COVENTRY UNIVERSITY

5011CEM BIG DATA PROGRAMMING PROJECT REPORT

SPEED COMPARISON OF DATA DENSITY BASED CLUSTERING (DDC) ALGORITHM USING MATLAB EXTERNAL LANGUAGE INTERFACES

April,2020.

Table of Contents

Table of Figures

1. Introduction

The four V's of Big Data- Volume, Variety, Veracity, and, Velocity. Keeping in mind these principles, clustering algorithms are considering to be adept for extracting data from large and complex datasets. Consider, velocity, i.e. the speed of data analysis. This project uses a clustering algorithm to compare speed of data analysis, on a specific dataset, in various languages. The project addresses the need for fast and accurate execution of data analysis.

The Data Density based Clustering algorithm is executed in various languages on the dataset of ozone emissions in Europe, as measured by the European Centre for Medium Range-Weather Forecasts (ECMWF). The project makes use of Matlab External Language Interfaces, to compare the DDC mathematical formula of data analysis in Matlab, Python, C++, and Java. Data Density based Clustering algorithm was opted for its computational abilities, and identification of clusters on the grid-based data which is provided. The Data Density based Clustering algorithm inputs in a 4-D array of scaled data (dimensions mxn), with m rows of sample data of ozone emissions given by grid spaces of 0.1-degree latitude and longitude. It outputs a structure of cluster centre co-ordinates and radii, and an array (49 million x 2 dimensions) of data points with the corresponding cluster number.

2. Specification

<u>SMART Analysis</u>:

- Specific- Do speed comparison of DDC algorithm,
- Measurable- by measuring execution time in four languages (Matlab, Python, C++, Java) ,
- Achievable- spending about sixty hours,
- Relevant/Reliable- and working six hours a week,
- Time-bound- to finish by the deadline April 27th,2020.

<u>Risk and Assumptions</u>:

Hardware Specifications: The project ran on a system with an Intel® Core™ 17-7500U CPU @ 2.70Ghz 2.90GHz with an 8.00 GB RAM and a 128GB capacity Solid State Drive (SSD) and 1TB Hard Drive. It is assumed that a system with the same of higher hardware specifications will be adequate.

Software Specifications: The operating system considered here is Windows 10. The assumption that programs would run on different OS will pose a risk due to discrepancies in library dependencies and compiling and execution of programs. The program call is only done synchronously in each application, asynchronous connections to Matlab have not been explored. Programs were evaluated with no background processes running but system interrupts and hidden background processes are not taken into consideration.

Algorithm: The DDC algorithm used, was provided at the start of the module and any inconsistencies have not been considered. The speed comparison is done on the data without a merging clusters and visualizing the output. The measure of execution is not evaluated for the above and any results found are assumed for a DDC function without merging , a single radius and no visualization.

User Knowledge: It is assumed that the user has prolific knowledge of Matlab, Python, C++, and Java syntax. The user should also have the ability to understand documentation to implement the various Matlab External Language Interface API's.

Data analysed and size: The data analysed is provided in a mat data file and a different format would need change of importing data in Matlab. The Matlab External Language Interface API's are limited by a 2GB array size and recursive functions cannot be called between applications.

Related Work:

Table 1: Related Works

Name (with link to the document)
Hyde, R. and Angelov, P. (2014) *Data Density Based Clustering*.   http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6930157&isnumber=6930143
Reddy, K.S.S. and Bindu, C.S. (2017) *A Review on Density-Based Clustering Algorithms for Big Data Analysis*.   https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8058322
Bendechache, M., Kechadi, M.-T., and Le-Khac, N.-A. (2016) *Efficient Large-Scale Clustering Based on Data Partitioning*. 612-621   https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7796948
Mai, S.T., He, X., Feng, J., Plant, C., and Böhm, C. (2014) *Anytime Density-Based Clustering of Complex Data*.   https://dl.acm.org/doi/10.1007/s10115-014-0797-0
Mai, S.T., Assent, I., and Storgaard, M. (2016) *AnyDBC: An Efficient Anytime Density-Based Clustering Algorithm for Very Large Complex Datasets*.   https://dl.acm.org/doi/10.1145/2939672.2939750
Hong, L. and Cai, J. (2010) *The Application Guide of Mixed Programming between MATLAB and Other Programming Languages*.   https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5452058
Brandt, S. (1998) *Data Analysis*. New York [u. a.]: Springer   http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.177.5019&rep=rep1&type=pdf

Out of Scope:

- OS- Windows. Linux and Mac OS not in scope of project.
- DDC algorithm is called using Matlab External Language Interface API's. The rewriting of the mathematical formula in each language is not in scope.
- Asynchronous connection to Matlab is not explored.
- Matlab External Language Interface API's limits the size of data arrays passed to 2GB.
- Recursive data structures cannot be passed to Python. Large output arrays cannot be handled by Python.
- Mex files for C++ is not considered.
- Visualization of clusters is not considered.
- Merging of cluster is not applicable.
- Connection to Matlab on remote machines.
- Connection to more than one shared Matlab session is also not in scope.
- Although programs were executed without other background processes running, there might still be discrepancies in speed due to system interrupts or hidden processes as execution in an isolated environment to remove interferences is not in scope.

System/Solution Overview:

Matlab- Version R2019b. Import data provided in mat file. Run and time DDC algorithm.

Python- Version 3.7. Connection to Matlab shared session with Matlab python engine. Run and time DDC algorithm.

C++- Visual Studio 2019. Connection to Matlab shared session with in-built C++ engine in Matlab. Run and time DDC algorithm.

Java- NetBeans IDE 8.2., JDK 8. Connection to Matlab shared session with java engine. Run and time DDC algorithm.

All applications execute and demonstrate cluster and its results on their output screen (python uses nargout=0 to manage the output). The best speed of execution is taken out of numerous simulations.

Project Management:

Project Management is implemented with weekly sprints documented in a logbook. A customized Kanban board is used to keep track of task breakdown. Burn-down chart is updated after each weekly sprint to manage delivery of project. Continuous Professional Development (CPD) is tracked with the help of logbook. Any changes made for weekly sprint are denoted by file versions numbered by the sprint running currently. Testing is manged with sample data and a uniform test file as mentioned in the Portfolio. For further details, refer to Appendix B and Appendix C.

- Week 1- Introduction to Project.
- Week 2- Matlab Course/ Specification
- Week 3- Matlab Course/ Code Planning
- Week 4- Data files manipulation.
- Week 5- DDC Algorithm Matlab
- Week 6- DDC Algorithm Python
- Week 7- DDC Algorithm C++
- Week 8- DDC Algorithm Python and C++
- Week 9- DDC Algorithm Java
- Week 10- Report/ Simulations
- Week 11- Portfolio/ C++ Simulation
- Week 12- Viva/ Final simulation

Open Issues:

- Issue: C++ Executable takes about two hours to reach second cluster in results.
- Raised on: Week 10 of project timeline.
- Decision: Problem due to system hardware, Visual Studio dependency, and low processor speed.
- Resolved By: Next Deployment.
- Not yet resolved.
- Status: Executable runs and outputs result but current system is not viable for computing execution time.

For further details, refer to Appendix A.

2.1 Limitations

- Operating System Limitations: project is configured for Windows only.
- Visualization, and merging of cluster is not considered in the Data Density based Clustering algorithm.
- A single radius of 0.1257 is used.
- Matlab can only be connected synchronously, asynchronous connection was not explored.
- Engine cannot be connected to remote machine.
- Data array size passed between Matlab and python is limited to 2GB.
- Although programs were executed without other background processes running, there might still be discrepancies in speed due to system interrupts or hidden processes as execution in an isolated environment to remove interferences is not in scope.

3. Code

The given O3Scaled.mat data is imported to Matlab. The Data Density based Clustering algorithm is executed in a user script and timed. Matlab engine is denoted as shared engine with a specific name. Matlab engine is connected to Python, C++, and Java using the respective external language interfaces. Each application is executed and time elapsed is computed. All four applications output their runtime, cluster structure containing centre co-ordinates and radii value, and result array of data points and their corresponding cluster number. The best runtime of each application is compared to find the application with high speed performance. Refer to Appendix A(5.1, Flow Diagram).

Outstanding issues:

C++ executable reaches second cluster in results after about two hours of execution (executes at very low processor speed). Executable runs and outputs result but current system properties are not ideal for computing execution time. Refer to Appendix A (7. Open Issues).

4. Results and Conclusion

The runtime was observed over numerous simulations for each application. Matlab had runtimes of 1251.120, 697.731, 394.597, 516.725, 525.464 seconds. Python had runtimes of 5264.541, 488.643, 363.362 seconds. Java had runtimes of 3448.343, 3620.881, 2784.379 seconds. C++ takes too long to execute, as mentioned in the outstanding issues. Outputs are in the respective application folder in the code zip file. As stated above, Python and Matlab have the highest speed performance. Negligible discrepancies exist as system interrupts and hidden process could affect time elapsed, as we are not running on an isolated system environment.

C++ code could have been converted from Matlab using Matlab coder but that would raise disparities between the approaches implemented in different applications. The outputs are executed in each applications output screen, but they could have been collated and either visualized or tabularised.

5. Future Work

- Compare speed in C using Matlab External Language Interface engine for C.
- Using mex structures to run and compute C++ code.
- Compare speed including verbose mode (visualization) and merging of clusters.
- Compare speed in asynchronous mode.
- Rewrite the Data Density based Clustering Algorithm in each language and compare. This would remove the limitation of data size that can be passed between applications.
- Including code instructions for Linux and Mac operating systems.
- Possible speed comparison between different operating systems with the application which resulted in high-speed performance.

6. Summary

After comparison of all runtimes, it is found that Matlab has best runtime of 394.597 seconds. Python has best runtime of 363.362 seconds. Java has best runtime of 2714.379 seconds. C++ takes too long to run in the current system as previously stated.

SMART Target Fulfilment:

- Specific: Speed comparison of Data Density based Clustering algorithm is achieved.
- Measurable: Execution time is measured precisely in three languages (Matlab, Python, Java) and vaguely in one language (C++).
- Achievable: Time spent over the span of the weeks is almost sixty hours as noted by burn-down chart in Appendix C.
- Relevant/Reliable: Work done each week is between five to six hours as shown by burn-down chart in Appendix C.
- Time-bound: Submission and documentation is done by the deadline April 27th, 2020.

Appendices

1. Appendix A

COVENTRY UNIVERSITY

5011CEM BIG DATA PROGRAMMING PROJECT
SPECIFICATION DOCUMENT

SPEED COMPARISON OF DATA DENSITY BASED CLUSTERING (DDC) ALGORITHM USING MATLAB EXTERNAL LANGUAGE INTERFACES

Document Version 2

April,2020.

Table of Contents

Table of Figures

7. Introduction

The four V's of Big Data- Volume, Variety, Veracity, and, Velocity. Keeping in mind these principles, clustering algorithms are considering to be adept for extracting data from large and complex datasets. Consider, velocity, i.e. the speed of data analysis. This project uses a clustering algorithm to compare speed of data analysis, on a specific dataset, in various languages.

8. Project Requirements

The project addresses the need for fast and accurate execution of data analysis. The DDC algorithm is executed in various languages on the dataset of ozone emissions in Europe, as measured by the European Centre for Medium Range-Weather Forecasts (ECMWF). The project makes use of Matlab External Language Interfaces, to compare the DDC mathematical formula of data analysis in Matlab, Python, C++, and Java. DDC was opted for its computational abilities, and identification of clusters on the grid-based data which is provided. The DDC algorithm inputs in a 4-D array of scaled data (dimensions mxn), with m rows of sample data of ozone emissions given by grid spaces of 0.1-degree latitude and longitude. It outputs a structure of cluster centre co-ordinates and radii, and an array (49 million x 2 dimensions) of data points with the corresponding cluster number.

SMART Analysis

- Specific- Do speed comparison of DDC algorithm,
- Measurable- by measuring execution time in four languages (Matlab, Python, C++, Java) ,
- Achievable- spending about sixty hours,
- Relevant/Reliable- and working six hours a week,
- Time-bound- to finish by the deadline April 27th,2020.

8.1 Related documents

Table 2: Table of related documents

Name (with link to the document)
Hyde, R. and Angelov, P. (2014) *Data Density Based Clustering.* http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6930157&isnumber=6930143
Reddy, K.S.S. and Bindu, C.S. (2017) *A Review on Density-Based Clustering Algorithms for Big Data Analysis.* https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8058322
Bendechache, M., Kechadi, M.-T., and Le-Khac, N.-A. (2016) *Efficient Large-Scale Clustering Based on Data Partitioning.* 612-621 https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7796948
Mai, S.T., He, X., Feng, J., Plant, C., and Böhm, C. (2014) *Anytime Density-Based Clustering of Complex Data.* https://dl.acm.org/doi/10.1007/s10115-014-0797-0
Mai, S.T., Assent, I., and Storgaard, M. (2016) *AnyDBC: An Efficient Anytime Density-Based Clustering Algorithm for Very Large Complex Datasets.* https://dl.acm.org/doi/10.1145/2939672.2939750
Hong, L. and Cai, J. (2010) *The Application Guide of Mixed Programming between MATLAB and Other Programming Languages.* https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5452058
Brandt, S. (1998) *Data Analysis.* New York [u. a.]: Springer http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.177.5019&rep=rep1&type=pdf

8.2 Terms/Acronyms and Definitions

Table 3: Table of Acronyms

Term/Acronym	Definition	Description
DDC	Data Density Based Clustering Algorithm	This algorithm uses density of data to identify clusters.
ECMWF	European Centre for Medium-Range Weather Forecasts.	Research center which provides global weather predictions.
SSD	Solid State Drive	A type of mass storage device like hard drive.
OS	Operating System	Software that communicates with computer hardware and runs programs.
API	Application Programming Interface	Software mediator that allows communication between two applications.
JDK	Java Development Kit	Software development environment for Java applications.
CPD	Continuous Professional Development	Tracking and documenting skills gained.

9. Risks and Assumptions

Hardware Specifications: The project ran on a system with an Intel® Core™ 17-7500U CPU @ 2.70Ghz 2.90GHz with an 8.00 GB RAM and a 128GB capacity Solid State Drive (SSD) and 1TB Hard Drive. It is assumed that a system with the same of higher hardware specifications will be adequate.

Software Specifications: The operating system considered here is Windows 10. The assumption that programs would run on different OS will pose a risk due to discrepancies in library dependencies and compiling and execution of programs. The program call is only done synchronously in each application, asynchronous connections to Matlab have not been explored. Programs were evaluated with no background processes running but system interrupts and hidden background processes are not taken into consideration.

Algorithm: The DDC algorithm used, was provided at the start of the module and any inconsistencies have not been considered. The speed comparison is done on the data without a merging clusters and visualizing the output. The measure of execution is not evaluated for the above and any results found are assumed for a DDC function without merging , a single radius and no visualization.

User Knowledge: It is assumed that the user has prolific knowledge of Matlab, Python, C++, and Java syntax. The user should also have the ability to understand documentation to implement the various Matlab External Language Interface API's.

Data analysed and size: The data analysed is provided in a mat data file and a different format would need change of importing data in Matlab. The Matlab External Language Interface API's are limited by a 2GB array size and recursive functions cannot be called between applications.

10.Out of Scope

- OS- Windows. Linux and Mac OS not in scope of project.
- DDC algorithm is called using Matlab External Language Interface API's. The rewriting of the mathematical formula in each language is not in scope.
- Asynchronous connection to Matlab is not explored.
- Matlab External Language Interface API's limits the size of data arrays passed to 2GB.
- Recursive data structures cannot be passed to Python. Large output arrays cannot be handled by Python.
- Mex files for C++ is not considered.
- Visualization of clusters is not considered.
- Merging of cluster is not applicable.
- Connection to Matlab on remote machines.
- Connection to more than one shared Matlab session is also not in scope.
- Although programs were executed without other background processes running, there might still be discrepancies in speed due to system interrupts or hidden processes as execution in an isolated environment to remove interferences is not in scope.

11.System/ Solution Overview

Matlab- Version R2019b. Import data provided in mat file. Run and time DDC algorithm.

Python- Version 3.7. Connection to Matlab shared session with Matlab python engine. Run and time DDC algorithm.

C++- Visual Studio 2019. Connection to Matlab shared session with in-built C++ engine in Matlab. Run and time DDC algorithm.

Java- NetBeans IDE 8.2. JDK 8. Connection to Matlab shared session with java engine. Run and time DDC algorithm.

All applications execute and show cluster and its results on their output screen (python uses nargout=0 to manage the output). The best speed of execution is taken out of numerous simulations.

11.1 Context Diagram/ Interface Diagram/ Data Flow Diagram, Application Screen Flow, Sitemap, Process Flow

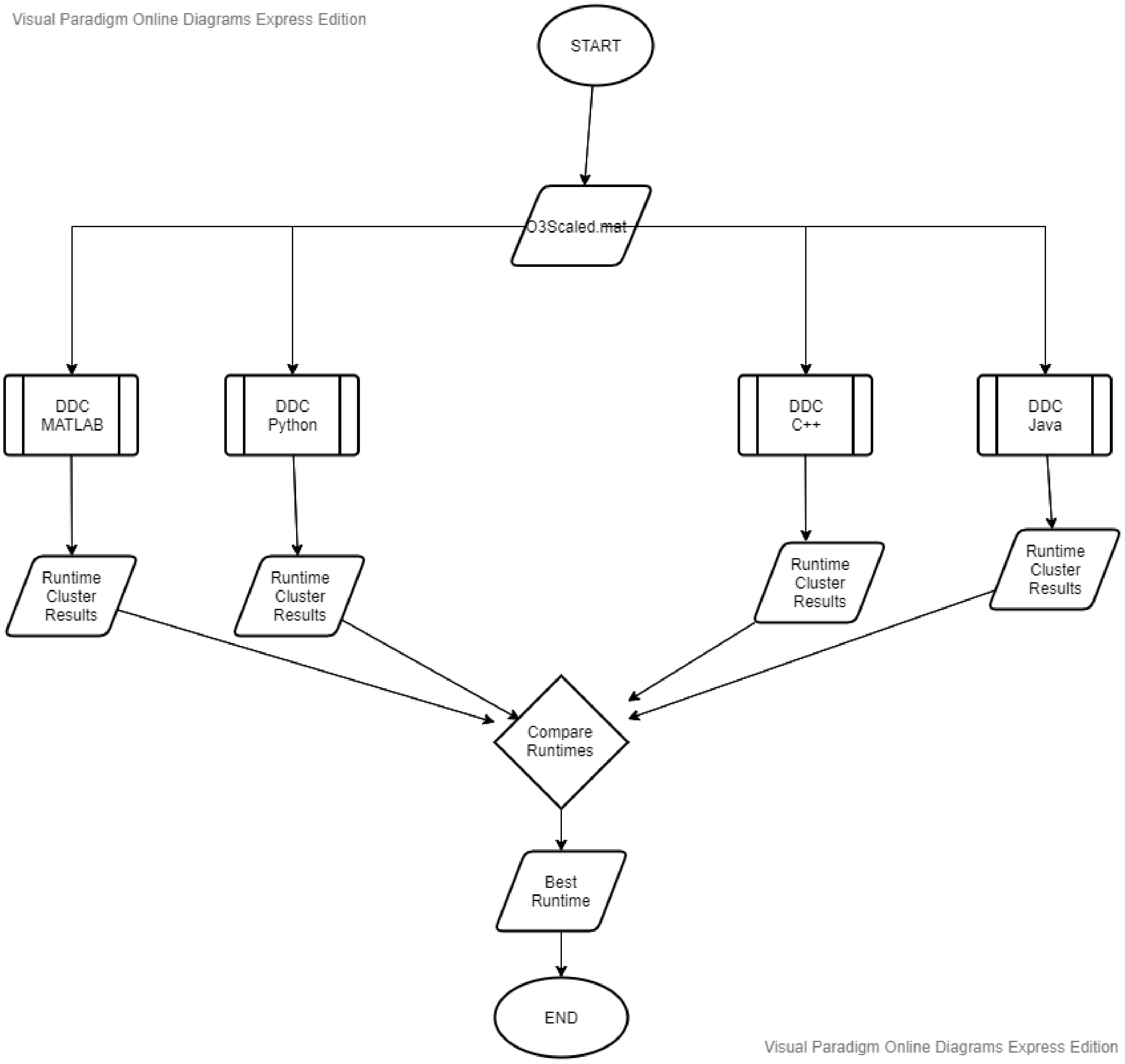


Figure 1: Project Flow Chart

12.Project Management

Project Management is implemented with weekly sprints documented in a logbook. A customized Kanban board is used to keep track of task breakdown. Burn-down chart is updated after each weekly sprint to manage delivery of project. Continuous Professional Development (CPD) is tracked with the help of logbook. Any changes made for weekly sprint are denoted by file versions numbered by the sprint running currently. Testing is manged with sample data and a uniform test file. For further details, refer to Appendix A (Portfolio)

- Week 1- Introduction to Project.
- Week 2- Matlab Course/ Specification
- Week 3- Matlab Course/ Code Planning
- Week 4- Data files manipulation.
- Week 5- DDC Algorithm Matlab
- Week 6- DDC Algorithm Python
- Week 7- DDC Algorithm C++
- Week 8- DDC Algorithm Python and C++
- Week 9- DDC Algorithm Java
- Week 10- Report/ Simulations
- Week 11- Portfolio/ C++ Simulation
- Week 12- Viva/ Final simulation

7. Open Issues

Table 4: Issues

Issue	Raised On	Solution/ Decision	Resolved By	Resolved On	Status
C++ Executable takes about two hours to reach second cluster in results.	Week 10 of project timeline.	System hardware, Visual Studio dependency, low processor speed.	Next Deployment	-	Executable runs and outputs result but current system is not viable for computing execution time.

8. References

Bachnak, R.A. and Lee, R. (2002) *The MATLAB Compiler Suite: M-Files to C/C++ Executable Programs.* [online] 151–158. available from <https://pennstate.pure.elsevier.com/en/publications/the-matlab-compiler-suite-m-files-to-cc-executable-programs> [27 April 2020]

Bendechache, M., Kechadi, M.-T., and Le-Khac, N.-A. (2016) *Efficient Large Scale Clustering Based on Data Partitioning*. held October 2016. 612–621

Brandt, S. (1998) *Data Analysis*. New York [u. a.]: Springer

Hong, L. and Cai, J. (2010) *The Application Guide of Mixed Programming between MATLAB and Other Programming Languages.*

Hyde, R. and Angelov, P. (2014) *Data Density Based Clustering*. held 2014. 1–7

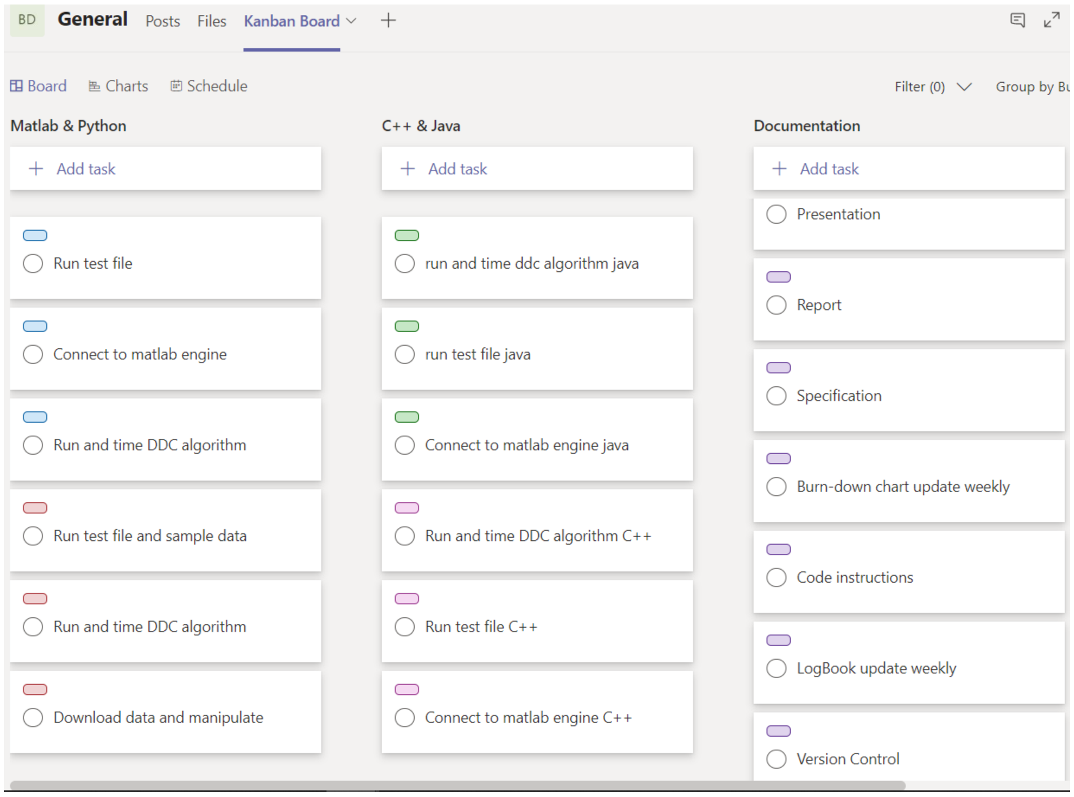
Mai, S.T., Assent, I., and Storgaard, M. (2016) *AnyDBC: An Efficient Anytime Density-Based Clustering Algorithm for Very Large Complex Datasets*. held 2016

Mai, S.T., He, X., Feng, J., Plant, C., and Böhm, C. (2014) *Anytime Density-Based Clustering of Complex Data.*

Mathworks External Language Interfaces (n.d.) available from <Matlab External Language Interface API's> [2020]

Reddy, K.S.S. and Bindu, C.S. (2017) *A Review on Density-Based Clustering Algorithms for Big Data Analysis.*

2. Appendix B

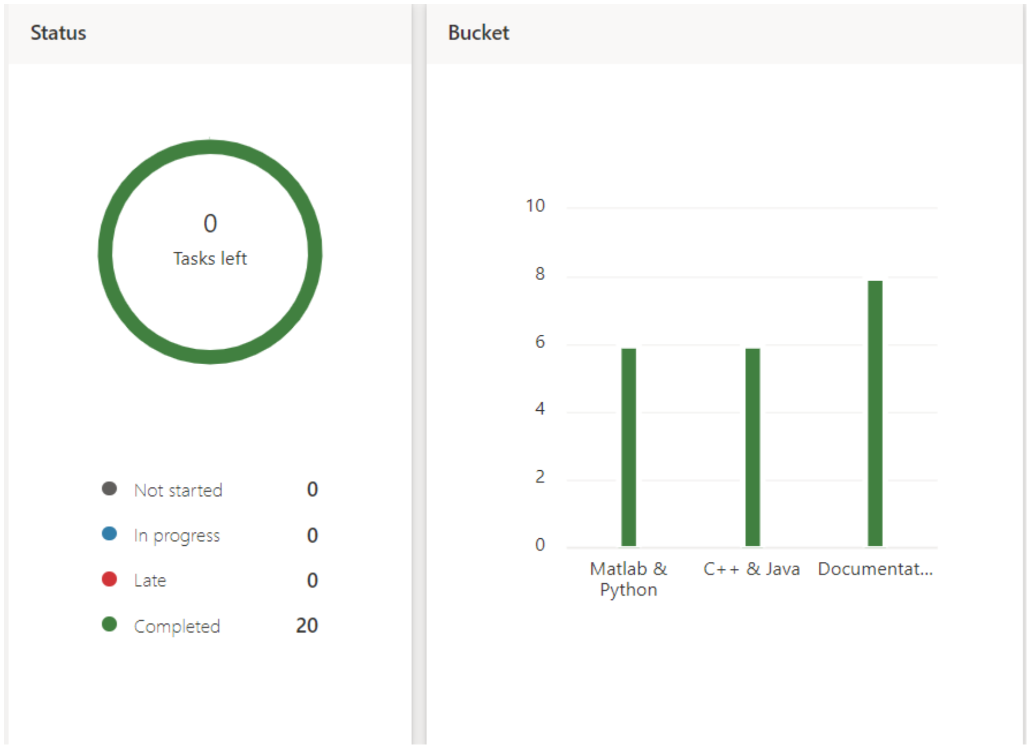


BD **General** Posts Files Kanban Board

Board Charts Schedule Filter (0) Group by Bu

Matlab & Python	C++ & Java	Documentation
Add task	Add task	Add task
Run test file	run and time ddc algorithm java	Presentation
Connect to matlab engine	run test file java	Report
Run and time DDC algorithm	Connect to matlab engine java	Specification
Run test file and sample data	Run and time DDC algorithm C++	Burn-down chart update weekly
Run and time DDC algorithm	Run test file C++	Code instructions
Download data and manipulate	Connect to matlab engine C++	LogBook update weekly
		Version Control

Figure 2: Kanban Board Tasks



● Not started	0
● In progress	0
● Late	0
● Completed	20

Figure 3: Kanban Board Analysis

3. Appendix C

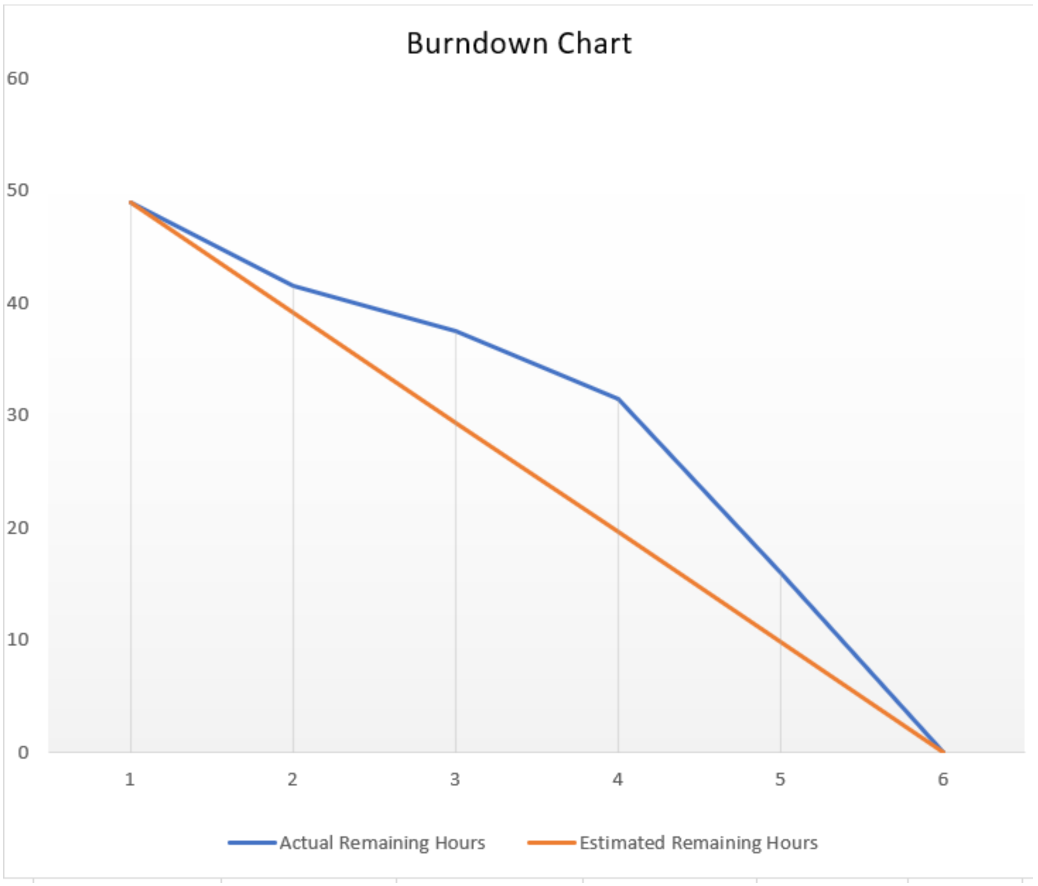
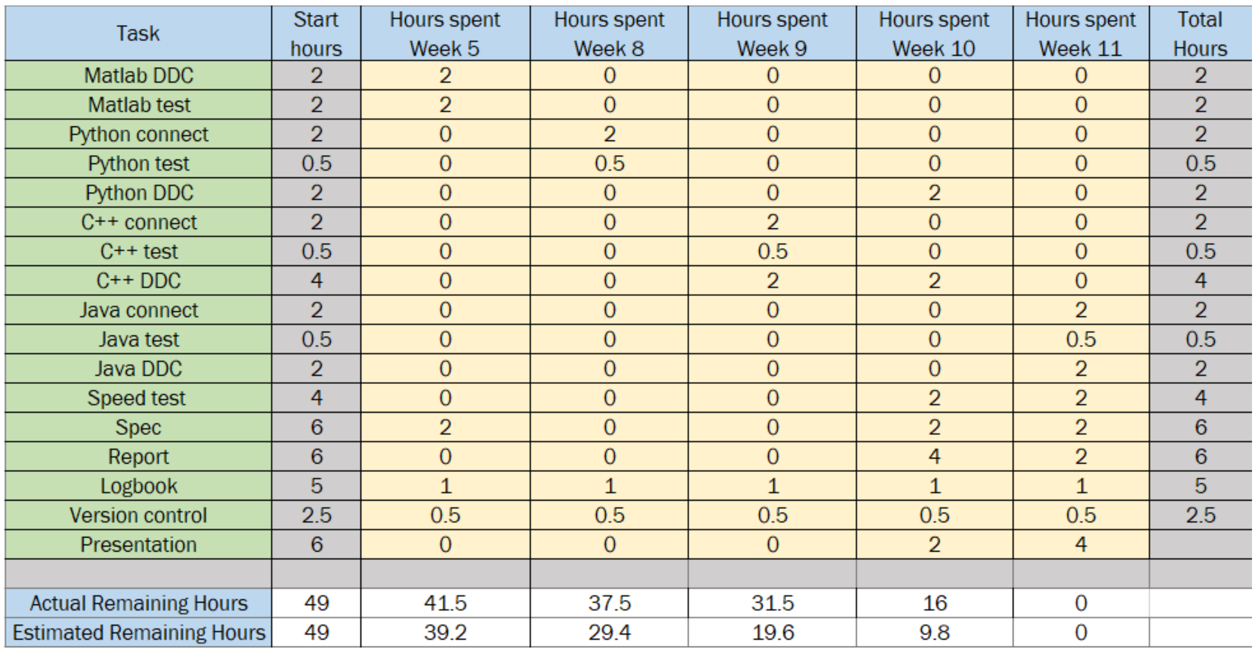


Figure 4: Burndown Chart



Task	Start hours	Hours spent Week 5	Hours spent Week 8	Hours spent Week 9	Hours spent Week 10	Hours spent Week 11	Total Hours
Matlab DDC	2	2	0	0	0	0	2
Matlab test	2	2	0	0	0	0	2
Python connect	2	0	2	0	0	0	2
Python test	0.5	0	0.5	0	0	0	0.5
Python DDC	2	0	0	0	2	0	2
C++ connect	2	0	0	2	0	0	2
C++ test	0.5	0	0	0.5	0	0	0.5
C++ DDC	4	0	0	2	2	0	4
Java connect	2	0	0	0	0	2	2
Java test	0.5	0	0	0	0	0.5	0.5
Java DDC	2	0	0	0	0	2	2
Speed test	4	0	0	0	2	2	4
Spec	6	2	0	0	2	2	6
Report	6	0	0	0	4	2	6
Logbook	5	1	1	1	1	1	5
Version control	2.5	0.5	0.5	0.5	0.5	0.5	2.5
Presentation	6	0	0	0	2	4	
Actual Remaining Hours	49	41.5	37.5	31.5	16	0	
Estimated Remaining Hours	49	39.2	29.4	19.6	9.8	0	

Figure 5: Burndown Chart Calculation