Home

Welcome to the knowledge pool for the danish EuroHPC competence center. Here you will find best practices, material about HPC, HPDA and AI software and hardware, Analysis of the danish national HPCs infrastructure, relevant resources and literature. All the material can be found in pdf form for offline consultation by clicking on the symbol on the top-right corner of this page.

This knowledge pool is meant to be available both to completely new users and advanced/expert users. If you are a new user, visit the page *HPC 101* to learn the basics about High Performance Computing and get up to speed. Facilitation in accessing HPCs and dissemination activities are offered - if any is needed, feel free to contact samuele@chem.au.dk.

If you desire a specific topic or material to be covered, or need any type of assistance related with the topics of the knowledge pool, contact the HPC facilitator at samuele@chem.au.dk.

This documentation is part of the EuroHPC Competence Center in Denmark, managed by DeiC.dk.

Basics of HPC

HPC (High Performance Computing) consists in clustering together a large amount of computing hardware, so that a large amount of operations can be executed at once. A supercomputer consists of different types of hardware. In general, hardware is structured in this hierarchy:

- **CPU**, the unit that can execute a single sequence of instructions. A CPU can consists of multiple cores, so that multiple chains of instructions can be executed independently.
- **Node**, one of the computers that are installed into an HPC system.
- **Cluster**, a group of nodes that are connected together and therefore able to communicate and work together on a single task.

Moreover, a storage section connected with all with one or more types of storage hardware is present in an HPC. A **node** can consists of only one or more CPUs and some RAM memory. There are other types of nodes containing different hardware combinations. The most common hardware that can be found in a node beyond RAM and CPUs is:

- **GPU**, a graphic card. This type of hardware was used for gaming and graphics softwares, but it has building up a lot of computational power, and is particularly indicated for specific types of linear algebra operations that requires repeating the same task on hundreds of parallel processes. Nvidia and AMD are the main GPU producers.
- **FPGA**, a programmable piece of hardware that can do specific operations many times faster than the other available solutions. It can be used to accelerate specific processes that are usually carried out by CPU calculations.

Access an HPC

An HPC allows for many users to log into the system at the same time, and to use part of those resources, usually after they are assigned by an administrator to each users (so that using more resources than assigned will result in stopping whatever software is executed at that time). In Denmark, you have two ways of logging into an HPC: the first is through an user-friendly interactive interface, the second is through a classic command line, that requires some knowledge of the UNIX shell language (here a good introduction to the linux shell).

Usually, a user gets assigned

- a number of CPUs and eventually GPUs/FPGAs/...
- an amount of RAM
- an amount of total time those resources can be used

To use a danish HPC, you can get in contact with the local front office or with samuele@chem.au.dk (HPC facilitation responsible), and get help in submitting a request to obtain resources.

What can I use an HPC for

HPCs have a large computational power, but this does not mean they are only to be used for large scale projects. You can indeed request entire nodes as well as a single CPU with some GB of RAM. The danish HPCs are available for any academic application:

- research projects
- student exercises in classroom teaching/lecturing
- student projects

Students are not authorized to ask for resources. It will be responsibility of the lecturer/professor to obtain resources through the front office or facilitator. Any student can then be invited/authorized in accessing the project whose resources have been allocated to.

Best practices for HPC

This page lists some useful best practices to keep in mind when coding and running applications and pipelines on an HPC.

Code coverage, testing, continuous integration

Every time we code, testing is a concern and is usually performed by the coder(s) regularly during the project. One can identify some basic main types of test:

Regression test

Given an expected output from a specific input, the code is tested to reproduce that same output.

Unit test

Tests the smallest units of the software (e.g. single functions) to identify bugs, especially in extreme cases of inputs and outputs

Continuous integration

A set of tests for the software runs automatically everytime the software code is updated. This is useful to spot bugs before someone even uses the code.

More things one might need to test are the performance/scalability of the code, usability, response to all the intended types of input data.

Unit and regression test can be useful, but at some point not really feasible, since the code can scale to be quite large and complex, with a lot of things to control. It is thus a good practice to use continuous integration, and implement simple but representative tests that cover all the code, so that bugs can be spotted often before the final users do that. Code coverage tools to implement such tests exists for several programming languages, and also for testing code deployed on Github version control.

Links

Link	Description
pyTest	a package to test `python` code
Cmake	to test both `C`, `C++` and `Fortran code`
Travis CI	tool for continuous integration in most of the used programming languages. Works on Git version control.
covr	test coverage reports for R

Code styling

An important feature of a computer code is that it is understandable by other people reading it. To make this happen, a clean and coherent style of coding should be used in a project. Some languages have a preferred coding style, and in some GUI those styling rules can be set to be required. One can also use its own coding style, but it should be one easily readable by others, and it should be the same style over the whole project.

Links

Link	Description
styleguide	Google guide for coding styles of the major programming languages
awesome guidelines	A guide to coding styles covering also documentations, tools and development environments
Pythonic rules	Intoduction to coding style in python.
R style	A post on R coding style

Containerized applications

In this page we show the benefits of project and package managers, that are a way of organizing packages in separated environments. However, a higher degree of isolation than environments can be achieved by containerization. By containerizing, a user can virtualize the entire operating system, and make it ready to be deployed on any other machine. One can for example deploy a container without the need of installing anything on the hosting machine! Note that containers are a different concept from Virtual Machines, where it is the hardware being instead virtualized.

Links

Link	Description
Docker	An open source widespread container that is popular both in research and industry
Docker course	A course to use Docker, freely hosted on youtube
Docker curriculum	Beginner introduction to docker
Docker basics	Intoduction tutorials to Docker from the official documentation page
Singularity	Singularity is another containerization tool. It allows you to decide at which degree a container interacts with the hosting system
Singularity tutorial	A well done Singularity tutorial for HPC users
Singularity video tutorial	A video tutorial on Singularity
Reproducibility by containerization	A video on reproducibility with Singularity containers

Documentation

When creating a piece of software, it is always a good idea to create a documentation explaining the usage of each element of the code. For packages, there are softwares that create automatically a documentation by using functions´ declarations and eventually some text included into them as a string.

Links

Link	Description
MkDocs	A generator for static webpages, with design and themes targeted to documentation pages, but also other type of websites. This website is itself made with MkDocs.
mkdocstrings	Python handler to automatically generate documentation with MkDocs
pdoc3	A package who creates automatically the documentation for your coding projects. It is semi automatic (infers your dependencies, classes, ... but adds a description based on your docstrings)
pdoc3 101	How to run pdoc to create an html documentation
Roxygen2	A package to generate `R` documentation - it can be used also with `Rcpp`
Sphinx	Another tool to write documentation - it produces also printable outputs. `Sphinx` was first created to write the `python` language documentation. Even though it is a tool especially thought for `python` code, it can be used to generate static webpages for other projects.

Documents with live code

Programming languages like `python` and `R` allows users to write documents that contain text, images and equations together with executable code and its output. Text is usually written using the very immediate markdown `language`. Markdown files for `R` can be created in the GUI `Rstudio`, while `python` uses `jupyter notebooks`.

Links

Link	Description
Introduction to Markdown	Markdown for `R` in `Rstudio`
Jupyter notebooks	create interactive code with `python`. You can write `R` code in a jupyter notebook by using the `python` package rpy2

Package/Environment management systems

When coding, it is essential that all the projects are developed under specific software conditions, i.e. the packages and libraries used during development (dependencies) should not change along the project's lifetime, so that variations in things such as output formats and new algorithmic implementations will not create conflicts difficult to trace back under development. An environment and package manager makes the user able to create separated frameworks (environments) where to install specific packages that will not influence other softwares outside the environment in use. A higher degree of isolation can be achieved through containers (see the related voice in this page).

Links

Link	Description
Conda	an easy to use and very popular environment manager
Getting started with conda	Introduction to `conda` setup and usage from the official documentation
Conda cheat sheet	Quick reference for `conda` usage
YARN	An alternative to `conda`

Many short jobs running

Everytime a job is submitted to the job manager (e.g. SLURM) of a computing cluster, there is an overhead time necessary to elaborate resource provision, preparation for output, and queue organization. Therefore it is wise to create, when possible, longer jobs. One needs to find the correct balance for how to organizing jobs: if these are too long and fail because of some issue, than a lot of time and resources have been wasted, but such problem can be overcome by tracking the outputs of each step to avoid rerunning all computations. For example, at each step of a job outputting something relevant, there can be a condition checking if the specific output is already present.

Massive STDOUT outputs

Try to avoid printing many outputs on the standard output STDOUT. This can be problematic when a lot of parallel jobs are running, letting STDOUT filling all the home directory up, and causing errors and eventual data loss. Use instead an output in software-specific data structures (such as `.RData` files for the `R` language) or at least simple text files.

Packaging a coding project

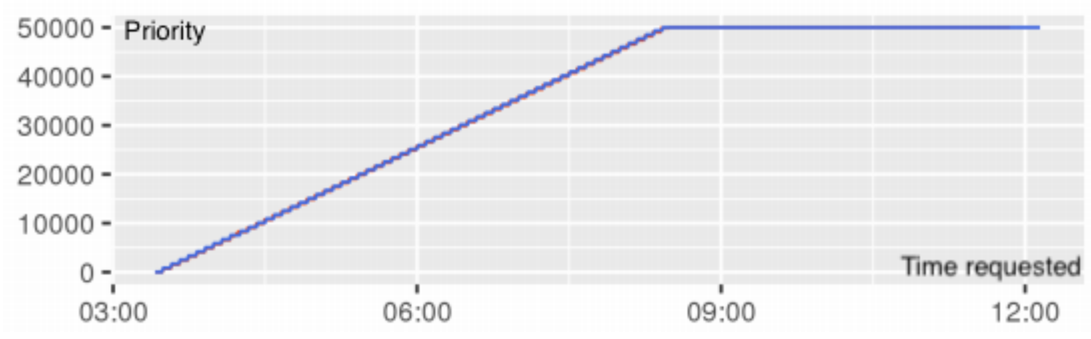
When coding a piece of software in which there are multiple newly implemented function, it can be smart to organize all those functions as a package, that can be reused and eventually shared with ease. Such a practice is especially easy and can be mastered very quickly for coding projects in `python` and `R`.

Links

Link	Description
pyPA	`python` packaging user guide
R package development	Develop an `R` package using `Rstudio`

Pipelining and submitting jobs in SLURM

SLURM is a job scheduler. It allows a user to specify a series of commands and resources requirements to run such commands. Slurm does consider the job submission on an HPC system together with all the other jobs, and prioritize them according to the resources requirement and the available computational power.



In figure above, the priority assigned to a SLURM job when the requested time increases, by keeping the memory and CPUs fixed. Decreased priority has higher values. Adapted from *A Slurm Simulator: Implementation and Parametric Analysis. Simakov et al 2017.*

The danish national HPCs, and most of the other EuroHPC supercomputers, use Slurm as job manager.

Links

Link	Description
SLURM example 1 and SLURM example 2	Some examples of how to make a Slurm script to submit a job from the danish HPC GenomeDK and from Princeton Research Computing.
Gwf, a simple python tool to create interdependent job submissions	Gwf, developed at the University of Aarhus, makes it easy to create Slurm jobs and organize them as a pipeline with dependencies, using the python language (you need python 3.5+). You get to simply create the shell scripts and the dependencies, without the complicating syntax of Slurm. The page contains also a useful guide.

Version control

Version control is the tracking of your development history for a project. This allows multiple people working on the same material to keep changes in sync without stepping over each other's contributions. Version control tools allow to commit changes with a description, set up and assign project objectives, open software issues from users and contributors, test automatically the code to find bugs before users step into them. Version control is useful for both teams and single users, and it is a good practice to have version control as a standard for any project.

Links

Link	Description
GitHub	the most used tool for version control
Github 101	quick introduction to get started on Github
GitLab and BitBucket	Two other popular alternatives to `Github`

Software for HPC

This page contains useful links to tutorial, learning resources, benchmark and scripts for softwares usable on HPCs. Softwares are separated into broad scientific/application categories. Each tool has small numbers on its right side, indicating on which national HPC types those softwares are already installed. The symbol 🐍 means that the software is available through the conda package manager for installation, for example, on the GenomeDK Type2 HPC. Remember, you can always contact an HPC front office to inquire about the installation of packages that you need for your applications.

ML and AI

tensorflow 1️⃣ 🐍

`Tensorflow` is the Google's open source library used to develop, train and deploy machine learning models. Tensorflow is implemented for `python`, but now also available in `R`

Links

Link	Description
Tensorflow homepage	Tensorflow home page. Contains many tutorials and howtos.
Learning material	Books and courses in machine learning using tensorflow
R guide to tensorflow	How to install and use tensorflow in `Rstudio`

pyTorch 1️⃣ 🐍

`Pytorch` is another Machine Learning framework created mainly by Facebook.

Links

Link	Description
PyTorch homepage	The official `pyTorch` homepage, contains tutorials to learn how to use the framework
PyTorch book	A free book about deep learning with `pyTorch`
PyTorch for deep learning	A full video tutorial on `PyTorch`

Astrophysics and Cosmology

COSMOMC 1

CosmoMC is a Fortran 2008 Markov-Chain Monte-Carlo (MCMC) engine for exploring cosmological parameter space, together with Fortran and python code for analysing Monte-Carlo samples and importance sampling (plus a suite of scripts for building grids of runs, plotting and presenting results). The code does brute force (but accurate) theoretical matter power spectrum and Cl calculations with CAMB.

Links

Link	Description
CosmoMC homepage	The official homepage contains the documentation of `CosmoMC`
Introduction to CAMB and CosmoMC	A presentation to introduce `CosmoMC` and `CAMB`
Tutorial for CosmoMC	A lecture with usage example of `CosmoMC`
CosmoMC Paper	Reference paper of `CosmoMC`

Big data and HPDA

KNIME 1

A tool to easily read and transform data, model and visualize it, integrate data science practices and produce insights.

Links

Link	Description
Getting started	A practical example of data reading and manipulation
KNIME platform presentation	Slides listing the KNIME properties and tools
FORUM	Koin the community of users and ask/answer questions and problems
KNIME learning	A wide range of learning material to use KNIME proficiently

TRACER 1️⃣ 🐍

Tracer is graphical tool for visualization and diagnostics of MCMC output. It can read output files from MrBayes and BEAST.

Links

Link	Description
TRACER documentation	Home page of `Beast2`, of which `tracer` is a tool. It contains tutorials and documentation.

Bioinformatics

ANGSD 🐍

A command line tool that integrates many different tools to analyze NGS data efficiently through native C++ implementation, multithreading support and calculation of genotype likelihoods instead of genotype calling. It is easily installed by downloading it from the `GitHub` repository and compilation of the code as per the instructions in the repository readme file.

Links

Link	Description
Github repository	Repository for download and installation
Documentation	Documentation for ANGSD

BEDtools 1

Collectively, the bedtools utilities are a swiss-army knife of tools for a wide-range of genomics analysis tasks. The most widely-used tools enable genome arithmetic: that is, set theory on the genome. For example, bedtools allows one to intersect, merge, count, complement, and shuffle genomic intervals from multiple files in widely-used genomic file formats such as BAM, BED, GFF/GTF, VCF. While each individual tool is designed to do a relatively simple task (e.g., intersect two interval files), quite sophisticated analyses can be conducted by combining multiple bedtools operations on the UNIX command line.

Links

Link	Description
Bedtools documentation	The documentation page. Contains well done tutorials and examples, both for beginners and advanced users.

Cellranger 1

The software pipeline used to align single cell sequencing data having Unique Molecular Identifier and coming as output from a 10X sequencing machine.

Links

Link	Description
What is cellranger	Description of the cellranger pipeline
Install cellranger	Installation guide
Use cellranger	User guide and tutorials

HOMER 1

HOMER (Hypergeometric Optimization of Motif EnRichment) is a suite of tools for Motif Discovery and next-gen sequencing analysis. It is a collection of command line programs for UNIX-style operating systems written in Perl and C++. HOMER was primarily written as a de novo motif discovery algorithm and is well suited for finding 8-20 bp motifs in large scale genomics data. HOMER contains many useful tools for analyzing ChIP-Seq, GRO-Seq, RNA-Seq, DNase-Seq, Hi-C and numerous other types of functional genomics sequencing data sets.

Links

Link	Description
HOMER homepage	Software homepage
Configure HOMER	Setup and configuration
Tutorial	Tutorial from the official home page

Kallisto 1

`Kallisto` is a program for quantifying abundances of transcripts from bulk and single-cell RNA-Seq data, or more generally of target sequences using high-throughput sequencing reads. It is based on the novel idea of pseudoalignment for rapidly determining the compatibility of reads with targets, without the need for alignment.

Links

Link	Description
Homepage	Software home page with tutorials and documentation
Kallistobus	Workflow to use kallisto to generate the expression data matrix
Reference	The Kallisto paper

MACS2 1

This tool identifies statistically significantly enriched genomic regions in ChIP- and DNase-seq data using the MACS2 algorithm (Model-Based Analysis of ChIP-Seq).

Links

Link	Description
Manual	MACS2 manual
Project page	Project page with installation and usage instructions
Tutorial	A tutorial for peak calling
Reference	Reference paper

Salmon 1️⃣

Highly-accurate & wicked fast transcript-level quantification from RNA-seq reads using selective alignment. Alevin, a tool integrated in Salmon, works on 3' tagged data, such as single cell data from 10X machines.

Links

Link	Description
Salmon	Documentation and tutorial page for Salmon
Alevin	Documentation and tutorial page for Alevin-Salmon
Homepage	Software homepage
Reference	Reference paper

SAMtools 1️⃣

Samtools is a suite of programs for interacting with high-throughput sequencing data. It consists of three separate repositories: - **Samtools**, Reading/writing/editing/indexing/viewing SAM/BAM/CRAM format - **BCFtools**, Reading/writing BCF2/VCF/gVCF files and calling/filtering/summarising SNP and short indel sequence variants - **HTSlib**, A C library for reading/writing high-throughput sequencing data

Links

Link	Description
Homepage	Documentation and tutorial page
Tutorial 1	Samtools tutorial
Tutorial 2	Blogpost on SAMtools

Scanpy 🐍

Scanpy is a scalable toolkit for analyzing single-cell gene expression data built jointly with anndata. It includes preprocessing, visualization, clustering, trajectory inference and differential expression testing. The Python-based implementation efficiently deals with datasets of more than one million cells.

Links

Link	Description
Homepage	Documentation page with tutorials, package documentation and access to example datasets ready to use

Seurat 🐍

Parallely to Scanpy in python, Seurat is the leading package for single cell data analysis in R.

Links

Link	Description
Homepage	Documentation page with tutorials, package documentation and access to example datasets ready to use

Seqtk 1️⃣ 🐍

Seqtk is a fast and lightweight tool for processing sequences in the FASTA or FASTQ format. It seamlessly parses both FASTA and FASTQ files which can also be optionally compressed by gzip.

Links

Link	Description
Homepage	Documentation page with command usage
Repository and examples	Online repository with some usage examples

STAR 1️⃣ 🐍

Alignment method utilizes to align sequence reads to the reference genome

Links

Link	Description
Homepage	STAR software repository
Tutorial	Online tutorial to use the STAR aligner

Computational fluid dynamics

openFOAM 1️⃣

OpenFOAM is the free, open source CFD software developed primarily by OpenCFD Ltd since 2004. It has a large user base across most areas of engineering and science, from both commercial and academic organisations. OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to acoustics, solid mechanics and electromagnetics.

Links

Link	Description
Homepage, openFOAM foundation	Software homepage
Forum	OpenFOAM users forum
Wiki	OpenFOAM (unofficial) wiki

ANSYS 1

Links

Link	Description
Tutorials for Undergraduate Mechanical Engineering Courses	Exercises intended only as an educational tool to assist those who wish to learn how to use ANSYS
Basics video tutorial	Tutorial for the ANSYS basics
Introduction to static structural	Video introduction to static structural with ANSYS
Simulation of 3d centrifugal pump	Video guide to simulate a centrifugal pump with ANSYS
Support resources for students	Support resources for students. Includes the possibility for a free download of the software for a tryout.
University of Alberta - ANSYS Tutorials	ANSYS tutorials from the UNiversity of Alberta

Computational chemistry

GROMACS 1

GROMACS is a versatile package to perform molecular dynamics, i.e. simulate the Newtonian equations of motion for systems with hundreds to millions of particles. It is primarily designed for biochemical molecules like proteins, lipids and nucleic acids that have a lot of complicated bonded interactions, but since GROMACS is extremely fast at calculating the nonbonded interactions (that usually dominate simulations) many groups are also using it for research on non-biological systems, e.g. polymers.

Links

Link	Description
GROMACS	Official homepage
KIGAKI3 tutorial	Tutorial with KIGAKI3 peptide model
MD tutorials	Six tutorials updated to one of the latest GROMACS software versions
More Tutorials	Six tutorials updated to one of the latest GROMACS software versions
Complex MD tutorial	Complex and detailed MD tutorial from BioExcel. Requires the BioExcel biobb package

NWchem

The NWChem software contains computational chemistry tools that are scalable both in their ability to efficiently treat large scientific problems, and in their use of available computing resources from high-performance parallel supercomputers to conventional workstation clusters.

NWChem can handle: - Biomolecules, nanostructures, and solid-state - From quantum to classical, and all combinations - Ground and excited-states - Gaussian basis functions or plane-waves - Scaling from one to thousands of processors - Properties and relativistic effects

Links

Link	Description
NWCHEM	Official homepage with tutorials and installation information
Argonne NL tutorial	Tutorial from the Argonne national lab

LAMMPS 1️⃣ 🐍

LAMMPS is a classical molecular dynamics code with a focus on materials modeling. It's an acronym for Large-scale Atomic/Molecular Massively Parallel Simulator.

Links

Link	Description
NWCHEM	Official homepage with tutorials and installation information
Very Basic Tutorial	Basic tutorial with one or more diffusing particles
Tutorials for beginners	Tutorials for beginners
Supramolecular structures	Tutorials for modeling supramolecular structure in LAMMPS from Stan Moore, Sandia NL, USA
Material from Sandia NL	Tutorial collection from the Sandia National Lab's workshops

quantumESPRESSO 1️⃣ 🐍

An integrated suite of Open-Source computer codes for electronic-structure calculations and materials modeling at the nanoscale. It is based on density-functional theory, plane waves, and pseudopotentials.

Links

Link	Description
QuantumEspresso	Homepage with installation, documentation and learning resources
QE and GPUs	How to use QuantumEspresso on GPU based HPC systems
Updated tutorials	Some updated tutorials from University of Udine, IT
Step by step tutorial	A detailed tutorial (based on a specific supercomputing facility)
Small tutorial	A small QE tutorial

SIESTA 1️⃣ 🐍

SIESTA is both a method and its computer program implementation, to perform efficient electronic structure calculations and ab initio molecular dynamics simulations of molecules and solids. SIESTA's efficiency stems from the use of a basis set of strictly-localized atomic orbitals. A very important feature of the code is that its accuracy and cost can be tuned in a wide range, from quick exploratory calculations to highly accurate simulations matching the quality of other approaches, such as plane-wave methods.

Links

Link	Description
large tutorial collection	A large collection of tutorials for Siesta
Tel Aviv Siesta tutorial	Siesta Workshop material
SIMUNE material	A range of tutorials to guide the learning process in using Siesta. Developed by Simune Atomistics
Tutorial Repository	A big repository with exercises for Siesta

Development

Dash

Dash is a productive Python framework for building web analytic applications. Dash is ideal for building data visualization apps with highly custom user interfaces in pure Python. It's particularly suited for anyone who works with data in Python.

Through a couple of simple patterns, Dash abstracts away all of the technologies and protocols that are required to build an interactive web-based application. Dash is simple enough that you can bind a user interface around your Python code in an afternoon.

Dash apps are rendered in the web browser. You can deploy your apps to servers and then share them through URLs. Since Dash apps are viewed in the web browser, Dash is inherently cross-platform and mobile ready.

Links

Link	Description
Dash homepage	The Dash homepage. It contains tutorial for a quick
Video introduction	A video introduction to Dash
Step by step tutorial	A step by step tutorial to produce a Dash web application

Jupyter Lab 1

JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning.

Link	Description
Homepage	The Homepage of project jupyter

Matlab 1

MATLAB® combines a desktop environment tuned for iterative analysis and design processes with a programming language that expresses matrix and array mathematics directly. It includes the Live Editor for creating scripts that combine code, output, and formatted text in an executable notebook.

Link	Description
Homepage	The Homepage of Mathworks Matlab

Rstudio 1 2

An integrated development environment for R and Python, with a console, syntax-highlighting editor that supports direct code execution, and tools for plotting, history, debugging and workspace management.

Link	Description
Homepage	The Homepage of Rstudio

Visual Studio code 1

Visual Studio Code is a lightweight but powerful source code editor. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity). Manages your Git repository and has a lot of plugins freely installable from the dedicated store.

Link	Description
Homepage	The Homepage of Visual Studio Code

Spark 1

Apache Spark is a lightning-fast unified analytics engine for big data and machine learning. It was originally developed at UC Berkeley in 2009.

Natural sciences and engineering

COMSOL 1

COMSOL Multiphysics is a cross-platform finite element analysis, solver and multiphysics simulation software. It allows conventional physics-based user interfaces and coupled systems of partial differential equations (PDEs). COMSOL provides an IDE and unified workflow for electrical, mechanical, fluid, acoustics, and chemical applications.

Link	Description
COMSOL models	A gallery of applications from the COMSOL homepage

FEniCS 1

FEniCS is a popular open-source (LGPLv3) computing platform for solving partial differential equations (PDEs). FEniCS enables users to quickly translate scientific models into efficient finite element code. With the high-level Python and C++ interfaces to FEniCS, it is easy to get started, but FEniCS offers also powerful capabilities for more experienced programmers. FEniCS runs on a multitude of platforms ranging from laptops to high-performance clusters.

Link	Description
Homepage	Homepage of the FEniCS project
Tutorials and book	Free official tutorials and book

FreeCAD

FreeCAD is a parametric 3D modeler made primarily to design real-life objects of any size.

Link	Description
Beginners tutorial	Beginners video tutorial
Free documentation	Freecad wiki

Probabilistic programming

pyMC3

PyMC3 allows you to write down models using an intuitive syntax to describe a data generating process.

Cutting edge algorithms and model building blocks Fit your model using gradient-based MCMC algorithms like NUTS, using ADVI for fast approximate inference — including minibatch-ADVI for scaling to large datasets — or using Gaussian processes to build Bayesian nonparametric models.

Link	Description
Documentation	Documentation page with tutorials, videos, examples and books
While my MCMC gently samples	A blog about pyMC3 and Bayesian inference
Bayesian modeling cookbook blog	A blog on bayesian modeling
Probabilistic Programming blog	A blog on probabilistic programming from a pyMC3 contributor

Social Sciences, Humanities

NetLogo 1

NetLogo is a multi-agent programmable modeling environment. It is used by many hundreds of thousands of students, teachers, and researchers worldwide. It also powers HubNet participatory simulations. It is authored by Uri Wilensky and developed at the CCL.

Link	Description
Homepage	Netlogo homepage with tutorials, extensions, resources.

oTree 1

An open-source platform for behavioral research

oTree lets you create: - Controlled behavioral experiments in economics, market research, psychology, and related fields - Multiplayer strategy games, like the prisoner's dilemma, public goods game, and auctions. - Surveys and quizzes, especially those that require customized or dynamic functionality not available with conventional survey software.

Link	Description
Homepage	oTree homepage with tutorials, extensions, resources.

HPC programming

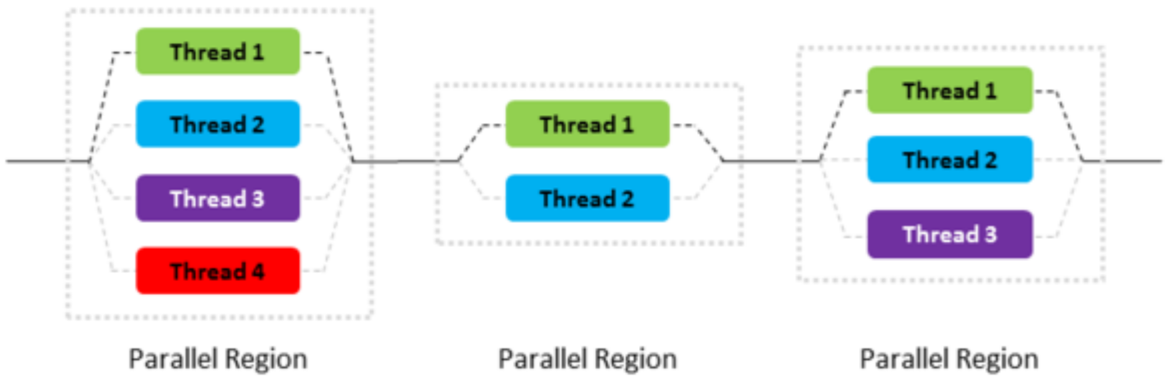
As in any other computer, an HPC can be used with sequential programming. This is the practice of writing computer programs executing one instruction after the other, but not instructions simultaneously in parallel, i.e. parallel programming.

Parallel programming

There are different ways of writing parallelized code, while in general there is only one way to write sequential code, generally as a logic sequence of steps.

openMP (multithreading)

A popular way of parallel programming is through writing sequential code and pointing at specific pieces of code that can be parallelized into threads (fork-join mechanism, see figure below from ADMIN magazine). A thread is an independent execution of code with its own allocated memory.



If threads vary in execution time, when they have to be joined together to collect data, some threads might have to wait for others, leading to loss of execution time. It is up to the programmer to best balance the distribution of threads to optimize execution times when possible.

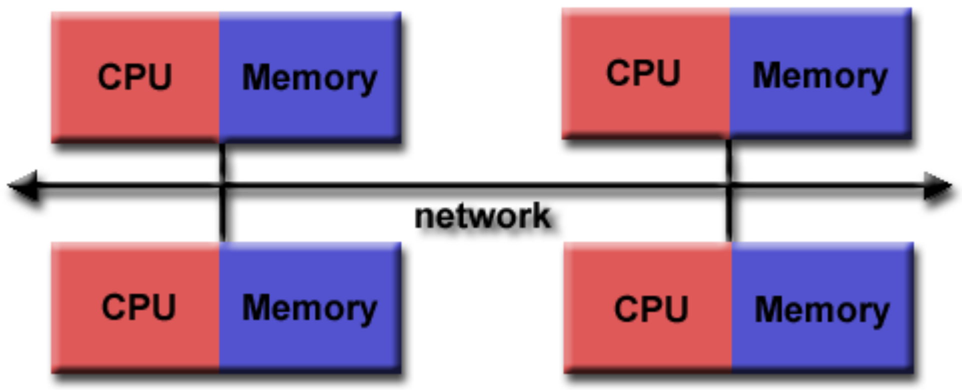
Modern CPUs support openMP in a natural way, since they are usually multicore CPUs and each core can execute threads independently. OpenMP is available as an extension to the programming languages C and Fortran and is mostly used to parallelize for loops that constitute a time bottleneck for the software execution.

Link	Description
Video course	a video course (here the link to the first lesson, you will be able to find all the other lessons associated to that) held by ARCHER UK.

Link	Description
OpenMP Starter	A starting guide for OpenMP
Wikitolearn course	An OpenMP course from Wikitolearn
MIT course	A course from MIT including also MPI usage (next section for more info about MP)

MPI (message passing interface)

MPI is used to distribute data to different processes, that otherwise could not access to such data (picture below, from LLNL).

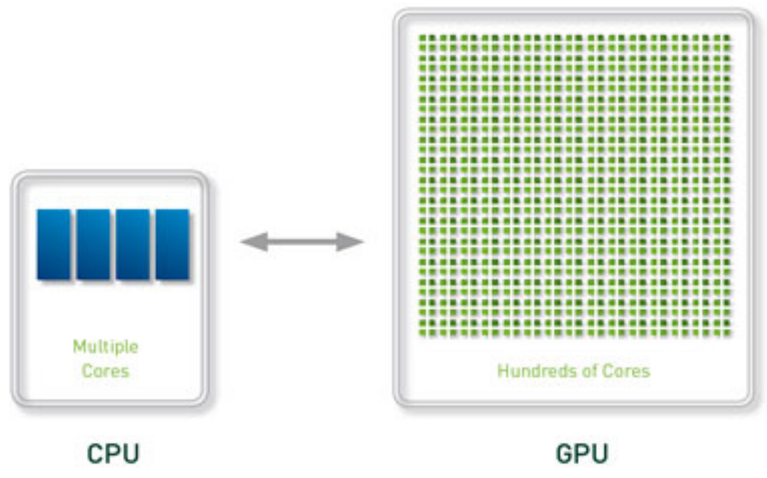


MPI is considered a very hard language to learn, but this reputation is mostly due to the fact that the message passing is programmed explicitely.

Link	Description
Video course	a video course (here the link to the first lesson, you will be able to find all the other lessons associated to that) held by ARCHER UK.
MPI Starter	A starting guide for OpenMP
PRACE course	A prace course on the MOCC platform futurelearn

GPU programming

GPUs (graphical processing units) are computing accelerators that are used to boosts heavy linear algebra applications, such as deep learning. A GPU usually features a large number of special processing units that can make the computer code extremely parallelized (figure below from astrocomputing).



AMD and Nvidia are the two main producers of GPUs, where the latter has dominated the market for a long time. The danish HPCs Type 1 and 2 feature various models of Nvidia graphic cards, while Type 5 (LUMI) has the latest AMD Instinct.

The distinction between AMD and Nvidia is mainly due to the fact that they are programmed with two different dialects, and softwares with dedicated multithreading on GPUs need to be coded specifically for the two brands of GPUs.

Nvidia CUDA

CUDA is a C++ dialect that has also various library for the most popular languages and packages (e.g. python, pytorch, MATLAB, ...).

Link	Description
Nvidia developer training	Nvidia developer trainings for CUDA programming
Book archive	An archive of books for CUDA programming
Advanced books	Some advanced books for coding with CUDA
pyCUDA	Code in CUDA with python

AMD HIP

HIP is a dialect for AMD GPUs of recent introduction. It has the advantage of being able to be compiled for both AMD and Nvidia hardware. CUDA code can be converted to HIP code almost automatically with some extra adjustments by the programmer.

The LUMI HPC consortia has already organized a course for HIP coding and CUDA-to-HIP conversion. Check out their page for new courses.

Link	Description
Video introduction 1	Video introduction to HIP
Video introduction 2	Video introduction to HIP
AMD programming guide	Programming guide to HIP from the producer AMD

Choose your HPC

Here you can quickly fill in an assessment form to understand which HPC is best suited to your application. Answers are not mandatory in case you are uncertain of some specific choice.