New Simulation Methods to More Effectively Integrate High Levels of Renewable Energy Resources

DESIGN DOCUMENT

Team Number: 21

Client: MISO

Adviser: James McCalley

Team Members: Jared Rickard, Jeremy Nash, Nidhi Rawell, Collins Ntwali

Team Email: sdmay20-21@iastate.edu

Team Website: http://sdmay20-21.sd.ece.iastate.edu/

Revised: 12/07/2019

Executive Summary

Development Standards & Practices Used

We will be using the PEP 8 style guide for all python code.

Summary of Requirements

Phase 1:

- Develop a working method to effectively, efficiently, and accurately compress the 8760 -hour data profile

- Validate the developed method and demonstrate its effectiveness from a statistical perspective

Phase 2:

- Simulate in actual production cost database using industry applications and real-world system data

- Automate the profile reduction process for use in industry applications

Applicable Courses from Iowa State University Curriculum

EE 458

EE 285

New Skills/Knowledge acquired that was not taught in courses

Experience in coding with the Python Language

Contents

List of Tables

1. Introduction

1.1. Acknowledgement

We would like to thank James Okullo, Armando Figueroa-Acevedo, Yifan Li, David Severson, and Ryan Hay from the MISO team for their experience and insights.

We would also like to thank Dr. James McCalley for his helpful suggestions during our discussions.

1.2. Problem and Project Statement

MISO runs thousands of production cost simulations (year-long hour-by-hour simulations of the electric and economic performance of the MISO grid) every planning study cycle to investigate a host of topics including the efficacy of proposed transmission upgrades, the impact of federal policy, and the complexity of integrating large amounts of renewables to the system. These simulations, which model the entire Eastern Interconnect of the US power system, take large amounts of processing time due to high model dimension with a large number of load and generation profiles. The projected high growth of renewable penetration in the MISO footprint, and the resulting increase in modeling data, will only exacerbate the situation.

To more efficiently and effectively integrate more renewable energy onto the grid, new modeling techniques are needed. The goal of this study is to, therefore, research and implement various methods to appropriately reduce the fidelity of the data profiles while maintaining an adequate amount of the key production cost information. The study will validate the methods' reliability and quantify the effects that profile approximation has on simulation runtime and results. It is expected that the properly designed profile reduction method would make the normal 8760 hours production cost simulation more efficient. The increased efficiency has the potential to enable us to explore more ways to improve our current planning study processes by introducing cutting edge research in academia and industry.

1.3. Operation Environment

The final product of this project will be used by MISO engineers and modelers to assist in running simulations

1.4. Requirements

Our most important requirement is to develop an instruction set of mathematical steps to find approximated data from a full data set. This instruction set will have multiple user determined inputs. The next requirement is to wrap this instruction set into an easy to use Python script. Then we have to make sure that the output of this instruction set actually runs through PCM simulations faster than un-approximated data without losing result fidelity.

Another requirement of this project is to test this instruction set with as many possible combinations of inputs to determine their effect on the outcome. Then we will determine which input set is ideal for finding accurate representations of the data being approximated. Finally, as a requirement of this project we will present this research to our client.

1.5. Intended Users and Uses

Our intended user is a transmission engineer or modeler who plans to run production cost modeling simulations and needs the simulation to produce results faster without losing too much accuracy.

1.6. Assumptions and Limitations

Assumption	*Reason*
Input CSVs will have the format: - Row 1 is CPnode ID - Col 1 is sequential time stamps - Cells will state a MW value for that CPnode at that time stamp. The value will be relevant to the data being represented by the CSV	This is the only format that we have been given so far
Python will be the only coding language used in this project	MISO engineers use Python more than other data analysis languages, such as R, and are very familiar with the libraries in Python.
Limitation	*Reason*
There are currently no options for any other input CSV formats	We have not been given any other formats to design for.
Currently the code only uses K-means as a clustering algorithm	K-means was the easiest to implement so that we could quickly get a prototype to start data analysis.
Currently the code only creates attributes about the data set based on information inside the data set	We have only just begun discussing ways to implement spatial data. We also haven't been given any PCM output data, so we don't know how to design attribute creation based on that.

Table 1.1 Assumptions and Limitations

1.7. Expected End Product and Deliverables

Phase 1: (Fall 2019)
Develop a Working method to effectively, efficiently, and accurately find representative subsets of an 8760-hour data profile.
Validate the developed method and demonstrate its effectiveness from a statistical perspective
Phase 2: (Spring 2020)
Simulate our representative subsets using industry software tools and compare the results with application output from original data
Automate the profile approximation process for use in industry applications
Phase 3: (Summer 2020 and beyond)
Beyond the ISU capstone project, MISO will independently validate the proposed methods
Deliverables
Weekly to bi-weekly updates on the status of our progress and research
Phase 1 Report (end of Fall Semester)
Conference paper (end of Spring Semester)
Working Python script to approximate any data set given

Table 1.2 Expected End Product and Deliverables

2. Specification and Analysis

2.1. Proposed Design

Our proposed instruction set follows four steps:

- Divide the input data into subsets

- Calculate attributes related to each subset
- Find ideal, imaginary attributes, clustered around what was calculated in the previous step
- Find the subsets whose attributes are closest to these ideal attributes. These are now ideal subsets

These instructions leave open many options such as size of the subsets, what attributes to calculate, how many ideal, imaginary attribute sets to cluster around, and what method to cluster the data. All of these will be choices left to the user.

After finalizing these steps, we wrote code to implement each of these steps into one large script. The code starts by reading in all the CSVs related to a PCM profile. Next, the code breaks up each CSV into subsets and calculates attributes based on the data inside these subsets. The calculated attributes are stored in a 2D matrix of data where each column is an attribute of a single node's data, and each row is a subset of the full data. Then the code feeds this matrix into a clustering algorithm, currently K-means, to find the ideal attributes. Because we can't rebuild a full data subset from these ideal attributes, we instead opt to find the subsets with attributes closest to these ideal ones. We then call these subsets the ideal subsets. Finally, the code writes all these ideal subsets into CSVs with the same format as the original CSVs, along with a list of which subsets are approximated by the ideal subsets.

2.2. Design Analysis

We have discussed different possible attributes of the data sets that we can base our code around. We have written code to perform all the functions outlined in the first paragraph of the proposed design section above.

So far, we have designed the code so we could easily change the number of attributes, size of data subset (hours, days, weeks, etc.), and number of clusters (find 1 representative day, 2, 20 representative days, etc). We have run tests on how changing each of these three variables affects the code's runtime and its output. This is all to find out which combinations of the inputs is ideal for finding accurate representations of the data being approximated. Our project strength is being able to easily modify our code to better represent the data based on our rigorous tests.

We have discussed many different attributes of the 8760 data profile that we can base our code around. We looked at each of the attributes that our code uses to cluster and modified them so one attribute does not over power the others. In order to have our code output the correct representative weeks, further extensive testing will need to be done on our chosen attributes and other attributes will likely need to replace ones that aren't accurately describing the data. For example, solar will always be a value of 0 at night, so minimum daily value doesn't accurately describe solar power. A weakness of this project is that until we run the output data profile in Plexos and compare that output with the known un-approximated output, we won't know if our representative data is "good" or "bad."

2.3. Development Process

We currently develop using a lightweight model that starts by created a tool with basic functionality, testing it, and iteratively adding more functionality while testing to make sure nothing breaks in the process.

2.4. Design Plan

Our first step is to develop a tool for calculating different aspects of the data we determine to be relevant to our testing. We will determine their relevancy by asking MISO engineers, running our own tests, and evaluating the results with our expected results. Next, we will write software to determine representative data points based on these aspects. This software will be in Python language. Finally, we will input this representative approximation into the PCM simulation software and compare the output with previous test results. Our process will be to refine which aspects are chosen to create the approximations and our final product will incorporate all these different steps.

3. Statement of Work

3.1. Previous Work and Literature

Background Information: Literature we've reviewed was lectures and notes about Production Cost Modelling

Previous Work: MISO has done previous work into this topic by looking at approximating hourly data by looking at the generation or load value of a profile at that time. Our work will expand on this work by trying approximated daily, weekly, and monthly data with many different aspects of the data being considered. Some previous studies that were looked at on this subject included Planning for the renewable future: Long-term Modelling and tools Expand Variable Renewable Power in Emerging Economies by International Renewable Energy Agency (IRENA), Temporal and Spatial tradeoffs in Power System Modeling with Assumption about Storage: An Application of the Power Model by Bethany A. Frew and Mark Z. Jacobson, and Hierarchical Clustering to Find Representative Operating Periods for Capacity-Expansion Modeling by Yixian Liu, Ramteen Sioshansi, Senior Member, IEEE, and Antonio J. Conejo, Fellow, IEEE.

3.2. Technology Considerations

For the approximation tool we considered writing it in either R or Python. The two languages are very similar and offer about equal amounts of relevant functions in the form of libraries. Ultimately, we chose to write our tool in Python as more MISO employees are familiar in Python and would benefit from the ability to more easily read the code.

3.3. Task Decomposition

Code design:

- Write a function that takes in a load/generation profile, computes relevant aspects of the data based on the time interval being considered, and outputs the data in a table.
    - Write function to load profile CSVs and store relevant data
    - Write function to calculate attributes from a subset of data
    - Write function to loop through full data set, calculating given attributes and storing them in an organized table
- Write a function that takes in a table of data with each column being a different aspect and a number of clusters, compute a number of representative clusters, and output this data in a form fit for input to our PCM tool.
    - Write function that puts attributes table into K-means and outputs ideal attributes
    - Write function that finds the data subsets closest to the ideal attributes

    - Write function that formats these ideal subsets with the correct surrounding information
    - Write function that writes each of the ideal subsets with surrounding info into individual CSVs.
- Combine everything into one self-contained script
    - Comment code
    - Make main as simple and modifiable as possible
- Expand script options
    - Add option to normalize data before clustering
    - Write more possible attribute calculations
    - Write additional clustering algorithms besides K-means and make them options for the users
    - Add attributes specific to spatial data
- Develop UI for this tool that incorporates all the different steps of this process and gives the user options for how they want to approximate data.
    - Create function that generates window with text boxes for input and output CSV locations, text box for subset size and cluster count, check marks for which attribute calculations the user wants, and drop-down menu for which clustering algorithm the user wants.
    - Input all this user defined data and feed it into the self-contained script.

Code Analysis:

- Determine the aspects that are important in analyzing the data.
    - Run tests that use every combination of attributes that we have
    - Use the output of these tests to determine which attributes overpower others
    - Scale/add/subtract attributes, test again
- Run the PCM simulation with the approximated data and compare it to un-approximated data. Searching for similarities and differences that could refine the next approximation.

Research:

- Research and brainstorm to determine different aspects of the load/generation data that could be used to find approximations of profiles.
    - Note the differences between datasets - Solar tends to follow a daily pattern; wind is more sporadic.
- Research different clustering methods, or different approaches to grouping similarities among data points based on various attributes.
    - Compare various approaches to the approach taken in the project
    - Develop a criterion based on the weaknesses and strengths of various approaches
- Research different studies related to the project

Plexos related tasks:

- Acquire two virtual machines from Iowa State
- Download and install Plexos on VMs
- Run un-approximated data to generate a benchmark set of output
- Run as many different sets of approximated data that can be created

- Compare approximated output to benchmark output and determine closeness

3.4. Possible Risks and Risk Management

Some major areas that could hinder progress are our developing knowledge of Python and PCM software like Plexos. Downloading Plexos has proven to be very difficult and time consuming, as the company which owns Plexos has much more important things to do than give software to college students. Our knowledge of power grid data attributes will be another roadblock that we will have to overcome. Our knowledge overall is good, but we will definitely ask the MISO engineers for more advice.

3.5. Project Proposed Milestones and Evaluation Criteria

Milestone	*Criteria*
Completed function to compute aspects of data profiles	Evaluated by reviewing code and possibly hand computing test samples
Complete function that clusters data	Evaluated by reviewing code and trying simple, one-dimensional sample clusters
Run Multiple PCM simulations with different approximated data sets	Evaluated by comparing to results of un-approximated data set
Complete UI	Evaluated by reviewing code and handing it off to MISO engineers to try and figure out

Table 3.1 Project Proposed Milestones and Evaluation Criteria

3.6. Project Tracking Procedures

We currently have an excel sheet of all tasks to complete and who is most responsible for completing them. Considering updating to GitLab.

3.7. Expected Results and Validation

Our desired outcome is that our tool will approximate data quickly and this data will produce simulation results equivalent to the original data in less time.

4. Project Timeline, Estimated Resources, and Challenges

4.1. Project Timeline

Task	**Expected Finish**
Code Design	
Write a function that takes in a load/generation profile, computes relevant aspects of the data based on the time interval being considered, and outputs the data in a table	Sep. 27, 2019
Write function to load profile CSVs and store relevant data	Sep. 23, 2019
Write function to calculate attributes from a subset of data	Sep. 25, 2019
Write function to loop through full data set, calculating given attributes and storing them in an organized table	Sep. 30, 2019
Write a function that takes in a table of data with each column being a different aspect and a number of clusters, compute a number of representative clusters, and output this data in a form fit for input to our PCM tool	Oct. 27, 2019
Write function that puts attributes table into K-means and outputs ideal attributes	Oct. 15, 2019
Write function that finds the data subsets closest to the ideal attributes	Oct. 19, 2019

Write function that formats these ideal subsets with the correct surrounding information	Oct. 23, 2019
Write function that writes each of the ideal subsets with surrounding info into individual CSVs.	Oct. 27, 2019
Combine everything into one self-contained script	Nov. 21, 2019
Comment code	Undetermined
Make main as simple and modifiable as possible	Nov. 21, 2019
Expand script options	Mar. 14, 2020
Add option to normalize data before clustering	Dec. 1, 2019
Write more possible attribute calculations	Jan. 30, 2020
Write additional clustering algorithms besides K-means and make them options for the users	Mar. 14, 2020
Add attributes specific to spatial data	Feb. 14, 2020
Develop UI for this tool that incorporates all the different steps of this process and gives the user options for how they want to approximate data.	Apr. 28, 2020
Create function that generates window with text boxes for input and output CSV locations, text box for subset size and cluster count, check marks for which attribute calculations the user wants, and drop-down menu for which clustering algorithm the user wants.	Apr. 14, 2020
Input all this user defined data and feed it into the self-contained script.	Apr. 28, 2020
Code Analysis	
Determine runtime for different numbers of attributes, clusters, and different subset sizes	Jan. 17, 2020
See how attributes relate to outputs, correct for overpowered attributes	Jan. 31, 2020
Add or subtract attributes based on Plexos outputs	Feb. 28, 2020
Research:	
Research and brainstorm to determine different aspects of the load/generation data that could be used to find approximations of profiles.	Undetermined
Note the differences between datasets - Solar tends to follow a daily pattern; wind is more sporadic.	Undetermined
Research different clustering methods, or different approaches to grouping similarities among data points based on various attributes.	Feb. 5, 2020
Compare various approaches to the approach taken in the project	Mar. 9, 2020
Develop a criterion based on the weaknesses and strengths of various approaches	Mar. 19,2020
Research different studies related to the project	Dec. 6, 2020
Plexos Related Tasks:	
Acquire two virtual machines from Iowa State	Dec. 4 , 2020
Download and install Plexos on VMs	Jan. 18, 2020

Run un-approximated data to generate a benchmark set of output	Jan. 26, 2020
Run as many different sets of approximated data that can be created	Mar. 25, 2020
Compare approximated output to benchmark output and determine closeness	Apr. 28, 2020

Table 4.1 Project Timeline

4.2. Feasibility Assessment

This project will be a test of many different aspects of data to find the best approximation of that data. This project will result in a tool that can approximate any data sets based on the most successful aspects but will allow the user to develop and test their own aspects.

4.3. Personnel Effort Requirements

Task	Time Requirement (group hours)	Reasoning
Code Design	150	Heavily dependent on developing group comprehension of Python and data manipulation. Requires brainstorming of attribute list, clustering implementations, and data formatting.
Code Analysis	80	Heavily dependent on developing group comprehension of Python and data manipulation. Brainstorming what aspects of the code we want to compare and how to interpret those results
Research	100	Dependent on searches for tangentially related topics, comprehending and correlating results to our methods and processes
Plexos Related Tasks	170	Dependent on how many simulations we decide to perform and the simulation runtime.

Table 4.2 Personal Effort Requirements

4.4. Other Resource Requirements

We need access to all the excel datasets that MISO inputs to Plexos so that we can approximate them. The only other resources needed are workstations (Python IDE) to develop code, copies or licenses of the PCM software Plexos, and virtual machines to run the PCM simulation tools.

4.5. Financial Requirements

Our project is mostly coding and research based, therefore no financial requirements are necessary.

5. Testing and Implementation

Testing will be needed for runtime and accuracy of the production cost model estimate. This will require running the Plexos program with the original data and then a second time with our modified inputs.

We will be looking at total runtime and the percent difference between the estimate's output and the normal output. Our runtime will be significantly shorter than the normal method, testing will show us if the time saved is satisfactory enough for MISO.

We will input many different test cases into the program to see how it reacts to the different data profiles.

There will also be testing for usability and universal understanding of the code.

Once testing has begun, this section will be updated to reflect our findings

5.1. Interface Specifications

We will not be developing any interfacing techniques in this project. Our tool will interpret data profiles and output approximations in the same format.

5.2. Hardware and Software

Software used in the test will include Python and a Python compiling software to be determined later, Plexos, and Excel. The code will be written in Python as this a widely used language and will be the easiest for MISO employees to learn. Moreover, Python is industry standard, and has a lot of accessible documentation online. Plexos will be used to run the production cost models with the new inputted data from our Python code. Excel is used for all data inputs. Each software is essential for our successful implementation of this new method.

5.3. Functional Testing

Functional testing will include but is not limited to total computation time and accuracy of final estimate. The goal for the computation time is one business day (less than 8 hours with the goal being around 2 hours total). The final estimate needs to be accurate enough to provide the user with a reliable forecast before they run the non-estimated model.

5.4. Non-Functional Testing

We will also test for readability and usability. The code will be used by many different MISO employees and will need to be commented, clear, and concise. Once the code is developed, we will shorten it to the best of our ability.

5.5. Process

Testing has been started on how different numbers of attributes, clusters, and different sized subsets affect the output of the python code as well as its runtime. Further, more extensive testing will take place once we have the Plexos software. The testing process follows like: make a unique main to test different aspects of the clustering program, output the results of these tests, modify the clustering program to reflect these results, test again.

5.6. Results

Results from previous tests suggest a linear relation between adding attributes and runtime, decreasing subset size and runtime, and adding clusters and runtime. Further tests will determine which attributes we'll use for the final code and which subset size and number of clusters has the best runtime to accuracy ratio.

6. Closing Material

6.1. Conclusion

The work we have done so far includes but is not limited to: started the process to acquire Plexos, developed a list of attributes to test in the upcoming weeks, completed the onboarding process to fully understand the task at hand, and developed a plan for the rest of this semester as well as team roles and tasks for each member. Also, completed filling out license forms in the process of acquiring Plexos. The process of obtaining VMs has been completed. Research on various studies or pervious work was done, and research on various clustering methods was completed. The best way to complete our goals on time will be to stick to the schedule, adapt as needed, and seek advice from MISO and our advisor.

6.2. References

References will be made once all code production and testing has been finished.

6.3. Appendices

In future document versions, we will post the python files used as well as flowcharts that describe the process used in the code. Plexos files won’t be useful because only a select few have the Plexos software.