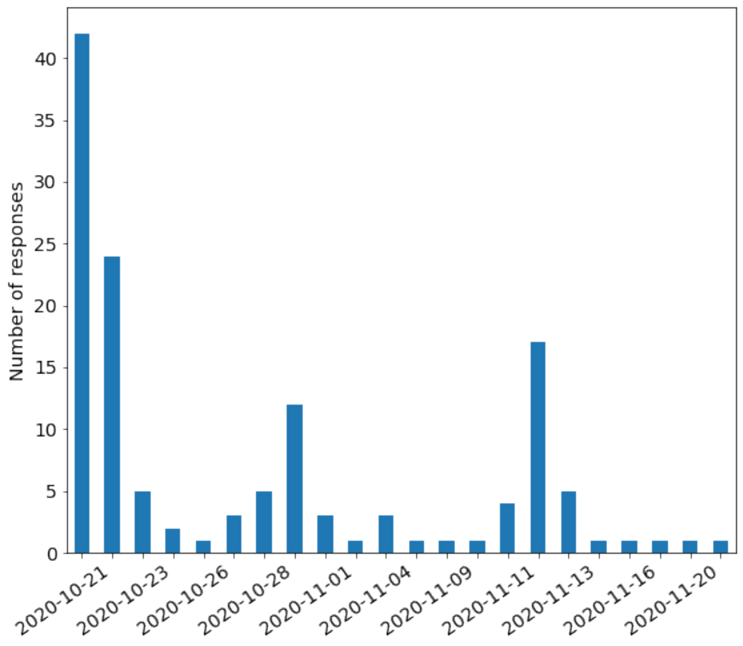

Figure 1: Number of survey replies over time.

3 Results

3.1 Respondent background

3.1.1 Job title and field of work

Answers to the question *Please specify your current position* are shown in Fig. 2. It can be seen that a majority of survey respondents are assistant, associate or full professors. The emails sent to all principal investigators of medium and large SNIC allocations were clearly the most effective dissemination channel. Ph.D. students, postdocs and researchers sum up to approximately as many responses as from professors, while researchers/developers in industry and research software engineers each represent a small share of the total number of responses. It will be important to more effectively reach these groups for the next iteration of the survey in 2021. One way to better reach more junior academic researchers could be to have the survey open over a longer time period and advertise it during all ENCCS events during the survey period. ENCCS will also hopefully have more followers on Twitter and subscribers to the ENCCS newsletter next autumn which should increase the response rate. To better reach industrial researchers and developers, it might be worth to consider reformulating some of the questions to better match industrial terminology and user experiences among SMEs and large companies. This can be accomplished by working together with RISE in adapting the survey questions.

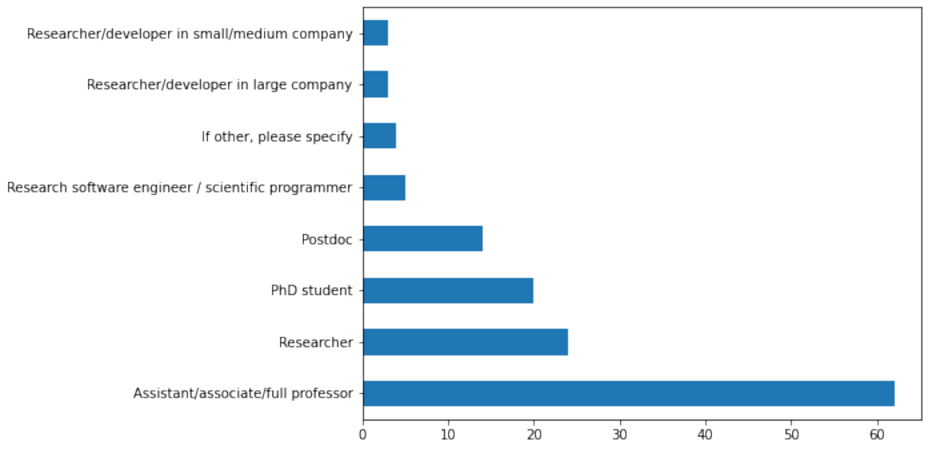

Figure 2: Job title

Answers to the question *Please specify your main scientific discipline or field of work. Please take the entry which is closest to your main field* is shown in Fig. 3. The largest number of responses comes from the physical sciences followed by biological sciences, computer and information sciences, and chemical sciences. This distribution corresponds well with the number of SNIC compute and storage projects from these different disciplines and reflects that the most effective dissemination channels this year were the emails to SNIC PIs and the SNIC training newsletter. A challenge for next year's survey will be to reach a broader range of disciplines, which we hope will be possible through an expanding network of contact points to actors in academia, industry and public management.

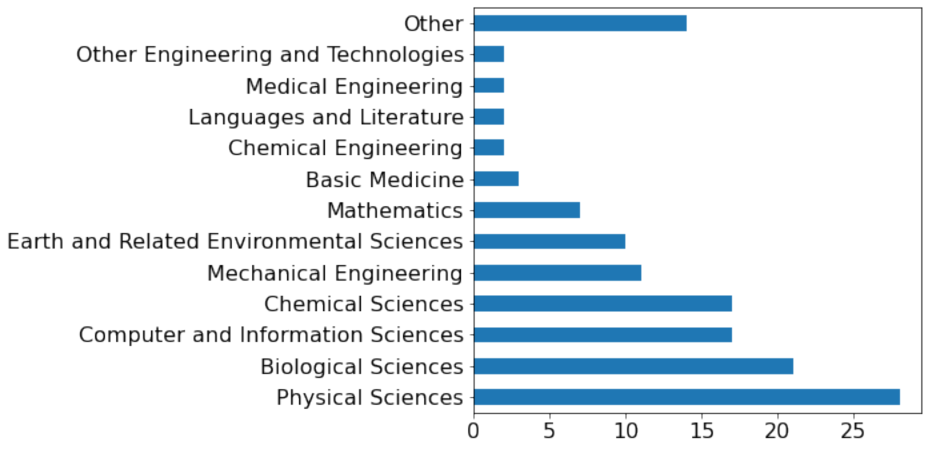

Figure 3: Specified scientific discipline or field of work.

3.1.2 Prior experience and current use

Answers to the question *Roughly how many years of HPC or AI/HPDA experience do you have?* are shown in Fig. 4. Survey respondents have widely different experience in HPC topics, with most answers in the range 0-10 years but a significant number of respondents report 20 years of experience. On the other hand, most respondents have little AI experience; a majority has no experience (0 years) but a sizeable fraction has 1-5 years of experience with AI.

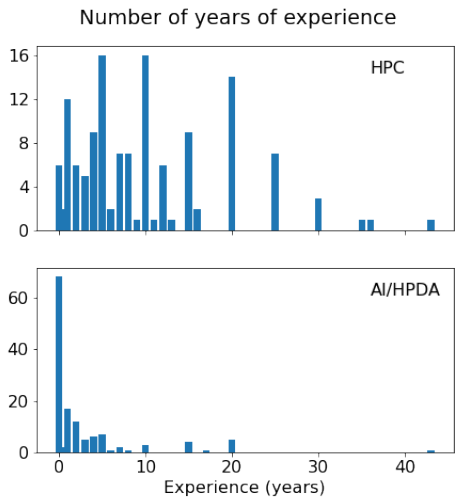

Figure 4: Question: Roughly how many years of HPC or AI/HPDA experience do you have?

To the question *Are you using AI methods in your work?*, 65% answer "No" and 35% answer "Yes", as seen Fig. 5. This agrees well with the results in Fig. 4(lower panel) where around a third of respondents have more than 0 years experience in AI.

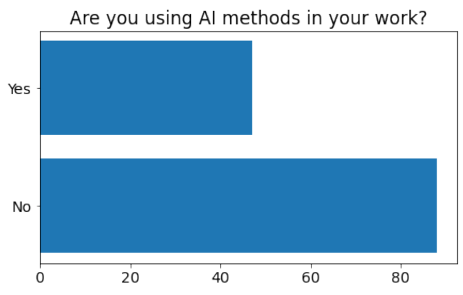

Figure 5: Are you using AI methods in your work?

Answers to the question *Please give us an idea of your regular HPC usage* are shown in Fig. 6. It can be seen that roughly 50% of respondents develop own HPC software (out of which 34% also use existing software) while 45% use only existing HPC software. The fact that around half of the respondents report developing own HPC software emphasizes the importance of providing training in HPC programming methods. In the "Other" category the following answers have been provided:

> *Use existing but script much to combine and pipeline*
>
> *I solve optimization problems (dynamic programming problems) using Intel Fortran.*
>
> *In collaboration with the university of Oulu, department of Physics (NMR), which develops new packages for advanced computing on paramagnetic systems and in imaging. Also CASTEP package is used.*
>
> *I develop my own AI tools and simulators, not sure if this goes under "HPC" though*
>
> *Currently not using HPC*
>
> *Modules to existing software*
>
> *Installing existing software which is specific and not available on HPC resources*
>
> *Using GROMACS with modifications*
>
> *local ai computation*
>
> *We customize the existing codes (OpenFOAM)*

The answer *I develop my own AI tools and simulators, not sure if this goes under "HPC" though* points to possible improvements of this question - "HPC software" could be replaced by "HPC/AI software" to cover more use cases.

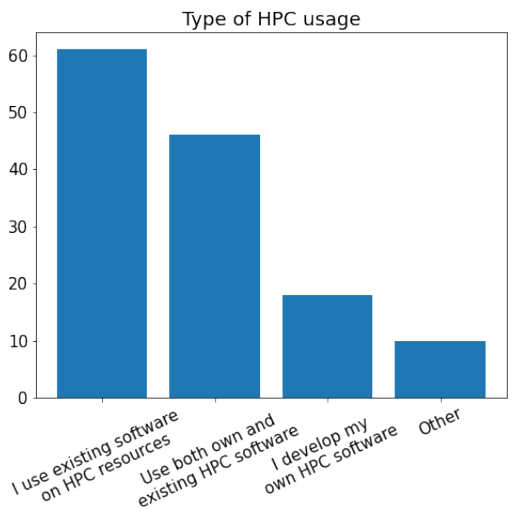

Figure 6: Question: Please give us an idea of your regular HPC usage

The survey asked participants to specify typical and maximum number of CPU cores and GPUs used in their research and the results are shown in Fig. 7. Note the logarithmic x-axis in these plots which is used because of the wide range of numerical answers. The distribution of typical and maximum number of CPU cores is wide; most answers are between 10 and 1000 cores but a few outliers report a maximum number of cores over 100,000. Most respondents on the other hand use only one or a handful of GPUs. It is noteworthy that several respondents report using over 10 GPUs (and even up to 1000 GPUs). Many survey respondents are principal investigators leading research groups and might not run own HPC/AI jobs, which explains the large fraction of responses giving zero cores/GPUs.

Most survey respondents use less than 200,000 CPU hours per month, see Fig. 8(left). Considering that most respondents report having research problems that could be scaled up to larger resources (see Fig. 9 and the discussion below), the distribution around relatively low average monthly CPU usage might reflect the limited CPU cycles available within SNIC. It could also indicate a problem with the question - many PIs do not run own HPC workloads so the question could be reformulated to reflect the usage of the entire research group.

Most respondents report using no GPU hours at all, as seen in Fig. 8(right), but there is a long tail up to very large GPU usage (two respondents use 2 and 3 million CPU hours and 200 and 500 thousand GPU hours per month, this is not shown in the graph). It will be interesting to compare these numbers with results from next year's survey to see whether GPU usage will increase significantly as more SNIC and EuroHPC GPU-based systems become available.

3.1.3 Scaling up to larger resources

Interestingly, most respondents state that they have research problems or cases which could be scaled up immediately given larger computational resources, as seen in Fig. 9. From this question alone it is not completely clear whether this means that researchers use research software which can scale out to larger numbers of cores/GPUs, or have research problems that benefit from running larger numbers of jobs at the same level of parallelization.

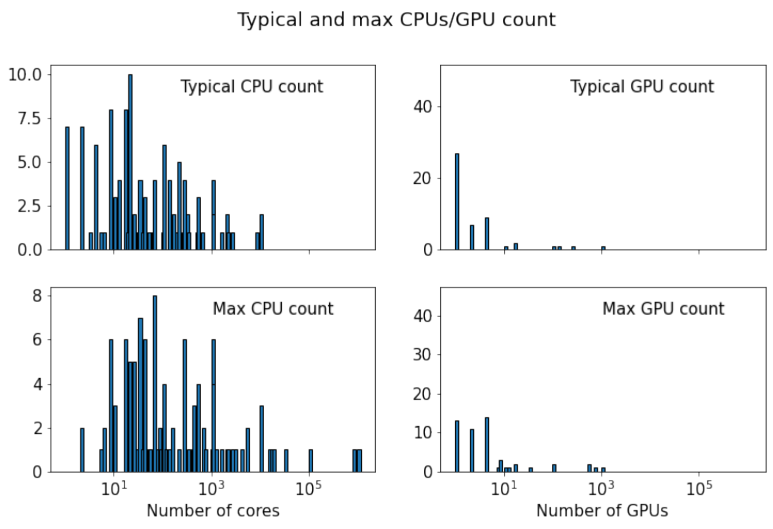

Figure 7: Question: How many CPU cores and/or GPUs do you usually use for your jobs?

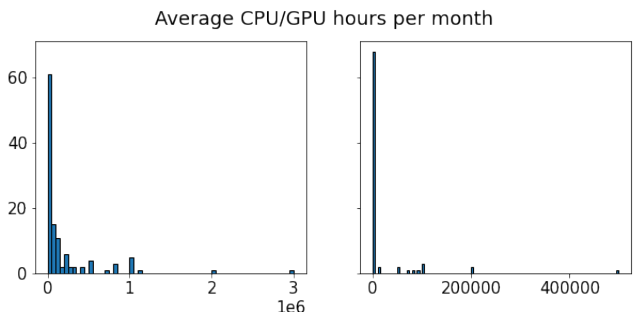

Figure 8: Question: How many CPU cores and/or GPUs do you usually use for your jobs?

However, further information is obtained from the free-form question *If you tomorrow got access to 10 times larger computational resources than you currently have, what would be the main impediments for you to taking advantage of it?*; a few representative answers are provided in Appendix A. A frequent theme in these answers is that data processing becomes a bottleneck when trying to scale up computational research. A valuable lesson that can be drawn is that training on big data, handling large datasets and high-performance data analytics could have a large positive impact. Other perceived impediments to scaling up to larger computational resources found in the answers include:

- Access to large high-performance storage and fast data transfer for big data
- Limited staff time, need to learn new methods or how to use the large system
- Need for training and support from software experts to port codes
- Need to adapt and reformulate research problems and questions
- Limited scalability of currently used codes

The answers to this free-form question, listed in Appendix A, are generally highly useful but it is evident that respondents have interpreted the question in different ways. In the next iteration of the survey the question could be made more specific and focused on impediments relating to simulation software, data, running jobs etc.

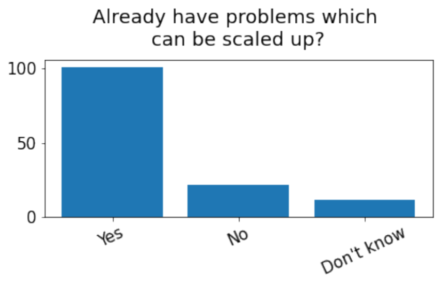

Figure 9: Do you already have research problems or cases which could be scaled up if you had access to much larger computational resources?

3.1.4 I/O bottlenecks and HPDA methods

Confirming the conclusions drawn in the above section, Fig. 10 shows that a significant fraction of survey respondents face IO bottlenecks.

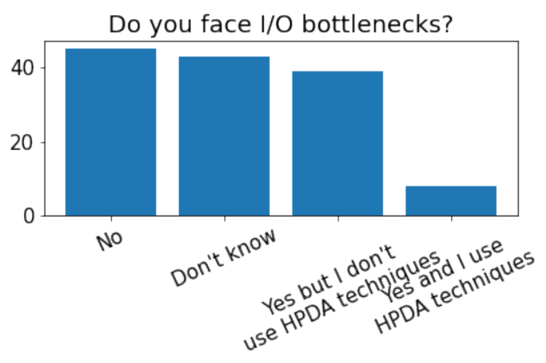

Figure 10: Question: Do you face I/O bottlenecks?

3.2 Training needs

3.2.1 Training topics

A key question in the survey asks participants to specify which topics they would want to attend training on and at which level. Figure 11 shows how survey respondents report interest in various training topics. There is clearly a significant demand for advanced training in programming languages and in domain-specific training, as well as in introductory level training in AI and HPDA methods. Traditional HPC training topics like MPI and OpenMP (for CPU-threading) are not among the most popular topics; there is interest however in introductory training in OpenMP for GPU offloading. It's also interesting to note that OpenACC training is the least popular topic and trails behind CUDA.

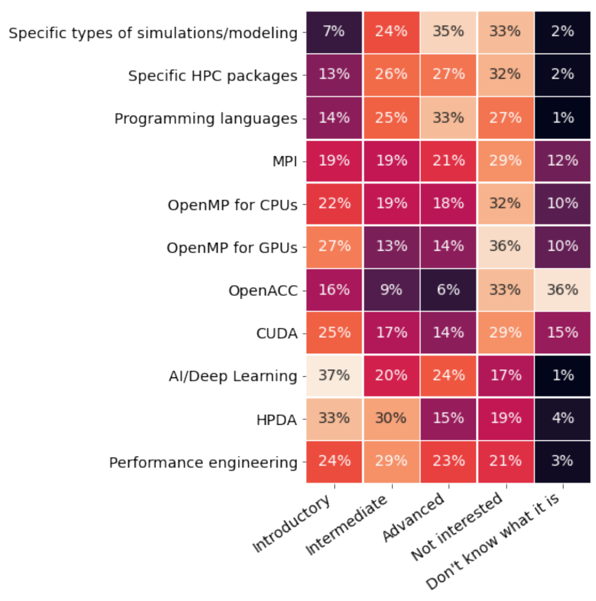

Figure 11: Question: What type of training are you interested in attending and at what level?

Since around half of the survey respondents are university professors, and professors rarely attend training events, it can be interesting to split results into three groups: 1) researchers, research software engineers, Ph.D. students and postdocs; 2) university professors; and 3) industrial researchers/developers. Group 1 can be considered the most likely audience for ENCCS training events. Figure 12 shows topic heatmaps for these three categories and clear differences between the different groups of survey respondents can indeed be discerned. Junior researchers are significantly more interested in advanced training in programming languages, training in OpenMP for GPU-offloading and in performance engineering. Although the sample size for industrial researchers is small, it can be seen that this group is interested in intermediate-level training in domain-specific topics, programming languages, HPDA and performance engineering as well as introductory-level training in OpenMP for GPUs and CUDA.

Free form comments to this question are listed in Appendix B. A few interesting training topics suggested by respondents include:

- Software engineering skills such as testing, debugging, refactoring.
- Modern programming languages with high performance but high-level syntax like Julia or Python (with fast libraries).

3.2.2 Training types

Figure 13 shows answers to the question *Please rate how valuable you find these different types of training events*. Survey respondents overall find live-demos (type-along sessions) and hands-on exercises somewhat more useful than lectures with slides. Self-paced learning is also considered useful, while group work / discussions and supervised project work is less useful.

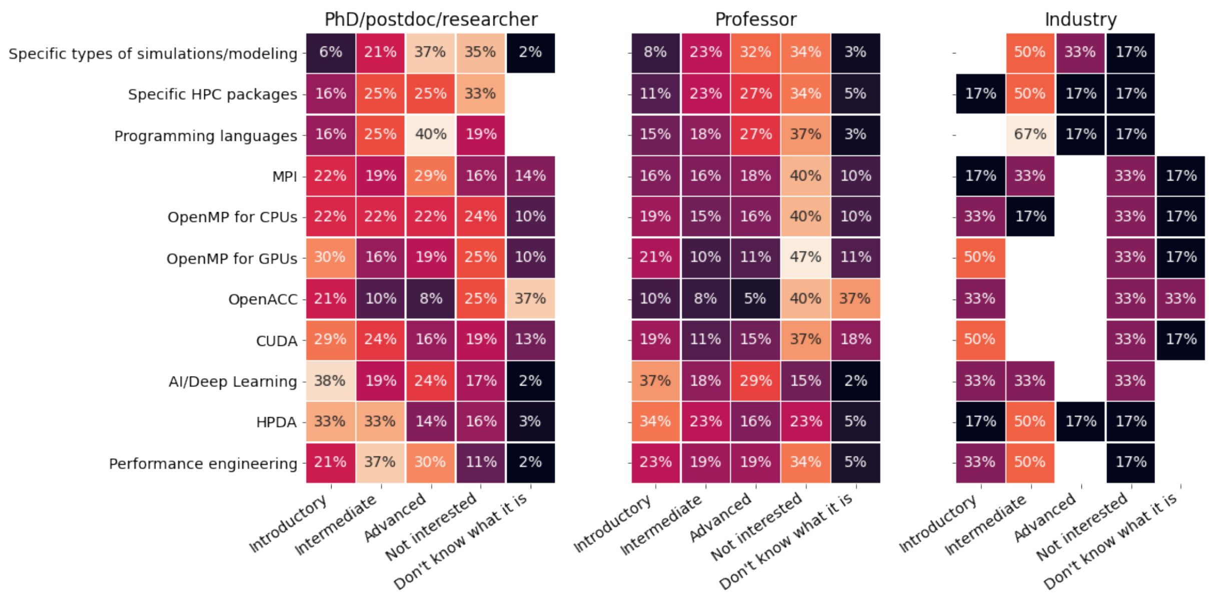

Figure 12: Same question as in Fig. 11 except split into three groups: (left) Ph.D. students, postdocs and researchers; (middle) assistant/associate/full professors; (right) industry researchers/developers.

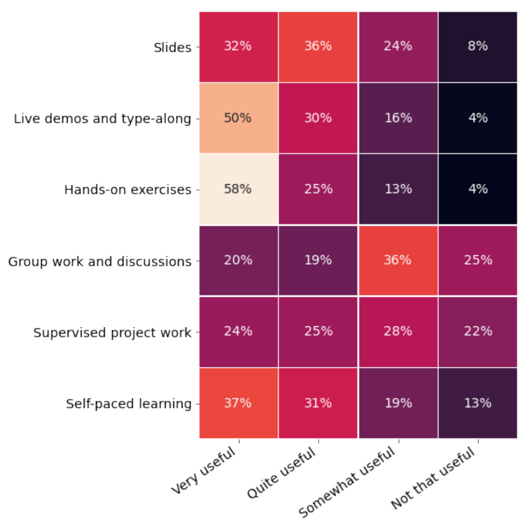

Figure 13: Question: Please rate how valuable you find these different types of training events

Interestingly, most people prefer online events, as can be seen in Fig. 14. This raises the question whether online events should continue to be the norm even after the COVID-19 pandemic.

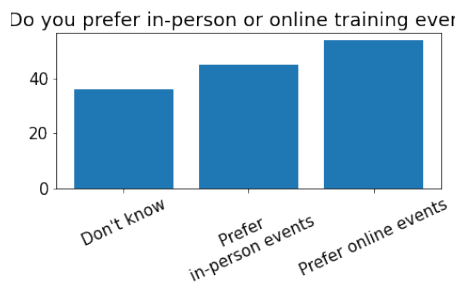

Figure 14: Question: *Do you prefer in-person or online training events?*

Most respondents prefer 1-3 day training events, as can be seen in Fig. 15

3.3 General questions

3.3.1 Cloud

To the question *Would you like to be able to access HPC systems through a cloud environment?*, 46% of survey respondents answer "Yes", as can be seen in Fig. 16.

3.3.2 Final comments

Survey respondents were invited to write any final comments at the end of the survey. Appendix A lists all answers provided.

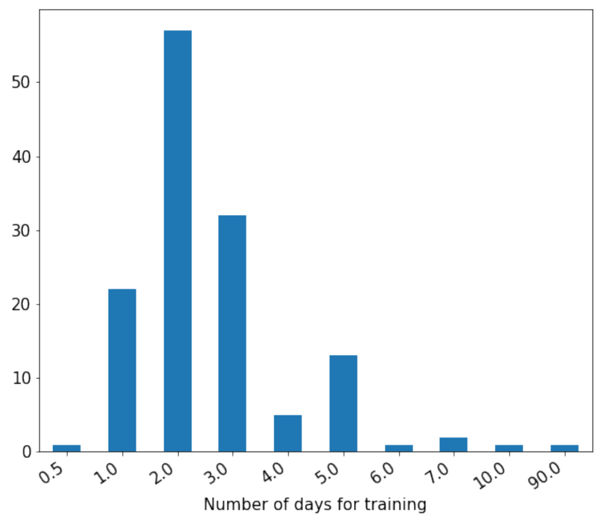

Figure 15: Question: How many days do you think is optimal for training events?

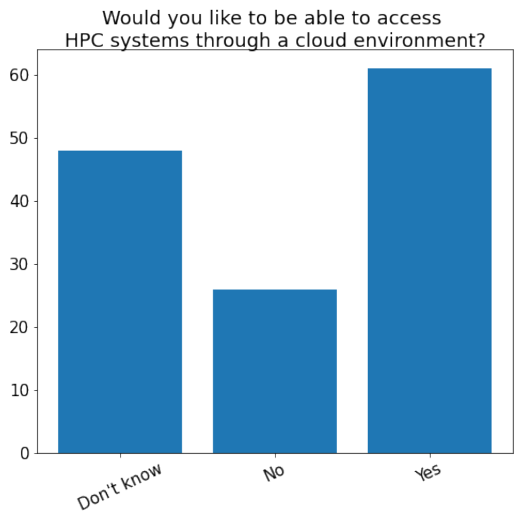

Figure 16: Question: *Would you like to be able to access HPC systems through a cloud environment?*

4 Conclusions

Obtaining 134 responses to the 2020 ENCCS training survey can be considered a success given the total number of HPC users in Sweden is not likely to exceed one thousand. However, around half of the survey responses came from professors and the other half from researchers, postdocs, Ph.D. students and industry researchers/developers. Opinions from professors and PIs are highly valuable but for next year's survey it will be important to reach more junior researchers and in particular also industrial researchers, as these are the groups that perform the technical research work on a daily basis.

It is also clear that survey respondents have widely different backgrounds and experience in HPC, AI and HPDA methods, their usage of computational resources range from small to very large scale, and they also have different requirements and training needs. No realistic training portfolio could satisfy these diverse preferences.

A few main points can however be summarized to serve as a guideline for the development of future training events in ENCCS. The following recommendations are based on a comparison of the survey results with training currently offered by training providers in Sweden:

- Training on high-performance data analytics is strongly needed. Many researchers already face IO bottlenecks, and the need to handle large datasets will increase as more computing power will become available to researchers in the pre-exascale era. Most survey respondents are interested in introductory and intermediate level training in HPDA.
- A majority of researchers already have research problems that can be scaled up to larger computational resources, but several impediments exist:
  - Many researchers need help in scaling up or porting their HPC codes to upcoming pre-exascale systems. This suggests that more hackathons should be organized where researchers can get help from expert mentors. GPUs will dominate upcoming pre-exascale systems so these hackathons should focus on some aspects of GPU programming.
  - Some researchers use software that can not scale up to larger resources, suggesting that training should be offered on domain-specific HPC software that does scale well.
  - Some researchers report having to reformulate their research problems to be able to take advantage of upcoming larger resources. ENCCS and other knowledge hubs should offer consultancy to

researchers on how this can be done. Another possibility could be to offer training on automated workflows - manually running 10 times as many HPC jobs with subsequent post-processing and analysis might not be feasible for small research group, but workflow tools exist to automate large scale computational work.

- Existing HPC training has too little focus on programming languages, possibly because universities offer courses on languages such as C and C++. Especially junior researchers and industrial users report a need for intermediate and advanced-level training in programming languages. Since conventional HPC programming languages (C/C++/Fortran) take long to master, most benefit could be drawn from developing training on Python or Julia for HPC and AI.
- Introductory-level training in AI methodologies is in high demand among both junior and senior academic researchers as well as industry users. Training providers should make sure to develop their own competence in AI fields to be equipped to deliver such training.
- Interest in MPI workshops is rather low compared to other topics - this might also partly be reflected in low attendance in past SNIC MPI training events. Although regular MPI training events will still be enormously important to a certain group of HPC developers, it might be more efficient to devote time to training in new topics, such as AI, HPDA, advanced-level programming and GPU programming.
- All training events should include live demonstrations, type-along sessions and hands-on exercises, while presentations with slides should be minimized.
- More people prefer online events to in-person events. Online events should be continued even after the pandemic and possibly even become the norm. Online events are also more inclusive since geographical location (or physical ability) is irrelevant.
- Training events should last 1-3 days.

Appendix A

Free-form answers to the question *If you tomorrow got access to 10 times larger computational resources than you currently have, what would be the main impediments for you to taking advantage of it?*

> *Post processing capabilities*
>
> *Money for researchers to carry out the work*
>
> *I work alot with text-based analysis of social media posts. One issue is that the analyses cannot easily be parallelized, because you need to include all of data into a model (for example, topic models). This makes these processes extremely RAM intensive for large datasets - also perhaps because the functions for running these models are not so efficient.*
>
> *I would be able to simulate larger systems than I do right now. Also I would be able to complete several publishable projects which would have a high impact for the society.*
>
> *scalability*
>
> *- Scaling - helping team members/students find ways to scale the *problem*, rather than merely execution of a given simulation in parallel. - Adapting to new architectures - it's a major challenge that funding agencies think it's reasonable for them to take six months to evaluate a project, while the researchers should be able to adapt in two weeks, or 4h ;-). Scaling up/down plan is fine, but to have a chance to plan work e.g. for students, we need longer notice :-)*
>
> *Limited working time (personal or students) to spend on and lack of scientific problems that require that much computational power.*
>
> *- modify our in-house code, porting partly to GPU - perform large scale simulations increasing the number of droplets - research and develop "large data" specific post processing tools*
>
> *Disk space. Downloading data from repositories.*
>
> *Improved quantum monte carlo simulations for electronic structures described with dynamical mean field theory; Improved exact diagonalization algorithms; longer time scales and larger simulation cells in spin-dynamics simulations; faster throughput for data-driven materials science*
>
> *Moving the data to the appropriate resource would be the biggest hurdle. After that probably keeping up with generating enough data to keep using 10x more resources.*
>
> *None, my project runs in 4/8 GPUs (single node) at the moment, I should be able to adapt it.*
>
> *Set-up and configuration*
>
> *High performance data analysis*
>
> *No impediments in particular. The tools for gene network inference and machine learning we have developed, and are already using on SNIC HPCs, are using scalable technologies (e.g. OpenMPI or CUDA), so an increase in magnitude of computational resources would result in an almost linear decrease in wall time of the same importance.*
>
> *Tomorrow is a bit late because I'm at the end of my PhD. But I could have done so many more simulations and projects if I had more computer time. Or rather, if the resources were faster.*
>
> *1. Licenses for a commercial software 2. Storage to the generated data so that analysis can be properly done and not have to be deleted*
>
> *We would need about a week to re-plan our work and set everything up, but since computational time is always a limiting factor for us, actually using 10x more in a productive and meaningful way (not just burning core hours for the sake of it but addressing scientific issues) would not be very challenging.*
>
> *Not much, Hebbe cluster has a good queue system for large simulations.*
>
> *Permissions from supervisor. Cost. Permissions from university about data storage and IP.*
>
> *No major impediment. Just run 10 times more calculations.*
>
> *Processing and visualizing the resulting data.*
>
> *clear interface and accessibility hub*
>
> *Code adaptation to resources hardware.*
>
> *At this point in my research, I have no need for more computational resources. But if I had access to more time, I would probably perform calculations in larger systems (clusters of molecules, instead of a single molecule).*
>
> *Taking the time to learn new tools probably*
>
> *I use Intel Fortran and OpenMP. I don't know other methods than OpenMP to handle parallelization. So if I got access to more resources, it would need to either be in the form of more cores per node, or I would need assistance in switching to another suitable programming language. However, I have already taken a small course on MPI and found that to be impractical with too many pitfalls.*
>
> *Use higher spatial and temporal resolution models (speed/performance + data storage)*
>
> *None; we would just use it.*
>
> *QM calculations and MD simulations on ionic materials in bulk and, in particular, at interfaces of a range of metal oxides and composites. The latter needs high computational resources for larger cations and anions, such as in ionic liquids.*

The price per core hour.

Inter-processor communication overhead in case of CPU scaling. Ambiguity over future of GPU libraries standards (CUDA/OpenACC/OpenMP etc) prevents writing code to exploit GPU scaling.

None. I could start using the infrastructure right away.

Most likely the coding involved in farming out jobs to multiple CPUs.

The time it takes to change research plans to take into account the greater access. To be more precise: in one project I could use it right away, since I'm in the process of doing data analysis right now. In another project I have invested close to a year with running small-scale simulations. To change the size of my simulations at this point in time would be hard, basically forcing me to re-run a lot of work to make sure it's not due to scale I get different results.

Time needed to workout experimental setups, compile models, and analyze results

Not enough time to set up all possible simulations.

information on how to best use this resource

I would like to simulate more realistic systems at the first-principles level.

The simulation code I am using can be run on GPU clusters, but not with full physics content. It would need further development in that direction.

I am a biologist with no experience with this. My group has recently gotten into bioinformatics work using next generation sequencing data. We have a lot more in our future so we are learning how this works now. I currently am not personally using this but my group members are. I will slowly learn for myself

1. Larger systems / higher accuracy 2. Testing of quantum computer simulations and combination of quantum and classical computers

The time to adapt the software to the new hardware environment.

I am working with running and combining outputs of various Climate and ESMs. This needs runs of several large models in parallel or as dynamically coupled runs for estimating current and future Earth resilience issues. Larger capacity would allow for much more thorougly and shorter runs especially for the coupled runs

Difficulty of installing and using our custom software at UPPMAX.

Lack of data to process, and that many of our ideas are well adapted to our current infrastructure. We would need to think bigger if we had more resources!

finally being able to do sensitivity studies, i.e. ensemble simulations

Technical assistance with porting codes to the new facility

Knowing how to scale up my current projects.

Hampered workflow

Human capacity for programming and cleaning/organizing data

How to manage the resources and keep them healthy and operating.

1. Understanding the infrastructure. 2. Getting the code compiled inside of the new infrastructure. An example: The PDC cluster at KTH has a module system and the modules have unintuitive names like "tpsl" and it took a long time to figure out which modules to load to get my program compiled (tpsl contains the sundials solver suite). Programming is hard for scientists and we basically get no support when it comes to build systems, CI, testing, debugging. So, none of our software is very reliable and sometimes fails when ported. I would appreciate people who are hired to explain things and help with code development much more than more computing power actually.

1. A customized code for taking advantage of the resources. 2. Experience from working with command-line environments.

There will be no major impediment

Not being far along enough in my research to take advantage of them. The tests to be done are not finalised yet.

There would be no major impediments. Ten times more resources would mean computational results with higher fidelity and a possibility to broaden the parameter range in our studies. Hence, taking advantage of the resources to their full capacity would not be a problem. The only small impediment that comes to mind is that my local infrastructure for long term storage of raw data would have to be increased and better structured to accommodate 10 times more data.

The generation of realistic ordered models of disordered structures for further dft calculations using program jdftx. The problem itself is rational use of available resources to maximize the ratio (results/used resources). Completely random models, obtained from SQS approach are not always the best, especially for constrained disorder e.g. metal hydrides. I'd like to master cluster expansion technique, as a solution of such issues. However I do not realize the limits of its applicability and usage for complicated systems.

Not much ... CPU-time is my the main bottleneck.

If the computational resources would consist of CPU time, I do not expect any problems in utilizing them (the code I am using has been tested on large machines and performs very well under heavy parallelization). However, my code base is not currently GPU-accelerated (work on this is ongoing), so, at least for the moment, I would not be able to take advantage of GPU computational resources.

Post-processing the data efficiently.

Learn how to use it in the right way

Having access to enough high-performance storage connected to the [cluster]

Hands-on knowledge of implementing non-parallel R solutions in a parallel context. Harnessing the powers of unix, convincing colleagues to start using these resources to initiate a topic-centered userbase large enough to get momentum enough to establish HPC as a valuable tool in our analysis pipelines. Too little knowledge of distributed collaboration and version control is another possible obstacle.

The tight time-table and competing duties/jobs

More complex ML model than what my GPU can handle

- Disk storage - Missing infrastructure/software to create ensemble statistics on the fly to omit disk storage (multiple instances of a run with different bcs)

My PhD students could run an extended set of experiments and instantly, which will improve the manuscripts to be submitted.

Restrictions for GPU cluster access on national resources. My research group develops novel HPC programming models and tools for GPU clusters. We need now and then access to some cluster with GPU-accelerated nodes for short runs for testing and evaluation. I.e., test programs and benchmarks, not applications per se, and not many core hours at all. This usage scenario is apparently not foreseen in SNIC.

Improved AI methods (but 10times is not sufficient - competitors have 1000x more than me)

Setting up workflows to deploy run input files. Ensuring best possible resource usage throughout diverse inputs.

None. I run ensembles of simulations that don't require much communication.

I have access to DGX-2 and I am mainly interested in a larger GPU resource or more time dedicated to my needs on the resource. Training on how to run code more efficiently on a GPU cluster would be beneficial.

No problem, our codes scale well.

We would be able to screen more compounds and sample more accurately.

It depends on the underlying hardware and software platform. Our research relies on large scale physics simulations based on partial differential equations, using finite element methods on unstructured meshes, for example, computational fluid dynamics. We develop our own software platform (FEniCS), but which has dependencies, e.g. linear algebra packages such as PETSc. Some type of one-sided communication is likely needed on a 10x compute system, which we develop as part of our research e.g. into hybrid MPI+PGAS parallel programming models.

Basically nothing, I am running automatic workflow which can use up all the time that is availible

Virtually no fundamental issue in terms of weak scaling, as long as I keep running on CPUs; since I need double-precision for very large systems, I won't probably benefy too much from GPUs, at least in terms of weak scaling.

Scaling, porting code to accelerators.

Data processing and handling. At present, I use Matlab on the login node of the supercomputer to process large datasets before I reduce the data to a format and size, which I can handle on my PC.

Parallelization of my problem is difficult to improve

Knowledge on how to handle large numbers of files, and storage.

Make sure my program scales. Make it use several GPUs

Manpower: analysing results from simulations takes a lot of work.

Reliance on specific software and visualization tools, as I almost only have experience with for example VASP and vesta for simulations and data analysis, respectively. I do not exactly see much of a hindrance given that the resources would support the most recent slurm version and is based on some linux system, and have some version of python/c++ compilers installed. There could be difficulties in adjusting the parallelization to fit the new resource layouts, but nothing that I would call a hindrance.

Storage speed and availability

I could scale up my problems to make them more realistic. I could get results faster. I could test new computational methods not feasible today.

Need better programming skills/time to learn

manpower with correct education

None. We have the calculations ready and waiting in queue.

The size of the problem and a restraint not to waste resources.

Then I could run even more analysis in parallel.

Finding the time to organize all experiments I want to run

I cannot foresee many problems with that, but perhaps for some specific problems I should work at parallelizing them better.

Time to set up the model e.g. acquiring and formatting the input data

transferring and maintaining the current codes/files from/to the larger computing resources.

10 times larger usually means 10 times in one direction (usually CPU power) but the same, or almost the same in others - eg RAM per core.

Increased numerical resolution. Intensified numerical and physical validation. Parametric studies.

Run experimentes in larger scale to cover more ground for publications.

We would be able to study systems at actual relevant physiological concentrations, instead of always having to argue for why we do not...

Would produce more data which needs more resources to interpret.

Limited scalability of quantum chemistry codes

- Developing protocols for parallelization - Deployment of commercial and/or own/open source software to compute resources

We are working with data-mining of very large datasets of DNA sequence data where we are trying to identify new forms of genes that make bacteria resistent to antibiotics. Our main challanges in scaling up (we currently work with datasets up to 100 Tb using our own servers) is a) help with parallelization of IO-heavy activities and b) access to the necessary computer resources that enables processing of petabytes of data (e.g. large and fast storage).

Processing/interpreting the results

Pre- and postprocessing, more the latter though. Create computational meshes is a problem but probably manageable. But to decide what data to save, how often, and how it's stored, and primarily then how to handle the large data set for extraction of data and visualisation.

The complexity of the problems we are solving in the simulation (DNS of multiphase reactive flow). For example, we have to carry out the simulation by applying simplification such as constant temperature inside particles, fixed particles, limiting the number of particles etc. With 10 times larger computational resources, we can overcome some of these limitation.

Upscaling my code and/or having personal/people to analys the data

Scaling the code to use larger resources.

Faster computations, more simulations, better publications.

*It would need to be *large nodes* on which many OpenMP threads could be runs. Usually hard to find...*

none

Having enough people in the lab to take advantage of using a large compute resource.

We need more funding for additional data collection before we can do additional computation on the collected data. Currently we have enough HPC resources for our current level of data collection.

Running real geometries

All our simulations can be speed up 10 times faster, save a lot of waiting time for numerical experiments, and possible to launch long runs.

Identifying enough relevant research/simulations to keep the resources occupied for a longer period of time.

Computational resources would no longer limit my research

Appendix B

Free-form answers to the question *What type of training are you interested in attending and at what level?.*

For reasons outlined above, a specific focus on text-based analysis would be helpful.

Python for HPC (Specific programming languages)

Most of the development is done by my team members. I have asked them to fill in the survey individually, so as a team, we are actually interested in training on all these topics (different group members in different areas)

We run hundreds-to-thousands of individual jobs. The scaling isn't a problem, but the data processing is.

It would be nice to eventually replace Intel Fortran by a programming language that is equally fast but more modern. Python is unfortunately a bit too slow. Some people in Economics have started to use Julia. In general, the parallelization is key for speed.

Massive scale genetic programming

How to design software to help users run on a SLURM system. How to run software on SLURM that uses different number of cores for different parts of its runtime without wasting core hours.*

I have very little time to attend training, unfortunately.

I'm interested in refactoring techniques since I'm facing users that often bring little programming experience but need to modify Fortran/C code, thus a de-facto speed optimization would be to improved "legibility" of code.

I work with people that know far more than I about those techniques, so there is no point in me learning more about them. For the packages I use and develop, I know the codes very well.

Building, managing and operating HPC systems platforms with code.

GPU stuff is too complicated writing it takes too much time compared to conventional computing, I would rather switch to a device with hundreds of ARM cores or similar but no dev overhead for me as a scientist. Optimizing code (say with preprocessor instructions) is a lot of work and requires more time than scientists typically have, so I have no interest in that. It's much better to have reliable code than optimized code. The by far much greater issue is: how can one person or a very small team of (e.g. two) people develop reliable code, with testing and continuous integration in a fast language (C,C++,Rust). I know just enough C to write code and also use MPI and OpenMP. I do not know how to write automated tests in C and I build my program using a hand written Makefile. There is no continuous integration, I don't even use a debugger, because I don't know how they work and what use they are. Learning these skills online is a risk because you may loose lots of time and get stuck with questions noone is willing to answer while making no progress at all on your scientific problem. I would bet that your money would be best spend to hire personnel that teaches young researchers development skills (CI, testing, build systems, debuggers, general procedures) in their given language and their specific problems. For example: I would under no circumstances go to a workshop on "continuous integration in python", that knowledge does not transfer to C.*

Many data analytics tasks are inherently embarassingly parallel, so they are frrequently realized as separate tasks or array jobs. However, their IO dependencies are more intricate, with similar concerns of data sharing vs. data independence as in a shared-memory system with a memory hierarchy. This is given limited attention in most training, and even many frameworks focus on either data distribution, with very light compute, or compute distribution, with an assumption that overall I/O (not including communication between tasks), is light.

I do offer training events for some of the above topics myself.

I think it'd be good to cover all levels (introductory-advanced) in most of these topics. However, it doesn't necessarily require scheduled workshops, but there can be on-demand resources in github etc (examples: coderefinery, futurelearn).

Efficient DFT methods: practical basics, including recent IPR methods

Advanced VASP usage

I would be interested in COARRAYs in Fortran 2018

Pytorch

Appendix C

Free-form answers to the question *Any final comments?*

In general there is a big problem about which projects are getting SNIC LARGE CPU/GPU time. In my group we need a lot of time, but CPU time from SNIC goes to those who are "well-known" as computational groups in Sweden. Basically, if you are not a guy from a gang you won't get any time regardless that your project might be more valuable than what others do. Also rules for SNIC LARGE project seem not to be applied for those well-established while in case of a new applicant for SNIC LARGE the reviewer of a proposal can find a way to give a negative answer, because nobody will check what he wrote in his review, since he considered as an expert. I think SNIC shall consider those who come out as new groups and apply for LARGE projects when they grant CPU time.

We are still experimenting a lot with GPU and ML to develop it in the right way for our needs. We are still working on pilots, but I can envision that in the very close future we will need to scale that up significantly.

Getting access to cloud infrastructure would allow the lab to save money on up-front server investment (co-localized).

My answers reflect my research group (about 7 PhD students).

As faculty, participating in training events personally is harder due to all the competing commitments on my time. However, there are many people in my team who would greatly benefit from the suggested training events, so as a group we are really enthusiastic.

I bought a 100k EUR worth of nodes for my local SNIC resource so I have been very happy with the resources available to me.

I am a very irregular user of HPC resources. It depends on the stage of the research project.

I'm missing a question about how much RAM we need. Now only focused on CPUs/GPUs. It is important in a future facility that a variety of equipment is available, and not just to participate in the useless "as many flops as possible" competition. We also need nodes with large RAM (5TB and more).

Thanks

I already access HPC through cloud - Google Colab

The more cloud-like you can make HPC access the better. Right now it's hard not to get tempted to go for large AWS/GCP grants, or budget for their use as part of project proposals. Some funders/reviewers take it as a good thing, and if not part of proposals, might look bad.

Apart from myself, training for PhD students is very valuable

I am very new to this but think this is very important and am glad it is available

As researcher in computational mathematics and stochastic simulation the size of our simulations testing new algorithms is limited by the resources and we are used to adapt our problems to that. There are no real upper limits i resources, we are used to take what we get and to find solutions that work.

Regarding 9: I don't pursue any science question per se, usage of computation hours are decided by the scientists I'm supporting General comment: background of scientists I'm working with (including myself) is science but not informatics. In other words, on average there is little programming experience and modifying code thus takes a long time. "Collaborations" like profiling/improving existing code together with SNIC personnel proved very useful. There is little to no room for dedicated courses, so an idea could be to do the supervised project work from Q15 on our own model code.

accounts and ssh access to a bash terminal is fine with me regarding access. Just more experienced developers to talk to would be nice, and I would not class those people under "support". Support is when it goes wrong. Developer advice is preventative so that it doesn't go wrong.

I think the strength of the system means very little if people do not understand how to use it. Give courses/presentations where you give actual examples that the people can understand. Otherwise you will only propagate the already disastrous state of code as it exists now.

I do not know whether benefits of access through cloud environment will worth the investment needed. I think that it may be beneficial mostly for educational purposes like workshops with all software incl.IDE installed. But I think that the clients of clouds must support very wide variety of client machines/OS including outdated ones.
I think that ordinary job queue is better for production runs of MPI programs. It will avoid say 30% performance loss due to virtualization. Also I think that it is better to set up number of cores needed, job placement, exclusive use of nodes etc in advance by hand or by scripts. In such a way it seems to be more straightforward to optimize performance and avoid competition of different tasks for ram bandwidth.

Thanks for nice considerations for us

Width of jobs listed relates to my point about training above. An array job using the same reference data set for a hyperparameter search or an ensemble, or processing of data from N individuals in N arrays, can mean single-node jobs, but hundreds of them with shared I/O dependencies.

fast access to lots of small files needed

I would likely not follow the training, but doctoral students and postdocs in my group would.

Comment to question 15: I think that a successful workshop combines many of the elements (slides, code-along, exercises, hacking in team/alone) in suitable portions.

As a complement to regular training events some kind of workshops where you can get help with your own data/problems would be very useful.

Weights & Biases has turned out to be very useful