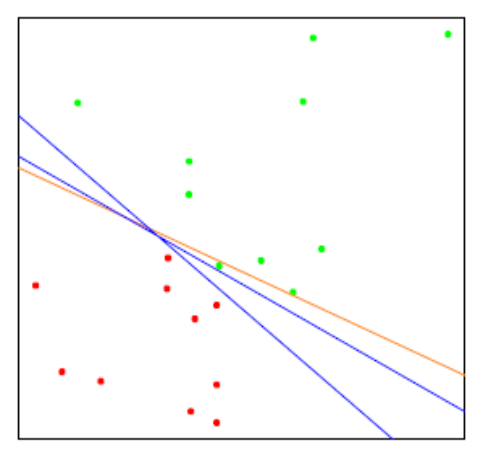


Parallel & Scalable Machine Learning

Introduction to Machine Learning Algorithms

Dr. Gabriele Cavallaro

Postdoctoral Researcher

High Productivity Data Processing Group

Juelich Supercomputing Centre

LECTURE 11

Unsupervised Clustering and Applications

February 27th, 2019

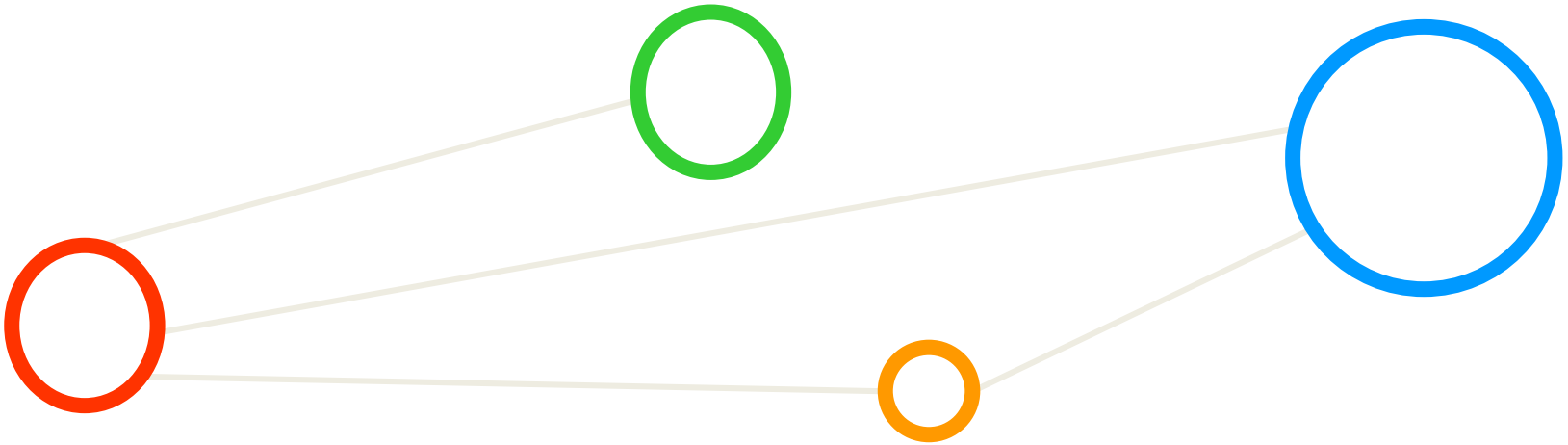
JSC, Germany

UNIVERSITY OF ICELAND

SCHOOL OF ENGINEERING AND NATURAL SCIENCES

FACULTY OF INDUSTRIAL ENGINEERING, MECHANICAL ENGINEERING AND COMPUTER SCIENCE

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

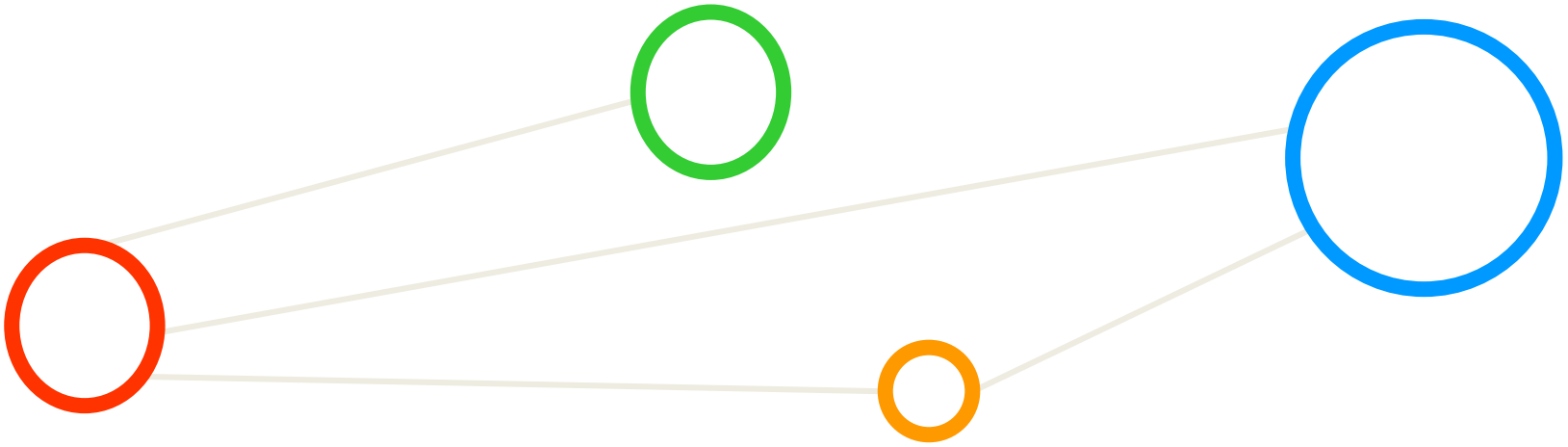
Theoretical Lectures

Practical Lectures

Outline

- Unsupervised Learning
  - Clustering Methods
  - K-Means Clustering Algorithm
  - DBSCAN Clustering Algorithm
- High Performance Computing
  - TOP500
  - Architectures of HPC Systems
  - Batch System
- Point Clouds
  - LIDAR - Operational Theory
  - Example of LIDAR Attributes
  - Point Cloud within Buildings

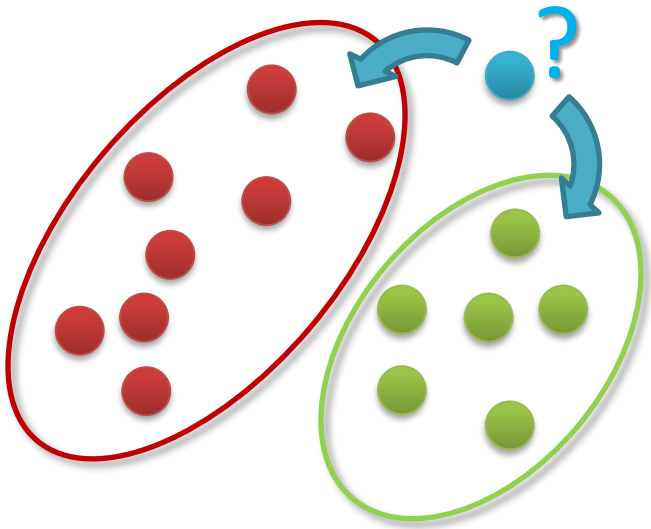
Unsupervised Clustering



Methods Overview

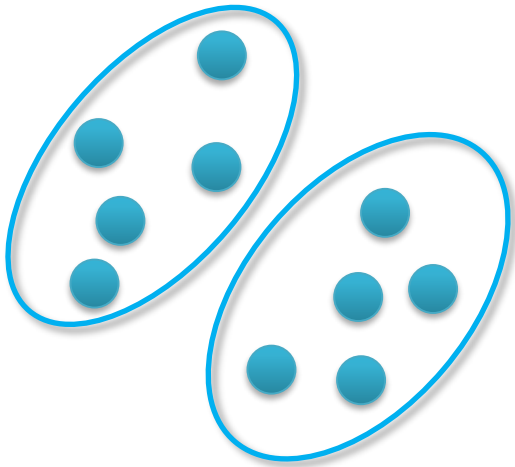
- **Statistical data mining methods can be roughly categorized in classification, clustering, or regression augmented with various techniques for data exploration, selection, or reduction**

Classification



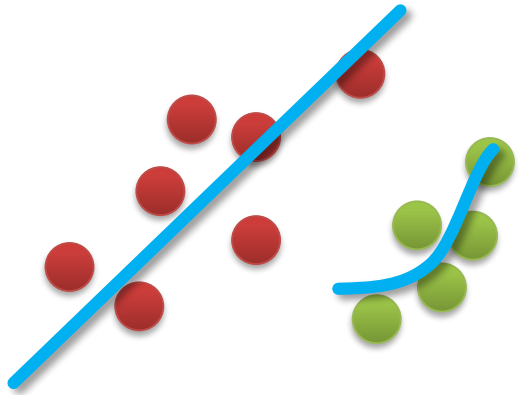
- Groups of data exist
- New data classified to existing groups

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

What means Learning – Revisited

- **The basic meaning of learning is ‘to use a set of observations to uncover an underlying process‘**
- **The three different learning approaches are supervised, unsupervised, and reinforcement learning**

- Supervised Learning
  - Majority of methods follow this approach in this course
  - Example: credit card approval based on previous customer applications
- Unsupervised Learning
  - Often applied before other learning → higher level data representation
  - Example: Coin recognition in vending machine based on weight and size
- Reinforcement Learning
  - Typical ‘human way‘ of learning
  - Example: Toddler tries to touch a hot cup of tea (again and again)

Learning Approaches – Unsupervised Learning – Revisited

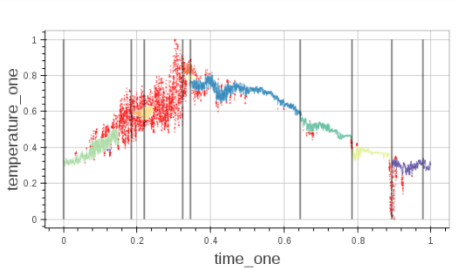
- Each observation of the predictor measurement(s) has no associated response measurement:
  - Input $\mathbf{x} = x_1, \ldots, x_d$
  - No output
  - Data $(\mathbf{x}_1), \ldots, (\mathbf{x}_N)$
- Goal: Seek to understand relationships between the observations
  - Clustering analysis: check whether the observations fall into distinct groups
- Challenges
  - No response/output that could supervise our data analysis
  - Clustering groups that overlap might be hardly recognized as distinct group

- **Unsupervised learning approaches seek to understand relationships between the observations**
- **Unsupervised learning approaches are used in clustering algorithms such as k-means, etc.**
- **Unupervised learning works with data = [input, ---]**

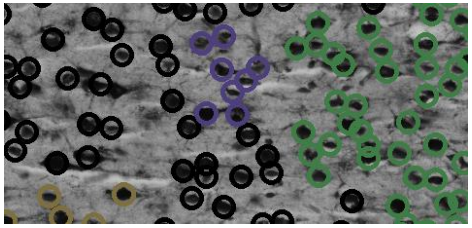
[1] An Introduction to Statistical Learning

Learning Approaches – Unsupervised Learning Use Cases

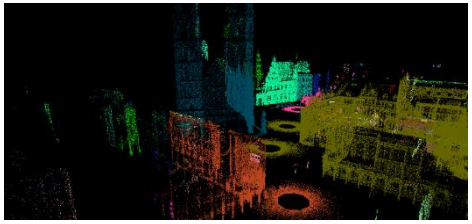
- Earth Science Data (PANGAEA, cf. Lecture 1)
  - Automatic quality control and event detection
  - Collaboration with the University of Gothenburg
  - Koljoefjords Sweden – Detect water mixing events



- Human Brain Data
  - Analyse human brain images as brain slices
  - Segment cell nuclei in brain slice images
  - Step in detecting layers of the cerebral cortex



- Point Cloud Data
  - Analysis of point cloud datasets of various sizes
  - 3D/4D LIDAR scans of territories (cities, ruins, etc.)
  - Filter noise and reconstruct objects



➢ **This clustering lecture uses a point cloud dataset of the city of Bremen as one concrete example**

Unsupervised Learning – Clustering Methods

- Characterization of clustering tasks
  - No prediction as there is no associated response Y to given inputs X
  - Discovering interesting facts & relationships about the inputs X
  - Partitioning of data in subgroups (i.e. ‘clusters‘) previously unknown
  - Being more subjective (and more challenging) than supervised learning

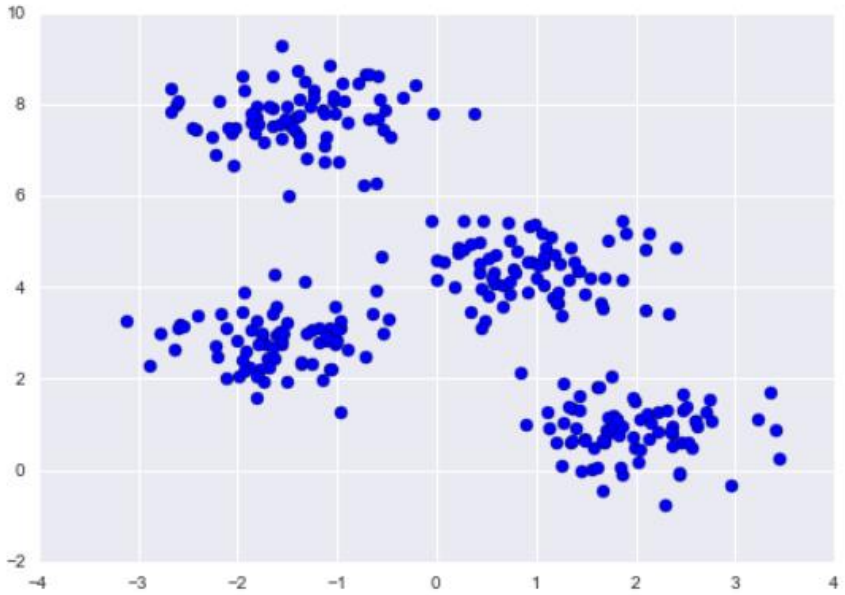
- Considered often as part of ‘exploratory data analysis‘
  - Assessing the results is hard, because no real validation mechanism exists
  - Simplifies data via a ‘small number of summaries’ good for interpretation

- **Clustering are a broad class of methods for discovering previously unknown subgroups in data**

Unsupervised Learning Example

- Two-dimensional dataset containing four distinct blobs

By eye, it is relatively easy to pick out the four clusters



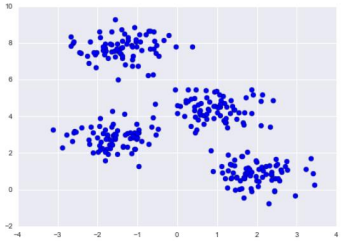
- How can one algorithm find automatically these clusters?

- Also, the number of possible combinations of cluster assignments is exponential in the number of data points
  - An exhaustive search can be very costly

How to Partition Data Into Distinct Groups?

- Data in same (homogenous) groups are somehow ‘similiar‘ to each other
- Data not in same sub-groups are somehow ‘different‘ from each other
- Concrete definitions of ‘similiarity‘ or ‘difference‘ often domain-specific
- Wide variety of similiarity measures exist, e.g. distance measures

  Jaccard Distance, Cosine Distance, Edit Distance, Hamming Distance, …



- **A distance measure in some space is a function distance(x,y) that takes two points in the space as arguments and produces a real number**

- E.g., Euclidean distance
  - n-dimensional Euclidean space: a space where points are vectors of n real numbers

$$distance([x_1, x_2, \dots, x_n], [y_1, y_2, \dots, y_n]) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

(ruler distance)

Clustering is an Optimization Problem

- For a single cluster c:

$$variability(c) = \sum_{e \in c} distance\ (mean(c), \boldsymbol{x})^2$$

$\boldsymbol{x} = data\ sample$

- For a group of clusters C :

$$similarity(C) = \sum_{c \in C} variability\ (c)$$

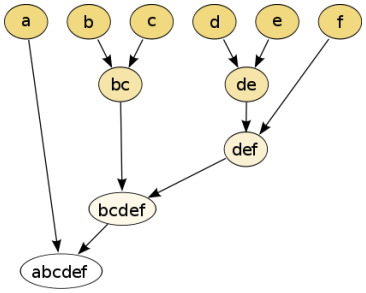
- What is the optimization problem that we want to solve by clustering?

- Is to find a C that minimizes similarity(C)?
  - No, otherwise we could put each example in its own cluster
  - The variability would be 0 (problem solved?)

- Need to apply constraints, e.g.,
  - Minimum distance between clusters
  - Number of clusters

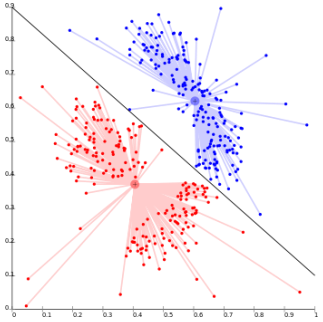
- **Optimization problem: minimize the similarity between the clusters by being constrained to a certain rule, e.g., not allowed to use less than N clusters**

Different Clustering Approaches



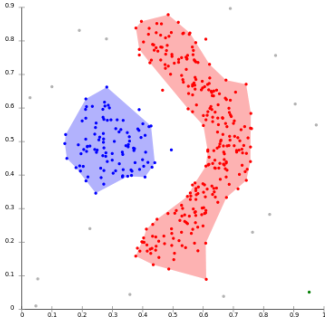
(hierarchical)

Data points closer in data space exhibit more similarity to each other than the data points lying farther away.



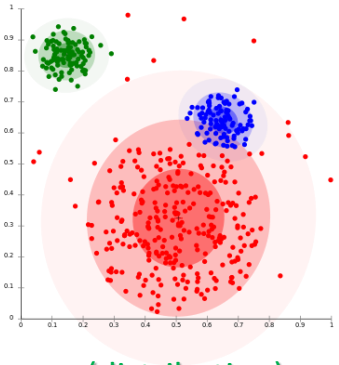
(centroid)

Similarity is derived by the closeness of a data point to the centroid of the clusters (Iterative algorithms).



(density)

Search the data space for areas of varied density of data points. Isolate different density regions and assign the data points within these regions in the same cluster.



(distribution)

Based on the notion of how probable is it that all data points in the cluster belong to the same distribution (e.g., Normal, Gaussian).

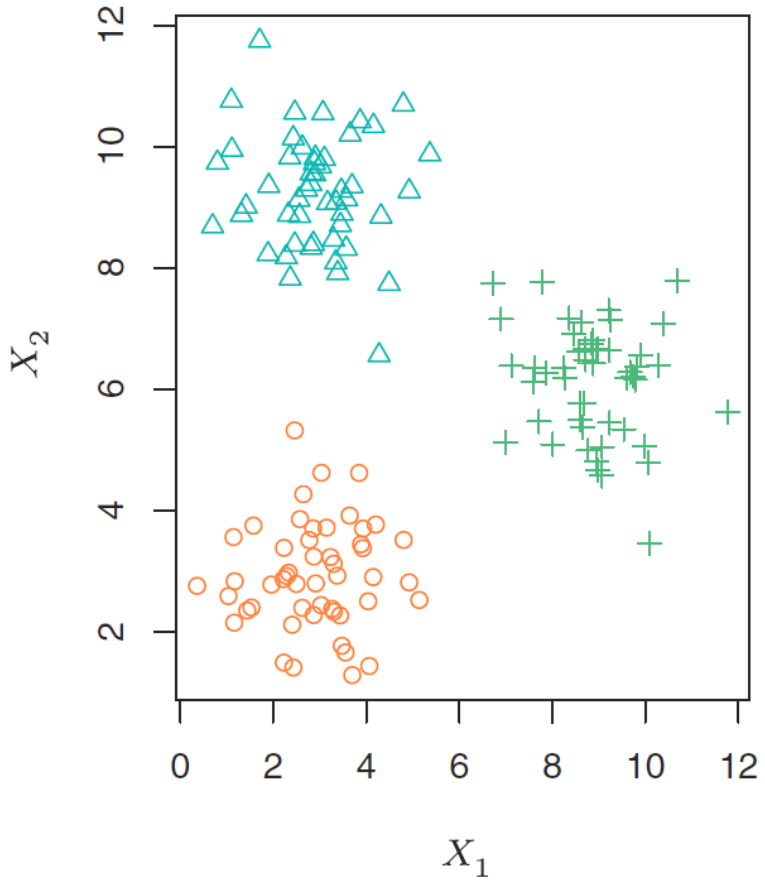
- **Clustering approaches can be categorized into four different approaches: (1) hierarchical, (2) centroid, (3) density, (4) distribution**

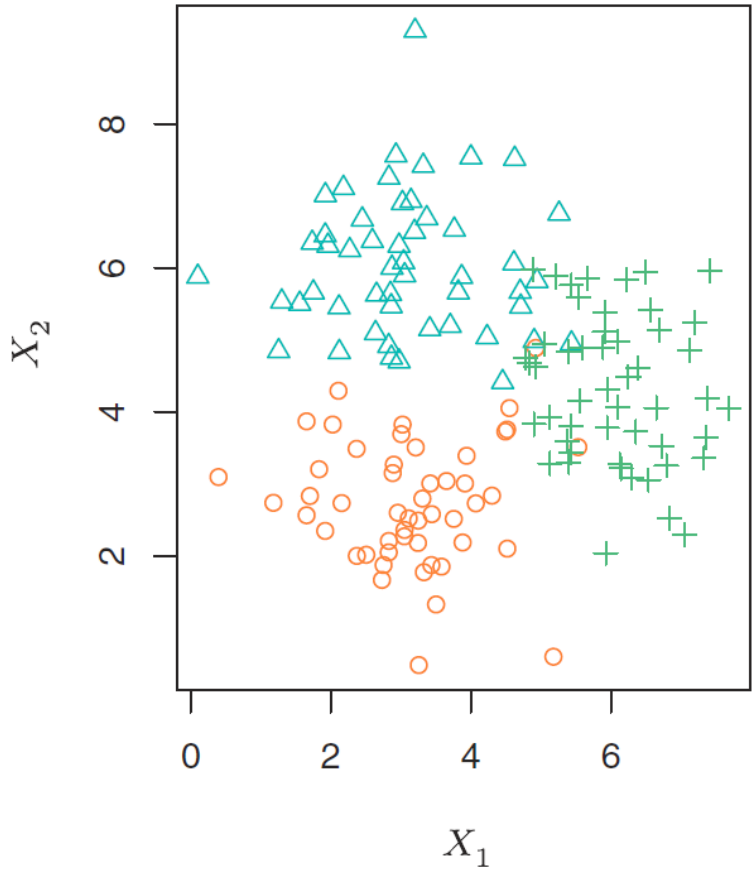
Selected Clustering Methods

- K-Means Clustering – Centroid based clustering
  - Partitions a data set into K distinct clusters (centroids can be artificial)
- K-Medoids Clustering – Centroid based clustering (variation)
  - Partitions a data set into K distinct clusters (centroids are actual points)
- Sequential Agglomerative hierarchic nonoverlapping (SAHN)
  - Hiearchical Clustering (create tree-like data structure → 'dendrogram')
- Clustering Using Representatives (CURE)
  - Select representative points / cluster – as far from one another as possible
- Density-based spatial clustering of applications + noise (DBSCAN)
  - Assumes clusters of similar density or areas of higher density in dataset

Learning Approaches – Unsupervised Learning Challenges

- Practice: The number of clusters can be ambiguities

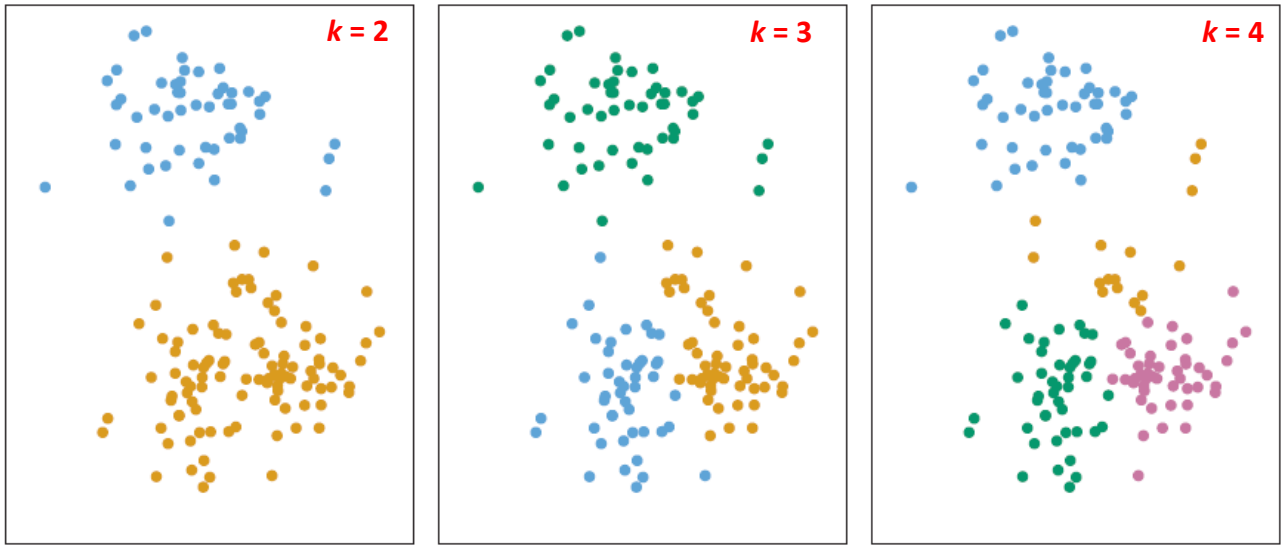




[1] An Introduction to Statistical Learning

Clustering Methods – K-Means Approach

- Approach Overview
  - Partitions a data set into *K* distinct (i.e. non-overlapping) clusters
  - Requires the definition of the desired number of clusters *K* in advance
  - Assigns each observation / data element to exactly one of the *K* clusters
  - Example: 150 observations; 2 dimensions; 3 different values of *K*



[1] An Introduction to Statistical Learning

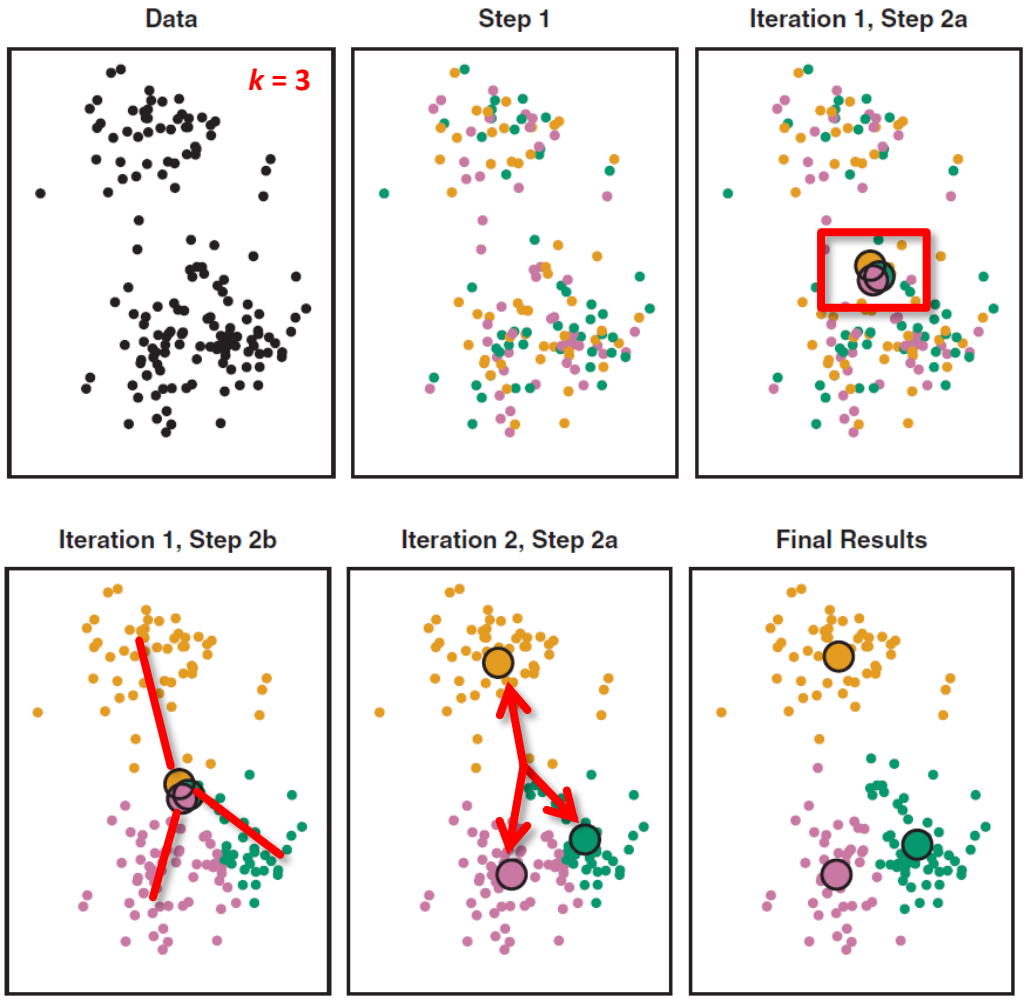
Clustering Methods – K-Means Algorithm

0. Set the desired number of clusters *K*
   - Picking the right number k is not simple (→ later)

1. Randomly assign a number from 1 to K to each observation
   - Initializes cluster assignments for the observations
   - Requires algorithm execution multiple times
     (results depend on random assignment, e.g. pick ‘best‘ after 6 runs)

2. Iterate until the cluster assignments stop changing
   a. For each of the K clusters: compute the cluster centroid
      - The kth cluster centroid is the vector of the p feature means
        for all the observations in the kth cluster
   b. Assign each observation to the cluster K whose centroid is closest
      - The definition of ‘closest‘ is the Euclidean distance

Clustering Methods – K-Means Algorithm Example

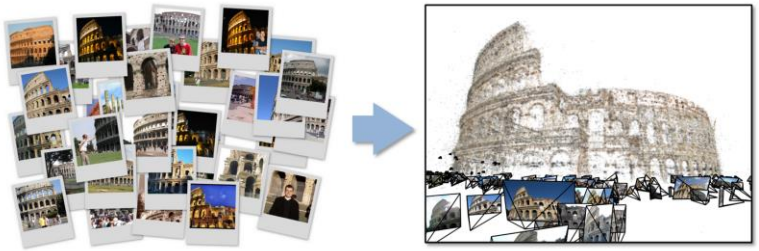


[1] An Introduction to Statistical Learning

1. Randomly assign a number from 1 to K to each observation
2. Iterate until the cluster assignments stop changing
   a. For each of the K clusters: compute the cluster centroid [centroids appear and move]
   b. Assign each observation to the cluster K whose centroid is closest [Euclidean distance]

Clustering Methods – K-Means Usage

- Advantages
  - Handles large datasets (larger than hierarchical cluster approaches)
  - Move of observations / data elements between clusters (often improves the overall solution)
- Disadvantages
  - Use of ‘means‘ implies that all variables must be continous
  - Severaly affected by datasets with outliers (→ means)
  - Perform poorly in cases with non-convex (e.g. U-shaped) clusters
- ‘Big Data‘ Application Example
  - Image processing: 7 million images
  - 512 features/attributes per image;
  - 1 million clusters
  - 10000 Map tasks; 64GB broadcasting;
  - 20 TB intermediate data in shuffling;



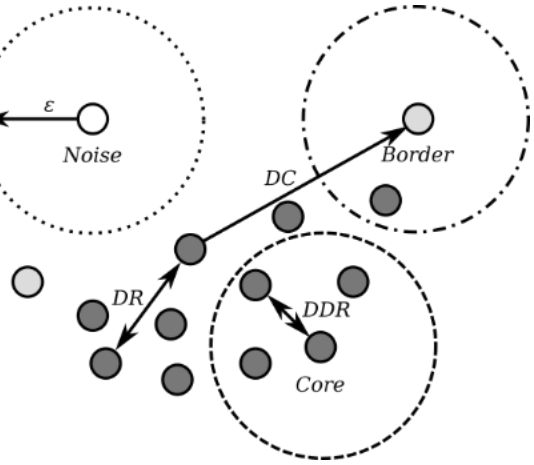
[2] Judy Qiu, ‘Collective communication on Hadoop’, 2014

Selected Clustering Methods

- K-Means Clustering – Centroid based clustering
  - Partitions a data set into K distinct clusters (centroids can be artificial)
- K-Medoids Clustering – Centroid based clustering (variation)
  - Partitions a data set into K distinct clusters (centroids are actual points)
- Sequential Agglomerative hierarchic nonoverlapping (SAHN)
  - Hiearchical Clustering (create tree-like data structure → 'dendrogram')
- Clustering Using Representatives (CURE)
  - Select representative points / cluster – as far from one another as possible
- Density-based spatial clustering of applications + noise (DBSCAN)
  - Assumes clusters of similar density or areas of higher density in dataset

DBSCAN Algorithm

- DBSCAN Algorithm *[3] Ester et al.*
  - Introduced 1996 and most cited clustering algorithm
  - Groups number of similar points into clusters of data
  - Similarity is defined by a distance measure (e.g. *euclidean distance*)



(MinPoints = 4)

- Distinct Algorithm Features
  - Clusters a variable number of clusters
  - Forms arbitrarily shaped clusters (except ‘bow ties‘)
  - Identifies inherently also outliers/noise

- Understanding Parameters
  - Looks for a similar points within a given search radius → Parameter *epsilon*
  - A cluster consist of a given minimum number of points → Parameter *minPoints*

(DR = Density Reachable)

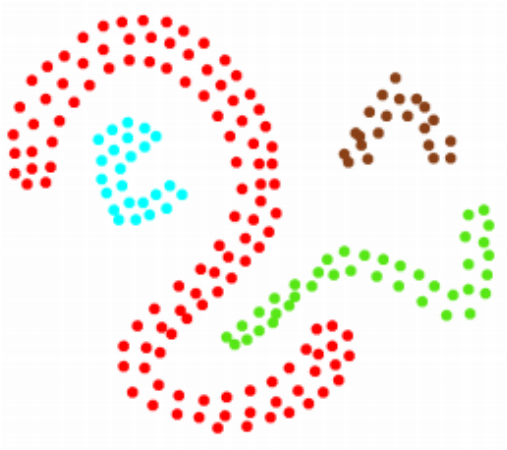
(DDR = Directly Density Reachable)

(DC = Density Connected)

DBSCAN Algorithm – Non-Trivial Example

- Compare K-Means vs. DBSCAN – How would K-Means work?

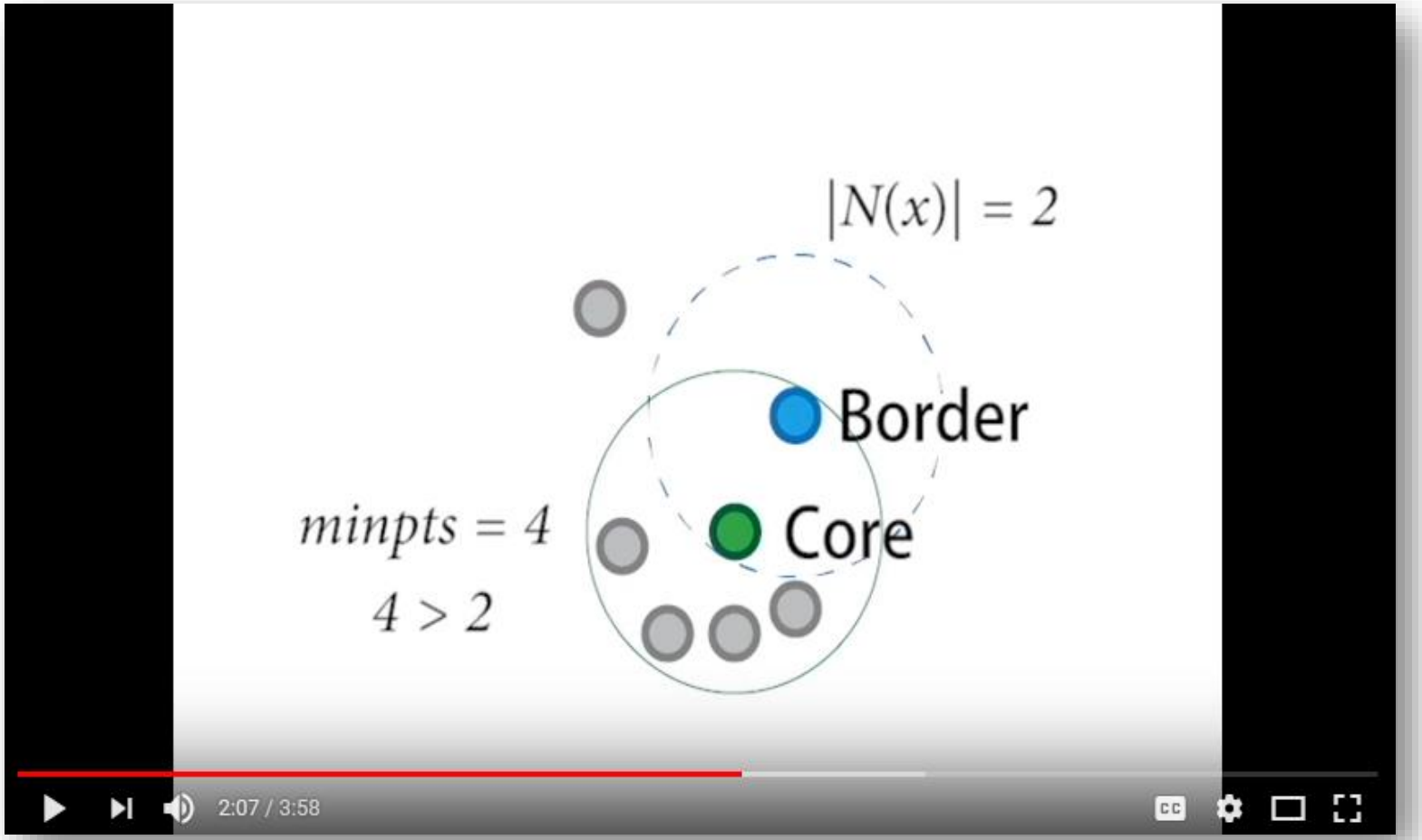
Unclustered Data



Clustered Data

- **DBSCAN forms arbitrarily shaped clusters (except 'bow ties') where other clustering algorithms fail**

[Video] DBSCAN Clustering

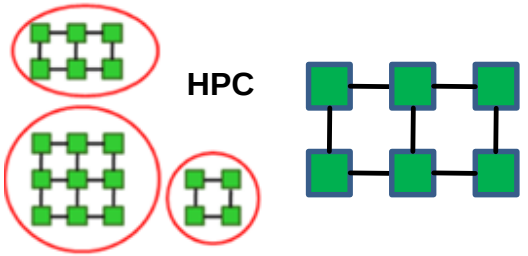


[4] DBSCAN, YouTube Video

JURECA HPC System at JSC

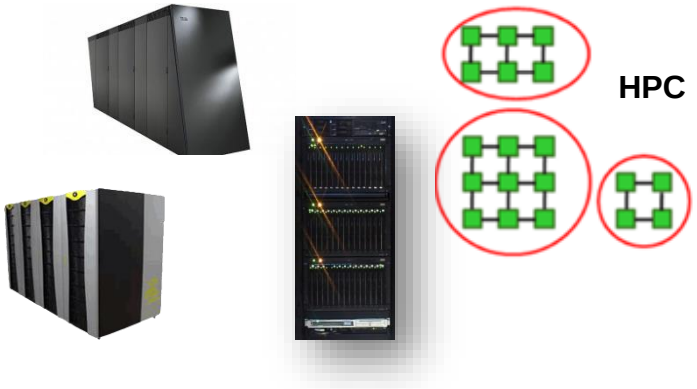
[5] JURECA HPC System

- **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

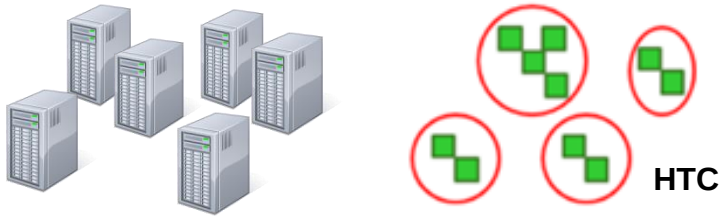


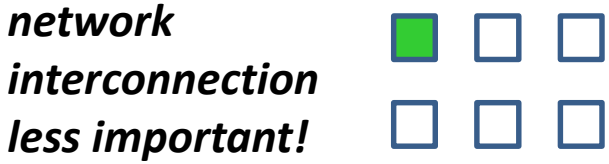
Understanding High Performance Computing

- **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





Parallel Computing

- All modern supercomputers heavily depend on parallelism

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

[6] Introduction to HPC for Scientists and Engineers

- ‘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

[7] TOP 500 supercomputing sites

TOP 500 List (Novemeber 2018)

- Based on the LINPACK benchmark
- **LINPACK** solves a dense system of linear equations of unspecified size

[8] LINPACK Benchmark implementation

  - It covers only a single architectural aspect ('critics exist')

26	Forschungszentrum Juelich (FZJ) Germany	**JUWELS Module 1** - Bull Sequana X1000, Xeon Platinum 8168 24C 2.7GHz, Mellanox EDR InfiniBand/ParTec ParaStation ClusterSuite Bull, Atos Group	114,480	6,177.7	9,891.1	1,361
44	Forschungszentrum Juelich (FZJ) Germany	**JURECA** - T-Platforms V-Class/Dell C6320P, E5-2680v3/Phi 7250-F, EDR/Intel Omni-Path/ParTec ParaStation, Tesla K80/K40 T-Platforms, Intel, Dell	155,150	3,782.6	6,563.8	1,345

- Alternatives realistic applications, benchmark suites and criteria exist

[9] HPC Challenge Benchmark Suite

[10] JUBE Benchmark Suite

[11] The GREEN500

Architectures of HPC Systems

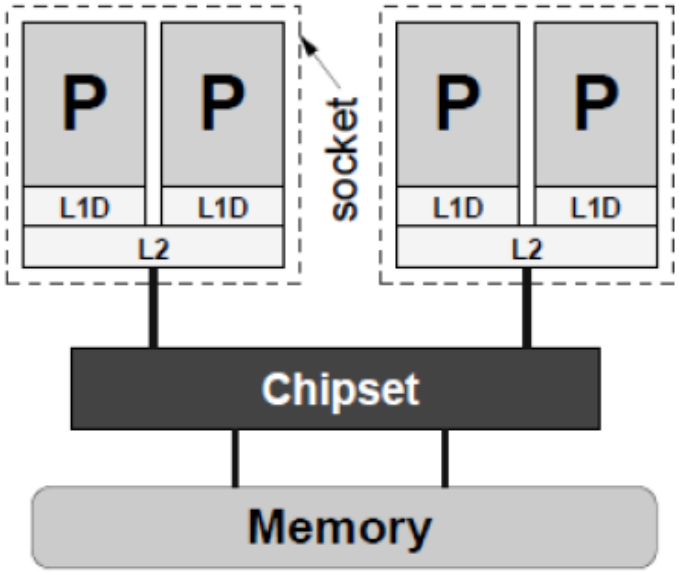
- Two dominant types of architectures
  - **Shared-Memory Computers**
  - **Distributed-Memory Computers**

- Often hierarchical (hybrid) systems of both in practice

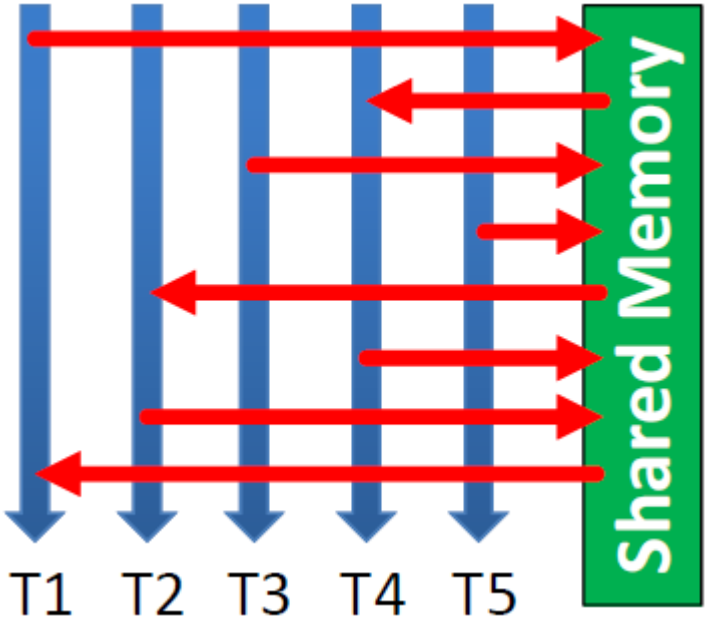
- More recently both above are considered as **'programming models'**
  - **Shared-Memory** parallelization with **OpenMP**
  - **Distributed-Memory** parallel programming with **MPI**

Shared-Memory Computers

- System where a number of CPUs work on a common, shared physical address space
- Programming using OpenMP (set of compiler directives to ‘mark parallel regions’)
- Enables immediate access to all data by all processors without explicit communication



[6] Introduction to HPC for Scientists and Engineers

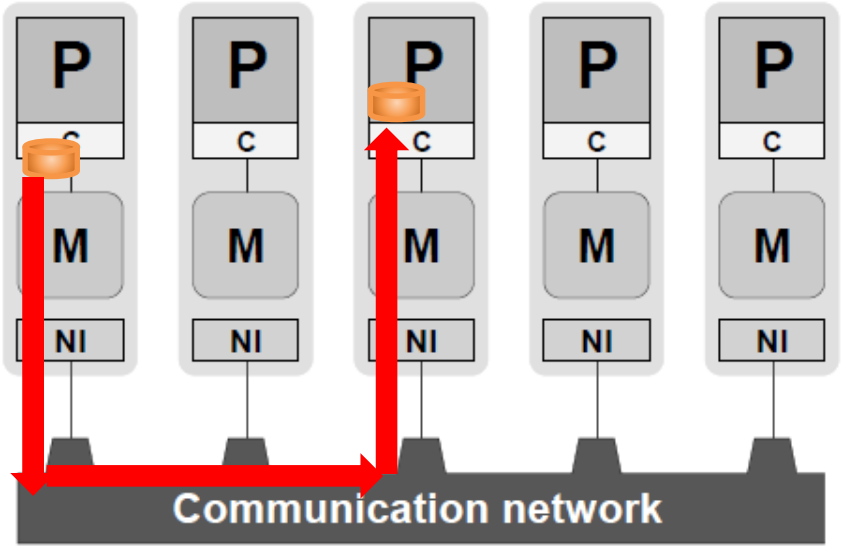


[12] OpenMP API Specification

Distributed-Memory Computers

- Establishes a ‘system view’ where no process can access another process’ memory directly

Programming Model: Message Passing

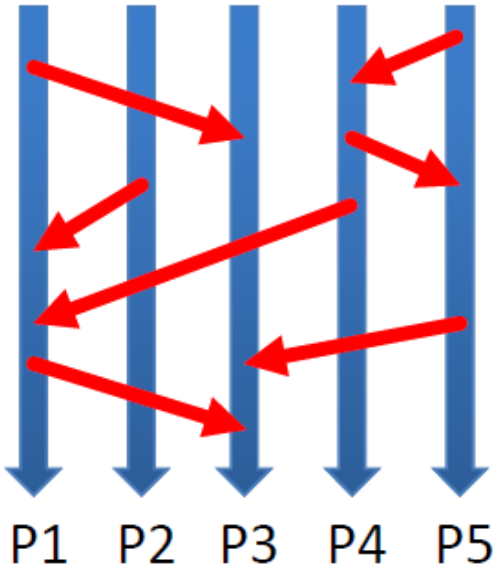


[6] Introduction to HPC for Scientists and Engineers

- Processors communicate via Network Interfaces (NI)
- NI mediates the connection to a Communication network
- This setup is rarely used -> a programming model view today

Programming with Distributed Memory using MPI

- Enables explicit message passing as communication between processors



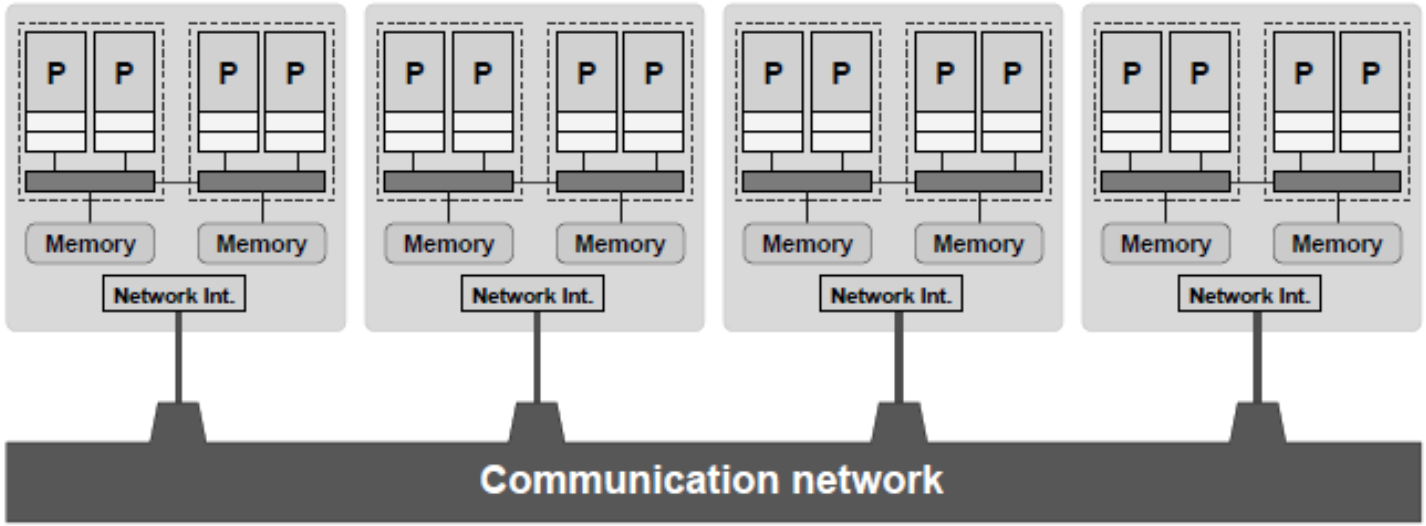
[9] MPI Standard

[6] Introduction to HPC for Scientists and Engineers

- No **remote memory access** on distributed-memory systems
- Require to '**send messages**' back and forth between processes PX
- Programming is tedious & complicated, but **most flexible method**

Hierarchical Hybrid Computers

- Mixture of shared-memory and distributed-memory systems
- Nowadays large-scale ‘hybrid’ parallel computers have shared-memory building blocks interconnected with a fast network (e.g., InfiniBand)



HPC Environment – What is a Batch System?

- Mechanism to control access by many users to shared computing resources
- Queuing / scheduling system for the jobs of the users
- Manages the reservation of resources and job execution
- Allows users to “fire and forget”
  - Long calculations or many jobs (“production runs”)

Why do we need a Batch System?

- Opposite of interactive processing
- Ensure all users get a fair share of compute resources
  - Demand usually exceeds supply
- To ensure the machine is utilized as efficiently as possible
- To track usage - for accounting and budget control
- To mediate access to other resources e.g. software licences

The only access to significant resources on the HPC machines is through the batch process

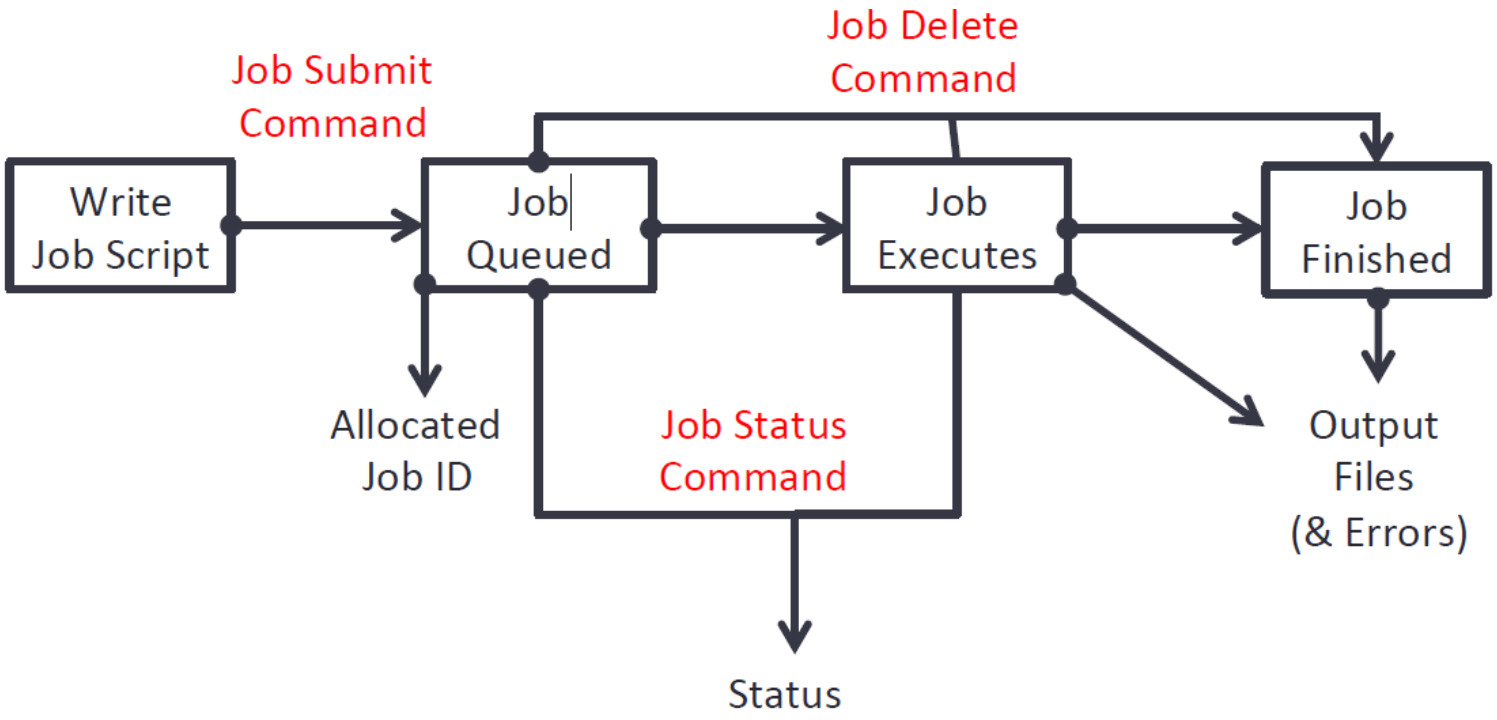
How to use a Batch System

1. Setup a job, consisting of:
   - Commands that run one or more calculations / simulations
   - Specification of compute resources needed to do this

2. Submit your job to the batch system
   - Job is placed in a queue by the scheduler
   - Will be executed when there is space and time on the machine
   - Job runs until it finishes successfully, is terminated due to errors, or exceeds a time limit

3. Examine outputs and any error messages

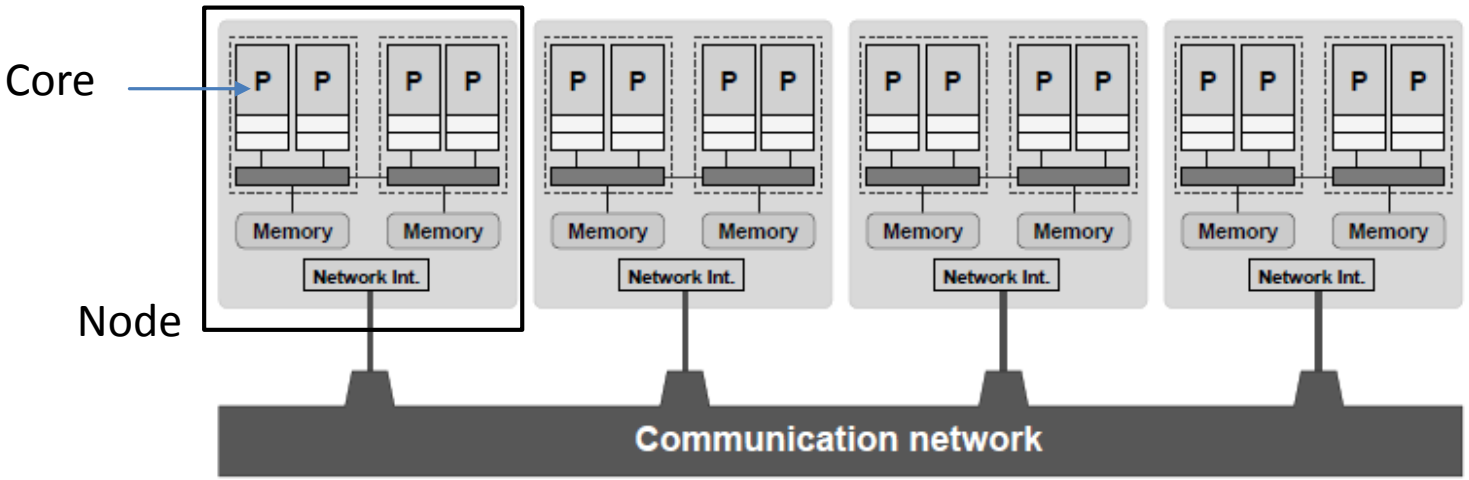
Batch System Flow



[14] Batch Systems

JSC Batch Model

- **Slurm is the chosen Batch System (Workload Manager) for JURECA**
  - Scheduling according to priorities: jobs with the highest priorities will be scheduled next
  - Backfilling scheduling algorithm: the scheduler checks the queue and may schedule jobs with lower priorities that can fit in the gap created by freeing resources for the next highest priority jobs
  - No node-sharing: the smallest allocation for jobs is one compute node. Running jobs do not disturb each other.
  - Accounted CPU-Quotas/job = Number-of-nodes x Walltime (x cores/node)



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 → needed before HPDBSCAN run**
  - Loads particular modules into the current work environment, E.g.:

```
[cavallaro1@jrl06 ~]$ module load  GCC/8.2.0  ParaStationMPI/5.2.1-1

Due to MODULEPATH changes, the following have been reloaded:
  1) binutils/.2.31.1

The following have been reloaded with a version change:
  1) GCCcore/.7.3.0 => GCCcore/.8.2.0

[cavallaro1@jrl06 ~]$ module load HDF5/1.10.1
```

Slurm – User Commands (1)

- **salloc** to request interactive jobs/allocations
- **sattach** to attach standard input, output, and error plus signal capabilities to a currently running job or job step
- **sbatch** to submit a batch script (which can be a bash, Perl or Python script)
- **scancel** to cancel a pending or running job or job step
- **sbcast** to transfer a file to all nodes allocated for a job
- **sgather** to transfer a file from all allocated nodes to the currently active job.This command can be used only inside a job script
- **scontrol** provides also some functionality for the users to manage jobs or query and get some information about the system configuration
- **sinfo** to retrieve information about the partitions, reservations and node states
- **smap** graphically shows the state of the partitions and nodes using a curses interface. We recommend llview as an alternative which is supported on all JSC machines

[15] SLURM Workload Manager

Slurm – User Commands (2)

- **sprio** can be used to query job priorities
- **squeue** to query the list of pending and running jobs
- **srun** to initiate job-steps mainly within a job or start an interactive jobs. A job can contain multiple job steps executing sequentially or in parallel on independent or shared nodes within the job's node allocation
- **sshare** to retrieve fair-share information for each user
- **sstat** to query status information about a running job
- **sview** is a graphical user interface to get state information for jobs, partitions, and nodes
- **sacct** to retrieve accounting information about jobs and job steps in Slurm's database
- **sacctmgr** allows also the users to query some information about their accounts and other accounting information in Slurm's database.

** For more detailed info please check the online documentation and the man pages*

[15] SLURM Workload Manager

Slurm – Job Submission

- There are 2 commands for job allocation: **sbatch** is used for batch jobs and **salloc** is used to allocate resource for interactive jobs. The format of these commands:
  - sbatch [options] jobscript [args...]
  - salloc [options] [<command> [command args]]
- List of the most important submission/allocation options:

```
-A|--account       Charge CPU-Quota to specified account (budget ID).
-c|--cpus-per-task Number of logical CPUs (hardware threads) per task.
-e|--error         Path to the job's standard error.
-i|--input         Connect the jobscript’s standard input directly to a file.
-J|--job-name      Set the name of the job.
--mail-user        Define the mail address for notifications.
--mail-type        When to send mail notifications. Options: BEGIN,END,FAIL,ALL
-N|--nodes         Number of compute nodes used by the job.
-n|--ntasks        Number of tasks (MPI processes).
--ntasks-per-node  Number of tasks per compute node.
-o|--output        Path to the job's standard output.
-p|--partition     Partition to be used from the job.
-t|--time          Maximum wall-clock time of the job.
--gres             Request nodes with specific Generic Resources.
```

[15] SLURM Workload Manager

What is LIDAR?

- **LIght Detection And Ranging**
  - Active Sensing System
  - Day or Night operation.
  - Ranging of the reflecting object based on time difference between emission and reflection.

- **What's NOT LiDAR?**
  - NOT Light/Laser Assisted RADAR
    - RADAR uses electro-magnetic (EM) energy in the radio frequency range;
    - LIDAR does not.
  - NOT all-weather
    - The target MUST be visible. Some haze is manageable, but fog is not
  - NOT able to `see through' trees
    - LIDAR sees around trees, not through them. Fully closed canopies (rain forests) cannot be penetrated
  - NOT a Substitute for Photography
    - For MOST users, LIDAR intensity images are NOT viable replacements for conventional or digital imagery

[16] G. Camps-Valls

Credits: Jiunn-Der (Geoffrey) Duh, Portland, USA

What is LIDAR?

- **LIDAR Characteristics**
  - Vertical accuracy for commercial applications at 15 cm on discrete points
  - Collects millions of elevation points per hour
  - Produces datasets with much greater density than traditional mapping
  - Some systems capable of capturing multiple returns per pulse and/or intensity images

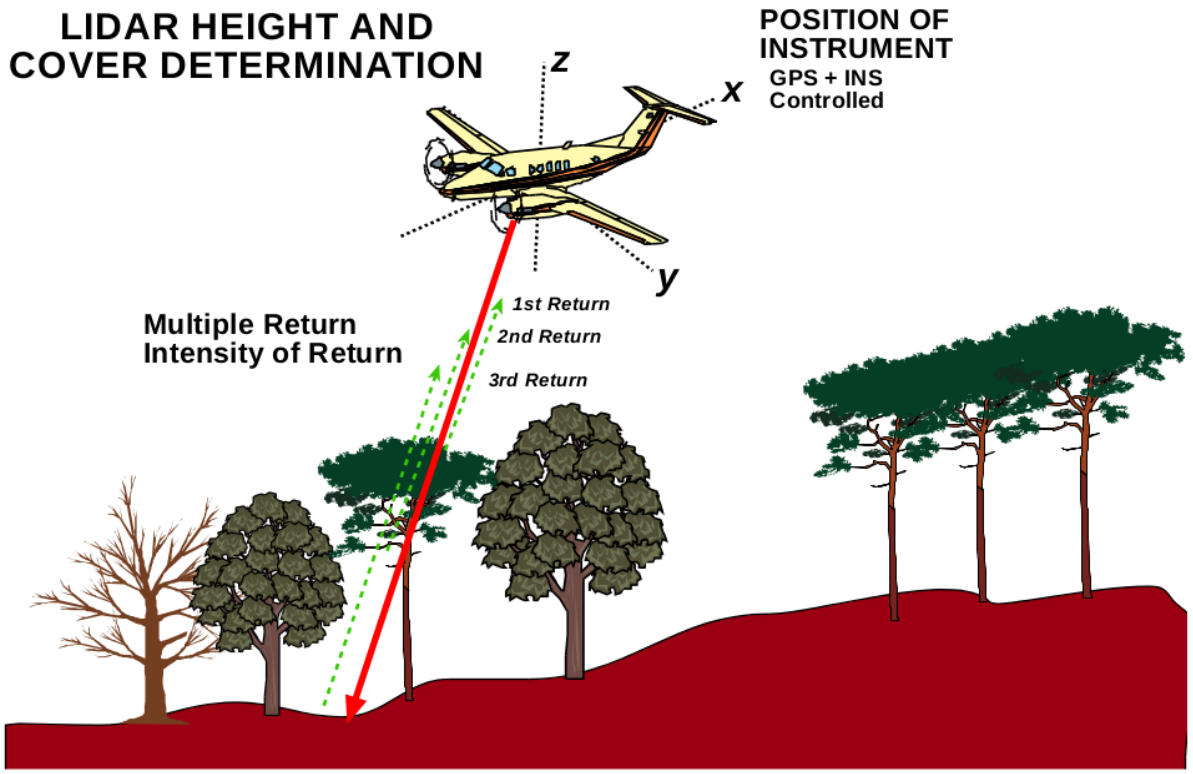
- **LiDAR Operational Theory**
  - A pulse of light is emitted and the precise time is recorded
  - The reflection of that pulse is detected and the precise time is recorded
  - Using the constant speed of light, the delay can be converted into a “slant range” distance.
  - Knowing the position and orientation of the sensor, the XYZ coordinate of the reflective surface can be calculated

[16] G. Camps-Valls

Credits: Jiunn-Der (Geoffrey) Duh, Portland, USA

Multiple-Returns vs. Single-Return Systems

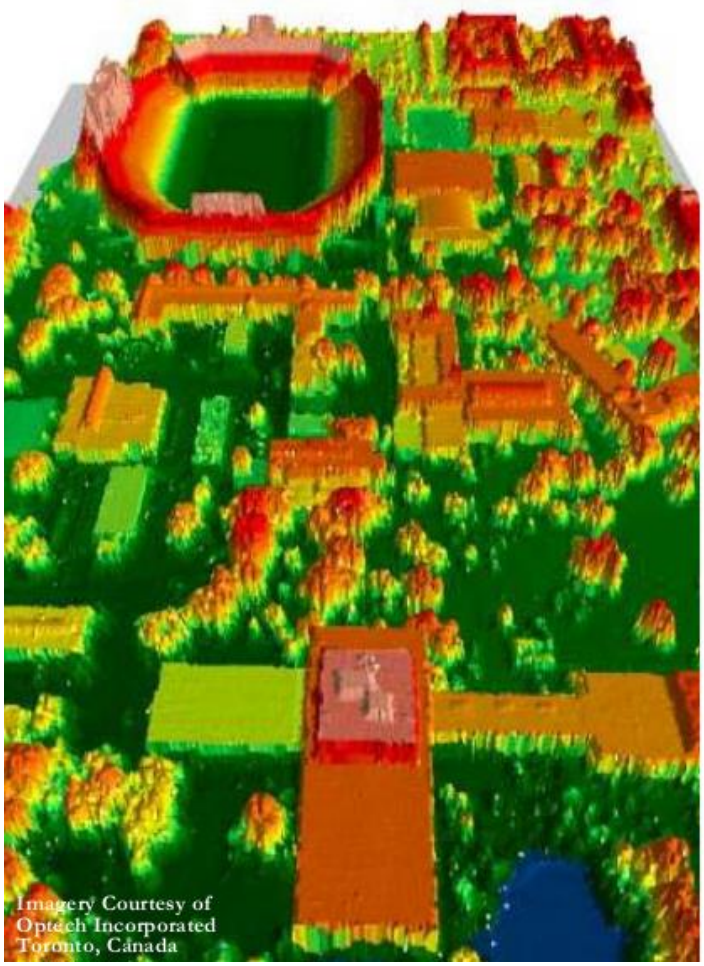
- Single-Return systems: returns come from the canopy top
- Multiple-Return systems: first returns also from the canopy top, but successive returns will come from lower surfaces, such as vegetation and the ground



[16] G. Camps-Valls

Credits: Jiunn-Der (Geoffrey) Duh, Portland, USA

LiDAR Return Intensity

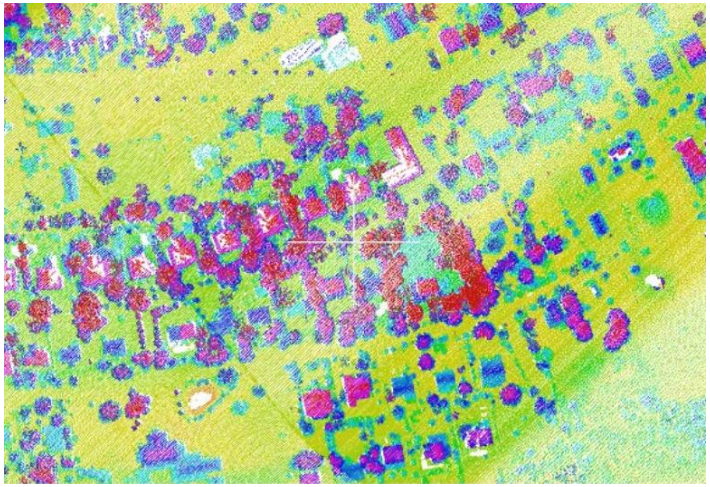


[16] G. Camps-Valls

Credits: Jiunn-Der (Geoffrey) Duh, Portland, USA

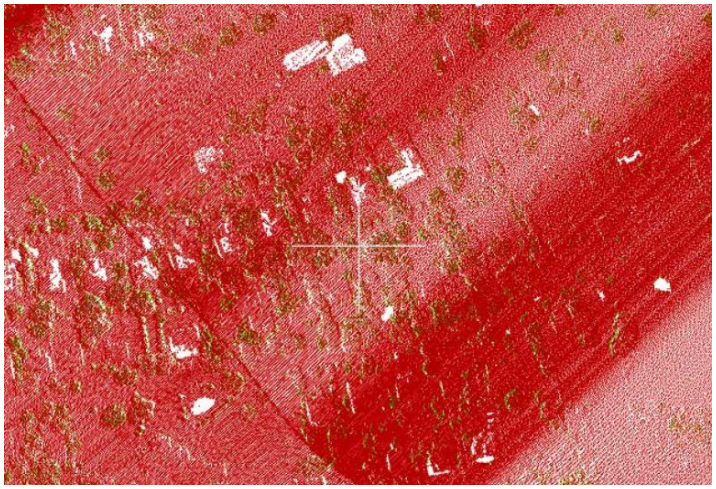
Example of Attributes

Elevation

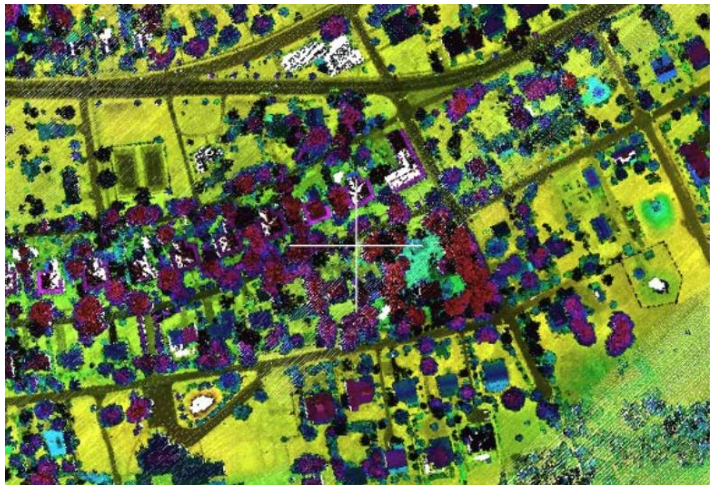


Intensity

Return number



Intensity+Elevation



LiDAR points colored to represent different attributes of the data

[16] G. Camps-Valls

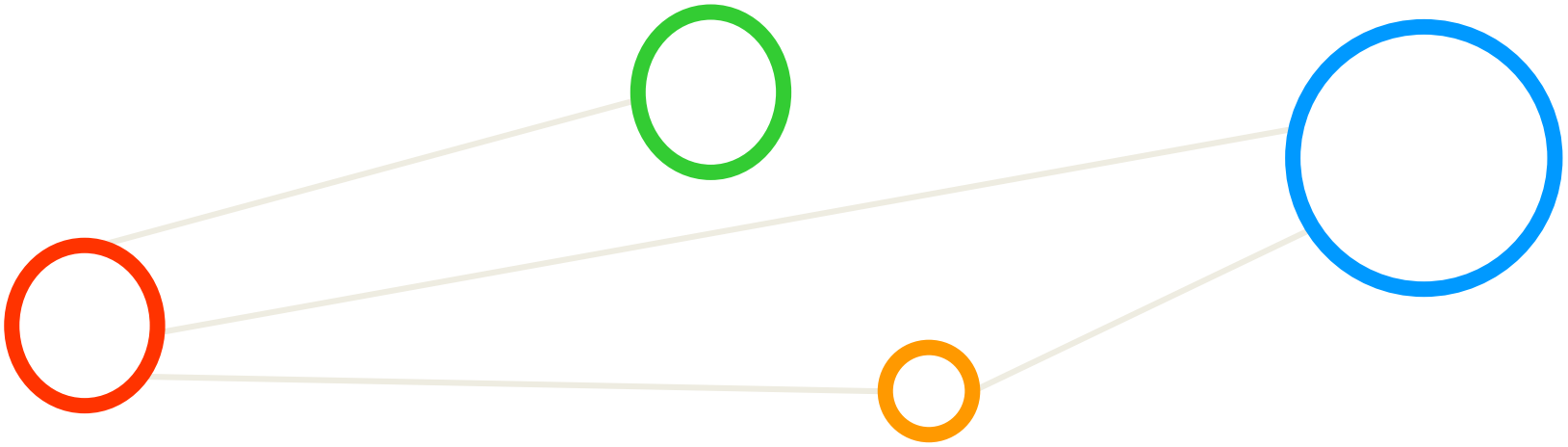
Credits: Jiunn-Der (Geoffrey) Duh, Portland, USA

Point Cloud - Within Buildings

- Point based rendering example
  - Aachen Cathedral based on 3D laser scans and photos
  - Points are rendered as textured and blended splats
  - Visualisation can run in real-time on a desktop PC showing 6 million splats based of a 120 million point laser scan

[17] Aachen Cathedral Point Cloud Rendering, YouTube Video

Lecture Bibliography



Lecture Bibliography (1)

- [1] An Introduction to Statistical Learning with Applications in R, Online: http://www-bcf.usc.edu/~gareth/ISL/index.html
- [2] Judy Qiu,'Harp: Collective Communication on Hadoop', 2014
- [3] Ester, Martin, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." Kdd. Vol. 96. 1996.
- [4] YouTube Video, 'CSCE 420 Communication Project – DBSCAN', Online: https://www.youtube.com/watch?v=5E097ZLE9Sg
- [5] JURECA HPC System at JSC Online: http://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/JURECA/JURECA_node.html
- [6] Introduction to High Performance Computing for Scientists and Engineers, Georg Hager & Gerhard Wellein, Chapman & Hall/CRC Computational Science, ISBN 143981192X
- [7] TOP500 Supercomputing Sites Online: http://www.top500.org/
- [8] LINPACK Benchmark Online: http://www.netlib.org/benchmark/hpl/
- [9] HPC Challenge Benchmark Suite Online: http://icl.cs.utk.edu/hpcc/
- [10] JUBE Benchmark Suite, Online: http://www.fz-juelich.de/ias/jsc/EN/Expertise/Support/Software/JUBE/_node.html
- [11] The GREEN500 Online: https://www.top500.org/green500/

Lecture Bibliography (2)

- [12] The OpenMP API specification for parallel programming
  Online: http://openmp.org/wp/openmp-specifications /
- [13] The MPI Standard
  Online: http://www.mpi-forum.org/docs/
- [14] Batch Systems – archer Running your jobs on an HPC machine
  Online: https://www.archer.ac.uk/training/course-material/2017/07/intro-epcc/slides/L10_Batch_Execution.pdf
- [15] SLURM Workload Manager
  Online, https://slurm.schedmd.com/
- [16] G. Camps-Valls, “Hyperspectral and Lidar“ in the 5th ESA Advanced Training Course on Land Remote Sensing
  Online: http://seom.esa.int/landtraining2014/files/D4T2b_CampsValls.pdf
- [17] YouTube Video ‚Point Based Rendering of the Aachen Cathedral‘
  Online: https://www.youtube.com/watch?v=X_wyoroo4co

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