Expanding user communities with HPC Carpentry

Alan Ó Cais*
Jülich Supercomputing Centre
Jülich, Germany
a.ocais@fz-juelich.de

Peter Steinbach
Scionics Computer Innovation GmbH
Dresden, Germany
Max Planck Institute of Molecular Cell Biology and Genetics
Dresden, Germany
steinbach@scionics.de

ABSTRACT

Adoption of HPC as a research tool and industrial resource is a priority in many countries. The use of data analytics and machine learning approaches in many areas also attracts non-traditional HPC user communities to the hardware capabilities provided by supercomputing facilities. As a result, HPC at all scales is experiencing rapid growth of the demand for training, with much of this at the introductory level.

To address the growth in demand, we need both a scalable and sustainable training model as well as a method to ensure the consistency of the training being offered. Adopting the successful training model of The Carpentries (https://carpentries.org/) for the HPC space provides a pathway to collaboratively created training content which can be delivered in a scalable way (serving everything from university or industrial HPC systems to national facilities).

We describe the ongoing efforts of HPC Carpentry to create training material to address this need and form the collaborative network required to sustain it. We outline the history of the effort and the practices adopted from The Carpentries that enable it. The lessons being created as a result are under active development and being evaluated in practice at sites in Europe, the US and Canada.

KEYWORDS

Carpentries, Software Carpentry, Education, Training, HPC

1 INTRODUCTION

Many countries are now spending substantial budgets on high performance computing (HPC) related research initiatives [1]. These initiatives are creating computational laboratories of unprecedented scale. At the same time, big data analytics, cloud computing and deep neural networks are influencing the development of HPC to the extent that the possible future convergence of HPC and big data applications is being discussed [10]. Cross-domain interest in Deep Learning alone is driving new sets of users to seek out access to the latest hardware, and in many cases this hardware is to be found in the national and regional supercomputing facilities. The explosion of interest in all of these fields brings with it many challenges, not least of these is providing researchers with adequate training so they can effectively and efficiently leverage these computational laboratories for their research.

To highlight a specific example of this growth, we consider EuroHPC (https://eurohpc-ju.europa.eu) which contains 25 participating EU states. In EuroHPC, each of the participant states is expected to have an *HPC Competence Centre* which will provide HPC services to industry, academia and public administrations. These HPC Competence Centres will be a gateway into the European HPC landscape and, in many cases, the first landing point for HPC interest in those countries. Since it is a European initiative one must strive for consistent levels of service across all states, despite greatly varying experience of the HPC domain. From a training perspective, how does one achieve this? Addressing this challenge is of critical importance to the long-tail impact of the investment in EuroHPC.

In this paper we will address the initial training requirements of a new user who is freshly exposed to HPC resources. Within academic research there is always a constant flow of early career researchers entering the field and accessing resources at all levels of the hardware pyramid. For this reason, we see the "HPC novice" profile as something that is relevant across the spectrum of HPC facilities, from institutional resources all the way up to national and international facilities.

A scalable, collaborative training model is an effective and sustainable way to tackle large increases in demand for "HPC novice" training. We propose to adopt and adapt the model developed by the Software Carpentry initiative [11] and apply it to the novice HPC learner. In Section 2, we introduce aspects of that model and why it has the potential to map well to the HPC space. In Section 3, we outline some of the training material design principles and how they influence the creation process. In Section 4, we look at two distinct evaluation processes, a review process that happens during material creation and learner evaluations that occur during/after training events. Finally, in Section 5, we consider future development efforts in light of progress and outcomes that have occurred to date.

*Authors are listed alphabetically
[1]See for example,
- The Exascale Computing Project, https://www.exascaleproject.org/;
- The EuroHPC Joint Undertaking, https://eurohpc-ju.europa.eu/;
- The Collaboration of Oak Ridge, Argonne, and Livermore (CORAL), https://www.energy.gov/downloads/fact-sheet-collaboration-oak-ridge-argonne-and-livermore-coral/;

but also comparable initiatives in China, Taiwan, Japan and India (see https://en.wikipedia.org/wiki/Exascale_computing).

© 2020 Journal of Computational Science Education
https://doi.org/10.22369/issn.2153-4136/11/1/4

2 SOFTWARE CARPENTRY AND HPC CARPENTRY

In the lessons-learned review of Software Carpentry initiative [12], the author draws a distinction between the "minority who do high-performance computing" and other scientists who use computing as a research tool. However, many who have been involved with training of novice HPC users will recognise that the philosophy of Software Carpentry to teach "researchers the computing skills they need to get more done in less time and with less pain" is just as relevant to the HPC novice, but that the scope of necessary skills is wider. The goal of HPC Carpentry is to target the skills gap as well as the technical and conceptual barriers faced by the HPC novice as they transition from desktop/server computation to HPC resources.

At it's core, Software Carpentry is a volunteer project dedicated to teaching basic computing skills to researchers. This is done by treating lessons the same way you would an open source software project: collaboratively created and free to access or use. This approach is inherently scalable as the lesson itself is a shared resource. Most importantly, Software Carpentry provides training on modern research in education, and associated evidence-based teaching practices [3]. In short, the instructor is taught how to deliver training material effectively and the community provides them with the high quality training content that they will teach. Just as importantly though, a shared lesson and a shared approach to teaching opens up the possibility to create a community around that material. Such a community is a resource in and of itself, and provides a platform for further collaboration on more complex training topics.

HPC Carpentry seeks to crystallise such a community in the HPC domain. This approach resembles the same process that has already been undertaken with the Data Carpentry and Library Carpentry initiatives[2].

2.1 The Importance of Collaboration

For some time, there has been significant interest among the HPC training community in the Software Carpentry approach to generating and delivering training content. There have already been two distinct efforts to develop the type of HPC novice material that is being considered here:

- HPC in a day (https://psteinb.github.io/hpc-in-a-day/),
- Introduction to High-Performance Computing (https://github.com/hpc-carpentry/hpc-intro/releases/tag/v9.1.2).

Further interest has included a Birds of a Feather session at SC17 [9].

HPC Carpentry is not, currently, a formal part of the Carpentries but the current development drive is being carried with the knowledge and participation of the Carpentries. At CarpentryCon 2018, HPC Carpentry had 2 sessions [4, 7] (each with ~40 participants), where much of the discussion centred around how to merge existing efforts and form a single group of collaborators to drive the lesson development forward. The creators and maintainers of the previously listed novice material were central to this discussion and have agreed to engage in a unification effort under the HPC Carpentry umbrella. This has resulted in the creation of a

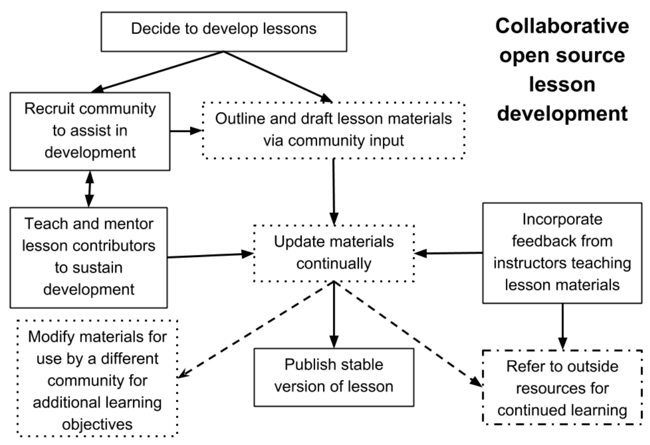
Collaborative open source lesson development
Decide to develop lessons
Recruit community to assist in development
Outline and draft lesson materials via community input
Teach and mentor lesson contributors to sustain development
Update materials continually
Incorporate feedback from instructors teaching lesson materials
Modify materials for use by a different community for additional learning objectives
Publish stable version of lesson
Refer to outside resources for continued learning

Figure 1: Collaborative open lesson development (reproduced from [6]).

`maintainers-hpc` mailing list under the Carpentries umbrella[3] to coordinate the activities of HPC Carpentry, and contributions to the lessons.

The importance of this step is perhaps best represented if we consider [6], where the authors succinctly capture collaborative lesson development in one of their figures (which we reproduce in Fig. 1). Fig. 1 emphasises the crucial step of recruiting a community to

- define the target audience
- assist in material development
- provide feedback and continuous quality control
- link to other projects or resources

among other aspects. Without a community, the continuous improvement, practicability and sustainability of the lesson is in jeopardy (since the maintenance burden falls on a single set of shoulders). Defragmenting previous initiatives within a single collaborative group was an important first step, and a signal to other interested parties that it would form a cohesive effort.

3 LESSON DESIGN

The "rules" that govern lesson integration and development are reproduced from [6] in Fig. 2.

One of the first things the interested parties did was to clarify our audience by outlining a set of *Learner Profiles* that are representative of the intended audience (Rule 1)[4]. Furthermore, we restricted the training window to be a single day since this is what many sites already do in practice for the targeted level of this material. In addition, there was lengthy discussion about what should, and should not, be included in the lesson. In particular, there is the issue of what should be considered as required prerequisite knowledge.

The most significant potential prerequisite is the UNIX shell, whether it should be incorporated or not was intensely discussed.

[2]Software Carptentry, Data Carpentry and Library Carpentry are now collectively termed *The Carpentries*, https://carpentries.org/

[3]https://carpentries.topicbox.com/groups/maintainers-hpc

[4]https://github.com/hpc-carpentry/hpc-carpentry.github.io/blob/master/why-hpc-carpentry.md#learner-profiles

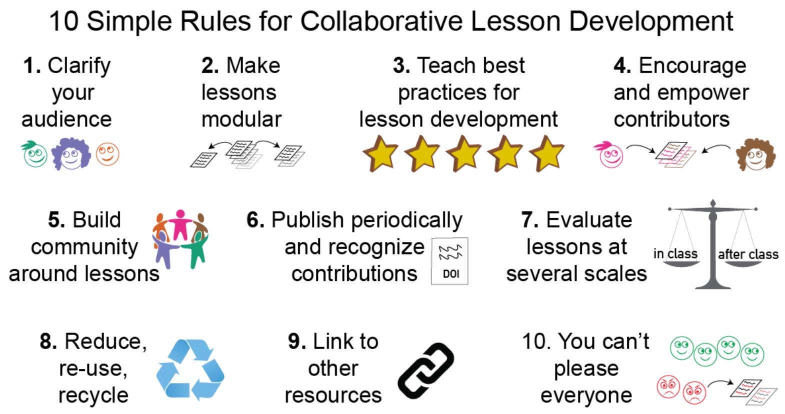
10 Simple Rules for Collaborative Lesson Development
1. Clarify your audience
2. Make lessons modular
3. Teach best practices for lesson development
4. Encourage and empower contributors
5. Build community around lessons
6. Publish periodically and recognize contributions
DOI
7. Evaluate lessons at several scales
in class
after class
8. Reduce, re-use, recycle
9. Link to other resources
10. You can't please everyone

Figure 2: 10 rules for collaborative lesson development (reproduced from [6]).

Teaching UNIX shell fundamentals is already core material within Software Carpentry [2] but there are some additional topics (such as `ssh` sessions, for example) that are specific to remote computing. Having introduced the restriction that there be only one full day to address all the material, the decision was made to split the lesson in two equal parts: the first part dealing with the shell (in the context of remote computing)[5] and the second part with the introduction to working with an HPC resource[6]. Our decision aligns with Rule 2 about modularity of lessons, it provides the instructor with an opportunity to omit the first part if the prerequisite knowledge exists among the participants. In addition, every lesson is structured into chapters. For example:

(1) Why use HPC?
(2) Logging in to a cluster with `ssh`
(3) Submitting jobs
(4) Environment modules
(5) Transferring data

By virtue of creating chapters with high independence and low to no overlap, an instructor is free to skip or remove portions of the content without loss of material coherence.

To respect Rule 3 as regards best practices, we take the Carpentries Instructor Training [1] as our standard[7]. We begin from the end: what do we want our learners to be able to do by the end of the lesson? We work backwards from this to create a *concept map* and give the lesson structure. The map also helps us decide where to introduce formative assessments to help learners commit each concept to memory. Formative assessment ties into Rule 7 of Fig. 2 to infer the lesson audience knowledge intake for a given topic. Furthermore, the HPC Carpentry community strives to implement continuous feedback loops during a workshop, as has become a standard with the Carpentries. In practice this is done through the use of an Etherpad[8] to create collaborative notes and collect questions within the group without interrupting the teaching flow. Anonymous feedback on sticky notes at the end of every half-day session is also collected.

[5]https://hpc-carpentry.github.io/hpc-shell/
[6]https://hpc-carpentry.github.io/hpc-intro/
[7]In particular, please see https://carpentries.github.io/instructor-training/05-memory/
[8]https://etherpad.org/

Lesson design is also influenced by the instructional approach of Carpentries' instructors, who use *guided practice* as the instructional tool by means of *participatory live coding*[9]: each lesson is a set of prepared/faded examples that is done together with the user.

During the lesson design and the enabling collaborative workshops [4, 7], it was repeatedly made evident, by the vigorous discussions on what to include into an introduction to HPC, that the HPC community is very heterogeneous. Expectations by traditional HPC users with a Fortran/C/C++ background and experience in shared or distributed memory parallelism were confronted at equal measure with expectations from a high-throughput community with a map-reduce based understanding of parallelism. This evidence of the convergence or divergence of HPC and big data is mapped directly into a community like HPC Carpentry. This can be considered as evidence for Rule 10, but also emphasises the challenge within HPC Carpentry to implement Rule 1 to 9 in such a environment. It is an ongoing effort to dissect clearly the central topics that HPC carpentry wants to focus on and what motivates them.

3.1 Lesson Portability

A key technical issue for lesson portability is the fact that the learner environment can vary significantly between HPC sites. For example, a truly portable lesson should be possible to be configured for different resource managers, scheduler queue configurations, MPI launchers, ...

Since the Carpentries lessons are delivered via GitHub pages using Jekyll[10], we leverage liquid templates[11] to enable portability. For example, in the YAML configuration file for the Jekyll website, we can set a variable with:

```
scheduler: "slurm"
```

which can then be referenced throughout the lesson material with:

```
{{ site.scheduler }}
```

Lesson portability then means adding a YAML configuration file appropriate for the target site. This approach is based on that originally taken in [8].

[9]see https://carpentries.github.io/instructor-training/14-live/
[10]see https://help.github.com/en/articles/about-github-pages-and-jekyll
[11]https://jekyllrb.com/docs/liquid/

It should be noted that this also influences lesson design since we must make an additional effort to avoid referencing specific features of tools that may not be available in all of the expected configurations. While this increases the burden for contribution and makes the lesson source code harder to read, we choose to embrace the heterogeneity of HPC as it exists and thus enlarge the number of possible lesson users.

4 EVALUATION

There are two different types of evaluation of the lesson material: the evaluation of contributions to the lessons themselves and the evaluation from learners in workshops where the lessons are delivered.

In the case of evaluation of contributions, the contributions themselves are made via Pull Requests[12] to the lesson repositories. These contributions are reviewed by lesson maintainers and by any interested parties who wish to engage in discussing the contribution. The intention is that this process is not only open (*anyone* can make a Pull Request) but also provides a platform for exposing new contributors to the design principles used in the lessons. It also gives us a mechanism by which we can follow Rule 4 of Fig. 2: we can actively encourage new contributors and incorporate them into the community.

From the learners, there are three methods of gathering feedback[13]:

(1) pre- and post-workshop surveys;
(2) minute cards - *anonymous* notes gathered at lunch time and the end of the day which have one positive and one negative comment;
(3) the "one up, one down" technique - the instructor asks the learners to alternately give one positive and one negative point about the day, without repeating anything that has already been said.

Summaries of these evaluations are communicated during the meetings of the lesson collaborators. The surveys help us to assess whether the learning goals have been achieved, while the minute cards and the "one up, one down" technique are usually an evaluation of the instructor as much as the lesson content. Where appropriate, matters arising from the evaluations are raised as issues in the relevant repository.

5 CONCLUSIONS

The "HPC novice" lessons https://hpc-carpentry.github.io/hpc-shell and https://hpc-carpentry.github.io/hpc-intro are both undergoing significant development currently. These lessons have been delivered by a number of people within the HPC Carpentry community and initial feedback has been largely positive. There has not, as yet, been a more objective evaluation of the lessons as the lessons are still (in software terms) in an "alpha" state. The immediate goal is to continue the development process and stress-test the lessons with new learners, instructors and teaching environments to be found in Canada, the US and Europe (home to many of the collaborators).

While doing so, a community of contributors is being created, nourished and expanded. It is the development of this community, based on open and transparent governance structures (motivated by [5]), that is key to generating the capacity and energy to sustainably develop HPC training material based on modern teaching methods and flexible enough to be adopted by the heterogeneous HPC community of the 21st century. This community effort is, however, unfunded volunteer effort which restricts the possibility of providing concrete timelines for achieving lesson maturity. The lesson repositories on GitHub are the best place to keep track of recent developments and contributions[14].

Much in the same way that Software Carpentry developed, there is a desire from collaborators to ultimately move beyond novice lessons to more advanced topics. Priority has been given to novice content because it is common ground before we approach the branching of learner profiles expected when we consider more advanced topics (software users, software developers, hardware-specific training, domain-specific training,…).

ACKNOWLEDGMENTS

Alan Ó Cais acknowledges support from the European Union's Horizon 2020 research and innovation program, under grant agreement No. 676531 (project E-CAM) and grant agreement No. 823964 (project FocusCoE).

REFERENCES

[1] Aron Ahmadia, James Allen, Piotr Banaszkiewicz, Erin Becker, Trevor Bekolay, John Blischak, Andy Boughton, Erik Bray, Abigail Cabunoc Mayes, Steve Crouch, Neal Davis, Matt Davis, Jonah Duckles, Rémi Emonet, FÃĺlix-Antoine Fortin, Ivan Gonzalez, Chris Hamm, Michael Hansen, Rayna Harris, Felix Henninger, Konrad Hinsen, Amy Hodge, Mike Jackson, W. Trevor King, Justin Kitzes, Christina Koch, Tom Liversidge, FranÃğois Michonneau, Bill Mills, Lex Nederbragt, Aaron O'Leary, Elizabeth Patitsas, Aleksandra Pawlik, Fernando Perez, Jon Pipitone, Timothée Poisot, Ariel Rokem, Raniere Silva, Tracy Teal, Tim TrÃűndle, Fiona Tweedie, Jill-Jênn Vie, Jordan Walker, Alistair Walsh, Belinda Weaver, Ethan White, Chandler Wilkerson, Jason Williams, Greg Wilson, and Anelda van der Walt. 2016. Software Carpentry: Instructor Training. (June 2016). https://doi.org/10.5281/zenodo.57571

[2] Inigo Aldazabal Mensa, Harriet Alexander, James Allen, Areej Alsheikh-Hussain, Daniel Baird, Piotr Banaszkiewicz, Pauline Barmby, Rob Beagrie, Trevor Bekolay, Evgenij Belikov, Jason Bell, Kai Blin, John Blischak, Simon Boardman, Maxime Boissoneault, Jessica Bonnie, Andy Boughton, Ry4an Brase, Amy Brown, Dana Brunson, Orion Buske, Abigail Cabunoc Mayes, Daniel Chen, Kathy Chung, Gabriel A. Devenyi, Emily Dolson, Jonah Duckles, Rémi Emonet, David Eyers, Filipe Fernandes, Hugues Fontenelle, Francis Gacenga, Matthew Gidden, Ivan Gonzalez, Norman Gray, Varda F. Hagh, Michael Hansen, Emelie Harstad, Adina Howe, Fatma Imamoglu, Damien Irving, Mike Jackson, Emily Jane McTavish, Michael Jennings, Dan Jones, Alix Keener, Kristopher Keipert, Tom Kelly, Jan T. Kim, W. Trevor King, Christina Koch, Bernhard Konrad, Sherry Lake, Doug Latornell, Philip Lijnzaad, Eric Ma, Joshua Madin, Camille Marini, Kunal Marwaha, Sergey Mashchenko, FranÃğois Michonneau, Ryan Middleson, Jackie Milhans, Bill Mills, Amanda Miotto, Sarah Mount, Lex Nederbragt, Daiva Nielsen, Aaron O'Leary, Randy Olson, Adam Orr, Nina Therkildsen, Kirill Palamartchouk, Adam Perry, Jon Pipitone, Timothée Poisot, Hossein Pourreza, Timothy Povall, Adam Richie-Halford, Scott Ritchie, Noam Ross, Halfdan Rydbeck, Mahdi Sadjadi, Pat Schloss, Bertie Seyffert, Genevieve Shattow, Raniere Silva, Sarah Simpkin, John Simpson, Byron Smith, Nicola Soranzo, Ashwin Srinath, Daniel Standage, Meg Staton, Peter Steinbach, Marcel Stimberg, Bartosz Telenczuk, Florian Thoele, Tiffany Timbers, Stephen Turner, Jay van Schyndel, Anelda van der Walt, David Vollmer, Jens von der Linden, Andrew Walker, Josh Waterfall, Ethan White, Carol Willing, Greg Wilson, Donny Winston, Lynn Young, and Lee Zamparo. 2016. Software Carpentry: The Unix Shell. (June 2016). https://doi.org/10.5281/zenodo.57544

[3] Susan A. Ambrose, Michael W. Bridges, Michele DiPietro, Marsha C. Lovett, and Marie K. Norman. 2010. *How Learning Works: Seven Research-Based Principles for Smart Teaching*. Jossey-Bass. https://www.amazon.com/How-Learning-Works-Research-Based-Principles/dp/0470484101

[12] https://help.github.com/en/articles/about-pull-requests
[13] https://carpentries.github.io/instructor-training/06-feedback
[14] https://hpc-carpentry.github.io/hpc-shell and https://hpc-carpentry.github.io/hpc-intro

[4] Alan Ó Cais and Daniel Smith. 2018. Breakout 8: HPC Carpentry. https://github.com/carpentries/carpentrycon/tree/master/CarpentryCon-2018/Sessions/2018-05-31/05-Breakout-8-HPC-Carpentry. (2018). Breakout session at CarpentryCon 2018.

[5] The Carpentries. 2019. Governance. https://carpentries.org/governance/. (2019).

[6] Gabriel A. Devenyi, Rémi Emonet, Rayna M. Harris, Kate L. Hertweck, Damien Irving, Ian Milligan, and Greg Wilson. 2017. Ten simple rules for collaborative lesson development. *CoRR* abs/1707.02662 (2017). arXiv:1707.02662 http://arxiv.org/abs/1707.02662

[7] Christina Koch and Peter Steinbach. 2018. Workshop 5: HPC Carpentry. https://github.com/carpentries/carpentrycon/tree/master/CarpentryCon-2018/Sessions/2018-06-01/05-Workshop-5-HPC-Carpentry. (2018). Workshop at CarpentryCon 2018.

[8] Peter Steinbach, Aaron O'Leary, Abigail Cabunoc, Andy Boughton, Ashwin Srinath, Bill Mills, Francois Michonneau, Greg Wilson, James Allen, John Blischak, Jon Pipitone, Michael Hansen, Olav Vahtras, Piotr Banaszkiewicz, Raniere Silva, RÃĺmi Emonet, Stephan Janosch, TimothÃĺe Poisot, Toby Hodges, Trevor Bekolay, and W. Trevor King. 2019. HPC in a day. (March 2019). https://doi.org/10.5281/zenodo.2612065

[9] Andrew Turner, Christina Koch, Tracy Teal, Robert Freeman Jr, Chris Bording, and Martin Callaghan. 2017. HPC Carpentry - Practical, Hands-On HPC Training. https://sc17.supercomputing.org/index.html%3Fpost_type=page&p=5407&id=bof125&sess=sess359.html. (2017). BoF session at SC17.

[10] Theo Ungerer and Paul Carpenter. 2018. Eurolab-4-HPC Long-Term Vision on High-Performance Computing. *CoRR* abs/1807.04521 (2018). arXiv:1807.04521 http://arxiv.org/abs/1807.04521

[11] Greg Wilson. 2010. Software Carpentry web site. http://software-carpentry.org. (2010). Main web site for Software Carpentry, replacing http://swc.scipy.org.

[12] G Wilson. 2016. Software Carpentry: lessons learned [version 2; peer review: 3 approved]. *F1000Research* 3, 62 (2016). https://doi.org/10.12688/f1000research.3-62.v2