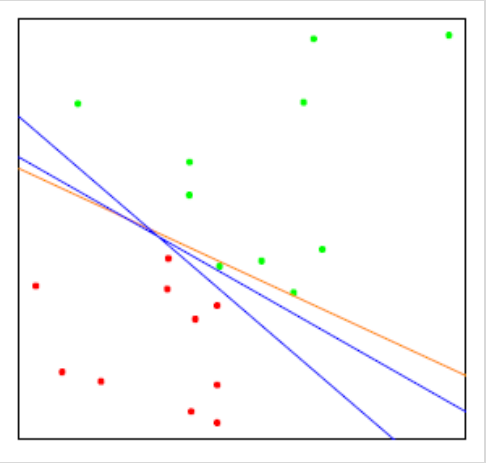


Parallel & Scalable Machine Learning

Introduction to Machine Learning Algorithms

Prof. Dr. – Ing. Morris Riedel

Adjunct Associated Professor

School of Engineering and Natural Sciences, University of Iceland

Research Group Leader, Juelich Supercomputing Centre, Germany

LECTURE 1

Parallel & Scalable Machine Learning driven by HPC

February 25th, 2019

Juelich Supercomputing Centre, Juelich, Germany

PRACE PARTNERSHIP FOR ADVANCED COMPUTING IN EUROPE TRAINING PORTAL

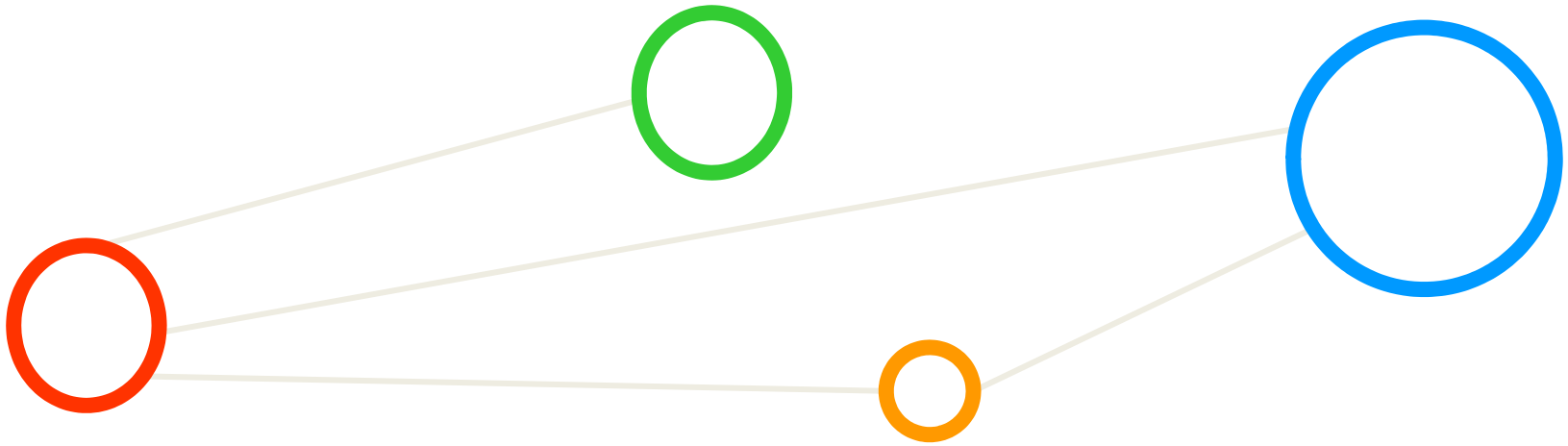
UNIVERSITY OF ICELAND
SCHOOL OF ENGINEERING AND NATURAL SCIENCES
FACULTY OF INDUSTRIAL ENGINEERING, MECHANICAL ENGINEERING AND COMPUTER SCIENCE

JÜLICH Forschungszentrum

HELMHOLTZ RESEARCH FOR GRAND CHALLENGES

DEEP Projects

Outline



Outline of the Course

1. Parallel & Scalable Machine Learning driven by HPC
2. Introduction to Machine Learning Fundamentals
3. Introduction to Machine Learning Fundamentals
4. Feed Forward Neural Networks
5. Feed Forward Neural Networks
6. Validation and Regularization
7. Validation and Regularization
8. Data Preparation and Performance Evaluation
9. Data Preparation and Performance Evaluation
10. Theory of Generalization
11. Unsupervised Clustering and Applications
12. Unsupervised Clustering and Applications
13. Deep Learning Introduction

Theoretical Lectures

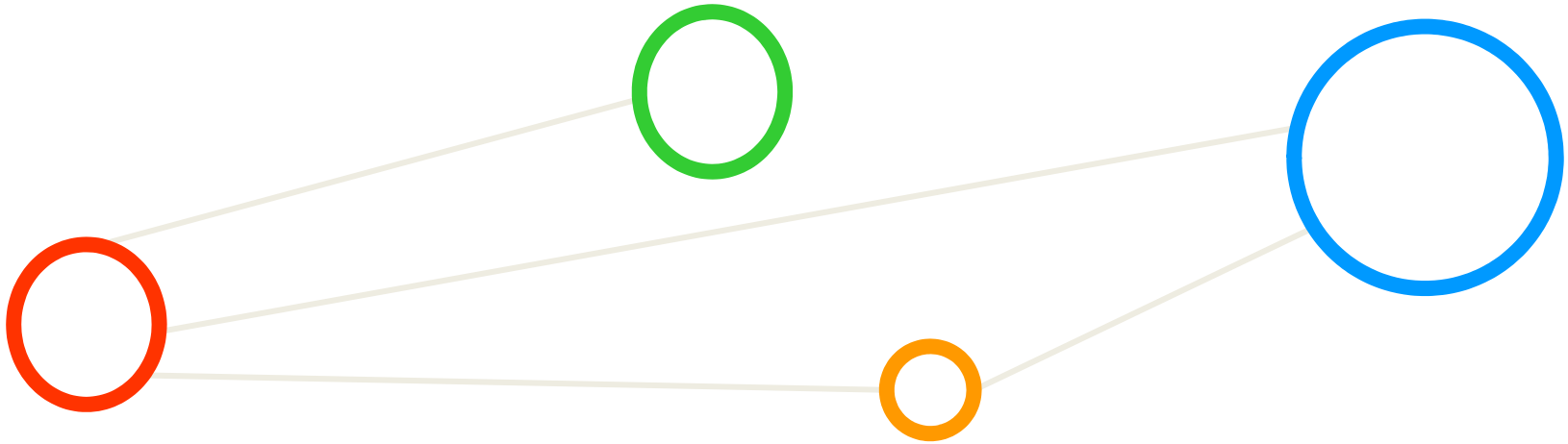
Practical Lectures

Outline

- Machine Learning driven by HPC
  - Welcome & Course Goals, Content & Timeliness
  - Big Data & Machine/Deep Learning
  - Understanding High Performance Computing
  - Multi-Core vs. Many-Core Technologies
  - Pan-European HPC Infrastructure PRACE

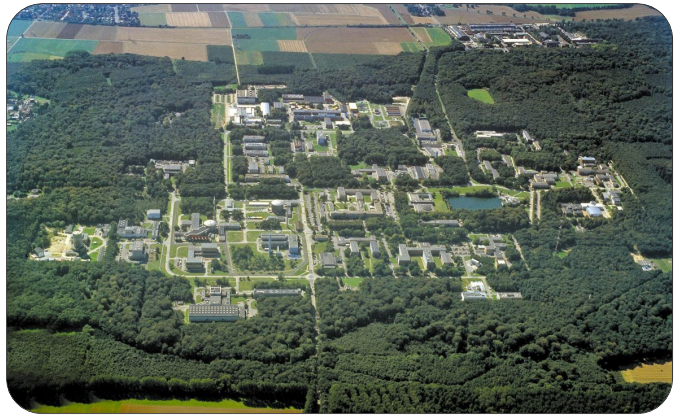
- Parallel & Scalable Technology
  - DEEP-EST & Modular Supercomputing
  - JSC Infrastructure for Tutorial
  - Python, NumPy & Vectorization Approach
  - Account Checks & Jupyter Notebooks
  - SSH Clients & HPC Modules

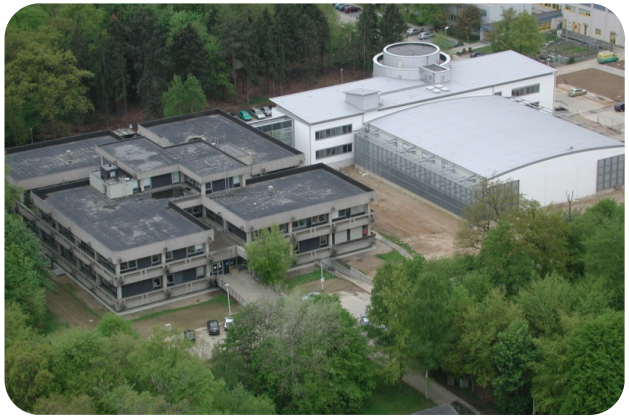
Machine Learning driven by HPC



Welcome @ Juelich Supercomputing Centre (JSC)

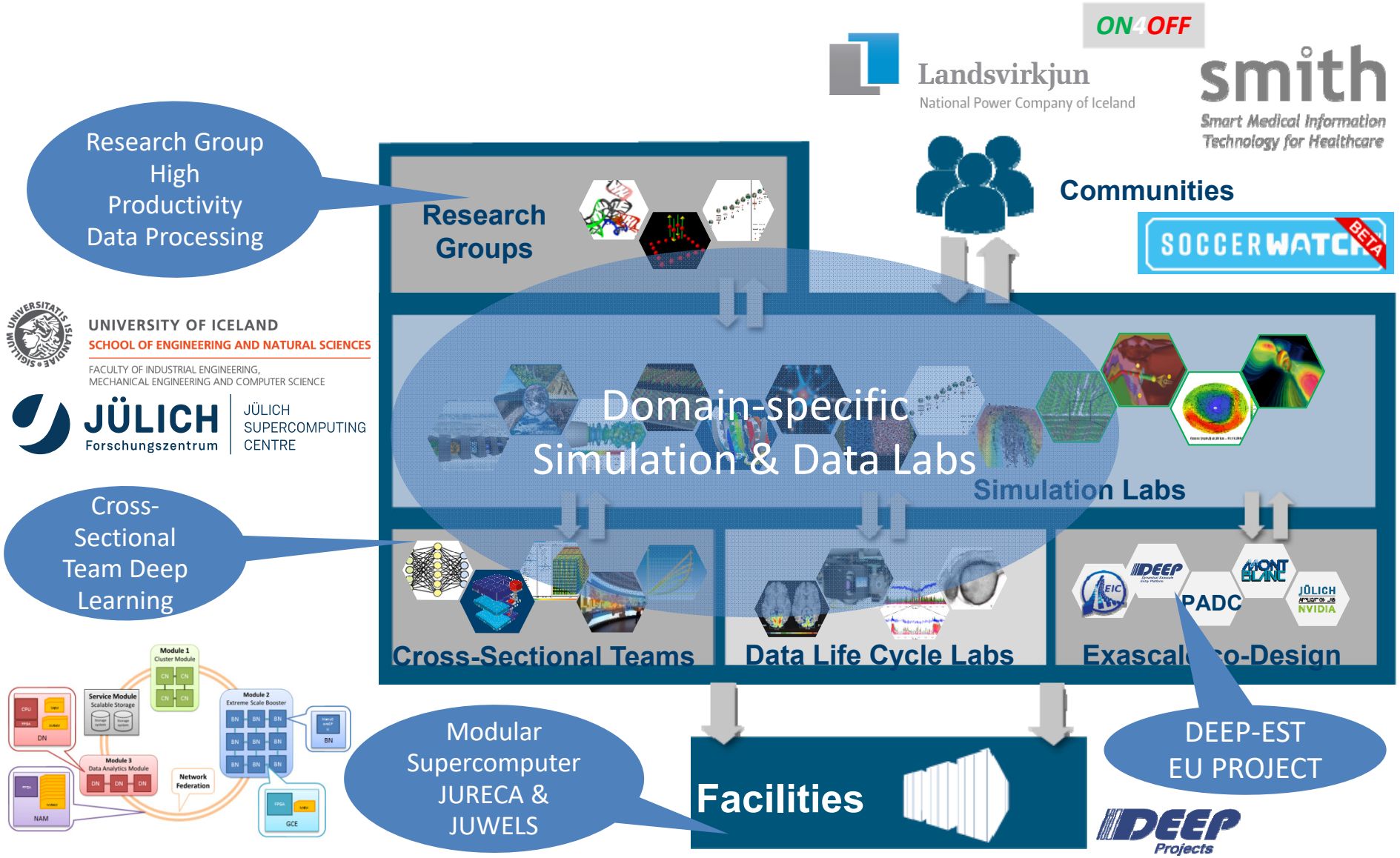
- Part of Research Centre 'Forschungszentrum Jülich' in Germany
  - Founded in 1956 as nuclear research facility, staff ~4500
  - 90% Federal Republic of Germany; 10% State of North Rhine-Westphalia
  - Research Areas: Health, Energy & Environment, Information, Key Competency (e.g. HPC and computational science, big data processing)
- JSC is part of the Insitute for Advanced Simulation (IAS)
  - Operate supercomputers (→ TOP500), cloud systems, and large storages
  - Enable scientists to solve complex problems via simulations and analytics





[12] Juelich Supercomputing Centre Web page

High Productivity Data Processing Research Group @ JSC



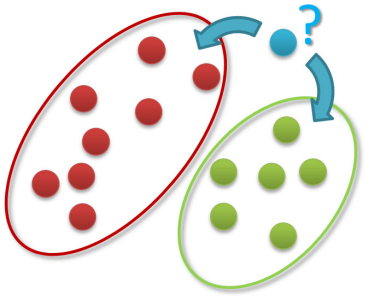
Course Goals

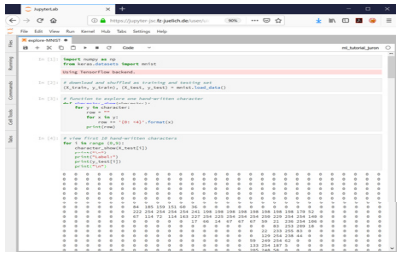
Fundamentals

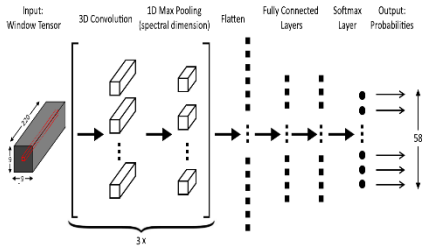
Practical Skills

Advancements

Technology







Community Building

HELMHOLTZ
RESEARCH FOR GRAND CHALLENGES

JÜLICH
Forschungszentrum

UNIVERSITY OF ICELAND
SCHOOL OF ENGINEERING AND NATURAL SCIENCES
FACULTY OF INDUSTRIAL ENGINEERING,
MECHANICAL ENGINEERING AND COMPUTER SCIENCE

NCSA

→ **Join our JOint International Machine Learning Laboratory (JOIML)**

Machine Learning in Context & Course Focus

Artificial Intelligence (AI)

A wide area of techniques and tools that enable computers to mimic human behaviour

Machine Learning (ML)

Learning from data without explicitly being programmed with common programming languages

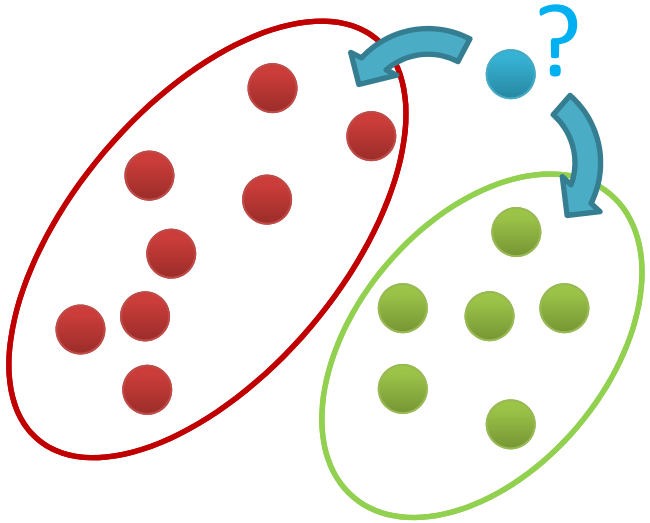
Deep Learning (DL)

Systems with the ability to learn underlying features in data using large neural networks

Machine Learning Models – Overview

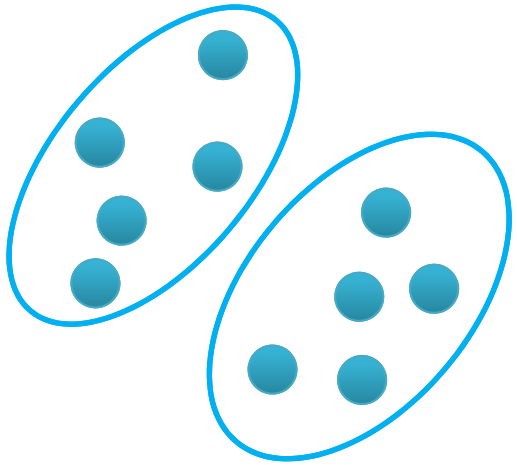
- **Machine learning methods can be roughly categorized in classification, clustering, or regression augmented with various techniques for data exploration, selection, or reduction**

Classification



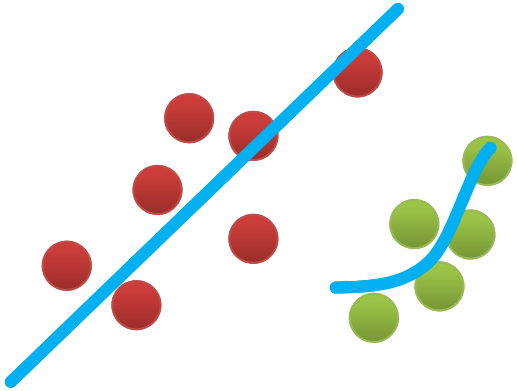
- Groups of data exist
- New data classified to existing groups

Clustering



- No groups of data exist
- Create groups from data close to each other

Regression



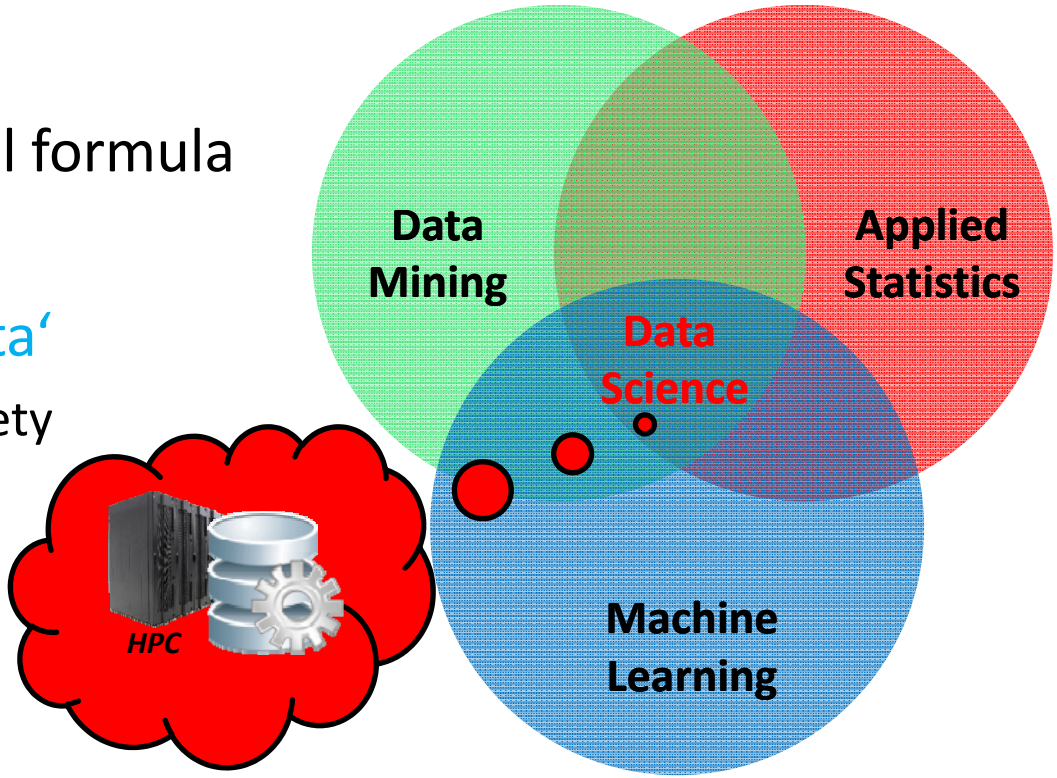
- Identify a line with a certain slope describing the data

➢ **This course focus on supervised classification techniques and unsupervised clustering methods**

Machine Learning Prerequisites & Challenges

1. Some pattern exists
2. No exact mathematical formula
3. Data exists

- Idea ‘Learning from Data‘
  - Shared with a wide variety of other disciplines
  - E.g. signal processing, data mining, etc.
- Challenges
  - Data is often complex
  - Learning from data requires processing time → Clouds



- **Machine learning is a very broad subject and goes from very abstract theory to extreme practice (‘rules of thumb’)**
- **Training machine learning models needs processing time**

➢ **This course addressed machine learning & data science with a focus on parallel & scalable HPC**

Course Timeliness

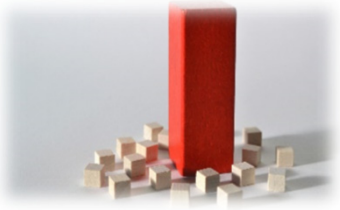
- Machine learning dates back decades
  - We observe a momentum around AI, DL, and ML – why today?

1952 Stochastic Gradient Descent
- Optimization

1958 Perceptron Linear Learning Model
- Learnable Weights

1958 Backpropagation of Error in Learning through Layers
- Multi-Layer Perceptron

1995 Deep Convolutional Neural Networks
- Image Recognition



Big Data
- Larger datasets
- Easier access to collections
- Better transport to storage

Hardware
- More memory
- Graphical Processing Units (GPUs)
- Massively parallel systems

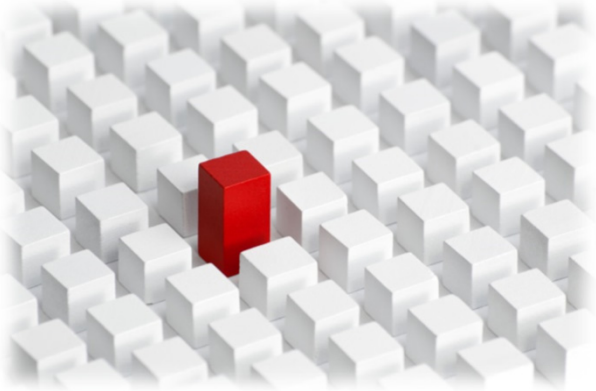
Software
- Improved scalable techniques
- New models
- Open source frameworks & toolsets

[3] NVIDIA

[2] TensorFlow

[1] Keras

Big Data Motivation: Intertwine HPC & Machine Learning



- Rapid advances in data collection and storage technologies in the last decade
  - Extracting useful information is a challenge considering ever increasing massive datasets
  - Traditional data analysis techniques cannot be used in growing cases (e.g. memory, speed, etc.)

(yellow blocks very important to learn for exams)

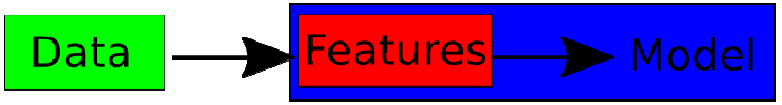
- **Machine learning / Data Mining is a technology that blends traditional data analysis methods with sophisticated algorithms for processing large volumes of data**
- **Machine Learning / Data Mining is the process of automatically discovering useful information in large data repositories ideally following a systematic process**

modified from [4] Introduction to Data Mining

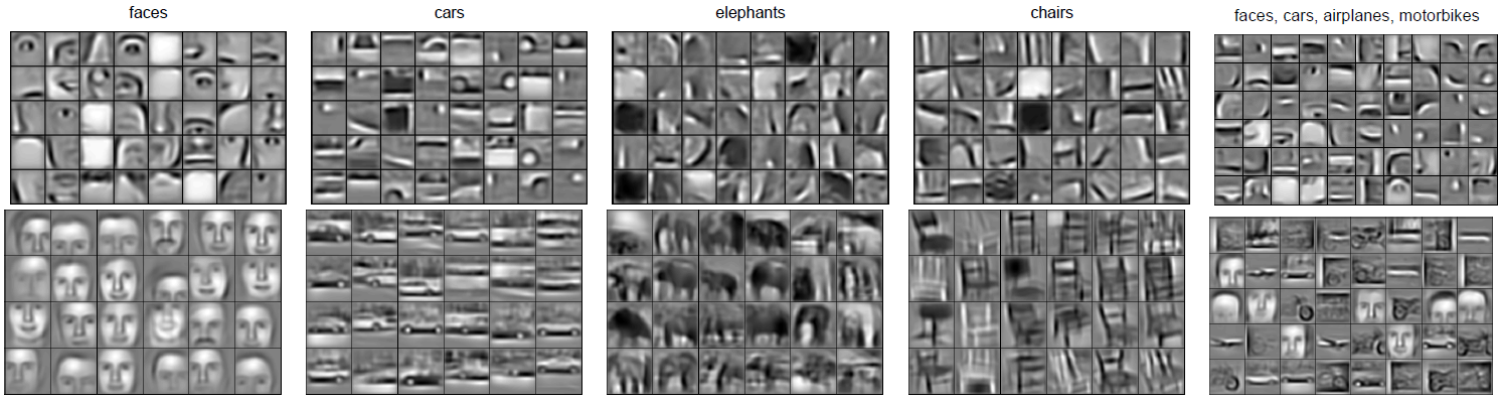
- Machine Learning & Statistical Data Mining
  - Traditional statistical approaches are still very useful to consider
  - Deep Learning tools become effective and are available in Clouds today

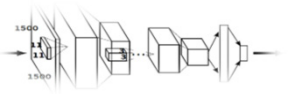
‘Big Data‘ drives Deep Learning Models

- Approach: Learn Features
  - Classical Machine Learning
  - (Powerful computing evolved)
  - Deep (Feature) Learning



  - Very succesful for image recognition and other emerging areas
  - Assumption: data was generated by the interactions of many different factors on different levels (i.e. form a hierarchical representation)





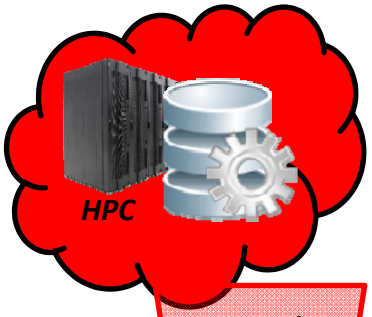
[5] H. Lee et al.

➢ **We offer a short introduction to deep learning at the enf of this machine learning course**

Relationship Machine/Deep Learning & Big Data & HPC

(large amounts of GPUs
= Graphical Processing Units)

#GPUs	images/s	speedup	Performance per GPU [images/s]
1	55	1.0	55
4	178	3.2	44.5
8	357	6.5	44.63
16	689	12.5	43.06
32	1230	22.4	38.44
64	2276	41.4	35.56
128	5562	101.1	43.45



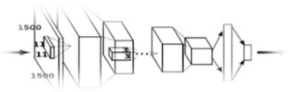
Model Performance / Accuracy

'small training sets'

manual feature engineering changes ordering

Processing Time

Large Deep Learning Network

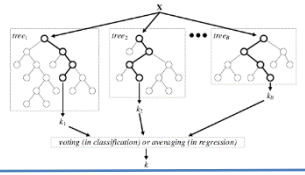


Medium Deep Learning Network

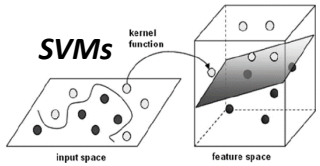
Small Neural Network

Traditional Learning Models

Random Forests



SVMs



MatLab

Statistical Computing with R

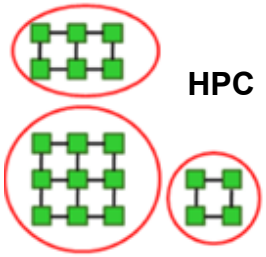
scikit-learn *Weka* *Octave*

Dataset Volume → 'Big Data'

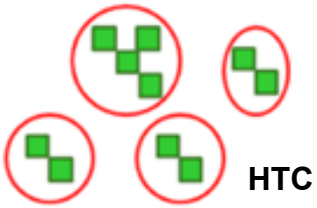
➢ JSC & University of Iceland offers a focussed deep learning course later this year @ JSC

Understanding High Performance Computing

- **High Performance Computing (HPC) is based on computing resources that enable the efficient use of parallel computing techniques through specific support with dedicated hardware such as high performance cpu/core interconnections.**



- **High Throughput Computing (HTC) is based on commonly available computing resources such as commodity PCs and small clusters that enable the execution of 'farming jobs' without providing a high performance interconnection between the cpu/cores.**



➢ **This course focus on parallel & scalable methods using High Performance Computing (HPC)**

Parallel Computing

- All modern supercomputers depend heavily on parallelism

  - **We speak of parallel computing whenever a number of 'compute elements' (e.g. cores) solve a problem in a cooperative way**

[7] Introduction to High Performance Computing for Scientists and Engineers

- Often known as 'parallel processing' of some problem space
  - Tackle problems in parallel to enable the 'best performance' possible

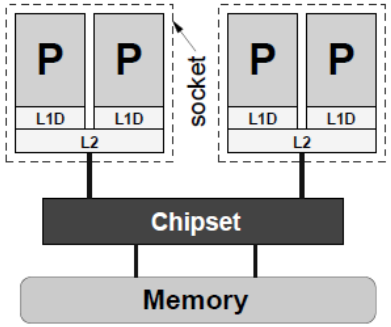
- 'The measure of speed' in High Performance Computing matters
  - Common measure for parallel computers established by TOP500 list
  - Based on benchmark for ranking the best 500 computers worldwide

network interconnection important

[8] TOP 500 supercomputing sites

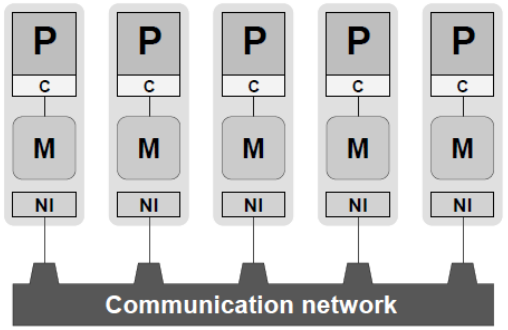
Parallel Computing: Shared Memory vs. Distributed Memory

- Shared Memory



Massively Parallel Computers

- Distributed Memory

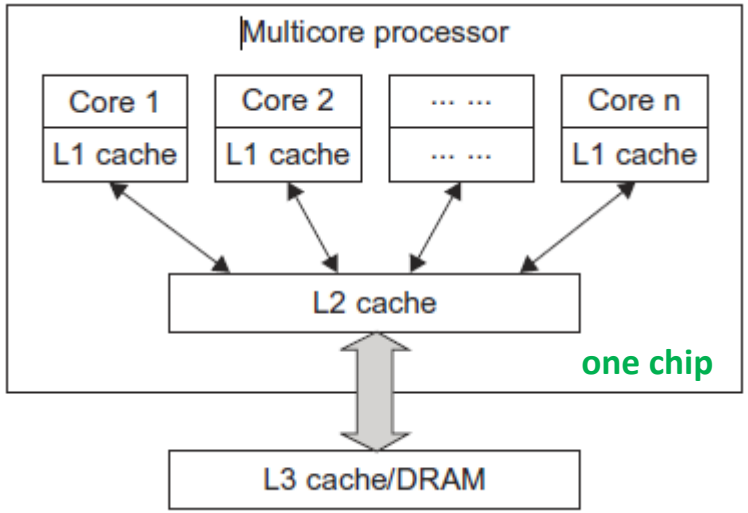


[7] Introduction to High Performance Computing for Scientists and Engineers

- **A shared-memory parallel computer is a system in which a number of CPUs work on a common, shared physical address space**
- **A distributed-memory parallel computer establishes a 'system view' where no process can access another process' memory directly**

Multi-core CPU Processors

- Significant advances in CPU (or microprocessor chips)
  - Multi-core architecture with dual, quad, six, or n processing cores
  - Processing cores are all on one chip
- Multi-core CPU chip architecture
  - Hierarchy of caches (on/off chip)
  - L1 cache is private to each core; on-chip
  - L2 cache is shared; on-chip
  - L3 cache or Dynamic random access memory (DRAM); off-chip



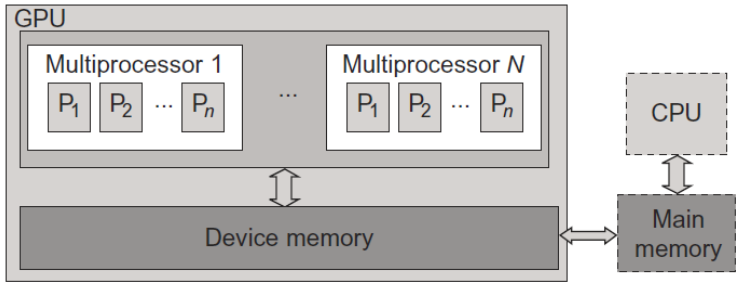
[21] Distributed & Cloud Computing Book

- **Clock-rate for single processors increased from 10 MHz (Intel 286) to 4 GHz (Pentium 4) in 30 years**
- **Clock rate increase with higher 5 GHz unfortunately reached a limit due to power limitations / heat**
- **Multi-core CPU chips have quad, six, or n processing cores on one chip and use cache hierarchies**

Many-core GPUs

- **Graphics Processing Unit (GPU) is great for data parallelism and task parallelism**
- **Compared to multi-core CPUs, GPUs consist of a many-core architecture with hundreds to even thousands of very simple cores executing threads rather slowly**

- Use of very many simple cores
  - High throughput computing-oriented architecture
  - Use massive parallelism by executing a lot of concurrent threads slowly
  - Handle an ever increasing amount of multiple instruction threads
  - CPUs instead typically execute a single long thread as fast as possible
- Many-core GPUs are used in large clusters and within massively parallel supercomputers today
  - Named General-Purpose Computing on GPUs (GPGPU)

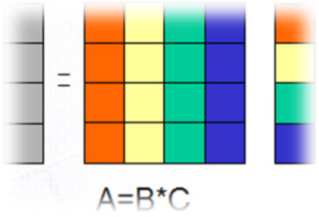


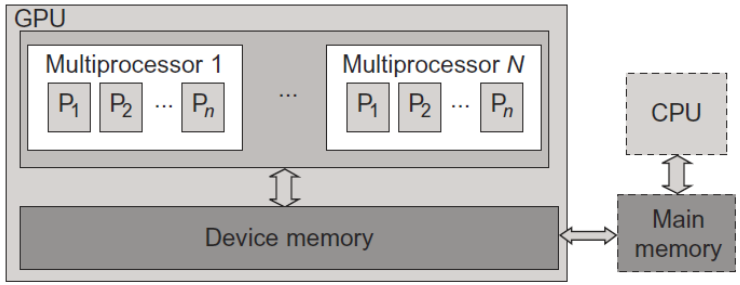
[21] Distributed & Cloud Computing Book

GPU Acceleration

- **CPU acceleration means that GPUs accelerate computing due to a massive parallelism with thousands of threads compared to only a few threads used by conventional CPUs**
- **GPUs are designed to compute large numbers of floating point operations in parallel**

- GPU accelerator architecture example (e.g. NVIDIA card)
  - GPUs can have 128 cores on one single GPU chip
  - Each core can work with eight threads of instructions
  - GPU is able to concurrently execute 128 * 8 = 1024 threads
  - Interaction and thus major (bandwidth) bottleneck between CPU and GPU is via memory interactions
  - E.g. applications that use matrix – vector multiplication

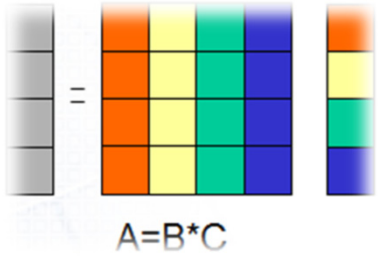




[21] Distributed & Cloud Computing Book

Parallel Matrix-Vector Multiplication Example on GPUs

- PO – P4 are processes on four GPU cores



(nice parallelization possible via independent computing)

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} b_{0,0}c_0 + & b_{0,1}c_1 + & b_{0,2}c_2 + & b_{0,3}c_3 \\ b_{1,0}c_0 + & b_{1,1}c_1 + & b_{1,2}c_2 + & b_{1,3}c_3 \\ b_{2,0}c_0 + & b_{2,1}c_1 + & b_{2,2}c_2 + & b_{2,3}c_3 \\ b_{3,0}c_0 + & b_{3,1}c_1 + & b_{3,2}c_2 + & b_{3,3}c_3 \end{bmatrix}$$

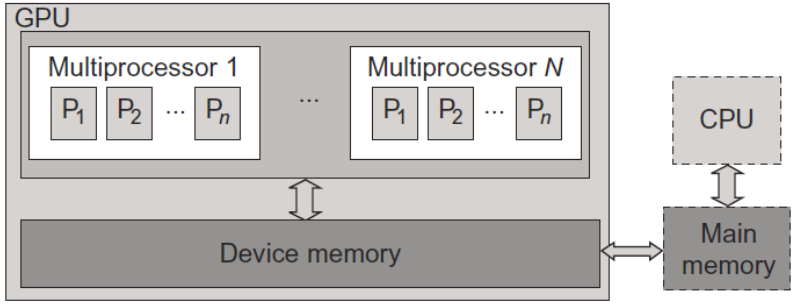
P0 P1 P2 P3

- **Step one: each GPU core has a column of matrix B (named as Bpart)**
- **Step one: each GPU core has an element of column vector C (named Cpart)**

- **Step two: Each GPU core performs an independent vector-scalar multiplication (based on their Bpart and Cpart contents)**

- **Step three: Each GPU core has a part of the result vector A (named Apart) and is written in device memory**

(GPUs are designed to compute large numbers of floating point operations in parallel)



[21] Distributed & Cloud Computing Book

Summary: Centralized Computing & Parallel Computing

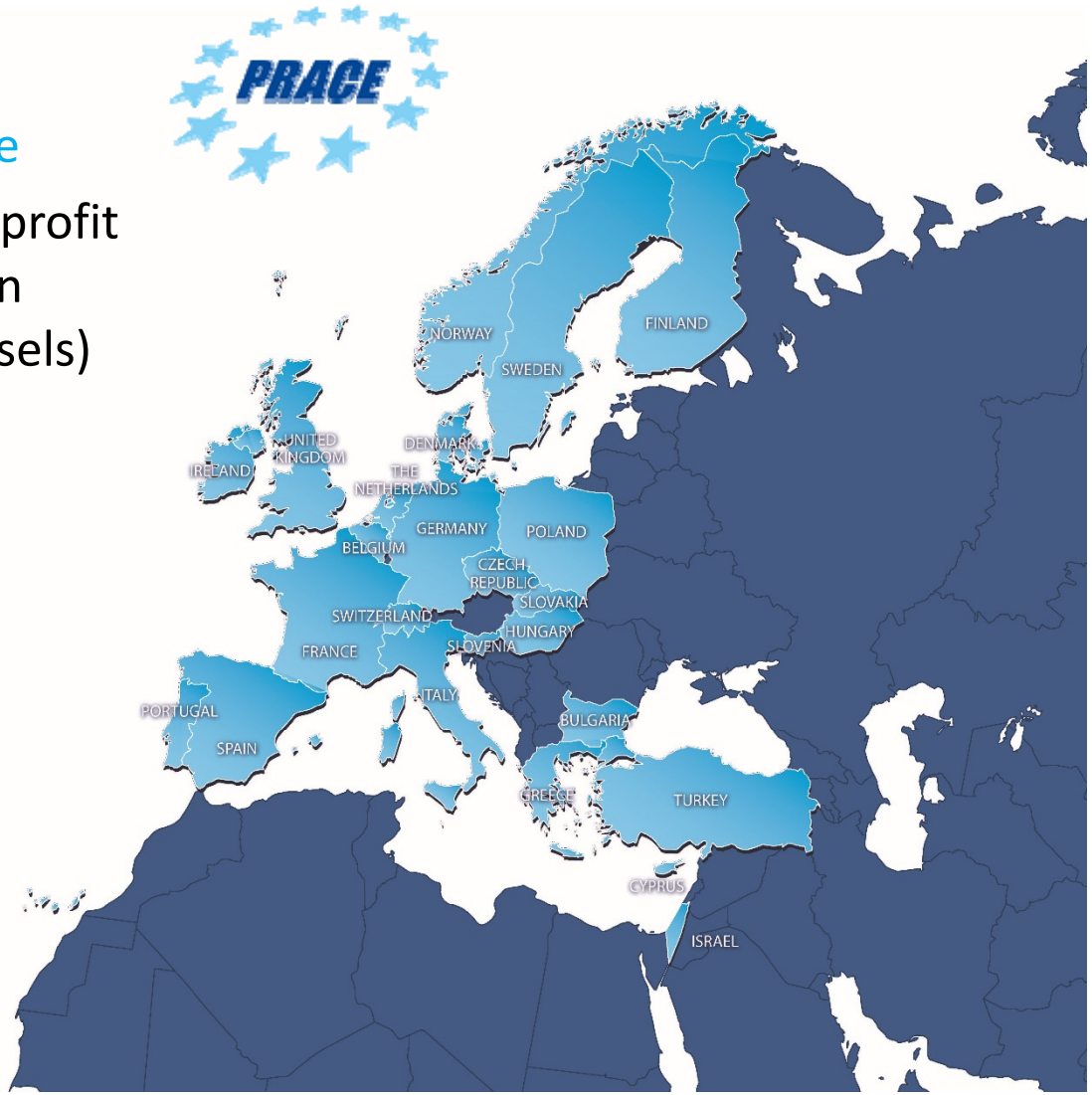
- Centralized Computing
  - All computer resources are centralized in one physical system
  - All resources (e.g. processors, memory, storage, etc.) fully shared and tightly coupled within one integrated operating system
  - Many data centers & supercomputers are centralized systems

- Parallel Computing
  - All processors can be tightly coupled using shared memory
  - All processors can be loosely coupled using distributed memory
  - Interprocessor communication via shared memory or message passing
  - Computer systems capable of parallel computing is a parallel computer
  - Programs running in a parallel computer are called parallel programs
  - Process of writing parallel programs is referred to as parallel programming

[Video] JUQUEEN – Supercomputer Upgrade Example

Partnership for Advanced Computing in Europe (PRACE)

- Basic Facts
  - HPC-driven infrastructure
  - An international not-for-profit association under Belgien law (with its seat in Brussels)
  - Has 25 members and 2 observers
  - Governed by the PRACE Council in which each member has a seat
  - Daily management of the association is delegated to the Board of Directors



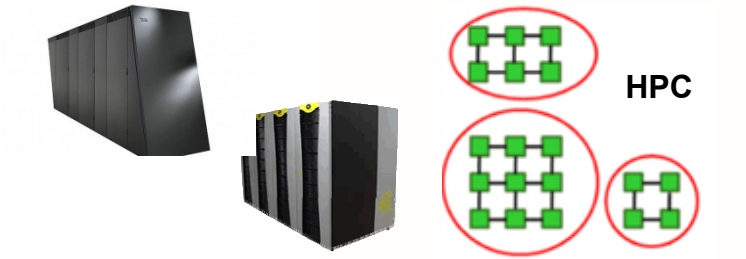
[9] PRACE

PRACE as Persistent pan-European HPC Infrastructure

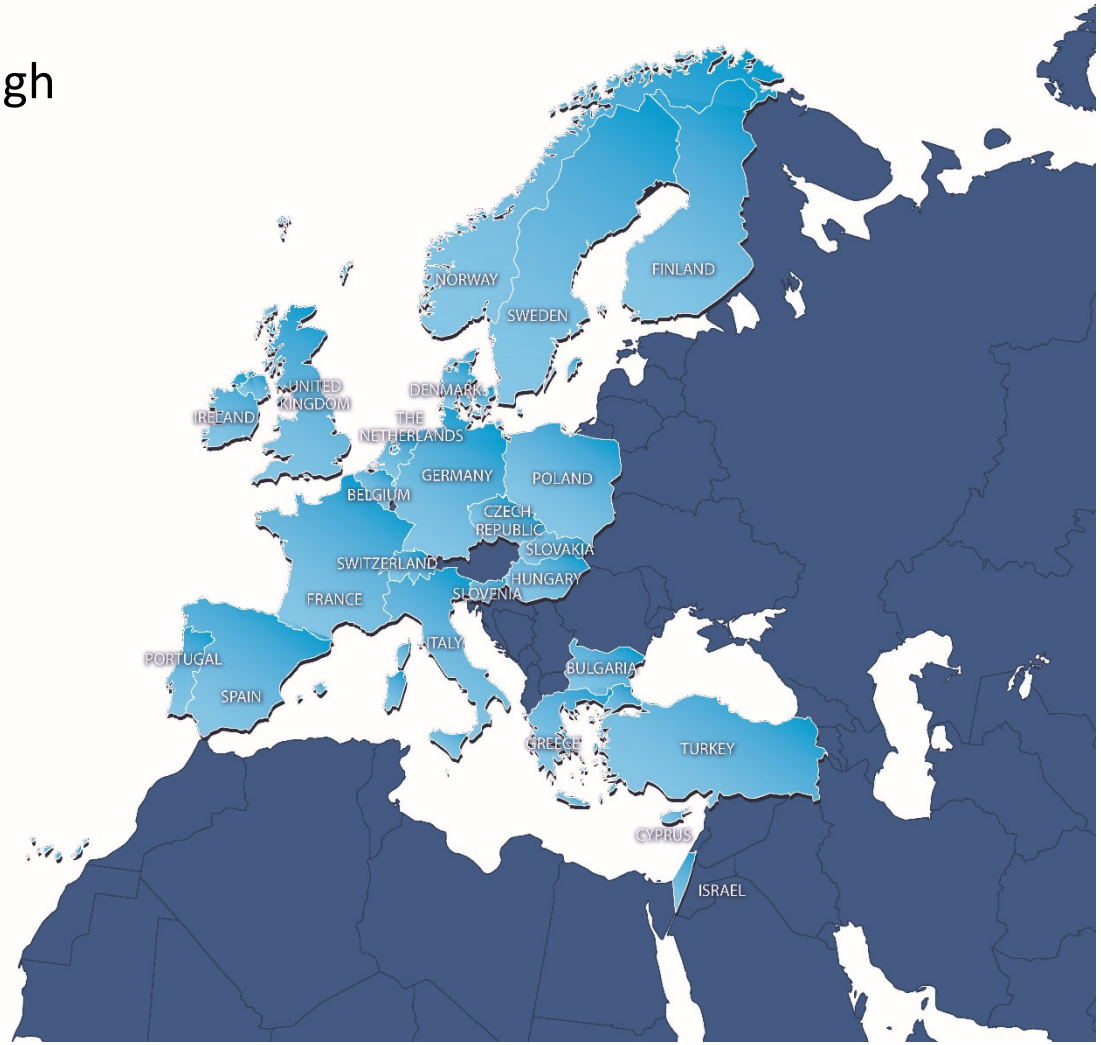
Mission:
enabling world-class science through large scale simulations

Offering:
HPC resources on leading edge capability systems

Resource award:
through a single and fair pan-European peer review process for open research



[9] PRACE



PRACE – History

2004-2006
- HPCEUR
- HET
- Scientific case

2007
- HPC in ESFRI roadmap
- HPC vision involving 15 countries
- PRACE initiative started

2008-2010
- PRACE preparatory phase project
- EC funding from the FP7

April 2010
- PRACE AISBL created
- HQ office in Brussels
- 1° call for project access launched

2010-2015
- PRACE IP projects (1-4)
- PCP for energy efficiency launched
- Regular calls (2 per year)
- Industry and SME support
- Expanding training (PATC)
- End of PRACE initial period

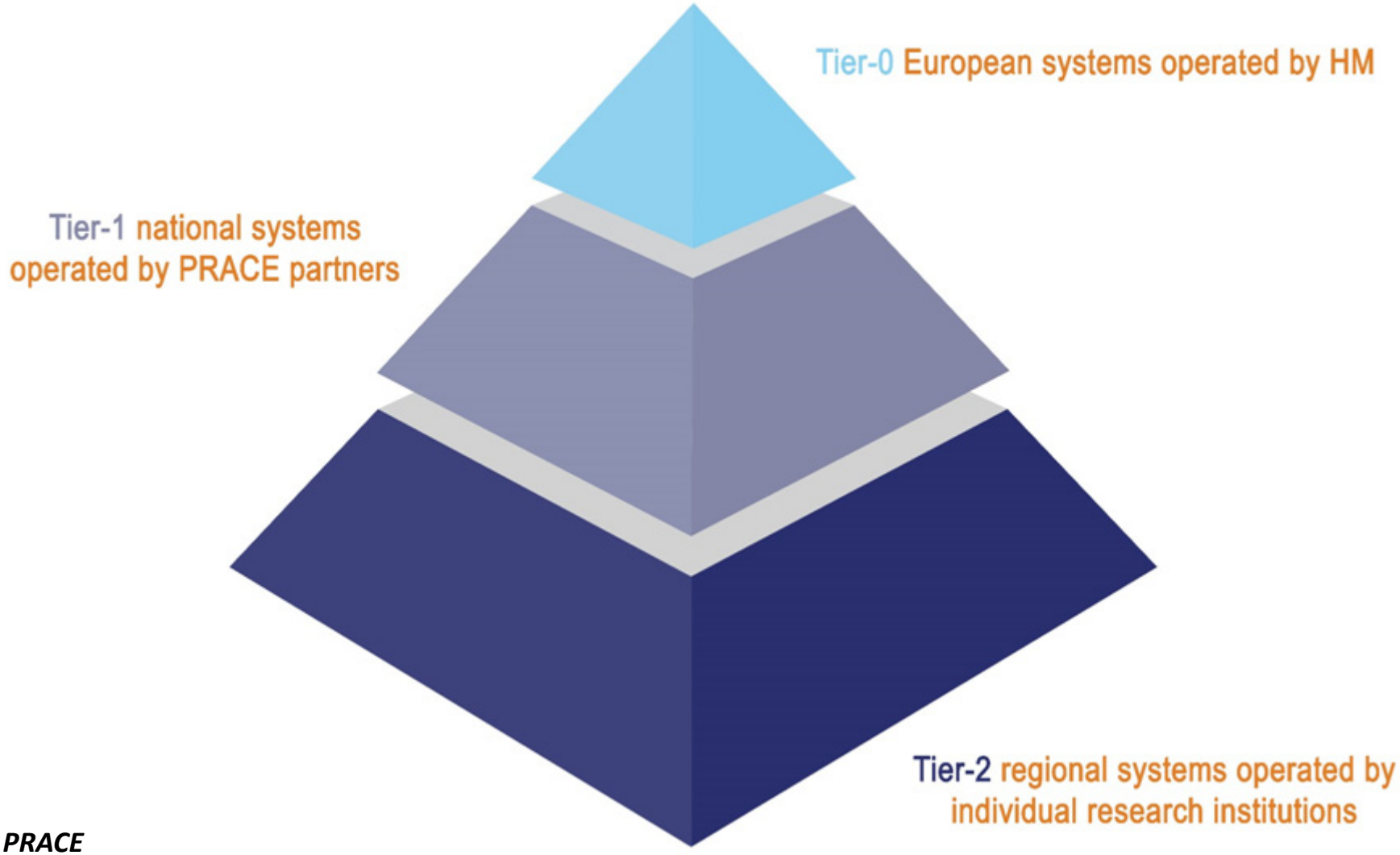
2016
- PRACE members agreement on new funding model
- New HM : Switzerland
- Call 14 for more than 2000 M core/h

2017
- PRACE 2 starts issuing Call 15
- PRACE-5IP kick-off

[9] PRACE

- **The Partnership for Advanced Computing in Europe (PRACE) is a persistent pan-European supercomputing infrastructure that offers HPC resources through a single and fair pan-European peer review process for open research**

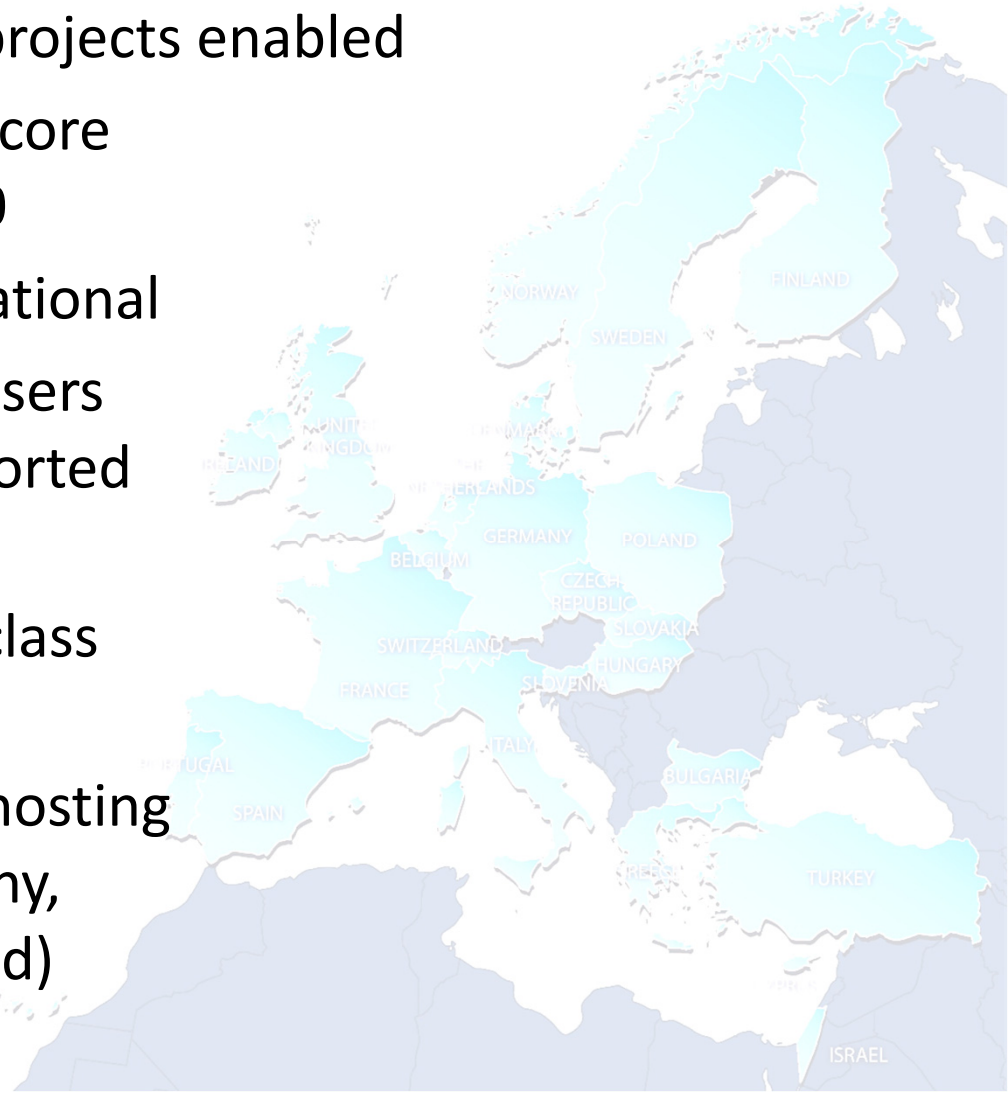
PRACE European vs. Regional Systems



[9] PRACE

PRACE Achievements

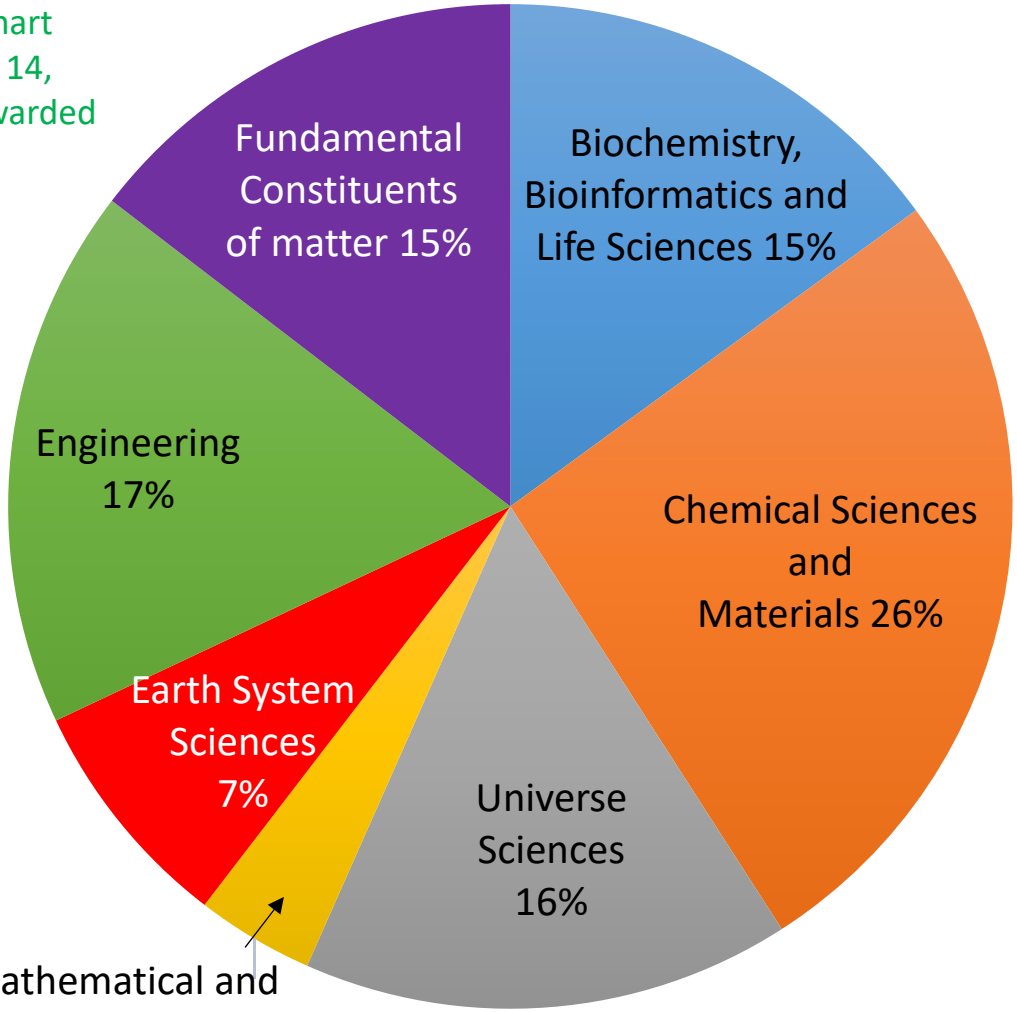
- More than 500 scientific projects enabled
- Over 14 thousand million core hours awarded since 2010
- Of which 63% are trans-national
- R&D access to industrial users with >50 companies supported
- More 60 Pflop/s of peak performance on 7 world-class systems
- 25 members, including 5 hosting members (France, Germany, Italy, Spain and Switzerland)



[9] PRACE

PRACE – Scientific Domains

Research Domain Pie Chart
up to and including Call 14,
% of total core hours awarded



[9] PRACE

PRACE – Current Services at a Glance

For academia and industry

Tier-0 systems (open R&D)
- Project Access
  1-3 years
- Preparatory Access
  Type A, B, C, D

Tier-1 systems (open R&D)
- DECI Programme

Application Enabling & Support
- Preparatory access Type C
- Preparatory access Type D
  - Tier-1 for Tier-0
- SHAPE
- HLST support

Training
- Training Portal
- PATC, PTC
- Seasonal Schools & on demand
- International HPC Summer School
- MOOC
- Code Vault
- Best Practice Guides
- White Papers

PRACE Partners

Communication, Dissemination, Outreach
- Website
- PR
- Scientific Communication
- Summer of HPC

Operation & Coordination of the common PRACE Operational Services
- Service Catalogue
- PRACE MD-VPN network
- Security

HPC Commissioning & Prototyping
- Technology Watch, PCP activity
- Infrastructure WS
- Best Practices
- UEABS

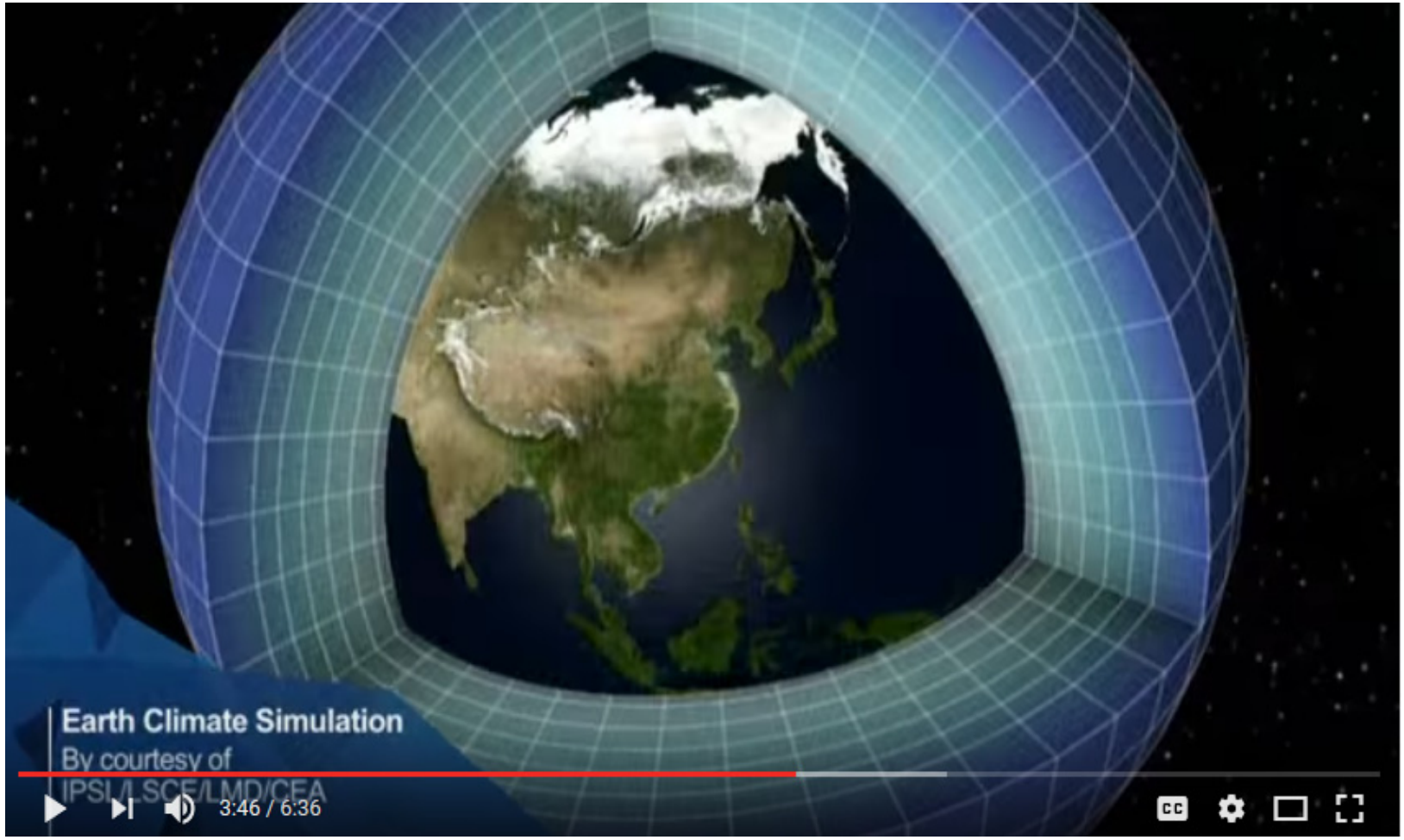
[9] PRACE

PRACE Advanced Training Centers (PATC) & Portal

- Selected Facts
  - More than 10 000 people trained by 6 PRACE Advanced Training Centers (PATC) and other events
  - Training portal consists of valuable material in all fields related to HPC & supercomputing
  - Easy search function to find materials of past events
  - Material of this training will be also available after the event

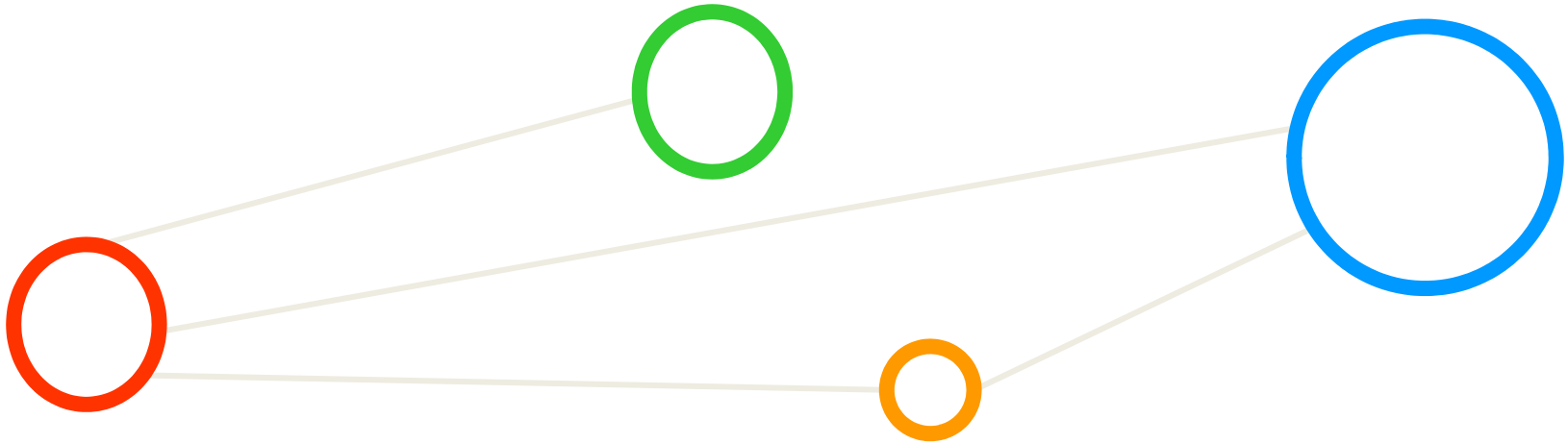
[10] PRACE Training Portal

[Video] PRACE – Introduction to Supercomputing



[6] YouTube, 'PRACE – Introduction to Supercomputing'

Parallel & Scalable Technologies



Python Programming Language

- Selected Benefits
  - Simple & flexible programming language
  - Is an interpreted powerful programming language
  - Has Efficient high-level data structures
  - Provides a simple but effective approach to object-oriented programming
  - Powerful libraries like ‘math’ library for inputs of functions with real numbers
  - Python script support is offered in almost every cloud computing environment as a programming language today

[13] Webpage Python

```
# usual start script hello world

print('Hello World!')
print("2 + 2 =", 2 + 2)
print("3 * 4 is", 3 * 4)
print('Goodbye!')
```

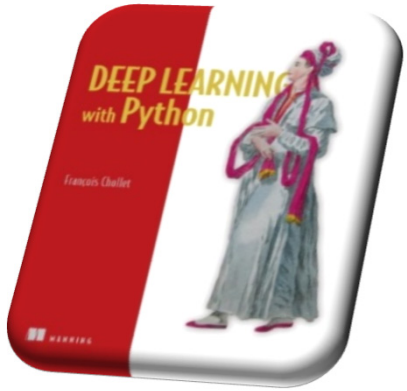
```
Hello World!
2 + 2 = 4
3 * 4 is 12
Goodbye!
```

- **Python is an ideal language for fast scripting and rapid application development that in turn makes it interesting for the machine learning modeling process and easy access to cloud resources**

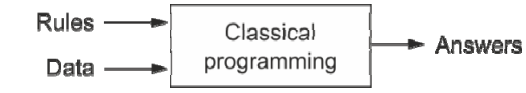
➢ **Our course assignments take advantage of Python, but nobody needs to be a full Python Expert**

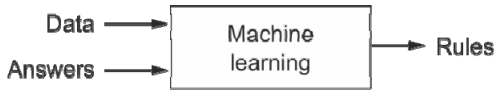
Python Programming Language – Importance & Reasoning

- Key Benefits for this course (and beyond)
  - Relatively small number of lines of code needed
  - Great libraries & community support (e.g. numpy, tensorflow, keras, etc.)
  - Work with many students reveal: qucky & easy to learn for experiments



[16] F. Chollet, 'Deep Learning with Python' Book





- **The machine learning modeling process in general and the deep learning modeling process in particular requires iterative and highly flexible approaches when working with 'big data'**
- **E.g. network topology prototyping, hyper-parameter tuning, data sampling, etc.**

➢ **Our course assignments take advantage of Python, but nobody needs to be a full Python Expert**

Powerful NumPy Python Library

- Selected features
  - Useful linear algebra and random number capabilities
  - Supports powerful N-dimensional array objects
  - Interesting broadcasting functions
  - Tools for integrating C/C++ and Fortran code
  - Particularly nicely supports work on vectors & matrices that are useful when working with machine learning & big data

[14] NumPy Library Web Page

```
import numpy as np

vector = np.random.randn(1,3)
print(vector)
```

```
[[1.01803478 0.21455826 0.5541203 ]]
```

(small number of code lines to create & print a 1x3 row vector with 3 random values)

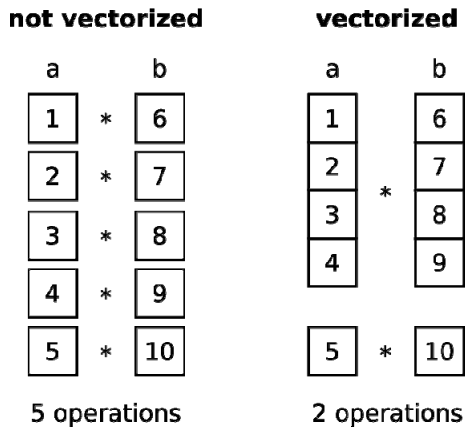
- **NumPy is one of the most important Python libraries used in cloud computing and big data analysis**
- **NumPy is used as an efficient multi-dimensional container of generic data, vectors, matrices, etc.**
- **Instead of math library used for functions with real numbers we use NumPy for vectors & matrices**

➢ **Our course assignments take advantage of NumPy for various aspectcs like working with vectors**

Simple Python/NumPy & Vectorization

- ‘Small-scale example‘ of the power of ‘parallelization‘
  - Enables element-wise computations at the same time (aka in parallel)
  - ‘small-scale‘ since we are still within one computer – but perform operations in parallel on different data
  - Supported on the hardware-level (i.e. register, etc.) & less lines of code
  - Impact matters much for ‘big data‘ computing (e.g. machine learning & deep learning : matrix/vector multiplications)

[15] Blog



- **Vectorization in Python uses optimized & pre-compiled C code to perform operations over data sequences of the same data type (i.e. numpy array/vector instead of Python tuples or lists)**
- **Vectorized functions are multiple times faster than using operations in explicit for-loop statements**
- **Avoid explicit for-loops via vectorized Single Instruction Multiple Data (SIMD) functions in NumPy**

➢ **Underlying machine learning frameworks we use in the tutorial take advantage of vectorization**

Simple Python/NumPy & Vectorization Example

```
import numpy as np
import time

vector_a = np.random.rand(100000)
vector_b = np.random.rand(100000)

t_start = time.time()
vector_c = np.dot(vector_a, vector_b) # vectorized function
t_end = time.time()

print(vector_c)
print("Computing time using vectorization: " + str(1000*(t_end-t_start)) + " ms")
```

```
25013.11080224216
Computing time using vectorization: 15.622138977050781 ms
```

```
t_start = time.time()
vector_c = 0
for i in range (100000):
  vector_c += vector_a[i] * vector_b[i]
t_end = time.time()

print(vector_c)
print("Computing time using explicit for-loops: " + str(1000*(t_end-t_start)) + " ms")
```

```
25013.110802242478
Computing time using explicit for-loops: 55.59992790222168 ms
```

Jupyter Tool – Interactive Data Analysis using Clouds

- Selected Facts
  - Interactive & Web-based Computing tool using the Web browser on local machine
  - Facilitates access to different available cloud platforms
  - Often used in data analysis
  - Part of the Anaconda distribution

[17] Jupyther Web page

```
In [ ]: # usual start script hello world

        print('Hello World!')
        print("2 + 2 =", 2 + 2)
        print("3 * 4 is", 3 * 4)
        print('Goodbye!')
```

(press 'shift + enter')

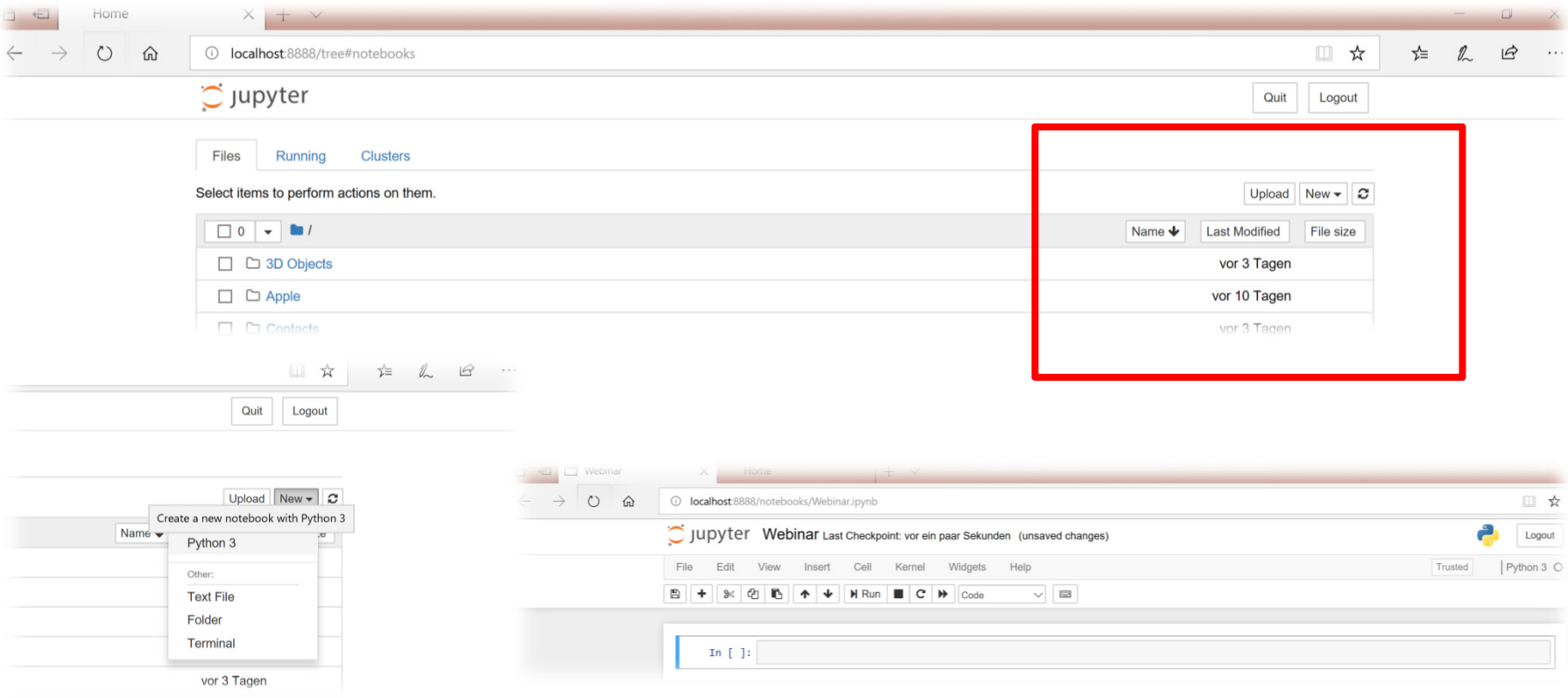
```
In [1]: # usual start script hello world

        print('Hello World!')
        print("2 + 2 =", 2 + 2)
        print("3 * 4 is", 3 * 4)
        print('Goodbye!')

Hello World!
2 + 2 = 4
3 * 4 is 12
Goodbye!
```

- **Jupyther enables the creation and sharing of source codes using so-called 'Jupyther notebooks'**
- **Jupyther notebooks contain live code, equations, formulas, visualizations & explanatory text**

➢ **The Jupyter tool is used in practical lectures & our assignments via various Jupyther notebooks**

Jupyther Tool Example – Starting a Python Command Shell

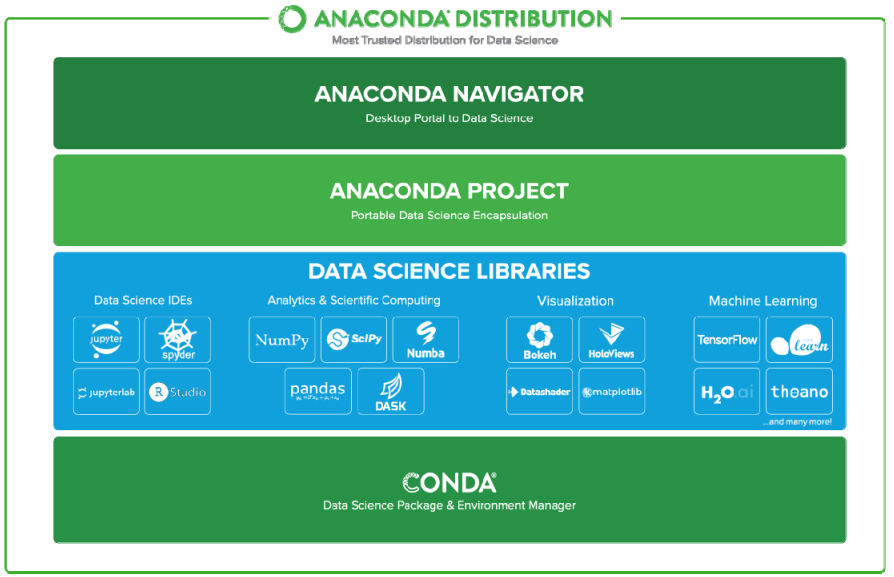
- Startup Jupyter
  - E.g. new menu → Python 3



Anaconda Distribution (for ‘serial computing‘ at home)

- Selected Facts
  - Free & open source distribution
  - Installs Python as well separate from other installed versions
  - Installation of 1,400+ data science packages for Python & Statistical Computing with R
  - Available for Windows, Mac & Linux
  - Anaconda significantly simplifies the Windows setup

[18] Anaconda Web page



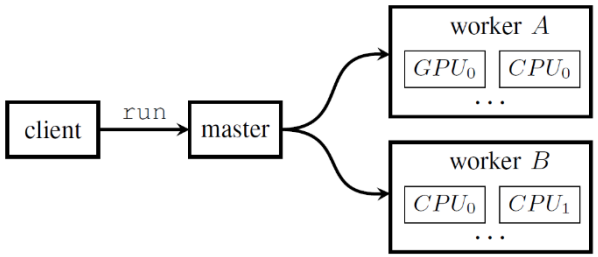
- **The Anaconda distribution includes multiple tools for installing & updating Python & its package**
- **Anaconda represents one of the most popular Python data science platforms (~6 million users)**

➢ **Recommendation to install Anaconda to experiment locally on the laptop before using Clouds**

Using Python & TensorFlow in Tutorial

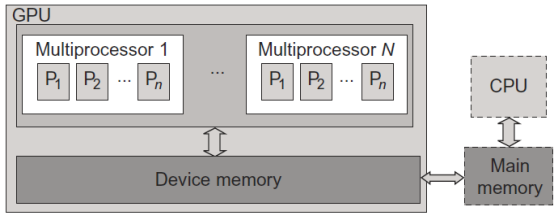
- Cloud Computing
  - Enables large-scale deep learning from ‘big data‘
  - Deep learning library TensorFlow

[19] Tensorflow Deep Learning Framework



[20] A Tour of Tensorflow

(works well on graphical processing units = GPUs)



[21] Distributed & Cloud Computing Book

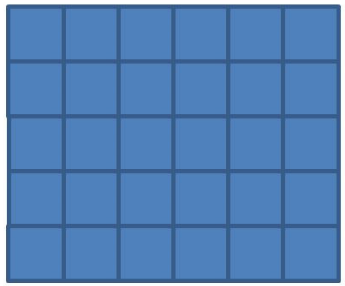
- **Tensorflow is an open source library for deep learning models using a flow graph approach**
- **Tensorflow nodes model mathematical operations and graph edges between the nodes are so-called tensors (also known as multi-dimensional arrays)**
- **The Tensorflow tool supports the use of CPUs and GPUs (much more faster than CPU versions)**
- **Tensorflow work with the high-level deep learning tool Keras in order to create models fast**

➢ **Our course assignments take advantage of TensorFlow in conjunction with Python scripts & libs**

Deep Learning in Clouds – What are Tensors?

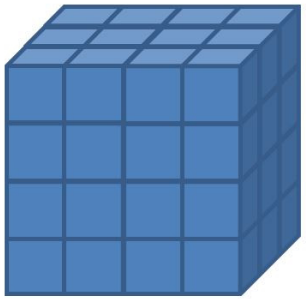
(one dimensional tensor)

(vector of dimension [5])



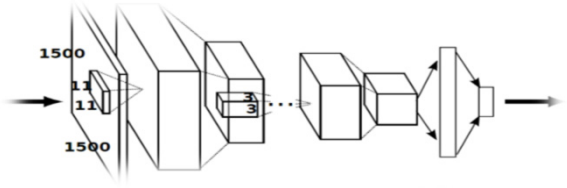
(two dimensional tensor)

(matrix of dimensions [5,6])



(three dimensional tensor)

(tensor of dimension [4,4,3])



(deep learning network
use tensors much and
NumPy is good to work
in Python scripts with these)

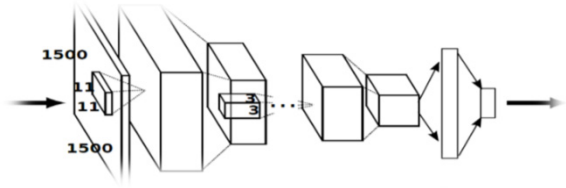
[22] Big Data Tips, What is a Tensor?

- A Tensor is nothing else than a multi-dimensional array often used in scientific & engineering environments
- Tensors are best understood when comparing it with vectors or matrices and their dimensions
- Those tensors 'flow' through the deep learning network during the optimization / learning & inference process

➢ Tensors will be used throughout the tutorial for data & used during machine learning methods

Using Python & Keras on top of TensorFlow in Tutorial

- Cloud Computing
  - Enables large-scale deep learning from ‘big data‘
  - Deep learning library Keras on top of TensorFlow




```
keras.layers.Dense(units,
                   activation=None,
                   use_bias=True,
                   kernel_initializer='glorot_uniform',
                   bias_initializer='zeros',
                   kernel_regularizer=None,
                   bias_regularizer=None,
                   activity_regularizer=None,
                   kernel_constraint=None,
                   bias_constraint=None)
```

```
keras.optimizers.SGD(lr=0.01,
                     momentum=0.0,
                     decay=0.0,
                     nesterov=False)
```

[1] Keras Python Deep Learning Library

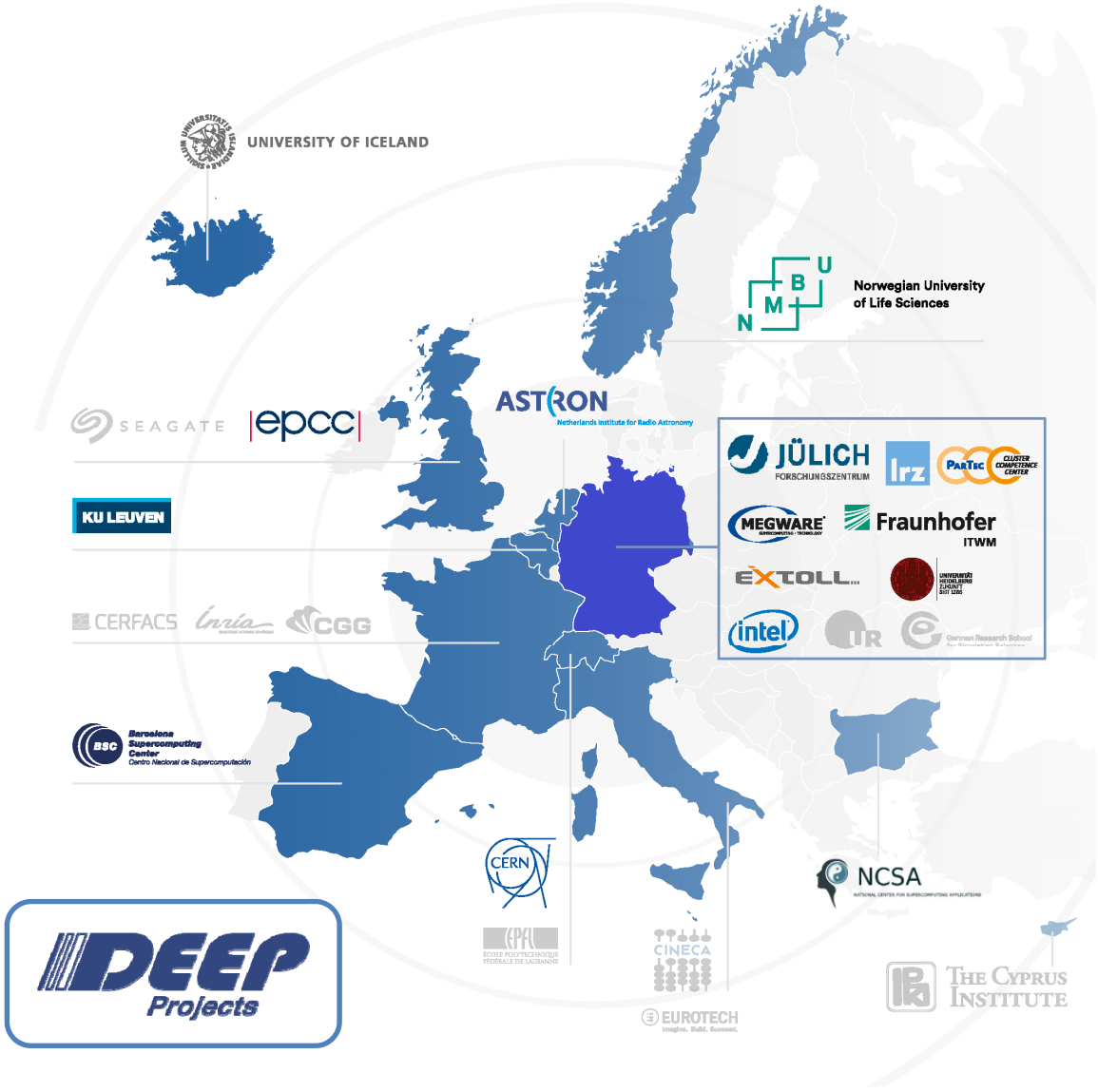
- **Keras is a high-level deep learning library implemented in Python that works on top of existing other rather low-level deep learning frameworks like Tensorflow, CNTK, or Theano**
- **The key idea behind the Keras tool is to enable faster experimentation with deep networks**
- **Created deep learning models run seamlessly on CPU and GPU via low-level frameworks**

➢ **Our course assignments take advantage of Keras in conjunction with Python scripts & libs**

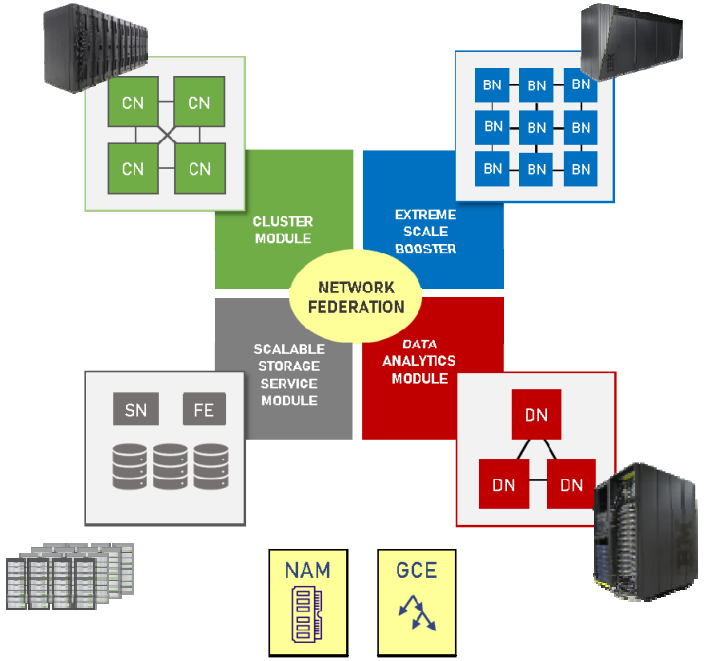
DEEP Projects & Partners

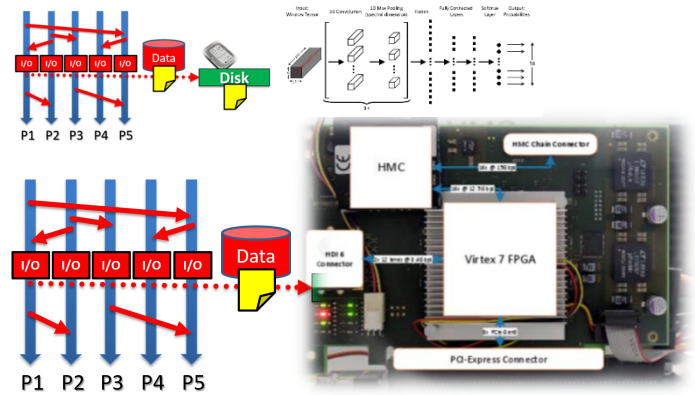
- DEEP
  - Dynamic Exascale Entry Platform
- 3 EU Exascale projects
  DEEP
  DEEP-ER
  DEEP-EST
- 27 partners
  Coordinated by JSC
- EU-funding: 30 M€
  JSC-part > 5,3 M€
- Nov 2011 – Mar 2021

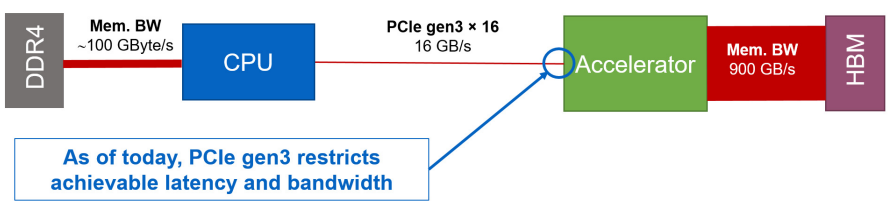
[24] DEEP-EST EU Project

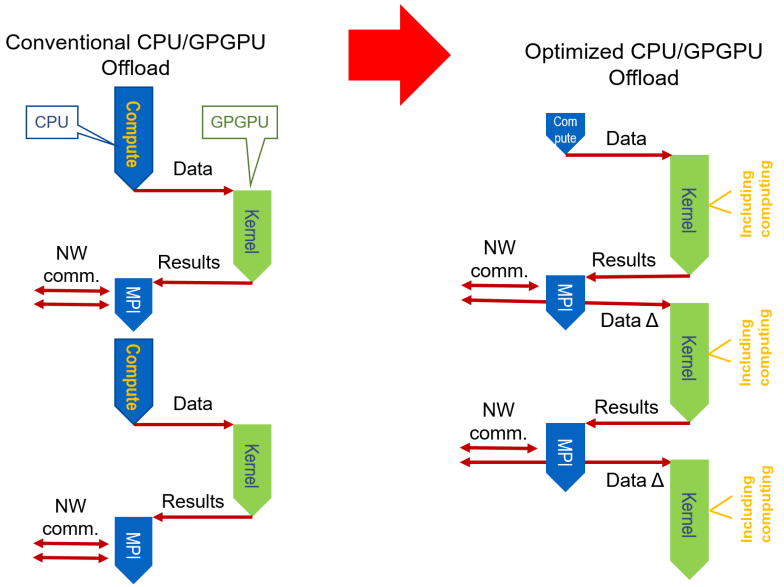


DEEP-EST Approach – Modular Supercomputing





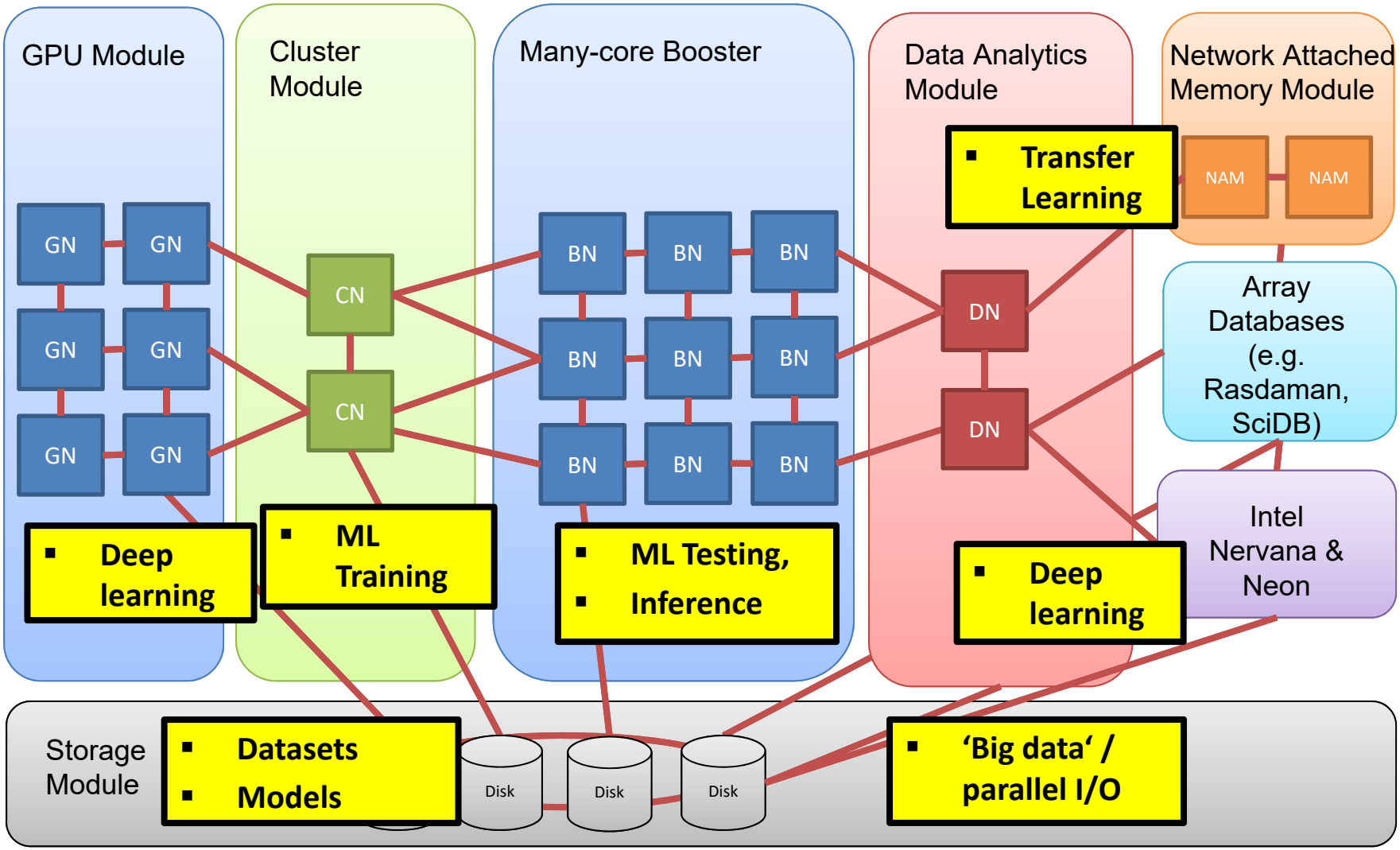


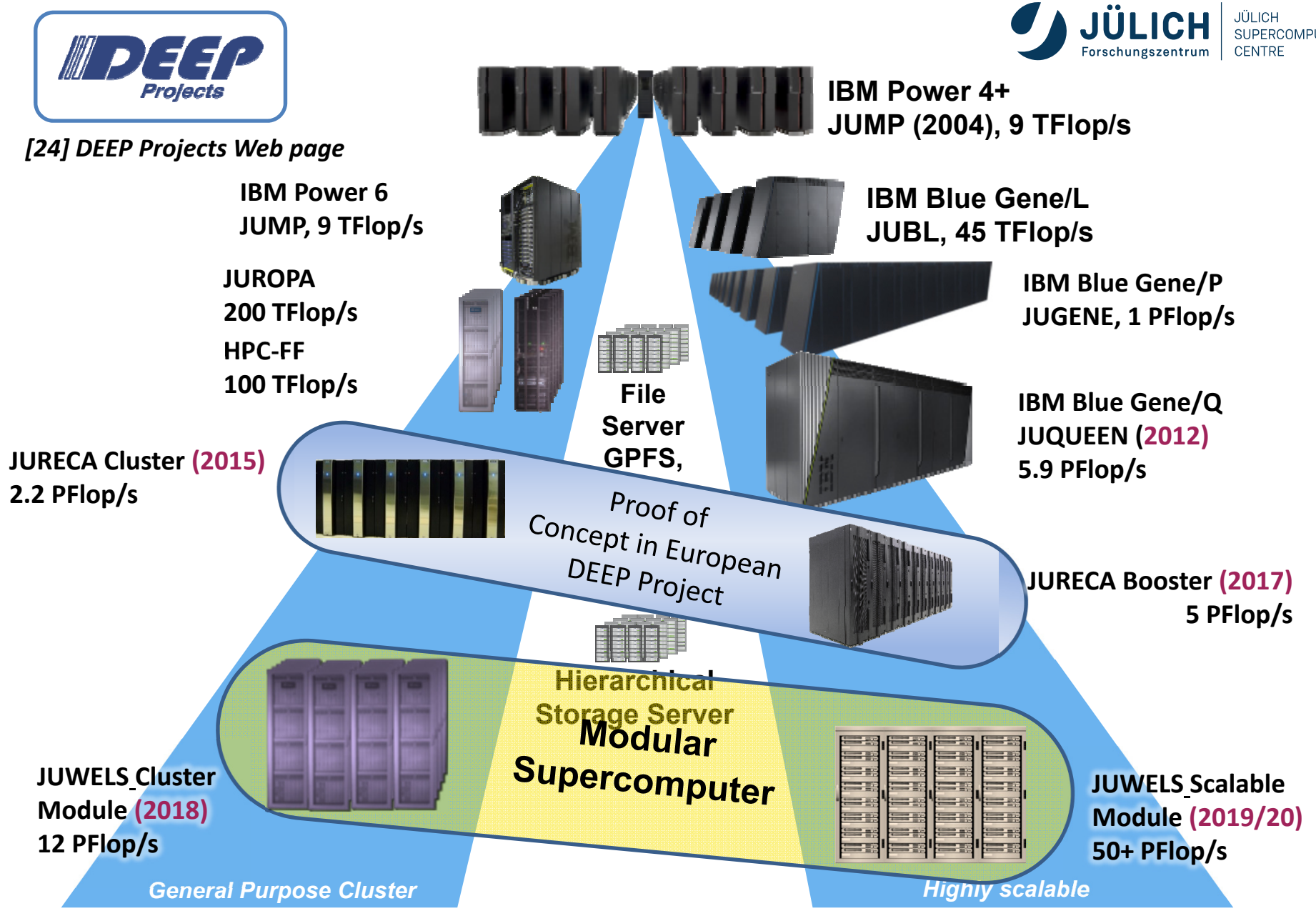


[24] DEEP-EST EU Project

[27] E. Erlingsson, M. Riedel et al., IEEE MIPRO Conference, 2018

DEEP-EST EU Project & Modular Supercomputing




DEEP Projects
JÜLICH Forschungszentrum
JÜLICH SUPERCOMPUTING CENTRE
[24] DEEP Projects Web page
IBM Power 4+
JUMP (2004), 9 TFlop/s
IBM Power 6
JUMP, 9 TFlop/s
IBM Blue Gene/L
JUBL, 45 TFlop/s
JUROPA
200 TFlop/s
HPC-FF
100 TFlop/s
IBM Blue Gene/P
JUGENE, 1 PFlop/s
File
Server
GPFS,
IBM Blue Gene/Q
JUQUEEN (2012)
5.9 PFlop/s
JURECA Cluster (2015)
2.2 PFlop/s
Proof of
Concept in European
DEEP Project
JURECA Booster (2017)
5 PFlop/s
Hierarchical
Storage Server
Modular
Supercomputer
JUWELS Cluster
Module (2018)
12 PFlop/s
JUWELS Scalable
Module (2019/20)
50+ PFlop/s
General Purpose Cluster
Highly scalable

Tutorial: JURECA HPC System

[24] DEEP-EST EU Project

JURECA

- T-Platforms V210 blade server solution
  - Dual-socket Intel Xeon Haswell CPUs
- Mellanox InfiniBand EDR network
- Peak: 1.8 PF (CPUs) + 0.4 PF (GPUs)
- 281 TiB main memory
- 100 GBps storage bandwidth

- Dell PowerEdge C6320P solution
  - Intel Xeon Phi "Knights Landing" 7250-F
- Intel OPA network
- Peak: 5 PF
- 157 TiB main memory + 26 TiB MCDRAM
- 200 GBps storage BW

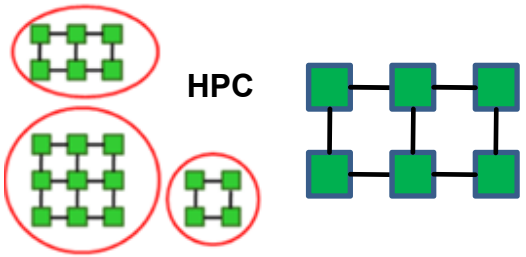
JURECA CLUSTER & BOOSTER Module @ JSC

Tutorial Machine: JSC JURECA System – CLUSTER Module

- Characteristics
  - Login nodes with 256 GB memory per node
  - 45,216 CPU cores
  - 1.8 (CPU) + 0.44 (GPU) Petaflop/s peak performance
  - Two Intel Xeon E5-2680 v3 Haswell CPUs per node: 2 x 12 cores, 2.5 GhZ
  - 75 compute nodes equipped with two NVIDIA K80 GPUs (2 x 4992 CUDA cores)
- Architecture & Network
  - Based on T-Platforms V-class server architecture
  - Mellanox EDR InfiniBand high-speed network with non-blocking fat tree topology
  - 100 GiB per second storage connection to JUST

[24] JURECA HPC System

- **Use our ssh keys to get an access and use reservation**
- **Put the private key into your ./ssh directory (UNIX)**
- **Use the private key with your putty tool (Windows)**

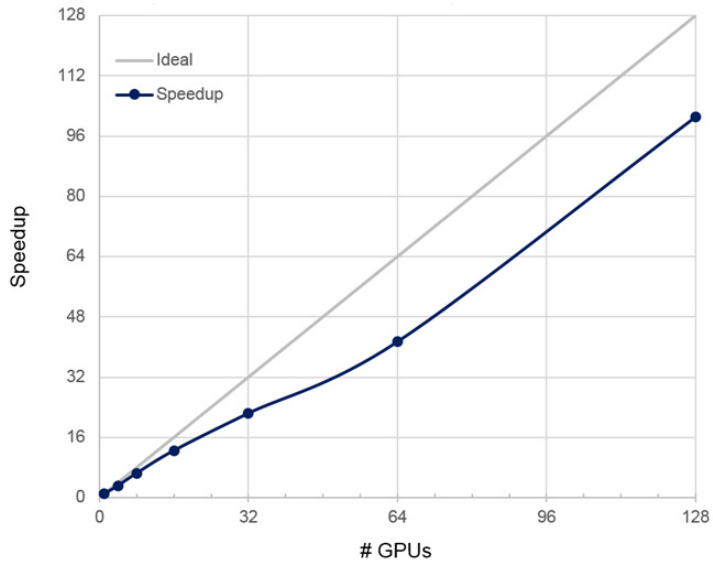


Interconnect GPUs with Horovod – JURECA System Example

- Simple Image Benchmark on JURECA JSC HPC System
  - 75 x 2 NVIDIA Tesla K80/node – dual GPU design
  - 1.2 mio images with 224 x 224 pixels

#GPUs	images/s	speedup	Performance per GPU [images/s]
1	55	1.0	55
4	178	3.2	44.5
8	357	6.5	44.63
16	689	12.5	43.06
32	1230	22.4	38.44
64	2276	41.4	35.56
128	5562	101.1	43.45

(absolute number of images per second and relative speedup normalized to 1 GPU are given)

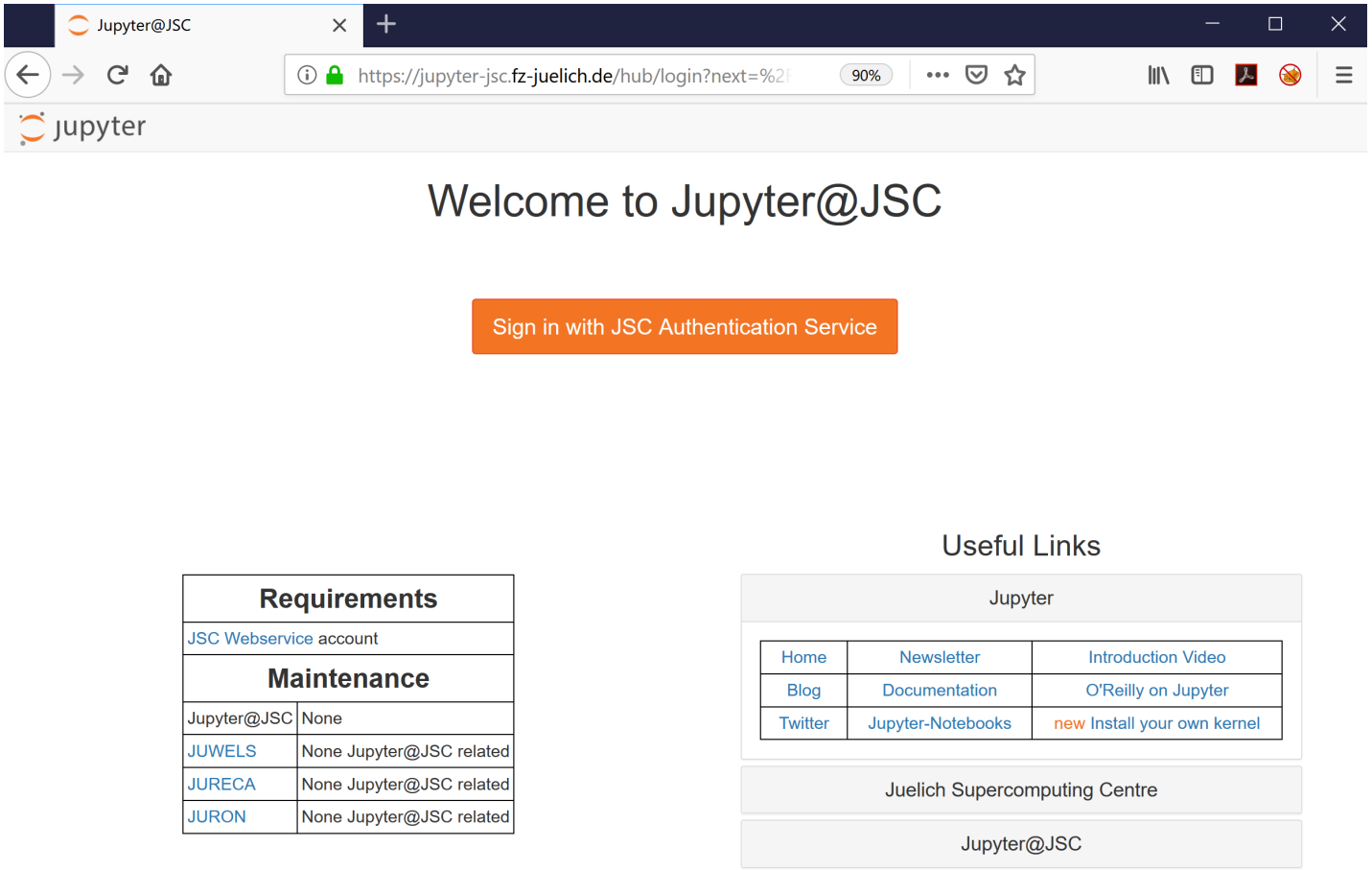


[23] A. Sergeev, M. Del Balso,'Horovod', 2018

(setup: TensorFlow 1.4, Python 2.7, CUDA 8, cuDNN 6, Horovod 0.11.2, MVAPICH-2.2-GDR)

- **Open source tool Horovod enables distributed deep learning with TensorFlow / Keras**
- **Machine & Deep Learning: speed-up is just secondary goal after 1st goal accuracy**
- **Speed-up & parallelization good for faster hyperparameter tuning, training, inference**
- **Third goal is to avoid much feature engineering through 'feature learning'**

Jupyter Access to JURECA System



Exercise Plan A – Using Jupyter

➢ **Use with your prepared account: https://jupyter-jsc.fz-juelich.de**

Exercise Plan B – Login to JURECA with SSH

JURECA System – SSH Login

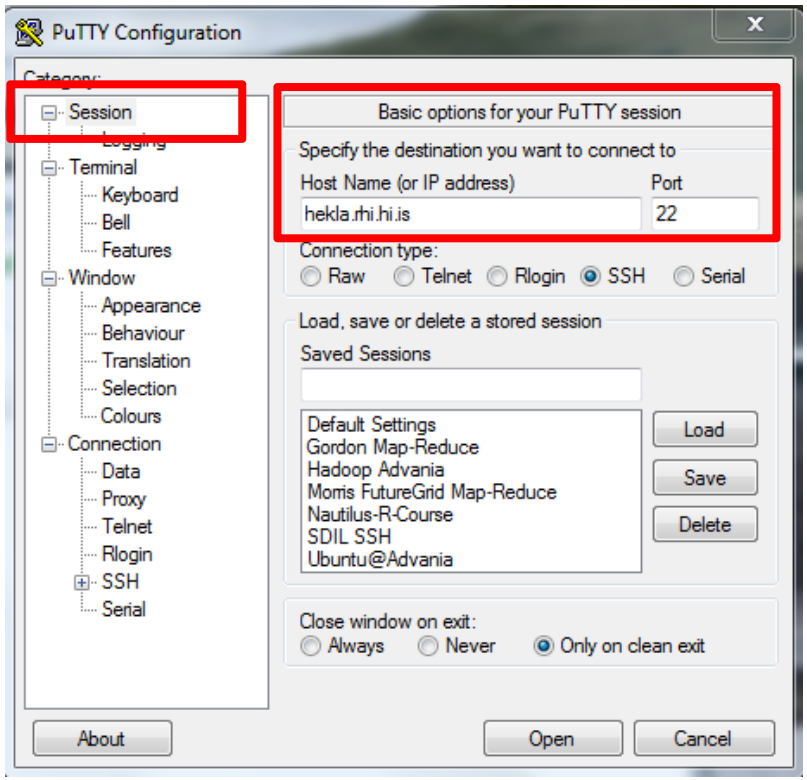
- Use your account train004 - train050
- Windows: use putty or MobaXterm (better with x-server)
- UNIX: ssh yourname@jureca.fz-juelich.de
- Example

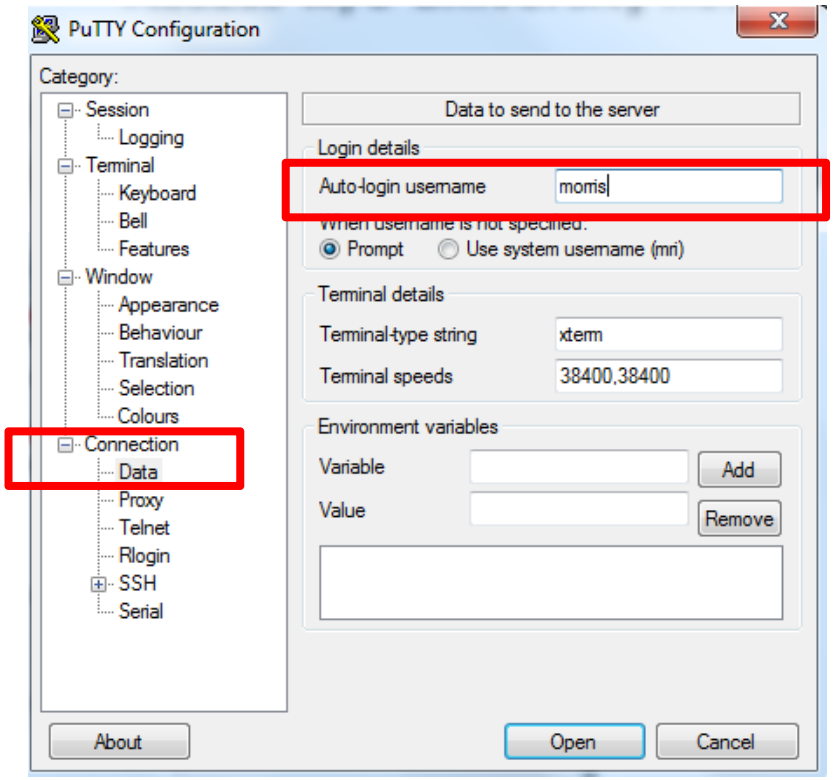
```
adminuser@linux-8djg:~> ssh train001@jureca.fz-juelich.de
Warning: the ECDSA host key for 'jureca.fz-juelich.de' differs from the key for the IP address '134.94.33.
9'
Offending key for IP in /home/adminuser/.ssh/known_hosts:12
Matching host key in /home/adminuser/.ssh/known_hosts:19
Are you sure you want to continue connecting (yes/no)? yes
]Last login: Mon Aug 21 14:29:03 2017 from zam2036.zam.kfa-juelich.de
*******************************************************************************
*                            Welcome to JURECA                                *
*                                                                             *
*  Information about the system, latest changes, user documentation and FAQs:  *
*                  http://www.fz-juelich.de/ias/jsc/jureca                    *
*******************************************************************************
*                           ### Known Issues ###                              *
*                                                                             *
* An up-to-date list of known issues on the system is maintained at           *
*               http://www.fz-juelich.de/ias/jsc/jureca-known-issues          *
* Open issues:                                                                *
*   - Intel compiler error with std::valarray and                             *
*     optimized headers, added 2016-03-20                                     *
```

Using SSH Clients for Windows

- Example: using the Putty SSH client
  (other SSH tools exist, e.g. could be MoabXTerm, etc.)

[25] PUTTY tool



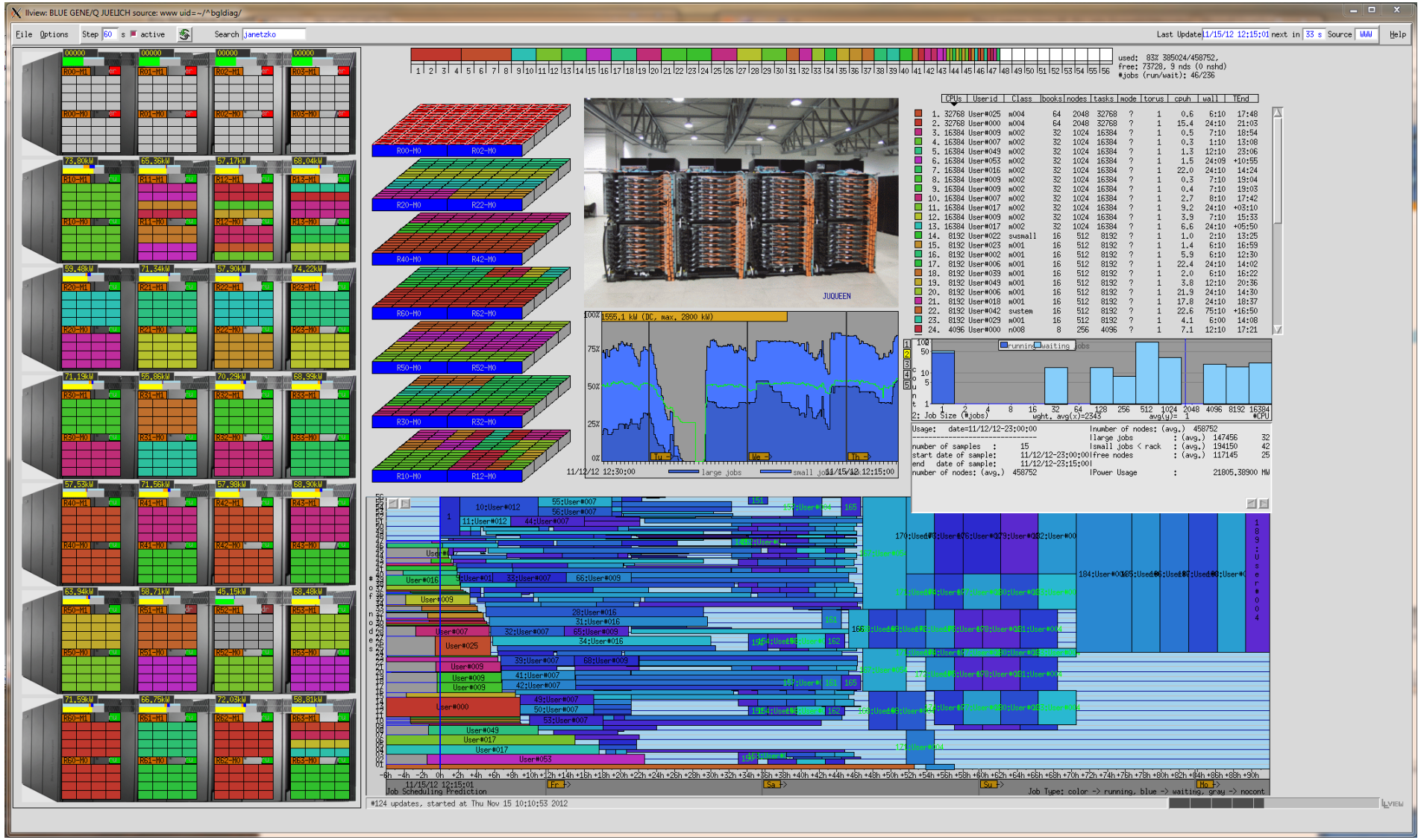


Scheduling Principles – SLURM Scheduler in Tutorial

- HPC Systems are typically not used in an interactive fashion
  - Program application starts ‘processes‘ on processors (‘do a job for a user‘)
  - Users of HPC systems send ‘job scripts‘ to schedulers to start programs
  - Scheduling enables the sharing of the HPC system with other users
  - Closely related to Operating Systems with a wide varity of algorithms
- E.g. First Come First Serve (FCFS)
  - Queues processes in the order that they arrive in the ready queue.
- E.g. Backfilling
  - Enables to maximize cluster utilization and throughput
  - Scheduler searches to find jobs that can fill gaps in the schedule
  - Smaller jobs farther back in the queue run ahead of a job waiting at the front of the queue (but this job should not be delayed by backfilling!)

- **Scheduling is the method by which user processes are given access to processor time (shared)**

Example: Supercomputer BlueGene/Q



[26] LLView Tool

Jobscript JURECA Example & Reservation

```
#!/bin/bash -x
#SBATCH--nodes=2
#SBATCH--ntasks=48
#SBATCH--ntasks-per-node=24
#SBATCH--output=mpi-out.%j
#SBATCH--error=mpi-err.%j
#SBATCH--time=06:00:00
#SBATCH--partition=batch
#SBATCH--mail-user=m.riedel@fz-juelich.de
#SBATCH--mail-type=ALL
#SBATCH--job-name=train-indianpines-2-48-24
#SBATCH--reservation=ml-hpc-1

### location executable
PISVM=/homea/hpclab/train001/tools/pisvm-1.2.1/pisvm-train
#PISVM=/homeb/zam/mriedel/tools/pisvm-1.2.1jurecanew/pisvm-train

### location data
TRAINDATA=/homea/hpclab/train001/data/indianpines/indian_processed_training.el
#TRAINDATA=/homeb/zam/mriedel/bigdata/172-indianpinesrawproc/indian_processed_training.el

### submit
srun $PISVM -D -o 1024 -q 512 -c 100 -g 8 -t 2 -m 1024 -s 0 $TRAINDATA
```

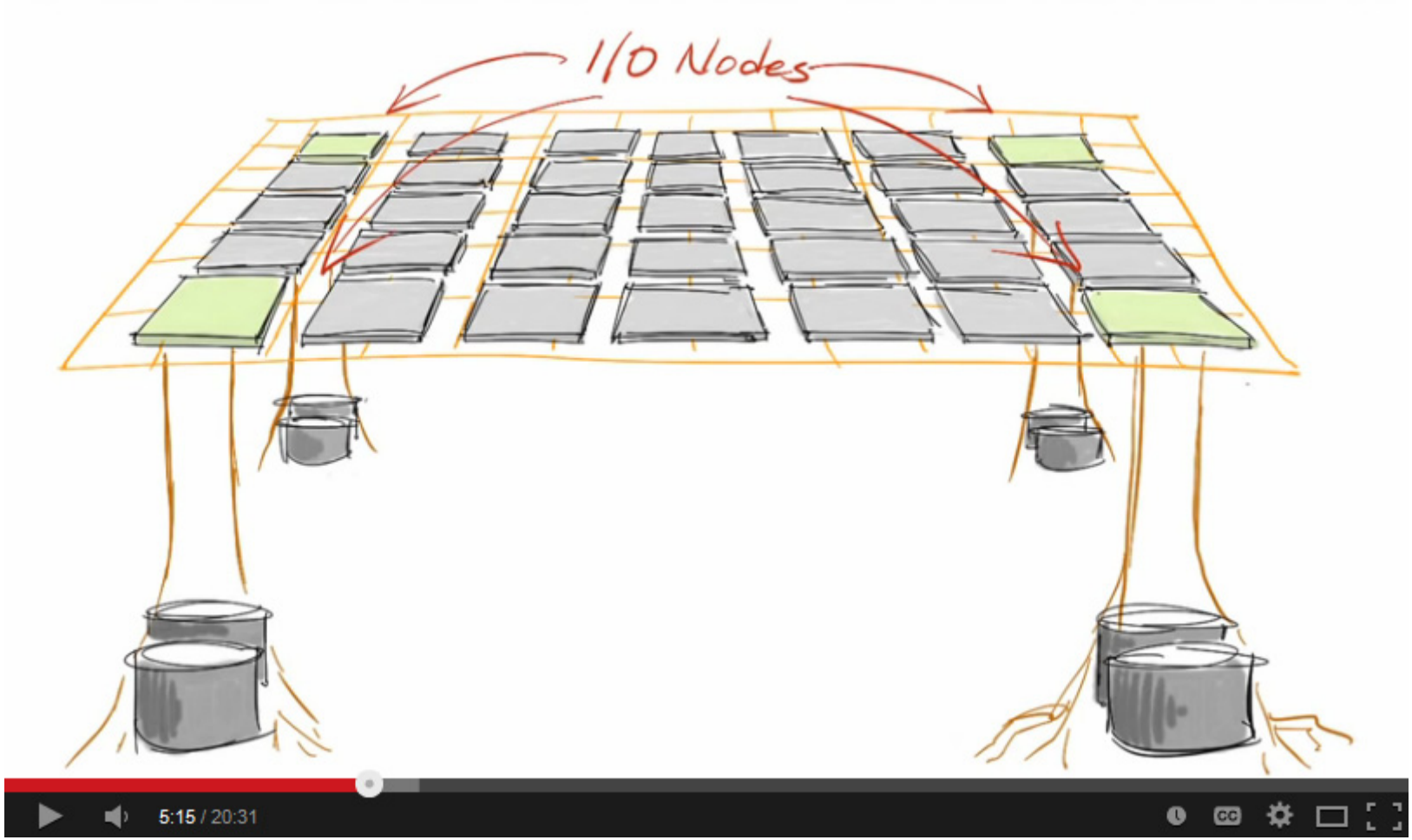
- Every day the reservation string is changed on the HPC systems (below)
- Change the number of nodes and tasks to use more or less CPUs for jobs
- Use the command sbatch <jobsript> in order to submit parallel jobs to the supercomputer and remember your <job id> returned
- Use the command squeue –u <userid> in order to check the status of your parallel job

HPC Environment – Modules

- Module environment tool
  - Avoids to manually setup environment information for every application
  - Simplifies shell initialization and lets users easily modify their environment
- Module avail
  - Lists all available modules on the HPC system (e.g. compilers, MPI, etc.)
- Module spider
  - Find modules in the installed set of modules and more information
- Module load
  - Loads particular modules into the current work environment, E.g.:

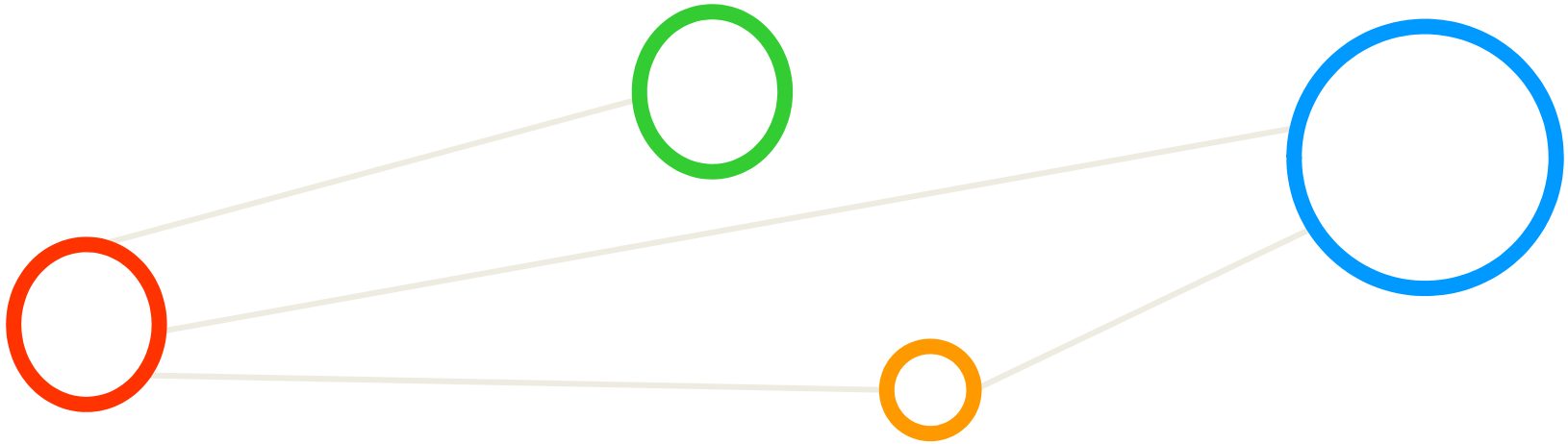
```
[train001@jrl12 ~]$ module load GCC

Due to MODULEPATH changes, the following have been reloaded:
  1) binutils/.2.29

The following have been reloaded with a version change:
  1) GCCcore/.5.4.0 => GCCcore/.7.2.0

[train001@jrl12 ~]$ module load ParaStationMPI/5.2.0-1
[train001@jrl12 ~]$ module load HDF5/1.8.19
```

[Video] Parallel I/O with I/O Nodes



[11] Simplifying HPC Architectures, YouTube Video

Appendix A – Selected NumPy Commands



Powerful NumPy Python Library

- Selected features
  - Useful linear algebra and random number capabilities
  - Supports powerful N-dimensional array objects
  - Interesting broadcasting functions
  - Tools for integrating C/C++ and Fortran code
  - Particularly nicely supports work on vectors & matrices that are useful when working with machine learning & big data

[14] NumPy Library Web Page

```
import numpy as np

vector = np.random.randn(1,3)
print(vector)
```

```
[[1.01803478 0.21455826 0.5541203 ]]
```

(small number of code lines to create & print a 1x3 row vector with 3 random values)

- **NumPy is one of the most important Python libraries used in cloud computing and big data analysis**
- **NumPy is used as an efficient multi-dimensional container of generic data, vectors, matrices, etc.**
- **Instead of math library used for functions with real numbers we use NumPy for vectors & matrices**

➢ **Our course assignments take advantage of NumPy for various aspectcs like working with vectors**

Basic Variables

In [1]:
```
# usual start script hello world

print('Hello World!')
print("2 + 2 =", 2 + 2)
print("3 * 4 is", 3 * 4)
print('Goodbye!')
```

```
Hello World!
2 + 2 = 4
3 * 4 is 12
Goodbye!
```

Basic Loops

- For Loop
  - Syntax:
    for <variable> in <sequence>:
      <statements>
    else:
      <statements>

```
In [2]: languages = ["German", "Spanish", "Icelandic", "Italian"]
        for x in languages:
            print(x)

        German
        Spanish
        Icelandic
        Italian
```

If Statements

In [5]:
```
languages = ["German", "Spanish", "Icelandic", "Italian"]
for x in languages:
    print(x)
    if x == "Icelandic":
        print("Welcome to University of Iceland!")
        break
```

```
German
Spanish
Icelandic
Welcome to University of Iceland!
```

Working with Vectors – Load NumPy Library

```
import NumPy as np

vector = np.random.randn(1,5)
print(vector)
```

(could be imported in another cell than the current one – was it really executed? otherwise np is unknown – the cells are connected throughout the script)

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-1-4a91256c61d5> in <module>()
----> 1 import NumPy as np
      2
      3 vector = np.random.randn(1,5)
      4 print(vector)

ModuleNotFoundError: No module named 'NumPy'
```

```
import numpy as np

vector = np.random.randn(1,5)
print(vector)
```

(notice Python & case sensitivity)

```
[[ 0.5540269  -0.8310974  -2.14350484 -0.62055725  1.19389052]]
```

Vectors

- Note difference rank 1 arrays & true vectors
  - Difference is important and lead to many bugs in scripts when working with machine & deep learning models

```
In [3]: import numpy as np

        vector = np.random.randn(5)
        print(vector)
        print(vector.shape)
```

```
[-1.53573965  0.51872255 -0.82611958  0.94804364 -0.5318634 ]
(5,)
```

```
In [5]: vectornew = np.random.randn(1,5)
        print(vectornew)
        print(vectornew.shape)
```

```
[[ 1.61773902 -0.75865461  0.81150118  0.73271955 -1.75471367]]
(1, 5)
```

Matrices

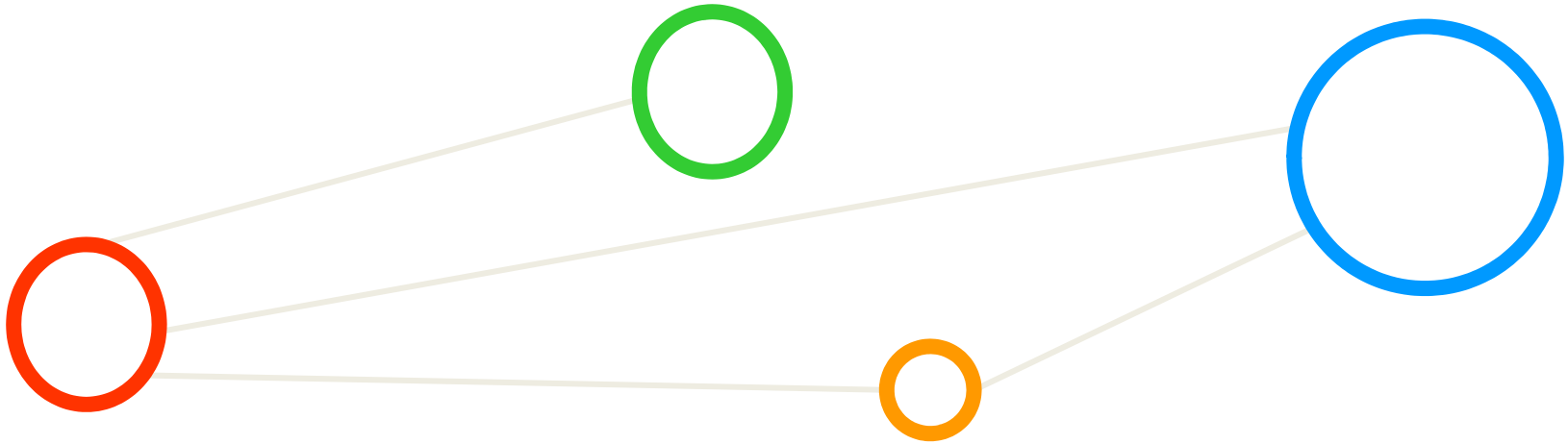
In [7]:
```
matrixA = np.array([[1,2,3],[4,5,6]
                    ])
print(matrixA)
```

```
[[1 2 3]
 [4 5 6]]
```

In [8]:
```
matrixB = np.random.randn(2,5)

print (matrixB)
```

```
[[-1.09395069  0.57571687  0.25944657  0.02205309  0.8279017 ]
 [-1.8557193  -1.53983971 -0.43262141  0.64270127  0.38064167]]
```

Lecture Bibliography



Lecture Bibliography (1)

- [1] Keras Python High-Level Deep Learning Library,
  Online: https://keras.io/
- [2] TensorFlow Python Low-Level Deep learning Library,
  Online: https://www.tensorflow.org/
- [3] NVIDIA Web Page,
  Online: https://www.nvidia.com/en-us/
- [4] Introduction to Data Mining, Pang-Ning Tan, Michael Steinbach, Vipin Kumar, Addison Wesley, ISBN 0321321367, English, ~769 pages, 2005
- [5] H. Lee et al., 'Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations', Proceedings of the 26th annual International Conference on Machine Learning (ICML), ACM, 2009
- [6] PRACE – Introduction to Supercomputing,
  Online: https://www.youtube.com/watch?v=D94FJx9vxFA
- [7] Introduction to High Performance Computing for Scientists and Engineers, Georg Hager & Gerhard Wellein, Chapman & Hall/CRC Computational Science, ISBN 143981192X, English, ~330 pages, 2010
- [8] TOP500 Supercomputing Sites,
  Online: http://www.top500.org/
- [9] Partnership for Advanced Computing in Europe (PRACE),
  Online: http://www.prace-ri.eu/
- [10] PRACE Training Portal,
  Online: http://www.training.prace-ri.eu/material/index.html

Lecture Bibliography (2)

- [11] YouTube Video, 'Big Ideas: Simplifying High Performance Computing Architectures', Online: https://www.youtube.com/watch?v=ISS_OGVamBk
- [12] Juelich Supercomputing Centre Web page, Online: http://www.fz-juelich.de/ias/jsc/EN/Home/home_node.html
- [13] Official Web Page for Python Programming Language, Online: https://www.python.org/
- [14] Numpy Python Library Web Page, Online: http://www.numpy.org/
- [15] Blog 'Vectorization and parallelization in Python with NumPy and Pandas', Online: https://datascience.blog.wzb.eu/2018/02/02/vectorization-and-parallelization-in-python-with-numpy-and-pandas/
- [16] François Chollet 'Deep Learning with Python', Book, ISBN 9781617294433, 384 pages, 2017, Online: https://www.manning.com/books/deep-learning-with-python
- [17] Jupyther Web page, Online: http://jupyter.org/
- [18] Anaconda Web page, Online: https://www.anaconda.com/
- [19] Tensorflow Deep Learning Framework, Online: https://www.tensorflow.org/
- [20] A Tour of Tensorflow, Online: https://arxiv.org/pdf/1610.01178.pdf

Lecture Bibliography (3)

- [21] K. Hwang, G. C. Fox, J. J. Dongarra, 'Distributed and Cloud Computing', Book, Online: http://store.elsevier.com/product.jsp?locale=en_EU&isbn=9780128002049
- [22] Big Data Tips, 'What is a Tensor?', Online: http://www.big-data.tips/what-is-a-tensor
- [23] A. Sergeev, M. Del Balso,'Horovod: fast and easy distributed deep learning in TensorFlow', 2018 Online: https://arxiv.org/abs/1802.05799
- [24] DEEP-EST EU Project, Online: http://www.deep-projects.eu/
- [25] Putty Tool, Online: http://www.putty.org/
- [26] LLView Tool, Online: http://www.fz-juelich.de/ias/jsc/EN/Expertise/Support/Software/LLview/_node.html
- [27] E. Erlingsson, G. Cavallaro, A. Galonska, M. Riedel, H. Neukirchen, 'Modular Supercomputing Design Supporting Machine Learning Applications', in conference proceedings of the 41st IEEE MIPRO 2018, May 21-25, 2018, Opatija, Croatia

Slides Available at http://www.morrisriedel.de/talks