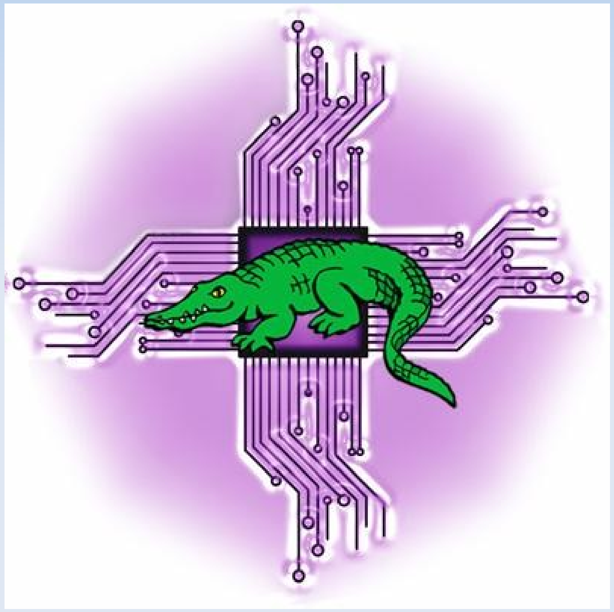
Programming Principles in a High Performance Computing Environment



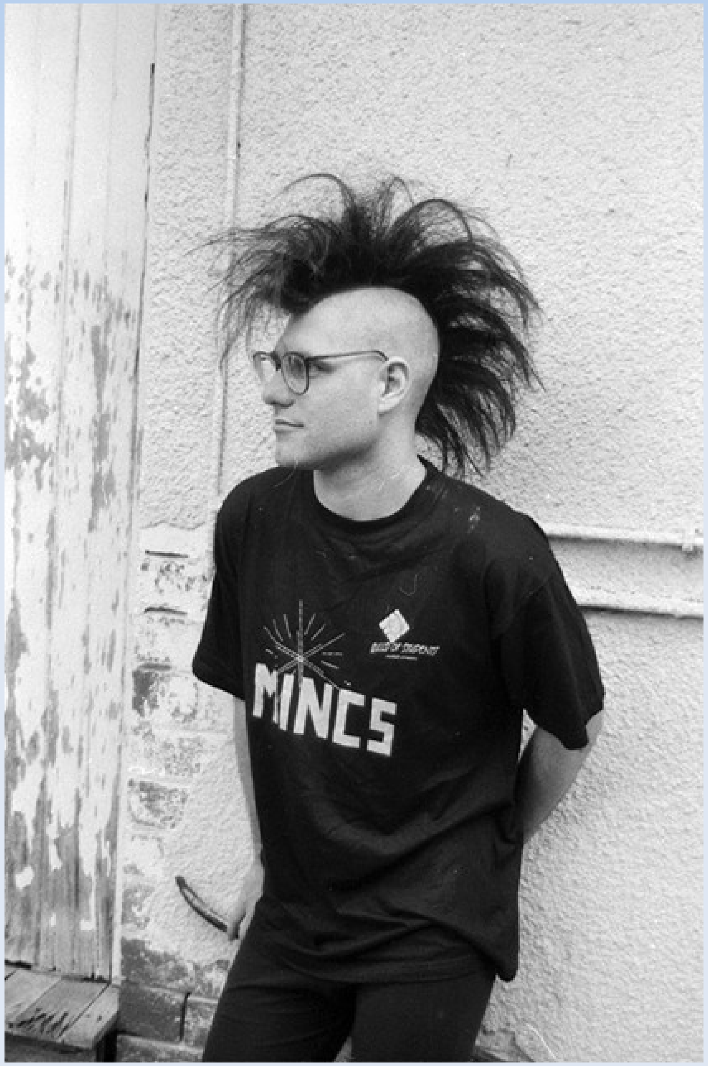
New Zealand Research Software Engineering Conference, Sept 2025

Research Computing Services, University of Melbourne

lev@levlafayette.com

About Your Speaker

- Lev is a Senior HPC DevOps engineer and the HPC Solutions Team Leader at the University of Melbourne, where he has worked for the past ten years. Prior to that, he worked at the Victorian Partnership for Advanced Computing in a similar role.
- He has presented at several conferences in Aotearoa New Zealand including eResearchNZ (2025, 2022, 2021, 2015, 2011), Multicore World (2020, 2017, 2016), IEEE eScience (2016) and at many other locations. See: http://levlafayette.com/papers
- An incorrigible collector of degrees and with a wide range of interests, he currently is completing his ninth degree (a PhD) in climatology, economics, and international torts. His academic post-nominals currently reads: BA (Hons), GradCertPM, GradCertAdult&TertEd, GradDipAppPsych, MBA, MSc, MHEd, MCCSAP
- He does not always refer to himself in the third person.



Initial Forays in HPC Training

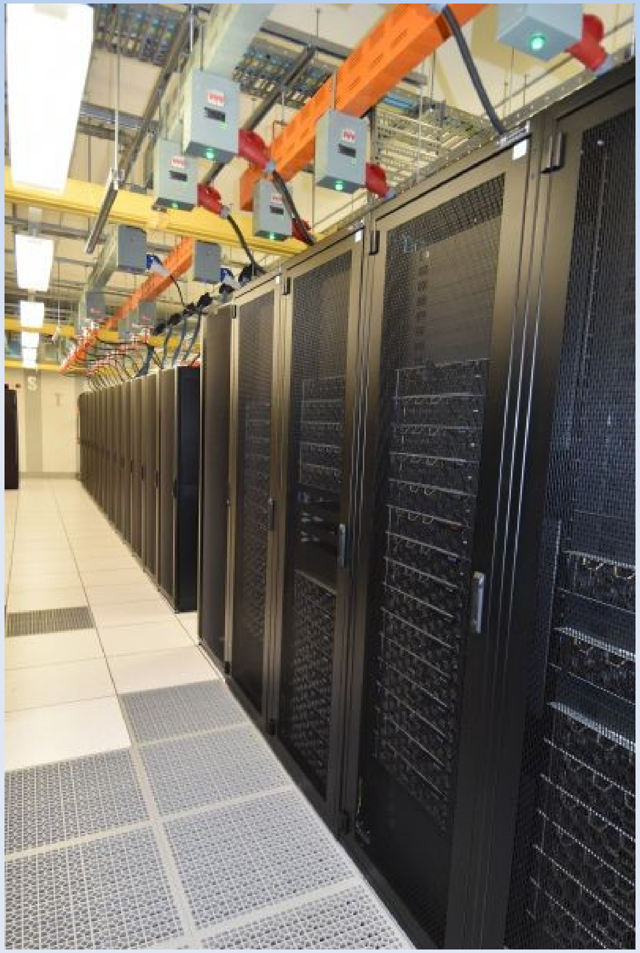
- VPAC's HPC training initially consisted of short workshops on job submission and short MPI programs. As data volumes and problem complexity increased, more researchers required access to HPC systems. The original VPAC model assumed too much for the beginner, and didn't provide enough for the advanced user.
- New course material was developed which included greater differentiation in courses, smaller class sizes, greater interaction with class and session plans. Attention to advanced adult education requirements, including a feedback system (ISO requirement).
- Introduction (basic CLI Linux, file transfers, environment variables, job submission and PBS scripts, Grid submissions), Intermediate (more advanced CLI, regular expressions, sed/awk, complex job submission, compiling), and Advanced (MPI with C and FORTRAN, messages and communicators, collective communicators, derived datatypes, dynamic message sizes, profiling).

Advanced Adult Education

- Adult education is significantly different to childhood education and, within that, the needs of young undergraduates differs to that of older graduates; a distinction between training, teaching, and researching.
- A core concept in adult and higher education is the idea of lifelong learning, with competing interests and motivations between proximal and distal goals. The individuals involved are part of the construction of knowledge; consider presage (requirements), process (their implementation), and product (subject matter).
- The preferred result is a level of learner autonomy and self-efficacy, generated from regulation and direction. This is achieved from a number of intrinsic and extrinsic motivational factors.
- Delivery should make use of discipline-based learning styles. For computer use, connectivism (e.g., pair programming) and direct usage ("yield to the hands-on imperative").

UniMelb HPC Training

- The University of Melbourne runs regular day-length onboarding workshops. The two fundamental workshops are "Introduction to Linux and High Performance Computing" and "Advanced Linux and Shell Scripting for HPC". Like “software carpentry” but for HPC.
- With these recommended prerequisites there are several other workshops available: "Parallel and High Performance Python", "Regular Expressions on HPC", "Parallel Programming", "GPU Programming", "From Spartan to Gadi", "Mathematical Applications and Programming", "Data and Databases on HPC".
- Plus specialist versions of courses for bioinformatics, mechanical engineering, and neuroscience. Guest lectures and assignment reviews for Master’s-level course, “Cluster and Cloud Computing”, Researcher Tech presentations, Spartan Champions
- Workshops have been conducted in-person and via Zoom. Access is provided to text information, videos, and extensive job submission examples. Presentations include formative spot questions and with extensive opportunities for researchers to elaborate on their own issues.



HPC Programming Content

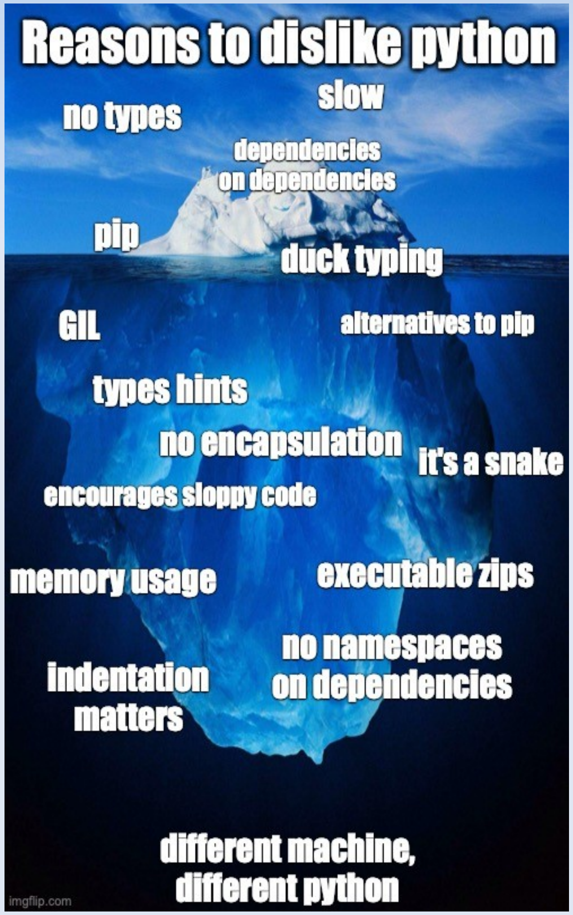
- **Programming principles presented within workshops with structured and dependent content.**
- **Introductory (job submission) workshops emphasis on high-level architecture considerations: command-line for performance, toolkit-pipeline approach, throughput, environment modules and output variation, internode vs intranode performance, scaling with MPI applications, job arrays and dependencies, job testing and output logs, data management.**
- **Intermediate (shell scripting, RegEx, Python) workshops emphasise the ability to automate, code reuse, structured and procedural code, formatting and style (consistency, simplicity, and clarity), extensions for parallelisation (e.g., Python's multithreaded, multiprocessing, and MPI4Py), shell (bash) and regex scripting (sed, awk, perl).**
- **Advanced (Python, OpenMP and MPI, GPGPU) workshops take a deeper dive into performance of different languages, datatypes, data structures, multithreaded and message-passing programming, directives vs APIs, race conditions, use of profiling and debugging tools, unit and functional testing.**

HPC Programming Challenges

- Challenges are contextual to our time and culture. Challenges are conceptual and practical. Challenges are structured and with dependencies!
- Introductory challenges arise from near-exclusive exposure to computing through GUI/WIMP interfaces and conventional desktop operating systems, feature-rich enterprise software, authoritative educational systems. Conceptual difficulties include relationship between shell and GUI, need for robustness an simplicity, local and remote systems, engagement and responsibility.
- Intermediate challenges arise from a lack of experience in converting workflows into code, overwhelming number of applications that do the same or similar things ("There's more than one way to do it"), not adopting well-established computer science conventions.
- Advanced challenes arise from adaptability, functionality and popularity of Python and the lack of polyglot programming, "fire and forget" and loss of maintenance, race conditions and locks, lack of separation of concerns, cognitive load, understanding trade-offs and abstractions..

Pithy Quotations to Remember

- **"On two occasions I have been asked, – 'Pray, Mr. Babbage, if you put into the machine wrong figures, will the right answers come out?'.. I am not able rightly to apprehend the kind of confusion of ideas that could provoke such a question." (Charles Babbage, 1864)**
- **"Premature optimization is the root of all evil" (Donald Knuth, 1974)**
- **"Simplicity is a prerequisite for reliability" (Edsger W. Dijkstra, 1975)**
- **"Controlling complexity is the essence of computer programming." (Brian Kernighan, P. J. Plauger, 1976)**
- **"Between here and the satellite there were a very large number of nanoseconds" (Grace Hopper, 1986)**
- **"If you have a problem and you think awk is the solution, then you have two problems." (David Tibrook, 1989)**
- **"Software gets slower faster than hardware gets faster." (Niklaus Wirth, 1995)**
- **"Always implement things when you actually need them, never when you just foresee that you [will] need them." (Ron Jeffries, 1998)**
- **"C is a razor-sharp tool, with which one can create an elegant and efficient program or a bloody mess." (Brian Kernighan, 1999)**
- **"Every piece of knowledge must have a single, unambiguous, authoritative representation within a system" (Andy Hunt and Dave Thomas, 1999)**
- **"There should be one-- and preferably only one --obvious way to do it." (Tim Peters, 1999)**
- **"Wrong creates unnecessary work, impossible situations and major failures. Wrong is evil, and it must be defeated." (Jeff Ello, 2009)**
- **"Simplicity enables partitioning (horizontal separation between components) and stratification (vertical separation)." (Rich Hickey, 2011)**



The Scoreboard

- The provision of HPC systems correlates with research output (Apon et al, 2010). HPC generates excellent return-on-investment (Joseph, et al, 2013); about $44 in new income or cost-savings per $1 invested (almost entirely as positive externalities).

- VPAC study from change of cluster, usage hours, and divergence in training hours.

Year	RMIT	La Trobe
2012	1,729,837h	1,719,554h Tango
2013	8,108,695h	3,301,052h Trifid
2014	9,760,919h	4,964,297h Trifid

  Trifid RMIT enrolments 229 La Trobe enrolments 29

- At UoM in 2023, at least 54.14% of cluster utilisation measured by job submission was conducted by users after receiving training.

- Get the training right! Use people who are qualified and experienced in advanced adult education and HPC (no, there’s not many!)

- Institutions that do not invest in HPC and user education have *will not survive*.

- Future directions in our training programme includes adding introductory content to shorter online modules, more extensive “Spartan Champions” lectures (see Dan Tosello’s talk), more specialist workshops and mentoring of projects.

(Some) References

- Slide #1 image of integrated crocodile by Vicky Jankowski, 2002
- Slide #2 image of Lev Lafayette, 1993
- Slide #3 image from Craig West, VPAC, 2015
- Slide #4 image from Code Fellows, nd
- Slide #5 image of Spartan HPC, 2020
- Slide #6 image of Grace Hopper from Lyn Gilbert, 1978.
- Slide #7 image of Kalinga style Ganesha with rat below from 700 CE, Someshwara Temple, Mukhalingam, Andhra Pradesh, by G.N. Subrahmanyam, 2021.
- Slide #8 image of "Reasons to Dislike Python", source unknown.
- Knuth (1968 and continuing), *The Art of Computer Programming*
- Gropp, Lusk, and Skjellum (1994), Using MPI: portable parallel programming with the message-passing interface
- Kernighan and Pike (1999), *The Practice of Programming*
- Hunt and Thomas (1999), *The Pragmatic Programmer: From Journeyman to Master*
- Oram and Wilson (eds) (2007), *Beautiful Code*

THANKS FOR WATCHING
& LISTENING PATIENTLY
makeameme.org