Objectives of the workshop HPC

This is an entry level workshop, aimed at reserachers with little to reasonable programming background, but no significant experience with the usage of a high performance computer.

You will learn how to run your programs in parallel on a computer cluster. A computer cluster is a system with a lot of memory and a lot of processors which you share with hundreds of other users. A scheduler keeps track of the resources available and decides when to run your programs. You will learn how to write scripts telling the scheduler which programs to run, which resources are needed and for how long.

The operating system of a computer cluster will nearly always be Linux. The basic Linux concepts and commands will be introduced to you. But before you can learn Linux, you have to make a connection to the cluster. We will tell you how to login from your desk- or laptop on a cluster with a terminal emulation program. Editing and transferring files between your computer and the remote cluster is also on the menu of this workshop.

The programming examples are in R. So, basic understanding of R and RStudio is required. Some experience with text editing and command line interfaces is a plus.

Content of the workshop HPC

Brief Introduction into High Performance Computing

Without delving to deep in the theory of HPC the most important concepts and characteristics of cluster computers and computing are explained:

Components of a computer cluster:

- Nodes & cores
- Memory & Storage
- Batch Scheduler
- Inlog node
- Scratch space

Programming concepts: * Pleasingly Parallel * Parallel Programming * Advanced Parallel Programming Techniques

This part we will end discussing for what types of computing jobs cluster computing is suited.

<hand-outs>

Example program in R: Recognizing Hand-written Digits with Machine Learning

Throughout the workshop we will use a small program to solve the challenge of recognizing handwritten digits by a computer. This is a rather famous example in Machine Learning. We will run an R script on your local computer and notice why and when we are in need of an HPC. *<course material>*

Getting started with LISA

Before you can start working on LISA, you have to register your account, and install some software packages on your computer. This software is necessary to login on LISA (terminal session) and to transfer files between your workstation and LISA. You also learn basic Linux commands. There are differnt software packages for Mac users and Window users. On LISA you all will work with the same operating system: LINUX. Linux on Lisa doesn't support a graphical user interface (GUI). We will teach you how to edit text files on a window-less system.

<course material>

Running your first program on Lisa

Submitting an R script

Watching the output

Pleasingly parallel: finding the best hyper parameter settings

This is the moment you all have been waiting for. Now you are going to run a few dozen programs in parallel. You will train several machine learning model in parallel and, afterwards, pick the best performing one. This is the same as the grid search as we did on your local workstation, but now in **parallel** instead of **sequential**.

In the previous chapter submission and batch scripts were introduced. You have made them with an text editor. You can imagine that yopu can't maken them that way if there are thousands of different tasks to run. You will learn how to make an R script to generate those batch and the overall submission script

<course material for this lesson>

Enhancements to your batch jobs

Your scripts need some refinements to work in a "real life" setting.

Logfiles

Your program can contain errors. So you need log files where your code can writes its error messages.

Email notifications

In this workshops your jobs are running almost immediately after submitting. But that's because we heve reserved quit a few nodes in advance. In "real life" you could be waiting for hours, sometimes days, before it is your turn to run. It would be nice if the scheduler sends you an email upon starting and ending your batch jobs.

scratch space

When you have a lot of data to process, for reasons of performance you should make use of the fast scratch storage every node has. You will change your scripts so that input and output data are transferred between you home filesystem and the scratch of every node you use.

<course material>

Second session

The second session will consist of: **left-overs from first sessions Q&A about the first session Parallel Programming in R Wrap up and Evaluation**